# Systematic Exploration and Classification of Useful Comments in Stack Overflow

Prasadhi Ranasinghe, Nipuni Chandimali, Chaman Wijesiriwardana
Faculty of Information Technology
University of Moratuwa
Katubedda, Sri Lanka

***Abstract*—Stack Overflow is a public platform for developers to share their knowledge on programming with an engaged community. Crowdsourced programming knowledge is not only generated through questions and answers but also through comments which are commonly known as developer discussions. Despite the availability of standard commenting guidelines on Stack Overflow, some users tend to post comments not adhering to those guidelines. This practice affects the quality of the developer discussion, thus adversely affecting the knowledge-sharing process. Literature reveals that analyzing the comments could facilitate the process of learning and knowledge sharing. Therefore, this study intends to extract and classify useful comments into three categories: request clarification, constructive criticism, and relevant information. In this study, the classification of useful comments was performed using the Support Vector Machine (SVM) algorithm with five different kernels. Feature engineering was conducted to identify the possibility of concatenating ten external features with textual features. During the feature evaluation, it was identified that only TF-IDF and N-grams scores help classify useful comments. The evaluation results confirm Radial Basis Function (RBF) kernel of the SVM classification algorithm performs best in classifying useful comments in Stack Overflow regardless of the usage of the optimal combinations of hyperparameters.**

***Keywords*—Stack overflow; useful comments; machine learning; SVM; classification**

## I. Introduction

The software engineering community considers Stack Overflow as a learning site and a learning community for software developers and practitioners [1], [2]. Comments in Stack Overflow (SO) are temporary *"Post-It"* notes relevant to a particular question or an answer which has already been posted [3]. They clarify and enrich the content conveyed through questions and answers. Examining comments is particularly beneficial because they go beyond the questions and answers to facilitate the process of learning and knowledge construction. For example, the importance of analyzing Stack Overflow comments for the recommendation of source code fragments has been extensively discussed in the literature [4], [5], [6].

In Stack Overflow, there is a standardized method to add comments which are suggested in the standard commenting guidelines of Stack Overflow. *Request clarification*, *constructive criticism*, and *relevant but transient information* in the comments are being encouraged in commenting guidelines in Stack Overflow [7]. Comments related to *request clarification* contain requests for extra information for better understanding the post. *Constructive criticism* comments point out flaws, obsolescence, and coding errors thus encouraging the author to improve it. Comments related to *relevant but transient information* guide the users in retrieving more information that is relevant to a certain post in Stack Overflow [7]. Nevertheless, Stack Overflow does not recommend comments related to suggesting corrections, compliments, answering a question, criticisms, secondary discussions, and discussions of site policies and community behavior to be posted on the site[1]. However, it is observed that most of the developers tend to respond to posts with comments which are not adhering to Stack Overflow's guidelines on comments [8]. Hence, useful comments get ignored by the authors of relevant posts as well as the members of the community. According to previous studies, 27.5% of the comments which required an update from the author were ignored [9]. Therefore, comments in Stack Overflow should be studied in depth to identify whether users post comments by adhering to the commenting guidelines of Stack Overflow.

It is believed that systematic categorization of such comments could provide valuable insights to software practitioners when using Stack Overflow as a learning source. Thus, there is a need for data-driven solutions to retrieve useful comments and categorize them. However, this direction has not been thoroughly investigated in the literature [7], [10], [11]. Therefore, this study exploits machine learning and natural language processing to automatically classify the comments in Stack Overflow that follow the standard commenting guidelines based on three types: request clarification, constructive criticism, and relevant information. This paper expects to address the following research question (RQ):

**RQ :** How to correctly classify the useful comments in Stack Overflow into standard comment categories: request clarification, constructive criticism, and relevant information?

The remainder of this paper is organized as follows. Next, we present the related work of this study. Then the methodology of classifying useful comments in Stack Overflow is presented. Next, the results and evaluation are described. Finally, the conclusion of the study and the further work are being discussed.

## II. Related Work

During the past years, researchers have involved themselves in various studies related to comments and crowd source knowledge in Stack Overflow. The author in [7] analyzed commenting activities in respect of timing, content and individuals who perform commenting focusing on the comments

[1] https://stackoverflow.com/help/privileges/comment

which were posted in answers in Stack Overflow. In their study, comment classification was done by a light-weight open coding process in which authors were involved in deriving a draft list of comment types. The time in which users take to post comments once the answer was posted was taken into consideration. They mention the need of a methodical and organized study related to the comments in Stack Overflow to better understand how comments are being used. Moreover, improving the current commenting system was stated as necessary since users post comments in unrecommended manners. Furthermore, they state the possibility of future research to leverage approaches from Machine Learning and NLP communities to automatically identify such comment categories [7].

In their research, Sengupta and Haythornthwaite performed a qualitative analysis for the purpose of introducing a coding schema which comprises nine comment categories through classification. The purpose of their study was the provision of insights into commenting in Stack Overflow. They mention the requirement of further research on comment usage [1]. Identification of inadequate comments in Wikipedia's talk page edits and classification of the same into different categories was performed by Sulke and Varude. In this analysis, SVM provided the best results out of the utilized classification algorithms [12]. Contextual tagging mechanism was utilized to classify posts in Stack Overflow in the exploration done by Chimalakonda et.al. SVM promised the highest accuracy of 78.5% in this approach. In this analysis, Limited number of posts were examined during the study. Furthermore, Some of the statistical distributions were not balanced and they were biased towards one certain topic [13].

