Embedded Emotion-based Classification of Stack Overflow Questions Towards the Question Quality Prediction

Amit Kumar Mondal Mohammad Masudur Rahman Chanchal K. Roy
University of Saskatchewan, Canada
{amit.mondal, masud.rahman, chanchal.roy}@usask.ca

***Abstract*—Software developers often ask questions in Stack Overflow Q & A site, and their posted questions sometimes do not meet the standard guidelines. As a consequence, some of the questions are edited by expert users, some of them are down-voted, or some are even deleted permanently. Besides, the users (i.e., developers) might not get the expected solutions for their problems. In this paper, we study up-voted and down-voted questions from Stack Overflow, and analyze the relationship of embedded emotions with question quality. We use Sentiment140 API for identifying embedded emotions in the question texts, and then apply Feed-Forward Multilayer Perceptron (MLP) and Support Vector Machine (SVM) on the emotion data for developing a quality prediction model. Experiments using 38,920 Stack Overflow questions suggest about 70% precision and about 74% recall for our model with 10-fold cross-validation, and these findings clearly reveal the impact of human emotions upon the quality of a question.**

Keywords – Sentiment analysis; question quality; machine learning; StackOverflow

I. Introduction

Software developers often search for solutions in the web for their encountered problems. Programming Q & A sites host millions of programming related questions that are answered by expert individuals from the community. Stack Overflow (hereby SO) is one of the most widely used programming Q & A sites that helps the developers harvest knowledge on programming problems and solutions. It contains millions of questions and answers, and has a median response time of 11 minutes [18]. However, due to poor quality in the question content, the users (i.e., developers) might not get the appropriate answers, and the questions might need to be edited by the experts, or it could be down-voted or even deleted by moderators. Asaduzzaman et al. [3] suggest that unanswered questions are increasing rapidly. They report about 299K questions that were posted during 2008 to 2012 and still are unanswered. Rahman and Roy [18] report that up to February, 2015, about 27% of StackOverflow questions were unresolved. Correa and Sureka [6] identify about 293K questions that were posted during 2009 to 2013 and were later deleted due to off-topic or low quality content. Thus, question quality is a major concern for Stack Overflow. The site also attempts to ensure the content quality standard and minimize the noise by setting up a clear guideline[1] for asking questions. However, the volume of the question is rapidly increasing, and manual checking and assurance of the questions' quality is prohibitively costly and highly challenging. One way to handle this challenge is to employ an automatic technique that can predict the quality of a question reliably during its submission. Based on our preliminary observation on 1000 question samples, we believe that human emotions or sentiments embedded in the texts of a question might affect the perception about the quality of the question. Too much negative words (i.e., pessimistic views) in the text of a question might increase the possibility of getting the question down-voted or deleted permanently or at least the question needs to be edited by the experts. This leads us to consider a special aspect, the embedded emotions or sentiments of an about-to-post question's text for predicting its quality. We thus attempt to answer the following two research questions in this paper:

- **RQ$_1$**: Is quality of a question affected by its author's emotions/sentiments embedded in the texts?
- **RQ$_2$**: Can a predictive model effectively predict the quality of the question based on the sentiment metrics?

Existing studies focus on different dimensions for classification or efficient management of SO questions and answers. For classification or post quality prediction, textual features of a question [2, 16], characteristics of a code segment [8], social and emotional factors [14], and various key-terms [20] are analyzed. Serva et al. [20] mine negative code examples from Stack Overflow using sentiment analysis where they use a set of key-terms as negativity indicators. However, such indicators might not be sufficient enough to classify a programming related question given that the question generally contains both texts and code segments. Thus, we leverage the human emotions embedded in the text of a question for predicting its quality from Stack Overflow. In this paper, we report an empirical study using 38,920 questions from Stack Overflow, and propose a sentiment metric based machine learning model for identifying low-quality and high-quality questions. For question quality prediction, we combine two categories of features– TF-IDF [19] of the key-terms and sentiment polarity of the sentences in the text from a question. In particular, we consider the frequencies of three types of sentences *–positive, negative* and *neutral* from the question texts as sentiment metrics detected by Sentiment140 API.

