Empirical Software Engineering (2021) 26:77
https://doi.org/10.1007/s10664-021-09960-w

Assessment of off-the-shelf SE-specific sentiment analysis tools: An extended replication study

Nicole Novielli[1] · Fabio Calefato[1] · Filippo Lanubile[1] · Alexander Serebrenik[2]

Accepted: 5 March 2021 Published online: 05 June 2021

Abstract

Sentiment analysis methods have become popular for investigating human communication, including discussions related to software projects. Since general-purpose sentiment analysis tools do not fit well with the information exchanged by software developers, new tools, specific for software engineering (SE), have been developed. We investigate to what extent off-the-shelf SE-specific tools for sentiment analysis mitigate the threats to conclusion validity of empirical studies in software engineering, highlighted by previous research. First, we replicate two studies addressing the role of sentiment in security discussions on GitHub and in question-writing on Stack Overflow. Then, we extend the previous studies by assessing to what extent the tools agree with each other and with the manual annotation on a gold standard of 600 documents. We find that different SE-specific sentiment analysis tools might lead to contradictory results at a fine-grain level, when used *off-the-shelf*. Conversely, platform-specific tuning or retraining might be needed to take into account differences in platform conventions, jargon, or document lengths.

Keywords Sentiment analysis · Human aspects of software engineering · Replication study

Communicated by: Meiyappan Nagappan

✉ Nicole Novielli
nicole.novielli@uniba.it

Fabio Calefato
fabio.calefato@uniba.it

Filippo Lanubile
filippo.lanubile@uniba.it

Alexander Serebrenik
a.serebrenik@tue.nl

1 Dipartimento di Informatica, University of Bari Aldo Moro, via E. Orabona, 4, 70125, Bari, Italy

2 Mathematics and Computer Science, Eindhoven University of Technology, P.O. Box 513, 5600, MB, Eindhoven, The Netherlands

1 Introduction

Sentiment analysis, i.e., the task of extracting the positive or negative semantic orientation a text (Pang and Lee 2008), has emerged as a tool for empirical software engineering studies to mine emotions and opinions from textual developer-generated content (Novielli et al. 2019) in the 'social programmer' (Storey 2012) ecosystem. Its popularity is also due to the availability of a plethora of sentiment analysis tools released for public use.

With a few notable exceptions (Blaz and Becker 2016; Ortu et al. 2016; Panichella et al. 2015), early empirical studies in this field have exploited off-the-shelf general-purpose sentiment analysis tools. However, general-purpose sentiment analysis tools tend to produce unreliable results in the software engineering context as they are trained on documents from non-technical domains, such as film critics and product reviews. Specifically, Jongeling et al. (2017) tried to replicate previously published empirical studies and showed that the choice of the sentiment analysis tool has an impact on the validity of the conclusions.

To overcome such limitations, researchers have started developing SE-specific sentiment analysis tools to mine developers' emotions (e.g., Calefato et al. (2018a), Ahmed et al. (2017), Islam and Zibran (2017), and Chen et al. (2019)) and opinions (e.g., (Lin et al. 2019; Uddin and Khomh 2017)). In previous benchmarking studies, (Novielli et al. 2018b; 2020) showed how SE-specific customization gives a boost in accuracy in terms of both agreement with manual annotation and agreement among tools, provided that model-based annotation of emotions is implemented and that a robust gold-standard dataset for retraining is available.

In this paper, we go beyond the simple assessment of performance of SE-specific tools and their agreement, which is the main goal of the aforementioned benchmarking studies. Here, we investigate to what extent the SE-specific retraining operated by the original authors of sentiment analysis tools enables researchers to address the threat to conclusion validity induced by the use of general-purpose tools (Jongeling et al. 2017). Specifically, we aim at understanding if we can reliably and safely reuse SE-specific tools *off-the-shelf*, i.e., without further tuning or training beyond that performed by the original authors on technical texts.

As such, we formulate our first research question as follows

– *RQ1: Does the choice of the sentiment analysis tool introduce a threat to conclusion validity in a software engineering study?*

To address RQ1, we replicate two previously published studies on sentiment analysis in software engineering. We decided to perform exact, dependent replications (Shull et al. 2008) to mitigate threats to validity that are inherent in replications. As such, we select two studies in which a subset of the authors of the current paper were involved as co-authors. By doing so, we are able to replicate the original study design, use the same dataset as the original studies, and change only the sentiment analysis tools used to mine the developers' sentiment. Specifically, we replicate the analysis of sentiment in GitHub security discussion by Pletea et al. (2014) and the study of the role of emotions in effective question-writing in Stack Overflow by Calefato et al. (2018b). First, we replicate each study with four SE-specific tools and compare the conclusions we obtain. Then, we compare the findings obtained with SE-specific tools with those previously published in the original studies to verify whether they still hold.

Other than suggesting a threat to conclusion validity, Jongeling et al. showed that general-purpose sentiment analysis tools also disagree with human annotation and with each other.

To get further insights in this direction, but this time focusing on SE-specific sentiment analysis tools, we inherit and revise two additional research questions from Jongeling et al. (2017):

- *RQ2: To what extent do results from different SE-specific sentiment analysis tools agree with each other?*
- *RQ3: To what extent do different SE-specific sentiment analysis tools agree with human raters?*

We address RQ2 by measuring the agreement between the SE-specific tools themselves, based on the manual annotation of a subset of 600 documents randomly selected from the original datasets. We also use the obtained gold standard to address RQ3 by computing the agreement of SE-specific tools with the manual labels. We release the annotated corpus[1] as an additional contribution of the study to encourage further studies on emotion detection in software engineering.

This paper enhances the state of the art on sentiment analysis in empirical studies in software engineering by furthering our understanding of *off-the-shelf* tool reuse, even in presence of SE-specific tuning. We show how the choice of the specific sentiment analysis tools might lead to contradictory results, thus representing a threat to conclusion validity, when the analysis is at a fine-grained level. Our results suggest that SE-specific fine-tuning of sentiment analysis tools to the software engineering domain might not be enough to improve accuracy, and platform-specific tuning or retraining might be needed to adjust the model performance to the shifts in lexical semantics due to different platform jargon or conventions. Further fine-tuning should be implemented in order to adjust the sentiment operationalization in line with the specific research goals.

The remainder of the paper is organized as follows. In Section 2, we review the related work and describe the SE-specific sentiment analysis tools that we employ in our study. In Section 3, we summarize the two studies that we replicate, including their goals, settings, and results. We report the results of our replications in Section 4 and the results of the analysis of the agreement between tools and with the manual labels in Sections 5 and 6, respectively. In Section 7, we discuss our findings while in Section 8 we assess the main limitations of the current study. Finally, we provide conclusions in Section 9.

2 Background

2.1 Related work

In recent years, a trend has emerged to use sentiment analysis as a new tool for empirical studies in software engineering research. Other than the studies by Pletea et al. (2014) and Calefato et al. (2018b) that we replicate in this paper, several others have applied sentiment analysis to mine emotions of software developers from their communication traces. In particular, researchers investigated the role of affect in social software engineering by applying sentiment analysis to the content available in collaborative software development platforms such as GitHub (Guzman et al. 2014; Sinha et al. 2016) and Jira (Ortu et al. 2016; Mäntylä et al. 2016; Murgia et al. 2018). In the scope of requirements engineering research, sentiment analysis has been also leveraged to mine users' opinions about software products from

[1] https://figshare.com/s/faecffd83e65155ebc2a

their reviews in the app stores (Panichella et al. 2015; Kurtanovic and Maalej 2018), from user-generated contents in microblogging platforms (Guzman et al. 2016) and customers' tickets (Werner et al. 2018).

In spite of the popularity of the topic, only a few papers report about replicated studies on sentiment analysis in software engineering. Our study builds upon previous research by (Jongeling et al. 2017) who compared the predictions of general-purpose, off-the-shelf sentiment analysis tools, showing that not only they disagree with human annotation, but also with each other. Given the disagreement among these tools, Jongeling and colleagues conducted a replication of previous studies on sentiment analysis in software engineering to understand to what extent the choice of an instrument affects the results. They observed contradictory findings and concluded that previous studies' results cannot be replicated when different, general-purpose sentiment analysis tools are used, i.e., the instrument choice can induce threats to conclusion validity.

Novielli et al. (2018b) investigated to what extent different SE-specific sentiment analysis tools agree with gold standard annotations of developers' emotions and with each other. To this aim, they performed a benchmarking study aimed at assessing the performance of three SE-specific sentiment analysis tools on four gold standard datasets including developers' communication traces from collaborative platforms, such as Stack Overflow and Jira. The performance of each tool was assessed on a held-out test set extracted from the gold standard, which was also used to build the baseline performance represented by a general-purpose tool. The findings of Novielli et al. indicate that reliable sentiment analysis in software engineering is possible, provided that manual annotation of gold standards is inspired by theoretical models of affect. Furthermore, they found that, regardless of the approach adopted for annotation, SE-specific customization/retraining does provide a boost in accuracy with respect to the baseline approach represented by SentiStrength (Thelwall et al. 2012), an off-the-shelf general-purpose tool. Based on the findings of their qualitative error analysis, Novielli et al. suggest that we should be aware that tools and datasets are built by having in mind specific research goals and different conceptualization of affect. Thus, a sanity check is always recommended to assess the suitability of existing tools with respect to the research goals.

Mäntylä et al. (2017) replicated a previous study on the identification of burnout and productivity of software developers (Mäntylä et al. 2016). The original study was performed by using a general-purpose lexicon of 14,000 English words with known emotion scores, including the arousal associated to each word. In the replication study the authors built a Software Engineering Arousal lexicon (SEA) to correctly measure the arousal, i.e., the emotional activation (calm vs. excited), by means of linguistic analysis of developers' communication traces. They validated SEA and its ability to correctly capture the emotional arousal of developers from the dataset 700,000 Jira issue reports containing over 2,000,000 comments used in the original study (Mäntylä et al. 2016). The results of the replicated study confirm previous findings that emotion-related metrics can be used to identify different types of issue reports as well as their priority. Furthermore, the replication with SEA shows that an SE-specific lexicon is able to better differentiate between issue priorities. In particular, a unified approach, combining SEA with the general purpose lexicon used in the original study, offers a clear improvement over previous work.