Beyer et.al in their study automated the classification of Stack Overflow's posts into seven categories of questions. Manual Analysis of phrases was performed to find patterns and training classification models based on Machine Learning Algorithms was done in their analysis. Since manual analysis was performed there exists a possibility for this categorization of the posts to be biased [14]. Saif et.al performed online toxic comment classification by making use of three Artificial Neural Network Approaches and Logistic Regression [15]. Quantitative as well as a qualitative analysis related to the obsolete answers of Stack Overflow was conducted in an Empirical study related to the obsolete knowledge on Stack Overflow. The utilized heuristics based approach contained the accuracy of 75%. In this analysis, it was believed that Machine Learning could provide better results [7]. In a study related to the prediction of who will answer a specific question in Stack Overflow, a hybrid approach which amalgamates both knowledge from the question and the asker to retrieve more error-free candidate lists was considered as essential [16]. In a mining approach which suggests insightful comments in Stack Overflow, recommendation of source code comments was accomplished with the accuracy and precision of 80%. In this approach, Dataset which was considered for the empirical evaluation was limited. Furthermore, it was stated that the approach might be inadequate in recommending comments for the code segment from proprietary or legacy projects [4]. The value of comments revolves around many important aspects such as improving the source code, analyzing the code further, and facilitating code reuse [4] [17] [18] [19]. During the Exploration of the means of which comments have an effect on answer updates, comments and answer updates which involve code segments were only being taken into consideration. Moreover, it was mentioned that there exists a tendency for the comments to be mislabeled if the code element in the comment is not correctly identified by the system, which leads to false positives [9]. A Gold Standard for Emotion Annotation in Stack Overflow was introduced by Novielli et.al. In this exploration, manual annotation of Stack Overflow gold standard data set with emotion labels was performed. Identification of Emotions was based on clear guidelines of a conceptual framework which is based on theory. Moreover, Final gold labels were assigned through agreement of the majority of 3 coders [20].

Automatic comment generation approach which mines comments from Q&A sites includes code description mapping extraction, refinement of description, code clone detection, code clone pruning and selection of comments. Failure at identifying comments that comprise an incorrect description of the code segment was stated as a limitation in this study [21]. In the study that extracted candidate method documentation from discussions of Stack Overflow, JavaDoc descriptions were created and the mining of source code descriptions from developers' discussions was recognized as an aspect which needed more improvement when regarding its usability [22]. Wikipedia's comment dataset by Jigsaw was used in analysing any section of text and detecting distinct types of toxicity by Chakrabarty. In this analysis, TF-IDF with 6 headed Machine Learning promised the highest accuracy which is 98.98%. Chakrabarty states that the utilization of Grid search Algorithm can obtain more accurate results [23].

## III. Methodology

Useful comments in Stack Overflow can be defined as the comments, which adheres to Stack Overflow's commenting guidelines. The facilitation of the process of learning and knowledge construction could be improved by analyzing useful comments. Classification is an approach for analyzing useful comments. This study aims at extracting useful comments and classifying them into standard useful comment categories based on their features. In this study, Data Extraction, Qualitative Analysis and Review, Feature Engineering and Preprocessing, Feature Extraction, Training and Evaluation of the classification model were carried out. Fig. 1 represents the high-level architectural diagram of exploration of useful comments in Stack Overflow.

### A. Data Extraction

This step was necessary to obtain the comments data which is needed to perform the study. In this study comments posted in the Stack Exchange Data Explorer during the past 5 years (i.e., between 1st of January 2015 and 08th of November 2020 were taken into consideration).

Listing 1: Query utilized in obtaining comments data from Stack Exchange Data Explorer

```
SELECT Comments.Id, Comments.PostId, Comments.Score,
       Comments.Text, Comments.CreationDate,
       Comments.UserId, Comments.ContentLicense,
       Posts.Tags
FROM   Comments
```

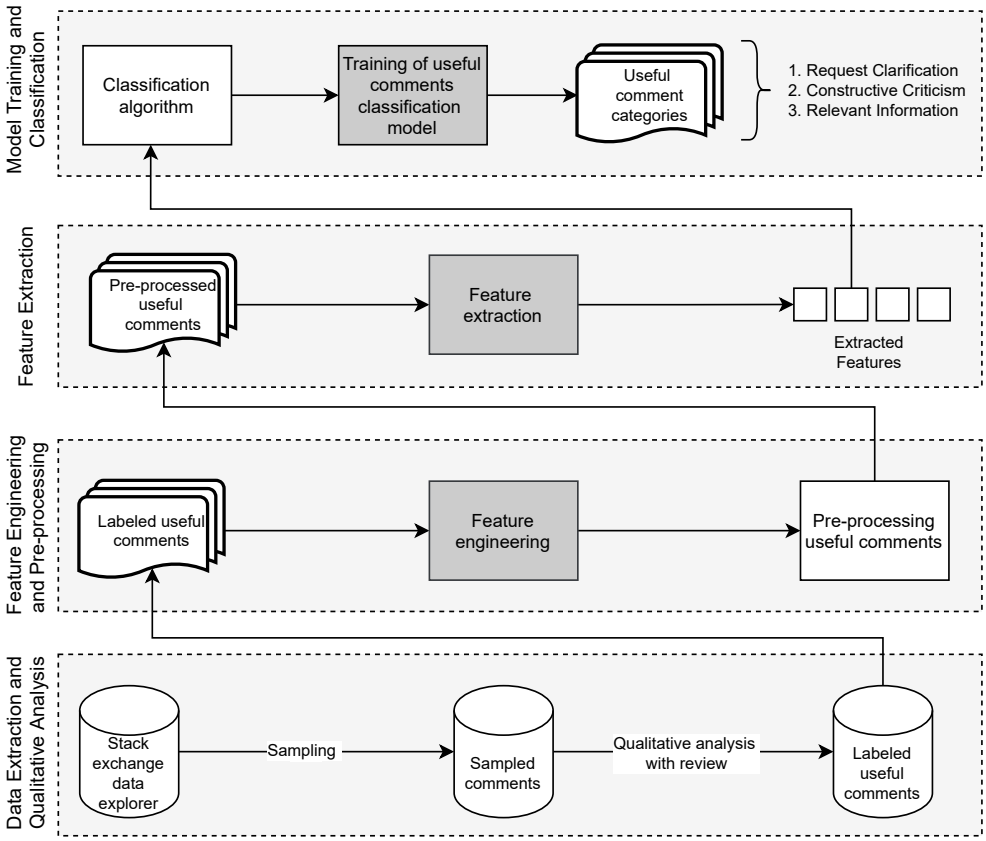


Fig. 1. High Level Architecture Diagram of Exploration of Useful Comments in Stack Overflow

```
INNER JOIN Posts ON Comments.PostId = Posts.Id
WHERE  Comments.Score > 0 AND Comments.CreationDate
   <'2020-11-08 00:00:00' AND Comments.CreationDate>
        '2015-01-01 00:00:00' AND Posts.Tags !=' '
ORDER BY Score Desc
```

The Stack Exchange Data Explorer was queried to fetch the comments which were made in response to questions and answers, that contain tags and with a score which is greater than 0. Comments with the comment score which is greater than 0 was considered as a measure of importance of comments. The query utilized in obtaining the data is presented in Listing 1.

