Choosing Your Weapons: On Sentiment Analysis Tools for Software Engineering Research

Robbert Jongeling
Eindhoven University of Technology
The Netherlands
r.m.jongeling@student.tue.nl

Subhajit Datta
Singapore University of Technology and Design
Singapore
subhajit.datta@acm.org

Alexander Serebrenik
Eindhoven University of Technology
The Netherlands
a.serebrenik@tue.nl

***Abstract*—Recent years have seen an increasing attention to social aspects of software engineering, including studies of emotions and sentiments experienced and expressed by the software developers. Most of these studies reuse existing sentiment analysis tools such as SentiStrength and NLTK. However, these tools have been trained on product reviews and movie reviews and, therefore, their results might not be applicable in the software engineering domain.**

In this paper we study whether the sentiment analysis tools agree with the sentiment recognized by human evaluators (as reported in an earlier study) as well as with each other. Furthermore, we evaluate the impact of the choice of a sentiment analysis tool on software engineering studies by conducting a simple study of differences in issue resolution times for positive, negative and neutral texts. We repeat the study for seven datasets (issue trackers and STACK OVERFLOW questions) and different sentiment analysis tools and observe that the disagreement between the tools can lead to contradictory conclusions.

I. INTRODUCTION

Sentiment analysis is "the task of identifying positive and negative opinions, emotions, and evaluations" [1]. Since its inception sentiment analysis has been subject of an intensive research effort and has been successfully applied e.g., to assist users in their development by providing them with interesting and supportive content [2], predict the outcome of an election [3] or movie sales [4]. The spectrum of sentiment analysis techniques ranges from identifying polarity (positive or negative) to a complex computational treatment of subjectivity, opinion and sentiment [5]. In particular, the research on sentiment polarity analysis has resulted in a number of mature and publicly available tools such as SentiStrength [6], Alchemy[1], Stanford NLP sentiment analyser [7] and NLTK [8].

Sentiment polarity analysis has been recently applied in the software engineering context to study commit comments in GitHub [9], GitHub discussions related to security [10], productivity in Jira issue resolution [11], activity of contributors in Gentoo [12] and evolution of developers' sentiments in the openSUSE Factory [13]. It has also been suggested when assessing technical candidates on the social web [14]. Not surprisingly, all the aforementioned software engineering studies reuse the existing sentiment polarity tools, e.g., Guzman et al. [15], [9] and Rousinopoulos et al. [13] use NLTK, while Garcia et al. [12], Pletea et al. [10] and Ortu et al. [11] opted for SentiStrength. While the reuse of the existing tools facilitated the application of the sentiment polarity analysis techniques in the software engineering domain, it also introduced a commonly recognized threat to validity of the results obtained: those tools have been trained on non-software engineering related texts such as movie reviews or product reviews and might misidentify (or fail to identify) polarity of a sentiment in a software engineering artefact such as a commit comment [9], [10].

Therefore, in this paper we focus on sentiment polarity analysis [1] and investigate to what extent are the software engineering results obtained from sentiment analysis depend on the choice of the sentiment analysis tool. For the sake of simplicity, from here on, instead of "existing sentiment polarity analysis tools" we talk about the "sentiment analysis tools". Specifically, we aim at answering the following questions:

- *RQ1*: To what extent do different sentiment analysis tools agree with emotions of software developers?
- *RQ2*: To what extent do different sentiment analysis tools agree with each other?

We have observed disagreement between sentiment analysis tools and the emotions of software developers but also between different sentiment tools themselves. However, disagreement between the tools does not *a priori* mean that sentiment analysis tools might lead to contradictory results in software engineering studies making use of these tools. Thus, we ask

- *RQ3*: Do different sentiment analysis tools lead to contradictory results in a software engineering study?

The remainder of this paper is organized as follows. In Section II we study agreement between the tools and the results of manual labeling, and between the tools themselves, i.e., *RQ1* and *RQ2*. In Section III we conduct a series of experiments based on the results of different sentiment analysis tools. We observe that conclusions one might derive using different tools diverge, casting doubt on their validity (*RQ3*). In Section IV we discuss related work and conclude in Section V.

