MarValous: Machine Learning Based Detection of Emotions in the Valence-Arousal Space in Software Engineering Text

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

Md Kauser Ahmmed
University of New Orleans, USA
mahmmed@uno.edu

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

ABSTRACT

Emotion analysis in text has drawn recent interests in the software engineering (SE) community. Existing domain-independent techniques for automated emotion/sentiment analysis perform poorly when operated on SE text. Thus, a few SE domain-specific tools are recently developed for detecting *sentimental polarities* (e.g., positivity, negativity). But, for capturing *individual emotional states* such as excitation, stress, depression, and relaxation, there is only one recent tool named DEVA, which uses a lexicon-based approach.

We have developed `MarValous`, the first *Machine Learning* based tool for improved detection of the aforementioned emotional states in software engineering text. We evaluate `MarValous` using a dataset containing 5,122 comments collected from JIRA and Stack Overflow. From a quantitative evaluation, `MarValous` is found to have substantially outperformed DEVA achieving more than 83% precision and more than 79% recall.

CCS CONCEPTS

• **Software and its engineering → Programming teams**;

KEYWORDS

Emotion; sentiment; valence; arousal; machine learning

ACM Reference Format:
Md Rakibul Islam, Md Kauser Ahmmed, and Minhaz F. Zibran. 2019. MarValous: Machine Learning Based Detection of Emotions in the Valence-Arousal Space in Software Engineering Text. In *The 34th ACM/SIGAPP Symposium on Applied Computing (SAC '19), April 8–12, 2019, Limassol, Cyprus.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3297280.3297455

1 INTRODUCTION

Emotions greatly influence people's decision making and collaboration with others [21]. Thus, emotions affect people's task quality, productivity, creativity, group rapport and job satisfaction [33]. Software development activities being highly dependent on human efforts and interactions, are more susceptible to emotions of the individuals.

Recent studies [25, 28, 29, 32, 35, 39, 57] examined the impacts of emotions on various performance factors (e.g., productivity, quality and efficiency). Emotions are also captured and used as an important factor in prioritizing applications' features to develop [18], in release planning [40], in predicting bug severity [59], and in determining the qualities of developers' interactions in technical forums [16]. All these recent studies indicate emotion analysis as an emerging area in software engineering (SE).

While approaches such as, interviews, surveys [57], and biometric measurements [37] are used to detect developers' emotions, due to the current distributed nature of Open Source Software (OSS) development, detecting emotions from textual artifacts including issue comments [15, 28, 29, 45, 48], email contents [24], and forum posts [26, 41] is becoming more popular lately. For capturing emotions in SE textual artifacts, initially, domain-independent sentiment analysis tools (e.g., `SentiStrength` [53], `NLTK` [6], and `Stanford NLP` [52]) were tried. It is found that those domain-independent tools do not perform well when applied to SE text [15, 28, 34, 42, 48, 55] mainly due to the variations in meanings of domain-specific technical terms [30, 33] and high proportion of noises (e.g., code snippets, URLs, and API names) [10, 16, 30].

Recently, a few SE domain-specific tools [10, 14, 16, 30] are developed for detecting *sentimental polarities* (e.g., positivity, negativity) only. Those tools are limited in capturing emotional states at the necessary levels such as in capturing *excitation, stress, depression,* and *relaxation* while it is important to capture these emotional states [31, 42]. Islam and Zibran recently addressed this limitation and developed `DEVA` [31], which, till date, is the only tool available for detecting those four emotional states in SE texts. However, `DEVA` is lexicon-based and such a technique is limited by the quality and sizes of the underlying dictionaries in use [19, 21]. Arguably, a dictionary-based approach can fail to capture the organization of a text that contributes information relevant to the emotion of the text writer [49].

In this paper, we present `MarValous` (**Ma**chine Lea**r**ning Based Emotion Detector in **Val**ence-Ar**ous**al Space), a tool that we have developed for automatic detection of individual emotional states expressed in SE text. In particular, this paper makes the following three contributions:

- We develop the first *Machine Learning (ML) based* tool for improved detection of four emotional states *excitation, stress, depression,* and *relaxation* expressed in software engineering texts.
- We evaluate nine supervised ML algorithms incorporated in our tool to measure their applicabilities in detecting the aforementioned four emotional states.
- We create a unified dataset by combining two different benchmark datasets under a uniform format. This unified dataset can be reused in training and testing similar tools as we have done for our ML-based tool.

SAC '19, April 8–12, 2019, Limassol, Cyprus

ACM ISBN 978-1-4503-5933-7/19/04…$15.00
https://doi.org/10.1145/3297280.3297455

Our MarValous tool exploits supervised ML techniques. It includes nine text preprocessing steps and seven feature extraction modules. In empirical evaluations using the aforementioned unified dataset, MarValous demonstrates 83.37% precision, 79.33% recall, and 80.90% F-score. Both the MarValous tool and the dataset are made publicly available [1].

2 EMOTIONAL MODEL

We use a two-dimensional model [27, 35] of emotions to classify texts in software engineering. Figure 1 presents the most widely used emotion classification model in the two-dimensional approach [23] proposed by Russell and Mehrabian [51]. As shown in Figure 1, each dimension is bipolar where the valence dimension ranges from negative valence (i.e., pleasant) to positive valence (i.e., unpleasant) and the arousal dimension ranges from low to high. Total 28 emotional states of a person can be determined by combining different levels of valence and arousal in each of the four quadrants marked as Q1, Q2, Q3, and Q4 in Figure 1.

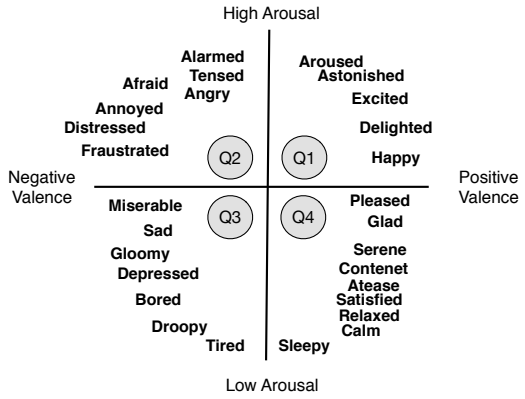

Figure 1: Two-dimensional emotion classification model.