As a result of querying, 50000 comments in Stack Overflow were obtained. From the obtained 50000 comments, 6164 comments were sampled using the simple random sampling technique which is a probability sampling technique that utilized randomization. This technique was used as there was no prior knowledge related to the comments data and therefore each element contained an equal chance of getting selected for the purpose of being a part of the sample[2].

### *B. Qualitative Analysis and Review*

The objective of Qualitative Analysis was to create the dataset that is required to train the machine learning model. In Qualitative Analysis, the unclassified comments data was labeled to filter out the useful comments. A total of 6164 comments which were the output from data extraction were taken for the Qualitative Analysis. As the first step of Qualitative Analysis, deductive coding was used to label unclassified data [24]. In deductive coding a predefined set of label values were used to assign label values to unclassified comments. If a given comment was assigned multiple labels, such comments were removed from the dataset. As the output of deductive coding 3587 useful comments and 2577 not-useful comments were identified. Then as the next step of Qualitative Analysis, useful comments were filtered out and categorized into the three categories mentioned in Table I.

Table II discusses the measures used to categorize and label the extracted useful comments into standard comment categories during the qualitative analysis. After completing the qualitative analysis it was necessary to review the labeling process of the comments in order to measure the consistency, quality and accuracy of the labeled data. The review was performed by reviewing all the labels of the 3587 useful comments. The intra rater reliability percentage was calculated because both the data labeling and review was performed by a single annotator. Therefore, as a result of the review 3120 comments were properly labeled with the intra rater reliability of 86.98% with regard to correct labeling of comments in both occasions. Therefore 467 comments were disregarded in the study further as they were identified as misclassified in the review. 3120 useful comments were identified from the review process and those comments were used in the implementation

[2]https://towardsdatascience.com/sampling-techniques-a4e34111d808

TABLE I. QUALITATIVE ANALYSIS RELATED TO USEFUL COMMENTS IN STACK OVERFLOW FOR LABELING OF USEFUL COMMENTS

| Useful Comment Category | Property Description |
|---|---|
| Request Clarification (Zhang et al., 2019a) (Requesting clarification from the author) | Requesting provision of more information or Expression of lack of understanding. These comments can be identified with the keywords such as 'please clarify', 'please elaborate', 'how', 'what' etc. |
| Constructive Criticism (Zhang et al., 2019a) (Guiding the author in improving the post) | Contains both positive and negative comments that are stated in a pleasant manner. Areas of improvement of posts are stated. Formatting and indentation issues are included more often. |
| Relevant Information (Zhang et al., 2019a) (Relevant but minor or transient information) | These comments may include a link to a related post, a link that redirects to other websites. Statements about question updates and answer updates were rarely found. |

of the classification model. Out of the 3120 useful comments 1010 comments were of Constructive Criticism, 1047 comments were of Relevant Information and 1063 comments were of Request Clarification.

### C. Feature Engineering and Pre-Processing

Features are independent individual variables that act as an input to a machine learning model. Machine Learning models use features for making predictions. Therefore, feature engineering was performed which included feature creation and evaluation of the created features through histogram plots. Feature creation was needed to construct new features from the 3120 useful comments identified after the review process. Moreover, feature engineering was important to identify whether external features other than the textual features can be used in building the classification model. Textual features include text scores. II presents the new features which were created and evaluated. Overlapping of data was clearly identified through horizontal scaling of histograms plotted for all comment categories separately and as a whole.

TABLE II. MEASURES USED FOR THE CATEGORIZATION

| Feature | Description |
|---|---|
| Comment Length | Total number of characters in the useful comments |
| Comment Score | The number of upvotes a specific useful comment obtained in Stack Overflow |
| Punctuation Percentage | Percentage of the total number of punctuation used in a specific useful comment |
| Average Word Count | The mean word count of a specific useful comment |
| Capitalization Usage | The total number of capital letters used in a specific useful comment |
| Stop Words Count | The total number of stop words used in a specific useful comment |
| Positive Sentiment Score | The probability of the sentiment of a specific useful comment to be positive |
| Negative Sentiment Score | The probability of the sentiment of a specific useful comment to be negative |
| Neutral Sentiment Score | The probability of the sentiment of a specific useful comment to be neutral |
| Normalized Compound Score | The sum of negative sentiment score, positive sentiment score neutral sentiment score of a specific useful comment which is then normalized and ranges between -1 and +1 |

NLTK's VADER which is a parsimonious rule based model and was used in calculating the Sentiment Score of the comments data. VADER calculates the sentiment score of a text in terms of positive sentiment score, negative sentiment score, neutral sentiment score and normalized compound score. Fig. 2 shows the feature evaluation with horizontal scaling comment length and comment score of useful comments. Fig. 3 depicts the feature evaluation with horizontal scaling for punctuation percentage and average word count of useful comments. Fig. 4 depicts the feature evaluation with horizontal scaling for the count of capital letters and stop words of useful comments. Fig. 5 shows the feature evaluation for positive sentiment score and negative sentiment score of useful comments. Fig. 6 depicts the feature evaluation for horizontal scaling for count of capital letters of useful comments. Fig. 7 depicts the feature evaluation for horizontal scaling for count of stop words. Fig. 8 and Fig. 9 presents the feature evaluation for positive sentiment score and negative sentiment score respectively. Fig. 10 shows the feature evaluation for neutral sentiment score and Fig. 11 shows the feature evaluation for compound score of useful comments. Since majority overlapping areas were obtained in 2 or all 3 classes, the features created in the feature engineering process were disregarded and not utilized in building the classification model.

