Comparative Study of Toxic Comment Classification using NLP and various Machine Learning Models

Zawad Alam
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
zawad.alam@g.bracu.ac.bd

Sazal Kanti Kundu
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
sazal.kanti.kundu@g.bracu.ac.bd

Arnob Kumar Dey
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
arnob.kumar.dey@g.bracu.ac.bd

Mahmudul Hasan Shakil
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
mahmudul.hasan.shakil@g.bracu.ac.bd

Shihab Sharar
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
shihab.sharar@g.bracu.ac.bd

Annajiat Alim Rasel
Dept. of Computer Science and Engineering
BRAC University
Dhaka, Bangladesh
annajiat@bracu.ac.bd

***Abstract*— The rapid growth of information technology and thedisruptive transformation of social media have happened in recent years. Websites like Facebook, Twitter, Instagram, where people can express their thoughts or feelings by posting text, photos or videos, have become incredibly popular. But unfortunately, it has also become a place for hateful activity, abusive words, cyberbullying and anonymous threats. There are many existing works in this field but those are not fully successful yet to provide accuracy in satisfactory level. In this work, we employ Natural Language Processing (NLP) with Multinomial Naive Bayes, Logistic Regression and Linear Support Vector Machine for segmenting toxic comments atfirst and then classifying them in six types from a large pool of documents provided by Kaggle's regarding Wikipedia's talk page edits. Using this dataset and based on the 'toxic' label, we got the result applying confusion matrix that a word is toxic and classifier detect it as toxic in case of Multinomial Naive Bayes model is 53%, Logistic Regression model is 71% and Linear SVC model is 78%.**

Keywords— cyberbullying, natural language processing, multinomial naive Bayes, logistic regression, linear support vector machine.

I. INTRODUCTION

Social networking and social media are growing rapidlyover the years. Today, people can use this social platform to communicate with others, express themselves and their opinions. Sometimes there is a possibility that discussion may arise due to disagreement. However, in many cases these discussions have a dirty side and are likely to lead to quarrels surrounding social media. Meanwhile, toxic comments are used on one side. These toxic comments canbe intimidating, obscene, offensive, or identity-based hatred.Our project aims to build a multi-headed model that's capable of detecting different types of toxicity commentslike threats, obscenity, insults, and identity-based hate. In this paper, we are using four basic neural networks which are Natural language processing (NLP), Multinomial Naive Bayes, Logistic Regression and Linear Support Vector Machine. Natural language processing (NLP) mainly forremoving the unusual comments. Following data preprocessing, the proposed architecture is organized implementing procedures for data cleaning and adopting NLP methods such as tokenization, lemmatization,stemming, and vector word translation method. Then we have used three different classifiers to classify the comments. on different toxicity labels. Our model has been tested and accuracy checks have been used to see how efficiently it's working. This paper is organized as follows: Section II discuss on the existing works and approaches used by other researchers, Section III describes the Proposed Methodology including the dataset used, data visualizations and model construction, Section IV deals with the Results and Discussion, Finally, the paper is concluded with future scope in Section V.

II. EXISTING WORKS

Mechanization of data extraction from qualification models will give a discovery in successful usage of data for quiet pursuit in clinical information bases. A larger part of qualification standards contains fleeting data related to ailments and occasions. This venture makes a novel natural language processing (NLP) pipeline for extraction and characterization of transient data as memorable, current, and arranged from free-text qualification models [1]. The pipeline utilizes design learning calculations for extracting fleeting data and prepares a Random Forest classifier for arrangement.

Authors investigate text classification problems and equate SVM to kNN and naive Bayes on binary classification assignments. It's imperative to analyze advanced variants of these algorithms, which is the thing that we've done. Researches show that all the classifiers have accomplished equivalent execution on most issues. SVM had fairly good overall results [2].

SMS Complaint is an electronic public complaint tool for reporting issues on government performance. Text mining classification utilized to determine the value of each complaint category. The SMS data in this study sourced from the SMS Complaint Service of Ambon City Government. There were 6 categories of classification, namely Public Service, Infrastructure, Bureaucracy, Health, Education, and Social. The classification is performed to measure levels of accuracy of the Stemming process and non-Stemming process represented in Matrix with values of recall, precision, and f1

score [3]. The methods used in the measurement were Naive Bayes Multinomial. With the Naive Bayes method, an accuracy level with stemming of 91.38% obtained and while the accuracy level without stemming was 90.73%. The results showed that the Naive Bayes method could be used effectively to predict complaint data through stemming.