Several studies [22, 27, 31] use simplified versions of the aforementioned two-dimensional model of Russel and Mehrabian [51], where each quadrant is represented by a unique emotional state. For example, the quadrants Q1, Q2, Q3 and Q4 are represented by emotions *excitation, stress, depression,* and *relaxation* respectively, in the work of Islam and Zibran [31]. The four classes of emotions are very distinct, as each state constitutes emotions, which are quite different compared to the emotions of other states. Moreover, the model is simple and easy to perceive. Therefore, the simplified model of Islam and Zibran [31] is also adopted in our work.

3 MARVALOUS

We develop MarValous in Python and use *scikit-learn* [7] for supervised learning algorithms. For improved classification performances, MarValous consists of two major modules: (i) data preprocessing and (ii) feature selection, which are described in the following subsections.

3.1 Preprocessing

In the preprocessing phase, we sanitize the input text to get rid of probable noises, which otherwise could mislead classification.

URL and code snippet removal: Natural language texts (e.g., commit and issue comments) generated during software development may often include noisy texts such as URL references and code snippets, which do not convey any emotion of writers of those texts. However, those URLs and code snippets may contain emotional words that can easily mislead emotion detection approaches/tools [33]. Moreover, keeping those noises in texts will increase the size of features' vector for a ML classifier. We use a simple regular expression technique to identify and remove all URLs and code snippets from our dataset. Such simple regular expression technique is found to be effective [12] and also used for similar purpose in other studies [10, 33] too.

Removal of numeric expressions: Similar to URLs and code snippets, any numbers in texts do not indicate any emotion of the writers and increase size of features' vecrtor. Hence using a regular expression, we identify and remove all numbers from our dataset.

Slang removal: Due to informal nature of communications among developers, they frequently use slangs in their writings. For example, in the comment, "`Thanx a lot! Could you tell me, how can I download ... OM 2.2? :`," the writer uses the slang 'Thanx' instead of the English word 'Thanks'. We replace such slangs in texts with formal English words to reduce features' vector size.

Stop word removal: Removing stop-words (such as articles, prepositions, and conjunctions) to reduce number of features is a common practice in ML based techniques. While predicting emotions in texts, such removal of stop-words is highly expected as those stop-words do not play any significant roles to express emotions.

Table 1: A customized stop-words list

'it,' 'itself,' 'this,' 'that,' 'these,' 'those,' 'is,' 'are,' 'was,' 'were,' 'be,' 'been,' 'being,' 'have,' 'has,' 'had,' 'having,' 'do,' 'does,' 'did,' 'doing,' 'a,' 'an,' 'the,' 'and,' 'if,' 'or,' 'as,' 'until,' 'while,' 'of,' 'at,' 'by,' 'for,' 'between,' 'into,' 'through,' 'during,' 'to,' 'from,' 'in,' 'out,' 'on,' 'off,' 'then,' 'once,' 'here,' 'there,' 'all,' 'any,' 'both,' 'each,' 'few,' 'more,' 'other,' 'some,' 'such,' 'than,' 'too,' 's,' 't,' 'can,' 'will,' 'don,' 'should'

Although popular natural language processing tools (e.g., Stanford CoreNLP [9] and NLTK [6]) provide lists of stop-words, we use a customized stop-word list presented in Table 1. As the popular tools' provided stop-words list includes personal pronouns (e.g., 'he', 'she', and 'my'), temporal terms (e.g., now), booster words (e.g., very) and few others terms that are used for negations (e.g., 'no', 'not') and asking questions (e.g., 'why', 'what'), which play important roles in expressing emotions [30, 31, 53]. Thus, those types of terms are removed from stop-words collection to prepare our customized list.

Name replacement: Developers typically mention their colleagues' names in texts immediately after salutation words such as 'Dear', 'Hi', 'Hello', 'Hellow' or after the character '@' [30], which do not convey any emotion rather increase size of features' vector. Hence, all words that start after the words 'Dear', 'Hi', 'Hello', 'Hellow' and the symbol '@' are replaced by the single word 'UserName'.

Software-specific named entity replacement. Similar to colleagues' names, software-specific named entities do not express any

emotion in texts and increase size of features vector. Hence, we identify the named entities and replace those using the keyword 'NamedEntity'. To identify those named entities, we use the gazetteer [60] prepared for software engineering domain. The gazetteer includes total 400,147 entries divided into five categories, which are presented in Table 2. However, all the entries in the API category are detected and removed by the logic of detection of code snippet.

Table 2: Categories of Software-specific Name Entities

Named-Entity Category	# of Entries
Programming language (e.g., Java, C)	419
Platform (e.g., x86, AMD64)	175
API (e.g., Java ArrayList, toString())	396,968
Tool-library-framework (e.g., JProfiler, Firebug)	2,196
Software standard (e.g., HTTP, FTP)	389

Dealing with negations: Generally, a negation word (e.g., 'no', 'never') is used for reversing or weakening the meaning of the word it qualifies [30]. Since ML based classifiers operate on unigrams and bi-grams representations of sentences, those classifiers often fail to identify negated opinions [10]. To overcome that problem, an earlier ML based sentiment analysis tools [46] adopted a simple approach by prepending 'not' with the succeeding words that are found after a negation word.

However, negations only affect verbs, adjectives, and adverbs but do not alter nouns, determiners, articles, and particles [10]. Thus, we modify verbs, adjectives, and adverbs (instead of all the words) by prepending 'not_', which are found within the scope of a negation word in a sentence. To determine scope of negations, we use *chunking* or *shallow parsing* [10] to divide a text into syntactically correlated parts of words. The work of Ahmed et al. [10] also follows the same procedure.

Word tokenizing and stemming: Among available popular tokenizer tools e.g., `NLTK`, `Stanford CoreNLP`, `SyntaxNet` [3] and `spaCy` [5], we use NLTK as it shows the best performance [44] in software engineering domain to tokenize words in texts. After tokenization, we apply word stemming [8] to convert each word to its root. We apply Snowball Stemmer [8] as it is used in another software engineering study [10].

Expansion of contractions: Contractions are shortened forms of a group of words, which are commonly used in informal written communications. When a contraction is written in English, the omitted letters are replaced by an apostrophe. Some frequently used contractions and their expanded forms include: aren't → are not, and I'm → I am and won't → will not. Writing the same thing in two different ways (i.e., using or not using contraction) increases size of features vector.