Preprocessing was done to remove the noisy data that might be present in the identified useful comments. During the preprocessing stage the identified useful comments data was preprocessed using Natural Language Processing Techniques. In preprocessing lowercasing the data, replacing URLs with a keyword , punctuation removal, stop words removal, tokenization of data, stemming and lemmatization, removal of numbers, removal of emojis and emoticons in comments, and handling of chat words were carried out. Comments that will be classified as Relevant Information contain URLs. So, it was necessary to capture the URLs and hence replacement of URLs with the keyword 'link' was done without the removal of URLs. As useful comments are also developer discussions, they contained emojis and emoticons in them and it was necessary to remove them in the preprocessing stage. It was also required to replace the chat words with their meaningful phrases. This was done by maintaining a list of chat words and their meaningful phrases as key value pairs in a text file. This chat words document included chat words derived from the glossary dictionary of Stack Exchange and some common chat words observed during the qualitative analysis.

### D. Feature Extraction

Feature extraction was done by using the preprocessed comment data. Feature extraction was performed to extract the textual features from the useful comments. In this step comment text i.e. words of each comment were taken as the

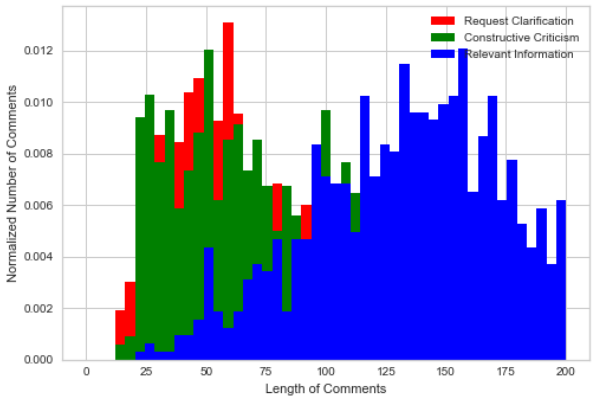


Fig. 2. Feature Evaluation with Horizontal Scaling for Useful Comment Length

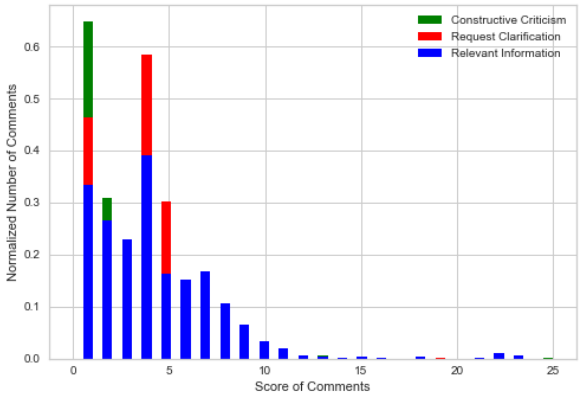


Fig. 3. Feature Evaluation with Horizontal Scaling for Useful Comment Score

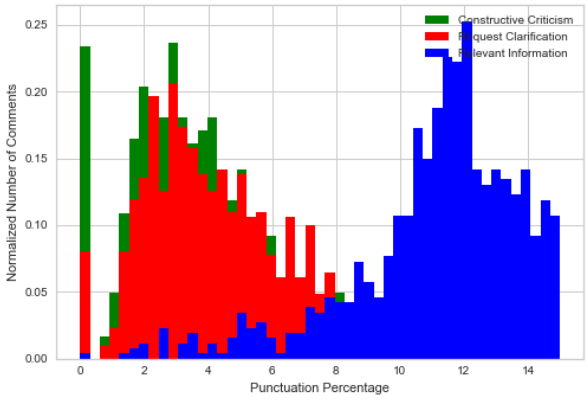


Fig. 4. Feature Evaluation with Horizontal Scaling for Punctuation Percentage of Useful Comments

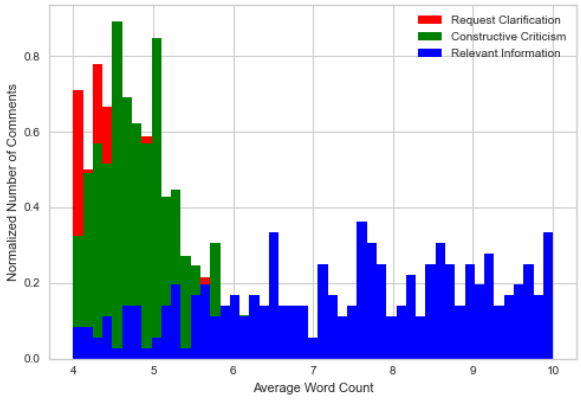


Fig. 5. Feature Evaluation with Horizontal Scaling for Average Word Count of Useful Comments

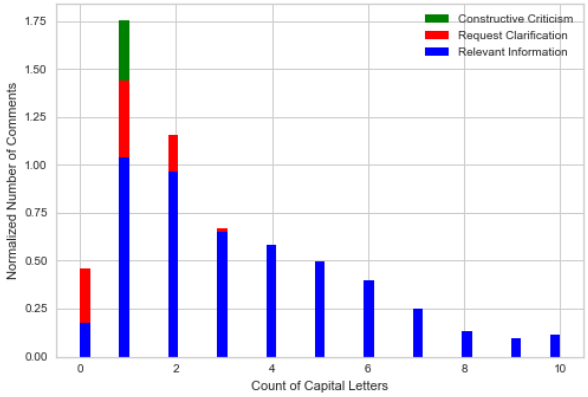


Fig. 6. Feature Evaluation with Horizontal Scaling for Count of Capital Letters of Useful Comments

