Empirical Software Engineering manuscript No.
(will be inserted by the editor)

Usage and Attribution of Stack Overflow Code Snippets in GitHub Projects

Sebastian Baltes · Stephan Diehl

Received: 19 Januar 2018 / Accepted: 14 August 2018

Abstract Stack Overflow (SO) is the most popular question-and-answer website for software developers, providing a large amount of copyable code snippets. Using those snippets raises maintenance and legal issues. SO's license (CC BY-SA 3.0) requires attribution, i.e., referencing the original question or answer, and requires derived work to adopt a compatible license. While there is a heated debate on SO's license model for code snippets and the required attribution, little is known about the extent to which snippets are copied from SO without proper attribution. We present results of a large-scale empirical study analyzing the usage and attribution of non-trivial Java code snippets from SO answers in public GitHub (GH) projects. We followed three different approaches to triangulate an estimate for the ratio of unattributed usages and conducted two online surveys with software developers to complement our results. For the different sets of projects that we analyzed, the ratio of projects containing files with a reference to SO varied between 3.3% and 11.9%. We found that at most 1.8% of all analyzed repositories containing code from SO used the code in a way compatible with CC BY-SA 3.0. Moreover, we estimate that at most a quarter of the copied code snippets from SO are attributed as required. Of the surveyed developers, almost one half admitted copying code from SO without attribution and about two thirds were not aware of the license of SO code snippets and its implications.

Keywords code snippets · licensing · stack overflow · github · online survey · mining software repositories

Sebastian Baltes
University of Trier, Germany
Orcid: 0000-0002-2442-7522
E-mail: research@sbaltes.com

Stephan Diehl
University of Trier, Germany
Orcid: 0000-0002-4287-7447
E-mail: diehl@uni-trier.de

1 Introduction

Stack Overflow (SO) is the most popular question-and-answer website for software developers. As of December 2017, its public data dump (Stack Exchange Inc, 2017a) lists over 13 million answered questions and over 8 million registered users. Many answers contain code snippets together with explanations (Yang et al, 2016). The availability of this large amount of code snippets lead to changes in software developers' behavior: Nowadays, they regularly face the "build or borrow" question (Brandt et al, 2010): Should they try to understand and solve an issue on their own or just copy and adapt a solution from SO? Assuming that developers also copy and paste snippets from SO without trying to thoroughly understand them, *maintenance issues* arise (Scalabrino et al, 2017). For instance, it may later be more difficult for developers to refactor or debug code that they did not write themselves. Moreover, if no link to the corresponding question or answer is added to the copied code, it is not possible to check the SO thread for a corrected or improved solution in case problems occur.

Beside possible maintainability implications, copying and pasting code from SO may also lead to *licensing issues*: All content on SO is currently licensed under the Creative Commons Attribution-ShareAlike 3.0 Unported license (CC BY-SA 3.0) (Creative Commons Corporation, 2007), which allows to share and adapt the published content, but requires *attribution* and demands contributions based on the content to be published under a compatible license (*share-alike*). Regarding the *attribution* requirement, SO terms of service (Stack Exchange Inc, 2018a) stated—until May 2018—which information is required when content from SO is republished. In particular, they required a link to the original post together with the names of the authors on SO (see Section 10).

The *share-alike* requirement of CC BY-SA 3.0 requires derived work to use a compatible license. It further requires adaptations of licensed content to add a credit identifying how the content is used. The license defines an adaptation as "a work based upon" the licensed content (Creative Commons Corporation, 2007), which "manifests sufficient new creativity to be copyrightable" (Creative Commons Corporation, 2017b). Regarding the licensing of such adaptations, CC BY-SA 3.0 restricts the way authors may distribute them, where distribution is defined as making the original work or an adaptation "available to the public". It is only allowed to publish adaptations under the following licenses:

1. CC BY-SA 3.0,
2. a later version of CC BY-SA 3.0 (i.e., CC BY-SA 4.0),
3. a ported version of CC BY-SA 3.0 (e.g., CC BY-SA 3.0 DE),
4. a Creative Commons compatible license.

However, Creative Commons (CC) licenses are typically not used for software (Vendome, 2015) and there is currently no non-CC license that is considered share-alike compatible to CC BY-SA 3.0 (Creative Commons Corpo-

ration, 2017a). CC even recommends not to use CC licenses for software (Creative Commons Corporation, 2017b), because, "unlike software-specific licenses, CC licenses do not contain specific terms about the distribution of source code, which is often important to ensuring the free reuse and modifiability of software". They further state that "it would be difficult to integrate CC-licensed work with other free software".

The situation is even more complicated, because code on Stack Overflow may have been copied from a source that has either a more permissive or a more restrictive license than SO (*dual licensing*, see Section 12). If such an external source does not provide a license at all, the author of the code still has the exclusive copyright and CC BY-SA 3.0 is the only license that applies for the code (GitHub Inc, 2017a; Sojer and Henkel, 2011). This situation makes the usage of code snippets from Stack Overflow problematic in terms of possible licensing conflicts (see Sections 3 and 9).

In May 2018, SO changed their terms of service, among other reasons, in response to the new *European Union General Data Protection Regulation* (GDPR) (Tim Post, 2018). With that change, the attribution requirements mentioned above were silently removed from the terms of service (Stack Exchange Inc, 2018b). However, the requirements are still (as of July 20, 2018) mentioned and linked in the footer of the website, which is visible for each thread, and in the help page.[1] Moreover, the terms of service now refer to version 4.0 of the CC BY-SA license, but the data dump is still licensed under version 3.0 (see Section 9 for information about the compatibility of versions 3.0 and 4.0).

GitHub (GH) is one of the most popular code hosting platforms with more than 58 million repositories (as of September 2017) (Gousios, 2017). It is not only used by developers for their personal projects, but also by large companies such as Google, Microsoft, or Facebook. Since the source code of public GH projects is available online, copying and pasting code from SO posts into source code available on GH can be considered republication—the projects containing non-trivial code from Stack Overflow may even be considered adaptations of the copied code (see Section 3 for details on when code is copyrightable). Thus, the *attribution* and the *share-alike* requirements defined by CC BY-SA 3.0 apply. If developers copy non-trivial code snippets from SO into their GH projects and fail to comply with those requirements, the license is terminated, which means that using the code may constitute *copyright infringement* (St. Laurent, 2004; Creative Commons Corporation, 2017b). For closed source software projects, the *attribution* requirement does not apply (Creative Commons Corporation, 2017b). However, the *share-alike* requirement prevents using code from SO in closed source projects, it would only be allowed if the copied code is additionally licensed under a more permissive license.

To the best of our knowledge, there is currently no empirical evidence on how common it is to copy and paste non-trivial code snippets from SO into public GH projects without the required attribution (→ RQ1). It is also un-

[1] `https://stackoverflow.com/help/licensing`

clear how many of the projects using code from SO have a license conflict with Stack Overflow's license (→ RQ2). In the following, we present the research design and results of a large-scale analysis of the usage and attribution of Java code snippets from SO in public software projects hosted on GH. We both analyze attributed usages and utilize three different approaches to estimate the ratio of unattributed usages. To complement our results, we investigated if developers adhere to SO's attribution requirements (→ RQ3) and conducted two surveys with software developers on their attribution practice and their awareness regarding the licensing of code from SO posts (→ RQ4).

2 Research Design

The main goal of our research was to quantify the ratio of unattributed usages of code snippets from Stack Overflow in GitHub projects. By usage we mean copying (and possibly slightly adapting) a code snippet from a post on SO and pasting it into a public GH project. The following four research questions guided our research design:

RQ1: How often is code from Stack Overflow posts used in public GitHub projects without the required attribution?

RQ2: How often does the license of repositories containing code copied from Stack Overflow conflict with Stack Overflow's license?

RQ3: Do developers adhere to the attribution requirements defined in the Stack Overflow terms of service?

RQ4: Are software developers aware of the licensing of Stack Overflow code snippets and its implications?

We started our research with a preliminary survey to get first insights into developers's work practices regarding code snippets from SO (see Section 4). Our main research was then divided into three phases that focused on different files on GH, different code snippets from SO, and used different methods to triangulate an estimate for the ratio of unattributed usages (RQ1, see Sections 5, 6, and 7). For all three phases, we retrieved the licenses of the repositories containing code from SO to assess their compatibility with CC BY-SA 3.0 (RQ2, see Section 9). To analyze the adherence to the SO attribution requirements, we manually analyzed a sample of Java files containing a link to an answer on SO (RQ3, see Section 10). To assess the awareness of developers regarding the licensing of code from SO, we conducted a second online survey with GH project owners (RQ4, see Section 11).

We used three main data sources to answer our research questions: The *BigQuery GitHub data set* (Google Cloud Platform, 2017a), the *BigQuery GHTorrent data set* (Gousios, 2013, 2017), and the *BigQuery Stack Overflow data set* (Google Cloud Platform, 2017b). Google BigQuery provides a web-based console that allows to execute SQL queries on various public data sets, including the three data sets listed above. For some aspects of our re-

search, we retrieved additional information from the *Stack Overflow data dump* released March 14, 2017 (Stack Exchange Inc, 2017b), the *GHTorrent data dump* released February 16, 2016 (Gousios, 2013), the *GitHub API* (GitHub Inc, 2017b), and the *Stack Exchange API* (Stack Exchange Inc, 2016).

We decided to restrict our analyses to Java, which is one of the most popular programming languages today (TIOBE software BV, 2017). Using the *BigQuery SO data set*, we retrieved the frequency of question tags. The most common tag (as of March 2017) was `javascript` (1,339,747 questions), followed by `java` (1,223,171 questions). Moreover, we used the *BigQuery GHTorrent data set* to get the most common languages of non-fork projects. Again, JavaScript was the most common language (2,194,750 projects) followed by Java (1,788,748). According to GH's yearly report, Java was, considering the number of opened pull requests in 2017, the third most popular language on GH in that year (after JavaScript and Python) (GitHub Inc, 2018).

We chose Java over JavaScript, because Java has a unique file extension and is usually not embedded in other files (like JavaScript in HTML), which makes isolating Java code in SO posts and searching Java files on GH easier.

In our research, we distinguish between *attributed* and *unattributed* usages of SO code snippets. Attributed usages are relatively easy to detect due to the presence of a link to the content on SO. To detect unattributed usages, we followed three different approaches: In the first phase (see Section 5), we employed regular expressions to find copies of the snippets from the ten most frequently referenced Java answers on SO in all Java files in the *BigQuery GH data set* (10 SO Java snippets, all Java files on GH). In the second phase (see Section 6), we employed a code-clone detector to find clones of a sample of popular SO snippets in a sample of popular GH projects (227 SO Java snippets, 2,313 GH Java projects). In the third phase (see Section 7), we searched for exact matches of as many SO snippets in as many GH Java files as computationally feasible with BigQuery (29,370 SO Java snippets, 1,720,587 GH Java files). Our research mainly focused on finding type-1 clones of snippets, i.e., copied code that only varies in whitespace, layout, or comments (Roy et al, 2009). For such clones, we can be relatively sure that they have actually been copied from SO, assuming that the matches are not too short, the snippets are not too trivial, and there exists no other source.

In the following section, we briefly describe the legal situation, before we present the methods and results for each step of our research. We use framed boxes to summarize the results of each section and provide the raw data and all analysis scripts as supplementary material (Baltes, 2018).

3 Legal Situation

In the following, we first describe the copyright status of SO code snippets, then classify SO's license as a strong copyleft license, and finally point to related discussions on different sites of the Stack Exchange network and related lawsuits.

3.1 Copyright Status of Stack Overflow Code Snippets

First of all, not all code snippets on SO are copyrightable. Generally, "copyright exists automatically whenever someone creates a work of authorship" that is "the author's intellectual creation" (Engelfriet, 2016). While this definition applies for software in general, many SO code snippets are only used to explain or demonstrate a solution, for example showing how to call a particular API. In that scenario, the code would not be creative enough to be copyrightable (Engelfriet, 2016). During the famous *Oracle v. Google* lawsuit (ongoing since 2012), Judge William H. Alsup ruled that APIs itself are generally not copyrightable (Alsup, 2012). However, this decision has been overturned by the Federal Circuit and the lawsuit is still ongoing (Electronic Frontier Foundation, 2018).

Arnoud Engelfriet, a Dutch IT law specialist, provides a rule of thumb that states "if two programmers would provide substantially the same piece of code, the code is not creative under copyright law." He also mentions the often-quoted rule that "anything less than ten lines of code is 'trivial' and therefore not copyrighted", but states that it is not grounded in any copyright legislation he is aware of. Engelfriet concludes that "a [Stack Overflow code] snippet that is more than one or two lines of standard function calls would typically be creative enough for copyright." and also argues against a fair use or quotation argument for such code snippets, mentioned for example by Jeff Atwood, the co-founder of SO (Stack Exchange Meta, 2009).

Since there exists no "international standard for originality" (Creative Commons Corporation, 2017b) that defines when a code snippet is protected by copyright, we used popularity (phase 1), our own judgment (phase 2), and the snippets' length (phase 3) as proxy variables for their originality. In a related study, we found that, as of December 2017, the mean size of code blocks on SO was 12 lines or 455 characters (Baltes et al, 2018), which supports our assumption that many snippets on SO are, at least according to their length, not trivial.

As outlined in the introduction, the code on SO may have been copied from a different source, with additional licensing and copyright implications. We considered this in our research design by analyzing the external availability of the snippets (see Sections 5 and 6) and by excluding snippets that are also available from other sources (see Section 7).

3.2 Classification of Stack Overflow's License

Generally, one can distinguish between *permissive* and *copyleft* licenses. Permissive licenses permit using the licensed source code in proprietary software without publishing changes or the derived work. Examples for permissive licenses include the MIT, Apache, and BSD license families. In contrast to that, copyleft licenses have a *share-alike* requirement that requires either modifications to the licensed content or the complete derived work to be published

under the same or a compatible license. Examples for the former, weaker, copyleft licenses include the Mozilla and the Eclipse Public Licenses (e.g., MPL 2.0 and EPL 2.0); examples for the latter, stronger, copyleft licenses, which are sometime also called "viral" licenses (St. Laurent, 2004), include the GNU General Public Licenses (e.g., GPL 2.0 and 3.0) and the Creative Commons Share-Alike Licenses (e.g., CC BY 2.0). The licenses that apply for the content on SO (CC BY-SA 3.0 and 4.0) fall into the latter category and can thus be classified as strong copyleft licenses.