Thus, such expansion of contractions reduces the number of unique lexicons (i.e., feature vectors), which, in turns, helps to improve ML based classifiers' performances. We expand total 124 frequently used contractions [10] if they are found in texts.

3.2 Feature Selection

For machine learning, we identify a set of features in text, which we describe below along with the rationale why they can be useful in emotion classification.

n-gram: An n-gram is a contiguous sequence of n items from a given piece of text. The items can be phonemes, syllables, letters, words or base pairs and n can be any natural number. In our work, an item refers to a *word* in a given text and n ranges from one to two i.e., we use *unigram* (where $n = 1$) and *bigram* (where $n = 2$) as features.

For example, if a given text consists of the words w_1 w_2 w_3, then unigram feature will contain each of the words i.e., w_1, w_2 and w_3 and bigram feature will contain pair of words i.e., w_1 w_2 and w_2 w_3. We compute TF-IDF (Term Frequency - Inverse Document Frequency) [4] for each of the unigram and bigram features. The n-gram method is language independent and works well in the case of noisy-texts [36]. Thus, it suits very well due to informal and noisy natures of software engineering texts.

Emoticons: In informal written communications emoticons are frequently used to express different emotions of writers. Hence, emoticons have been commonly used to classify emotions in several studies [10, 16, 27, 31]. We use total 38 emoticons in this work and categorize those in the four quadrants according to their emotions as presented in Table 3 (see the first and third columns). This categorization of emoticons is used in other studies [31, 58] too. We use a binary feature (e.g., hasEmoticon) that keeps record of existence of emoticons in a given text.

Moreover, to reduce features' vector size, we replace 38 emoticons in texts with four keywords according to their emotions. For example, if we find an emoticon, which expresses emotion *excitation*, then that emoticon is replaced with the keyword 'Excited'. All such four keywords are mentioned in the second column of Table 3 with respect to their emotions.

Interjections: The interjections are special parts of speech (POS), which are also frequently used to express emotional states [31]. As presented in the fourth column of Table 3, we use total 37 interjections that are categorized into four emotions according to their meanings in an earlier work [31]. Similar to emoticons, we use another binary feature (e.g., hasInterjection) that identifies presence of interjections in a given text. Again, to reduce features' vector size, we replace the interjections with four keywords presented in the second column of Table 3 (similar to replacement process of emoticons) according to their emotions.

Exclamation marks: Writers often use exclamation mark when they want to express their intense feelings [16, 27]. For example, the comment, "`I will fix it immediately!`," expresses a high level of *stress* of the writer. On the other hand, the comment, "`Yonik, Thanks and Congratulations!`," indicates a high level of *excitation*. In both cases, the writers of the comments use exclamation marks to put more emphasize on their feelings. Thus, we use a binary feature (e.g., hasExclamation) that captures presence of exclamation marks in texts.

Uppercase words: To put higher emphasize on emotional expressions, writers sometimes write few words using all uppercase/capital letters [16, 31] (e.g., GOOD, AWESOME, and BAD). For example, in the comment, "`SORRY Oliver, this is really my fault`," the writer expresses a higher level of sadness by writing the word 'Sorry' using all capital letters. We identify if any word written in all capital letters in a piece of text and record that information to use as a binary feature (e.g., hasAllCapitalLettersWord). In many cases, API names and code elements are written

Table 3: Emoticons and interjections expressing different emotions

Emotion	Keyword	Emoticon	Interjection
Excitation	Excited	:*, :?>, :x, :D, :)), 0:), @};- , =P~, :), :">	'Gee', 'Hurray', 'Ooh', 'Oooh', 'Wee', 'Wow', 'Yah', 'Yeah', 'Yeehaw'
Stress	Stressed	:((, X-(, :O, =;, :-/ , (:|, >:P	'Aah!', 'Aaah', 'Argh', 'Augh', 'Bah', 'Boo', 'Boo!', 'Booh', 'Eek', 'Eep', 'Grr', 'Yikes'
Depression	Depressed	:(, :-$, :-&, 8-), :-<, (:|, :-S, I-), :|	'Duh', 'Doh', 'Eww', 'Gah', 'Humph', 'Harumph', 'Oops', 'Oww', 'Ouch', 'Sheesh', 'Jeez', 'Yick'
Relaxation	Relaxed	B-), :>, :P, ;;), ;), =D, ;)), :-?, [-o<, /:)	'Ahh', 'Phew'

in all capital letters, which are discarded at preprocessing phase (see subsection 3.1).

Elongated words: Similar to uppercase word, writers use elongated words (e.g., Goood, Hurraaaay) to express intense emotions in informal written communications [16]. We identify existing of such elongated words in texts and record that information to use as a binary feature (e.g., hasElongatedWord). Similar to contractions, elongated words also adversely contribute to increase the size of features vector. For example, the elongated word 'Goood' will be considered as a unique word/feature, although it is a deviated form of the English word 'Good'. To reduce size of features vector, we correct the spellings of such identified elongated words found in texts.

Use of +1 and −1 in sentences: It is a common practice of developers to put +1 and −1 in comments while discussing technical issues among them in Stack Overflow and JIRA. When a developer likes or agrees on any issue with his colleague(s), then he puts +1 while commenting on that issue [2]. For example, in the comment "`+1, the new patch looks good`," the writer uses +1 (at the beginning) to express his positive disposition to a patch generated by his colleague. Thus, +1 in a comment indicates positive emotion, while −1 indicates the opposite. We identify presence of +1 and −1 in a text and create two binary features: i) hasPlusOne and ii) hasMinusOne.

3.3 Algorithm Selection

There are many supervised ML algorithms available [7] for classification problems. Among those, we select Scikit-learn's [7] implementations of following nine ML algorithms as those are popular and frequently used in sentiment/emotion classification.

(a) Adaptive Boosting (AB) [27], (b) Decision Tree (DT) [13, 27], (c) Gradient Boosting Tree (GBT) [47], (d) K-nearest Neighbors (KNN) [27], (e) Naive Bayes (NB) [27, 46], (f) Random Forest (RF) [56], (g) Multilayer Perceptron (MLP) [11], (h) Support Vector Machine with Stochastic Gradient Descent (SGD) [13], and (i) Linear Support Vector Machine (SVM) [16, 27].