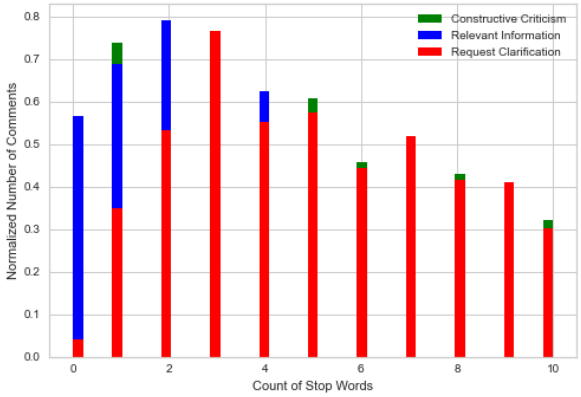


Fig. 7. Feature Evaluation with Horizontal Scaling for Count of Stop Words of Useful Comments

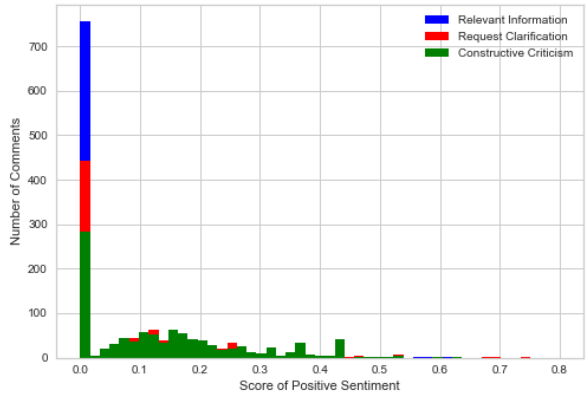


Fig. 8. Feature Evaluation for Positive Sentiment Score of Useful Comments

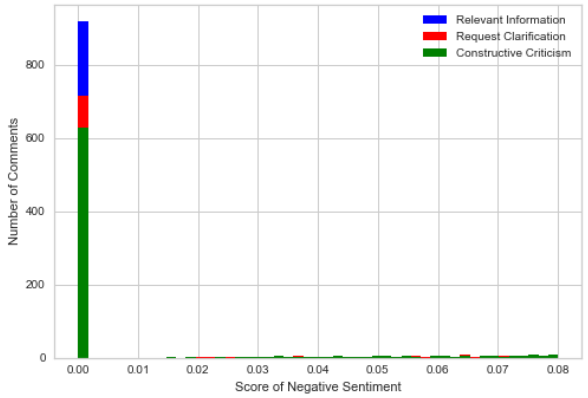


Fig. 9. Feature Evaluation for Negative Sentiment Score of Useful Comments

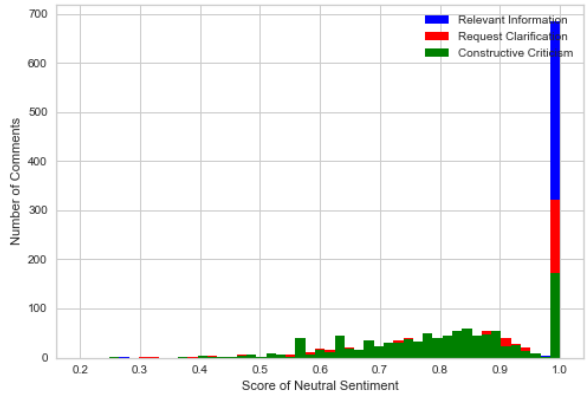


Fig. 10. Feature Evaluation for Neutral Sentiment Score of Useful Comments

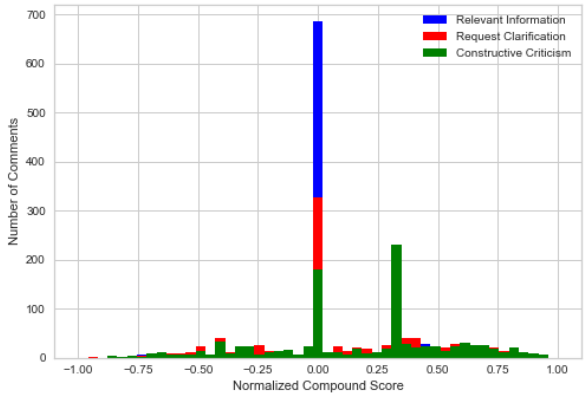


Fig. 11. Feature Evaluation for Compound Score of Useful Comments

features since the numeric features gained through feature engineering were exposed to overlapping of data in several comment categories. The TF-IDF feature extraction technique (Sulke & Varude, 2019) was utilized for the text feature extraction process along with N-grams. TF-IDF calculates the term frequency and inverse document frequency [25] [26]. IDF suppresses the effect of words which occurs in all 3 comment categories. Moreover, the TF-IDF vector looks the same as the word vector. This mechanism is used to find the meaning of sentences and it cancels out the incapableness of the bag of words feature extraction technique. TF answers how many times a particular word is used in the entire document. IDF calculates the importance of a certain term in a list of documents. For the feature extraction purpose TF-IDF Vectorizer was used as it performs the task of count vectorizer which is followed by TF-IDF transformer. Unigrams and Bigrams were utilized as N-grams. N-grams were used to boost the accuracy of the classification model. The N-gram frequency method provided an inexpensive and highly effective method of classifying documents. Encoding the categorical labels was done using the LabelEncoder before extracting features of useful comments. TF-IDF and N-grams scores were the extracted textual features. Shuffling of data was important to avoid the biases of data location within the data set. After the completion of feature extraction, sparse matrices with 275 features were obtained.

### *E. Training and Evaluating the Model*

For the classification of useful comments, a multi-class SVM classifier was designed as the classification model. The features extracted in the previous step were fed into the designed classification model for the training purpose. 80% of data in the dataset was utilized for training the model and 20% was utilized for testing the model. The initial SVM classifier model was trained without any parameters. In this initial model a RBF kernel was used by default since the kernel was not specified.