Source code used to obtain the results of this paper has been made available on GitHub[2].

[1] http://www.alchemyapi.com/products/alchemylanguage/sentiment-analysis/

[2] https://github.com/RobbertJongeling/ICSME2015ERA

II. Agreement between Sentiment Analysis Tools

In this section we address *RQ1* and *RQ2*, i.e., to what extent do different sentiment analysis tools agree with emotions of software developers and to what extent do different sentiment analysis tools agree with each other. To perform the evaluation we use the manually labeled emotions dataset [16].

A. Methodology

1) Sentiment Analysis Tools: We have considered four sentiment analysis tools SentiStrength, Alchemy, Stanford NLP sentiment analyser and NLTK. SentiStrength and NLTK have been used in earlier software engineering studies. Moreover, SentiStrength had the highest average accuracy among fifteen Twitter sentiment analysis tools [17]. The Stanford NLP parses the text into sentences and performs a more advanced grammatical analysis as opposed to a simpler bag of words model used in NLTK. Alchemy provides several text processing APIs, including a sentiment analysis API which promises to work on very short texts (e.g., tweets) as well as relatively long texts (e.g., news articles).

SentiStrength assigns an integer value between 1 and 5 for the positivity of a text, p and similarly, a value between -1 and -5 for the negativity, n. In order to map these scores to a document-level sentiment (positive, neutral or negative) for an entire text fragment, we follow the approach by Thelwall et al. [18] 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 are removed from the datasets.

Alchemy API returns for a text fragment a status, a language, a score and a type. The score is in the range $(-1, 1)$, the type is the sentiment of the text and is based on the score. For negative scores, the type is negative, conversely for positive scores, the type is positive. For a score of 0, the type is neutral. We ignore texts with status "ERROR" or a non-English language.

NLTK returns for each text a probability of it being negative, one of it being neutral and one of it being positive. If the probability score for neutral is greater than 0.5, the text is considered neutral. Otherwise, it is considered to be the other sentiment with the highest probability [10]. To call NLTK, we use the API provided at text-processing.com.[3]

Stanford NLP breaks down the text into sentences and assigns each a sentiment score in the range $[0, 4]$, where 0 is very negative, 2 is neutral and 4 is very positive. We note that the tool may have difficulty breaking the text into sentences as comments sometimes include pieces of code or e.g. URLs. The tool does not provide a document-level score, to determine such a document-level sentiment we compute $-2*\#0-\#1+\#3+2*\#4$, where $\#0$ denotes the number of sentences with score 0, etc.. If this score is negative, neutral or positive, we consider the text to be negative, neutral or positive, respectively.

[3]API docs for NLTK sentiment analysis: http://text-processing.com/docs/sentiment.html

2) Manually-Labeled Software Engineering Data: As the "golden set" we use the data from a developer emotions study by Murgia et al. [16]. In this study, four evaluators manually labeled 392 comments with emotions "joy", "love", "surprise", "anger", "sadness" or "fear". Emotions "joy", "love" and "surprise" are taken as indicators of positive sentiments and "anger", "sadness" and "fear"—of negative sentiment.

We focus on consistently labeled comments. We consider the comment as positive if at least three evaluators have indicated a positive sentiment and no evaluators have indicated negative sentiments. Similarly, we consider the comment as negative if at least three evaluators have indicated a negative sentiment and no evaluators have indicated positive sentiments. We consider an evaluation neutral when an evaluator indicates no emotions. A text is then considered as neutral when three or more evaluators have evaluated it as neutral. Using these rules we can conclude that 295 comments have been labeled consistently: 24 negative, 54 positive and 217 neutral. The remaining $392 - 24 - 54 - 217 = 97$ comments from the study Murgia et al. [16] have been labeled with contradictory labels e.g. "fear" by one evaluator and "surprise" by another.