[1] http://stackoverflow.com/help/how-to-ask

DOI reference number: 10.18293/SEKE2016-146

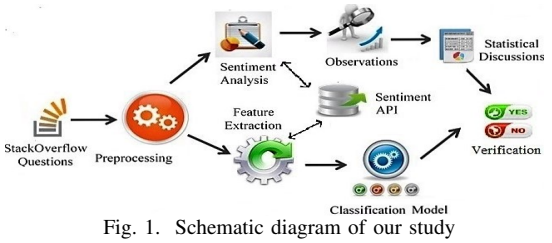

Fig. 1. Schematic diagram of our study

Our proposed classifier model(s) can classify questions with about 70% precision and 74.2% recall. These are 8% and 2.72% improvements respectively over baseline performance [20]. We also note that sentiment-metrics alone can classify the question from Stack Overflow with about 68.50% precision and about 73.80% recall. Such findings validate our problem statement that human emotions/sentiment embedded in the question text can be a potential indicator of its quality. Thus, we make the following contributions in this paper:

- An exploratory study on how embedded emotions in the text of a question can affect the perception of its overall quality using 38,920 questions from Stack Overflow.
- A question quality prediction model by employing novel sentiment based metrics.

II. Methodology

Salient feature extraction from the raw data for item classification is a challenging task. In Stack Overflow, each question generally has natural language texts and code segment(s) or code-like elements. Since we focus on embedded emotions in the question texts, we extract the items related to human emotions or sentiments carefully from the questions. Our research methodology comprises of three main steps–(1) Dataset preparation, (2) Exploratory analysis, and (3) Question quality model design. Fig. 1 shows the schematic diagram of these steps. In the following section we describe each of these steps in details.

A. Dataset Preparation

In this research, we wanted to study recent questions from Stack Overflow. We thus collect questions submitted between January, 2014 and January, 2015. We first collect 50K questions using Stack Exchange Data Explorer[1]. We found 3,239 down-voted questions, among them only 163 have vote-value less than - 4. In order to increase the down-voted samples in our dataset, we collect another 7,235 questions that were posted between January, 2013 and January, 2014, and they have vote value < -4. We discard the questions with 0 vote or having #words < 5 in the text body from the collected questions. Thus we have a collection of 38,920 questions, 73% of them are up-voted (i.e., $score > 0$) whereas the rest 10,474 (27%) are down-voted (i.e., $score < 0$). Given that concept of quality might be subjective, we consider that a question is of low quality if it is down-voted by the technical crowd of Stack Overflow and vice versa as was also considered by [2, 8, 16, 20]. We then label the low quality and high quality questions using '-1' and '+1' respectively which gives two classes convenient for our SVM and MLP classifier models. The texts from each Stack Overflow question contain various HTML tags including `<code>` tags. These `<code>` tags generally contain code segments or code like elements [18]. Since we focus on natural language texts from a question, we separate the rest content from `<code>` tags, and consider them as the content of interest from the question.

[1]http://data.stackexchange.com/stackoverflow/query/new

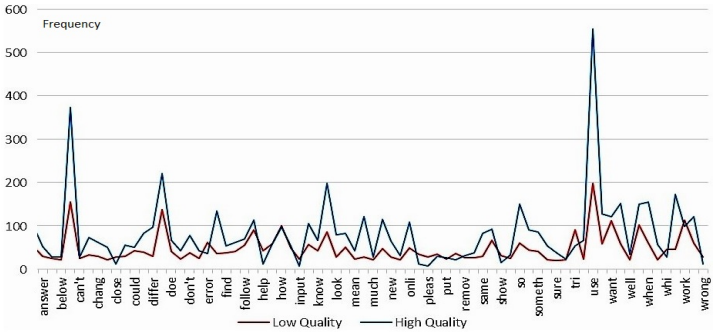

Fig. 2. Base term histograms for low quality and high quality questions from Stack Overflow (the graph is for 76 terms while the diagram producing systems hide most of the terms)

B. Exploratory Study

In order to answer RQ_1, we randomly select 1,000 questions from our dataset containing 500 up-voted questions and 500 down-voted questions. We first perform natural language preprocessing such as stop words (i.e., I, we, to, and, Java...etc.) removal, word splitting and stemming [18], and then determine the term frequency to derive meaningful insights. Fig. 2, and 3 summarize our comparative analysis between different types of questions.