The authors discussed how the computing environment provides the possibility of carrying out various data-intensive natural language processing tasks. Language tokenization methods applied for multi-class text classification are recently investigated by many data scientists. The authors of this paper [4] investigate the Logistic Regression method by evaluating classification accuracy which correlates on the size of the training data, POS and number of n-grams. Logistic Regression method is implemented in Apache Spark, the in-memory intensive computing platform. Experimental results have shown that applied multi-class classification method for Amazon product-review data using POS features has higher classification accuracy.

.

III. Proposed Methodology

It consists of 4 subsections: Section A describes the dataset we used, Section B deals with Data Visualization, Section C deals with the Feature Engineering, Section D discusses on benchmarking different vectorizers and Section E describes on modeling and Evaluation.

A. The Dataset

We used a dataset of comments from Wikipedia Talk Page which is collected from Kaggle. We took 3 types of dataset: train, test and test y. It is to be noted that the training data contains 159,571 observations with 8 columns and the test data contains 153,164 observations with 2 columns. The dataset consists of 6 levels of comments: toxic, severe toxic, obscene, threat, insult, identity hate.

B. Data Visualization

For data visualization, we use a bar chart showing the number of comments length by the label's frequency. It is seen that label toxic has the most observations in the training dataset while threat has the least.

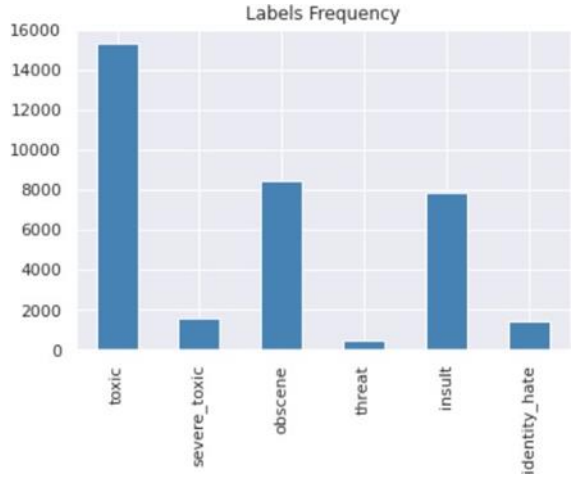

Fig. 1: Sub setting labels from the training data

There is significant class imbalance since the majority of the comments are considered non-toxic. Toxic comments are considered as 1 and non-toxic comments considered as 0. Below is the plot for the labeled data frequency.

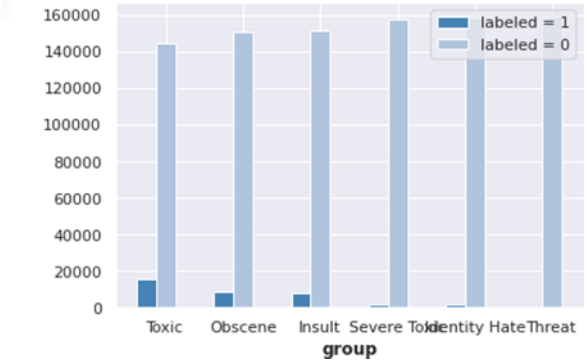

Fig. 2: Visualizing distribution of classes within each label

As seen in the cross-correlation matrix, there is a high chance of obscene comments to be insulting. In order to get an idea of what are the words that contribute the most to different labels, we write a function to generate word clouds. The function takes in a parameter label (i.e. toxic, insult, threat, etc.).

C. Feature-Engineering

Before fitting models, we need to break down the sentence into unique words by tokenizing the comments. In the tokenize function, all the punctuations and special characters in the comments are removed. Frequently occurring common words like articles, prepositions etc. are called stop words. We also removed stop words, filtered out non-ascii characters after observing the results of feature engineering [5].

In many languages, words appear in several inflected forms. The process of grouping together the inflected forms of a word so they can be analyzed as a single item is known as lemmatization. For example, in English, the verb 'to run' may appear as 'run', 'runner', 'runs' or 'running'. The base form, 'run' that one might look up in a dictionary is considered a lemma. So, we lemmatize the comments and filter out comments with length below 3. Besides lemmatization, we also tried stemming but did not get a better result.