3) Evaluation Metrics: Since more than 73% of the comments have been manually labeled as neutral, i.e., the dataset is unbalanced, traditional metrics such as accuracy might be misleading [19]: indeed, accuracy of the straw man sentiment analysis predicting "neutral" for any comment can be easily higher than of any of the four tools. Therefore, we use the Adjusted Rand Index (ARI) [20].ARI measures the correspondence between two partitions of the same data: to answer the first research question we look for correspondence between the partition of the comments into positive, neutral and negative groups provided by the tool and the partition based on the manual labeling. Similarly, to answer the second research question we look for correspondence between partition of the comments into positive, neutral and negative groups provided by different tools. The expected value of ARI ranges for independent partitions is 0. The maximal value, obtained e.g., for identical partitions is 1, the closer the value of ARI to 1 the better the correspondence between the partitions.

B. Results and Discussion

None of the 295 consistently labeled comments produce SentiStrength results with $p = -n$ and $p \geq 4$. Hence, no comments are excluded from the evaluation of SentiStrength. Five comments produce the "ERROR" status with Alchemy; those comments have been excluded from consideration, i.e., Alchemy has been evaluated on 290 comments.

When comparing the partitions induced by the sentiment analysis tools with the partition based on the manual labeling, the highest ARI has been obtained for NLTK (0.239), followed by SentiStrength (0.113), Stanford NLP (0.108) and Alchemy (0.079). When comparing the partitions induced by the sentiment analysis tools with each other we obtain low values of ARI: Alchemy API and NLTK have the highest ARI (0.104), followed by SentiStrength and NLTK (0.090).

TABLE I
AGREEMENT BETWEEN THE MANUAL LABELING, NLTK [8] AND SENTISTRENGTH [6] ON 295 COMMENTS

		Manual				SentiStrength		
		neg	neu	pos		neg	neu	pos
NLTK	neg	19	51	11	neg	17	39	25
	neu	0	138	7	neu	15	96	34
	pos	5	28	36	pos	6	20	43

Based on this evaluation we select SentiStrength and NLTK for further evaluation: these tools show the highest (albeit still low) degrees of correspondence with the golden set. Moreover, these two sentiment analysis tools have one of the higher (albeit also low) agreements with each other, and, thus, we would expect the results obtained using those tools to be more consistent with each other than for pairs of tools with even lower agreement. Further details on the agreement between NLTK, SentiStrength and the manual labeling are shown in Table I.

C. Threats to Validity

As any empirical evaluation, the study presented in this section is subject to threats to validity. On top of the threats to validity inherent to the choice of the dataset used for evaluation and its construction [16] validity of our evaluation might have been affected by our decision to interpret emotions "joy", "love" and "surprise" as indicators of a positive sentiment and "anger", "sadness" and "fear"—as indicators of a negative sentiment. For instance, one might argue that not all surprises are positive. Therefore, we consider replication of this study on a manually labeled dataset as an important sent in the follow-up research.

Furthermore, the exact ways tools have been applied and the sentiment has been determined based on the tools' output, e.g., calculation of a document-level sentiment as $-2 * \#0 - \#1 + \#3 + 2 * \#4$ for Stanford NLP, might have affected validity of the conclusions.

III. IMPACT OF THE CHOICE OF SENTIMENT ANALYSIS TOOL

In Section II we have seen that not only is the agreement of the sentiment analysis tools with the manual labeling limited, but also that different tools do not necessarily agree with each other. However, this disagreement does not necessarily mean that conclusions based on application of these tools in the software engineering domain are affected by the choice of the tool. Therefore, next we address *RQ3* and discuss a simple set-up of a study aiming at understanding differences in response times for positive, neutral and negative texts. While we do not aim at replicating an existing study, we note that similar questions have been considered in the literature [11], [12].

A. Methodology

We study whether differences can be observed between response times (issue resolution times or question answering times) for positive, neutral and negative texts. We do not claim that the type of comment (positive, neutral or negative) is the main factor influencing response time: indeed, certain topics might be more popular than others and questions asked during the weekend might lead to higher resolution times. However, if different conclusions are derived for the same dataset when different sentiment analysis tools are used, then we can conclude that the disagreement between sentiment analysis tools affects validity of conclusions in the software engineering domain.