Overall, the results from previous studies suggest that SE-specific tools are required to overcome the limitations of general-purpose tools when applied to the software engineering domain. Furthermore, empirical evidence was provided that SE-specific tuning of tools improves the accuracy of sentiment analysis in empirical software engineering studies.

2.2 SE-specific Sentiment Analysis Tools

Sentiment analysis is the task of extracting positive, negative, or neutral orientation of opinions and emotions conveyed by a text. Despite the popularity of general-purpose sentiment analysis tools, a consensus in the research community is that such tools are not appropriate for detecting emotions in developers' technical discussions (Jongeling et al. 2017; Novielli et al. 2015; Lin et al. 2018). As such, researchers have implemented and released their own tools specifically customized for the software engineering domain. To enable comparison with previous findings, we select four tools that were already included in previous benchmarking studies on sentiment analysis in software engineering (Novielli et al. 2018b; 2020). Two of the tools are lexicon-based, namely SentiStrength-SE (Islam and Zibran 2017) and DEVA (Islam and Zibran 2018b), that is they rely on sentiment lexicons including polarity scores at the word level. The other two, namely Senti4SD (Calefato et al. 2018a) and SentiCR (Ahmed et al. 2017), implement a supervised approach and are trained on manually labeled gold standards.

SentiStrength-SE (Islam and Zibran 2017) is an SE-specific adaptation of the general-purpose tool SentiStrength (Thelwall et al. 2012). It leverages a manually adjusted version of the SentiStrength lexicon and implements *ad hoc* heuristics to correct the misclassifications observed when running it on the dataset by Ortu et al. (2016). The sentiment scores of words in the lexicon were manually adjusted to reflect the semantics and neutral polarity of domain words such as 'support' or 'default.' SentiStrength-SE assigns an integer value between 1 and 5 for the positivity of a text, p and, analogously, a value between 1 and 5 for the negativity n (respectively, the positive and negative scores in second and third columns

Table 1 Examples of the annotation produced by the SE-specific sentiment analysis tools

	Senti4SD	SentiStrength-SE		DEVA	SentiCR
Example documents from our datasets	Polarity label	Pos Score	Neg Score	Score	Label
I have Ubuntu 14.04 , I've installed JULIA, and then I tried to install IJULIA, writting Pkg.add("IJulia") on julia, but this appears: Do any one knows what to do? What I want to achieve is to run ipython notebook –profile julia	neutral	1	−1	1	0
I'm trying to use this cod for checking expiry date. I'm using Code before to open different file and copy cells as below. Please can any one help why I have Run time Error 13 Type mismatch. Any ideas ?? I was trying to put If empty cell exit sub but still that same :(	negative	1	−2	−2	−1
Here is my code: } It works fine so far without warning or errors. The only problem is, in the applications it shows for example Dauer: 0:4:3 instead of 00:04:03. Does anyone know how I can fix this? Thank you very much... :)	positive	3	−1	3	0

of Table 1). The evaluation performed by the authors shows that SentiStrength-SE achieves .74 precision and .85 recall, which are higher than the performance of the general purpose tool SentiStrength on technical texts.

DEVA (Islam and Zibran 2018b) leverages a lexicon-based approach for the identification of both emotion activation (arousal) and polarity from text. To this end, the tool uses two separate dictionaries developed by exploiting a general-purpose lexicon as well as one specific for software engineering text. To further increase its accuracy, DEVA also includes several heuristics, some of which are borrowed from SentiStrength-SE. Given an input text, the tool issues an integer score defined in the interval $\pm[1, 3]$, where values < -1 are considered negative and those > 1 are considered positive; otherwise values equal to ± 1 are neutral (see fourth column of Table 1). For the empirical evaluation, the authors constructed a ground-truth dataset consisting of 1,795 JIRA issue comments, manually annotated by three human raters, on which DEVA was found to achieve a precision of .82 and a recall of .79.

Senti4SD (Calefato et al. 2018a) is a polarity classifier trained to support sentiment analysis in developers' communication channels using supervised machine learning. It leverages a suite of features based on Bag of Words (BoW), sentiment lexicons, and semantic features based on word embedding. Along with the toolkit, the authors distribute a classification model, trained and validated on a gold standard of about 4K questions, answers, and comments from Stack Overflow, and manually annotated for sentiment polarity. Furthermore, the toolkit provides a training method that enables the customization of the classifier using a gold standard as input. Given any text in input, the tool returns a predicted label in {*positive*, *neutral*, *negative*} (see the first column of Table 1). Compared to the performance obtained by SentiStrength on the same Stack Overflow test set, Senti4SD reduces the misclassifications of neutral and positive posts as emotionally negative (precision = .87, recall = .87, F1=.87). The authors also report a good performance (F1=.84) of the tool with a minimal set of training documents.

SentiCR (Ahmed et al. 2017) is a supervised tool that leverages a feature vector generated by computing tf-idf for Bag-of-Words (BoW) extracted from the input text. SentiCR implements basic preprocessing of the raw input text to expand contractions, handle negations and emoticons, remove stop-words, derive word stems, and remove code snippets. Furthermore, it performs SMOTE (Chawla et al. 2002) to handle the class imbalance in the train set. Given a text in input, SentiCR issues either -1, for sentences predicted as negative, or 0 for non-negative ones—i.e., as opposed to Senti4SD, SentiStrength-SE and DEVA, SentiCR does not distinguish between positive and neutral content (see the last column of Table 1). Ahmed et al. (2017) report a mean accuracy of .83 for the tool. The currently distributed version implements a training approach based on Gradient Boosting Tree and requires a training set as an input in order to retrain the model and use it on the target document collection.

In our replications, we follow the methodology of Jongeling et al. (2017) and use all the SE-specific tools as 'off-the shelf' resources. For the two supervised tools Senti4SD and SentiCR we use the classifier trained on the original gold standard by their respective authors. Analogously, we use Sentistrength-SE and DEVA as they are originally implemented, that is leveraging the lexical resources as they are distributed by the authors after fine-tuning for the software engineering domain.

Table 2 Dataset of security-related comments and discussions

Data source	Topic	Distribution	
		Comments	Discussions
Commits	Security	2509 (4.13%)	1706 (9.28%)
	Total	60658	18378
Pull Requests	Security	1801 (3.28%)	1091 (11.36%)
	Total	54892	9601

3 Original Studies

In this section, we summarize the two original studies replicated in this paper, giving an overview of their research goals, settings, methodology, and results, following the replication report guidelines for software engineering studies by Carver (2010). We aim at performing exact replications (Shull et al. 2008) of the studies, except for the use of a SE-specific sentiment analysis tool. Specifically, the original study of Pletea et al. (2014) used NLTK[2], while the original study by Calefato et al. (2018b) used SentiStrength.[3] In our replications, we employ the four SE-specific tools reviewed in Section 2.2.

3.1 Sentiment Analysis of Security Discussions on GitHub (Pletea et al. 2014)

Research goals The first study we have chosen to replicate is the one by Pletea et al. (2014), which analyzes discussions about security in GitHub. The authors conclude that security-related threads contain more negative emotions than non-security related ones.

Dataset and methodology The dataset used by Pletea et al. consists in 60,658 commits and 54,892 pull requests from the MSR 2014 Mining Challenge Dataset (Gousios 2013), consisting of data for 90 GitHub projects (repositories) and their forks.

In their analysis, Pletea et al. distinguish between *i*) commits comments, *ii*) commit discussions, *iii*) pull request comments, and *iv*) pull request discussions. The term 'discussion' refers to the entire collection of comments belonging to a commit or a pull request. Furthermore, the authors distinguish between *a*) security-related and *b*) non-security related comments and discussions, thus resulting in eight different categories of texts. In this study, we use the same dataset used in the original study, as reported in Table 2.

To address the research questions, the authors apply NLTK to GitHub comments and discussions. Given an input text, NLTK outputs the probabilities that the text is neutral, negative, or positive as well as a polarity label summarizing the three scores. The probabilities for positive and negative sum up to 1 while the probability for neutral is computed *per se*. A text is classified as neutral if the neutral probability is greater than chance. Conversely, it is either classified as positive or negative based on the highest probability score. Depending on the aggregation considered (comments vs. discussions), Pletea et al. provide in input to NLTK either the individual comments or the entire pull request/commit discussions. Hence,

[2]Natural Language Processing APIs and Python NLTK Demos, http://text-processing.com

[3]http://sentistrength.wlv.ac.uk

the polarity level is issued at the level of either individual comment or the entire discussion, according to the chosen unit of analysis.

Findings The authors report that negative sentiment occurs more frequently in security related texts for both commit/pull request comments and discussions. Moreover, they observe that the NLTK results are characterized by both strong negative and strong positive sentiment. As such, Pletea et al. conclude that the security-related discussions tend to be more emotional than those not related to security.

A first replication of this study was performed by Jongeling et al. (2017) using both NLTK and SentiStrength. SentiStrength (Thelwall et al. 2012) is a general-purpose sentiment analysis tool widely adopted in social computing at the time of their study. SentiStrength is capable of dealing with short informal text, including abbreviations, intensifiers, and emoticons. Based on the assumption that a sentence can convey mixed sentiment, SentiStrength outputs both positive and negative sentiment scores for an input text. Positive sentiment scores range from +1 (absence of positive sentiment) to +5 (extremely positive) while negative sentiment scores range from 1 (absence of negative sentiment) to 5 (extremely negative).

Comparing the results obtained using NLTK in the original study by Pletea et al. and SentiStrength, Jongeling et al. observe differences in term of polarity label distributions. Overall, NLTK issues more negative labels, thus suggesting that both comments and discussions are predominantly negative. Conversely, SentiStrength classifies the texts as predominantly neutral. In spite of these differences, overall the study of Jongeling et al. confirms the findings originally reported by Pletea et al., i.e., security pull request/commit comments and discussions convey more negative sentiment than those revolving around other topics.