3.3 Stack Overflow's License Change Attempt

Licensing issues of source code posted on SO have been controversially discussed on different sites of the Stack Exchange network (Stack Exchange Meta, 2009, 2013, 2015). In December 2015, SO tried to switch to the more permissive MIT license for code snippets in new posts. First, they planned to require attribution only upon request of the copyright holder or upon request of SO (Stack Exchange Meta, 2015) but after criticism from the community, they changed their proposal such that attribution would always be required (Stack Exchange Meta, 2016). In January 2016, after a heated discussion, SO delayed the implementation of a new license and since then, no new proposal has been made. Thus, as of July 2018, all source code posted on SO is licensed under CC BY-SA 3.0 (and 4.0) and the *attribution* and *share-alike* requirements apply.

3.4 Related Lawsuits

In the past, courts in the US and Europe ruled that open source licenses are enforceable contracts and that violations of open source licenses can be handled like copyright claims. In the *Jacobsen v. Katzer* lawsuit (2006–2010), the United States Court of Appeals for the Federal Circuit ruled that the terms and conditions of the Artistic License 1.0, including attribution, are "enforceable copyright conditions" (White, 2008). In the *Artifex v. Hancom* lawsuit (since 2016), the United States District Court for the Northern District of California denied a motion to dismiss (Corley, 2017), arguing that a copyleft license like the GNU GPL can be treated like a legal contract. This means that developers are able to sue when the terms of such a license are violated, e.g., when derived work is not shared under a compatible license (see the *share-alike* requirement of CC BY-SA 3.0). Moreover, it is possible to interdict the distribution of such derived work or claim monetary damages: In 2004, the German District Court of Munich affirmed an injunctive relief interdicting the distribution of a software based on source code licensed under the GNU GPL, without complying with its license terms (Kaess et al, 2004). In the United States, open source projects failing to comply with open source licenses can be targeted by DMCA takedown notices, which may force platforms like GH to remove projects that allegedly infringed copyright (Poteat, 2016). Recently,

the Regional Court in Bochum, Germany, affirmed an obligation to pay compensation for damages in a case where source code licensed under the GNU GPL was used in violation of the license terms (Achte Zivilkammer, 2016).

Licensing issues may also be a risk in mergers and acquisitions of companies using source code licensed under a copyleft license (Cavaretta, 2015). A famous case was *Free Software Foundation v. Cisco Systems* (Software Freedom Law Center, 2008): Cisco acquired the networking company Linksys, which used GPL-licensed code in some of their products without publishing the source code. After the Free Software Foundation (FSF) sued Cisco, they reached a settlement agreement, in which Cisco agreed to publish the source code and made an undisclosed financial contribution to the FSF (Wikipedia, 2017).

4 Preliminary Study

We started our research with a preliminary study to get first insights into developers' practices regarding the usage and attribution of code snippets from Stack Overflow.

4.1 Method:

The preliminary study was part of an online survey we conducted in October 2015. For this survey, we contacted users who were active on both SO and GH. To match users on both platforms, we followed the approach of Vasilescu et al., utilizing the MD5 hash value of users' email addresses (Vasilescu et al, 2013). We derived our sampling frame from the data dumps provided by *Stack Exchange* (August 18, 2015) (Stack Exchange Inc, 2015) and *GHTorrent* (September, 25 2015) (Gousios, 2013). To identify active users, we checked if they contributed to a question (asked, answered, or commented) on SO and committed to a project on GH since January 1, 2014. This resulted in a sampling frame of 71,400 users from which we drew a random sample of 1,000 users. Of the 1,000 contacted users, 122 responded (12.2% response rate).

4.2 Results:

Of all 122 respondents, 115 identified themselves as male, one as female and six did not provide their gender. The majority of respondents (67%) reported their main software development role to be *software developer*, the second-largest group were *software architects* (14%). The average age of participants was 28.9 years (SD=9.1) and they had an average programming experience of 11.8 years (SD=6.7). Most participants answered from Europe (49%) and North America (38%).

We asked participants for what purpose they use SO and GH. Most users answered that they use SO (98%) and GH (66%) for both private and work-

related projects. Almost one third of the respondents reported to use GH only for private projects (28%).

A central question of the survey was: "When was the last time you copied or adapted a code snippet from Stack Overflow?" Most participants copied or adapted a snippet not more than one month ago (79%) and over a third (39%) not more than one week ago. To get first insights into the attribution practice, we asked how they referred to the corresponding SO question or answer when they copied or adapted the snippet. Half of the respondents (49%) "just copied/adapted the code snippet without any reference", the others "added a source code comment with a link to the Stack Overflow question/answer" (40%) or referred to SO in another way, e.g., in a commit message (9%). Two participants did not answer this question.

Preliminary Study: Almost all participants (98%) stated that they use SO for both private and work-related projects. Half of them (49%) reported that the last time they copied or adapted a code snippet from SO, they did not attribute its origin; 40% added a source code comment with a link to the corresponding question or answer.

5 Usage Without Attribution (RQ1 – Phase 1)

In our preliminary study, many developers reported that they did not attribute code snippets copied from SO. Most participants who did attribute the snippets added a source code comment with a link to the corresponding question or answer. Thus, we decided to utilize BigQuery to find all links to SO questions and answers in all Java files in the GitHub data set. Afterwards, we built regular expressions matching the snippets from the ten most frequently referenced Java answers and searched for matches in all Java files in the data set to detect unattributed usages of those snippets.

5.1 Method:

Figure 1 visualizes our initial workflow for finding attributed and unattributed usages (including the connection to other research questions). We considered all files ending with `.java` to be Java source code files and applied the following regular expression (regex) to each line of those files:

```
(?i:https?://stackoverflow\.com/[^\s)\.\"]*)
```

Because there are different ways of referring to questions and answers on SO, i.e., using full URLs or short URLs, we mapped all extracted URLs to their corresponding sharing link (ending with `/q/<id>` for questions and `/a/<id>` for answers). In the following, we use the term *reference* to denote a link to content on SO. In the database schema of the *BigQuery GH data set*, copied files have the same ID (hash value of the content). For our analysis, we only considered

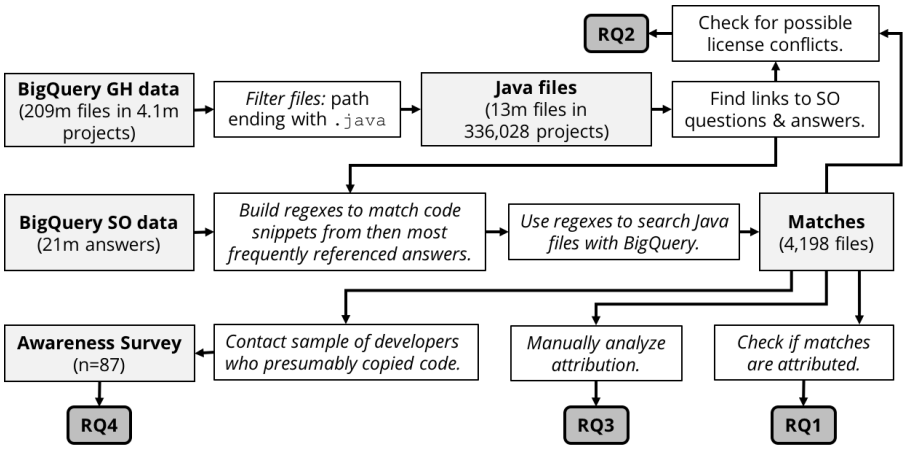

Fig. 1 RQ1-4 – Phase 1: We searched for attributed and unattributed usages of the code snippets from the ten most frequently referenced answers on Stack Overflow (SO) in all Java files in the BigQuery GitHub (GH) data set using regular expressions and used this data to answer our research questions (time span of this phase: 07/2016–11/2016).

distinct references, meaning that we counted references in files with the same content only once. Because many files on GitHub are duplicates (Gharehyazie et al, 2017; Lopes et al, 2017), we distinguish between the number of *distinct referencing files*, meaning the number of distinct files in which a URL was present in a source code comment, and the number of *distinct referencing lines*, meaning the number of distinct source code lines in which a URL was used (exact string match including whitespaces). The former may exaggerate the number of distinct references as files may be copied and then slightly changed, the latter may understate the number of distinct references as two developers may independently use the same source code line to reference a question or an answer. According to Google, most forks were excluded in their *BigQuery GH data set*. In the first phase, we relied on the unique file IDs to exclude copied files. In the third phase, we further excluded all repositories that were marked as forks in the *BigQuery GHTorrent data set* (see Section 7).

Our first approach for finding unattributed usages of SO snippets utilized manually created regular expressions to find matches of non-trivial code snippets in all public GH projects containing Java code. Since this approach is time-consuming, we had to carefully select the snippets for which we then built the regular expressions. We decided to extract the code snippets from the ten most frequently referenced Java answers on SO, because we thought that these snippets are likely to be also used without attribution (assuming that the attribution ratio is relatively stable across posts). In a next step, we randomly chose (up to) ten Java source code files referencing the corresponding SO answer. Then, we manually created a regex for each SO snippet and iteratively modified it to match both the snippet and as many of the refer-

Table 1 RQ 1 – Phase 1: Ten most frequently referenced code snippets from SO Java answers; one asterisk: link was broken and referred to a question, we selected two referenced snippets; two asterisks: snippet based on external resource, but adapted.

Rank	Answer ID	Description	Type	Ext. Availability
1	3758880	human readable byte size	method	blog, no license
2	5445161**	read InputStream to String	method	blog, no license
3	9855338**	convert byte array to hex String	method	other SO post
4	26196831	Android: RecyclerView onClick	class	none
5	7696791*	Android: close soft keyboard	snippet	none
6	140861	hex dump String to byte array	class	none
7	2581754	sort Map<Key, Value> by values	class	none
8	5599842	format file size as MB, GB, etc.	method	none
9	326440	create Java String from file cont.	method	none
10	3145655	Android: get current location	class	none

encing Java files as possible, while taking care that it does not become too generic, leading to false positives.

Table 1 lists the ten most frequently referenced Java answers. In the table, we included a short description of the thread's topic and mention whether the code in the answer is a whole class, a single method, or just a few lines of code (snippet). We also added information about the external availability of the source code from the SO post. The top-ranked Java snippet was also available on a personal blog post by the same author. However, the author has the copyright for his blog post and provides no license, thus only the SO post allows the usage of that snippet. Further, the SO thread is the first result on Google (as of June 8, 2017) when searching for "human readable byte size java". Therefore, the SO post is likely the primary source for copying this particular snippet. The second snippet is based on a blog post by a different author, also copyrighted without providing a license. Moreover, this blog post is only available using the *Internet Archive Wayback Machine*.[2] Therefore, also for this snippet SO is likely to be the primary source. The third snippet is based on a different SO post, but had been adapted. Thus, the license is still CC BY-SA 3.0. For the other snippets, we could not identify an external source with a different license.

Table 2 shows, for each of the ten Java answers, the number of distinct referencing lines (L_A) and the number of distinct referencing files (F_A). Further, we provide the number of distinct files with a reference to either to the answer or to the corresponding question (F_{AQ}). For this value we do not know if the developer actually wanted to refer to the snippet from the specific answer we are considering or to another answer from the same thread. The table also shows the number of GH references we used to test the regular expression and how many of those references the regex matched.

We used BigQuery's `REGEXP_MATCH` function to check all Java files in the GH data set for matches of each regex. We provide the extracted SO snippets,

[2] `http://web.archive.org/`

Table 2 RQ 1 – Phase 1: Ten most frequently referenced code snippets from SO Java answers, references in GH Java files and testing of regular expressions for those snippets; L_A: number of distinct referencing lines, F_A: number of distinct referencing files, F_{AQ}: number of distinct referencing files including references to corresponding question.

Rank	References			Regex		
	L_A	F_A	F_{AQ}	TESTED	MATCHED	RECALL
1	21	43	122	10	9	90.0%
2	20	39	100	10	7	70.0%
3	19	27	108	10	10	100.0%
4	12	15	19	10	9	90.0%
5	9	20	34	9	4	44.4%
6	8	12	74	7	7	100.0%
7	8	9	41	8	8	100.0%
8	7	17	36	7	5	71.4%
9	7	12	47	7	1	14.3%
10	7	12	26	6	6	100.0%
All	118	206	607	84	66	*M* 78.0%

the referencing Java code from GH, the regular expressions, and the SQL scripts as supplementary material (Baltes, 2018).

5.2 Results:

Table 3 shows how many files of the data set each regex matched and how many of those matches were distinct files. We report how many of the matched files contained a reference to the answer or the corresponding question (REF) and how many files did not contain a reference (NO-REF). We also calculated the recall by comparing F_{AQ} and REF, i.e., the number of distinct files with a reference to either the answers or the corresponding question and the number of matched files containing such a reference. This allowed us to assess how good the regex was in matching possible duplicates of the snippet.

We calculated two estimates for the ratio of files with attributed snippets: First, we compared the number of distinct referenced matches (REF) to the total number of distinct matches (DISTINCT). The second estimate is the number of distinct matches with a reference either to the answer or to the corresponding question (F_{AQ}) compared to the number of distinct matches (DISTINCT). Please note that the comparisons with F_{AQ} understate the recall and overstate the attribution ratio, because F_{AQ} likely includes references to other answers of the thread. To evaluate the number of false positives, we checked (up to) 50 matches for each regex and found no match that we did not consider to be a clear copy of the snippet.

To illustrate the procedure, we present the snippet from the most frequently referenced Java answer and the corresponding regex below. The snippet is a method returning a human-readable string representation of a byte value (e.g., for 1024 it returns 1.0 kB or 1.0 KiB) (Aioobe, 2010). It was referenced in 21 distinct lines and in 43 distinct files, meaning that several files used the

Table 3 RQ 1 – Phase 1: Ten most frequently referenced code snippets from SO Java answers; estimated ratio of unattributed usages detected using regular expressions; number of matched files (ALL), distinct matches (DISTINCT), distinct matches with reference to SO (REF), distinct matches without reference to SO (NO-REF).