We repeat the study for seven different datasets: titles of issues of the ANDROID issue tracker, descriptions of issues of the ANDROID issue tracker, titles of issues of the Apache Software Foundation (ASF) issue tracker, descriptions of issues of the ASF issue tracker, descriptions of issues of the GNOME issue tracker, titles of the GNOME-related STACK OVERFLOW questions and bodies of the GNOME-related STACK OVERFLOW questions. As opposed to the ANDROID dataset, GNOME issues do not have titles. For each dataset we determine the sentiment using NLTK and SentiStrength. Moreover, we repeat each study on the subset of texts where NLTK and SentiStrength agree.

1) Datasets:

a) ANDROID *Issue Tracker:* A dataset of 20,169 issues from the ANDROID issue tracker was part of the mining challenge of MSR 2012 [21]. Excluding issues without a closing date, as well as those with *bug_status* "duplicate", "spam" or "usererror", results in the dataset with 5,216 issues.

We analyze the sentiment of the issue titles and descriptions. Five issues have an *undetermined* description sentiment. We remove these issues from further analysis on the titles and the descriptions. To measure the response time, we calculate the time difference in seconds between the opening (*openedDate*) and closing time (*closedOn*) of an issue.

b) GNOME *Issue Tracker:* The GNOME project issue tracker dataset containing 431,863 issues was part of the 2009 MSR mining challenge. Similarly to the ANDROID dataset, we have looked only at issues with a value for field *bug_status* of `resolved`. In total 367,877 have been resolved. We analyze the sentiment of the short descriptions of the issues (*short_desc*) and calculate the time difference in seconds between the creation and closure of each issue. Recall that as opposed to the ANDROID dataset, GNOME issues do not have titles.

c) GNOME-*Related* STACK OVERFLOW *Discussions:* We use the StackExchange online data explorer[4] to obtain all STACK OVERFLOW posts created before May 20, 2015, tagged `gnome` and having an accepted answer. For all 410 collected posts, we calculate the time difference in seconds between the creation of the post and the creation of the accepted answer. Before applying a sentiment analysis tool we remove HTML formatting from the titles and bodies of posts. In the results, we refer to the `body` of a post as its description.

d) ASF *Issue Tracker:* We use a dataset containing data from the ASF issue tracking system JIRA. This dataset was

[4]http://data.stackexchange.com/

collected by Ortu et al. [11] and contains 701,002 issue reports. We analyze the sentiments of the titles and the descriptions of 95,667 issue reports that have a non-null resolved date, a *resolved* status and the resolution value being *Fixed*.

2) Statistical Analysis: To answer our research questions we need to compare distributions of response times corresponding to issues/questions bearing positive, neutral and negative sentiments. Traditionally, a comparison of multiple groups follows a two-step approach: first, a global null hypothesis is tested, then multiple comparisons are used to test sub-hypotheses pertaining to each pair of groups. The first step is commonly carried out by means of ANOVA or its non-parametric counterpart, the Kruskal-Wallis one-way analysis of variance by ranks. The second step uses the t-test or the rank-based Wilcoxon-Mann-Whitney test, with correction for multiple comparisons, e.g., Bonferroni correction. Unfortunately, the global test null hypothesis may be rejected while none of the sub-hypotheses are rejected, or vice versa [22]. Moreover, simulation studies suggest that the Wilcoxon-Mann-Whitney test is not robust to unequal population variances, especially in the case of unequal sample sizes [23]. Therefore, one-step approaches are preferred: these should produce confidence intervals which always lead to the same test decisions as the multiple comparisons. We use the $\widetilde{\mathbf{T}}$-procedure [24] for Tukey-type contrasts [25], the probit transformation and the traditional 5% family error rate (cf. [26], [27]).

The results of the $\widetilde{\mathbf{T}}$-procedure are series of probability estimates $p(a, b)$ with the corresponding p-values, where a and b are selected from "positive", "neutral" or "negative". The probability estimate $p(a, b)$ is interpreted as follows: if the corresponding p-value exceeds 5% then no evidence has been found for difference in response times corresponding to categories a and b. If, however, the corresponding p-value does not exceed 5% and $p(a, b) > 0.5$ then response times in category b tends to be larger than those in category a. Finally, if the corresponding p-value does not exceed 5% and $p(a, b) < 0.5$ then response times in category a tends to be larger than those in category b.

