

Two Datasets for Sentiment Analysis in Software Engineering

ABSTRACT

Sentiment analysis is attracting more and more attention in the software engineering domain, where it is used, for example, to assess the sentiments in app reviews, or to analyze developers' emotions. However, previous studies have disclosed that most existing sentiment analysis tools do not achieve satisfactory performance when used to identify sentiments in software-related contexts. At the same time, there are currently not many ready-to-use datasets reporting the sentiment expressed in sentences extracted from software-related artifacts. This might lead to biased and insufficient validations of techniques designed to support sentiment analysis in software engineering.

In this paper we present two datasets with labeled sentiments: (i) a dataset of 341 sentences extracted from mobile app reviews; and (ii) a dataset of 1,500 sentences from Stack Overflow, for which we manually labeled all the ~20k syntactic nodes composing them, which can be useful to leverage deep learning techniques.

CCS CONCEPTS

• **Information systems** → **Sentiment analysis**; • **Software and its engineering** → *Software libraries and repositories*;

KEYWORDS

sentiment analysis, Stack Overflow, app reviews

1 INTRODUCTION

Recently, researchers have shown an increased interest in techniques supporting the automatic mining of opinions from online sources [14]. These techniques are often used to identify the mood and feelings expressed in textual reviews, such as those posted by customers on online stores. In this context, a specific contribution is represented by sentiment analysis [14] techniques, which aim to identify affective states and subjective opinions reported in sentences. In its basic usage scenario, sentiment analysis is used to classify customers' written opinions as negative, neutral, or positive.

In addition to the wide use in customer reviews, sentiment analysis has also been adopted by the software engineering research community, since sentiment is commonly expressed in software artifacts such as commit messages, issues, and app reviews. For example, sentiment analysis has been used to detect the psychological state of developers [5, 6, 8, 16], as developers' emotions could impact their productivity, task completion quality, and job satisfaction [18]. Studies have also applied sentiment analysis to (i) identify the polarity of app reviews, with the aim of supporting the evolution of mobile apps [4, 7, 9, 15], and (ii) classify the sentiment expressed in Q&A websites, such as Stack Overflow, to assess code quality [17] and identify problematic API design features [22].

Most of the previous work adopted sentiment analysis tools such as SentiStrength¹, NLTK², and Stanford CoreNLP³ without any customization aimed at adapting them to the specific context in which they are used. However, these tools have not been designed to work on software-related textual documents (e.g., the sentiment analysis component of Stanford CoreNLP [19] has been trained on movie reviews). This "out-of-the-box usage" of sentiment analysis tools has been recently criticized, with researchers highlighting the poor performance of these tools when applied in a context different from the one they have been designed and/or trained for [11, 13, 20].

Tourani *et al.* [20] used SentiStrength to extract sentiment information from user and developer mailing lists and found that SentiStrength achieved a very low precision, i.e., 29.56% for positive sentences and 13.1% for negative sentences. Jongeling *et al.* [11] examined the performance of four widely used sentiment analysis tools: SentiStrength, NLTK, Stanford CoreNLP, and AlchemyAPI. The results indicate that none of these tools can provide accurate predictions of sentiment expressed in software-related texts. Given these results, and in order to achieve good performance on software-related datasets, it is necessary to develop new sentiment analysis techniques and/or customize existing ones. On this line of research, Islam and Zibran [10] developed SentiStrength – SE, which adapts SentiStrength by creating a domain dictionary tailored for software-related datasets.

While further development and customization of sentiment analysis to software engineering applications is needed and welcome, their evaluation is still challenging and requires manually validated datasets reporting the sentiment expressed in sentences. For example, both Jongeling *et al.* [11] and Islam and Zibran [10] used the "golden set" of emotion-annotated dataset of JIRA issue comments to assess the performance of the experimented tools. However, this dataset has two major issues when used in the context of sentiment analysis detection. First, it was originally proposed for emotion (i.e., *joy, love, sadness, anger, fear, surprise*) detection. Thus, the dataset requires extra processing to make the dataset ready for sentiment inspection: While it is possible to automatically map certain emotions to sentiment (e.g., "joy, love" to "positive"), it still requires human effort to label the sentiment of sentences expressing *surprise* or no emotion. Second, the dataset is only representative of emotions expressed in a very specific environment such as the one represented by the JIRA issue tracker, where the range of emotions expressed in this context might be limited.