4 EVALUATION

We use *precision* ($\wp$), *recall* ($\Re$), and *F-score* ($\exists$) to measure the accuracy of emotion detection of `MarValous` for each of the five emotional states (as described in Section 2). Given a set $\mathcal{T}$ of texts, *precision* ($\wp$), *recall* ($\Re$), and *F-score* ($\exists$) for a particular emotional state e is calculated as follows:

$$\wp = \frac{|\ \mathcal{T}_e \cap \mathcal{T}_e'\ |}{|\ \mathcal{T}_e'\ |}, \quad \Re = \frac{|\ \mathcal{T}_e \cap \mathcal{T}_e'\ |}{|\ \mathcal{T}_e\ |}, \quad \exists = \frac{2 \times \wp \times \Re}{\wp + \Re},$$

where $e \in \{excitation, stress, depression, relaxation, neutral\}$, $\mathcal{T}_e$ represents the set of texts expressing the emotional state e, and $\mathcal{T}_e'$ denotes the set of texts for which `MarValous` identifies the emotional state e.

4.1 Dataset

There is only one publicly available dataset where software engineering texts are manually annotated by Islam and Zibran [31] with four emotions *excitation, stress, depression,* and *relaxation.* Emotion-wise number of comments in that dataset are mentioned in the second column of Table 4.

Table 4: Number of comments in categories of emotions

Emotion	# of comments annotated by		Total # of Comments
	Islam and Zibran [31]	Novielli et al. [43]	
Excitation	411	1,709	2,120
Stress	252	988	1,240
Depression	289	230	519
Relaxation	227	0	227
Neutral	616	400	1,016
Overall total number of comments:			5,122

The number of comments are not adequate to train and test a ML classifier [53]. Hence, we increase the number of comments by leveraging the dataset created by Novielli et al. [43]. They release a dataset of 4,800 questions, answers, and comments collected from Stack Overflow, which are manually annotated using six basic emotions, namely *love*, *joy*, *anger*, *sadness*, *fear*, and *surprise*. According to emotion classification model in the two-dimensional approach [23] (as seen in Figure 1) the emotions *love* and *joy* fall in the first quadrant, emotions *anger* and *fear* fall in the second quadrant and emotion *sadness* falls in the third quadrant. Hence, we select those comments that are annotated with the emotions *love, joy, anger, fear* and *sadness* and assigned those comments the representative emotions of their respective quadrants in which they belong.

In some cases, an expression of surprise can be positive, while in other cases it can convey a negative sentiment/valence [30]. As the polarities along the valence dimension are not defined for the surprised comments in the dataset of Novielli et al., we exclude those surprised comments to minimize ambiguity. Finally, we randomly pick 400 neutral comments of the dataset. For each of the emotions, we mention the number of comments collected from the dataset of Novielli et al. in the third column of Table 4. The combined dataset consists of 5,122 comments.

Table 5: Comparison of ML algorithms in classification of emotional states

Emotion	Metrics	AB	DT	GBT	KNN	NB	RF	MLP	SGD	SVM
Excitation	$\wp$	80.91%	86.82%	**90.12%**	82.31%	50.92%	80.67%	89.62%	85.88%	89.65%
	$\Re$	88.47%	86.98%	93.13%	89.36%	**98.72%**	90.64%	92.34%	93.60%	93.77%
	$\exists$	84.48%	86.88%	91.59%	85.68%	67.13%	85.31%	90.96%	89.28%	**91.65%**
Stress	$\wp$	55.08%	63.70%	**79.67%**	75.63%	85.94%	70.42%	74.61%	78.50%	75.81%
	$\Re$	44.79%	54.63%	71.26%	40.33%	22.31%	49.19%	75.55%	69.80%	**76.99%**
	$\exists$	48.81%	58.64%	75.10%	52.39%	35.26%	57.83%	74.96%	73.44%	**76.32%**
Depression	$\wp$	61.97%	56.01%	**83.97%**	59.26%	20.00%	63.74%	70.83%	74.56%	78.27%
	$\Re$	25.26%	51.25%	**63.66%**	56.52%	00.37%	42.91%	62.25%	62.32%	60.73%
	$\exists$	34.88%	53.17%	**72.20%**	57.36%	00.72%	51.13%	66.02%	67.27%	68.14%
Relaxation	$\wp$	54.20%	64.38%	84.77%	67.41%	00.00%	80.07%	82.22%	80.35%	**86.77%**
	$\Re$	52.07%	59.17%	71.37%	62.72%	00.00%	51.65%	75.62%	76.38%	**77.15%**
	$\exists$	52.46%	61.22%	77.04%	64.48%	00.00%	62.21%	78.51%	77.29%	**81.29%**
Neutral	$\wp$	58.37%	68.86%	76.83%	59.56%	75.51%	67.54%	86.18%	86.33%	**86.35%**
	$\Re$	77.45%	84.06%	88.94%	84.96%	50.21%	**91.61%**	82.09%	84.38%	87.99%
	$\exists$	65.42%	75.65%	84.45%	69.91%	59.96%	77.68%	84.00%	85.33%	**87.09%**
Overall average accuracy	$\wp$	62.10%	67.95%	83.07%	68.83%	46.47%	72.49%	80.69%	81.12%	**83.37%**
	$\Re$	57.61%	67.22%	77.67%	66.78%	34.32%	65.20%	77.57%	77.29%	**79.33%**
	$\exists$	57.21%	67.11%	80.08%	65.96%	32.61%	66.83%	78.89%	78.52%	**80.90%**

Table 6: Comparison of features in `MarValous`

Emotion	Met.	All feat.	η	$\eta + \beta$	$\eta + \gamma$	$\eta + \delta$
Excitation	$\wp$	**89.65%**	88.03%	87.81%	88.42%	88.17%
	$\Re$	93.77%	**94.81%**	94.77%	94.51%	94.34%
	$\exists$	**91.65%**	91.27%	91.14%	91.35%	91.14%
Stress	$\wp$	**75.81%**	72.36%	72.30%	72.70%	72.47%
	$\Re$	**76.99%**	74.85%	75.13%	76.17%	74.89%
	$\exists$	**76.32%**	73.41%	73.62%	74.31%	73.52%
Depression	$\wp$	**78.27%**	69.08%	69.71%	70.53%	70.21%
	$\Re$	**60.73%**	56.23%	55.20%	55.72%	56.11%
	$\exists$	**68.14%**	61.85%	61.32%	62.14%	61.94%
Relaxation	$\wp$	86.77%	88.10%	88.18%	88.05%	**88.96%**
	$\Re$	**77.15%**	73.79%	76.26%	76.66%	75.65%
	$\exists$	81.29%	79.91%	81.46%	81.56%	**81.57%**
Neutral	$\wp$	86.35%	89.58%	**89.29%**	88.97%	88.53%
	$\Re$	**87.99%**	82.37%	82.17%	83.15%	83.88%
	$\exists$	**87.09%**	85.76%	85.52%	85.88%	86.11%
Overall average accuracy	$\wp$	**83.37%**	81.43%	81.46%	81.73%	81.67%
	$\Re$	**79.33%**	76.41%	76.70%	77.24%	76.97%
	$\exists$	**80.90%**	78.44%	78.61%	79.05%	78.86%

