2017 IEEE/ACM 14th International Conference on Mining Software Repositories (MSR)

Leveraging Automated Sentiment Analysis in Software Engineering

Md Rakibul Islam
University of New Orleans, USA
Email: mislam3@uno.edu

Minhaz F. Zibran
University of New Orleans, USA
Email: zibran@cs.uno.edu

***Abstract*—Automated sentiment analysis in software engineering textual artifacts has long been suffering from inaccuracies in those few tools available for the purpose. We conduct an in-depth qualitative study to identify the difficulties responsible for such low accuracy. Majority of the exposed difficulties are then carefully addressed in developing `SentiStrength-SE`, a tool for improved sentiment analysis especially designed for application in the software engineering domain. Using a benchmark dataset consisting of 5,600 manually annotated JIRA issue comments, we carry out both quantitative and qualitative evaluations of our tool. `SentiStrength-SE` achieves 73.85% precision and 85% recall, which are significantly higher than a state-of-the-art sentiment analysis tool we compare with.**

I. Introduction

Emotions are an inseparable part of human nature, which influence people's activities and interactions, and thus emotions affect task quality, productivity, creativity, group rapport and job satisfaction [10]. Software development being highly dependent on human efforts and interactions, is more susceptible to emotions of the practitioners. Hence, a good understanding of the developers' emotions and their influencing factors can be exploited for effective collaborations, task assignments [14], and in devising measures to boost up job satisfaction, which, in turn, can result in increased productivity and projects' success.

Several studies have been performed in the past for understanding the role of human aspects on software development and engineering. Some of those earlier studies address *when* and *why* employees get affected by emotions [10], [21], [22], [45], [56], whereas some other work address *how* [19], [25], [26], [30], [31], [35], [57], [58] the emotions impact the employees' performance at work.

Attempts are made to capture the developers' emotions in the workplace by means of traditional approaches such as, interviews, surveys [57], and biometric measurements [32]. Capturing emotions with the traditional approaches is more challenging for projects relying on geographically distributed team settings and voluntary contributions (e.g., open-source projects) [13], [21]. Moreover, the traditional approaches involving direct observations and interactions with the developers often hinder their natural workflow. Thus, to supplement or complement those traditional approaches, recent attempts detect sentiments from the software engineering textual artifacts such as issue comments [9], [12], [21], [25], [26], [31], [40], [45], email contents [17], [56], and forum posts [22], [38].

For automated extraction of sentiments from textual artifacts in the software engineering domain, three tools (i.e., `SentiStrength` [54], `NLTK` [36], and `Stanford NLP` [52]) are used while the use of `SentiStrength` is found dominant [28], [37]. However, software engineering studies [9], [12], [25], [28], [39], [45], [55], [56] involving sentiment analysis repeatedly report concerns about the accuracy of those sentiment analysis tools in the detection of sentimental polarities (i.e., negativity, positivity, and neutrality) of plain text contents. For example, when applied in the software engineering domain, `SentiStrength` and `NLTK` are respectively reported to have only 29.56% and 52.17% precision in identifying positive sentiments, and even lower precision of 13.18% and 23.45% respectively in the detection of negative sentiments [28], [56].

Those sentiment analysis tools are developed and trained using data from non-technical social networking media (e.g., twitter posts, forum posts, movie reviews) and when operated in a technical domain such as software engineering, their accuracy substantially degrades largely due to domain-specific variations in meanings of frequently used technical terms. Although such a domain dependency is indicated as a general difficulty against automated sentiment analysis in textual content, we need a deeper understanding of why and how such domain dependencies affect the performance of the tools, and how we can mitigate them. Indeed, the software engineering community demands a more accurate automatic sentiment analysis tool [9], [12], [26], [28], [39], [42], [45], [51], [56]. In this regard, this paper makes two major contributions:

- Using a large benchmark dataset, we carry out an in-depth exploratory study for exposing the difficulties in automatic sentiment analysis in textual content in a technical domain such as software engineering. To the best of our knowledge, in the literature, no such study exists that investigates public benchmark dataset to identify challenges to sentiment analysis in software engineering.
- We propose techniques and realize those in `SentiStrength-SE`, a prototype tool that we develope for improved sentiment analysis in software engineering textual content. The tool is also made freely available online [50].

Instead of building a tool from scratch, we develop our `SentiStrength-SE` on top of `SentiStrength` [54],

978-1-5386-1544-7/17 $31.00 © 2017 IEEE
DOI 10.1109/MSR.2017.9

which, till date, is the most widely used tool for automated sentiment analysis in software engineering. From quantitative comparison with the original `SentiStrength` [54] as operated in the software engineering domain, we found that our `SentiStrength-SE` significantly outperforms the state-of-the-art tool `SentiStrength`. We further conduct a qualitative evaluation of our tool. Based on the exploratory study and the qualitative evaluation, we outline plans for further improvements in automated sentiment analysis in the software engineering area.

II. Exploratory Study of the Difficulties in Sentiment Analysis

To explore the difficulties in automated sentiment detection in text, we conduct our qualitative analysis around the *Java version* of `SentiStrength` [54]. This *Java version* is the latest release of `SentiStrength`, while the older version, strictly for use on Windows platform, is still available. `SentiStrength` is a state-of-the-art sentiment analysis tool most widely adopted in the software engineering community. The reasons for choosing this particular tool are further justified in Section VI.

English dictionaries consider the words 'emotion' and 'sentiment' as synonymous, and accordingly the words are often used in practice. Although there is arguably a subtle difference between the two, in describing this work, we consider them synonymous. We formalize that a human expression can have two perceivable dimensions: sentimental *polarity* and sentimental *intensity*. Sentimental *polarity* indicates the positivity, negativity, or neutrality of expression while sentimental *intensity* captures the strength of the emotional/sentimental expression, which sentiment analysis tools often report in numeric emotional scores.

A. Benchmark Data

In our work, we use a "Gold Standard" dataset [2], [42], which consists of 5,992 issue comments extracted from JIRA issue tracking system. The entire dataset is divided in three groups named as Group-1, Group-2 and Group-3 containing 392, 1,600 and 4,000 issue comments respectively. Each of the 5,992 issue comments are manually interpreted by n distinct human raters [42] and annotated with emotional expressions as found in those comments. For Group-1, $n = 4$ while for Group-2 and Group-3, $n = 3$. This is the only publicly available such dataset in the software engineering domain [42].

A closed set $\mathcal{E}$ of *emotional expressions* are used in the annotation of the issue comments in the dataset, where $\mathcal{E}$ ={*joy, love, surprise, anger, sad, fear*}. The human raters labeled each of the issue comments depending on whether or not they found the sentimental expressions in the comments. Formally,

$$\mathcal{F}_{\mathcal{E}_i}^{r_j}(\mathcal{C}) = \begin{cases} 1, & \text{if emotion } \mathcal{E}_i \text{ is found in } \mathcal{C} \text{ by rater } r_j. \\ 0, & \text{otherwise.} \end{cases}$$

An example of human annotations of an issue comment from the dataset is shown in Table I.

TABLE I
Annotation of an issue comment by four human raters

Issue comment (Comment ID-53257): `Thanks for the patch; Michale. Applied with a few modifications.`

Human Raters (r_j)	**Emotions** ($\mathcal{E}_i$)					
	Joy	**Love**	**Surprise**	**Anger**	**Sadness**	**Fear**
Rater-1 (r_1)	**1**	**1**	0	0	0	0
Rater-2 (r_2)	0	0	0	0	0	0
Rater-3 (r_3)	**1**	0	0	0	0	0
Rater-4 (r_4)	**1**	0	0	0	0	0

Interpretation: rater-1 found 'joy' and 'love' in the comment, while rater-3 and rater-4 found the presence of only 'love' but rater-2 did not identify any of the emotional expressions.