We collect and investigate frequent terms from down-voted questions where we consider a heuristic term frequency of 20 (i.e., to limit the list to the words with greater significance). Since existing studies [20] analyze such questions to extract negative code examples, we select those questions as a starting point for our analysis. This step provides us a collection of 76 terms which we call as *base terms*. We then determine frequencies of the base terms in up-voted questions. Fig. 2 shows the histograms of base terms from the two sets of questions. We note that up-voted questions have relatively greater frequencies compared to the down-voted questions. For example, we see the base terms–'below', 'differ', and 'use' for up-voted questions have the greater frequencies. Mann Whitney U-tests (for up-voted and down-voted questions, the $p-value$ are 0.000060 and 0.00016 which are less than threshold 0.05) also show significant differences in their frequencies. All these findings suggest that there might exist a significant and meaningful difference in the term distribution between up-voted and down-voted questions from Stack Overflow.

We also qualitatively investigate the base terms (i.e., highly frequent words), and notice that some of them reflect human emotions, and are also considered by existing sentiment-based studies [20]. Due to this, it is not enough to apply indicators words for sentiment for the posts.

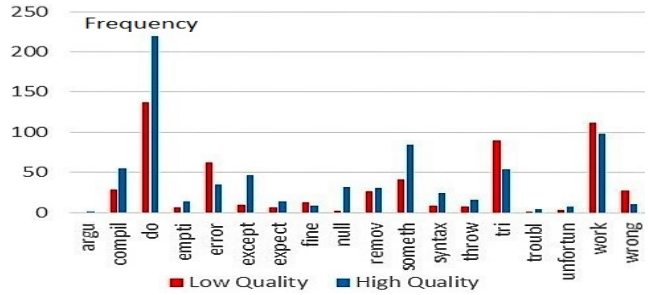

Fig. 3. Histograms of indicator terms from low quality and high quality questions (words are stemmed)

TABLE I
INDICATOR TERMS (TAKEN FROM [20]) AND TOP RANKED NEGATIVE SYNSETS (TAKEN FROM[4])

Indicator Terms		Negative Synsets	
error	expect	abject	unfortunate
wrong	doing	distressing	dispossessed
except	remove	pitiful	tawdry
null	fine	sad	hapless
work	syntax	sorry	miserable
something	try	bad	misfortunate
compile	empty	unfit	pathetic
argue	trouble	unsound	piteous
throw	trying	scrimy	pitiable
unfortunate	getting	shoddy	poor

1) Key Term Analysis: Serva et al. [20] propose a list of indicator terms (also called key terms) for extracting low-quality code examples from programming questions and answers of SO. The left column of Table I shows some of their indicator terms. In order to investigate if such key terms can be a distinguishing feature between high quality and low quality questions, we determine their frequencies from our sample questions. Fig. 3 shows the key term histograms for the two sets of our sampled questions. Although the diagrams show moderate frequency differences for the key terms between up-voted and down-coted questions, they are not significant. In particular, Mann-Whitney U tests confirm that the differences are not significant (i.e., $U = 127, p - value = 0.258 > 0.05$). The findings show that while key terms might be effective for negative code example extraction, they are not sufficient enough to separate low quality questions from high quality questions. Despite this, as the test is for 1000 samples and key-terms have moderate histogram differences we consider these terms for further verification in the classification model. In fact, as several studies [4, 21] suggest, human emotions are often captured in sentences or phrases rather than in isolated words.