Given the strong need for datasets that can be used to (re-)train and to evaluate sentiment analysis tools in the software engineering context, in this paper we present two new datasets: (i) a dataset of 341 sentiment-labeled sentences extracted from mobile app reviews; and (ii) a dataset of 1,500 sentences along with ~40k nodes

¹<http://sentistrength.wlv.ac.uk/>

²<http://www.nltk.org/>

³<https://nlp.stanford.edu/sentiment/>

Table 1: Sample sentiment annotated sentences.

	positive	neutral	negative
app reviews	excellent app no complaints at all but still waiting for urdu localization	this is funny but creepy at the same time	very poor it's a video app that refuses to play videos.
Stack Overflow	The simplest solution would be to use Java 8's Date/Time API.	Lastly, if you use tomcat, you might want to clear the work directory with the temporary compiler outputs of the jsps you created.	I am not sure why I can not get it to work and have been struggling with it for a while.

composing them (all of them labeled with sentiment), extracted from Stack Overflow discussions.

The availability of our two datasets can (i) strengthen the evaluation of sentiment analysis tools in the software engineering context, and (ii) allow researchers to train new tools proposed in the software engineering community.

2 DATASET

This section provides information about the datasets and the process we adopted to build them. Also, we present the sentiment analysis results of several commonly used tools applied to our datasets.

2.1 Dataset of App Reviews

The polarity of app reviews has been studied by researchers to support app evolution. Here we describe a dataset containing 341 sentiment-annotated sentences from mobile app reviews.

Villarroel *et al.* [21] provides a dataset of 3k mobile app reviews, which are manually classified into categories based on the main information they contain. The classified categories include *bug reporting*, *suggestion for new feature*, *request for improving non-functional requirements* (e.g., performance of the app), and *other* (i.e., not belonging to any of the previous categories). Based on this dataset, we randomly selected 341 reviews. During the selection process, the proportion of reviews from each category remains same as in the original population (e.g., if 50% of the 3k reviews belonged to the “other” category, we randomly selected 50% of our sample from that category). The 341 selected reviews represent a statistically significant sample with 95% confidence level $\pm 5\%$ confidence interval.

After the sentence selection, we manually labeled the sentiment of each review. We used three scores to represent the sentiment: 1 for positive, 0 for neutral, and -1 for negative. The labeling process was performed by two of the authors (from now on, evaluators). When a conflict occurred, a third evaluator was involved to give the third sentiment score, and the majority vote was used as the final result. In total, we solved 51 conflict cases, and the dataset contains 130 positive, 25 neutral, and 186 negative reviews. Table 1 reports examples of sentences from our dataset.

2.2 Dataset of Stack Overflow Discussions

Sentiment expressed in Stack Overflow discussions has been leveraged by researchers for various purposes, such as evaluating code quality [17] and identifying problematic API design features [22]. Here we present a dataset containing 1,500 sentiment-annotated sentences extracted from Stack Overflow discussions.

Since Stack Overflow covers many topics, we decided to focus our dataset on discussions about Java libraries or APIs, in order to make the collected set of sentences more coherent. We extracted all discussions which are tagged with Java and contain one of the following words: *library/libraries* or *API(s)*, from the Stack Overflow dump (dated July 2017).

In total, we collected 276,629 discussions from which we extracted 5,073,452 sentences by using the Stanford CoreNLP toolkit [12]. Due to the huge amount of extracted sentences, it was impossible to label all of them. Thus, we randomly selected 1,500 sentences.

Since deep learning has achieved promising results in the natural language processing domain, instead of only labeling the whole sentences, we decided to build a dataset which can be also used to train deep learning models, such as the one implemented in Stanford CoreNLP. This model leverages a Recursive Neural Network (RNN), and was originally trained on the sentiment of movies’ reviews. To train the RNN, it is not sufficient to simply provide the polarity for a sentence, since the model needs to learn how sentences are grammatically built on top of positive/negative terms (i.e., the polarity of all intermediate nodes composing a sentence is needed). An example can be found in Fig. 1.

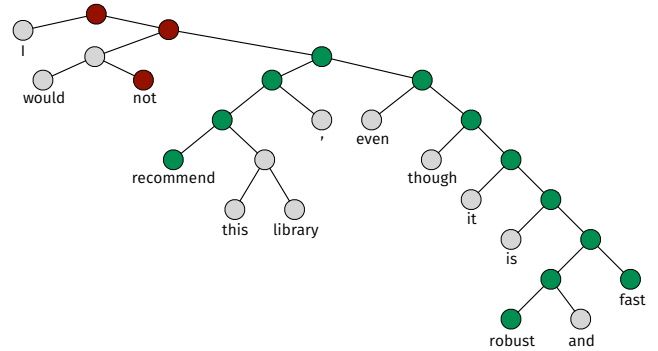


Figure 1: Example of the labeling needed to build the Stanford CoreNLP training set.