B. Emotional Expressions to Sentimental Polarities

Emotional expressions joy and $love$ convey *positive* sentimental polarity, while $anger$, $sadness$, and $fear$ express *negative* polarity. In some cases, an expression of $surprise$ can be positive in polarity, denoted as $surprise^{+}$, while other cases can convey a *negative surprise*, denoted as $surprise^{-}$. Thus the issue comments in the benchmark dataset, which are annotated with *surprise* expression, need to be further distinguished based on the sentimental polarities they convey. Hence, we get each of such comments reinterpreted by three additional human (computer science graduate students) raters, who independently determine polarities of the surprise expressions in each comments.

We consider a *surprise* expression in a comment polarized negatively (or positively), if two of the three rates identify negative (or positive) polarity in it. We found 79 issue comments in the benchmark dataset, which were annotated with the *surprise* expression. 20 of them express *surprise* with positive polarity and the rest 59 convey negative *surprise*.

Then we split the set $\mathcal{E}$ of emotional expressions into two disjoint sets as $\mathcal{E}_{+} = \{joy, love, surprise^{+}\}$ and $\mathcal{E}_{-} = \{anger, sad, fear, surprise^{-}\}$. Thus, $\mathcal{E}_{+}$ contains only the positive sentimental expressions and $\mathcal{E}_{-}$ contains only the negative sentimental expressions. A similar approach is also used in other studies [28], [29] to categorize emotional expressions according to their polarities.

C. Computation of emotional scores from human rated dataset

For each of the issue comments in the "Gold Standard" dataset, we compute sentimental polarity using the polarity labels assessed by the human raters. For an issue comment $\mathcal{C}$ rated by n human raters, we compute a pair $\langle \rho_c^{r_j}, \eta_c^{r_j} \rangle$ of values for each of the n raters r_j (where $1 \leq j \leq n$) using Equation 1 and Equation 2:

$$\rho_c^{r_j} = \begin{cases} 1, & \text{if } \sum\limits_{\mathcal{E}_i \epsilon \mathcal{E}_{+}} \mathcal{F}_{\mathcal{E}_i}^{r_j}(\mathcal{C}) > 0 \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

$$\eta_c^{r_j} = \begin{cases} 1, & \text{if } \sum\limits_{\mathcal{E}_i \epsilon \mathcal{E}_{-}} \mathcal{F}_{\mathcal{E}_i}^{r_j}(\mathcal{C}) > 0 \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Thus, if a rater r_j finds the presence of any of the positive sentimental expressions in the comment $\mathcal{C}$, then $\rho_c^{r_j} = 1$, otherwise $\rho_c^{r_j} = 0$. Similarly, if any of the negative sentimental

expressions are found in the comment $\mathcal{C}$, then $\eta_c^{r_j} = 1$, otherwise $\eta_c^{r_j} = 0$.

An issue comment $\mathcal{C}$ is considered neutral in sentimental polarity, if we get the pairs $\langle \rho_c^{r_j}, \eta_c^{r_j} \rangle$ for at least $n-1$ (i.e., majority) raters where $\rho_c^{r_j} = 0$ and $\eta_c^{r_j} = 0$. If the comment is not neutral, then we determine the positive and negative sentimental polarities of that issue comment. To do that, using the following equations, we count the number of human raters, $\mathcal{R}_+(\mathcal{C})$ who found positive sentiment in the comment $\mathcal{C}$ and also the number of raters, $\mathcal{R}_-(\mathcal{C})$, who found negative sentiment in the comment $\mathcal{C}$.

$$\mathcal{R}_+(\mathcal{C}) = \sum_{j=1}^{n} \rho_c^{r_j} \qquad \text{and} \qquad \mathcal{R}_-(\mathcal{C}) = \sum_{j=1}^{n} \eta_c^{r_j}$$

An issue comment $\mathcal{C}$ is considered exhibiting positive sentiment, if at least $n-1$ human raters found positive sentiment in the message. Similarly, we consider a comment having negative sentiment if at least $n-1$ raters found negative sentiment in it. Finally, we compute the sentimental polarities of an issue comment $\mathcal{C}$ as a pair $\langle \rho_c^h, \eta_c^h \rangle$ using Equation 3 and Equation 4.

$$\rho_c^h = \begin{cases} 0, & \text{if } \mathcal{C} \text{ is neutral} \\ +1, & \text{if } \mathcal{R}_+(C) \geq n-1 \\ -1, & \text{otherwise.} \end{cases} \quad (3)$$

$$\eta_c^h = \begin{cases} 0, & \text{if } \mathcal{C} \text{ is neutral} \\ +1, & \text{if } \mathcal{R}_-(C) \geq n-1 \\ -1, & \text{otherwise.} \end{cases} \quad (4)$$

Thus, $\rho_c^h = 1$, only if the comment $\mathcal{C}$ has positive sentiment and $\eta_c^h = 1$ only if the comment contains negative sentiment. Note that, a given comment can exhibit both positive and negative sentiments at the same time. A comment is considered sentimentally neutral when the pair $\langle \rho_c^h, \eta_c^h \rangle$ for the comment appear to be $\langle 0, 0 \rangle$. Similar approach is also followed to determine sentiments of comments in another study [28].

1) Illustrative Example of Computing sentimental Polarity: Consider the issue comment in Table I. For this issue comment, we compute the pair $\langle \rho_c^{r_j}, \eta_c^{r_j} \rangle$ for all four raters (i.e., $n = 4$). As for only one (the second rater) out of four raters we get the pair as $\langle 0, 0 \rangle$, the comment is not considered neutral. Hence, we compute the values of $\mathcal{R}_+(C)$ and $\mathcal{R}_-(C)$, which are three and zero respectively. $\mathcal{R}_+(C)$ being three satisfies the condition of $\mathcal{R}_+(C) \geq n-1$. Thus, $\rho_c^h = 1$, which means that the comment in Table I has positive sentiment. For the same comment $\mathcal{R}_-(C) < n-1$ and so $\eta_c^h = -1$, which signifies that the comment has no negative sentiment.

D. Sentiment Detection Using `SentiStrength`

We apply `SentiStrength` to determine the sentiments expressed in the issue comments in Group-1 of the "Gold Standard" dataset. Sentiment analysis using `SentiStrength` on a given piece of text (e.g., an issue comment) $\mathcal{C}$ computes a pair $\langle \rho_c, \eta_c \rangle$ of integers, where $+1 \leq \rho_c \leq +5$ and $-5 \leq \eta_c \leq -1$. Here, ρ_c and η_c respectively represent the positive and negative sentimental scores for the given text $\mathcal{C}$. A given text $\mathcal{C}$ is considered to have positive sentiment if $\rho_c > +1$. Similarly, a text is held containing negative sentiment when $\eta_c < -1$. Besides, a text is considered sentimentally neutral when the sentimental scores for the text appear to be $\langle 1, -1 \rangle$.

Hence, for the pair $\langle \rho_c, \eta_c \rangle$ of sentimental scores for an issue comment $\mathcal{C}$ computed by `SentiStrength`, we compute another pair of integers $\langle \rho_c^t, \eta_c^t \rangle$ as follows:

$$\rho_c^t = \begin{cases} 1, & \text{if } \rho_c > +1. \\ 0, & \text{otherwise.} \end{cases} \qquad \eta_c^t = \begin{cases} 1, & \text{if } \eta_c < -1. \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\rho_c^t = 1$ signifies that the issue comment $\mathcal{C}$ has positive sentiment, and $\eta_c^t = 1$ implies that the issue comment $\mathcal{C}$ has negative sentiment.

We apply `SentiStrength` to compute sentimental scores for each of the issue comments in the Group-1 portion of the "Gold Standard" dataset and then for each issue comment $\mathcal{C}$, we compute the pair $\langle \rho_c^t, \eta_c^t \rangle$, which represents the sentimental polarity scores for $\mathcal{C}$.