D. Benchmarking Different Vectorizer

We are determined to use TF IDF to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus. TFIDF is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. This is done by multiplying two metrics: how many times a word appears in a document, and the inverse document frequency of the word across a set of documents.

Besides TFIDF, we also tried CountVectorizer. CountVectorizer is a great tool provided by the scikit-learn library in Python. It is used to convert a collection of text documents to a vector of term/token counts. It also enables the pre-processing of text data prior to generating the vector representation. However, it is not performing as well as TFIDF. The TfidfVectorizer is actually the CountVectorizer followed by TfidfTransformer. TfidfTransformer transforms a count matrix to a normalized Term-Frequency or Term Frequency-Inverse Document Frequency representation. The goal of using TF IDF instead of the raw frequencies of

occurrence of a token in a given document is to scale down the impact of tokens that occur very frequently in a given corpus and that are hence empirically less informative than features that occur in a small fraction of the training corpus [6]. That's why we can improve the accuracy here.

For example: Since this corpus consists of data from the Wikipedia's talk page edits, the words such as wiki, Wikipedia, edit, page are very common. But for our classification purposes they do not provide us useful information and that should probably be the reason why TFIDF worked better than CountVectorizer.

E. Modeling and Evaluation

1) Baseline Model

We choose Naive Bayes as our baseline model, specifically Multinomial Naive Bayes. Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). It is suitable for classification with discrete features. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Also, we want to compare between different models, especially models that perform well in text classification. Thus, we choose to compare Multinomial Naive Bayes with Logistic Regression and Linear Support Vector Machines. Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist [7]. On the other hand, Linear SVC (Support Vector Classifier)

is to fit the data, returning a best fit hyperplane that divides or categorizes the data. From there after getting the hyperplane, feed some features to the classifier to see what the predicted class is.

2) Evaluation Metrics

Our main metric for measuring model performance is F1 score, since we have 6 labels, the F1-score would be the average of 6 labels. We will also take other metrics into consideration while evaluating models, e.g, Hamming loss, recall, accuracy and precision.
Cross Validation

Cross validation (CV) is one of the techniques used to test the effectiveness of machine learning models. It is also a re-sampling procedure used to evaluate a model if we have limited data. It is mainly used where the goal is prediction, and one wants to estimate how accurately a predictive model will perform in practice. We use Cross Validation to compare between the baseline model and the other two models that we have chosen (Logistic Regression and Linear SVC).

3) Confusion matrix

In classification problems, confusion matrix indicates performance measurement. It shows the confused condition of a model during prediction. The output can be two or more than two depending on the problem statement.

IV. Results and Discussion

In any analysis or project, result's the foremost very important portion because it shows the end result or findings of any analysis or model. During this portion, we are going to assess the results through some metrics of performance and show them mistreatment graphical illustration.

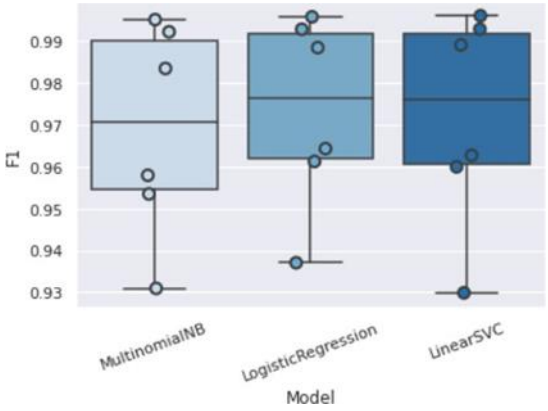

Fig. 4: Visualizing F1 score results through box-plot

Above are the result table and plot showing a comparison between these different models after training them and seeing how these models perform on the test data. Notice that Multinomial Naive Bayes does not perform as well as the other two models while Linear SVC in general outperforms the others based on F1 score.

Based on the cross validation, we noticed that overall, the linear SVC model and Logistic Regression model perform better. Multinomial Naive Bayes does not perform well, especially for the threat label and identity_hate label because these two labels have the least number of observations. We have calculated four metrics such as accuracy, precision, recall and f1 score for all three different classifiers.