Gray nodes represent (sequences of) words having a *neutral* polarity, red ones indicate *negative* sentiment, and green ones *positive* sentiment. Overall, the sentence has a negative sentiment (see the root of the tree in Fig. 1), despite the presence of several positive terms (the tree’s leaves) and intermediate nodes. In practice, we labeled the sentiment of all gray, red, and green nodes for each sentence. To use this sentence composed of 14 words as part of

Table 2: Evaluation results for sentiment analysis tools applied in software engineering domain. Best results are in bold.

dataset	tool	# correct prediction	positive precision	positive recall	neutral precision	neutral recall	negative precision	negative recall
App reviews	<i>SentiStrength</i>	213	0.745	0.866	0.113	0.320	0.815	0.338
	<i>NLTK</i>	184	0.751	0.812	0.093	0.440	1.000	0.169
	<i>Stanford CoreNLP</i>	237	0.831	0.715	0.176	0.240	0.667	0.754
	<i>SentiStrength-SE</i>	201	0.741	0.817	0.106	0.400	0.929	0.300
	<i>Senti4SD</i>	218	0.712	0.866	0.098	0.200	0.813	0.400
	<i>SentiCR</i>	46	0	0	0.063	0.760	0.675	0.208
Stack Overflow	<i>SentiStrength</i>	1043	0.200	0.359	0.858	0.772	0.397	0.433
	<i>NLTK</i>	1168	0.317	0.244	0.815	0.941	0.625	0.084
	<i>Stanford CoreNLP</i>	604	0.231	0.344	0.884	0.344	0.177	0.837
	<i>SentiStrength-SE</i>	1170	0.312	0.221	0.826	0.930	0.500	0.185
	<i>Senti4SD</i>	1154	0.275	0.313	0.832	0.903	0.638	0.208
	<i>SentiCR</i>	1181	0	0	0.803	0.971	0.400	0.135

the training set of the RNN, we had to label the sentiment of all 27 nodes depicted in Fig. 1.

The labeling process was performed by five of the authors (from now on, evaluators) and supported by a Web application we built. The Web app randomly showed to each evaluator nodes (extracted from a sentence) to label with a sentiment going from -2 to +2, with -2 indicating strong negative, -1 weak negative, 0 neutral, +1 weak positive, and +2 strong positive score. The choice of the five-levels sentiment classification was driven by the observation of the movie reviews training set made publicly available by the authors of the Stanford CoreNLP [19] sentiment analysis tool⁴. Note that a node to evaluate could be a whole sentence, an intermediate node (thus, a sub-sentence), or a leaf node (*i.e.*, a single word). The Web application made sure to have two evaluators for each node, thus reducing the subjectivity bias.

This process, which took ~90 working hours of manual labeling, resulted in the total labeling of the sentiment polarity for 39,924 nodes (*i.e.*, 19,962 nodes extracted from the 1500 sentences \times 2 evaluators per node).

Once the labeling was completed, two of the authors worked on conflicts resolution (*i.e.*, cases in which two evaluators assigned a different sentiment to the same node). Concerning the complete sentences, there are 279 conflicts (18.6% of the labeled sentences). We fixed all of them to make our dataset usable as a ground truth to evaluate sentiment analysis tools. Concerning the intermediate/leaf nodes, we had a total of 2,199 conflicts (11.9% of the labeled intermediate/leaf nodes). We decided to only manually solve 123 strong conflicts, meaning those for which there was a score difference ≥ 2 (*e.g.*, one of the evaluators gave 1, the other one -1), while we kept the 2,076 having a conflict of only one point. Indeed, slight variations of the assigned sentiment (*e.g.*, one evaluator gave 1 and the other 2) are expected due to the subjectivity of the task. The final sentiment score was assigned with $\text{round}[(s_1 + s_2)/2]$, where round is the rounding function to the closest integer value and s_i is the sentiment assigned by the i^{th} evaluator. In total, the dataset

has 178 positive, 1,191 neutral, and 131 negative sentences. Table 1 demonstrates some example sentences from this dataset.