Rank	Matches				Recall	Attribution	
	ALL	DISTINCT	REF	NO-REF	REF/F_{AQ}	REF/DISTINCT	F_{AQ}/DIST.
1	997	448	97	351	79.5%	21.7%	27.2%
2	1,843	913	60	853	60.0%	6.6%	11.0%
3	2,662	902	87	815	80.6%	9.6%	12.0%
4	420	170	18	152	94.7%	10.6%	11.2%
5	1,492	402	25	377	73.5%	6.2%	8.5%
6	2,642	807	65	742	87.8%	8.1%	9.2%
7	160	124	12	112	29.3%	9.7%	33.1%
8	355	174	22	152	61.1%	12.6%	20.7%
9	295	225	5	220	10.6%	2.2%	20.9%
10	65	33	11	22	42.3%	33.3%	78.8%
All	10,931	4,198	402	3,796	*M* 61.9%	*M* 12.1%	*M* 23.2%

same line content to reference the snippet. Together with the corresponding question, we found 122 distinct referencing files (see Table 2).

```
String humanReadableByteCount(long bytes, boolean si) {
    int unit = si ? 1000 : 1024;
    if (bytes < unit) return bytes + " B";
    int exp = (int) (Math.log(bytes) / Math.log(unit));
    String pre = (si ? "kMGTPE" : "KMGTPE").charAt(exp-1) + (si
        ? "" : "i");
    return String.format("%.1f %sB", bytes / Math.pow(unit,
        exp), pre); }
```

Starting with the above snippet, we created a regular expression and iteratively refined it until it matched 9 out of 10 referencing files from GH. The final regular expression, which can be found below, matched 80% of the files containing a reference either to the answer itself or the corresponding question.

```
((?i:String[\s]+\w+\([^\{]*long[^\{]+\)[\s]*\{[\s\S]+if
[\s]*\([^<]+<[^\)]+\)[\s\S]*return[^;]+\+[^;]*\"\ B\"
[\s\S]+int[\s][^\=]+\=[\s]*\([\s]*int[\s]*\)[\s]*\([\s]*
Math[\s]*\.[\s]*log[\s]*\([^\)]+\)[\s]*\/[\s]*Math[\s]*
\.[\s]*log[\s]*\([^\)]+\)[\s]*\)[\s\S]+return[^\}]+
String[\s]*\.[\s]*format[\s]*\([^\}]+\}))
```

Only 21.7% of the matched files were attributed; compared to F_{AQ} the ratio was 27.2%. On average, the regular expressions we created matched 78.0% of the referencing GH Java files from the test sets (see Table 2) and 61.9% of the files in F_{AQ} (see Table 3). The average ratio of attributed matches was 12.1%; compared to F_{AQ}, the ratio was still only 23.2%. As motivated above, the latter ratio overstates the amount of referenced usages and can thus be considered an upper bound. Because the regular expressions were rather strict and false positives were not present in the samples we checked, we can estimate that at

most 23.2% of the copies of the ten most frequently referenced SO Java code snippets are being attributed when copied into Java files on GH.

Usage Without Attribution (RQ1 – Phase 1): At most 23.2% of the copies of code snippets from the ten most frequently referenced SO Java answers in Java files on GH were attributed using a link to SO.

6 Usage Without Attribution (RQ1 – Phase 2)

To triangulate our estimate from the first phase that at most 23.2% of the usages of SO code snippets in GH projects are attributed, we followed a second approach and used a token-based code clone detector, the *PMD Copy-Paste Detector* version 5.4.1 (PMD, 2016), to find unreferenced usages of SO code snippets in a random sample of popular GH Java projects.

6.1 Method:

We decided to use the *PMD Copy-Paste Detector* (CPD) for finding clones of SO snippets, because this tool is open-source, actively developed, and widely used. It is integrated into the IntelliJ Java IDE and there are plugins for other IDEs as well.

The detection of code clones within a set of source code files is a computationally expensive task. Therefore, we had to restrict our analysis to a sample of GH Java projects and a sample of Java code snippets from SO. A random sample of GH projects would contain many small personal projects, homework assignments, or other projects that are not "engineered software projects" (Kalliamvakou et al, 2014; Munaiah et al, 2017). Filtering projects according to their popularity, measured using the number of watchers or stargazers, has been used in several well-received studies and proved to have a very high precision (almost 0% false positives) (Munaiah et al, 2017). Hence we applied a similar filtering strategy.

Our sampling frame consisted of all Java projects in the *GHTorrent data set* (February 16, 2016) that were no forks, not deleted, and had at least 29 watchers (99% quantile for all Java projects, see Figure 2). We excluded forks, because they may skew the results by adding almost identical repositories to the sample. We excluded deleted repositories, because we would not be able to retrieve the source code of such repositories. From this sampling frame ($n = 9,437$), we randomly selected 3,000 Java projects. We were able to successfully download 2,313 of them. Some downloads failed, because our script only tried to retrieve the master branch, which may not exist, and some repositories may have been renamed or deleted between the creation of the *GHTorrent data dump* and the time we downloaded the sample (April 21, 2016).

We searched for two different sets of SO code snippets in the sample of GH projects: One set with snippets that had referenced usages in the GH projects

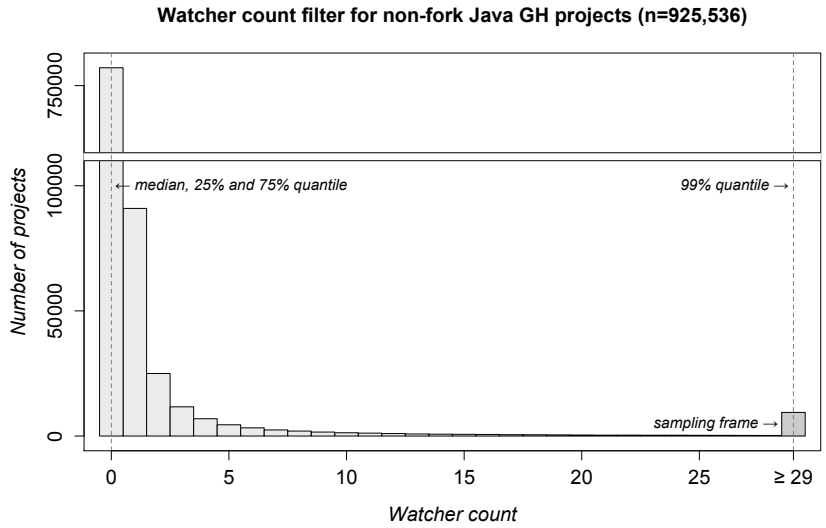

Fig. 2 RQ1 – Phase 2: Histogram visualizing the selected sampling frame of popular GitHub Java projects ($n = 9,437$); the 99% quantile of all non-fork Java projects was 29 watchers ($M = 2.77$, $Mdn = 0$, $Q_{1,3} = 0$); based on the GHTorrent data dump 2016-02-01.

under analysis (S_{gh}) and one set with popular Java snippets identified using the data from the first phase of our research (S_{top100}). The first set allowed us to compare referenced usages of SO snippets to unreferenced usages of the same snippets; the second set allowed us to analyze how many copies of popular SO Java snippets were being attributed in the sample of GH projects. For the first set, we searched for references to SO in all Java files in the project sample using the same regular expression as in the first phase. We then manually extracted the snippets from all referenced answers, dropping answers that did not contain code or only trivial snippets (e.g., simple API calls, snippets for conceptual questions, etc.). This resulted in a total number of 137 extracted SO snippets. For the second set of snippets, we manually extracted the code from the 100 most frequently referenced Java answers, identified using the same data and ranking approach as in the first phase (see Section 5). We used the number of distinct referencing lines as the primary and the number of distinct referencing files as the secondary sort key. This resulted in 111 snippets. We provide all extracted snippets and the names of all analyzed Java projects as supplementary material (Baltes, 2018).

As a last preparation step, we checked the intersection of the two snippet sets to prevent snippets that are in both sets biasing the results. We identified 26 snippets from 18 answers to be in $S_{\text{gh}} \cap S_{top100}$. We present the results for each snippet set separately and count the intersecting snippets and matches only once in the summary. Before presenting the results, we describe how we calibrated CPD for finding SO snippets in Java projects.

6.2 Calibration of the Code Clone Detector:

We iteratively optimized CPD's parameters using S_{gh} as ground truth, because for this set of snippets we already identified the attributed usages and could thus determine precision and recall. For Java files, the relevant parameters to configure CPD are the minimum token length that should be reported as a duplicate (`mt`) and three boolean flags to configure text comparison: One to ignore language annotations (`ia`), one to ignore constant and variable names (`ii`), and one to ignore number values and string contents (`il`). To compare the results of different iterations, we used the following definitions of *precision* and *recall*:

Definition 1 Let C (copies) be a relation over a set of code snippets S and a set of source code files F:

$$C \subseteq S \times F$$

Let $C_{\text{so}} \subseteq C$ be the set of copies identified by an SO answer URL in the source code file and $C_{\text{cpd}} \subseteq C$ be the set of copies identified by CPD. Then we define precision and recall as follows:

$$\text{precision} = \frac{|C_{\text{so}} \cap C_{\text{cpd}}|}{|C_{\text{cpd}}|} \qquad \text{recall} = \frac{|C_{\text{so}} \cap C_{\text{cpd}}|}{|C_{\text{so}}|}$$

Please note that the precision may be <1 even if all copies found by CPD are actually duplicates of a snippet in S_{so}. The reason for this is that the Java files in our test set may contain copies of these snippets that are either unreferenced or are referenced using a link to the question. As CPD cannot be configured to only find clones of one set of files in another, we wrote a wrapper to exclude matches within the snippets and within the analyzed projects. The wrapper returns the matches between snippets and Java files in the projects along with the line numbers of the exact positions of each match. From this data, we derived the relation C_{cpd}. An example for one entry is provided below:

```
so-answer-3054692, Floobits-floobits-intellij/.../Utils.java
```

In this example, the snippet extracted from the SO answer with ID `3054692` was found in the file identified by the given path (the root is the name of the GH repository).

We derived C_{so} from the references we already extracted (S_{gh}). Using these two relations, we calculated precision and recall for each test run according to the above definitions.

Figure 3 shows the results for different configurations of CPD. We conducted three test runs with `mt` $\in \{15, 20, ..., 95, 100\}$: (1) without further parameters, (2) with flag `ia` set, and (3) with flags `ia` and `ii` set. First, we also included the flag `il`, but with the relatively small values we used for `mt` this resulted in too many false positive results. Moreover, setting `il` lead to a runtime that was magnitudes longer than the other configurations. Because our goal was to increase precision and avoid false positives, we dropped `ii` despite

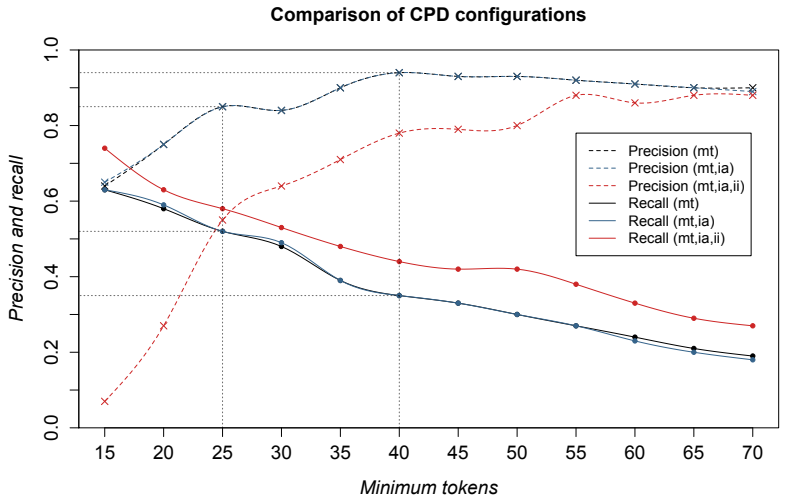

Fig. 3 RQ1 – Phase 2: Comparison of different CPD configurations: black: only `mt` set; blue: `mt` and `ia` set; red: `mt`, `ia`, and `ii` set; dashed line: precision, solid line: recall; final configuration: `mt` = 40 (precision = 0.94, recall = 0.35).

Table 4 RQ 1 – Phase 2: Results for different CPD configurations; all matches, distinct snippet-file pairs, true positive matches ($C_{\text{so}} \cap C_{\text{cpd}}$), false positive matches ($C_{\text{cpd}} \setminus C_{\text{so}}$), precision, and recall.

Configuration	Matches					
	ALL	DISTINCT	TRUE POS.	FALSE POS.	PRECISION	RECALL
`mt`=40	103	51	48	3	94%	35%
`mt`=25	268	84	72	12	85%	53%

the slightly higher recall. Since the flag `ia` had almost no effect on precision and recall (only few snippets with annotations in S_{gh}), we also dropped it.

We achieved the highest precision by setting `mt`=40 without further parameters (*prec*=0.94, *rec*=0.35). We selected `mt`=25 as a second candidate because of its higher recall (*prec*=0.85, *rec*=0.53). Table 4 shows the results for these two configurations. We divided the matched files into *true positive* ($C_{\text{so}} \cap C_{\text{cpd}}$) and *false positive* results ($C_{\text{cpd}} \setminus C_{\text{so}}$). We manually investigated all true and false positives for the two configurations and found that all matches were true positives; the false positives were clones that were not referenced. Nevertheless, the configuration with `mt`=25 contained some relatively small matches, e.g., parts of for-loops, that were likely to produce false positives outside of our test collection. Based on the results of our test runs, we chose `mt`=40 for the final CPD configuration. As we decided not to set `ia` and `ii`, this configuration can only detect type-1 clones of the snippets, i.e., copied code that only varies in whitespace, layout, or comments (Roy et al, 2009).