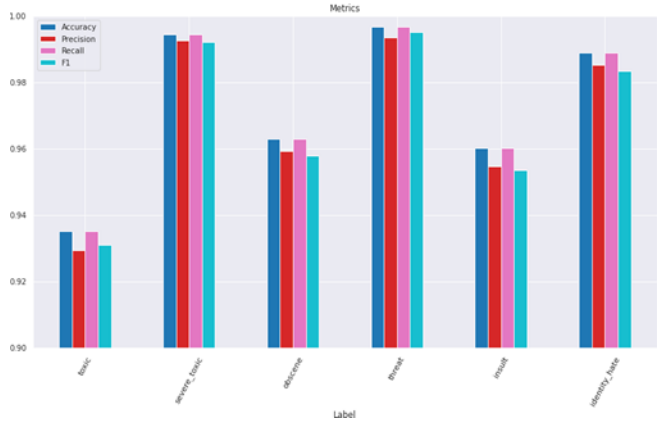

Fig. 5: Plot for Multinomial Naive Bayes

Figure 5 shows the classification report with accuracy, precision, recall and f1 score using Multinomial Naive Bayes as classification model.

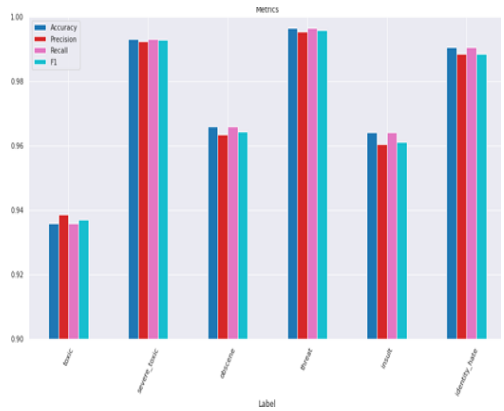

Fig. 6: Plot for Logistic regression

Figure 6 shows the classification report with accuracy, precision, recall and f1 score using Logistic regression as classification model.

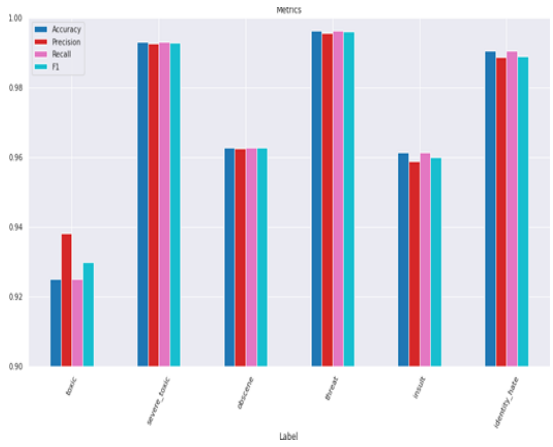

Fig. 7: Plot for Linear SVC

Figure 7 shows the classification report with accuracy, precision, recall and f1 score using Linear SVC as classification model.

Below shows the confusion matrix for label toxic. Notice that all models predict Non-toxic labels pretty well because most of our data are non-toxic. However, Multinomial NB tends to predict more toxic comments to non-toxic while Linear SVC is doing a great job in classifying toxic comments.

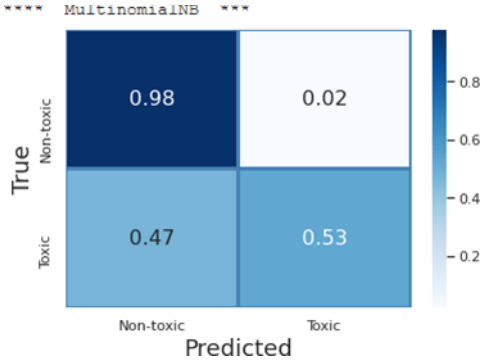

Fig. 8: Confusion Matrix for Multinomial Naive Bayes (toxic)

Figure 8 shows the confusion matrix for the model using MultinomialNB where we have used our dataset for class toxic. This dataset contains only toxic and non-toxic labeled data. In the matrix, we have true values on the Y axis and predicted values on the X axis. So, the proposed MultinomialNB model can detect toxic 53% correctly from the text and for the non-toxic one it is 98%.