**Hyperparameter Tuning:** Hyperparameters control the behaviour of the overall machine learning model. Therefore hyperparameter tuning was considered as a necessary step. The ultimate goal was to discover the optimal combination of hyperparameters of the SVM Model that minimizes the loss and maximizes the overall accuracy of the Model. Since certain

hyperparameter combinations are not supported with specific kernels in SVM, hyperparameters tuning was done separately for 5 SVM kernels which include Linear Kernel, RBF kernel, Polynomial kernel, Sigmoid kernel and Precomputed kernel. The objective was to identify the best Kernel with best hyperparameter combinations. For the Hyperparameter tuning sklearn's GridSearchCV was used. Since Cross Validation was done with GridSearchCV to obtain the best combination of hyperparameters, a validation dataset was not needed as the cross validation divided the training data set into k number of folds , and k-1 folds were used for the training purpose and the remaining fold was utilized as the validation set. In this hyperparameter tuning, cross validation with 3 folds was performed. C, kernel, degree, gamma, decision_function_shape, coefO were the utilized hyperparameters in hyperparameter tuning. Table III contains the optimal combination of hyperparameters obtained for each kernel. In SVM, gamma is the kernel coefficient, degree is the degree of the polynomial kernel function and coef0 is an independent term. When x,y are the data to be classified, the utilized kernels are as follows.

TABLE III. OPTIMAL COMBINATION OF HYPERPARAMETERS FOR EACH SVM KERNEL

| SVM Kernel | Hyperparameters | | | | | |
|---|---|---|---|---|---|---|
| | C | kernel | degree | gamma | decision_ function_ shape | coef0 |
| Linear | 1 | linear | - | 1 | ovo | - |
| RBF | 2 | rbf | - | scale | ovo | - |
| Polynomial | 1 | poly | 1 | scale | ovo | 0.01 |
| Sigmoid | 10 | sigmoid | - | 0.1 | ovo | 0.01 |
| Precomputed | 1 | precomputed | - | 1 | ovo | - |

**Linear Kernel:** Mostly Used when a large number of data is available and when the data is linearly separable. It is the simplest Kernel Function in SVM. The Linear Kernel function (LK(x,y)) is denoted by the equation 1.

$$LK(x, y) = SUM(x.y) \tag{1}$$

**RBF Kernel:** It is usually used in classifying non-linear data. Proper separation of data when there is no prior knowledge about the data is performed successfully. The formula of RBF(RBFK(x,y)) is denoted by equation 2. Note that gamma varies between 0 and 1.

$$RBFK(x, y) = exp(-gamma||x - y||^2) \tag{2}$$

**Polinomial Kernel:** This kernel is a generalized representation of the Linear kernel. The formula of the polynomial kernel (PK(x,y)) is denoted by equation 3.

$$PK(x, y) = (gamma < x, y > +coefO)^{degree} \tag{3}$$

**Sigmoid Kernel:** This is often known as hyperbolic tangent or multilayer perceptron. This kernel is mostly preferred in Neural Networks. The formula of the sigmoid kernel (SK(x,y)) is as denoted by equation (4).

$$SK(x, y) = tanh(gamma < x, y > +coefO) \tag{4}$$

One-Vs-One(OVO) decomposition strategy was used in training the SVM model as it results in higher performance when compared with Non-OVO approaches, disregarding the overlapping level. OVO benefits the multi-class classification while increasing the separability of classes. Moreover, it was identified as an approach that highly benefits SVM as it provides robust results and superior performance [27]. Building and training of the SVM Models for linear, rbf, polynomial, sigmoid and precomputed kernels was done using the best combination of hyperparameters obtained through hyperparameter tuning. Identification of the best SVM Kernel that fits the problem context was done through evaluation measures such as the Holdout Method, Confusion Matrix and Classification Report.

## IV. RESULTS AND EVALUATION

As the exploration of useful comments in Stack Overflow was based on building a classification model which categorizes useful comments with their respective useful comment Category, the evaluation of this model was carried out to evaluate classification accuracy, precision, recall, f1-score and the number of correctly classified instances of each class using the Holdout method, Confusion Matrix and the Classification Report. The test set contained 624 data entries. Before evaluation it was necessary at first to gain insight of the instances belonging to each category in test data. Therefore, data count of each comment category in the test set was obtained. Test set contained 202 instances of Request Clarification comments, 218 instances of Relevant Information comments and 204 instances of Constructive Criticism comments.

### A. Evaluation of Classification Accuracy and F1-Score

In the Holdout Method the data set was divided into two parts, such as train set and test set. Train set contained 80% of the data while the Test set contained 20% of the data. The Train set was utilized to train the data and the Test set was utilized to test the predictive power of the implemented classification model. Classification Accuracy and F1-Score was gained as the metric of evaluation in the Holdout method. Holdout method based evaluation was performed to the initial SVM model which was built without any parameters or hyperparameters and also to the SVM models built and trained with distinct kernels and the optimal combinations of hyperparameters. Table IV contains the summary of the results obtained through the holdout method for the SVM models. According to the results obtained through the holdout method, the SVM model with the RBF kernel can be identified as the best classification model built and trained with optimal combination of hyperparameters as it promised the highest Accuracy of 87.02 and highest F1-Score of 87.11.

According to the results obtained through the holdout method, initial SVM Model and the SVM Model with RBF kernel promised similar accuracies which is 87.02. The initial SVM model promised the highest F1-score which is 87.21 when compared with all the other SVM models.