6.3 Results:

Using the configuration `mt`=40, we searched for type-1 clones of the two snippet sets S_{gh} and S_{top100} in two separate runs. Each run took between 8 and 9 hours on a regular Desktop PC running Ubuntu 14.04 LTS (Intel Core i5-4670, 16 GB RAM, SSD). Table 5 lists the results for each snippet set. It shows the number of snippets in each set, the number of answers from which the snippets were extracted, and the number of matched snippets, answers, and files. In the analyzed GH projects, we found 634 Java files from 274 projects that contained a reference to SO (0.14% of all Java files in the sample and 12% of all projects. The table shows how many of the matched files contained a reference to SO and the number of repositories containing a matched file.

In a first data cleaning step, we analyzed the results and found that one of the snippets in S_{top100} was responsible for 272 matches (48% of all matches). This snippet contained a long list of invalid characters in file names. We looked at the matched files and found that most of the matches used this array in another context than the SO snippet. Thus, we considered these matches to be false positives and excluded them from our analysis. To estimate the number of false positives in the remaining matches, we randomly chose 100 distinct matches (snippet-file pairs) from each set and manually checked whether the files actually contained a copy of the snippet. We rated all analyzed matches as true positives.

We further checked if the snippets were available from an external source, meaning a website, blog, or source code repository outside of the SO platform. If snippets were also available outside of SO, more permissive licenses could apply that allow using the snippet without attributing SO as the source. We followed all links in the SO answers from which the snippets were taken and checked if the snippet was available in the linked resource. If it was available, we searched the websites for licenses or terms of service that apply for the content. Tables 6 and 7 summarize the results of this analysis. Table 6 shows how many answers provided an external source for the snippet (12%), together with the type of the source. We found copies of the snippets in blog posts (8), GitHub repos (6), Android or Java bug reports (5), and in the official Android or Java documentation (2). For the answers having an external source, Table 7 shows if this source allows to use the snippet under a more permissive license than CC BY-SA 3.0. Twelve of those 21 answers provided a license or terms of service, of which only three were more permissive than Stack Overflow's license: In one case,[3] the author added a comment indicating that the snippet is free to use: "There is no copyright on the code. You can copy, change and distribute it freely. Just mentioning this site should be fair"; two sources were licensed under the Apache 2.0 license. One source was licensed under the GNU GPL 2.0, which is also a copyleft license and hence not more permissive than CC BY-SA 3.0; the other eight sources had terms of service restricting the usage of the snippet. We can conclude that even if some snippets are also available

[3] `http://balusc.omnifaces.org/2007/07/fileservlet.html`

Table 5 RQ 1 – Phase 2: Results of searching copies of two sets of Stack Overflow snippets in a sample of GitHub projects ($n = 2,313$): Columns named MATCHED show number of distinct matched snippets, answers, files, and repos; column REF shows number of matched files containing a reference to Stack Overflow.

Set	Snippets			Files			Repos
	ALL	MATCHED	ANSWERS	MATCHED	MATCH.	REF	MATCHED
S_{gh}	137	53 (39%)	102	52 (51%)	163	58 (36%)	124 (5%)
S_{top100}	111	48 (43%)	85	46 (54%)	173	25 (14%)	125 (5%)
$\cup S$	222	101 (46%)	169	86 (51%)	297	70 (24%)	199 (9%)

Table 6 RQ 1 – Phase 2: External sources for snippets: The table shows the number of answers with snippets in the two sets and how many of those answers contained a link to an external source. Abbreviations: Snippets also available in a blog post (BLOG), in a GitHub repository (GH), in an Android or JDK bug description (BUG REPORT), in an Android or Java documentation page (DOC).

Set	External source in SO answers						
	ALL	NO	YES	BLOG	GH	BUG REPORT	DOC
S_{gh}	102	89 (87%)	13 (13%)	6 (6%)	2 (2%)	4 (4%)	1 (1%)
S_{top100}	85	76 (89%)	9 (11%)	2 (2%)	5 (6%)	1 (1%)	1 (1%)
$\cup S$	169	148 (88%)	21 (12%)	8 (5%)	6 (4%)	5 (3%)	2 (1%)

Table 7 RQ 1 – Phase 2: License of external sources for snippets: The table shows under which licenses the snippets from external sources can be used; NO: no license provided, FREE: author added a comment that the code is free to use, ToS: usage is restricted by the website's terms of service, APACHE 2.0: available under the Apache 2.0 license, GPL 2.0: available under the GPL 2.0 license.

Set	License of external sources						
	ALL	NO	YES	ToS	FREE	APACHE 2.0	GPL 2.0
S_{gh}	13	4 (31%)	9 (69%)	7	1	1	0
S_{top100}	9	6 (67%)	3 (33%)	1	0	1	1
$\cup S$	21	9 (43%)	12 (57%)	8	1	2	1

outside of SO, this does not necessarily mean that the external sources are more permissive than SO's license.

Overall, CPD found one or more copies of snippets from the two snippet sets in 297 distinct files. The identified clones were duplicates of 101 different snippets (46% of all distinct snippets in the sets) from 86 answers (51% of all answers in the sets). Only 70 matched files (24%) contained a reference to a SO question or answer. In summary, 199 repositories (9% of all repositories in the sample) contained files with copies of snippets from SO. As we did not observe any false positive results (except for the match we excluded in the data cleaning step, see above), the number of matches can be interpreted as a lower bound for the amount of copies that are actually present in the sample.

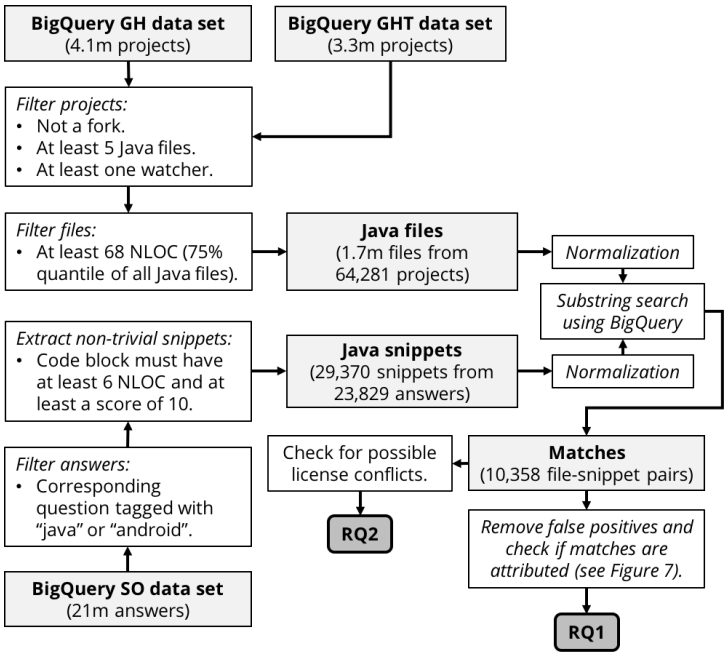

Fig. 4 RQ1 – Phase 3: We searched for as many exact matches of Java snippets from Stack Overflow (SO) in public GitHub (GH) projects as feasible. We filtered the GH Java projects to exclude small 'toy' projects and further excluded short and unpopular SO snippets. NLOC means that we normalized the source code before we determined its length. In the end, we searched for exact matches of 29,370 snippets in 1,7m Java files (50.5 billion combinations) (time span of this phase: 03/2017–04/2017).

Usage Without Attribution (RQ1 – Phase 2): Using CPD, we found that in a sample of popular Java projects (n=2,313), 199 repositories (9%) contained a copy of one of the 222 SO snippets we considered. Only 24% of the matched files contained a reference to SO as required by SO's license.

7 Usage Without Attribution (RQ1 – Phase 3)

Our third and last approach to answer RQ1 addressed the main shortcoming of the previous phases, which was the relatively small number of SO code snippets being analyzed. Since the approaches of the first two phases do not scale due to the manual creation of regular expressions (phase 1) or the performance of the code clone detector (phase 2), we focused on exact matches of SO snippets in the third phase, which are easier to find. We searched for exact matches of SO snippets in GH projects using the public BigQuery *GH*, *GHTorrent*, and *SO* data sets (Google Cloud Platform, 2017a,b; Gousios, 2017) and iteratively filtered the resulting matches to exclude false positives and snippets that were also available in other sources than SO.

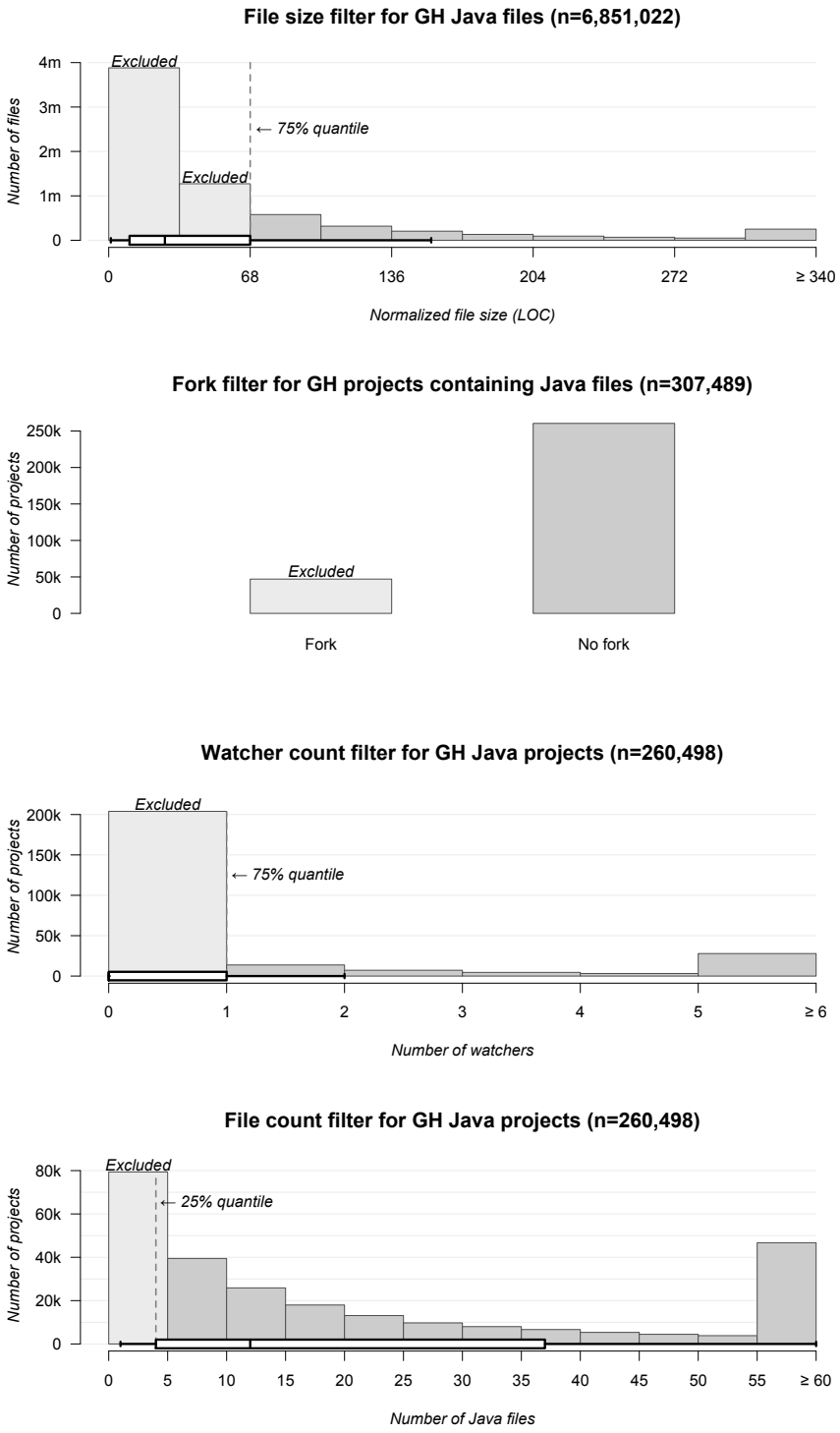

Fig. 5 RQ1 – Phase 3: Barplot and histograms with boxplots visualizing the applied filters to reduce the number of GitHub (GH) Java files we searched for exact matches of Stack Overflow (SO) snippets; 65 LOC was 75% quantile of all Java files on GH; 1 watcher was 75% quantile of all GH projects containing Java files; 4 files was 25% quantile of all GH projects containing Java files; based on the GHTorrent BigQuery data set 2017-01-19.

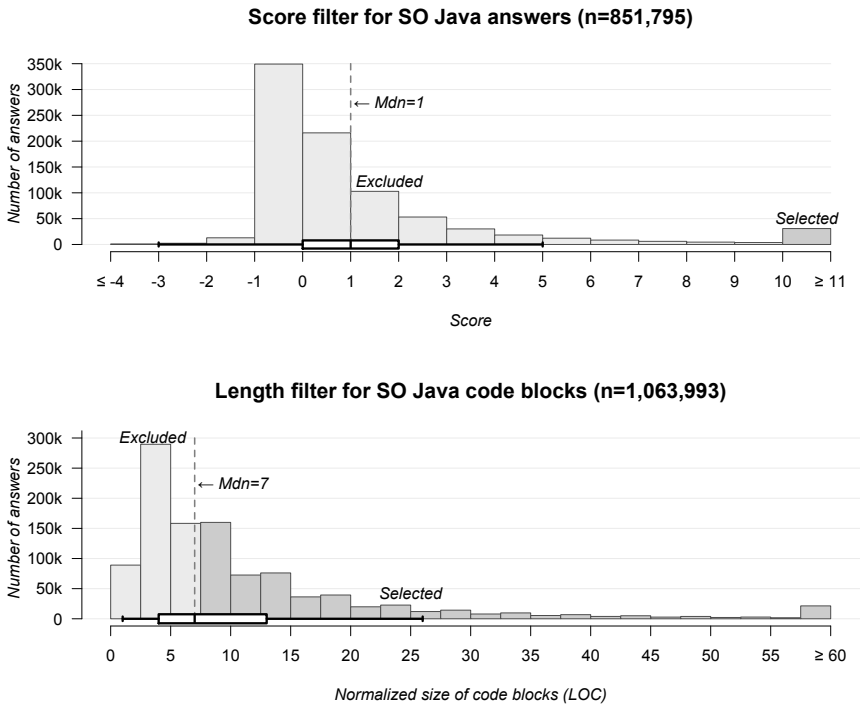

Fig. 6 RQ1 – Phase 3: Histograms with boxplots visualizing the applied filters to reduce the number of Java code snippets from Stack Overflow (SO) in our search for exact matches of these snippets in Java files hosted on GitHub (GH); based on the Stack Overflow BigQuery data set 2017-03-27.