We opt for comparison of distributions rather than a more elaborate statistical modeling (cf. [11]) since it allows for an easy comparison of the results obtained for different tools.

B. Results

Results of our study are summarized in Table II. For the sake of readability the relations found are aligned horizontally. Relations found for NLTK, SentiStrength *and* the intersection are typeset in boldface. We also report the number of issues/questions recognized as negative, neutral or positive.

We observe that NLTK and SentiStrength agree only on one relation for the ANDROID, i.e., that issues with the neutral sentiment tend to be resolved more slowly than issues formulated in a more positive way. We also observe that for GNOME and ASF the tools agree that the issues with the neutral sentiment are resolved faster than issues with the positive sentiment, i.e., the results for GNOME and ASF are opposite from those for ANDROID.

TABLE II
COMPARISON OF NLTK AND SENTISTRENGTH (SS). THRESHOLDS FOR STATISTICAL SIGNIFICANCE: 0.05 (*), 0.01 (**), 0.001 (***).

	NLTK neg-neu-pos	SS neg-neu-pos	NLTK ∩ SS neg-neu-pos
		ANDROID	
t[5]	1,230-3,588-398 ∅	1,417-3,415-384 ∅	396-2,381-36 ∅
d	2,690-1,657-869 neu $>$ neg*** **neu** $>$ **pos****	1,684-2,435-1,182[6] **neu** $>$ **pos**** neg $>$ pos***	893-712-299 neu $>$ neg* **neu** $>$ **pos***** neg $>$ pos*
		GNOME	
d	54,032-291,906-20,380 **neg** $>$ **neu***** **pos** $>$ **neu***** pos $>$ neg***	58,585-293,226-14,507 **neg** $>$ **neu***** **pos** $>$ **neu***** neg $>$ pos***	16,829-24,2780-1,785 **neg** $>$ **neu***** **pos** $>$ **neu*****
		STACK OVERFLOW	
t	84-285-41 ∅	53-330-27 ∅	16-240-8 ∅
d	249-71-90 ∅	90-183-137 neg $>$ pos*	62-35-42 ∅
		ASF	
t	19,367-67,948-8,348[7] pos $>$ neg*	24,141-62,016-9,510 neg $>$ neu** pos $>$ neu*** pos $>$ neg***	6,450-44,818-1,106 pos $>$ neu**
d[8]	30,339-42,540-13,129[9] neg $>$ neu*** **pos** $>$ **neu*****	29,021-41,043-15,971[10] **pos** $>$ **neu***** pos $>$ neg***	10,989-20,940-3,814 neg $>$ neu*** **pos** $>$ **neu***** pos $>$ neg***

Further inspection reveals that differences between NLTK and SentiStrength led to relations "neu $>$ neg" and "neg $>$ pos" to be discovered in ANDROID issue descriptions only by one of the tools and not by the other. In the same way, "pos $>$ neg" on the ASF descriptions data can be found only by SentiStrength. It is also surprising that while "pos $>$ neg" has been found for the ASF titles data both by NLTK and by SentiStrength, it cannot be found when one restricts the attention to the issues where the tools agree. Finally, contradictory results have been obtained for GNOME issue descriptions: while the NLTK-based analysis suggests that the positive issues are resolved more slowly than the negative ones, the SentiStrength-based analysis suggests the opposite.

C. Discussion

Our results suggest the choice of the sentiment analysis tool affects the conclusions one might derive when analysing differences in the response times, casting doubt on the validity of those conclusions. We conjecture that the same might be observed for any kind of software engineering studies dependent on off-the-shelf sentiment analysis tools. A more careful

[5] t denotes titles, d denotes descriptions
[6] Sentiment of 5 issues was "undetermined".
[7] The tool reported an error for 4 issues
[8] 9,620 empty descriptions where not included in this analysis
[9] The tool reported an error for 39 issues
[10] Sentiment of 12 issues was "undetermined".

sentiment analysis for software engineering texts is therefore needed: e.g., one might consider training more general purpose machine learning tools such as Weka [28] or RapidMiner on software engineering data. Indeed, the need for text-analysis tools specifically targeting texts related to software engineering has been recognized in the past and led to creation of e.g., software-engineering-specific dictionary of synonyms [29].