E. Analysis and Findings

For each of the 392 issue comments $\mathcal{C}$ in Group-1, we compare the sentimental polarity scores $\langle \rho_c^t, \eta_c^t \rangle$ produced from `SentiStrength` and the scores $\langle \rho_c^h, \eta_c^h \rangle$ computed using our approach described in Section II-C. We find a total of 151 comments, for which the $\langle \rho_c^t, \eta_c^t \rangle$ scores obtained from `SentiStrength` do not match with $\langle \rho_c^h, \eta_c^h \rangle$. This implies that for those 151 issue comments `SentiStrength`'s computation of sentiments are probably incorrect.

Upon developing a solid understanding of the sentiment detection algorithm of `SentiStrength`, we then carefully go through all of those 151 issue comments to identify the reasons/difficulties, which mislead `SentiStrength` in its identification of sentiments in textual content. We identify 12 such difficulties. Before discussing the difficulties, we first briefly describe the highlights of `SentiStrength`'s internal working mechanism to develop necessary context and background for the reader.

1) Insights into `SentiStrength`'s Internal Algorithm: `SentiStrength` is a lexicon-based classifier that also uses additional (non-lexical) linguistic information and rules to detect sentiment in plain text written in English [54]. `SentiStrength` maintains a dictionary of several lists of words and phrases as its key dictionaries to compute sentiments in texts. Among these lists, the *sentimental words list, list of booster words, list of phrases*, and *list of negations words* play a vital role in the computation of sentiments. The entries in all these lists except the list of negation words are pre-assigned with sentimental scores. The negation words in the fourth list are used to invert the sentimental polarity of a term when the term is located after a negation word in text.

For an input sentence, `SentiStrength` extracts individual words from the sentence and searches for each of the individual words in the *sentimental words list* to retrieve the corresponding sentimental scores. Similar search is made in the *list of booster words* to strengthen or weaken the sentimental scores. The *list of phrases* is used to distinguish groups of words as commonly used phrases. When such

TABLE II
ROLE OF THE DICTIONARY OF LISTS IN SENTISTRENGTH'S COMPUTATION OF SENTIMENTAL SCORES IN TEXT

Sample Sentence	Sent. Score		Dictionary Lists in Use	Explanation
	ρ_c	η_c		
It's a *good* feature.	2	-1	Sentimental Words	The sentimental score of the word *'good'* is pre-assigned to 02; so the sentence is assigned positive score 02.
It's a *very good* feature.	3	-1	Booster Words, Sentimental Word	As booster word *'very'* is used before the sentimental word, the sentence is assigned a positive score 03.
It's *not good* feature.	1	-2	Negations, Sentimental Word	Sentimental polarity of the sentimental word is inverted in here due to the use of the negation word *'not'* before sentimental word.
It's a *killer feature*.	2	-1	Phrases	*"killer feature"* is a phrase in the dictionary with positive score 02. Although the word *'kill'* carries negative sentiment, its effect is overridden by the sentimental score of the enclosing phrase.

TABLE III
FREQUENCIES OF DIFFICULTIES MISLEADING SENTIMENT ANALYSIS

Difficulties		Freq.
D_1 :	Domain-specific meanings of words	123
D_2 :	Context-sensitive variations in meanings of words	35
D_3 :	Misinterpretation of the letter 'X'	12
D_4 :	Sentimental words in copy-pasted content (e.g., code)	12
D_5 :	Difficulties in dealing with negations	08
D_6 :	Missing sentimental words in dictionary	02
D_7 :	Spelling errors mislead sentiment analysis	02
D_8 :	Repetitive numeric characters considered sentimental	01
D_9 :	Wrong detection of proper nouns	01
D_{10} :	Sentimental words in interrogative sentences	01
D_{11} :	Difficulty in dealing with irony and sarcasm	01
D_{12} :	Hard to detect subtle expression of sentiments	07

a phrase is identified, the sentimental score of the phrase overrides sentimental scores of the individual words, which constitute the phrase. The examples in Table II articulate how SentiStrength depends on the dictionary of lists for computing sentimental scores in plain texts.

2) Difficulties in Automated Sentiment Analysis: Table III presents the number of times we found SentiStrength being mislead by the 12 difficulties as discovered during manual investigation. It is evident in Table III that *domain-specific meanings of words* is the most prevalent among all the difficulties that are liable for low accuracy of the lexical approach of SentiStrength. We now describe 12 difficulties with illustrative examples.

(D_1) Domain-specific meanings of words: In a technical field textual artifacts include many technical jargons, which have polarities in terms of dictionary meanings, but do not really express any sentiments in their technical context. For example, the words 'Super', 'Support', 'Value' and 'Resolve' are English words with known positive sentiment, whereas 'Dead', 'Block', 'Default', and 'Error' are known to have negative sentiment, but none of these words really bear any sentiment in software development artifacts. As SentiStrength was originally developed and trained for non-technical texts written in plain English, it identifies those words as sentimental words, which is incorrect in the context of a technical field such as software engineering. In the following comment from the "Gold Standard" dataset, SentiStrength considers 'Error' as negative sentimental word and detects 'Support' and 'Refresh' as positive sentimental words. Thus, it assigns both positive and negative sentimental scores to the comment, although the comment is sentimentally neutral.

> "This was probably fixed by WODEN-86 which introduced **support** for the curly brace syntax in the http location template. This JIRA can now be closed. This test case is now passing ... There are now 12 **errors** reported for Woden on this test caseregenerated the results in r480113. I'll have the W3C reports **refreshed.**" (Comment ID: 18059)

(D_2) Context-sensitive variations in meanings of words: Apart from domain-specific meanings of words, in natural language, some words have multiple meanings depending on the context in which they are used. For example, the word 'Like' expresses positive sentiment when it is used in a sentence such as *"I like you"*. On the other hand, that same word expresses no sentiment in the sentence *"I would like to be a sailor, said George Washington"*. Words that are considered inherently sentimental often do not carry sentiments when used to express possibility and uncertainty. Distinguishing the context-sensitive meanings of such words is a big challenge for automated sentiment analysis in text and the lexical approach of SentiStrength also falls short in this regard.

For example, in the following issue comment, the sentimental word 'Nice' is used simply to express possibility regarding change of something, but SentiStrength incorrectly computes positive sentiment in the message.

> "The change you want **would be nice;** but is simply not possible. The form data ... Jakarta FileUpload library." (Comment ID: 51837)

Similarly, in the following comment, the sentimental word 'Misuse' is used in a conditional sentence, which does not express any sentiment, but SentiStrength interprets otherwise.

> "Added a couple of small points ... **if** anyone notices any **misuses** of the document formatting ..." (Comment ID: 2463)

(D_3) Misinterpretation of the letter 'X': In informal computer mediated chat, the letter 'X' is often used to mean an action of 'Kiss', which is a positive sentiment, and thus recorded in SentiStrength's dictionary. However, in technical domain, the letter is often used as a wildcard. For example, the sequence '1.4.x' in the following comment is used to indicate a collection of versions/releases.

"Integrated in Apache Wicket **1.4.x** ..." (Comment ID: 20748)

Since SentiStrength uses dot (.) as a delimiter to split a text into sentences, the 'x' is considered a one-word sentence and is misinterpreted to have expressed positive sentiment.

(D_4) Sentimental words in copy-pasted content (e.g., code): At commit, the developers often copy-paste code snippets, stack traces, URLs, and camel-case words (e.g., variable names) in their issue comment. Such copy-pasted contents often include sentimental words in the form of variable names and the like, which do not convey any sentiment of the committer, but SentiStrength detects those sentimental words and incorrectly associates those sentiments with the issue comment and the committer. Consider the following issue comment, which includes a copy-pasted stack trace.

"... Stack: [main] **FATAL** ... org.apache.xalan.templates .ElemTemplateElement.**resolve**PrefixTables ..." (Comment ID: 9485)

Here, the words 'Fatal' and 'Resolve' (part of the camel case word 'resolvePrefixTables'), are positive and negative sentimental words respectively in SentiStrength's dictionary. Hence, SentiStrength detects both positive and negative sentiments in the issue comment, but the stack trace content certainly does not represent the sentiments of the developer/committer.