2.3 Sentiment Analysis Tool Performance on the Datasets

We examined the performance of state-of-the-art sentiment analysis tools on our datasets. The tools we experimented include the previously mentioned *SentiStrength*, *NLTK*, *Stanford CoreNLP*, and *SentiStrength-SE*. Also, we consider two very recently proposed techniques: *Senti4SD* [3] and *SentiCR* [1].

Table 2 presents the evaluation results for these tools when applied to our datasets. As it can be seen, no tool achieves satisfactory results, especially for what concerns the classification of sentences reporting positive and negative sentiment. This clearly highlights the need for more research aimed at developing techniques and tools tailored for the software engineering domain, and we believe that our dataset can represent a challenging benchmark to work with.

2.4 Accessibility and Reproduction of Datasets

The datasets can be downloaded from <https://sentidata.github.io/>. Since the dataset was built via manual labeling, source code to recreate the data cannot be provided. However, we provide the Web app we use as support during the labeling process.

3 POTENTIAL RESEARCH DIRECTIONS

The main purpose of our datasets is to provide researchers with a wider choice when evaluating the performance of sentiment analysis tools. That is, researchers interested in proposing novel sentiment analysis techniques will be able to better assess their performance by running more solid empirical evaluations.

On top of that, our Stack Overflow dataset is the first one available in the software engineering literature that allows training a neural network thanks to the labeling of all nodes composing each sentence. This can be used for example to re-train the Stanford CoreNLP model, as we recently did [2] showing the low performance that also the re-trained RNN model achieves on our datasets.

⁴https://nlp.stanford.edu/sentiment/trainDevTestTrees_PTB.zip

In general, the building of effective sentiment analysis tools tailored for the software engineering field can represent the stepping stone for several interesting research directions, including e.g., the development of recommender systems exploiting opinions mined from the Web to support design decisions (e.g., which API to use in a given context).

4 THREATS AND POSSIBLE IMPRECISSIONS

As every manually built dataset, our data collection process is subject to several threats to validity.

We only labeled a small subset of sentences present in the two repositories (i.e., app reviews and Stack Overflow) we used. This was needed to limit the manual effort that, still, was substantial. Thus, while we considered statistically significant samples, we cannot guarantee that our samples are representative of the whole population.

Since perceiving sentiment is a subjective activity, our dataset can obviously suffer from subjectivity issues. To alleviate this threat, we (i) made sure that each sentence was labeled by at least two different persons, and (ii) involved a third person to solve conflicts, when needed. This also leads us to the next point.

While we observed that the performance of existing sentiment analysis tools is poor when applied to our datasets, it must be clarified that we should not expect 100% of accuracy by these tools. For example, in our manual evaluation, out of the 1,500 Stack Overflow sentences we manually labeled, there were 279 cases of disagreement (18.6%). This means that even humans are not able to agree about the sentiment expressed in a given sentence. Hence, it is hard to expect that an automated tool can do any better. Still, advances are needed to make sentiment analysis tools usable in the software engineering domain.

5 CONCLUSION

We described two new datasets for evaluating sentiment analysis approaches in the software engineering domain. The dataset feature manually-labeled sentences with a positive, neutral, or negative sentiment. The first dataset is composed of 341 sentences from mobile app reviews, while the second contains 1,500 sentences (composed of ~20k nodes in total) from Stack Overflow discussions. Considering the low availability of datasets to assess sentiment identification performance on software-related texts, our datasets can boost the quality of empirical evaluations in this field.

As future work, we would like to include additional datasets covering sentences extracted from other types of repositories, such as mailing lists, IRC chats, and commit messages.