IV. Related work

The existence of multiple sentiment analysis tools has lead to a number of comparison studies [17], [30]. However, to the best of our knowledge these studies have always used general social media data rather than software engineering data.

Observations similar to those we made for sentiment analysis tools, have been made in the past for software metric calculators [31] and code smell detection tools [32]. Similarly to our findings, disagreement between the tools was observed.

V. Conclusions

In this paper we have studied the impact of the choice of a sentiment analysis tool when conducting software engineering studies. We have observed that not only do the tools considered not agree with the manual labeling, but also they do not agree with each other, and that this disagreement can lead to contradictory conclusions.

Our results suggest a need for sentiment analysis tools specially targeting the software engineering domain.

Acknowledgements

We are very grateful to Alessandro Murgia and Marco Ortu for making their datasets available for our study, and to Bogdan Vasilescu for providing feedback on the preliminary version of this manuscript.

References

[1] T. Wilson, J. Wiebe, and P. Hoffmann, "Recognizing contextual polarity in phrase-level sentiment analysis," in *Human Language Technology and Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics, 2005, pp. 347–354.

[2] T. Honkela, Z. Izzatdust, and K. Lagus, "Text mining for wellbeing: Selecting stories using semantic and pragmatic features," in *Artificial Neural Networks and Machine Learning, Part II*, ser. LNCS. Springer, 2012, vol. 7553, pp. 467–474.

[3] A. Tumasjan, T. O. Sprenger, P. G. Sandner, and I. M. Welpe, "Predicting elections with twitter: What 140 characters reveal about political sentiment." in *International AAAI Conference on Weblogs and Social Media*, 2010, pp. 178–185.

[4] G. Mishne and N. S. Glance, "Predicting movie sales from blogger sentiment." in *AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs*, 2006, pp. 155–158.

[5] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Foundations and Trends in Information Retrieval*, vol. 2, no. 1-2, pp. 1–135, 2007.

[6] M. Thelwall, K. Buckley, G. Paltoglou, D. Cai, and A. Kappas, "Sentiment in short strength detection informal text," *J. Am. Soc. Inf. Sci. Technol.*, vol. 61, no. 12, pp. 2544–2558, Dec. 2010.

[7] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. D. Manning, A. Ng, and C. Potts, "Recursive deep models for semantic compositionality over a sentiment treebank," in *Empirical Methods in Natural Language Processing*. Ass. for Comp. Linguistics, October 2013, pp. 1631–1642.

[8] S. Bird, E. Loper, and E. Klein, *Natural Language Processing with Python*. O'Reilly Media Inc., 2009.

[9] E. Guzman, D. Azócar, and Y. Li, "Sentiment analysis of commit comments in github: An empirical study," in *MSR*. New York, NY, USA: ACM, 2014, pp. 352–355.

[10] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: Sentiment analysis of security discussions on github," in *MSR*. New York, NY, USA: ACM, 2014, pp. 348–351.

[11] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, and R. Tonelli, "Are bullies more productive? empirical study of affectiveness vs. issue fixing time," in *MSR*, 2015.

[12] D. Garcia, M. S. Zanetti, and F. Schweitzer, "The role of emotions in contributors activity: A case study on the gentoo community," in *International Conference on Cloud and Green Computing*, Sept 2013, pp. 410–417.

[13] A.-I. Rousinopoulos, G. Robles, and J. M. González-Barahona, "Sentiment analysis of Free/Open Source developers: preliminary findings from a case study," *Revista Eletrônica de Sistemas de Informação*, vol. 13, no. 2, pp. 6:1–6:21, 2014.

[14] A. Capiluppi, A. Serebrenik, and L. Singer, "Assessing technical candidates on the social web," *Software, IEEE*, vol. 30, no. 1, pp. 45–51, Jan 2013.