### B. Evaluating the Number of Correctly Classified Instances of Each Class

For the purpose of gaining insights of the performance of the classification model the confusion matrix can be used. The

TABLE IV. SUMMARY OF THE RESULTS OF HOLDOUT METHOD FOR SVM MODELS BUILT WITH THE BEST COMBINATIONS OF HYPERPARAMETERS

| Evaluation Metrics | Initial SVM Model | SVM Models with optimal combinations of hyperparameters | | | | |
|---|---|---|---|---|---|---|
| | | Linear SVM Model | RBF SVM Model | Polynomial SVM Model | Sigmoid SVM Model | Precomputed SVM Model |
| Accuracy | 87.02 | 85.58 | 87.02 | 85.74 | 85.42 | 85.58 |
| F1-Score | 87.21 | 85.77 | 87.11 | 85.92 | 85.61 | 85.77 |

information gained from the confusion matrix can be used to determine the usefulness of the classification model. As a result important metrics such as accuracy, precision and recall can be determined. In this study, since useful comments fall into three categories the 3x3 Confusion Matrix was used for evaluation. In the 3x3 Confusion Matrix diagonal values were identified as correctly classified and non-diagonal values were identified as misclassified. The confusion matrix was drawn for the initial SVM Model which was built before hyperparameter tuning. Afterwards, the confusion matrices were plotted for each SVM Model with distinct kernels and optimal combination of hyperparameters obtained after hyperparameter tuning. The results obtained through the Confusion Matrices relevant to each SVM Model are summarized in Table V.

### C. Evaluating the Quality of Predictions

The quality of the predictions relevant to a classification algorithm is measured by the Classification Report. It provides the main classification metrics based on each class. True Positives, True Negatives, False Positives and False Negatives are utilized for the purpose of predicting metrics in a Classification Report. Along with the Classification Report metrics relevant to macro average, micro average and weighted average were calculated for precision, recall and f1-score. Initially, the Classification Report was obtained for the initial SVM Model which was built without including any parameters. Afterwards, the Classification Reports were gained for each SVM Model with distinct kernels and optimal combination of hyperparameters. Table VI contains the summary of the results obtained from the Classification Report.

According to the evaluation mechanisms the initial SVM model which uses the RBF kernel in default has the highest weighted precision of 0.88 when compared to the rest of the SVM models. Among the SVM models trained with optimal combinations of hyperparameters, SVM model with the RBF kernel is the best kernel which classifies useful comments in stack overflow as it promised the highest accuracy, weighted average for recall and f1-score. Thus, research question is successfully addressed.

## V. CONCLUSION

This paper presents a machine learning-based novel approach to explore Stack Overflow comments by classifying them into the respective standard comment categories. The main stages of this study consist of Data Extraction, Qualitative Analysis, Feature Engineering and Preprocessing, Feature Extraction, Training, and Evaluation of the classification model. As per the results of feature engineering, it was observed that none of the ten external features used can be combined with the textual features to implement the classification model. The evaluation was conducted to analyze the classification accuracy, precision, recall, f1-score, and the instances of each class that have been correctly classified by using the Holdout Method, Confusion Matrix, and Classification Report. RBF was identified as the best Kernel in exploring useful comments in Stack Overflow regardless of the use of hyperparameters while training the classification model. The results of this study can be utilized in the long process of Stack Overflow's useful comment analysis approaches for improving the facilitation of the process of learning and knowledge construction.

Future research may leverage the Multi-label classification approach to identify and classify the useful comments which belong to multiple comment categories.

## ACKNOWLEDGMENT

The authors gratefully acknowledge the financial support provided by the University of Moratuwa SRC Long-term Grant (Grant no: SRC/LT/2021/02).

## REFERENCES

[1] S. Sengupta and C. Haythornthwaite, "Learning with comments: An analysis of comments and community on stack overflow," in *Hawaii: 53rd Hawaii International Conference on System Sciences*, 2020, pp. 2898 – 2907.

[2] A. Fontão, B. Ábia, I. Wiese, B. Estácio, M. Quinta, R. P. dos Santos, and A. C. Dias-Neto, "Supporting governance of mobile application developers from mining and analyzing technical questions in stack overflow," *Journal of Software Engineering Research and Development*, vol. 6, no. 1, pp. 1–34, 2018.

[3] A. Zagalsky, D. M. German, M.-A. Storey, C. G. Teshima, and G. Poo-Caamaño, "How the r community creates and curates knowledge: an extended study of stack overflow and mailing lists," *Empirical Software Engineering*, vol. 23, no. 2, pp. 953–986, 2018.

[4] M. M. Rahman, C. K. Roy, and I. Keivanloo, "Recommending insightful comments for source code using crowdsourced knowledge," in *The 15th IEEE International Working Conference on Source Code Analysis and Manipulation (SCAM 2015)*, 2015, pp. 81 – 90.

[5] J. Cheriyan, B. T. R. Savarimuthu, and S. Cranefield, "Norm violation in online communities–a study of stack overflow comments," in *Coordination, Organizations, Institutions, Norms, and Ethics for Governance of Multi-Agent Systems XIII*. Springer, 2017, pp. 20–34.

[6] A. Diyanati, B. S. Sheykhahmadloo, S. M. Fakhrahmad, M. H. Sadredini, and M. H. Diyanati, "A proposed approach to determining expertise level of stackoverflow programmers based on mining of user comments," *Journal of Computer Languages*, vol. 61, p. 101000, 2020.

[7] H. Zhang, S. Wang, T.-H. Chen, A. E. Hassan, and Y. Zou, "An empirical study of obsolete answers on stack overflow," in *The 42nd IEEE International Conference on Software Engineering*, 2020, pp. 1 – 14.

[8] P. Rani, S. Abukar, N. Stulova, A. Bergel, and O. Nierstrasz, "Do comments follow commenting conventions? a case study in java and python," in *2021 IEEE 21st International Working Conference on Source Code Analysis and Manipulation (SCAM)*. IEEE, 2021, pp. 165–169.