(D_5) Difficulties in dealing with negations: For automated sentiment detection, it is crucial to identify the presence of any negation term preceding a sentimental word, because the negation terms invert the polarity of the sentimental words. For example, the sentence *"I am not in good mood"* is equivalent to *"I am in bad mood"*. When the negation of the positive word 'Good' cannot be identified as equivalent to the negative word 'Bad', then detection of sentimental polarity goes wrong. The default configuration of SentiStrength enables it to detect negation of a sentimental word *only if* the negation term is placed *immediately before* the sentimental word. In all other cases, SentiStrength fails to detect negations correctly and often detects sentiments exactly opposite of what is expressed in the text. During our investigation, we find substantial instances where SentiStrength is misled by complex structural variations of negations present in the issue comments.

For example, SentiStrength incorrectly identifies positive sentiment in the following issue comment due to failing to identify the negation of the positive sentimental word 'Good'.

"3.0.0 has been released; closing ... I didn't change the jute - **don't** think this is a **good** idea; esp as also effects the ... Andrew could you take a look at this one?" (Comment ID: 1725)

In addition, we find that SentiStrength is unable to recognized shortened forms of negations such as, "haven't", "havent", "hasn't", "hasnt", "shouldn't", "shouldnt", and "not" since these terms are not included in the dictionary.

(D_6) Missing sentimental words in dictionary: Since the lexical approach of SentiStrength is largely dependent on its dictionary of lists of words (as discussed in Section II-E1), the tool often fails to detect sentiments in some texts when the sentimental words used in the texts are absent in the dictionary. For example, the words 'Apology' and 'Oops' in the following two comments express negative sentiments, but SentiStrength cannot detect them since those words are not included in its dictionary.

"...This is indeed not an issue. My **apologies** ..." (Comment ID: 20729)

"**Oops**; issue comment had wrong ticket number in it ..." (Comment ID: 36376)

(D_7) Spelling errors mislead sentiment analysis: Misspelled words are common in informal text, and the writer often deliberately misspells words to express intense sentiments. For example, the misspelled word 'Happpy' expresses more happiness than the correctly spelled word 'Happy'. Although SentiStrength can detect some of such intensified sentiments from such misspelled sentimental words, its ability is limited to only those intentional spelling errors where repetition of certain letters occur in a sentimental word. Most other types (unintentional) of misspelling of sentimental words cause SentiStrength fail to find those words in its dictionary and consequently lead to incorrect computation of sentiments. For example, the word 'Unfortunately' was misspelled as 'Unforunatly' in an issue comment (comment ID: 11978) and "I'll" was written as 'ill' in another (comment ID: 927). SentiStrength's detection sentiments in both of these comments are found incorrect.

(D_8) Repetitive characters considered sentimental: As described before, SentiStrength detects higher intensity of sentiments by considering deliberately misspelled sentimental word with repetitive letters. The tool also uses the same strategy for the same purpose by taking into account repetitive characters intentionally typed in words that are not necessarily sentimental by themselves. If anybody writes *"I am goooing to watch movie"* instead of *"I am going to watch movie"*, then the former sentence is considered positively sentimental due to emphasis on the word 'Going' by repetition of the letter 'O' for three times.

However, this strategy also misguides SentiStrength in dealing with some numeric values. For example, in the following comment, SentiStrength incorrectly identifies the number '20001113' as a positive sentimental word encountering repetition of the digits '0' and '1'.

"See bug 5694 for the ... **20001113** /introduction.html ... Zip file with test case (java source and XML docs) 1. Do you use deferred DOM? 2. Can you try to run it against Xerces2 beta4 (or the latest code in CVS?) 3. Can you provide a sample file? Thank you." (Comment ID: 6447)

(D_9) Wrong detection of proper nouns: A proper noun can rightly be considered neutral in sentiment. SentiStrength detects a word starting with a capital letter as a proper noun, when the word is located in the middle or end of a sentence. Unfortunately, grammar rules are often ignored in informal text and thus, sentimental words placed in the middle or end of a sentence often end up starting with a capital letter, which cause SentiStrength mistakenly disregard the sentiments in those sentimental words. The following issue comment is an example of such a case, where the sentimental word 'Sorry' starting with a capital letter is placed in the middle of the sentence and SentiStrength erroneously considers 'Sorry' as a neutral proper noun.

"Cool. Thanks for considering my bug report! ... About the title of the bug; in the description; I put: **Sorry** for the vague ticket title. I don't want to make presumptions about the issue ... work for passwords." (Comment ID: 76385)

However, the older *Windows version* of SentiStrength does not have this shortcoming.

(D_{10}) Sentimental words in interrogative sentences: Typically, negative sentimental words in interrogative sentences (i.e., in questions) either do not express any sentiment or at least weaken the intensity of sentiment [54]. However, we have found instances where SentiStrength fails to correctly interpret the sentimental polarities of such interrogative sentences. For example, SentiStrength incorrectly identifies negative sentiment in the comment below, although the comment merely includes a question expressing no negative sentiment as indicated by the human raters.

"... Did I submit something wrong or duplicate? ..." (Comment ID: 24246)

(D_{11}) Difficulty in dealing with irony and sarcasm: Automatic interpretation of irony in text written in natural language is very challenging, and SentiStrength also often fails to detect sentiments from texts, which express irony and sarcasm [54]. For example, due to the presence of the positive sentimental words "Dear God!" in the comment below, SentiStrength detects positive sentiment in the sentence, although the comment poster used it in a sarcastic manner and expressed negative sentiment only.

"The other precedences are OK; as far as I can tell ... **'zzz'; Dear God!** You mean the intent here is ... gotta confess I just saw the pattern and jumped to conclusions; hadn't examined the code at all. But you've just made the job tougher ...?" (Comment ID: 61559)

(D_{12}) Hard to detect subtle expression of sentiments: Text written in natural language can express sentiments without using any inherently sentimental words. The lexical approach of SentiStrength fails to identify sentiments in such a text due to its high dependency on the dictionary of lists of words, and not being able to properly capture sentence structure and semantic meanings. Consider the following issue comment, which was labeled with negative sentiment by three human raters although there is no sentimental words in it. Without surprise, SentiStrength interprets it as a sentimentally neutral text.

"Brian; I understand what you say and specification about 'serialization' in XSLT not 'indenting'. As I saied before; indenting is just the thing that we easily see the structure and data of XML document. Xalan output is not easy to see that. The last; I think the example of non-whitespace characters is no relationship to indenting. non-whitespace characters must not be stripped; but whitespace characters could be stripped. Regards; Tetsuya Yoshida." (Comment ID: 10134)

III. Leveraging Automated Sentiment Analysis

We address the challenges identified from our exploratory study as described in Section II-E2 and develop a tool particularly crafted for application in the software engineering domain. We call our tool SentiStrength-SE, which is built on top of the original SentiStrength. We now describe how we mitigate the identified difficulties in developing SentiStrength-SE for improved sentiment analysis in textual artifacts in software engineering.

Domain Dictionary Creation: Accuracy of sentiment analysis can be improved by adopting a domain-specific dictionary [18], [24], [46]. We, therefore, create a domain dictionary for software engineering texts, which helps in minimizing the domain difficulty D_1, the most frequent difficulty as identified in Table III. To create the dictionary, we collect a large dataset used in the work of Islam and Zibran [26]. This dataset consists of 490k commit messages drawn from 50 open-source projects from GitHub. Using Stanford NLP tool [53], we extract lemmatized forms of all the words in the commit messages. Among these words, we distinguish those, which are also included in the sentimental words' list of SentiStrength's dictionary. This, we distinguish a total of 716 words, which represent an initial software engineering vocabulary and also are considered sentimental words in general (by SentiStrength).