3.2 How to Ask for Technical Help (Calefato et al. 2018b)

Research goals The second study we have chosen to replicate is the one by Calefato et al. (2018b), which investigates how information seekers can increase the chance of eliciting a successful answer to their questions on Stack Overflow—i.e., the best answer marked as the accepted solution by the original asker. To achieve their research goal, they develop a framework of factors influencing the success of questions in Stack Overflow. The framework is built upon the community recommendations and the results from previous research in this field. Specifically, the authors focus on actionable factors that can be acted upon by software developers when writing a question to ask for technical help, namely *affect* (i.e., the positive or negative sentiment conveyed by text), *presentation quality*, and *time*. The asker's *reputation* is also included in the framework as a control factor. As far as sentiment is concerned, the authors investigate the correlation between expressing sentiments in a question on Stack Overflow and the question probability of eliciting a successful answer.

Dataset and methodology Calefato et al. analyze a dataset of 87,373 questions extracted from the official Stack Overflow dump, using logistic regression to estimate the probability of a successful question. The authors operationalize the framework by defining a set of metrics that are used as predictors in a logistic regression analysis. As for the affect factor, they operationalize it using two metrics, namely *Positive Sentiment* and *Negative Sentiment*, whose combination of values represents the overall sentiment orientation of a question. Specifically, they measure the overall positive polarity (*Positive Sentiment*) and negative polarity (*Negative Sentiment*) of the question body.

To capture the sentiment of questions, Calefato et al. use SentiStrength (Thelwall et al. 2012). They discretize the sentiment scores by treating Positive Sentiment and Negative Sentiment as Boolean variables. Specifically, they encode whether a question shows in its lexicon a positive (positive score in [+2, +5]) or negative (negative score in [2, 5]) affective load. A question is neutral when positive sentiment score is +1 and negative sentiment score is -1.

Findings Calefato et al. report that, regardless of user *reputation*, successful questions are short, contain code snippets, and do not abuse with uppercase characters. As regards *affect*, successful questions adopt a neutral emotional style. Based on that, they recommend avoiding both rudeness as well as positive sentiment in question writing. As regards *presentation quality*, they recommend including example code snippets, writing concise questions, and using capital letters only where appropriate. Finally, they observe the impact of *time* on the question success and advice Stack Overflow users to be aware of low-efficiency hours of the community.

4 Replications with SE-specific sentiment analysis tools

To address RQ1 (*Does the choice of the sentiment analysis tool introduce a threat to conclusion validity in a software engineering study?*), we replicate the original studies by Pletea et al. (2014) and Calefato et al. (2018b) using the four SE-specific sentiment analysis tools reviewed in Section 2.2. The mapping of the tool output to consistent polarity labels is described in Section 4.1. The results of the two replications are reported in Sections 4.2 and 4.3, respectively.

4.1 Mapping the Output of Tools to Polarity Labels

In our replications, we use all the SE-specific tools as 'off-the shelf' resources. Since the tools issue heterogeneous outputs, we need to map them into polarity labels by enforcing the same operationalization of sentiment adopted in the original studies.

In their study, Pletea et al. model sentiment using a single variable with three polarity classes, namely *negative*, *positive*, and *neutral* (see Section 3.1). The output of Senti4SD is consistent with such schema, so we do not need to apply any changes. Similarly, the scores produced by DEVA can be directly mapped into the corresponding negative ($score < 1$), positive ($score > 1$), and neutral ($score = 1$) labels. Because the output of SentiStrength-SE has the same structure as the output of SentiStrength, we adopt the mapping implemented for SentiStrength in the former replication by Jongeling et al. (2017): a text is considered positive when $p + n > 0$, negative when $p + n < 0$, and neutral if $p = n$ and $p < 4$. Texts with a score of $p = n$ and $p \geq 4$ are considered having an undetermined sentiment and, therefore, removed from the datasets. Finally, we translate the categorical scores of SentiCR into negative (*score* = -1) and non-negative (*score* = 0).

Calefato et al. use two Boolean values (*Positive Sentiment* and *Negative Sentiment*) to indicate the presence of positive and negative emotions, respectively. *Neutral* is modeled by assigning 'false' to both values. To represent mixed cases both values are equals to 'true' (see Section 3.2). As for Senti4SD and DEVA, we implemented a direct mapping by setting 'true'/'false' values based on the output of the tool. As for SentiStrength-SE, positive and negative sentiment are set to 'true' if *positive score* > 1 and *negative score* $<$ -1 , respectively, in line with the original study. Finally, the categorical scores of SentiCR are used to indicate the presence of negative sentiment. Conversely, the positive sentiment cannot be modeled in the empirical setting using SentiCR as this tool does not distinguish between positive and neutral.

4.2 Replication of Pletea et al.

As in the original study, we compute the proportions of negative, neutral, and positive in the *security* comments and discussions for both commits and pull requests. The resulting values are compared with the proportions of negative, neutral, and positive sentiment in the comments and discussions revolving around other topics, i.e. the *rest* of the texts in the dataset. We report the results from the replication and the comparison with the original work by Pletea et al. in Tables 3 and 4 for commits and pull requests, respectively. For the sake of completeness, we also report the results of the former replication of the work of Pletea et al. performed by (Jongeling et al. 2017), which used both NLTK and SentiStrength.

From Tables 3 and 4, we notice that there is always a larger proportion of negative polarity for security topics, regardless of the tool and the type of text. These results confirm the findings of the original study, as well as the findings of the replication by Jongeling and colleagues. In particular, we observe that different proportions of positive, negative, and neutral labels are issued by the different tools. Despite such differences, the original conclusion of Pletea et al. still holds: whether we consider comments or discussions, commits or pull requests, the proportion of negative sentiment among security-related texts is higher than among non-security related texts. As such, we can claim that, in the case of the replication of Pletea et al., different SE-specific tools do not lead to contradictory conclusions for this study. Furthermore, the choice of the sentiment analysis tool does not affect the conclusion validity of the results previously published.

However, the differences in the proportions of positive, negative, and neutral labels issued by the various tools indicate the need for further reflection on the possible threats to conclusion validity that might be due to the 'off-the-shelf' use of sentiment analysis tools, beyond the confirmation of the high level findings of the original study. In both Tables 3 and 4, we observe such differences. First of all, we observe that SE-specific tools for sentiment analysis all tend to classify pull request and commit comments as predominantly neutral. This is in line with previous evidence (Calefato et al. 2018a) showing how the SE-specific tools are able to solve the negative bias of general purpose ones, due to the inability of the latter in dealing with technical jargon and domain specific semantics of words, such as 'kill' or 'save,' which are considered non-neutral outside the technical domain. As such, we do not observe the prevalence of negative comments reported in the original study of Pletea and colleagues, which used the general purpose tool NLTK. Even if SE-specific tools agree on classifying the comments as mainly neutral, we still observe differences in the percentages. This pertains to the problem of assessing the agreement between the SE-specific tools (RQ3), which we address in Section 5.

As for pull request and commit discussions, overall we observe a lower proportion of neutral labels, even when the same tool is adopted. For example (see the lower half of Table 3), SentiStrength-SE classifies 68.84% of security commit comments as neutral, 15.46% as negative and 15.70% as positive. The situation changes when discussions are analyzed as a whole, i.e., as a group of comments belonging to the same thread originated by the commit (see the upper part of Table 3). In fact, SentiStrength-SE classifies 43.96% of discussions as neutral, with a resulting higher percentage of negative (24.73%) and positive (31.30%). Same considerations hold for the other tools ad for the pull request analysis

Table 3 Commit sentiment analysis statistics

Tool	Topic	Negative	Neutral	Positive
Commit Discussion				
Original study by Pletea et al. (2014)				
NLTK	Security	***72.52%***	10.88%	16.58%
	Rest	**54.28%**	20.37%	25.33%
Replication by Jongeling et al. (2017)				
NLTK	Security	***70.16%***	12.79%	17.05%
	Rest	**52.89%**	21.50%	25.61%
SentiStrength	Security	*30.66%*	**42.92%**	26.40%
	Rest	24.13%	**43.92%**	31.94%
Our replication with SE-specific tools				
SentiStrength-SE	Security	*24.73%*	**43.96%**	31.30%
	Rest	17.07%	**51.61%**	31.31%
DEVA	Security	***40.80%***	19.69%	39.51%
	Rest	31.79%	32.20%	**36.01%**
SentiCR	Security	***61.08%***	38.92% (non-neg.)	
	Rest	37.07%	**62.92%** (non-neg.)	
Senti4SD	Security	***36.93%***	27.72%	35.34%
	Rest	22.23%	**43.07%**	34.69%
Commit Comments				
Original study by Pletea et al. (2014)				
NLTK	Security	***55.59%***	23.42%	20.97%
	Rest	**46.94%**	26.58%	26.47%
Replication by Jongeling et al. (2017)				
NLTK	Security	***55.96%***	22.88%	21.16%
	Rest	**46.89%**	26.61%	26.50%
SentiStrength	Security	*32.60%*	**46.95%**	20.44%
	Rest	22.30%	**50.74%**	26.95%
Our replication with SE-specific tools				
SentiStrength-SE	Security	*15.46%*	**68.84%**	15.70%
	Rest	12.27%	**66.85%**	20.88%
DEVA	Security	*25.87%*	**51.13%**	23.00%
	Rest	21.10%	**51.38%**	27.52%
SentiCR	Security	*28.93%*	**71.06%** (non-neg.)	
	Rest	19.49%	**80.50%** (non-neg.)	
Senti4SD	Security	*17.58%*	**60.66%**	21.76%
	Rest	11.20%	**63.47%**	25.33%

The largest polarity group per study and topic is typeset in boldface. We report in *Italic* the higher percentage of negative comments between security and rest of the topics per study