2) Sentiment Analysis: Sentiment analysis identifies the emotions embedded within the texts. Emotions are annotated as- *positive, negative* and *neutral*. Over the past years sentiment analysis has been widely adopted, and a popular lexical resource SentiWordNet [4] is used in many research works [21]. In our exploratory study, we first make use of SentiWordNet to contrast between high quality and low quality questions from SO. We collect top-ranked negative synsets from SentiWordNet, and the right column of Table I shows the negative words [4]. We then determine the frequency of these terms in our sampled questions. Fig. 4 shows the histograms for our sampled up-

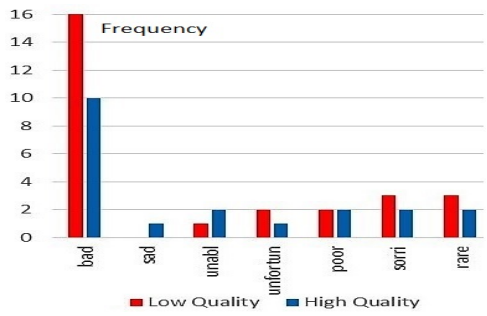

Fig. 4. Histogram of top negative terms (taken from [4]) from 2,000 sampled questions of Stack Overflow

TABLE II
POLARITY OF EXAMPLE QUESTIONS USING *Sentiment140* API

#	ID	Part of Question Text (with vote)	Polarity
1	19393251	This started happening lately, every I use a piece of code it counts [...] I have due in a month because of this **stupid problem**. {-10}	Negative
2	896532	That aside, I've managed to add Captcha to my comments, and I've learned that customizing the form is a **terrible idea** [...]. {-3}	Negative
3	8908508	[...] So I will have to separate all this in packages, [...] Is this separation a **good thing** in your opinion or is this a complete overload? {7}	Neutral
4	8908224	I do not mind keeping the executable but [...] I **would love** it if the library works just like MySQLdb does. {3}	Positive

voted and down-voted questions. It should be noted that we consider only a tiny fraction (top 10 from the origin list [4]) of the 1,105 synsets as they are well defined by Baccianella et al.[4]. However, the histogram suggests a moderate distribution for those terms. Furthermore, we note that negative words are more frequent in the down-voted (i.e., low quality) questions than the up-voted (i.e., high quality) questions. Existing studies [5, 15] are also found to be aligned with our finding. Bazelli et al. [5] analyze personality traits of the authors from SO questions, and report that the authors of high-voted artifacts (questions, comments, and answers) often express less negative emotions than the authors of down-voted artifacts.

We determine the sentiment polarity of each of the sentences from the example questions using Sentiment140 API. From Table II, we see that the API provides correct polarities for examples 1, 2 and 4. Unfortunately, the API produced Neutral polarity for sample 3, which should have been Positive. We also perform emotion extraction from the sampled 2000 questions (1000 down-voted and 1000 up-voted), and determine correlation among different attributes (Section II-C1) of the questions to better understand the impact of human emotions. From Table III, we note that positivity in the question is correlated with TF-IDF (term-frequency and inverse-document-frequency) and with overall score of the question. According to our base term analysis, high quality (i.e., up-voted) questions are richer in content, i.e., have greater TF-IDF than low quality (i.e., down-voted) questions. Thus, positive sentiment is

TABLE III
CORRELATION BETWEEN SENTIMENT POLARITY AND QUESTION QUALITY

	NP (Negative)	**PP** (Positive)	**OP** (Neutral)
TF-IDF	0.12	**0.54**	0.35
Question score	**0.08**	0.15	0.10

positively correlated with question quality as well. On the other hand, we see from Table III that negativity in the question is helpful neither for scoring votes from the crowd nor for improving the content quality of the question.

From the above empirical study, overall, we notice interesting differences between the two types of questions. First, high quality questions are more positive sentimentally than low quality questions, and this positivity also helps them score votes. Second, low quality questions are affected with negativity, and they are also poor in the textual content, i.e., less TF-IDF. Thus, to answer $\mathbf{RQ_1}$, the quality of a question is moderately affected by its author's emotions or sentiments embedded in its natural language texts.