However, some of these 716 words are simply software engineering domain-specific technical terms, which otherwise would express emotions when interpreted in a non-technical area such as social networking, but not in software engineering. Moreover, some other words such as *'Decrease'*, *'Eliminate'* and *'Insufficient'* are very unlikely to contain sentiments in the software engineering domain. We employ thee human raters (enumerated as A, B, C) to independently identify these domain words. Each of these three human raters has at least three years of software development experience. A human rater annotate a word as neutral if the word is highly unlikely to express any sentiment when interpreted in the software engineering domain. In Table IV, we present sentiment-wise percentage of cases where raters disagree. We also measure the degree of inter-raters agreement in terms

TABLE IV
INTER-RATER DISAGREEMENTS IN INTERPRETATION OF SENTIMENTS

	Disagreements between Human Raters		
Sentimental Polarity	**A, B**	**B, C**	**C, A**
Positive	11.81%	19.68%	17.32%
Negative	08.62%	10.19%	09.41%
Neutral	18.13%	11.81%	15.69%

of *Fleiss-*κ [16] value. The obtained *Fleiss-*κ value 0.739 signifies substantial agreement among the independent raters.

We consider a word as a neutral domain word when two of the three raters identify the word as neutral. Thus, 216 words are identified as neutral domain words, which we exclude from the list of 716 words. Such neutralization of words for a particular domain also suggested in other studies [26], [25], [45], [56].

For each of the remaining 500 words, we adjust the words spelling to discard possible variations at different tense and parts of speech. For example, the word 'Amaze' is converted to 'Amaz*' to capture the variations, 'Amaze', 'Amazingly', 'Amazed'. Finally, we settle with 167 positively and 293 negatively polarized words in the dictionary of `SentiStrength-SE`. The newly developed dictionary also solves the problem D_3 as the letter 'X' is kept out of the dictionary.

Adding contextual sense to minimize ambiguity: Indeed, neutralization of the 216 words is not always appropriate. For example, in the software engineering domain, the word 'Fault' typically indicates a program error and expresses neutral sentiment. However, the same word can also convey negative sentiment as found in the following comment.

> `"As WING ... **My fault:** I cannot reproduce after holidays ... I might add that one; too" the word 'Fault' expresses negative sentiment of the comment poster."` (Comment ID: 4694)

Again, the word 'Like' expresses positive sentiment if it is used as *"I like"*, *"We like"*, *"He likes"*, and *"They like"*. In the most other cases the word 'Like' is used as *preposition* or *subordinating conjunction* and the word can safely be considered sentimentally neutral. For example, the following comment used the word 'Like' without expressing any sentiment.

> `"Looks **like** a user issue to me ..."` (Comment ID: 40844)

We can observe from the above examples that some of the 216 neutralized words can actually express sentiments when those are preceded by *pronouns* referring to a person or a group of persons, e.g., 'I', 'We', 'My', 'He', 'She', 'You' and possessive pronouns such as 'My' and 'Your'. This contextual information is taken into account in `SentiStrength-SE` to appropriately deal with the contextual use of those words in software engineering field to minimize the difficulties D_1 and D_2. The complete list of such words is given in the `SentiStrength-SE` dictionary file named 'ModifiedTermsLookupTable', which are also vetted by the three raters. Note that to determine the Part-Of-Speech (POS) of words in sentences, we apply the Stanford POS tagger [53].

Extending dictionary with new sentimental words and negations: During our exploratory study, we find several informal sentimental words such as, 'Woops', 'Uhh', 'Oops' and 'zzz', which are not included in the original dictionary. The formal word 'Apology' is also missing from the dictionary. We have added to the dictionary of our `SentiStrength-SE` all these missing words as sentimental terms with appropriate sentimental polarities, which mitigate the difficulty D_6.

In addition, we also add to the dictionary the missing shortened from of negation words as mentioned in the discussion of difficulty D_5 in Section II-E2.

Bringing neutralizers in effect: Our observations from the exploratory study (as presented in Section II) reveal that sentiment of a word can be neutralized if that word is preceded by any of the neutralizer words such as, 'Would', 'Could', 'Should', and 'Might'. For example in the sentence "*It would be good if the test could be completed soon*" the positive sentimental word 'Good' does not express any sentiment as neutralized by the preceding word 'Would'. We add a method in `SentiStrength-SE` to enable it correctly detect uses of such neutralizer words in sentences to be more accurate in sentiments detection. This helps in minimizing the difficulty D_2 described before.

Integration of a preprocessing phase: To minimize the difficulties D_4, D_7, D_8, and D_9 (as described in Section II-E2), we include a preprocessing phase to `SentiStrength-SE` as its integral part. Before computation of sentiments in a given input text, `SentiStrength-SE` applies this preprocessing phase to filter out numeric characters and certain copy-pasted contents such as code snippets, URLs and stack traces. To locate code snippets, URL's and stack traces in text, we use regular expressions similar to the approach proposed by Bettenburg et al. [7]. In addition, a spellchecker [3] is also included to deal with the difficulty D_7 in identifying and rectifying misspelled English words. Spell checking also complements our regular expression based method in approximate identification of identifier names in code snippets.

To mitigate the difficulty D_9 in particular, the preprocessing phase also converts all the letters of a comment to small letters. However, converting all the letters to small letters can also cause failure of the detection of the proper nouns such as the names of developers and systems, which is also important as discussed in the description of difficulty D_9 in Section II-E2. From our exploratory study, we have observed that the developers typically mention their colleagues' names in comments immediately after some sort of salutation words such as 'Dear', 'Hi', 'Hello', 'Hellow' or after the character '@'. Hence, in addition to converting all letters to lower case, the preprocessing phase also discards those words, which are placed immediately after any of those salutation words or the character '@'. In addition, `SentiStrength-SE` maintains the flexibility to allow the user to instruct the tool to consider any particular words as neutral in sentiment, in case an inherently sentimental word must be recognized as

proper noun, for example, to deal with the situation where a sentimental word is used as a system's name.

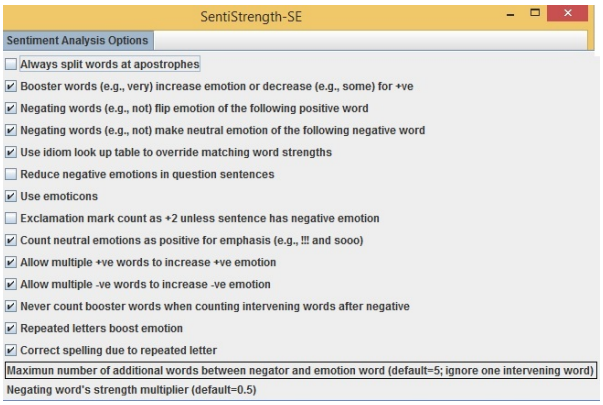

Fig. 1. Default configuration of parameters in our SentiStrength-SE

Parameter configuration for handling negations: We carefully set a number of configuration parameters as defaults to SentiStrength-SE as shown in Figure 1. This default configuration of SentiStrength-SE is different from that of the original SentiStrength. Particularly, to mitigate the difficulty D_5 in dealing with negations, the negation's configuration parameter marked with a black rectangle in Figure 1 is set to five in SentiStrength-SE, which enables the tool detecting negations over a larger range of proximity allowing zero to five intervening words between a negation and a sentimental word, as was also suggested in a previous study [24].

IV. Empirical Evaluation of SentiStrength-SE

While making the design and tuning decisions in the development of SentiStrength-SE, we remained careful about the possibility that the application of a particular heuristic for improvement in one area might have side-effects causing performance degradation in another criteria. We empirically evaluate the accuracy of SentiStrength-SE in two phases. In phase-1, we compare our tool with the original SentiStrength. In phase-2, we carry out a qualitative evaluation of the sentiment analysis of SentiStrength-SE.