7.1 Method:

For various reasons, it is not feasible and sensible to search for all code snippets on SO in all projects on GH. BigQuery's *GH data set* consists of (almost) all public files on GitHub, which includes many small software projects of single users and also repositories that are not used for hosting software projects (Kalliamvakou et al, 2014; Munaiah et al, 2017). Moreover, very small code snippets from SO would produce many false positives and it is likely that such snippets are not protected by copyright. Since there is no "international standard for originality" (Creative Commons Corporation, 2017b) that defines when a code snippet is protected by copyright, we based our filter on the length distribution of SO code snippets and only selected snippets having a certain size. We thus used the length of the snippets as a proxy variable for their originality.

Another reason to filter the snippets and projects was to reduce the complexity to make the search for exact matches feasible. For every filter we applied, we considered the distribution of values for the corresponding variables. Figure 4 visualizes how we filtered the Java files and the Java snippets to

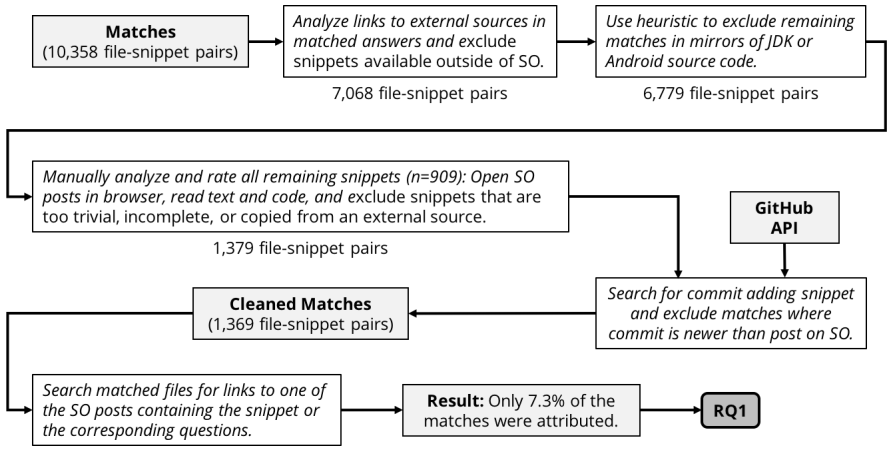

Fig. 7 RQ1 – Phase 3: Our workflow to remove false positive matches and snippets available in other sources than the SO post.

reduce the number of combinations to a level that allowed us to employ BigQuery's `STRPOS` function to search for matches of the snippets in the files.

We first used the BigQuery *GHTorrent data set* to filter out repositories that were forks of other repositories (see Figure 5). Then, we excluded projects with less than five Java files and less than one watcher to get rid of the many 'toy' projects hosted on GitHub (Kalliamvakou et al, 2014). Afterwards, we normalized the contents of the remaining Java files by removing all lines with import or package statements, deleting comments, and normalizing the whitespaces (removing empty lines and converting multiple newline characters to one newline character). We excluded Java files having less than 68 normalized lines of code, which was the 75% quantile for all Java files, resulting in a sample of 1,720,587 files from 64,281 projects. To improve the substring matching, we then further normalized the file contents by converting the characters to lower case and deleting semicolons, curly and regular braces, and all whitespace characters.

To retrieve the Java snippets for the substring search, we first extracted all answers to questions tagged with `java` or `android` from the BigQuery *SO data set*. Then, we analyzed the score of the answers. To concentrate on answers that gained a certain degree of attention, we excluded all answers with a score of less than ten (see Figure 6). Then, we used the Markdown representation of the posts to identify and extract continuous code blocks from the answers. We normalized the snippets exactly like the Java files and analyzed their length. As we were interested in non-trivial code snippets, we removed all code blocks with less than 6 normalized lines, which excluded about a third of the code blocks. We then further normalized the snippets, analogously to the file contents. To illustrate the normalization, we provide the normalized version of the snippet presented in Section 5.2:

```
stringhumanreadablebytecountlongbytes,booleansiintunit=si?1000:
1024ifbytes<unitreturnbytes+"b"intexp=intmath.logbytes/math.log
unitstringpre=si?"kmgtpe":"kmgtpe".charatexp-1+si?"":"i"return
string.format"%.1f%sb",bytes/math.pow(unit,exp),pre
```

After the substring search was complete, we employed different approaches to exclude false positives from the matches (see Figure 7): First, we manually investigated all matches of SO answers with links to external sources and checked whether the code on GH may have also been copied from there. We observed that many repositories contained mirrors of the *OpenJDK* or the *Android* source code. To exclude matches involving files from those sources, we used a heuristic based on path names. We then manually investigated the SO posts of all remaining snippets and excluded snippets that we either rated as being too trivial or incomplete, or where the post indicated that the snippet has been copied from a third source (without providing a link). As motivated above, there is no international standard defining when a snippet is original enough to be copyrightable. Our notion of 'too trivial' included snippets that consist only of a few API calls or that are so simple that another developer would likely come up with the same code. Moreover, we checked if the snippets are 'complete' in the way that they are ready to copy-and-paste, without substantial modifications. Since those judgments are, to some degree, subjective, we tried to mitigate a possible bias by discussing borderline cases.

As a last step, we employed the *GitHub API* and searched for the commit that added the snippet to the repo. We then removed matches where the commit on GH was older than the post on SO.

7.2 Results:

After removing potential false positive matches and snippets that were also available in other sources, the result set consisted of 1,369 snippet-file pairs. For the remaining matches, we are quite confident that they are in fact clones of the SO snippets and are not copied from a different source. Only 104 (7.6%) of the snippet-file pairs were attributed using a link to one of the SO posts containing the snippet (some snippets were present in more than one post) or to the corresponding questions. We found exact matches of SO snippets in 764 (1.19%) of the 64,281 analyzed GH repositories. Using the *BigQuery GH data set*, we also analyzed the licenses of those repositories. The results of this analysis can be found in Section 9.

Usage Without Attribution (RQ1 – Phase 3): We searched for exact matches of 23,829 Java snippets from SO in 64,281 GH projects and excluded snippets available in external sources. Only 7.6% of the 1,369 matches were attributed.

Table 8 Summary of results from phases 1 to 3: Distinct references to answers (A) or questions (Q) on Stack Overflow (SO) in the Java files from GitHub analyzed in each phase; number of analyzed files and repositories, files/repos containing a reference to SO, files/repos containing a copy of a SO snippet, attributed copies of SO snippets.

Ph.	References		Files				Repositories		
	A	Q	Count	Ref	Copy	Attr	Count	Ref	Copy
1	5,014	16,298	13.3m	18,605	4,198	402	336k	11,086	3,291
	23.5%	76.5%		0.09%	0.03%	9.6%		3.3%	1.0%
2	209	463	445k	634	297	70	2,313	274	199
	31.1%	68.9%		0.14%	0.07%	23.6%		11.9%	8.6%
3	1,551	4,843	1.7m	5,354	1,369	104	64,281	3,536	1,332
	24.3%	75.7%		0.31%	0.08%	7.6%		5.5%	2.1%

8 Summary (RQ1 – Phases 1–3)

In this section, we summarize our results from all three approaches to quantify the amount of unattributed usages of non-trivial Java code snippets from SO in public GH projects (RQ1). For each phase, Table 8 provides an overview on: (1) the number of distinct references to answers and questions in the analyzed Java files, (2) the number of Java files and repositories we analyzed, and (3) the number and ratio of attributed usages of SO code snippets in the analyzed files.

Taking all three phases into account, we consider one quarter to be a reasonable upper bound for the ratio of attributed usages of SO Java snippets in GH files (see Table 8, column Files → attr). Between 3.3% and 11.9% of the analyzed repositories contained references to Stack Overflow questions or answers (Repositories → Ref). The table further shows the number of distinct analyzed files (Files → count), along with the percentage of files containing a reference to SO (Files → ref).

Moreover, Table 8 lists the number of distinct references to SO posts we identified in each phase (References), where distinct means that we counted copied files only once. If one file contained the same URL several times, we also counted it only once. In our analysis, we ignored URLs that were either malformed or referred to other content on SO such as tags or users. For instance, of all SO URLs we found in the first phase, 2.16% did not refer to a question or an answer.

Generally, developers were more likely to refer to a question, that is to the whole thread, compared to a particular answer. In the first phase, only 0.09% of the analyzed files and only 3.3% of the analyzed projects contained a reference to SO. However, these results include all public files on GH in the *BigQuery data set*, which includes many small software projects of single users and also repositories that are not used for hosting software projects (Kalliamvakou et al, 2014; Munaiah et al, 2017).

Table 9 Five most common licenses of GitHub repositories matched in phase 1 containing attributed or unattributed copies of code snippets from Stack Overflow.

SPDX license name	Number of repos containing a SO code snippet clone that was: unattributed ($n = 2,962$)	attributed ($n = 329$)
Apache-2.0	921 (31.1%)	99 (30.1%)
MIT	621 (21.0%)	72 (21.9%)
GPL-3.0	435 (14.7%)	60 (18.2%)
GPL-2.0	284 (9.6%)	21 (6.4%)
BSD-3-Clause	82 (2.8%)	9 (2.7%)

9 Frequency of Licensing Conflicts (RQ2)

To assess how often the license of repositories containing code copied from SO conflicts with SO's license (RQ2), we retrieved the license of all repositories previously identified as containing code from SO. To this end, we employed the *GitHub API* (phase 1+2) and the *BigQuery GH data set* (phase 3). Tables 9, 10, and 11 show the five most common licenses of the matched repositories from each phase. We provide the complete lists as supplementary material (Baltes, 2018). Between 1.82% (attributed matches in phase 1) and 38.9% (unattributed matches in phase 2) of the matched repositories did not provide a license (or at least none that the GitHub API was able to identify). The relatively large number of repositories without a license may seem unusual, but it is in line with a recent study by Meloca et al., who found that it is common in open source projects to not provide a license (Meloca et al, 2018). Moreover, some files or directories could have their own license, differing from the repository's license. As we never observed this for the files we manually analyzed, we relied on the repository's license for our analysis.

None of the analyzed projects used the CC BY-SA 3.0 or the CC BY-SA 4.0 license, which would be share-alike compatible with the content from SO. One could leverage the upwards compatibility of CC BY-SA 3.0 and CC BY-SA 4.0 (Creative Commons Corporation, 2017a) and the share-alike compatibility of CC BY-SA 4.0 and GPL 3.0 to achieve a share-alike compatibility of CC BY-SA 3.0 and GPL 3.0. Still, only 60 (1.8% of all matched repos in phase 1), 6 (3.0% of all matched repos in phase 2), respectively 19 (1.4% of all matched repos in phase 3) repositories were licensed under GPL 3.0 and attributed the code copied from SO as required by the license. Thus, only those 85 repositories (1.8% of all matched repos) may have used the snippets in a way compatible with CC BY-SA 3.0, meaning with attribution and with a share-alike compatible license.

Frequency of Licensing Conflicts (RQ2): At most 1.8% of all analyzed repositories containing code from SO used the code in a way compatible with CC BY-SA 3.0.

Table 10 Five most common licenses of GitHub repositories matched in phase 2 containing attributed or unattributed copies of code snippets from Stack Overflow.

SPDX license name	Number of repos containing a SO code snippet clone that was: unattributed ($n = 144$)	attributed ($n = 55$)
None	56 (38.9%)	18 (32.7%)
Apache-2.0	33 (22.9%)	15 (27.3%)
GPL-3.0	17 (11.8%)	6 (10.9%)
MIT	6 (4.2%)	4 (7.3%)
GPL-2.0	4 (2.8%)	2 (3.6%)

Table 11 Five most common licenses of GitHub repositories matched in phase 3 containing attributed or unattributed copies of code snippets from Stack Overflow.

SPDX license name	Number of repos containing a SO code snippet clone that was: unattributed ($n = 1,169$)	attributed ($n = 163$)
Apache-2.0	353 (30.2%)	36 (37.4%)
MIT	239 (20.4%)	25 (15.3%)
GPL-3.0	211 (18.0%)	19 (11.7%)
None	153 (13.1%)	61 (37.4%)
GPL-2.0	89 (7.61%)	8 (4.9%)

10 Adherence to Attribution Requirements (RQ3)

Until May 2018, SO defined certain attribution requirements in their terms of service (Stack Exchange Inc, 2018a). The following information was required when content from SO was republished:

1. A visual indication that the content is from SO,
2. a hyperlink directly to the original question,
3. the authors' names for every question and answer,
4. a hyperlink for each author to their profile page on SO.

However, Creative Commons states that one cannot "insist on the exact placement of the attribution credit" (Creative Commons Corporation, 2017b). Thus, it is unclear if the above attribution requirements can actually be enforced by SO. Moreover, Creative Commons points to the fact that altering a CC license through "indirect means", like terms of service, could make the modified license incompatible with the CC license itself. Nevertheless, our goal was to find out to what degree developers adhere to SO's attribution requirements when they refer to SO posts in source code comments (RQ3). As described in the introduction, SO's revised terms of service do not mention the attribution requirements anymore, but they are still linked from the footer of the website (visible for each thread) and from the help page. Regardless of the enforceability of those requirements, the following analysis provides valuable insights into how GH users reference code copied or adapted from SO answers.

10.1 Method:

In the first phase of our research, we identified 2,443 distinct SO answers that were referenced from at least one Java file on GH. We drew a random sample of those answers to investigate how GH users attribute code snippets from SO ($n = 100$). If a URL in the sample had multiple references, we randomly chose one of them.