Table 4 Pull request sentiment analysis statistics

Tool	Topic	Negative	Neutral	Positive
Pull Request Discussion				
Original study by Pletea et al. (2014)				
NLTK	Security	***81.00%***	5.52%	13.47%
	Rest	**69.58%**	11.98%	18.42%
Replication by Jongeling et al. (2017)				
NLTK	Security	***77.61%***	7.03%	15.36%
	Rest	**67.43%**	13.82%	18.76%
SentiStrength	Security	*30.80%*	**45.51%**	23.68%
	Rest	24.15%	**51.17%**	24.67%
Our replication with SE-specific tools				
SentiStrength-SE	Security	*27.33%*	**46.63%**	26.04%
	Rest	18.81%	**58.90%**	22.29%
DEVA	Security	***47.48%***	21.17%	31.35%
	Rest	30.35%	**41.66%**	27.99%
SentiCR	Security	***76.08%***	23.92% (non-neg.)	
	Rest	**52.06%**	47.94% (non-neg.)	
Senti4SD	Security	***49.31%***	31.90%	18.79%
	Rest	28.26%	**49.55%**	22.18%
Pull Request Comments				
Original study by Pletea et al. (2014)				
NLTK	Security	***59.83%***	19.09%	21.06%
	Rest	**50.16%**	26.12%	23.70%
Replication by Jongeling et al. (2017)				
NLTK	Security	***59.67%***	18.83%	21.50%
	Rest	**49.81%**	26.45%	23.74%
SentiStrength	Security	*25.66%*	**51.22%**	23.11%
	Rest	18.14%	**62.87%**	18.97%
Our replication with SE-specific tools				
SentiStrength-SE	Security	*11.60%*	**78.29%**	10.10%
	Rest	8.58%	**79.97%**	11.45%
DEVA	Security	*18.77%*	**63.91%**	17.32%
	Rest	12.77%	**70.55%**	16.67%
SentiCR	Security	*28.76%*	**71.24%** (non-neg.)	
	Rest	19.62%	**80.37%** (non-neg.	
Senti4SD	Security	*11.60%*	**74.90%**	13.49%
	Rest	7.28%	**77.70%**	15.01%

The largest polarity group per study and topic is typeset in boldface. We report in *Italic* the higher percentage of negative comments between security and rest of the topics per study

(Table 4). This evidence suggests that different findings can be derived also depending on the level of granularity of the unit of analysis (comments vs. discussions, in this case).

Table 5 Distribution of polarity labels predicted by the SE-specific tools for the study by Calefato et al. (2018b)

Tool	Negative	Neutral	Positive
Original study by Calefato et al. (2018b)			
SentiStrength	11.00%	63.00%	26.00%
Our replications with SE-specific tools			
SentiStrength-SE	17.00%	**51.07%**	31.93%
DEVA	20.27%	40.05%	39.68%
SentiCR	39.33%	**60.67%** (non-negative)	
Senti4SD	34.39%	**40.73%**	24.88%

The larger group per study is typeset in boldface

4.3 Replication of Calefato et al.

In our replication, we use the dataset of the original study.[4] As in the original study, we apply a logistic regression for estimating the correlation of each factors on the probability of success of a Stack Overflow question. In particular, we use the output of the four SE-specific sentiment analysis tools to recompute the metrics associated with the *affect* and other factors of their framework, i.e., the positive and negative sentiment scores. Conversely, we do not recompute the metrics associated to the remaining factors and use those originally extracted in the original study, as they are distributed in the original dataset. Since SentiCR only distinguishes between negative and non-negative sentiment, in the replication using this tool we could only include the *Positive Sentiment* among the predictors. In line with the original study, we treat the success of a question—i.e., the presence of an accepted answer—as the dependent variable and the metrics that operationalize each factors as independent variables.

In Table 6, we report the results of the original study as well as the outcome of the replications with the four SE-specific sentiment analysis tools. For each predictor, we report the coefficient estimate, the odds ratio (OR), and indicate statistical significance of the correlation (p-value $<.05$) in bold. The sign of the coefficient estimate indicates the positive/negative association of the predictor with the success of a question. The odds ratio (OR) weighs the magnitude of this impact, with values close to 1 indicating a small impact. A value lower than 1 corresponds to a negative coefficients, and vice versa. Technically speaking, an $OR = x$ indicates that the odds of the positive outcome are x times greater than the odds of the negative outcome. OR is an asymmetrical metric, with positive odds varying from 1.0 to infinity and decreasing OR bounded by 0. To further investigate the outcome of classification performed with the different tools, in Table 5, we report the label distributions for the predictions issued by the SE-specific tools and provide comparison with those obtained with SentiStrength from the original study.

As for RQ1, we confirm the original findings related to *Reputation*, *Time*, and *Presentation Quality*, for which we observe comparable coefficients and ORs in all settings. More in detail, we confirm that Reputation is the most influential factor for the success of questions, with *Trusted* users having the highest probability of getting an accepted answers. As for Time, *evening* and *night* are confirmed as the most effective time zones. As far as Presentation Quality is concerned, the presence of code snippet remains the strongest predictor in

[4]The original dataset can be downloaded from https://goo.gl/whZEWA

Table 6 Results of the replication of the study by (Calefato et al. 2018b) on success factors for Stack Overflow questions

Factor	Predictor	Experimental setting: sentiment analysis tool used									
		SentiStrength (Calefato et al. 2018b)		Senti4SD		SentiStrength-SE		DEVA		SentiCR(binary)	
		Coeff.	OR	Coeff.	OR	Coeff.	OR	Coeff.	OR	Coeff.	OR
	(Intercept)	–1.60	–	-1.70	–	-1.74	–	-1.76	–	-1.74	–
Affect	Positive Sentiment	**-0.06**	0.94	**-0.06**	0.94	-0.03	0.97	0.00	1.01	–	–
	Negative Sentiment	**-0.21**	0.81	**-0.16**	0.85	-0.01	0.99	0.04	1.04	-0.09	0.92
Time	Weekend	**0.10**	1.10	**0.10**	1.11	**0.10**	1.11	**0.10**	1.11	**0.10**	1.11
	GMTHour Afternoon	**0.07**	1.07	**0.06**	1.07	**0.07**	1.07	**0.06**	1.07	**0.07**	1.07
	Evening	**0.15**	1.16	**0.14**	1.15	**0.14**	1.15	**0.14**	1.15	**0.15**	1.16
	Night	**0.13**	1.14	**0.13**	1.14	**0.13**	1.14	**0.13**	1.14	**0.13**	1.14
Presentation Quality	Presence of Code Snippet	**0.71**	2.04	**0.73**	2.08	**0.72**	2.05	**0.72**	2.05	**0.72**	2.06
	Low Uppercase Ratio	**0.12**	1.27	0.00	1.00	0.01	1.00	0.00	1.00	0.00	1.00
	Body Length:										
	Medium Body [90. 200 [	**-0.21**	0.81	**-0.20**	0.82	**-0.22**	0.80	**-0.23**	0.79	**-0.20**	0.82
	Long Body [200. +∞ [	**-0.42**	0.65	**-0.45**	0.64	**-0.46**	0.63	**-0.49**	0.61	**-0.44**	0.65
	Title Length:										
	Medium Title [6. 10[	0.05	1.05	0.05	1.05	**0.05**	1.06	**0.05**	1.06	**0.05**	1.05
	Long Title [10. +∞ [	**0.05**	1.05	0.05	1.06	**0.06**	1.06	**0.06**	1.06	**0.06**	1.06
	Presence of Multiple Tags	**-0.12**	0.88	**-0.13**	0.88	**-0.14**	0.87	**-0.14**	0.87	**-0.14**	0.87
	Presence of URLs	-0.01	0.99	-0.01	0.99	-0.02	0.98	-0.02	0.98	-0.01	0.99
Reputation	Low [10. 1K[	**0.87**	2.39	**0.87**	2.39	**0.88**	2.40	**0.88**	2.41	**0.88**	2.41
	Established [1K. 20K[	**0.98**	2.68	**0.98**	2.68	**0.99**	2.69	**0.99**	2.70	**1.00**	2.71
	Trusted [20K. +∞ [	**1.17**	3.23	**1.17**	3.21	**1.18**	3.25	**1.18**	3.26	**1.19**	3.30

Boldface indicates statistical significance with $p < .05$

Table 7 Weighting scheme for computation of κ

	Negative	Neutral	Positive
Negative	0	1	2
Neutral	1	0	1
Positive	2	1	0

all settings. The results from the replications also confirm the *Body Length* to be negatively correlated with the success of questions. Conversely, we could not confirm the positive correlation between low uppercase ratio with the success of questions. As for the impact of the *Affect* factor, we confirm the negative impact of both positive and negative sentiment on the success of questions when Senti4SD is used. Conversely, for SentiStrength-SE, DEVA, and SentiCR we do not find empirical support for this claim. As already observed in the replication of the study by Pletea and colleagues (see Tables 3 and 4 in Section 4.2), the proportions of polarity labels vary depending on the tools. As such, we conclude that the choice of the sentiment analysis tools leads to partially contradictory results for this study. The original findings about the impact of factors on the success of questions are mostly confirmed, with a couple of exceptions. Specifically, the sentiment from the original work are fully confirmed only when Senti4SD is used for performing sentiment analysis of questions, while the impact of writing in uppercase is not confirmed in any setting.

RQ1 Summary: The 'off-the-shelf' use of SE-specific sentiment analysis tools may induce a threat to conclusion validity depending on the chosen level of granularity of the analysis. In both our replications, the tools lead to the same conclusions at a coarse level of granularity while differences are observed in terms of distribution of polarity labels in the two datasets.
Our replications with SE-specific sentiment analysis tools confirm the findings from earlier studies at a coarse-grained level of analysis. However, we find that the off-the-shelf use of sentiment analysis tools might lead to different results at a fine level of granularity due to differences in the label distributions.

5 Agreement between SE-specific tools

To address RQ2 (*To what extent do results from different SE-specific sentiment analysis tools agree with each other?*), we measure the agreement among the four SE-specific tools using the weighted kappa (κ) by Cohen (1968). The κ metric is computed as the *observed agreement*, i.e., the number of times the raters (either humans or classifiers) issue the same labels, corrected by the *chance agreement*, that is the probability that the raters agree by chance. Consistently with previous research (Jongeling et al. 2017; Novielli et al. 2018b), we distinguish between *mild disagreement*, that is the disagreement between negative/positive and neutral annotations, and *severe disagreement*, that is the disagreement between positive and negative judgments. As such, in computing the weighted κ, we assigned a weight = 2 to severe disagreement and a weight = 1 to mild disagreement (see Table 7). We follow the interpretation of κ by Viera and Garrett (2005), suggesting that the agreement is less than chance if $\kappa \leq 0$, slight if $0.01 \leq \kappa \leq 0.20$, fair if $0.21 \leq \kappa \leq 0.40$, moderate if $0.41 \leq \kappa \leq 0.60$, substantial if $0.61 \leq \kappa \leq 0.80$, and almost perfect if $0.81 \leq \kappa \leq 1$.