In the comparative evaluations of phase-1, the accuracy of sentiment analysis is measured in terms of *precision* (p), *recall* (r), and *F-score* ($\beth$) computed for each of the three sentimental polarities (i.e., positivity, negativity and neutrality). Given a set S of textual contents, *precision* (p), *recall* (r), and *F-score* ($\beth$) for a particular sentimental polarity e is calculated as follows:

$$p = \frac{|\ S_e \cap S'_e\ |}{|\ S'_e\ |}, \quad r = \frac{|\ S_e \cap S'_e\ |}{|\ S_e\ |}, \quad \beth = \frac{2 \times p \times r}{p + r}$$

where $\mathcal{S}_e$ represents the set of texts having sentimental polarity e, and $\mathcal{S}'_e$ denotes the set of texts that are detected (by tool) to have the sentimental polarity e.

In both the two phases of empirical evaluation of our SentiStrength-SE, we use the 5,600 issue comments in Group-2 and Group-3 of the "Gold Standard" dataset introduced in Section II-A. The *ground truth* about the sentimental polarities of those issue comments are determined based on the manual evaluations by human raters as described in Section II-C.

TABLE V
Comparison between original SentiStrength and our SentiStrength-SE

Dataset	Sentiment	Metric	SentiStrength	SentiStrength-SE
Group-2	Positive	Precision	74.48%	89.74%
		Recall	98.81%	98.02%
		F-score	84.93%	93.70%
	Negative	Precision	28.22%	55.40%
		Recall	97.66%	96.09%
		F-score	43.78%	70.28%
	Neutral	Precision	96.83%	96.86%
		Recall	52.42%	84.00%
		F-score	68.01%	89.97%
Group-3	Positive	Precision	31.69%	41.79%
		Recall	87.79%	77.90%
		F-score	46.58%	54.40%
	Negative	Precision	47.61%	71.34%
		Recall	78.40%	72.43%
		F-score	59.25%	71.88%
	Neutral	Precision	91.28%	87.96%
		Recall	56.16%	81.51%
		F-score	69.54%	84.62%
Overall average accuracy		Precision	61.69%	73.85%
		Recall	78.54%	85.00%
		F-score	62.02%	77.48%

A. Phase-1: Comparison with Original SentiStrength

We separately operate the original SentiStrength and our SentiStrength-SE on the 'Group-2' and 'Group-3' datasets that contain 1,600 and 4,000 issue comments respectively. Then for each of the three sentimental polarities (i.e., positivity, negativity, and neutrality), we compare the tools' outcome with the ground truth and separately compute precision, recall, and F-score for both the tools in each dataset.

Table V presents the precision, recall, and F-score of both the original SentiStrength and our SentiStrength-SE in the detection of positive, negative and neutral sentiments, and also the average over all these three sentimental polarities. As evident in Table V, our SentiStrength-SE significantly outperforms the original SentiStrength. Notice that the precisions of SentiStrength-SE for both positive and negative sentiments are higher than the original SentiStrength by large margins. Moreover, at the cost of paltry loss of recall for non-neutral sentiments, SentiStrength-SE achieves substantial increase in recall for sentimental neutrality. Thus, F-score values are always higher in SentiStrength-SE compared to original SentiStrength. Overall, on average, for all sentimental polarities, SentiStrength-SE clearly outperforms original SentiStrength.

B. Phase-2: Qualitative Evaluation of SentiStrength-SE

Although from the comparative evaluations we found our SentiStrength-SE superior to the original SentiStrength, SentiStrength-SE is not a foolproof sentiment analysis tool. Indeed, 100% accuracy cannot be a pragmatic expectation. Nevertheless, we carry out another qualitative evaluation of SentiStrength-SE with two

objectives: first, to confirm that the achieved accuracy found in the comparative evaluations did not occur by chance, and second, to identify failure scenarios and scopes for further improvements.

We first randomly pick 150 issue comments (50 positive, 50 negative, and 50 sentimentally neutral) from the Group-2 and Group-3 of the "Gold Standard" dataset for which SentiStrength-SE correctly detected the sentimental polarities. From our manual verification over these 150 issue comments, we are convinced that the design decisions, heuristics, and parameter configuration adopted in SentiStrength-SE have positive impacts on the accurate detection of sentimental polarities.

Next, we randomly choose another 150 issue comments (50 positive, 50 negative, and 50 sentimentally neutral) for which SentiStrength-SE failed to correctly detect the sentimental polarities. Upon manual investigation of those 150 issue comments, we find a number of reasons for the inaccuracies, a few of which are within the scope of the design decisions applied to SentiStrength-SE, and the rest falls beyond, which we discuss in Section V. One of the reasons for failure is missing sentimental terms in our newly created doamain dictionary. For example, SentiStrength-SE incorrectly identified the following comment as neutral in sentiment by misinterpreting the sentimental word 'Stuck' as a neutral sentimental word, since the word was not included in the dictionary, which we add to the dictionary of SentiStrength-SE's release.

> "For the first part, I got **stuck** on two points" (Comment ID: 1610758_3)

Some other cases we have found inconsistencies in human rating of sentiments in issue comments, which are liable for inaccuracy in SentiStrength-SE. For example the following comment is rated as neutral in sentiment by human raters, although that contains the positive sentimental term 'Thanks' along with the exclamatory sign '!'.

> "And many **thanks** to you Oliver for applying this so quickly**!**" (Comment ID: 577184_1)

A detail investigation have revealed that 200 issue comments are wrongly interpreted in Group-3 by human raters that cause low accuracy in SentiStrength-SE for detecting positive sentiment.

Although the additional preprocessing phase of SentiStrength-SE filters out unwanted content such as source code, URL, numeric values from the input texts, we found several instances where such contents escaped the filtering technique and misguided the tool.

In a few cases, we found that our heuristics to identify proper nouns fell short for not taking into account probable cases. For example, SentiStrength-SE incorrectly computed negative sentiment in the following issue comment. As seen in the following comment a developer thanked his colleague name 'Harsh'.

> "Thanks **Harsh**, the patch looks good ... Since this is a new API, we are not sure if want to change it. Let's leave it as-is for the moment." (Comment ID: 899420)

For failing to identify 'Harsh' as a proper noun, SentiStrength-SE considered the word sentimentally negative and erroneously detects negative sentiment in the message. Our immediate future plan includes further extension to our heuristics for locating proper nouns in text.

V. Limitations and Future Possibilities

In the development of SentiStrength-SE, we have addressed the difficulties identified from the exploratory study described in Section II. Still there are scopes for further improvements, as we also found from the qualitative evaluation of the tool. For example, we have observed in Section IV-B that missing of sentimental words can mislead the SentiStrength-SE. We plan to further extend our domain dictionary to a comprehensive lexicons list.

Our approach for domain dictionary creation is different from existing approaches [8], [15], [44]. We have deliberately chosen this approach for two reasons. First, we wanted to introduce a new approach, and second, it was not possible to adopt existing approached due to limitation of resources such as sentiment-annotated texts in software engineering [42]. Through empirical evaluations, we have shown that our created domain dictionary is effective for sentiment analysis in software engineering.

Although our approach for filtering out code snippets may not correctly locate all code portions, but the filtering indeed minimizes them. Indeed, isolating inline source code from plain text content is a challenging task, especially when the text can have code written in diverse undeclared programming language. Such a code separation problem can be a separate research topic and limited scope attempts are made in the past [5]. We also plan to invest efforts along this direction to further improve SentiStrength-SE.

At this stage, we did not address the difficulties D_{10}, D_{11}, and D_{12}, which are included in our future plan. The detection of irony, sarcasm, and subtle emotions hidden in text is indeed a challenging research topic in NLP and not only related to software engineering texts. Even human interpretations of sentiments in text often disagree as such we also found in the "Gold Standard" dataset. Combining the dictionary-based lexical method with machine learning [48] and other specialized techniques [6] can lead to potential means to address these difficulties. We also plan to add to SentiStrength-SE the capability to identify interrogative sentences correctly mitigate the difficulty D_{10}.