REFERENCES

- [1] Toufique Ahmed, Amiangshu Bosu, Anindya Iqbal, and Shahram Rahimi. 2017. SentiCR: a customized sentiment analysis tool for code review interactions. In *Proceedings of ASE 2017 (32nd IEEE/ACM International Conference on Automated Software Engineering)*. IEEE Press, 106–111.
- [2] Anonymized. 2018. Anonymized. In *Proceedings of Anonymized*. to appear.
- [3] Fabio Calefato, Filippo Lanubile, Federico Maiorano, and Nicole Novielli. 2017. Sentiment Polarity Detection for Software Development. *Empirical Software Engineering* (2017), 1–31.
- [4] L. V. G. Carreño and K. Winblad. 2013. Analysis of user comments: an approach for software requirements evolution. In *Proceedings of ICSE 2013 (35th International Conference on Software Engineering)*. IEEE press, 582–591.
- [5] Daviti Gachechiladze, Filippo Lanubile, Nicole Novielli, and Alexander Serebrenik. 2017. Anger and Its Direction in Collaborative Software Development. In *Proceedings of ICSE 2017 (39th IEEE/ACM International Conference on Software Engineering)*. IEEE, 11–14.
- [6] David Garcia, Marcelo Serrano Zanetti, and Frank Schweitzer. 2013. The Role of Emotions in Contributors Activity: A Case Study on the GENTOO Community. In *Proceedings of CGC 2013 (3rd International Conference on Cloud and Green Computing) (CGC '13)*. 410–417.
- [7] Michael Goul, Olivera Marjanovic, Susan Baxley, and Karen Vizecky. 2012. Managing the Enterprise Business Intelligence App Store: Sentiment Analysis Supported Requirements Engineering. In *Proceedings of HICSS 2012 (45th Hawaii International Conference on System Sciences)*. 4168–4177.
- [8] Emitza Guzman and Bernd Brügge. 2013. Towards emotional awareness in software development teams. In *Proceedings of ESEC/FSE 2013 (9th Joint Meeting on Foundations of Software Engineering)*. ACM, 671–674.
- [9] Emitza Guzman and Walid Maalej. 2014. How do users like this feature? a fine grained sentiment analysis of app reviews. In *Proceedings of RE 2014 (22nd International Requirements Engineering Conference)*. IEEE, 153–162.
- [10] Md Rakibul Islam and Minhaz F Zibran. 2017. Leveraging automated sentiment analysis in software engineering. In *Proceedings of MSR 2017 (14th International Conference on Mining Software Repositories)*. IEEE Press, 203–214.
- [11] Robbert Jongeling, Proshanta Sarkar, Subhajit Datta, and Alexander Serebrenik. 2017. On negative results when using sentiment analysis tools for software engineering research. *Empirical Software Engineering* (2017), 1–42.
- [12] Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*. 55–60.
- [13] Nicole Novielli, Fabio Calefato, and Filippo Lanubile. 2015. The Challenges of Sentiment Detection in the Social Programmer Ecosystem. In *Proceedings of SSE 2015 (7th International Workshop on Social Software Engineering) (SSE 2015)*. 33–40.
- [14] Bo Pang and Lillian Lee. 2008. Opinion Mining and Sentiment Analysis. *Foundations and Trends in Information Retrieval* 2 (2008), 1–135.
- [15] Sebastiano Panichella, Andrea Di Sorbo, Emitza Guzman, Corrado A Visaggio, Gerardo Canfora, and Harald C Gall. 2015. How Can I Improve My App? Classifying User Reviews for Software Maintenance and Evolution. In *Proceedings of ICSME 2015 (31st International Conference on Software Maintenance and Evolution) (ICSME 2015)*. 281–290.
- [16] Daniel Pletea, Bogdan Vasilescu, and Alexander Serebrenik. 2014. Security and emotion: sentiment analysis of security discussions on GitHub. In *Proceedings of MSR 2014 (11th Working Conference on Mining Software Repositories)*. ACM, 348–351.
- [17] Mohammad Masudur Rahman, Chanchal K Roy, and Iman Keivanloo. 2015. Recommending insightful comments for source code using crowdsourced knowledge. In *Proceedings of SCAM 2015 (15th International Working Conference on Source Code Analysis and Manipulation)*. IEEE, 81–90.
- [18] Athanasios-Ilias Rousinopoulos, Gregorio Robles, and Jesús M González-Barahona. 2014. Sentiment analysis of Free/Open Source developers: preliminary findings from a case study. In *Revista Eletronica de Sistemas de Informacao*, Vol. 13. 1–6.
- [19] Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP 2013 (2013 Conference on Empirical Methods in Natural Language Processing)*. Citeseer.
- [20] Parastou Tourani, Yujuan Jiang, and Bram Adams. 2014. Monitoring sentiment in open source mailing lists: exploratory study on the apache ecosystem. In *Proceedings of CASCON 2014 (24th Annual International Conference on Computer Science and Software Engineering)*. IBM Corp., 34–44.
- [21] Lorenzo Villarroel, Gabriele Bavota, Barbara Russo, Rocco Oliveto, and Massimiliano Di Penta. 2016. Release planning of mobile apps based on user reviews. In *Proceedings of ICSE 2016 (38th International Conference on Software Engineering)*. 14–24.
- [22] Yingying Zhang and Daqing Hou. 2013. Extracting Problematic API Features from Forum Discussions. In *Proceedings of ICPC 2013 (21st International Conference on Program Comprehension)*. 141–151.