The κ index represents the *de facto* standard used in research to assess the inter-rater agreement. However, a debate is still open about the limitations of the κ statistic due to the presence of bias in the case of skewed distributions of labels (Di Eugenio B and Glass 2004). The main problem with κ applied to computational linguistics and text classification tasks is that it is possible to observe low κ values even in presence of high agreement if the distribution of one label largely outnumbers the others. In fact, the minimum value of chance agreement occurs when the labels are equally distributed among all the possible categories (i.e., *positive, negative*, and *neutral* in our case). Conversely, the maximum value of chance agreement (i.e., 1) occurs when all the labels belong to a single category. Indeed, the problem of unbalanced polarity labels affects both our replication datasets, for which we observe a prevalence of neutral cases (see Tables 3, 4, and 5).

Given these limitations and for the sake of completeness, in Tables 8 and 9 we report both the weighted Cohen's κ for the two datasets and the observed agreement, i.e., the the percentage of cases for which the tools in each pair (by row) issue the same prediction (perfect agreement). Furthermore, in line with previous research by Novielli et al. (2018b), we also report the percentage of cases for which severe and mild disagreement are observed. The only exception is represented by SentiCR that predicts two classes (i.e., negative vs. non-negative), thus making impossible to distinguish between severe and mild disagreement.

For both datasets (see Tables 8 and 9), the κ values indicate an agreement between tools ranging from slight to moderate. We observe the highest values of κ between SentiStrength-SE and DEVA is the highest (.45–.59). This is reasonable as they have been created by the same team and share the same underlying API and lexicon, which is an adaptation of the lexicon and the rules of SentiStrength. The second highest agreement (.33–.53) is observed between SentiStrength-SE and Senti4SD, which is again expected as Senti4SD also leverages the SentiStrength lexicon that significantly overlaps with the adapted version implemented by SentiStrength-SE. The lowest agreements involve SentiCR with both DEVA (.04–.18) and SentiStrength-SE (.08–.15). A possible explanation for this is that SentiCR does not leverage any sentiment dictionary, thus relying only on Bag-of-Words (BoW) as features for the supervised classification model trained on a gold standard of code reviews. Overall, Senti4SD is in the middle of the scale of observed κ values, showing a moderate agreement with lexicon-based tools and a fair agreement with SentiCR, probably because it leverages both lexicon-based features and BoWs.

These results are in line with findings from our previous benchmarking studies (Novielli et al. 2018a; Novielli et al. 2020). Furthermore, we observe an improvement from $\kappa = .25$, which is the highest agreement reported by Jongeling et al. for general purpose tool on GitHub comments, to $\kappa = .59$ and $\kappa = .57$, which we observe for SentiStrength and DEVA, respectively on pull request and commit comments, thus confirming the previous findings about the positive impact of SE-specific tuning of sentiment analysis tools (Novielli et al. 2018a). However, lower agreement is observed for longer texts such as GitHub discussions or Stack Overflow questions, for which we observe values of $\kappa = .47$ and $\kappa = .48$, respectively.

Looking at the *disagreement* rates, we observe that the main cause for disagreement is between neutral and either positive or negative polarity (mild disagreement). The highest rates of severe disagreement are observed for both pull request and commit discussions (between 8–16%, see Table 8) as well as for Stack Overflow questions (between 7–17%, see Table 9). For pull request and commit comments, the severe disagreement ranges between 2–6%, thus providing a further indication of the higher difficulty of issuing an overall polarity label for longer texts.

Finally, the κ values are generally lower than the ones observed with fine-tuned SE-specific tools in a within-platform condition. In our previous study (Novielli et al. 2020), we observed a drop in agreement between tools for the supervised classifiers (i.e., Senti4SD and SentiCR, in this case) when trained and tested in a cross-platform condition, that is, when using train and test sets from different platforms. In line with evidence provided in previous benchmark study, we observe the lowest agreement for pairs that include SentiCR. This might be explained by the approach implemented by SentiCR, which relies on features based on bag-of-words that hold a lower ability to generalize across-dataset thus causing overfitting to the training data of the platform-specific lexicon. This evidence confirms that shifts in lexical semantics also occur within the software development domain due to platform-specific jargon and communication style, and highlight that SE-specific classifiers are no silver bullet since problems arise if they are also used as 'off-the-shelf' solutions.

RQ2 Summary: Agreement between SE-specific tools ranges from slight to moderate. The highest agreement is observed for the two tools sharing the same lexicon and developed upon the same API (i.e., SentiStrength-SE and DEVA). More cases of severe disagreements are observed in GitHub discussions and Stack Overflow questions than in GitHub comments, thus suggesting that issuing an overall polarity label is more difficult for longer texts.

6 Agreement of SE-specific tools with manual annotation

To address RQ3 (*To what extent do different SE-specific sentiment analysis tools agree with human raters?*), we manually labeled a subset of 600 documents randomly selected from the GitHub and Stack Overflow datasets: 200 pull request and 200 commit comments, equally distributed between the 'security' and 'no security' groups from the dataset by (Pletea et al. 2014), plus 200 questions from the dataset by (Calefato et al. 2018b)

The labeling study was performed by the first two authors (*raters* hereinafter), following previously published annotation guidelines for sentiment polarity by Calefato et al. (2018a). Each document in the subsets was individually annotated by the two raters. For each document, the raters indicated whether positive or negative sentiment was conveyed. It was also possible to indicate both positive and negative polarity, to represent cases of *mixed* polarity. To indicate neutral sentiment, the raters were required to annotate absence of both positive and negative polarity. All the cases of disagreement were resolved through discussion, leading to the assignment of a gold label to each document. Two comments from the GitHub subset were discarded because the raters could not reach an agreement on the polarity label. Conversely, the raters successfully addressed all the disagreements in the Stack Overflow subset. The resulting label distributions for the two subsets are comparable, with the majority of cases labeled as neutral (see Table 10). The GitHub subset contains a slightly higher proportion of positive cases compared to the Stack Overflow one. Vice versa, negative sentiment occurs more often in Stack Overflow. A larger proportion of mixed cases appear in the Stack Overflow subset (7%) than in GitHub (2%). A possible explanation for this, as also emerged during the disagreement resolution discussions, is that longer texts are more likely to convey both positive and negative sentences. In fact, the average length of the documents in the GitHub sample is 134 characters whereas it is 430 in Stack Overflow.

Table 8 greement between SE-specific tools for the dataset used in the study by Pletea et al. (2014)

		Perfect	Disagreement	
Tool Pair	κ	Agreement	Severe	Mild
Pull request discussions				
SentiStrength-SE vs. DEVA	0.45	67%	10%	23%
SentiStrength-SE vs. Senti4SD	0.33	61%	10%	30%
DEVA vs. Senti4SD	0.29	58%	16%	26%
SentiStrength-SE vs. SentiCR	0.15	55%	45%	
DEVA vs. SentiCR	0.25	61%	39%	
SentiCR vs. Senti4SD	0.32	65%	35%	
Commit discussions				
SentiStrength-SE vs. DEVA	0.47	66%	10%	23%
SentiStrength-SE vs. Senti4SD	0.45	66%	8%	26%
DEVA vs. Senti4SD	0.35	60%	16%	24%
SentiStrength-SE vs. SentiCR	0.15	63%	36%	
DEVA vs. SentiCR	0.18	62%	38%	
SentiCR vs. Senti4SD	0.29	69%	31%	
Pull request comments				
SentiStrength-SE vs. DEVA	0.59	84%	2%	14%
SentiStrength-SE vs. Senti4SD	0.47	82%	2%	17%
DEVA vs. Senti4SD	0.36	75%	3%	22%
SentiStrength-SE vs. SentiCR	0.08	77%	23%	
DEVA vs. SentiCR	0.05	74%	26%	
SentiCR vs. Senti4SD	0.14	79%	21%	
Commit comments				
SentiStrength-SE vs. DEVA	0.57	76%	4%	20%
SentiStrength-SE vs. Senti4SD	0.53	76%	3%	21%
DEVA vs. Senti4SD	0.39	66%	6%	28%
SentiStrength-SE vs. SentiCR	0.09	75%	25%	
DEVA vs. SentiCR	0.04	69%	31%	
SentiCR vs. Senti4SD	0.16	77%	23%	

Table 9 Agreement between SE-specific tools for the dataset used in the study by Calefato et al. (2018b)

		Perfect	Disagreement	
Tool Pair	κ	Agreement	Severe	Mild
SentiStrength-SE vs. DEVA	0.48	68%	7%	25%
SentiStrength-SE vs. Senti4SD	0.28	53%	11%	36%
DEVA vs. Senti4SD	0.21	50%	17%	34%
SentiStrength-SE vs. SentiCR	0.10	61%	39%	
DEVA vs. SentiCR	0.14	63%	37%	
SentiCR vs. Senti4SD	0.23	64%	36%	

Table 10 Distribution of polarity labels resulting from the manual annotation

	Dataset	
	Pletea et al.	*Calefato et al.*
Unit of annotation	GitHub comments	Stack Overflow questions
#Documents	400	200
Negative	47 (12%)	35 (18%)
Neutral	290 (73%)	132 (66%)
Positive	52 (13%)	20 (10%)
Mixed	9 (2%)	13 (7%)
Discarded	2 (.05%)	–

We measure the agreement between the raters, as well as between manual gold labels and the tool predictions, using the same metrics and approach adopted for assessing the agreement between tools (see Section 5). The agreement metrics in Table 11 show a moderate agreement between the two raters for both datasets. Similarly to what observed for the agreement between tools, κ values indicate better agreement for shorter documents. This holds true for both inter-rater agreement in the manual labeling study and for the tool-based vs. manual labels. In fact, we observe better agreement for the GitHub commit and pull request comments (average length of the comments in the labeled set is 134 characters). Conversely, a lower agreement is observed for the Stack Overflow questions (average length of 430 characters) for which we also observe a larger proportion of severe disagreement cases and mixed cases (see Table 10).