VI. Related work

To the best of our knowledge, the qualitative study (Section II), is the first study that analyzes *public benchmark dataset* to expose the challenges to sentiment analysis in software engineering. And, we have developed the first sentiment analysis tool, SentiStrength-SE, crafted especially for software engineering domain, which we expect to produce superior performance in other technical domains as well.

Aside from our tool, there are only four prominent tools/toolkits namely, SentiStrength [54], Stanford NLP [53], NLTK [36], and Alchemy [1], which facilitate automatic sentiment analysis in plain texts. The first three of these tools have been used for sentiment analysis in software engineering domain, while SentiStrength is used most frequently as presented in Table VI. Those tools, which are previously used in software engineering area, but *not for sentiment analysis*, are excluded from the table.

TABLE VI
USES OF TOOLS FOR *sentiment analysis* IN SOFTWARE ENGINEERING

Tools	Uses in Software Engineering Research
SentiStrength [54]	[9], [12], [17], [20], [21], [22], [23], [27], [28], [39], [40], [41], [51], [55], [56]
NLTK [36]	[45], [49]
Stanford NLP [53]	[47]

Alchemy [1] is a commercial toolkit that offers limited sentiment analysis as a service through its published APIs. NLTK (Natural Language Toolkit) [36] and Stanford NLP [53] are general purpose natural language processing (NLP) library/toolkit, which expect the user to have some NLP background and to write scripting code for carrying out sentiment analysis in plain text. In contrast, SentiStrength is a dedicated tool that applied a lexical approach for automated sentiment analysis and is ready to operate without needing to write any scripting code (for natural language processing). Perhaps, these are among the reasons why, in software engineering community, SentiStrength has gained popularity over the alternatives. The same reasons also made us choose this particular tool as the basis of our work. Our SentiStrength-SE reuses the lexical approach of the original SentiStrength and is also ready to be used off the shelf.

All of the aforementioned four tools (i.e., SentiStrength [54], Stanford NLP [53], NLTK [36], and Alchemy [1]) are developed and trained to operate on non-technical texts drawn from social interactions, web pages, and they do not perform well enough when operated in a technical domain such as software engineering. Domain-specific (e.g., software engineering) technical uses of inherently emotional words seriously mislead the sentiment analyses of those tools [28], [39], [45], [56] and limit their applicability in software engineering area. Medhat et al. [33] conducted a survey on 54 studies, which included sentiments analysis in text, but none of the studies included text from software engineering domain.

In an attempt to minimize the domain difficulty, a variety of machine learning techniques such as, Naive Bayes classifier, Support Vector Machine (SVM) [43], and Logistic Regression [11] have been explored. However, the performances of all these three classifiers are reported lower than the lexical approach when operated on domain-specific texts [34].

Recently, Blaz and Becker [8] proposed three almost equally performing methods, a *Dictionary Method (DM)*, a *Template Method (TM)* and a *Hybrid Method (HM)* for sentiment analysis in "Brazilian Portuguese" texts in IT (Information Technology) job submission tickets. The DM is a pure lexical approach similar to that of our SentiStrength-SE. Although their techniques might be suitable for formally structured texts, those may not perform well in dealing with informal texts that are frequently used in software engineering artifacts such as commit comments. In contrast, from the empirical evaluation over commit comments, our SentiStrength-SE is found to have high accuracy in detecting sentiments in those informal software engineering texts. The proposed methods of Blaz and Becker [8] are developed and evaluated against text written in "Brazilian Portuguese" language instead of English. Thus, their approach and reported results are not directly comparable to ours.

Similar to the qualitative study included in our work, Novielli et al. [39] also conducted a relatively brief study of the challenges against sentiment analysis in "social programmer ecosystem". They also used SentiStrength for the detection of emotional polarities and reported only domain difficulty as a key challenge. In their work, they manually studied only 100 questions and their follow-up comments as well as 100 answers and their follow-up discussions obtained from Stack Exchange Data Dump [4]. In contrast, based on a deeper analysis over a publicly available benchmark data, our study exposes 12 difficulties including the domain dependency. In addition, we address those difficulties and develop an improved sentiment analysis tool for operation in software engineering domain.

VII. CONCLUSION

In this paper, we have first presented an in-depth qualitative study to identify the difficulties in automated sentiment analysis in software engineering texts. Among the difficulties, the challenges due to domain dependency are found the most dominant. Next, we have addressed the majority of the identified difficulties and developed a tool, SentiStrength-SE, for improved sentiment analysis in textual contents in a technical domain, especially in software engineering. Our tool reuses the lexical approach of SentiStrength [54], which, in software engineering, is the most widely adopted sentiment analysis technique.

Over a large dataset (i.e., Group-2 and Group-3) consisting of 5,600 issue comments, a quantitative empirical comparison with the original SentiStrength [54] suggests that our SentiStrength-SE is substantially superior to the state of the art tool in detecting emotions in software engineering textual contents. In addition, a qualitative evaluation also confirms the effectiveness of the design decisions and heuristics we have included in our SentiStrength-SE.

Both from the exploratory study and qualitative evaluation of our sentiment analysis tool, we have also identified scopes for further improvements of the tool, which remain within our future research plans. Using SentiStrength-SE and its future releases, we also plan to conduct large scale studies of emotional variations and their impacts in software engineering. The current release of our SentiStrength-SE is made freely available [50] for public use.

REFERENCES

[1] *AlchemyLanguage: Natural language processing for advanced text analysis*. http://www.alchemyapi.com/products/alchemylanguage/sentiment-analysis, last access: Feb 2017.

[2] *Gold Standard Dataset Labeled with Manually Annotated Emotions*. http://ansymore.uantwerpen.be/system/files/uploads/artefacts/alessandro/MSR16/archive3.zip, last access: Jan 2017.

[3] *Jazzy- The Java Open Source Spell Checker*. http://jazzy.sourceforge.net, last access: Jan 2017.

[4] *Stack Exchange Data Dump*. https://archive.org/details/stackexchange, last access: Jan 2017.

[5] A. Bacchelli, A. Cleve, M. Lanza, and A. Mocci. Extracting structured data from natural language documents with island parsing. In *Proceeding of the International Conference on Automated Software Engineering*, pages 476–479, 2011.

[6] A. Balahur, J. Hermida, and A. Montoyo. Detecting implicit expressions of sentiment in text based on commonsense knowledge. In *Proceedings of the Workshop on Computational Approaches to Subjectivity and Sentiment Analysis*, pages 53–60, 2011.

[7] N. Bettenburg, B. Adams, and A. Hassan. A lightweight approach to uncover technical information in unstructured data. In *In Proceedings of the International Conference of Program Comprehension*, pages 185–188, 2011.

[8] C. Blaz and K. Becker. Sentiment analysis in tickets for it support. In *Proceedings of the International Conference on Mining Software Repositories*, pages 235–246, 2016.

[9] F. Calefato and F. Lanubile. Affective trust as a predictor of successful collaboration in distributed software projects. In *Proceedings of the International Workshop on Emotion Awareness in Software Engineering*, pages 3–5, 2016.

[10] M. Choudhury and S. Counts. Understanding affect in the workplace via social media. In *Proceedings of the Computer supported cooperative work*, pages 303–316, 2013.

[11] M. Choudhury, M. Gamon, and S. Counts. Happy, nervous or surprised? Classification of human affective states in social media. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, pages 435–438, 2012.

[12] S. Chowdhury and A. Hindle. Characterizing energy-aware software projects: Are they different? In *Proceedings of the International Conference on Mining Software Repositories*, pages 508–511, 2016.

[13] G. Destefanis, M. Ortu, S. Counsell, M. Marchesi, and R. Tonelli. Software development: do good manners matter? *PeerJ PrePrints*, pages 1–17, 2015.

[14] P. Dewan. Towards emotion-based collaborative software engineering. In *Proceedings of the International Workshop on Cooperative and Human Aspects of Software Engineering*, pages 109–112, 2015.