4.2 Evaluation of ML Algorithms

Here we seek to identify which ML algorithm shows the best performance on the dataset. We use 10-fold cross-validations to validate each of the algorithms, where the dataset is randomly divided into 10 groups and each of the ten groups is used as test dataset once, while the remaining nine groups are used to train the classifier.

For each of the ML algorithms, we run `MarValous` on the dataset and compute averages of precisions, recalls, and f-measures for *10-fold cross-validations* for each of the emotional states. The computed metrics' values are presented in Table 5. For each ML algorithm, the overall average precision, recall, and F-score across all the emotional states are presented in the last three rows. The highest obtained value of a metric in each row is boldfaced for better interpretation of the results.

In Table 5, we see SVM obtains the highest F-score values for each of the emotional states except *depression*. For emotions *depression* and *relax*, GBT and SVM obtain the best performances respectively, for all the metrics. For emotions *excitation* and *stress*, GBT shows the best results for metric precision, while NB and SVM obtain the highest recall values for the particular emotions respectively. Although NB performs relatively better in detecting emotions *excitation*, *stress* and *neutral*, which have higher number of comments, it performs very low in detecting *relaxation* and *depression* that have lower number comments. On the other hand, for *neutral* comments, RF obtains the highest recall value, whereas, SVM obtains the highest precision value followed by SGD.

Notably, SGD and MLP algorithms show promising results, although their obtained overall average accuracies' values are lower than the metrics' values obtained by SVM and GBT algorithms. SVM performs the best in terms of overall average precision, recall and F-score (see last three rows of last column in Table 5) that indicates the algorithm shows steady performances across all the emotional states. Hence, among the nine ML algorithms, we select this ML algorithm as default for our `MarValous` tool and conduct our subsequent analyses.

4.3 Evaluation of Features in `MarValous`

The selection and quality of the features representing each class affect the accuracy and efficiency of classification of ML algorithms [50]. Identifying and removing irrelevant and unnecessary features increase learning accuracy and improve comprehensibility of results. Toward that we measure importance of the seven features of `MarValous` in classifying four emotional states of the comments in our dataset.

First, based on the similarities of the features we divide those features into four disjoint sets: i) η ={*unigram* and *bigram*}, ii) β ={*emoticons* and *exclamatory mark*}, iii) γ={*all capital letters*, *elongated word* and *interjection*} and iv) δ={*plus one* and *minus one*}. Then, for various combinations of those features' sets, we run MarValous on our dataset. The combinations of those features and computed results for each of the combinations are presented in Table 6.

As seen in Table 6, in all the cases, precision, recall and F-score values are always higher when MarValous is operated with all features except in four cases. Those four exception cases include: i) the highest recall value obtained using only n-gram feature for comments belong to *excitation* emotion, ii) the best precision and F-score values obtained using the combination of η and δ features for emotion *relaxation*, and iii) the highest precision value obtained using the combination of η and β features for *neutral* comments. However, those exception cases cannot prevent the combination of all features to obtain the highest overall average accuracies (see last three rows of third column in Table 6).

By observing the differences of metrics' value presented in the third and fourth columns of Table 6, we see n-gram (η) features i.e., unigram and bigram contribute the most significant part of the accuracies of our tool MarValous. Although other features' contributions are not significant, those features play their roles in increasing overall accuracies of our tool MarValous. As SVM algorithm with all the seven features has showed the best performance, we have released MarValous by setting SVM as its default algorithm while enabling all those features in it.

4.4 Comparison with DEVA

We compare our tool's accuracies with those of DEVA [31]. Until the public release of our MarValous, DEVA is the only available SE domain specific tool for the detection of the four emotional states that our tool also detects. While DEVA uses a lexicon based approach, our MarValous relies on ML techniques.

To ensure a fair comparison between the tools, we randomly divide the dataset into two subsets: i) training set contains 70% (i.e., 3,586) comments of the dataset and ii) test set contains remaining 30% (i.e., 1,536) comments. The training set is used to train our ML based tool MarValous and the test set is used to compute and compare performances of DEVA and MarValous.

We separately operate DEVA and MarValous to detect the emotional states in each of the comments in the test set. Then, we compute the precision ($\wp$), recall ($\Re$), and F-score ($\exists$) for their detections of each emotional states (i.e., excitation, stress, depression, relaxation, and neutral) as presented in Table 7. The overall average precision, recall, and F-score across all the emotional states are presented in the bottom three rows.

We see in Table 7 that MarValous outperforms the baseline tool DEVA in all cases by a large margin except for the recall values of *stressed* and *neutral* comments and precision value of comments belong to *excitation* emotion. MarValous maintains a proper balance between precision and recall for each emotional state resulting in higher F-score in each emotional state. Overall, on average, across all the emotions, MarValous clearly outperforms the baseline, as it has achieved 19.04% higher precision and 08.19% higher recall values compared to DEVA.