C. The Proposed Question Model

Our exploratory study suggests that high quality and low quality questions from Stack Overflow significantly differ in terms of TF-IDF [19] and human emotions embedded within the texts. That means, such features can be exploited to separate low quality questions from high quality questions. We thus incorporate those features in a question quality model based on machine learning. In this section, we discuss the detailed design of our model, and also attempt to answer RQ_2.

1) Feature Calculation: Based on the findings from the exploratory study, we select four features for our model– (1) TF-IDF of key/indicator terms (Table I) in the question texts, (2) negative sentence count, (3) positive sentence count and (4) neutral sentence count from a question. Since term distribution differs between high quality and low quality questions according to our exploratory analysis, we determine TF-IDF of a question using the indicator list of [20]. While the TF-IDF is based on the lexical aspect of a question, the remaining three counts are based on emotional signatures within the texts.

Sentiment Polarity Counts: We determine the counts of three types of sentences from a question d, and call them *negative sentence count* (NP_d), *positive sentence count* (PP_d) and *neutral sentence count* (OP_d). In order to determine sentiment polarity, we use Sentiment140 API[1], an implementation of Go et al. [9]. We first isolate each of the sentences (S) from the question texts, determine their polarities using the API, and then accumulate them to generate the three sentence counts:

$$Negative\ Sentence\ Count, NP_d = \sum_{s \in S} I(Pol_s = N_{eg})$$

$$Positive\ Sentence\ Count, PP_d = \sum_{s \in S} I(Pol_s = P_{os})$$

$$Neutral\ Sentence\ Count, OP_d = \sum_{s \in S} I(Pol_s = O_{bj})$$

Here, Pol_s is the polarity indicator of each sentence s, and the identity function $I(Pol_s = N_{eg})$ returns 1 only when the polarity Pol_s of s is negative. N_{eg} represents negative polarity, P_{os} means positive polarity, and O_{bj} means neutral polarity. Investigation shows that the up-voted (i.e., high quality) question not only contains meaningful textual content (i.e., higher TF-IDF) but also its content is enriched with positive emotions. On the other hand, the down-voted (i.e., low quality) question not only contains less meaningful content but also the content is affected by negative human emotions.

2) Question Classification: Since our dataset contains clearly labeled question samples, we apply supervised learning to our classification task. For classification, we use two machine learning algorithms– Multilayer Perceptron (MLP) and Support Vector Machine (SVM). We train our algorithms using 38,920 question samples from the dataset, and then evaluate the classifiers using 10-fold cross-validation.

Multilayer Perceptron: Multilayer feed forward neural networks are universal approximators and are used to extract patterns or detect trends that are too complex to be noticed [12]. We develop our MLP model using one input layer containing 4 nodes, one hidden layer with 4 nodes and one output layers with 2 nodes. Nodes in the input and output layers are based on the extracted features (i.e., 4) and the labeled classes (i.e., 2). There is no strict rule for selecting the number of nodes in the hidden layer. However, as a rule[2] of thumb, we use the following formula, and choose four nodes in the hidden layer.

$$\#hidden\ nodes = (\#input\ nodes + \#output\ nodes) \times \frac{2}{3}$$

Although we explored other values, finally we set learning rate=0.1 and number of iterations=500 for the best outcome of our experiments. Since our data might not be linearly separable, we use a logistic function based activation.

Support Vector Machine: Support Vector Machine is often used for detecting binary classes from nonlinear data space. SVM defines support vectors for separating hyperplanes [7]. Since our data might not be linearly separated, we choose Gaussian Radial Basis function [7] as our kernel function for producing non-linear hyperplanes. The value of regularization parameter λ can be arbitrarily adjusted, and through iterative investigations we found that the value 0.05 produces the best result. We run our experiments on WEKA[3], and verify our experimental findings for different configurations.