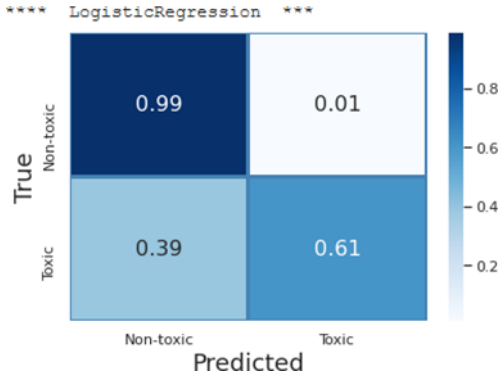

Fig. 12: Confusion Matrix for Logistic Regression (obscene)

Figure 12 shows the confusion matrix for the model using Logistic Regression for class obscene. It is seen that the Logistic Regression model can detect toxic 61% correctly from the text and for the non-toxic one it is 99%.

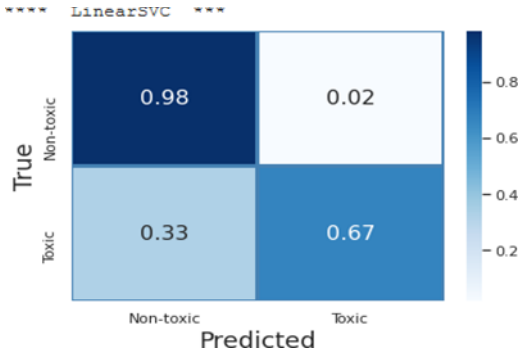

Fig. 13: Confusion Matrix for Linear SVC (obscene)

Figure 13 shows the confusion matrix for the model using Linear SVC for class obscene. It is seen that the Linear SVC model can detect toxic 67% correctly from the text and for the non-toxic one it is 98%.

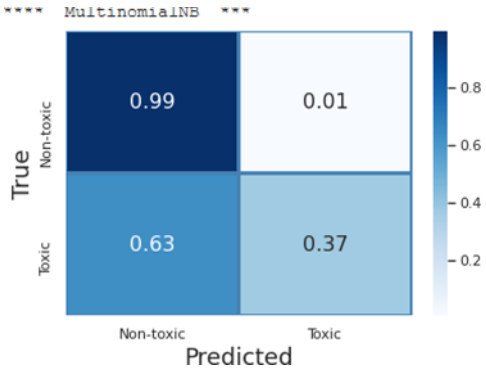

Fig. 14: Confusion Matrix for Multinomial Naive Bayes(insult)

Figure 14 shows the confusion matrix for the model using MultinomialNB for class insult. It is seen that the

MultinomialNB model can detect toxic 37% correctly from the text and for the non-toxic one it is 99%.

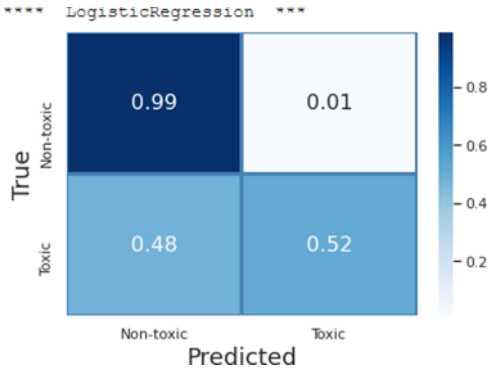

Fig. 15: Confusion Matrix for Logistic Regression(insult)

Figure 15 shows the confusion matrix for the model using Logistic Regression for class insult. It is seen that the Logistic Regression model can detect toxic 52% correctly from the text and for the non-toxic one it is 99%.

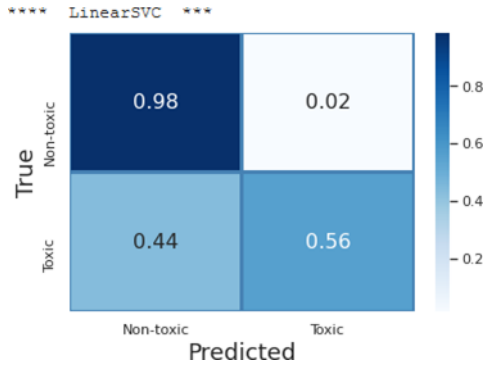

Fig. 16: Confusion Matrix for Linear SVC (insult)

Figure 16 shows the confusion matrix for the model using Linear SVC for class insult. It is seen that the Linear SVC model can detect toxic 56% correctly from the text and for the non-toxic one it is 98%.