Table 7: Comparison between DEVA and MarValous

Emotions	Metrics	DEVA	MarValous
Excitation	$\wp$	**91.26%**	86.86%
	$\Re$	78.07%	**93.36%**
	$\exists$	84.15%	**89.99%**
Stress	$\wp$	43.25%	**79.82%**
	$\Re$	**70.32%**	56.13%
	$\exists$	53.56%	**65.91%**
Depression	$\wp$	36.92%	**90.00%**
	$\Re$	55.81%	**73.26%**
	$\exists$	44.44%	**80.77%**
Relaxation	$\wp$	75.10%	**75.31%**
	$\Re$	48.35%	**76.08%**
	$\exists$	58.82%	**75.70%**
Neutral	$\wp$	74.41%	**84.18%**
	$\Re$	**93.38%**	88.08%
	$\exists$	82.82%	**86.08%**
Overall average accuracy	$\wp$	64.19%	**83.23%**
	$\Re$	69.19%	**77.38%**
	$\exists$	64.76%	**79.69%**

Table 8: Contingency matrix of McNemar's test

# of comments misclassified by both $\mathcal{T}_a$ and $\mathcal{T}_b$	n_{00} = 146	n_{01} = 113	# of comments misclassified by $\mathcal{T}_b$ but not by $\mathcal{T}_a$
# of comments misclassified by $\mathcal{T}_a$ but not by $\mathcal{T}_b$	n_{10} = **293**	n_{11} = 984	# of comments correctly classified by both $\mathcal{T}_a$ and $\mathcal{T}_b$

Here, $\mathcal{T}_a$ = DEVA and $\mathcal{T}_b$ = MarValous

Statistical significance. We apply a non-parametric *McNemar's* test [20, 30] at significance level $\alpha = 0.05$ to verify the statistical significance in the difference of the results obtained by the two tools, DEVA and MarValous. As the non-parametric test does not require normal distribution of data, this test suits well for our purpose.

We perform a *McNemar's* test on a 2×2 contingency matrix (Table 8) derived from the results obtained from the two tools. A number/frequency in a cell is denoted as n_{xy}, where x and y denote row and column numbers respectively of a cell in the contingency table. According to the table, MarValous ($\mathcal{T}_b$) performs better than DEVA ($\mathcal{T}_a$) as $n_{10} > n_{01}$. The difference of performances is found to be statistically significant with p-value = 2.69×10^{-11} where $p < \alpha$. We, therefore, conclude that the observed superior performance of our MarValous over DEVA is statistically significant.

5 LIMITATIONS AND THREATS TO VALIDITY

To have a large dataset, we combine two datasets of Islam and Zibran [31] and Novielli et al. [43], which are created by following *two-dimensional* and *discrete emotional models* respectively. While those two emotional models different from each other, we map categories of emotions from discrete emotional model to the two-dimensional model (see Section 4.1), which can be questioned. To validate our mapping process, we randomly pick 20 comments from each category of emotions and manually verify the correctness of mappings.

Although we combine two datasets to increase the size of the used dataset, the number of comments is still low (total 5,122) for training a ML-based classifier. Thus, the generalizability of MarValous can be questioned. However, higher performances of MarValous on combined dataset, which consists of comments collected from two different data sources (JIRA issue comments and Stack Overflow comments), give us confidence that it will perform good enough on different types of software engineering textual artifacts too. Moreover, the newly combined dataset is not perfectly balanced as the emotion relaxation consists of only 04.43% of all comments. However, such imbalance in the dataset could not adversely affect the performance of MarValous.

While there are many ML algorithms available, we have chosen the nine selected algorithms based on their popularity in the community. Moreover, in the selection of those algorithms, we ensure that the selection encompasses varieties of ML algorithms, e.g., Generalized Linear Models (GLM), Support Vector Machine (SVM), Stochastic Gradient Descent (SGD), Nearest Neighbors (NN), ensemble methods and others. There are still scopes to examine performances of other ML algorithms.

Overfitting can be a common threat to any ML based tool. To make sure there is no such overfitting of MarValous accuracies, we use ten-fold cross validations to measure accuracies of the tool. Moreover, we verify the superior performance of MarValous over DEVA using the 30% of the comments, which were never used at the training phase. Such precautions should have limited the threats of overfitting.

The lists of emoticons, interjections and slangs might not have included all available such items. Missing of items from those lists might have restricted the performance of MarValous. Keyword based detection of APIs' names in texts might not be 100% accurate either. However, these limitations also apply to lexicon-based approaches (such as DEVA). We plan to minimize these limitations in our future work.

6 RELATED WORK

Earlier research related to sentiment analysis in software engineering text mostly used three tools, SentiStrength [53], Stanford NLP [9], and NLTK [6]. SentiStrength is the most frequently [30] used tool among those three tools. All of the aforementioned three tools were developed and trained to operate on general purpose texts (e.g., movie and hotel reviews) and they did not perform well enough when operated in a particular domain such as software engineering [16, 33]. Sentiment analyses using these tools are misleading mainly due to the variations in meanings of domain-specific technical terms [30] and high amount of noises (e.g., code snippets, URL and API names) [10, 16, 30].

Blaz and Becker [14] proposed three lexical based methods that consists of a dictionary method, a template method, and a hybrid method for sentiment analysis in job submission tickets related to Information Technology (IT). Those three techniques were tested on formally written texts generated in closed official environment and may not perform well in dealing with *informal* and *noisy* texts frequently used in software engineering artifacts such as commit and code review comments [30]. SentiStrength-SE [30], Senti4SD [16] and SentiCR [10] are three recent sentiment analysis tools especially designed by targeting software engineering text. However, all the aforementioned tools can detect *valence (aka sentiment) only* and cannot other emotional states in more fine-grained levels.

To detect emotions at deeper levels, Murgia et al. [38] constructed a Machine Learning (ML) classifier to identify six emotions *joy, love, surprise, anger, sad,* and *fear* in issue comments related to software development. In another work, Calefato et al. [17] also developed a ML based toolkit named *EmoTxt* to detect those six emotions from technical posts in Stack Overflow. However, neither of these techniques are capable of detecting the emotional states in the well-established bi-directional emotional model that includes both *valence* and *arousal* dimensions [31].

To address the above mentioned issue, Islam and Zibran [31] developed a dictionary and rule-based tool DEVA to identify four emotional states *excitation, stress, depression/sadness*, and *relaxation* in the bi-directional emotional model. Our work is inspired by DEVA. The lexical approach of DEVA is based on limited-size dictionaries and a set of predefined rules. Thus, the performance of DEVA is inherently limited by the quality and size of the dictionaries [21]. Moreover, DEVA will fail to correctly classify emotions when encountered certain textual structures and content, which are not covered by the rules and dictionaries. To overcome such limitations, in MarValous, we have used ML techniques and we have also demonstrated that our ML approach performs substantially better than the lexical approach of DEVA.