III. EXPERIMENT AND DISCUSSION

We run both classifier models–MLP and SVM– on 38,920 labeled samples where each sample is represented as a vector of four numeric feature values and a class label (i.e., either '+1' or '-1'). We apply 10-fold cross-validation for testing the performance of our developed models. For evaluation and validation, we use two classical performance metrics– precision and recall. Table IV and VI summarize our experimental results using those metrics.

Table IV shows how our models perform with different sets of features considered with noisy data. We first run experiments

[1]https://cran.r-project.org/src/contrib/Archive/sentiment/

[2]http://stackoverflow.com/questions/10565868

[3]http://www.cs.waikato.ac.nz/ml/weka/

TABLE IV
EXPERIMENTAL RESULTS (WITH NOISE)

		SVM		MLP		Serva et al. [20]	
	Features	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Baseline Method	$\{S(t)\}$	–	–	–	–	61.61%	71.48%
Proposed Method	$\{TF-IDF, NP, PP, OP\}$	63.40 %	72.00%	**69.60**%	**74.20** %	–	–
	$\{TF-IDF, NP, PP\}$	63.10 %	72.30 %	69.40%	74.10%	–	–
	$\{NP, PP\}$	67.60 %	73.80%	**68.50**%	**73.80**%	–	–

TABLE V
RESULTS FOR BALANCED DATASET(WITH NOISE)

		SVM		MLP		Serva et al. [20]	
	Features	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Baseline Method	$\{S(t)\}$	–	–	–	–	66.44%	74.21%
Proposed Method	$\{TF-IDF, NP, PP, OP\}$	**82.7** %	**82.3**%	82.6%	82.1%	–	–
	$\{NP, PP, OP\}$	82.2 %	82.00%	**82.8**%	**82.6**%	–	–

TABLE VI
EXPERIMENTAL RESULTS FOR 700 RANDOM QUESTIONS (WITHOUT NOISE)

	SVM		MLP	
Features	**Precision**	**Recall**	**Precision**	**Recall**
{TF-IDF, NP, PP, OP }	**67.00** %	66.10%	**69.00** %	68.90 %
{ NP, PP }	66.50 %	65.40 %	68.50 %	68.80%

with indicator term based technique of Serva et al. on our dataset, and found 61.61% precision and 71.48% recall. We consider them as the baseline performance.

Then we perform experiments with our MLP and SVM models, and found MLP performing better than SVM. They show about 8% improvement in the precision and 3% improvement in the recall over the baseline performance. Since we are interested to investigate how emotional components (i.e., sentiments) in the texts affect the quality of a question, we filter out additional features gradually and collect the results. We first discard the neutral sentence count (OP), and notice very trivial change in the performance. This suggests that this feature has a very low predictability power for question quality. We then discard TF-IDF from our models, and notice that the performance is still almost the same. For example, with the two sentiment based features–negative sentence count (NP) and positive sentence count (PP), our MLP classifier provides a 68.50% precision and 73.80% recall which are promising. Although the accuracy is not too high, these findings clearly pose the sentiment based metrics as promising candidate features for quality prediction of Stack Overflow questions.

As our dataset contained unexpected noise that might be affecting our findings, we choose a random subset of 1,000 questions from the dataset, and discard the questions containing items other than pure texts–*equation, email address, code segment, hyper-links,* and *configuration information*. This leads us to a collection of 700 samples containing 400 up-voted questions and 300 down-voted questions, and our classifier provides 69.00% precision and 68.50% recall (Table VI) with these questions (and same configuration of the model). These are very close to our previously reported findings in Table IV.

Finally, we prepare balanced dataset (51 : 49) by separating 11,000 top up-voted questions and 10,474 down-voted questions, and employ our model on this. In this case, the outcome is significantly improved; precision rates for MLP and SVM are 82.6 % and 82.2 % respectively with the sentiment features only (Table V). Thus, to answer $\mathbf{RQ_2}$, machine learning model can effectively predict the quality and classify the questions based on sentiment metrics. Such findings also strengthen our RQ_1 that embedded human emotions in the question texts can really affect the quality of a question.