To determine the *margin of error* for this sample, we first calculated the *standard error* (SE) (Agresti, 2007), assuming that the probability to observe a correct attribution is 50% ($p = 0.5$). For our sample size of 100, this probability yields a standard error of 0.05. In fact, we did not observe a correct attribution in any case (see Section 10.2), thus the actual probability is likely to be much lower. Based on the standard error and a confidence level of 95% ($\alpha = 0.05$), we calculated the margin of error by multiplying the *z-score* (Agresti, 2007; Bartlett et al, 2001; Cochran, 1977): $z(\alpha/2) \cdot SE = 0.10$. Thus, with the above-mentioned assumptions, the margin of error for our estimation of references not adhering to the attribution requirements is 10 percentage points. This means that, even if we did not observe a correct attribution in any of the sampled cases, there could still be up to 10% references adhering to the attribution requirements (confidence level 95%).

We manually extracted the snippets from SO and the referencing code from GH and coded how and where the user attributed the snippet and if he or she just copied, or also adapted, the snippet. We provide the extracted snippets, files, and our coding as supplementary material (Baltes, 2018).

10.2 Results:

Of the 100 referenced answers we analyzed, 12 were conceptual and contained no code suitable for copying and pasting. Three references did not exist anymore when we tried to access the files (file or repository moved or deleted). Most references (89) included only the URL to the answer in a comment, eight references further included the username of the author, e.g.:

```
/**
  * Converts a double to a String in [...]
  * Based on Stack Overflow answer by corsiKa at http://Stack
      Overflow.com/a/5036540 [...]
**/
```

To introduce their reference, most developers (62) used formulations like ‘code from’, ‘based on’, or ‘adapted from’; 35 users only added the SO URL without any further comment. For the majority of references (60), the code had been adapted (e.g., variables renamed). In two of those cases, the comment named an additional source for the copied code beside the SO answer. In about a quarter of cases (22), the code had been copied without any modifications. In two references, the SO answer was only included to show an alternative

solution to a problem. Further, one GH user included a link to advertise his or her own answer on SO.

About half of the references were made in regular source code comments, most of which were placed above the copied snippet (only two were inline comments behind a statement); 41 references were JavaDoc comments for classes, methods, or class variables. It is unclear what SO considers a proper "visual indication" that the content is from SO (required according to the terms of service). Still, only 11 references explicitly mentioned the term 'Stack Overflow' (or other spellings like 'StackOverflow' or 'S.O.') in their comment. Further, none of the comments included a link to the author's profile page, which was also required according to SO's terms of service.

Adherence to Attribution Requirements (RQ3): Most comments referencing code snippets copied or adapted from Stack Overflow included only a link to the corresponding answer without naming the author of the code. No comment included a link to the author's profile page and only 11 out of 97 analyzed comments explicitly named SO as the source. In summary, none of the analyzed references fulfilled the four attribution requirements defined by SO.

11 Developers' Awareness Regarding SO's Licensing (RQ4)

To complement our estimation of unattributed usages of SO code snippets in GH projects, we conducted a second online survey investigating the awareness of GH developers regarding the licensing of SO content. We further used this survey to reveal false positives in our analysis. Moreover, we contacted the authors of the ten most frequently referenced SO Java answers, identified in phase 1 (see Section 5), and asked them about their view on the snippets' licensing situation.

11.1 Method:

For the online survey, we derived a sampling frame from the GH Java repositories that contained at least one file with a clone of the ten most frequently referenced SO Java snippets identified in the first phase of our research (see Section 5). We retrieved the owners for those repositories using our *api-retriever* tool (Baltes, 2017), which utilizes the *GitHub API*. We then filtered the GH users and organizations to only include the ones having a public email address on their profile page. Of the 3,031 email addresses we collected, 2,165 were valid. In a first iteration, we contacted all 211 organizations with valid email addresses and received 20 answers (9.5% response rate). For the second iteration, we removed owners of forked repositories and then contacted 528 developers, receiving 67 responses (12.7% response rate). In both iterations, we informed participants about all matches we found in their repositories and

asked them for one randomly selected match if the code has actually been copied from SO. We provide the questionnaire, the analysis scripts, as well as all closed-ended responses as supplementary material (Baltes, 2018).

To contact the authors of the ten most frequently referenced SO Java answers, we checked their SO profile and searched for their user name on the web. We collected the email address of two authors from their personal website and of five authors from their GH profile, but we were not able to retrieve the email address of four authors. Please note that we have eleven authors in total, because the answer ranked fifth actually pointed to a question and we selected two answers for that question (see Table 1). The email we sent to those authors contained three questions: One asking about their awareness regarding SO's licensing, one asking about an additional source for the snippet, and one asking whether they care about attribution for the particular snippet.

11.2 Results:

In total, 87 users responded to the online survey (11.8% response rate). Beside the survey responses, we received many emails from participants, thanking us that we informed them about the licensing of SO code snippets and in particular about unattributed usages in their projects. One participant, for instance, wrote that his/her team replaced the matched snippet in the repo due to the *share-alike* requirement of SO's license, which they "ignored until [we] called [their] attention." Another participant informed us that the match was in a mirror of the OpenJDK 9 Mercurial repo that was part of the GH repo we analyzed. We informed the OpenJDK team and they replaced the code due to legal concerns. In the corresponding bug description, the author points to possible legal issues and the fact that it is "not a good practice" to copy code from SO (Fazunenko, 2016).

Similar to our preliminary study (see Section 4), the majority of respondents (62%) reported their main software development role to be *software developer*, but there were fewer *software architects* (8%). The average age of the participants who reported their age (n=65) was 30.3 years (SD=9.4) and they had an average programming experience of 11.7 years (SD=8.9). Again, most users answered that they use SO (80%) and GH (61%) for both private and work-related projects; almost one third of them use GH only for private projects (28%).

As mentioned above, we asked participants for one match that we found in their repo whether the code has actually been copied from SO. Of the 74 participants who answered to that question, 43 answered 'Yes' (58%), 20 answered 'No' (27%), and 11 (15%) answered 'I don't know'. Of the 20 participants who answered that the snippet has not been copied from SO, seven claimed they wrote the code themselves, two claimed that a team member wrote it, and 11 answered that they copied it from another source. We manually inspected those matches: Eight of them (10.8%) were indeed relatively short and thus likely to be false positives. To us, the other 12 matches looked

like copies of the SO snippets. Three of them were copies of a SO snippet that was itself a copy of another SO snippet; five matches were also available in external sources like personal blogs (one licensed under the Apache License 2.0, the others were not licensed). Some of the participants who answered that they wrote the code themselves may either not remember copying the code or their answer could be affected by a social desirability bias (Nederhof, 1985). To mitigate the former and to enable tracing the source of code copied from SO, developers should add a comment with a link to SO as motivated in the introduction.

We asked the participants if they knew that SO's license requires them to attribute code copied from posts and in particular if they knew that content on SO is licensed under CC BY-SA 3.0. Regarding the need to attribute content copied from SO, 28 participants (32%) were aware of it, 58 (67%) not, and one preferred not to answer. As to the specific license, the answers were similar: 21 participants (24%) were aware of it, 65 (75%) not, and one preferred not to answer. The attribution requirements from SO's terms of service were even more unfamiliar to the participants: 11 (13%) knew them, 73 (84%) not, and 3 preferred not to answer. Thus, we can conclude that most developers are not aware of the licensing of code published on SO and the implications of this licensing.

With regard to the attribution practice, we asked the same questions as in the preliminary study (see Section 4) and got similar results: Again, not attributing the code when coping from Stack Overflow was a common practice (41%). This time, we asked if respondents referred to the question or a specific answer on SO in case they added a source code comment. Twelve participants preferred not to answer this question, seven named other information they included in the comment. Unlike the results from our quantitative analysis of attributed usages would suggest (see Section 8), participants more frequently reported that they referred to an answer (30%) than to a question (13%). One reason for this could be that many of the references to questions refer to conceptual threads on SO that do not contain code suitable for copying and pasting.

Of the seven contacted SO authors, four answered. Three were not aware of SO's licensing when they posted their answer, one was "vaguely aware". All respondents indicated that they do not know any other source for the code in their answers (except for the ones listed in Table 1). One author answered: "I invented it [the snippet] there and then. I would assume any other source would be a copy from SO." A different author wrote that his answer was "informed by, but not copied directly from, other Stack Overflow posts". Three authors responded that they do not care about attribution for this particular content and one author answered that he does "not really" care. The same author further noted that "it's Stack Overflow that collects the money for the ads. HOWEVER, if the situation would have been the same for an article on [URL removed] which I run myself, I would care deeply about attribution." Another author answered that he does not have the "desire to 'own' the information, only to share it". Those two comments, together with the discussion around

SO's attempt to change the license for code snippets (see Section 3), show that developers have diverse opinions about the attribution requirement. Further investigating the reasons to (not) care about attribution of online code snippets is an interesting direction for future work.

Awareness of Licensing (RQ4): Most developers answering to the online survey were not aware of the licensing of code published on SO and its implications. 75% of the participants did not know that content on SO is licensed under CC BY-SA 3.0 and 67% did not know that attribution is required. Not attributing the code when coping code from SO was a common practice (41%).

12 Limitations and Verifiability

The main limitation of our research is the focus on Java, because the attribution practice may differ between programming languages. Thus, the generalizability of our results to other programming languages is limited. To answer RQ1, we used three different approaches, optimized for precision and always chose conservative estimates. Thus, we do not see the construct validity of our research to be impaired. For the first two phases, we only considered a relatively small sample of snippets compared to all available snippets on SO, but we still found a considerable number of files with copies. The number of attributions was even smaller in the third phase, where we included more snippets and only searched for exact matches.

Another threat to validity is that both the SO snippet as well as the matched code on GH could have a different origin. To mitigate this threat, we analyzed and described all external sources that were linked in the SO answers. In most cases, those sources did not provide a license, thus CC BY-SA 3.0 is the only license which applies. Another possible issue is that if users include a license statement in their snippets on SO, they may allow a more permissive usage without attribution. However, this was only the case in very few of the snippets we manually investigated.

In phase 3 (Section 7), we used the length of the SO snippets as a proxy variable for their originality. However, as mentioned above, there is no "international standard for originality" (Creative Commons Corporation, 2017b) that defines when a code snippet is protected by copyright. Thus, even with the threshold we chose, some of the snippets may not be copyrightable. The survey (Section 11) revealed that 10.8% of the matches in phase 1 were false positives due to their short length. We addressed this issue in phase 3 with a higher threshold for the minimum snippet length.

In phases 2 and 3, we focused on rather popular GitHub repositories to reduce complexity and exclude projects that are not "engineered software projects" (Kalliamvakou et al, 2014; Munaiah et al, 2017). This approach has a very high precision, but also a relatively low recall (Munaiah et al, 2017).

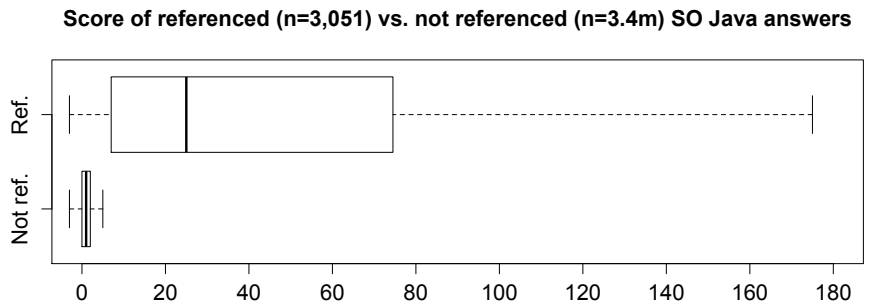

Fig. 8 Scores of Stack Overflow (SO) Java answers referenced in public GitHub (GH) projects compared to scores of Java answers not referenced in GH projects; outliers not depicted; data retrieved from BigQuery GH and SO data sets (11/2017).

Thus, the results of those two phases may only be generalizable to popular projects. Nevertheless, in popular projects the impact of licensing violations is much larger then in small personal projects.

In all phases, we focussed on rather popular SO answers. Thus, our results may not be generalizable to less popular SO answers. Our assumption was that code from unpopular SO answers is less likely to be used in GH projects. To assess this assumption, we utilized the *BigQuery GH* and *SO data sets* to compare the score of SO Java answers referenced in public GH projects to the score of Java answers not referenced on GH (see Figure 8). Referenced Java answers ($Mdn = 25$, $M = 95.33$, $SD = 511.55$) had a significantly higher score than Java answers that were not referenced ($Mdn = 1$, $M = 2.48$, $SD = 16.69$) (Wilcoxon rank sum test, $W = 9,705,800,000$, p-value $< 2.2 \cdot 10^{-16}$).

In phases 1 and 2, we did not check if the code on GH is older than the code on SO, which could indicate that the code has been copied from GH or from another source into the SO post. In phase 3, however, we filtered out matches for which the commit adding the snippet was older than the post on SO, but this was only the case for 10 out of 1,379 matches (0.7%).

To enable other researcher to verify our results, we provide all analysis scripts and data as supplementary material (Baltes, 2018). The supplementary material further includes instructions on how to apply the scripts to the data.

13 Related Work

In the following, we summarize related work from different research areas, highlight connections to our study, and point to directions for future work.

13.1 Stack Overflow and GitHub

Over the past years, there have been various research papers on leveraging knowledge from SO, e.g., to support developers by automating the search (Ponzanelli et al, 2013; Campbell and Treude, 2017) or by augmenting API documentation (Treude and Robillard, 2016). Moreover, different tools have been developed to help developers finding code examples on the web (Zagalsky et al, 2012; Brandt et al, 2010). However, researchers rarely mentioned the complex licensing and copyright situation when building tools to support code reuse from the web, and in particular from SO. Since our study indicates that many developers are not aware of SO's license and its implications (see Section 11), future tools should inform developers about this aspect.