TensiStrength [54] is a tool released before DEVA, and it that can detect *stress* and *relaxation* in texts, but cannot capture *excitation* or *depression*. Unlike MarValous and DEVA, TensiStrength is not particularly designed for any specific domain, and thus produce inferior performance compared to DEVA in dealing with software engineering texts [31]. Our MarValous substantially outperforms DEVA as found from the quantitative comparison in this work.

7 CONCLUSION

In this paper, we have presented MarValous, which is the first Machine Learning (ML) based tool especially designed for software engineering text to detect individual emotional states *excitation, stress, depression, relaxation* and *neutrality*. By using nine preprocessing steps and seven features, we have developed the tool MarValous that consists of nine popular and effective supervised ML algorithms for emotion/sentiment analysis.

For evaluating the ML algorithms in MarValous, we have combined two existing manually annotated datasets that consists of 5,122 comments collected form JIRA and Stack overflow. From a quantitative evaluation using this dataset, the Linear Support Vector Machine (SVM) algorithm is found to exhibit the best performance (overall average precision 83.37% and recall 79.33%) followed by GBT. The algorithms SGD and MLPC have also showed promising results.

Next, to find an optimal features' set, we have evaluated various combinations of the seven features of MarValous using the highest performed SVM algorithm. We have found that the set of all the seven features have achieved the best overall average accuracies

compared to the selected subsets of those features. As SVM algorithm with all the seven features has showed the best performance, we have released `MarValous` by setting SVM as its default algorithm while enabling all those features in it.

We have also compared the performance of `MarValous` (with default settings) against the state-of-the-art tool `DEVA`. From the quantitative comparisons, `MarValous` is found to achieve 19.04% higher precision and 08.19% higher recall compared to `DEVA`. A statistical test has confirmed the significant superiority of `MarValous` over `DEVA`. The current release of `MarValous` and combined benchmark dataset are freely available [1] for public use.

Creating a larger dataset of comments annotated with four emotional states remains within our immediate future work. We will put efforts in minimizing the limitations of `MarValous` to improve its performance. We will also conduct empirical studies of emotions and their probable impacts in software engineering using our `MarValous`.

ACKNOWLEDGEMENT

This work is supported in part by the SCoRe grant at the University of New Orleans.