IV. RELATED WORK

Mining of programming Q & A sites containing millions of technical questions, answers and comments are becoming more and more popular among the research community. Arai and Handayani [2] present a general model to predict quality of information from Yahoo! answers by using answer features (AF) as metrics (i.e., non-textual information). They use only 815 training samples and 302 test samples. Most of the metrics they use are calculated during post-submission phase of a question, and are heavily dependent on the time spent. Ponzanelli et al. [16] use SO metrics, Readability metrics, and Popularity metrics for predicting the quality of technical questions where they employ decision tree (DT) as the predictive model. SO metrics and Popularity metrics are post-submission metrics, and are dependent on questions age. Ponzanelli et al and colleagues [17] also suggest a linear quality function (QF) to submit the low quality questions into review queue. In a recent study, Duijn et al. [8] focus on code only information for classification of posts. One of the features they selected is the length of the code segment. However, from the manual investigation of posts it is observed that there exist numerous code segments which are positive but have fewer lines of code, even in some cases one line of code. Moreover, many questions do not contain code-segment, thus the quality is only dependent on the text of those questions.

Several studies are available to detect human emotions [1, 9, 21] in the informal texts from social networking website. However, corpora of Q & A sites mainly contain unstructured technical texts and mining them is challenging due to emoticons, slang, misspelled words, hash-tags, links, e-mail, equation, and even code-segments within a sentence [15]. Recently,

Novielli et al. [15] conduct an empirical study on 400 top scored SO posts (questions, answers, and comments) using SentiStrength tool, and identify several challenges of sentiment analysis in technology domain. Despite limitations of affect (i.e., human emotion) extraction, several researchers focus on software developers emotions and behavior in software artifacts [5, 11, 13]. Murgia et al. [13] investigate emotional information of the developers in the software development. They analyze issue reports from issue trackers (Apache Jira), and categorize the found emotions into six groups. Bazelli et al. [5] study SO questions and answers to explore the personality traits of the authors, and report that authors of high-voted posts reveal notably less negative emotions than the authors of down-voted posts. Serva et al. [20] investigate 240 posts, and extract key-terms having negative impact on question quality. Although, these terms are claimed to be negative, our study suggests their little power in distinguishing lower quality questions from higher quality questions (Section II-B). In a related work, Novielli et al. [14] argue about the impact of emotions upon the quality of SO posts. However, their exists no implementation of their work yet. In addition, there are other sentiment extraction related studies [10, 22] which motivate us to dig down more deeply about the problem. To summarize, most of the SO question's quality analysis use features (SO metrics and popularity metrics) which are based on the age of the questions along with textual metrics whereas our proposed models do not depend on them. We consider the sentiment polarity of each of the sentences from the question text as features for designing the model. Thus, our proposed approach is more feasible for quality prediction during question posting. Moreover, our experiment is also a feasibility evaluation of Sentiment140 API to the detection of human emotions from technical discussion text.

V. Threats to Validity

Although we conduct our study and analyze our results carefully, there exist several threats to the validity of our findings. First, programming questions contain various items beyond natural language texts such as equations, emails, symbols, hyper-links and code-segments which can pose as noise, and the reported results might be affected. In order to handle that threat, we experiment with 700 randomly selected questions after removing noise, and collect the results. Interestingly, we note that the results do not vary significantly from our original performance.

Second, our dataset is inherently skewed since there is a bit imbalance in the categories of questions (73% up-voted vs 27% down-voted). We did our experiment with what we get randomly. Thus the classifiers also do not perform equally for both types of questions. However, experiment with balanced dataset (51% up-voted and 49% down-voted), shows better performance (Table V) for our model, and thus that was not really a threat.

Third, we consider positive-scored questions as of high quality and negative-scored questions as of low quality as suggested by relevant literature [2, 8, 16, 20]. That means, we exploit crowd-sourced knowledge to develop the oracle for our classifiers, and still we cannot guarantee that all positive-scored questions contain high quality content.