Table 11 Agreement between raters in the labeling study and between tools and manual labels

	Weighted	Observed	Disagreement	
Pair	k	Agreement	Severe	Mild
Github comments				
Rater 1 vs. Rater 2 (manual labeling)	0.54	80%	3%	17%
SentiStrength-SE vs. manual labels	0.50	77%	8%	21%
Senti4SD vs. manual labels	0.44	72%	3%	24%
DEVA vs. manual labels	0.35	69%	6%	25%
SentiCR vs. manual labels	0.16	81%	19%	
Stack Overflow questions				
Rater 1 vs. Rater 2 (manual labeling)	0.45	69%	3%	27%
SentiStrength-SE vs. manual labels	0.29	57%	5%	37%
Senti4SD vs. manual labels	0.29	59%	7%	33%
DEVA vs. manual labels	0.19	51%	11%	37%
SentiCR vs. manual labels	0.13	68%	32%	

6.1 Follow-up analysis on majority voting

We run a follow-up analysis to investigate if the agreement with the manual gold label improves if we focus on the comments for which the tools agree with each other. To this aim, we replicate the approach followed by Jongeling et al. and select the subset of GitHub comments and Stack Overflow questions for which Senti4SD, SentiStrength-SE, and DEVA agree on the polarity classification. We cannot include SentiCR in this analysis as we are not able to disambiguate between positive and neutral cases for which the tool issues the 'non-negative' macro-label. As a further investigation, we also repeat the same assessment by considering only the two tools with higher agreement with manual labels, namely SentiStrength-SE and Senti4SD (see Table 11). Of course, given the observed agreement between tools, this analysis focuses on a reduced number of text items. We compute the agreement of tools with the manual labels on 262 comments from the GitHub sample (66%) and 73 questions from Stack Overflow (36%). For two tools, a larger proportion of texts is included in the analysis, i.e., 302 GitHub comments (75%) and 111 Stack Overflow questions (56%).

We summarize the results of this follow-up study in Table 12. As already reported by Jongeling et al., we observe that focusing on the comments where the tools agree improves the agreement with the manual labeling, both in terms of weighted κ and in terms of disagreement rates. Agreement with manual labels become substantial, raising from a maximum of .50 in the full dataset (SentiStrength-SE, see Table 11) to .69 for GitHub and from .29 (SentiStrength-SE and Senti4SD, see Table 11) to .60 for Stack Overflow. As for disagreement, the proportion of severe disagreement rates drops to 2% and 3% for GitHub and Stack Overflow, respectively, which is comparable to the disagreement between human raters (see Table 11). More recently, similar findings were presented by Zhang et al. (2020). In their study leveraging deep learning for sentiment analysis, they provided evidence on how composition of different classifiers may boost performance. In line with previous findings, this evidence suggests that in absence of a gold standard for retraining, implementing a voting system between off-the-shelf tools might be a way to increase classification performance. However, a trade-off exists between the accuracy and the agreement with manual labeling. This approach would not enable to classify the polarity of sentiment for those cases for which tools do not reach an agreement, thus leaving the sentiment for such documents undetermined.

Table 12 Agreement of group of tools with manual labeling

Tools	#agreement	Weighted	Observed	Disagreement	
	cases (%)	k	Agreement	Severe	Mild
GitHub					
Three tools	262 (66%)	.69	87%	2%	11%
Two tools	302 (75%)	.60	85%	2%	12%
Stack Overflow					
Three tools	73 (36%)	.60	78%	3%	19%
Two tools	73 (36%)	.60	78%	3%	19%

6.2 Error analysis

To better understand the difficulties inherent to sentiment detection in our dataset, we performed a qualitative analisys by manually examining the cases for which the gold label did not match the majority voting label based on tool predictions. In order to include SentiCR in this analysis, we mapped positive and neutral labels to 'non-negative'. Furthermore, we excluded from this analysis the text documents for which we could not assign a polarity class based on majority voting between tools (31 for GitHub and 34 for Stack Overflow). The resulting collection counted 67 text documents (referred to as the *disagreement set* hereinafter), of which 36 are GitHub comments and 30 are Stack Overflow questions. The two authors who performed the manual labeling also inspected the content of the documents in the disagreement set to identify the cause of error. Specifically, one rater assigned the error categories, while the other one reviewed them. The two raters labeled the disagreement set by using the error categories defined in the benchmarking study on SE-specific sentiment analysis tools by Novielli et al. (2018a). Therefore, each document in the disagreement set was annotated with possible causes of error. In Table 13, we report the error category distribution resulting from our analysis, ordered by frequency of observation. The overall number of documents belonging to each class is reported as well, with a breakdown by dataset. For each error category, we also indicate the percentage of items belonging to it. In 7 cases multiple error categories were selected, indicating multiple causes for misclassification. For this reasons the percentages do not sum up to 100%. In 9% of cases (10 Stack Overflow questions), the raters agreed that there was no error in the classification as the disagreement actually originated from the document conveying both positive and negative polarity (i.e., *mixed* cases in Table 13), and, therefore, they are not included in the error category count.

The most frequent error category is *general error*, that is the inability to identify lexical cues of sentiment, as in the following GitHub comment:

> "The alternative of course is to fixup the build system to split swscale and libav*, which wouldn't be a terrible idea as there are several places currently where the removal of ffmpeg would mean the removal of unrelated features (screenshots, as an example)."

that is erroneously classified as non-negative by the majority of tools, probably due to their inability to deal with the negation (*wouldn't* of the negative word *terrible*). Other

Table 13 Distribution of error categories in the Github and Stack Overflow (SO) datasets

Error category	Github	SO	overall (%)
General error	12	17	29 (43%)
Implicit sentiment polarity	13	8	21 (31%)
Figurative language	5	1	6 (9%)
Politeness	1	3	4 (6%)
Pragmatics	3	1	4 (6%)
Subjectivity in annotation	2	–	2 (3%)
Polar facts	1	–	1 (1%)
Overall labeled documents	35	30	
Overall labels assigned	37	30	
Documents with multiple error categories	2	–	

causes for general errors are related to the wrong preprocessing of the raw text (i.e., code or URL not removed properly before launching the tool-based classification), or to the fact that the classifiers may overlook lexical features because either they are not included in their sentiment lexicons or they are not modeled as relevant due to the low frequency in the training sets of the supervised classifiers.

In 31% of cases the tools fail because of the presence of *implicit sentiment polarity* in text, as in as in the following examples:

> "Any way to report as spam on comments like these"

> "Recently there were quite some changes which have not made their way to the wiki yet. Someone should update it..."

> "It is required to return HashWithIndifferentAccess instead of just plain regular Hash??? I mean what value does it bring in to the code???"

These cases received a *negative* gold label because human raters could identify the negative attitude of the authors' comments towards their interlocutor, even in the absence of explicit emotion lexicon. This evidence is in line with previous findings from our study on anger in collaborative software development, showing how detecting negative sentiments towards peers is more difficult than detecting anger towards self or objects (Gachechiladze et al. 2017). In fact, we observed that a hostile attitude towards the interlocutor is often phrased using implicit, indirect lexicon, probably to avoid too aggressive verbal behavior.

In 9% of misclassified texts, we observe the use of *figurative language* such as the humor, irony, sarcasm, or metaphors. For example:

> "@username oh, duh! I'll just go over to the corner and hang my head for awhile for forgetting about that. I suppose it may be still worth adding a method to do the comparison, as a convenience and 'nudge in the hey use this direction' for plugin a."

> "You are getting rusty in your old age..."

Such cases are extremely challenging to classify for sentiment analysis tools because much figurative language is based on conventions, such as idiomatic expressions and proverbs. Indeed dealing with figurative language, in general, and irony and sarcasm in particular, is still an open challenge for sentiment analysis research (Bosco et al. 2013; Weitzel et al. 2016).

A few cases are misclassified because of the presence of ambiguous lexical cues of *politeness* such as '*thank you*,' which might be either interpreted as a way to convey politeness or actual emotions (e.g., showing gratitude or appreciation) (Novielli et al. 2015). Finally, we observe a few cases for which misclassification is due to the inability of tools to deal with *pragmatics*. It is the case, for example, of suggestions formulated as questions, as in "*Would IE user agent sniffing be a bad idea?*," where the presence of negative lexicon (i.e., *bad*) causes misclassification of this neutral text as negative. This is in line with previous research showing how developers use questions to express emotional lexicon in order to induce critical reflection in the interlocutor, to express criticism but also actual emotions, such as anger or surprise, as well as to communicate perplexity or disagreement with respect to the implemented solution (Ebert et al. 2018). Being able to identify the actual communicative intention of a questions is not a trivial task. It requires being able to understand information about context and pragmatics that cannot be successfully processed by state-of-the-art sentiment analysis tools, which rather rely on lexical semantics of documents.

RQ3 Summary: The agreement between manual and tool annotation is higher for shorter documents (GitHub comments). The main causes of error are the presence of lexical cues of sentiment that are either wrongly processed or overlooked by the tools, the use of neutral lexicon to implicitly convey non-neutral polarity, and the use of figurative language.

7 Discussion

In the following, we summarize the key insights from our replications in the form of actionable guidelines to inform future research on sentiment analysis in software engineering.

Sentiment analysis tools should be retrained, if possible, rather than used off the shelf We observe that, when used off-the-shelf as in our replications, SE-specific sentiment analysis tools may lead to contradictory results if different levels of unit of analysis are considered. When replicating the study by Pletea et al., we could confirm the original findings that GitHub security comments and discussions convey more negative sentiment than non-security ones, regardless of the tool used. Similarly, when replicating the study on Stack Overflow by Calefato et al., we could confirm the impact on question success of users' reputation and the presentation quality of questions. However, in both cases, we observe a different distribution of polarity labels that may lead to different conclusions at a finer-grained level of analysis. As such, albeit specifically tuned for the software engineering domain, we argue that using sentiment analysis tools off-the-shelf poses a potential threat to conclusion validity, which might have an impact on the findings of previously published studies in the field. This evidence is also supported by the findings reported in a benchmarking study by Novielli et al. (2020), who report a drop in performance in cross-platform settings, i.e., when SE-specific sentiment analysis tools are used off-the-shelf in the absence of a gold standard for retraining.