TABLE V. SUMMARY OF THE RESULTS OBTAINED THROUGH THE CONFUSION MATRICES RELEVANT TO EACH SVM MODEL

| SVM Models | | Number of correctly classified comments in each useful comment category | | | Total number of correctly classified comments |
|---|---|---|---|---|---|
| | | Constructive Criticism | Relevant Information | Request Clarification | |
| Initial SVM Model | | 173 | 186 | 184 | 543 |
| SVM Models with optimal combinations of hyperparameters | Linear Kernel | 175 | 174 | 185 | 534 |
| | RBF Kernel | 179 | 185 | 179 | 543 |
| | Polynomial Kernel | 176 | 174 | 185 | 535 |
| | Sigmoid Kernel | 175 | 174 | 184 | 533 |
| | Precomputed Kernel | 175 | 174 | 185 | 534 |

TABLE VI. RESULTS OBTAINED THROUGH THE CLASSIFICATION REPORT

| Metrics of Evaluation | SVM Models | | | | | |
|---|---|---|---|---|---|---|
| | Initial SVM Model | SVM Models with optimal combinations of hyperparameters | | | | |
| | | Linear Kernel | RBF Kernel | Polynomial Kernel | Sigmoid Kernel | Precomputed Kernel |
| Accuracy | 0.87 | 0.86 | 0.87 | 0.86 | 0.85 | 0.86 |
| Micro Average for Precision | 0.87 | 0.86 | 0.87 | 0.86 | 0.85 | 0.86 |
| Macro Average for Precision | 0.88 | 0.87 | 0.87 | 0.87 | 0.87 | 0.87 |
| Weighted Average for Precision | 0.88 | 0.87 | 0.87 | 0.87 | 0.87 | 0.87 |
| Micro Average for Recall | 0.87 | 0.86 | 0.87 | 0.86 | 0.85 | 0.86 |
| Macro Average for Recall | 0.87 | 0.86 | 0.87 | 0.86 | 0.86 | 0.86 |
| Weighted Average for Recall | 0.87 | 0.86 | 0.87 | 0.86 | 0.85 | 0.86 |
| Micro Average for F1-Score | 0.87 | 0.86 | 0.87 | 0.86 | 0.85 | 0.86 |
| Macro Average for F1-Score | 0.87 | 0.86 | 0.87 | 0.86 | 0.86 | 0.86 |
| Weighted Average for F1-Score | 0.87 | 0.86 | 0.87 | 0.86 | 0.86 | 0.86 |

[9] A. Soni and S. Nadi, "Analyzing comment-induced updates on stack overflow," in *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, 2021.

[10] S. Mondal, C. K. Saifullah, A. Bhattacharjee, M. M. Rahman, and C. K. Roy, "Early detection and guidelines to improve unanswered questions on stack overflow," in *14th Innovations in Software Engineering Conference (formerly known as India Software Engineering Conference)*, 2021, pp. 1–11.

[11] H. Tang and S. Nadi, "On using stack overflow comment-edit pairs to recommend code maintenance changes," *Empirical Software Engineering*, vol. 26, no. 4, pp. 1–35, 2021.

[12] A. Sulke and A. Varude, "Classification of online pernicious comments using machine learning," *International Journal for Scientific Research and Development, Oune, India*, vol. 7, no. 8, pp. 2321–0613, 2019.

[13] A. S. M. Venigalla, C. S. Lakkundi, and S. Chimalakonda, "Sotagger - towards classifying stack overflow posts through contextual tagging," 2019, pp. 493 – 496.

[14] S. Beyer, C. Macho, M. Di Penta, and M. Pinzger, "What kind of questions do developers ask on stack overflow? a comparison of automated approaches to classify posts into question categories," *Empirical Software Engineering*, vol. 25, no. 3, pp. 2258–2301, 2020.

[15] M. A. Saif, A. N. Medvedev, M. A. Medvedev, and T. Atanasova, "Classification of online toxic comments using the logistic regression and neural networks models," in *AIP Conference Proceedings*, 2018, pp. 1 – 5.

[16] M. Choetkiertikul, D. Avery, H. K. Dam, T. Tran, and A. Ghose, "Who will answer my question on stack overflow?" in *2015 24th Australasian Software Engineering Conference*, 2015, pp. 155 – 164.

[17] R. Abdalkareem, E. Shihab, and J. Rilling, "On code reuse from stackoverflow : An exploratory study on android apps," *Information and Software Technology*, vol. 88, 2017.

[18] Y. Wu, S. Wang, C.-P. Bezemer, and K. Inoue, "How do developers utilize source code from stack overflow?" *Empirical Software Engineering*, vol. 24, p. 637 – 673, 2019.

[19] G. Digkas, N. Nikolaidis, A. Ampatzoglou, and A. Chatzigeorgiou, "Reusing code from stackoverflow: The effect on technical debt," 2019.

[20] N. Novielli, F. Calefato, and F. Lanubile, "A gold standard for emotion annotation in stack overflow," 2018, pp. 14–17.

[21] E. Wong, J. Yang, and L. Tan, "Autocomment: Mining question and answer sites for automatic comment generation," in *IEEE, Palo Alto, USA*, 2013, pp. 562–567.

[22] C. Vassallo, S. Panichella, M. Di Penta, and G. Canfora, "Codes: Mining source code descriptions from developers discussions," *CODES: mining sourCe cOde Descriptions from developErs diScussions. Hyderabad*, 2014.

[23] N. Chakrabarty, *A Machine Learning Approach To Comment Toxicity Classification*, 2012.

[24] N. Pearse, "An illustration of deductive analysis in qualitative research," in *18th European Conference on Research Methodology for Business and Management Studies*, 2019, p. 264.

[25] L. Havrlant and V. Kreinovich, "A simple probabilistic explanation of term frequency-inverse document frequency (tf-idf) heuristic (and variations motivated by this explanation)," *International Journal of General Systems*, vol. 46, pp. 27–36, 2017.

[26] S. Qaiser and R. Ali, "Text mining: Use of tf-idf to examine the relevance of words to documents," *International Journal of Computer Applications*, vol. 181, pp. 25–29, 2018.

[27] J. A. Sãez, M. Galar, and B. Krawczyk, "Addressing the overlapping data problem in classification using the one-vs-one decomposition strategy," pp. 83 396–83 411, 2019.