REFERENCES

[1] Sept 2018. *Download MarValous.* https://figshare.com/s/a3308b7087df910db38f.
[2] Verified: Aug 2018. *The Social Media Glossary: 226 Essential Definitions.* https://blog.hootsuite.com/social-media-glossary-definitions/.
[3] Verified: Aug 2018. *SyntaxNet: Neural Models of Syntax.* https://github.com/tensorflow/models/tree/master/research/syntaxnet.
[4] Verified: Aug 2018. *TF-IDF.* http://www.tfidf.com.
[5] Verified: Sept 2018. *Industrial-Strength Natural Language Processing.* https://spacy.io.
[6] Verified: Sept 2018. *Natural Language Toolkit for Sentiment Analysis.* http://www.nltk.org/api/nltk.sentiment.html.
[7] Verified: Sept 2018. *Scikit-learn: Machine Learning in Python.* http://scikit-learn.org/stable/.
[8] Verified: Sept 2018. *Snowball.* http://snowballstem.org.
[9] Verified: Sept 2018. *Stanford Core NLP Sentiment Annotator.* http://stanfordnlp.github.io/CoreNLP/sentiment.html.
[10] T. Ahmed, A. Bosu, A. Iqbal, and S. Rahimi. 2017. SentiCR: a customized sentiment analysis tool for code review interactions. In *ASE*. 106–111.
[11] D. Alboaneen, H. Tianfield, and Y. Zhang. 2017. Sentiment Analysis via Multi-Layer Perceptron Trained by Meta-Heuristic Optimisation. In *ICBD*. 4630–4635.
[12] N. Bettenburg, B. Adams, and A. Hassan. 2011. A Lightweight Approach to Uncover Technical Information in Unstructured Data. In *ICPC*. 185–188.
[13] A. Bifet and E. Frank. 2010. Sentiment knowledge discovery in twitter streaming data. In *DS*. 1–15.
[14] C. Blaz and K. Becker. 2016. Sentiment Analysis in Tickets for IT Support. In *MSR*. 235–246.
[15] F. Calefato and F. Lanubile. 2016. Affective Trust as a Predictor of Successful Collaboration in Distributed Software Projects. In *SEmotion*. 3–5.
[16] F. Calefato, F. Lanubile, F. Maiorano, and N. Novielli. 2017. Sentiment Polarity Detection for Software Development. *Emp. Softw. Eng.* (2017), 1–31.
[17] F. Calefato, F. Lanubile, and N. Novielli. 2017. EmoTxt: A Toolkit for Emotion Recognition from Text. In *ACII*.
[18] A. Ciurumelea, A. Schaufelbuhl, S. Panichella, and Harald Gall. 2017. Analyzing Reviews and Code of Mobile Apps for Better Release Planning. In *SANER*. 91–102.
[19] A. D'Andrea, F. Ferri, P. Grifoni, and T. Guzzo. 2015. Approaches, Tools and Applications for Sentiment Analysis Implementation. *International Journal of Computer Applications* 125, 3 (2015), 26–33.
[20] T. Dietterich. 1998. Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms. *J. of Neural Comp.* 10, 7 (1998), 1895–1923.
[21] J. Ding, H. Sun, X. Wang, and X. Liu. 2018. Entity-Level Sentiment Analysis of Issue Comments. In *SEmotion*. 7–13.
[22] T. Eerola and J. Vuoskoski. 2010. A Comparison of the Discrete and Dimensional Models of Emotion in Music. *Psychology of Music* 39, 1 (2010), 18–49.
[23] B. Martínez; A. Rodrigo; R. Cantabrana; J. García and R. Alcaraz. 2016. Application of Entropy-Based Metrics to Identify Emotional Distress from Electroencephalographic Recordings. *Entropy* 18, 6 (2016).
[24] D. Garcia, M. Zanetti, and F. Schweitzer. 2013. The Role of Emotions in Contributors Activity: A Case Study on the GENTOO Community. In *CCGC*. 410–417.
[25] D. Graziotin, X. Wang, and P. Abrahamsson. 2013. Are Happy Developers more Productive? The Correlation of Affective States of Software Developers and their self-assessed Productivity. In *PROFES*. 50–64.
[26] E. Guzman and B. Bruegge. 2013. Towards Emotional Awareness in Software Development Teams. In *ESEC/FSE*. 671–674.
[27] M. Hasan, E. Rundensteiner, and E. Agu. 2014. EMOTEX: Detecting Emotions in Twitter Messages. In *ASE*. 27–31.
[28] M. Islam and M. Zibran. 2016. Exploration and Exploitation of Developers' Sentimental Variations in Software Engineering. *Internation Journal of Software Innovation* 4, 4 (2016), 35–55.
[29] M. Islam and M. Zibran. 2016. Towards Understanding and Exploiting Developers' Emotional Variations in Software Engineering. In *SERA*. 185–192.
[30] M. Islam and M. Zibran. 2017. Leveraging Automated Sentiment Analysis in Software Engineering. In *MSR*. 203–214.
[31] M. Islam and M. Zibran. 2018. DEVA: Sensing Emotions in the Valence Arousal Space in Software Engineering Text. In *SAC*. 1536–1543.
[32] M. Islam and M. Zibran. 2018. Sentiment Analysis of Software Bug Related Commit Messages. In *SEDE*.
[33] M. Islam and M. Zibran. 2018. SentiStrength-SE: Exploiting domain specificity for improved sentiment T analysis in software engineering text. *Journal of System and Software* 145 (2018), 125–146.
[34] R. Jongeling, S. Datta, and A. Serebrenik. 2015. Choosing Your Weapons: On Sentiment Analysis Tools for Software Engineering Research. In *ICSME*. 531–535.
[35] I. Khan, W. Brinkman, and R. Hierons. 2010. Do Moods Affect Programmers' Debug Performance? *Cogn. Technol. Work* 13, 4 (2010), 245–258.
[36] L. Khreisat. 2006. Arabic Text Classification Using N-Gram Frequency Statistics A Comparative Study. In *CDM*. 78–82.
[37] D. McDuff, A. Karlson, A. Kapoor, A. Roseway, and M. Czerwinski. 2012. AffectAura: An Intelligent System for Emotional Memory. In *CHI*. 849–858.
[38] A. Murgia, M. Ortu, P. Tourani, and B. Adams. 2017. An Exploratory Qualitative and Quantitative Analysis of Emotions in Issue Report Comments of Open Source Systems. *Empirical Software Engineering* (2017), 1—44.
[39] A. Murgia, P. Tourani, B. Adams, and M. Ortu. 2014. Do Developers Feel Emotions? An Exploratory Analysis of Emotions in Software Artifacts. In *MSR*. 261–271.
[40] M. Nayebi, H. Farrahi, and G. Ruhe. 2017. Which Version Should be Released to the App Store?. In *ESEM*. 324–333.
[41] N. Novielli, F. Calefato, and F. Lanubile. 2014. Towards Discovering the Role of Emotions in Stack Overflow. In *SSE*. 33–40.
[42] N. Novielli, F. Calefato, and F. Lanubile. 2015. The Challenges of Sentiment Detection in the Social Programmer Ecosystem. In *SSE*. 33–40.
[43] N. Novielli, F. Calefato, and F. Lanubile. 2018. A Gold Standard for Emotion Annotation in Stack Overflow. In *MSR*. 14–17.
[44] F. Omran and C. Treude. 2017. Choosing an NLP Lib. for Analyzing Softw. Documentation: A Systematic Lit. Review and a Series of Exp.. In *MSR*. 187–197.
[45] M. Ortu, B. Adams, G. Destefanis, P. Tourani, M. Marchesi, and R. Tonelli. 2015. Are Bullies More Productive? Empirical Study of Affectiveness vs. Issue Fixing Time. In *MSR*. 303–313.
[46] B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment Classification Using Machine Learning Techniques. In *EMNLP*. 79–86.
[47] M. Pennacchiotti and A. Popescu Democrats. 2011. Republicans and Starbucks Afficionados: User Classification in Twitter. In *KDD*. 430–438.
[48] D. Pletea, B. Vasilescu, and A. Serebrenik. 2014. Security and Emotion: Sentiment Analysis of Security Discussions on GitHub. In *MSR*. 348–351.
[49] L. Polanyi and A. Zaenen. 2006. Contextual Valence Shifters. *Computing Attitude and Affect in Text: Theory and Applications* 20 (2006), 1–10.
[50] M. Raymer, W Punch, E Goodman, L Kuhn, and A Jain. 2000. Dimensionality reduction using genetic algo. *IEEE Trans. on Evol. Comp.* 4, 2 (2000), 164–171.
[51] J. Russell and A. Mehrabian. 1977. Evidence for a Three-factor Theory of Emotions. *Journal of Research in Personality* 11, 3 (1977), 273–294.
[52] R. Socher, A. Perelygin, J. Wu, J. Chuang, C. Manning, A. Ng, and C. Potts. 2013. Recursive Deep Models for Semantic Compositionality over a Sentiment Treebank. In *EMNLP*. 1631–1641.
[53] M. Thelwall. 2013. Heart and Soul: Sentiment Strength Detection in the Social Web with Sentistrength,. In *CyberEmotions*. 1–14.
[54] M. Thelwall. 2017. TensiStrength: Stress and relaxation magnitude detection for social media texts. *Information Processing and Management* 53, 1 (2017), 106–121.
[55] P. Tourani and B. Adams. 2016. The Impact of Human Discussions on Just-in-time Quality Assurance. In *SANER*. 189–200.
[56] Y. Wan and D. Gao. 2015. An Ensemble Sentiment Classification System of Twitter Data for Airline Services Analysis. In *ICDMW*. 1318–1325.
[57] M. Wrobel. 2013. Emotions in the Software Development Process. In *HSI*. 518–523.
[58] C. Yang, K. Lin, and H. Chen. 2009. Writer Meets Reader: Emo. Analysis of Soc. Media from both the Writer's and Reader's Perspectives. In *WIIAT*. 287–290.
[59] G. Yang, S. Baek, J. Lee, and B. Lee. 2017. Analyzing Emotion Words to Predict Severity of Softw. Bugs: A Case Study of Open Source Proj.. In *SAC*. 1280–1287.
[60] D. Ye, Z. Xing, C. Foo, Z. Ang, J. Li, and N. Kapre. 2016. Software-Specific Named Entity Recognition in Software Engineering Social Content. In *SANER*. 90–101.