Perform a preliminary sanity check to select the appropriate tool in line with the research goals One of the assumptions underlying the choice of a sentiment analysis tool is that we share the original goal and the same conceptualization of affect with the authors of the tool. This is not necessarily true, regardless of the fine-tuning for the SE domain, as different gold standard might be inspired by different theoretical models of affect, aiming at modeling different affective states, such as emotions, interpersonal stances, attitudes, or moods. The disagreement between tools observed in our replications shows how SE-specific tuning of sentiment analysis tools does not necessarily represent *per se* a silver bullet for improving the accuracy of sentiment analysis tools in software engineering studies. Therefore, a possible explanation for the low agreement observed is that the benchmarked tools have been originally validated and tuned on gold standards that include manual annotation following different guidelines. As already pointed out by previous research, sentiment annotation is a subjective task, thus even humans might disagree with each-others (Imtiaz et al. 2018) if model-driven annotation is not adopted (Novielli et al. 2018b). Moreover, Islam and Zibran (2018a) showed how tools exhibit their best performance on the dataset they were originally tested at the time of their release, whereas a drop in performance is observed when they are assessed on a different dataset. Furthermore, they report that the accuracy of the tools largely vary across different datasets in line with what was observed by Lin et al.

(2018) and further confirmed by our cross-platform benchmarking study (Novielli et al. 2020).

In our previous research, we already argued about the importance of grounding sentiment analysis research on theoretical models of affect (Novielli et al. 2018b). As such, a sanity check is always recommended to assess the suitability of existing tools with respect to the specific research goals (Novielli et al. 2018a; 2020). For example, in some cases the mining of opinions might be the target rather than the recognition of actual emotions, such as joy or sadness. (Lin et al. 2019) reported poor performance of classifiers designed to detect developers' *emotions* (e.g., *love*, *joy*, *fear*) when applied to the different task of detecting developers' *opinions* about software libraries.

If retraining is not possible, consider an ensemble of multiple tools to improve performance As for any classifier, the retraining of sentiment analysis tools on new datasets can largely improve their performance. However, not all the solutions support retraining, as in the case of the lexicon-based tools DEVA and SentiStrength-SE. Based on the results of our analysis on the agreement of tools (see Section 5) and in line with previous evidence (Jongeling et al. 2017; Zhang et al. 2020), we suggest implementing an ensemble of tools with a majority voting system as a possible way to increase the agreement with manual labels when the retraining of the selected solution is not an option.

Beware of the effects of different units of analysis on tool performance The results of the agreement study suggest that the choice of the unit of analysis may represents a potential threat to construct validity. We observe better agreement for shorter documents both between tools (see Tables 8 and 9) as well as between tools and manual annotation (see Table 11). Specifically, we found the best κ values of agreement between tools for GitHub comments (shorter unit of analysis), immediately followed by Stack Overflow posts, and the worst values for GitHub discussions (longer unit of analysis). This evidence suggests that sentiment analysis perform better on short text, whereas longer documents are more problematic as they might actually convey both positive and negative emotions–as confirmed by the higher percentage of manually-labeled mixed cases for the Stack Overflow dataset (7%) compared to GitHub comments (2%) (see Table 10).

Furthermore, we observe that each tool produces different distributions of positive, negative, and neutral labels for the dataset by Pletea et al., who considered as units of analysis individual comments vs. entire discussion. This confirms the evidence by (Jongeling et al. 2017) that care is needed when choosing the unit of analysis as it may affect the resulting polarity label distribution. Therefore, other than fine-tuning sentiment analysis tools to address the challenges specific of the software engineering domain (Novielli et al. 2018b) and the data source (Lin et al. 2018), tools might need further *ad hoc* tuning if different lengths of documents or interaction threads are considered. For example, for the two supervised tools considered in this study (Senti4SD and SentiCR), this tuning could be done by retraining the classification models using a training set with gold labels assigned at a different granularity, e.g., at the level of individual sentences rather than at the level of the entire document. If retraining is not feasible due to the absence of a gold standard, we recommend explicit modeling of mixed cases by using tools that are designed to denote the presence/absence of positive and negative sentiment, such as SentiStrength-SE.

Finally, the dependency of the label distribution on the tools used and on the unit of analysis suggest further implications on the replicability of findings. In spite of the different polarity distribution labels as issued by the four tools inlcuded in our replications, the claims of Pletea et al. are more stable regardless of the tool used since they do not explicitly refer to this distribution but to percentage over groups (i.e., security vs. non-security) as opposed to negative vs positive. These findings are also confirmed when discussions rather than comments are used as a unit of analyis. However, different scenarios and research goals might suffer seriously from the impact of the choice of the tools and of the unit of analysis. It is the case, for example, of identification of negative comments denoting a hostile attitude in the scope of community moderation activities, where optimizing by precision of the negative class on a single comment is crucial to avoid unnecessary ban of users.

8 Threats to Validity

Since we strictly replicated the two original studies, we also inherited some of the threats to validity reported in the original papers, e.g., the datasets under consideration are not representative for GitHub and Stack Overflow as a whole. Still, the limitations to the validity of each study are shared across both the original studies and the replications, and, therefore, unlikely to have influenced the findings reported here.

As for the annotation of the 600 document gold standard, threats to construct validity are mitigated by the fact that the labeling is performed by two independent raters, with discussion-based resolution of disagreements.

When running replications, one of the inherent risk is to introduce unintended minor changes in the settings used in the original study, thus observing major differences in the results that are due to confounding factors. To mitigate this risk, we decided to perform our own studies, to avoid weakening the conclusion validity of our study. In particular, we performed two dependent replications in which the original design of the studies was preserved. The only change we introduced consists in the choice of the sentiment analysis tool, in line with our research goal of investigating the impact of the choice SE-specific tools on the conclusion validity.

9 Conclusion

In this paper, we reported the results of an extended replication aimed at assessing to what extent SE-specific sentiment analysis tools mitigate the threats to conclusion validity highlighted by previous research. We found that, despite being tuned to address the challenges specific to the SE domain, the use of different sentiment analysis tools might lead to contradictory results when used off-the-shelf. This is especially true at a fine-grained level of analysis, as we found a moderate agreement between tools as well as differences in the distribution of polarity labels assigned by the different tools in the text items of our dataset.

Our results suggest that SE-specific fine-tuning of sentiment analysis tools to the software engineering domain might not be enough to improve accuracy. Conversely, platform-specific tuning or retraining might be needed to adjust the model performance to the shifts in lexical semantics due to different platform jargon or conventions. Further fine-tuning or retraining of tools might be required with respect to different lengths of documents as we found that longer texts might convey mixed sentiment polarity. Finally, a sanity check of the tool performance against manual labeling should always be performed as classifiers might be built according to different operationalization of emotions, which do not necessarily match the intended research goals. Dealing with figurative language and implicit sentiment polarity in texts still represent open challenges for sentiment classifiers that should be address by future work in this field.

Acknowledgments We would like to thank Marika Redavid for her contribution in the early stage of this project. The computational work has been executed on the IT resources made available by two projects, ReCaS and PRISMA, funded by MIUR under the program PON R&C 2007–2013.

Funding Open access funding provided by Università degli Studi di Bari Aldo Moro within the CRUI-CARE Agreement.

References

Ahmed T, Bosu A, Iqbal A, Rahimi S (2017) Senticr: A customized sentiment analysis tool for code review interactions. In: Proceedings of the 32nd IEEE/ACM international conference on automated software engineering. IEEE Press, Piscataway, pp 106–111. ASE 2017, http://dl.acm.org/citation.cfm?id=3155562.3155579

Blaz CCA, Becker K (2016) Sentiment analysis in tickets for it support. In: Proceedings of the 13th international conference on mining software repositories. ACM, New York, pp 235–246,. MSR '16, https://doi.org/10.1145/2901739.2901781

Bosco C, Patti V, Bolioli A (2013) Developing corpora for sentiment analysis: the case of irony and senti-tut. IEEE Intell Syst 28(2):55–63

Calefato F, Lanubile F, Maiorano F, Novielli N (2018a) Sentiment polarity detection for software development. Empirical Softw Engg 23(3):1352–1382. https://doi.org/10.1007/s10664-017-9546-9

Calefato F, Lanubile F, Novielli N (2018b) How to ask for technical help? evidence-based guidelines for writing questions on stack overflow. Inform Softw Technol 94:186–207. https://doi.org/10.1016/j.infsof.2017.10.009, cited By 10

Carver JC (2010) Towards reporting guidelines for experimental replications: A proposal. In: 1st International workshop on replication in empirical software engineering research

Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) Smote: Synthetic minority over-sampling technique. J Artif Intell Res 16:321–357

Chen Z, Cao Y, Lu X, Mei Q, Liu X (2019) Sentimoji: An emoji-powered learning approach for sentiment analysis in software engineering. In: Proceedings of the 2019 27th ACM joint meeting on european software engineering conference and symposium on the foundations of software engineering, association for computing machinery, New York, NY, USA, ESEC/FSE 2019, pp 841–852. https://doi.org/10.1145/3338906.3338977

Cohen J (1968) Weighted kappa: Nominal scale agreement with provision for scaled disagreement or partial credit. Psychol Bull 70:213–220

Di Eugenio B, Glass M (2004) The kappa statistic: A second look. Computat Linguist 30(1):95–101. https://doi.org/10.1162/089120104773633402

Ebert F, Castor F, Novielli N, Serebrenik A (2018) Communicative intention in code review questions. In: 2018 IEEE international conference on software maintenance and evolution, ICSME 2018, madrid, spain, september 23-29, 2018, pp 519–523. https://doi.org/10.1109/ICSME.2018.00061

Gachechiladze D, Lanubile F, Novielli N, Serebrenik A (2017) Anger and its direction in collaborative software development. In: Proceedings of the 39th international conference on software engineering: New ideas and emerging results track. IEEE Press, Piscataway, NJ, USA, ICSE-NIER '17, pp 11–14. https://doi.org/10.1109/ICSE-NIER.2017.18

Gousios G (2013) The ghtorent dataset and tool suite. In: Proceedings of the 10th working conference on mining software repositories, IEEE Press, Piscataway, NJ, USA, MSR '13, pp 233–236. http://dl.acm.org/citation.cfm?id=2487085.2487132