Regarding the populations of SO and GH users, studies described properties such as gender (Vasilescu et al, 2012), age (Morrison and Murphy-Hill, 2013), and geographic location (Schenk and Lungu, 2013). Wang et al. (Wang et al, 2013) analyzed the asking and answering behavior of developers on SO and found that most developers only answer or ask one question and only 8% answer more than 5 questions. Bosu et al. analyzed how reputation is build on SO and provide recommendations for contributors (Bosu et al, 2013). Xia et al. (Xia et al, 2017) found that it is common for developers to search for reusable code snippets on the web, which is in line with Sojer and Henkel's results from an earlier study (Sojer and Henkel, 2011): In 2009, they conducted an online survey with 869 software developers to investigate ad-hoc reuse of "internet code" (Sojer and Henkel, 2011). Even at that time, about one year after SO's launch, reuse of such code from internet sources was an essential part of developers' work. Our study has shown that it is common for developers to copy and paste code from SO into their projects without providing the required attribution. Moreover, we found that developers are not aware of SO's license. An interesting direction for future work would be to analyze if developers' usage of code snippets from the web, particularly from SO, decreases with an increase of awareness and knowledge about when code is copyrightable and which implications certain licenses have.

Regarding code snippets on SO, Yang et al. (Yang et al, 2016) found that Python and JavaScript snippets are more usable in terms of parsability, compilability and runnability, compared to Java and C#. Yang et al. (Yang et al, 2017) analyzed code clones between Python snippets from SO and Python projects on GH using a token-based clone detector and found a considerable number of non-trivial clones. Abdalkareem et al. found that reusing code from SO may have a negative impact on code quality (Abdalkareem et al, 2017). Other studies aimed at identifying API usage in SO code snippets (Subramanian and Holmes, 2013), describing characteristics of effective code examples (Nasehi et al, 2012), investigating whether SO code snippets are self-explanatory (Treude and Robillard, 2017), or analyzing the impact of copied SO code snippet on application security (Acar et al, 2016; Fischer et al, 2017). Recently, Zhang et al. analyzed potential API usage violations in SO posts and found that, of the 217,818 analyzed Java and Android SO posts, 31% may

contain potential API usage violations, which could lead to program crashes or resource leaks (Zhang et al, 2018).

There has also been work on the interplay between user activity on SO and GH (Vasilescu et al, 2013; Silvestri et al, 2015; Badashian et al, 2014). In particular, Vascilescu et al. (Vasilescu et al, 2013) showed that active GH committers ask fewer questions and provide more answers than others. With our study, we add a new aspect to this interplay, namely how code from SO is used and attributed in GH projects.

To describe the topics of SO questions and answers, different methods like manual analysis (Treude et al, 2011) and Latent Dirichlet Allocation (LDA) (Wang et al, 2013; Allamanis and Sutton, 2013) have been used. Automatically identifying high-quality questions and answers has been another research direction, where metrics based on the number of edits on a question (Yang et al, 2014), the author's popularity (Ponzanelli et al, 2014), and code readability (Duijn et al, 2015) yielded good results. A direction for future work is to investigate whether those high-quality questions and answers are actually referenced or used more often in GH projects.

13.2 Licensing and Code Clones

German and Hassan (German and Hassan, 2009) point to the license mismatch problem, that is combining software components with possibly conflicting licenses. As described above, such a license conflict may arise when developers copy non-trivial code snippets from SO into their projects, because SO's license requires derivative work to use a compatible license. An et al. (An et al, 2017) investigated whether developers respect license terms when reusing code from SO posts in a sample of 399 Android projects and found many potential license violations. They considered a project to violate SO's license if it, among other factors, didn't "use the CC BY-SA 3.0 or its later versions." However, they did not consider the compatibility of CC BY-SA 4.0 and GPL 3.0 (see Section 9). Moreover, none of the files they analyzed contained a reference "to the corresponding Stack Overflow post." It is unclear if the authors also considered links to the corresponding question. Nevertheless, these results do not contradict our estimation that in at most one quarter of the cases, code copied from SO is attributed as required (see Section 14).

A reason for such license violations may be developers struggling to understand the interaction of open source licenses. Almeida et al. conducted an online survey with 375 software developers and found that developers struggled to understand licensing scenarios involving multiple licenses (Almeida et al, 2018), as it may be the case when developers want to use SO code in their projects. As motivated above, the situation can even be more complex when code on SO is also available on other websites. SO could address this issue by making the licensing of the content more visible on their website, and by integrating a feature that allows SO authors to easily provide an additional (more permissive) license when posting code on SO.

German et al. (German et al, 2009) analyzed how code siblings, i.e., code clones that evolve in a different system than the original code, flow between systems with different licenses; Gharehyazie et al. (Gharehyazie et al, 2017) and Lopes et al. (Lopes et al, 2017) found that cross-project code reuse on GH is common. Tracing the flow of siblings between GH projects, posts on SO, and external sources is another possible direction for future work.

Two fields related to our study are source code plagiarism detection (Lancaster and Culwin, 2004) and code clone detection (Roy et al, 2009), which both rely on determining the similarity of code fragments. One of the most often cited tools for code plagiarism detection is *JPlag* (Prechelt et al, 2002; Burrows et al, 2007), which uses the same algorithm to determine token string similarity like *CPD* (Martins et al, 2014), the code clone detector we used in the second phase of our study. There has been recent work on scaling the detection of code clones to large source code corpora (Ragkhitwetsagul, 2016; Sajnani et al, 2016; Burrows et al, 2007) that we can build upon to be able to search for copies of all non-trivial SO code snippets in all public GH projects.

14 Conclusion

Our main goal was to quantify the amount of unattributed usages of code snippets from Stack Overflow (SO) in GitHub (GH) projects. In a preliminary survey, half of the participants answered that they did not attribute snippets copied from SO. However, our quantitative analysis shows that, for Java, at most a quarter of the usages of SO snippets are attributed. We used three different approaches to find unattributed usages, always chose conservative estimates, and tried to remove as many false positive results as possible. In the first phase, we searched for unattributed usages of the snippets from the ten most frequently referenced SO Java answers in all Java files in the *BigQuery GH data set* and found that only 23% of the copies had been attributed. In the second phase, we utilized the token-based code clone detector CPD to find clones of a sample of 222 SO Java snippets in a sample of 2,313 popular GH Java projects and found that only 24% of the snippet clones included a reference to SO. In the last phase, we searched for exact copies of 29,370 SO Java snippets in 64,281 GH projects and found that only 8% of the copies were attributed. Thus, we think that one quarter is a reasonable upper bound for the ratio of attributed usages. The higher ratio in the preliminary survey could be explained with a social desirability bias (Nederhof, 1985) affecting the respondents.

Our preliminary survey yielded that, if content from SO is attributed, developers usually add a link to the question or answer in a source code comment. We analyzed how often these URLs are present in Java files and found that developers more often refer to questions, i.e., the whole thread, than to specific answers. Adding a reference to a specific answer instead of the question could help to increase maintainability. For example, one could later on check whether this answer is still the accepted one or whether a bug fix has been

posted. However, there may be cases when the question is more appropriate, e.g., when a developer wants to refer to a controversially discussed topic or a conceptual issue. Analyzing when developers link to questions and when to answers is a direction for future work.

In the three phases of our research, between 3.3% and 11.9% of the analyzed repositories contained a file with a reference to SO. The popular projects from phase two were more likely to contain a reference than the broader samples of phases 1 and 3. Depending on the project's license, the *share-alike* requirement of CC BY-SA 3.0 may lead to licensing issues for those projects. Our second survey has shown that many developers admit copying code from SO without attribution and are not aware of the licensing and its implications. Moreover, we found that at most 1.8% of the GH projects with copies of SO code snippets attributed the copy and had a license that would allow a CC BY-SA 3.0-compatible usage of the SO content. The discussions on SO about a new code license show that developers care about this topic, yet many developers do not attribute code they copy from SO posts. A direction for future research is to investigate this dichotomy.

The next steps of our research are to automate and scale the extraction of copyable snippets form SO and the detection of unattributed usages in GH projects. The 'reverse engineering' of the missing link to SO can help developers mitigating possible *maintenance* and *legal issues*, as motivated in the introduction. Further, using SO's official data dump, we build a data set with the extracted version history of all SO code snippets (Baltes et al, 2018; Baltes and Dumani, 2018). We plan to use this data set to identify buggy revisions, and then search for copies of those revisions to warn developers who copied buggy code. We also want to expand our analysis to other programming languages and further investigate the relations between code snippets on SO, their copies on GH, and external sources.

Acknowledgements The authors would like to thank the participants of the online surveys, the anonymous reviewers, and Bernhard Baltes-Götz for their valuable feedback. Moreover, we thank Richard Kiefer for his help with the calibration of CPD and the extraction of the snippet sets and Florian Reitz for his help with database-related issues.

References

Abdalkareem R, Shihab E, Rilling J (2017) On code reuse from StackOverflow: An exploratory study on Android apps. Information and Software Technology 88:148–158

Acar Y, Backes M, Fahl S, Kim D, Mazurek ML, Stransky C (2016) You Get Where You're Looking For: The Impact Of Information Sources on Code Security. In: Locasto M, Shmatikov V, Erlingsson Ú (eds) 2016 IEEE Symposium on Security and Privacy (S&P 2016), IEEE Computer Society, San Jose, CA, USA, pp 289–305

Achte Zivilkammer (2016) AZ I-8 O 294/15. Landgericht Bochum URL http://www.justiz.nrw.de/nrwe/lgs/bochum/lg_bochum/j2016/I_8_O_294_15_Urteil_20160303.html

Agresti A (2007) An Introduction to Categorical Data Analysis, 2nd edn. John Wiley & Sons, Hoboken, NJ, USA

Aioobe (2010) How to convert byte size into human readable format in java? URL http://stackoverflow.com/a/3758880

Allamanis M, Sutton C (2013) Why, when, and what: Analyzing Stack Overflow questions by topic, type, and code. In: Zimmermann T, Di Penta M, Kim S (eds) 10th International Working Conference on Mining Software Repositories (MSR 2013), IEEE, San Francisco, CA, USA, pp 53–56

Almeida DA, Murphy GC, Wilson G, Hoye M (2018) Investigating whether and how software developers understand open source software licensing. Empirical Software Engineering 11(11):730

Alsup W (2012) Oracle America, Inc v. Google, Inc. United States District Court for the Northern District of California

An L, Mlouki O, Khomh F, Antoniol G (2017) Stack Overflow: A Code Laundering Platform? In: Pinzger M, Bavota G, Marcus A (eds) 24th IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER 2017), IEEE Computer Society, Klagenfurt, Austria, pp 283–293

Badashian AS, Esteki A, Gholipour A, Hindle A, Stroulia E (2014) Involvement, Contribution and Influence in GitHub and Stack Overflow. In: Ng J, Li J, Wong K (eds) 24th International Conference on Computer Science and Software Engineering (CASCON 2014), IBM / ACM, Markham, ON, Canada, pp 19–33

Baltes S (2017) sbaltes/api-retriever on GitHub. URL https://doi.org/10.5281/zenodo.1049419

Baltes S (2018) Usage and Attribution of Stack Overflow Code Snippets in GitHub Projects — Supplementary Material. URL https://doi.org/10.5281/zenodo.1148069

Baltes S, Dumani L (2018) SOTorrent Dataset. URL http://doi.org/10.5281/zenodo.1135262

Baltes S, Dumani L, Treude C, Diehl S (2018) SOTorrent: Reconstructing and Analyzing the Evolution Stack Overflow Posts. In: Zaidman A, Hill E, Kamei Y (eds) 15th International Conference on Mining Software Repositories (MSR 2018), ACM, Gothenburg, Sweden, pp 1–12

Bartlett JE II, Kotrlik JW, Higgins CC (2001) Organizational Research: Determining Appropriate Sample Size in Survey Research. Information Technology, Learning, and Performance Journal 19(1):43–50

Bosu A, Corley CS, Heaton D, Chatterji D, Carver JC, Kraft NA (2013) Building reputation in StackOverflow: An empirical investigation. In: Zimmermann T, Di Penta M, Kim S (eds) 10th International Working Conference on Mining Software Repositories (MSR 2013), IEEE, San Francisco, CA, USA, pp 89–92

Brandt J, Dontcheva M, Weskamp M, Klemmer SR (2010) Example-centric programming: Integrating web search into the development environment. In:

Mynatt E, Edwards K, Rodden T (eds) 2010 Conference on Human Factors in Computing Systems (CHI 2010), ACM, Atlanta, GA, USA, pp 513–522

Burrows S, Tahaghoghi SMM, Zobel J (2007) Efficient plagiarism detection for large code repositories. Software—Practice and Experience 37(2):151–176

Campbell BA, Treude C (2017) NLP2Code: Code Snippet Content Assist via Natural Language Tasks. In: Mei H, Zhang L, Zimmermann T (eds) 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME 2017), IEEE Computer Society, Shanghai, China, pp 628–632

Cavaretta MJ (2015) Open Source Issues in Mergers & Acquisitions. URL http://www.mbbp.com/news/open-source-issues

Cochran WG (1977) Sampling Techniques, 3rd edn. John Wiley & Sons, Hoboken, NJ, USA

Corley JS (2017) Artifex Software, Inc v. Hancom, Inc. United States District Court for the Northern District of California

Creative Commons Corporation (2007) Attribution-ShareAlike 3.0 Unported. URL https://creativecommons.org/licenses/by-sa/3.0/legalcode

Creative Commons Corporation (2017a) Compatible Licenses. URL https://creativecommons.org/share-your-work/licensing-considerations/compatible-licenses/

Creative Commons Corporation (2017b) Frequently Asked Questions. URL https://creativecommons.org/faq/#can-i-apply-a-creative-commons-license-to-software

Duijn M, Kucera A, Bacchelli A (2015) Quality Questions Need Quality Code: Classifying Code Fragments on Stack Overflow. In: Di Penta M, Pinzger M, Robbes R (eds) 12th Working Conference on Mining Software Repositories (MSR 2015), IEEE Computer Society, Florence, Italy, pp 410–413

Electronic Frontier Foundation (2018) Oracle v. Google. URL https://www.eff.org/cases/oracle-v-google

Engelfriet A (2016) What is the license status of StackOverflow code snippets? URL https://legalict.com/software/what-is-the-license-status-of-stackoverflow-code-snippets/