[15] E. Guzman and B. Bruegge, "Towards emotional awareness in software development teams," in *Joint Meeting on Foundations of Software Engineering*. New York, NY, USA: ACM, 2013, pp. 671–674.

[16] A. Murgia, P. Tourani, B. Adams, and M. Ortu, "Do developers feel emotions? an exploratory analysis of emotions in software artifacts," in *MSR*. New York, NY, USA: ACM, 2014, pp. 262–271.

[17] A. Abbasi, A. Hassan, and M. Dhar, "Benchmarking twitter sentiment analysis tools," in *International Conference on Language Resources and Evaluation*. Reykjavik, Iceland: ELRA, may 2014, pp. 823–829.

[18] M. Thelwall, K. Buckley, and G. Paltoglou, "Sentiment strength detection for the social web," *J. Am. Soc. Inf. Sci. Technol.*, vol. 63, no. 1, pp. 163–173, Jan. 2012.

[19] G. E. A. P. A. Batista, A. C. P. L. F. Carvalho, and M. C. Monard, "Applying one-sided selection to unbalanced datasets," in *Mexican International Conference on Artificial Intelligence: Advances in Artificial Intelligence*. London, UK, UK: Springer-Verlag, 2000, pp. 315–325.

[20] J. M. Santos and M. Embrechts, "On the use of the adjusted rand index as a metric for evaluating supervised classification," in *International Conference on Artificial Neural Networks*, ser. LNCS. Springer, 2009, vol. 5769, pp. 175–184.

[21] E. Shihab, Y. Kamei, and P. Bhattacharya, "Mining challenge 2012: The Android platform," in *MSR*, June 2012, pp. 112–115.

[22] K. R. Gabriel, "Simultaneous test procedures—some theory of multiple comparisons," *ANN MATH STAT*, vol. 40, no. 1, pp. 224–250, 1969.

[23] E. Brunner and U. Munzel, "The Nonparametric Behrens-Fisher Problem: Asymptotic Theory and a Small-Sample Approximation," *Biometrical Journal*, vol. 42, no. 1, pp. 17–25, 2000.

[24] F. Konietschke, L. A. Hothorn, and E. Brunner, "Rank-based multiple test procedures and simultaneous confidence intervals," *Electronic Journal of Statistics*, vol. 6, pp. 738–759, 2012.

[25] J. W. Tukey, "Quick and dirty methods in statistics, part II, Simple analysis for standard designs," in *American Society for Quality Control*, 1951, pp. 189–197.

[26] B. Vasilescu, A. Serebrenik, M. Goeminne, and T. Mens, "On the variation and specialisation of workload – a case study of the Gnome ecosystem community," *Empirical Software Engineering*, vol. 19, no. 4, pp. 955–1008, 2013.

[27] S. Wang, D. Lo, B. Vasilescu, and A. Serebrenik, "EnTagRec: An enhanced tag recommendation system for software information sites," in *ICSME*. IEEE, 2014, pp. 291–300.

[28] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18, Nov. 2009.

[29] M. J. Howard, S. Gupta, L. L. Pollock, and K. Vijay-Shanker, "Automatically mining software-based, semantically-similar words from comment-code mappings," in *MSR*, T. Zimmermann, M. D. Penta, and S. Kim, Eds. IEEE Computer Society, 2013, pp. 377–386.

[30] P. Gonçalves, M. Araújo, F. Benevenuto, and M. Cha, "Comparing and combining sentiment analysis methods," in *ACM Conference on Online Social Networks*. New York, NY, USA: ACM, 2013, pp. 27–38.

[31] H. Barkmann, R. Lincke, and W. Löwe, "Quantitative evaluation of software quality metrics in open-source projects," in *IEEE International Workshop on Quantitative Evaluation of large-scale Systems and Technologies*, May 2009, pp. 1067–1072.

[32] F. A. Fontana, E. Mariani, A. Morniroli, R. Sormani, and A. Tonello, "An experience report on using code smells detection tools," in *ICST Workshops*. IEEE, March 2011, pp. 450–457.