[15] E. Dragut, C. Yu, P. Sistla, and W. Meng. Construction of a sentimental word dictionary. In *International Conference on Information and Knowledge Management*, pages 1761–1764, 2010.

[16] J. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

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

[18] N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs. In *Procceding of the First International AAAI Conference on Weblogs and Social Media*, 2007.

[19] D. Graziotin, X. Wang, and P. Abrahamsson. Are happy developers more productive? The correlation of affective states of software developers and their self-assessed productivity. In *Proceedings of the International Conference on Product-Focused Software Process Improvement*, pages 50–64, 2013.

[20] E. Guzman. Visualizing emotions in software development projects. In *Proceedings of the Conference on Software Visualization*, pages 1–4, 2013.

[21] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: An empirical study. In *Proceedings of the International Conference on Mining Software Repositories*, pages 352–355, 2014.

[22] E. Guzman and B. Bruegge. Towards emotional awareness in software development teams. In *Proceedings of the International Symposium on the Foundations of Software Engineering*, pages 671–674, 2013.

[23] E. Guzman and W. Maalej. How do users like this feature? A fine grained sentiment analysis of app reviews. In *Proceedings of the International Requirements Engineering Conference*, pages 153 – 162, 2014.

[24] M. Hu and B. Liu. Mining and summarizing customer reviews. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 168–177, 2004.

[25] M. Islam and M. Zibran. Exploration and exploitation of developers' sentimental variations in software engineering. *International Journal of Software Innovation*, 4(4):35–55, 2016.

[26] M. Islam and M. Zibran. Towards understanding and exploiting developers' emotional variations in software engineering. In *Proceedings of the International Conference on Software Engineering Research Management and Applications*, pages 185–192, 2016.

[27] J. Jiarpakdee, A. Ihara, and K. Matsumoto. Understanding question quality through affective aspect in Q&A site. In *Proceedings of the International Workshop on Emotion Awareness in Software Engineering*, pages 12–17, 2016.

[28] R. Jongeling, S. Datta, and A. Serebrenik. Choosing your weapons: On sentiment analysis tools for software engineering research. In *Proceedings of the International Conference on Software Maintenance and Evolution*, pages 531–535, 2015.

[29] R. Jongeling, P. Sarkar, S. Datta, and A. Serebrenik. On negative results when using sentiment analysis tools for software engineering research. *Empirical Software Engineering*, pages 1–42, 2017.

[30] T. Lesiuk. The effect of music listening on work performance. *Psychology of Music*, 33(2):173–191, 2005.

[31] M. Mäntylä, B. Adams, G. Destefanis, D. Graziotin, and M. Ortu. Mining valence, arousal, and dominance – possibilities for detecting burnout and productivity. In *Proceedings of the International Conference on Mining Software Repositories*, pages 247–258, 2016.

[32] D. McDuff, A. Karlson, A. Kapoor, A. Roseway, and M. Czerwinski. Affectaura: an intelligent system for emotional memory. In *Proceedings of the Conference on Human Factors in Computing Systems*, pages 849–858, 2012.

[33] W. Medhat, A. Hassan, and H. Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams Engineering Journal*, 5:1093–1113, 2014.

[34] A. Muhammad, N. Wiratunga, R. Lothian, and R. Glassey. Domain-based lexicon enhancement for sentiment analysis. In *Proceedings of the SGAI International Conference on Artificial Intelligence*, 2013.

[35] A. Murgia, P. Tourani, B. Adams, and M. Ortu. Do developers feel emotions? An exploratory analysis of emotions in software artifacts. In *Proceedings of the International Conference on Mining Software Repositories*, pages 261–271, 2014.

[36] NLTK. *Natural Language Toolkit for Sentiment Analysis*. http://www.nltk.org/api/nltk.sentiment.html, last access: Oct 2016.

[37] N. Novielli. *List of Tools Used in Software Engineering to Detect Emotions*. http://www.slideshare.net/nolli82/the-challenges-of-affect-detection-in-the-social-programmer-ecosystem, last access: Sep 2016.

[38] N. Novielli, F. Calefato, and F. Lanubile. Towards discovering the role of emotions in stack overflow. In *Proceedings of the International Workshop on Social Software Engineering*, pages 33–40, 2014.

[39] N. Novielli, F. Calefato, and F. Lanubile. The challenges of sentiment detection in the social programmer ecosystem. In *Proceedings of the International Workshop on Social Software Engineering*, pages 33–40, 2015.

[40] 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 *Proceedings of the International Conference on Mining Software Repositories*, pages 303–313, 2015.

[41] M. Ortu, G. Destefanis, S. Counsell, S. Swift, R. Tonelli, and M. Marchesi. Arsonists or firefighters? Affectiveness in agile software development. In *Proceedings of the International Conference on Extreme Programming*, pages 144–155, 2016.

[42] M. Ortu, A. Murgia, G. Destefanis, P. Tourani, R. Tonelli, M. Marchesi, and B. Adams. The emotional side of software developers in JIRA. In *Proceedings of the International Conference on Mining Software Repositories*, pages 480–483, 2016.

[43] B. Panga, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 79–86, 2002.

[44] L. Passaro, L. Pollacci, and A. Lenci. Item: A vector space model to bootstrap an italian emotive lexicon. In *Second Italian Conference on Computational Linguistics CLiC-it*, pages 215–220, 2015.

[45] D. Pletea, B. Vasilescu, and A. Serebrenik. Security and emotion: Sentiment analysis of security discussions on github. In *Proceedings of the International Conference on Mining Software Repositories*, pages 348–351, 2014.

[46] G. Qiu, B. Liu, J. Bu, and C. Chen. Expanding domain sentiment lexicon through double propagation. In *Proceedings of the International Jont Conference on Artifical Intelligence*, pages 1199–1204, 2009.

[47] M. Rahman, C. Roy, and I. Keivanloo. Recommending insightful comments for source code using crowdsourced knowledge. In *Proceedings of the International Working Conference on Source Code Analysis and Manipulation*, pages 81–90, 2015.

[48] A. Reyes, P. Rosso, and D. Buscaldi. From humor recognition to irony detection: The figurative language of social media. *Data and Knowledge Engineering*, 74:1–12, 2012.

[49] A. Rousinopoulos, G. Robles, and J. Barahona. Sentiment analysis of free/open source developers: preliminary findings from a case study. *Revista Eletronica de Sistemas de Informacao*, 13(2):1–21, 2014.

[50] SentiStregth-SE. *Sentiment Analysis Tool, freely available for download*. https://gitlab.com/i__030218/SentiStrengthSE_Tool_MSR/tree/master, last access: Jan 2017.

[51] V. Sinha, A. Lazar, and B. Sahrif. Analyzing developer sentiment in commit logs. In *Proceedings of the International Conference on Mining Software Repositories*, pages 520–523, 2016.

[52] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 1631–1641, 2013.

[53] StanfordCoreNLP. *Stanford Core NLP Sentiment Annotator*. http://stanfordnlp.github.io/CoreNLP/sentiment.html, last access: Feb 2017.

[54] M. Thelwall, K. Buckley, and G. Paltoglou. Sentiment strength detection for the social web. *Journal of the American Society for Info. Science and Tech.*, 63(1):163–173, 2012.

[55] P. Tourani and B. Adams. The impact of human discussions on just-in-time quality assurance. In *Proceedings of the International Conference on Software Analysis, Evolution, and Reengineering*, pages 189–200, 2016.

[56] P. Tourani, Y. Jiang, and B. Adams. Monitoring sentiment in open source mailing lists – exploratory study on the apache ecosystem. In *Proceedings of the Conference of the Centre for Advanced Studies on Collaborative Research*, pages 34–44, 2014.

[57] M. Wrobel. Emotions in the software development process. In *Proceedings of the International Conference on Human System Interaction*, pages 518–523, 2013.

[58] M. Wrobel. Towards the participant observation of emotions in software development teams. In *Proceedings of the Federated Conference on Computer Science and Information Systems*, pages 1545–1548, 2016.