Fazunenko D (2016) Get rid of the humanReadableByteCount() method in openjdk/hotspot. URL https://bugs.openjdk.java.net/browse/JDK-8170860

Fischer F, Böttinger K, Xiao H, Stransky C, Acar Y, Backes M, Fahl S (2017) Stack Overflow Considered Harmful? The Impact of Copy&Paste on Android Application Security. In: Butler KRB, Erlingsson Ú, Parno B (eds) 2017 IEEE Symposium on Security and Privacy (S&P 2017), IEEE Computer Society, San Jose, CA, USA, pp 121–136

German DM, Hassan AE (2009) License integration patterns: Addressing license mismatches in component-based development. In: Fickas S, Atlee JM, Inverardi P (eds) 31st International Conference on Software Engineering (ICSE 2009), IEEE Computer Society, Vancouver, BC, Canada, pp 188–198

German DM, Di Penta M, Gueheneuc YG, Antoniol G (2009) Code siblings: Technical and legal implications of copying code between applications. In: Godfrey MW, Whitehead J (eds) 6th International Working Conference on

Mining Software Repositories (MSR 2009), IEEE Computer Society, Vancouver, BC, Canada, pp 81–90

Gharehyazie M, Ray B, Filkov V (2017) Some From Here, Some From There: Cross-Project Code Reuse in GitHub. In: Gonzalez-Barahona JM, Hindle A, Tan L (eds) 14th International Conference on Mining Software Repositories (MSR 2017), IEEE Computer Society, Buenos Aires, Argentina, pp 291–301

GitHub Inc (2017a) Choosealicense.com: No License. URL `https://choosealicense.com/no-license/`

GitHub Inc (2017b) GitHub Developer – API. URL `https://developer.github.com/v3/`

GitHub Inc (2018) The State of the Octoverse 2017. URL `https://octoverse.github.com/`

Google Cloud Platform (2017a) GitHub Data. URL `https://cloud.google.com/bigquery/public-data/github`

Google Cloud Platform (2017b) Stack Overflow Data. URL `https://cloud.google.com/bigquery/public-data/stackoverflow`

Gousios G (2013) The GHTorrent dataset and tool suite. In: Zimmermann T, Di Penta M, Kim S (eds) 10th International Working Conference on Mining Software Repositories (MSR 2013), IEEE, San Francisco, CA, USA, pp 233–236

Gousios G (2017) GHTorrent on the Google cloud. URL `http://ghtorrent.org/gcloud.html`

Kaess J, Müller J, Rieger J (2004) Welte v. Sitecom Deutschland GmbH. District Court of Munich I

Kalliamvakou E, Gousios G, Blincoe K, Singer L, Germán DM, Damian D (2014) The promises and perils of mining GitHub. In: Devanbu PT, Kim S, Pinzger M (eds) 11th Working Conference on Mining Software Repositories (MSR 2014), ACM, Hyderabad, India, pp 92–101

Lancaster T, Culwin F (2004) A comparison of source code plagiarism detection engines. Computer Science Education 14(2):101–112

Lopes CV, Maj P, Martins P, Saini V, Di Yang, Zitny J, Sajnani H, Vitek J (2017) DéJàVu: A Map of Code Duplicates on GitHub. Proc ACM Program Lang 1(OOPSLA):84:1–84:28

Martins VT, Fonte D, Henriques PR, Cruz Dd (2014) Plagiarism Detection: A Tool Survey and Comparison. In: Pereira MJV, Leal JP, Simoes A (eds) 3rd Symposium on Languages, Applications and Technologies (SLATE 2014), Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Bragança, Portugal, OpenAccess Series in Informatics (OASIcs), vol 38, pp 143–158

Meloca R, Pinto G, Baiser L, Mattos M, Polato I, Wiese IS, German D (2018) Understanding the Usage, Impact, and Adoption of Non-OSI Approved Licenses. In: Zaidman A, Hill E, Kamei Y (eds) 15th International Conference on Mining Software Repositories (MSR 2018), ACM, Gothenburg, Sweden, pp 1–11

Morrison P, Murphy-Hill E (2013) Is programming knowledge related to age? An exploration of Stack Overflow. In: Zimmermann T, Di Penta M, Kim S (eds) 10th International Working Conference on Mining Software Reposito-

ries (MSR 2013), IEEE, San Francisco, CA, USA, pp 69–72
Munaiah N, Kroh S, Cabrey C, Nagappan M (2017) Curating GitHub for engineered software projects. Empirical Software Engineering 22(6):3219–3253
Nasehi SM, Sillito J, Maurer F, Burns C (2012) What makes a good code example? A study of programming Q&A in StackOverflow. In: Tonella P, Di Penta M, Maletic JI (eds) 28th IEEE International Conference on Software Maintenance (ICSM 2012), IEEE Computer Society, Trento, Italy, pp 25–34
Nederhof AJ (1985) Methods of coping with social desirability bias: A review. European Journal of Social Psychology 15(3):263–280
PMD (2016) Finding duplicated code. URL `http://pmd.github.io/pmd-5.5.1/usage/cpd-usage.html`
Ponzanelli L, Bacchelli A, Lanza M (2013) Seahawk: Stack Overflow in the IDE. In: Notkin D, Cheng BHC, Pohl K (eds) 35th International Conference on Software Engineering (ICSE 2013), IEEE Computer Society, San Francisco, CA, USA, pp 1295–1298
Ponzanelli L, Mocci A, Bacchelli A, Lanza M (2014) Understanding and classifying the quality of technical forum questions. In: Wong WE, McMillin B (eds) 14th International Conference on Quality Software (QSIC 2014), IEEE, Allen, TX, USA, pp 343–352
Poteat H (2016) GitHub's 2015 Transparency Report. URL `https://github.com/blog/2202-github-s-2015-transparency-report`
Prechelt L, Malpohl G, Philippsen M (2002) Finding plagiarisms among a set of programs with JPlag. Journal of Universal Computer Science 8(11):1016–1038
Ragkhitwetsagul C (2016) Measuring Code Similarity in Large-Scaled Code Corpora. In: Kraft NA, Menzies T, Adams B, Poshyvanyk D (eds) 2016 IEEE International Conference on Software Maintenance and Evolution (ICSME 2016), IEEE Computer Society, Raleigh, NC, USA, pp 626–630
Roy CK, Cordy JR, Koschke R (2009) Comparison and evaluation of code clone detection techniques and tools: A qualitative approach. Science of Computer Programming 74(7):470–495
Sajnani H, Saini V, Svajlenko J, Roy CK, Lopes CV (2016) SourcererCC: Scaling code clone detection to big-code. In: Dillon L, Visser W, Williams L (eds) 38th International Conference on Software Engineering (ICSE 2016), ACM, Austin, TX, USA, pp 1157–1168
Scalabrino S, Bavota G, Vendome C, Linares-Vásquez M, Poshyvany D, Oliveto R (2017) Automatically Assessing Code Understandability: How Far Are We? In: Grigore Rosu, Massimiliano Di Penta, Tien N Nguyen (eds) 32nd IEEE/ACM International Conference on Automated Software Engineering (ASE 2017), IEEE Computer Society, Urbana, IL, USA, pp 417–427
Schenk D, Lungu M (2013) Geo-locating the knowledge transfer in StackOverflow. In: Ali R, Begel A, Maalej W (eds) 2013 International Workshop on Social Software Engineering (SSE 2013), ACM, Saint Petersburg, Russian Federation, pp 21–24

Silvestri G, Yang J, Bozzon A, Tagarelli A (2015) Linking Accounts across Social Networks: The Case of StackOverflow, GitHub and Twitter. In: Armano G, Bozzon A, Giuliani A (eds) 1st International Workshop on Knowledge Discovery on the WEB (KDWeb 2015), CEUR-WS.org, Cagliari, Italy, CEUR Workshop Proceedings, pp 41–52

Software Freedom Law Center (2008) Free Software Foundation, Inc v. Cisco Systems, Inc. United States District Court for the Southern District of New York

Sojer M, Henkel J (2011) License Risks from Ad Hoc Reuse of Code from the Internet. Communications of the ACM 54(12):74–81

St Laurent AM (2004) Understanding Open Source and Free Software Licensing. O'Reilly Media

Stack Exchange Inc (2015) Stack Exchange Data Dump: August 18, 2015. URL https://archive.org/details/stackexchange/

Stack Exchange Inc (2016) Stack Exchange API v2.2. URL https://api.stackexchange.com/docs

Stack Exchange Inc (2017a) Stack Exchange Data Dump 2017-12-01. URL https://archive.org/details/stackexchange/

Stack Exchange Inc (2017b) Stack Exchange Data Dump: March 14, 2017. URL https://archive.org/details/stackexchange/

Stack Exchange Inc (2018a) Stack Exchange Network Terms of Service. URL https://web.archive.org/web/20180228075555/http://stackexchange.com/legal

Stack Exchange Inc (2018b) Stack Exchange Network Terms of Service. URL http://stackexchange.com/legal

Stack Exchange Meta (2009) What is up with the source code license on Stack Overflow? URL http://meta.stackexchange.com/q/25956

Stack Exchange Meta (2013) Do I have to worry about copyright issues for code posted on Stack Overflow? URL http://meta.stackexchange.com/q/12527

Stack Exchange Meta (2015) Can we get some explicit clarification on the *intended* legal usage of code from SO answers? URL http://meta.stackoverflow.com/q/286582

Stack Exchange Meta (2016) A New Code License: The MIT, this time with Attribution Required. URL http://meta.stackexchange.com/q/272956

Subramanian S, Holmes R (2013) Making sense of online code snippets. In: Zimmermann T, Di Penta M, Kim S (eds) 10th International Working Conference on Mining Software Repositories (MSR 2013), IEEE, San Francisco, CA, USA, pp 85–88

Tim Post (2018) A new (2018) update to our Terms of Service is here. URL https://meta.stackexchange.com/questions/309746/a-new-2018-update-to-our-terms-of-service-is-here

TIOBE software BV (2017) TIOBE Index for February 2017. URL http://www.tiobe.com/tiobe-index/

Treude C, Robillard MP (2016) Augmenting API Documentation with Insights from Stack Overflow. In: Dillon L, Visser W, Williams L (eds) 38th Inter-

national Conference on Software Engineering (ICSE 2016), ACM, Austin, TX, USA, pp 392–403

Treude C, Robillard MP (2017) Understanding Stack Overflow Code Fragments. In: Mei H, Zhang L, Zimmermann T (eds) 2017 IEEE International Conference on Software Maintenance and Evolution (ICSME 2017), IEEE Computer Society, Shanghai, China, pp 509–513

Treude C, Barzilay O, Storey MAD (2011) How do programmers ask and answer questions on the web? In: Taylor RN, Gall HC, Medvidovic N (eds) 33rd International Conference on Software Engineering (ICSE 2011), ACM, Waikiki, Honolulu, pp 804–807

Vasilescu B, Capiluppi A, Serebrenik A (2012) Gender, Representation and Online Participation: A Quantitative Study of StackOverflow. In: Aberer K, Flache A, Jager W, Liu L, Tang J, Gueret C (eds) 4th International Conference on Social Informatics (SocInfo 2012), Springer, Lausanne, Switzerland, Lecture Notes in Computer Science, pp 332–338

Vasilescu B, Filkov V, Serebrenik A (2013) StackOverflow and GitHub: Associations between Software Development and Crowdsourced Knowledge. In: Chang LW, Srivastava J, Zhan J (eds) 2013 International Conference on Social Computing (SocialCom 2013), IEEE Computer Society, Washington, DC, USA, pp 188–195

Vendome C (2015) A large scale study of license usage on GitHub. In: Bertolino A, Canfora G, Elbaum S (eds) 37th International Conference on Software Engineering (ICSE 2015), IEEE, Florence, Italy, pp 772–774

Wang S, Lo David, Jiang L (2013) An empirical study on developer interactions in StackOverflow. In: Shin SY, Maldonado JC (eds) 28th Annual ACM Symposium on Applied Computing (SAC 2013), ACM, Coimbra, Portugal, pp 1019–1024

White JS (2008) Jacobsen v. Katzer, 535 F.3d 1373, 1379. United States Court of Appeals for the Federal Circuit

Wikipedia (2017) Free Software Foundation, Inc v. Cisco Systems, Inc. URL `https://en.wikipedia.org/wiki/Free_Software_Foundation,_Inc._v._Cisco_Systems,_Inc.`

Xia X, Bao L, Lo D, Kochhar PS, Hassan AE, Xing Z (2017) What do developers search for on the web? Empirical Software Engineering 22(6):3149–3185

Yang D, Hussain A, Lopes CV (2016) From Query to Usable Code: An Analysis of Stack Overflow Code Snippets. In: Kim M, Robbes R, Bird C (eds) 13th International Conference on Mining Software Repositories (MSR 2016), ACM, Austin, TX, USA, pp 391–402

Yang D, Martins P, Saini V, Lopes CV (2017) Stack Overflow in Github: Any Snippets There? In: Gonzalez-Barahona JM, Hindle A, Tan L (eds) 14th International Conference on Mining Software Repositories (MSR 2017), IEEE Computer Society, Buenos Aires, Argentina, pp 280–290

Yang J, Hauff C, Bozzon A, Houben GJ (2014) Asking the right question in collaborative Q&A systems. In: Ferres L, Rossi G, Almeida VAF, Herder E (eds) 25th ACM Conference on Hypertext and Social Media (HT 2014), ACM, Santiago, Chile, pp 179–189

Zagalsky A, Barzilay O, Yehudai A (2012) Example Overflow: Using social media for code recommendation. In: Maalej W, Robillard MP, Walker RJ, Zimmermann T (eds) 3rd International Workshop on Recommendation Systems for Software Engineering (RSSE 2012), IEEE, Zurich, Switzerland, pp 38–42

Zhang T, Upadhyaya G, Reinhardt A, Rajan H, Kim M (2018) Are Code Examples on an Online Q&A Forum Reliable? A Study of API Misuse on Stack Overflow. In: Crnkovic I, Chechik M, Harman M (eds) 40th International Conference on Software Engineering (ICSE 2018), ACM, Gothenburg, Sweden, pp 1–11