Guzman E, Azócar D, Li Y (2014) Sentiment analysis of commit comments in github: An empirical study. In: Proceedings of the 11th working conference on mining software repositories, ACM, New York, NY, USA, MSR 2014, pp 352–355. https://doi.org/10.1145/2597073.2597118

Guzman E, Alkadhi R, Seyff N (2016) A needle in a haystack: What do twitter users say about software? In: 24Th IEEE international requirements engineering conference, RE 2016, beijing, china, september 12-16, 2016, pp 96–105. https://doi.org/10.1109/RE.2016.67

Imtiaz N, Middleton J, Girouard P, Murphy-Hill E (2018) Sentiment and politeness analysis tools on developer discussions are unreliable, but so are people. In: Proceedings of the 3rd international workshop on emotion awareness in software engineering, ACM, New York, NY, USA, SEmotion '18, pp 55–61. https://doi.org/10.1145/3194932.3194938

Islam MR, Zibran MF (2017) Leveraging automated sentiment analysis in software engineering. In: Proceedings of the 14th international conference on mining software repositories, IEEE Press, Piscataway, NJ, USA, MSR '17, pp 203–214. https://doi.org/10.1109/MSR.2017.9

Islam MR, Zibran MF (2018a) A comparison of software engineering domain specific sentiment analysis tools. In: 2018 IEEE 25th international conference on software analysis, evolution and reengineering (SANER), pp 487–491

Islam MR, Zibran MF (2018b) Deva: Sensing emotions in the valence arousal space in software engineering text. In: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, ACM, New York, NY, USA, SAC '18, pp 1536–1543. https://doi.org/10.1145/3167132.3167296

Jongeling R, Sarkar P, Datta S, Serebrenik A (2017) On negative results when using sentiment analysis tools for software engineering research. Empirical Softw Engg 22(5):2543–2584. https://doi.org/10.1007/s10664-016-9493-x

Kurtanovic Z, Maalej W (2018) On user rationale in software engineering. Requir Eng 23(3):357–379. https://doi.org/10.1007/s00766-018-0293-2

Lin B, Zampetti F, Bavota G, Di Penta M, Lanza M, Oliveto R (2018) Sentiment analysis for software engineering: How far can we go? In: 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE), pp 94–104. https://doi.org/10.1145/3180155.3180195

Lin B, Zampetti F, Bavota G, Penta MD, Lanza M (2019) Pattern-based Mining of Opinions in Q&A Websites. In: Proceedings of the 41st ACM/IEEE international conference on software engineering ICSE

Mäntylä M, Adams B, Destefanis G, Graziotin D, Ortu M (2016) Mining valence, arousal, and dominance: Possibilities for detecting burnout and productivity? In: Proceedings of the 13th international conference on mining software repositories, ACM, New York, NY, USA, MSR '16, pp 247–258. https://doi.org/10.1145/2901739.2901752

Mäntylä MV, Novielli N, Lanubile F, Claes M, Kuutila M (2017) Bootstrapping a lexicon for emotional arousal in software engineering. In: Proceedings of the 14th international conference on mining software repositories, IEEE Press, Piscataway, NJ, USA, MSR '17, pp 198–202. https://doi.org/10.1109/MSR.2017.47

Murgia A, Ortu M, Tourani P, Adams B, Demeyer S (2018) An exploratory qualitative and quantitative analysis of emotions in issue report comments of open source systems. Empir Softw Eng 23(1):521–564. https://doi.org/10.1007/s10664-017-9526-0

Novielli N, Calefato F, Lanubile F (2015) The challenges of sentiment detection in the social programmer ecosystem. In: Proceedings of the 7th International Workshop on Social Software Engineering, ACM, New York, NY, USA, SSE 2015, pp 33–40. https://doi.org/10.1145/2804381.2804387

Novielli N, Calefato F, Lanubile F (2018a) A gold standard for emotion annotation in stack overflow. In: Proceedings of the 15th international conference on mining software repositories, ACM, New York, NY, USA, MSR '18, pp 14–17. https://doi.org/10.1145/3196398.3196453

Novielli N, Girardi D, Lanubile F (2018b) A benchmark study on sentiment analysis for software engineering research. In: Proceedings of the 15th international conference on mining software repositories, ACM, New York, NY, USA, MSR '18, pp 364–375. https://doi.org/10.1145/3196398.3196403

Novielli N, Begel A, Maalej W (2019) Introduction to the special issue on affect awareness in software engineering. J Syst Softw 148:180–182. https://doi.org/10.1016/j.jss.2018.11.016

Novielli N, Calefato F, Lanubile F (2020) Love, joy, anger, sadness, fear, and surprise: Se needs special kinds of ai: A case study on text mining and se. IEEE Softw 37(3):86–91

Novielli N, Girardi D, Lanubile F (2020) Can we use se-specific sentiment analysis tools in a cross-platform setting? In: Proceedings of the 17th international conference on mining software repositories, ACM, New York, NY, USA, MSR '20. https://doi.org/10.1145/3379597.3387446

Ortu M, Murgia A, Destefanis G, Tourani P, Tonelli R, Marchesi M, Adams B (2016) The emotional side of software developers in jira. In: Proceedings of the 13th international conference on mining software repositories. ACM, pp 480–483

Pang B, Lee L (2008) Opinion mining and sentiment analysis. Found Trends Inf Retr 2(1-2):1–135. https://doi.org/10.1561/1500000011

Panichella S, Di Sorbo A, Guzman E, Visaggio CA, Canfora G, Gall HC (2015) How can i improve my app? classifying user reviews for software maintenance and evolution. In: Proceedings of the 2015 IEEE international conference on software maintenance and evolution (ICSME), IEEE Computer Society, Washington, DC, USA, ICSME '15, pp 281–290. https://doi.org/10.1109/ICSM.2015.7332474

Pletea D, Vasilescu B, Serebrenik A (2014) Security and emotion: Sentiment analysis of security discussions on github. In: Proceedings of the 11th working conference on mining software repositories, ACM, New York, NY, USA, MSR 2014, pp 348–351. https://doi.org/10.1145/2597073.2597117

Shull FJ, Carver JC, Vegas S, Juristo N (2008) The role of replications in empirical software engineering. Empir Softw Eng 13(2):211–218. https://doi.org/10.1007/s10664-008-9060-1

Sinha V, Lazar A, Sharif B (2016) Analyzing developer sentiment in commit logs. In: Proceedings of the 13th international conference on mining software repositories, ACM, New York, NY, USA, MSR '16, pp 520–523. https://doi.org/10.1145/2901739.2903501

Storey MA (2012) The evolution of the social programmer. In: Proceedings of the 9th IEEE working conference on mining software repositories, IEEE Press, MSR '12, p 140

Thelwall M, Buckley K, Paltoglou G (2012) Sentiment strength detection for the social web. J Am Soc Inf Sci Technol 63(1):163–173. https://doi.org/10.1002/asi.21662

Uddin G, Khomh F (2017) Opiner: an opinion search and summarization engine for apis. In: Proceedings of the 32nd IEEE/ACM International Conference on Automated Software Engineering, ASE 2017, Urbana, IL, USA, October 30 - November 03, 2017, pp 978–983. https://doi.org/10.1109/ASE.2017.8115715

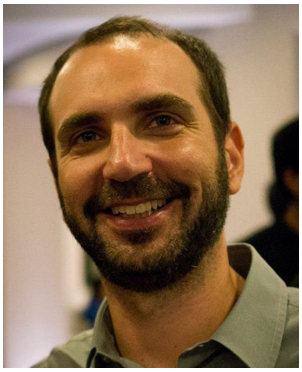
Viera AJ, Garrett JM (2005) Understanding interobserver agreement: the kappa statistic. Family Med 37(5):360–3

Weitzel L, Prati RC, Aguiar RF (2016) The Comprehension of Figurative Language: What Is the Influence of Irony and Sarcasm on NLP Techniques?. Springer International Publishing, Cham, pp 49–74. https://doi.org/10.1007/978-3-319-30319-2_3

Werner C, Tapuc G, Montgomery L, Sharma D, Dodos S, Damian D (2018) How angry are your customers? sentiment analysis of support tickets that escalate. In: 2018 1st International Workshop on Affective Computing for Requirements Engineering (AffectRE), pp 1–8. https://doi.org/10.1109/AffectRE.2018.00006

Zhang T, Xu B, Thung F, Haryono SA, Lo D, Jiang L (2020) Sentiment Analysis for Software Engineering: How Far Can Pre-trained Transformer Models Go? In: Proceedings of the 36th IEEE international conference on software maintenance and evolution (ICSME 2020), IEEE, New York, NY, USA, ICSME '20, pp 70–80. https://doi.org/10.1109/ICSME46990.2020.00017

Nicole Novielli is an Assistant Professor at the University of Bari, Italy. Her research interests lie at the intersection of software engineering and affective computing with a specific focus on mining emotions and opinions from developers communication traces and sensor-based recognition of developers cognitive and affective states.



Fabio Calefato is an Assistant Professor at the University of Bari, Italy, holding the national scientific qualification as an associate professor. He is the co-chair of ICSE-SCORE 2021 competition. He was the General Chair of ICGSE 2019 and co-organizer of the "Trust in Virtual Teams" workshop co-located with CSCW13 and SSE16 (Social Software Engineering) workshop co-located with FSE16. His research interests include human factors in software engineering, global software development, online communities, and software engineering for AI.

Filippo Lanubile is a Full Professor of computer science at the University of Bari, Italy, where he leads the Collaborative Development Research Group. His research interests include human factors in software engineering, global software development, online communities, and software engineering for AI. He is the General Chair of the 2020/2021 editions of the Int. Symp. on Empirical Software Engineering and Measurement (ESEM), the Chair of the Steering Committee of the Int. Conf. on Global Software Engineering (ICGSE), and the Chair of the Advisory Board of IEEE Software.

Alexander Serebrenik is a Full Professor of Social Software Engineering at Eindhoven University of Technology. His research goal is to facilitate evolution of software by taking into account social aspects of software development. He has co-authored a book "Evolving Software Systems" (Springer Verlag, 2014), and more than 100 scientific papers and articles. He has won several distinguished paper and distinguished review awards.