VI. Conclusion and Future Work

Programming questions in Stack Overflow are rapidly increasing, and maintaining their content quality is a major concern. In this paper, we report an exploratory study and experimental outcome of a proposed model using 38,920 questions from Stack Overflow where we investigate the impact of author's emotions on the quality of a question. We select four features including three sentiment based features for separating low quality questions from high quality questions, and perform the classification using two popular machine learning algorithms–MLP and SVM. Our best model, MLP provides a precision of 70% and a recall of 74% which are promising according to relevant literature. We also clearly demonstrate that human emotions embedded in the question texts have significant impact on the quality of the questions. In future, we plan to explore the scope of sentiment aspects in various classification, filtration and management of StackOverflow questions.

References

[1] A. Agarwal, B. Xie, I. Vovsha, O. Rambow, and R. Passonneau. Sentiment analysis of twitter data. In *Proc. LSM*, pages 30–38, 2011.

[2] K. Arai and A. N. Handayani. Predicting quality of answer in collaborative q and a community. *Intl. J. of Adv. Research in AI*, pages 21–25, 2013.

[3] M. Asaduzzaman, A. S. Mashiyat, C. K. Roy, and K. A. Schneider. Answering questions about unanswered questions of stack overflow. In *Proc. MSR*, pages 97–100, 2013.

[4] S. Baccianella, A. Esuli, and F. Sebastiani. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proc. LREC*, pages 2200–2204, 2010.

[5] B. Bazelli, A. Hindle, and E. Stroulia. On the personality traits of stackoverflow users. In *Proc. ICSM*, pages 460–463, 2013.

[6] D. Correa and A. Sureka. Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow. In *Proc. WWW*, pages 631–642, 2014.

[7] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, pages 273 – 297, 1995.

[8] M. Duijn, A. Kucera, and A. Bacchelli. Quality questions need quality code: classifying code fragments on stackoverflow. In *Proc. MSR*, pages 410–413, 2015.

[9] A. Go, R. Bhayani, and L. Huang. *Twitter sentiment classification using distant supervision*. Stanford University, Tech. Rep, 2009.

[10] E. Guzman, D. Azócar, and Y. Li. Sentiment analysis of commit comments in github: An empirical study. In *Proc. MSR*, pages 352–355, 2014.

[11] E. Guzman and B. Bruegge. Towards emotional awareness in software development teams. In *Proc. FSE*, pages 671–674, 2013.

[12] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Netw.*, pages 359–366, 1989.

[13] A. Murgia, P. Tourani, B. Adams, and M. Ortu. Do developers feel emotions? an exploratory analysis of emotions in software artifacts. In *Proc. MSR*, pages 262–271, 2014.

[14] N. Novielli, F. Calefato, and F. Lanubile. Towards discovering the role of emotions in stackoverflow. In *Proc. SSE*, pages 33–36, 2014.

[15] N. Novielli, F. Calefato, and F. Lanubile. The challenges of sentiment detection in the social programmer ecosystem. In *Proc. SSE*, pages 33–40, 2015.

[16] L. Ponzanelli, A. Mocci, A. Bacchelli, and M. Lanza. Understanding and classifying the quality of technical forum. In *Proc. QSIC*, pages 343–352, 2014.

[17] L. Ponzanelli, A. Mocci, A. Bacchelli, M. Lanza, and D. Fullerton. Improving low quality stack overflow post detection. In *Proc. ICSME*, pages 541–544, 2014.

[18] M. M. Rahman and C. K. Roy. An insight into the unresolved questions at stack overflow. In *Proc. MSR*, pages 426–429, 2015.

[19] S. Robertson. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60, 2004.

[20] R. Serva, Z. Senzer, L. Pollock, and k Vijay-Shanker. Automatically mining negative code examples from software developer Q & A forums. In *Proc. SoftMine ASE*, pages 410–413, 2015.

[21] C. Strapparava and R. Mihalcea. Learning to identify emotions in text. In *Proc. ACM Applied computing*, pages 1556–1560, 2008.

[22] Y. Zhang and D. Hou. Extracting problematic api features from forum discussions. In *Proc. ICPC*, pages 142 – 151, 2013.