Based on the above comparisons, we could say that for these three models with default settings, Linear SVC performs better than anyone for the 'toxic', 'obscene', 'insult' class.

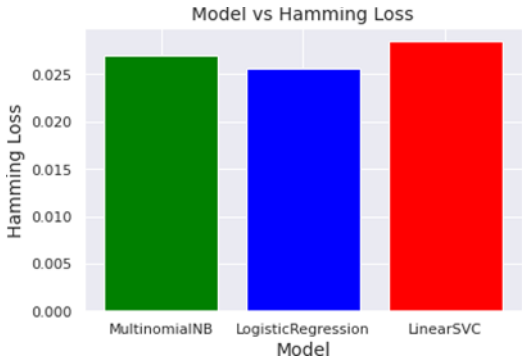

Fig. 11: Hamming Loss for three models

The Hamming loss is the fraction of labels that are incorrectly predicted.

Since hamming loss is designed for multi class while precision, recall, F1-Measure are designed for the binary class, we should decide how to extend accuracy to this case. Across all models, Logistic Regression is doing a great job overall since it has the lowest percentage of incorrect labels.

V. Conclusion

Today, users are using a lot of comments on various social platforms, news portals and forums. When they are communicating sometimes they use comments which are toxic or abusive. Due to the extensively large amount of comments, it is not possible to moderate all of the comments manually. Therefore, most systems use machine learning models to automatically detect specific toxicity levels. In this paper, we used three models for online toxic comments: Multinomial Naive Bayes, Logistic Regression and Linear Support Vector Classifier. Based on the test and training dataset and applying evaluation metrics on these models, we find out that Linear SVC performs better than other models. Using a confusion matrix on class toxicity, Linear SVC shows 78% whereas Multinomial NB and Logistic Regression shows 53% and 71% respectively. We have done a confusion matrix on class obscene and found out that Linear SVC shows 67% whereas Multinomial NB and Logistic Regression shows 46% and 61% respectively. Similarly, we have also done a confusion matrix on insult, found out that Linear SVC shows 56% whereas Multinomial NB and Logistic Regression shows 37% and 52% respectively. Therefore, it is seen that Linear SVC performs better on all classes other than Multinomial NB and Logistic Regression.

In future, we want to implement BERT technic instead of tfidfvectorizer for more precise data preprocessing. And, we also want to include a stratified k-fold method for cross validation section.

References

[1] A. O. G. Parthasarathy and P. Anderson, \Natural language processing pipeline for temporal information extraction and classification from free text eligibility criteria," International Conference on Information Society (i-Society), Dublin, 2016, pp. 120{121, 2016. doi: 10.1109/i-Society.2016.7854192.

[2] F. Colas and P. Brazdil, \On the behavior of svm and some older algorithms in binary text classificationtasks," in International Conference on Text, Speech and Dialogue, Springer, 2006, pp. 45{52}.I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in Magnetism, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.

[3] L. Yance Nanlohy, E. Mulyanto Yuniarno and S. Mardi Susiki Nugroho, "Classification of Public Complaint Data in SMS Complaint Using Naive Bayes Multinomial Method," 2020 4th International Conference on Vocational Education and Training (ICOVET), 2020, pp. 241-246, doi: 10.1109/ICOVET50258.2020.9229941.

[4] T. Pranckevičius and V. Marcinkevičius, "Application of Logistic Regression with part-of-the-speech tagging for multi-class text classification," 2016 IEEE 4th Workshop on Advances in Information, Electronic and Electrical Engineering (AIEEE), 2016, pp. 1-5, doi: 10.1109/AIEEE.2016.7821805.

[5] Chakrabarty, N. (2020). A Machine Learning Approach to Comment Toxicity Classification. In Computational Intelligence in Pattern Recognition (pp. 183-193). Springer, Singapore.M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[6] Gaydhani, A., Doma, V., Kendre, S., & Bhagwat, L. (2018). Detecting hate speech and offensive language on twitter using machine learning: An n-gram and tfidf based approach. arXiv preprint arXiv:1809.08651.

[7] Saif, M. A., Medvedev, A. N., Medvedev, M. A., & Atanasova, T. (2018, December). Classification of online toxic comments using the logistic regression and neural networks models. In AIP conference proceedings (Vol. 2048, No. 1, p. 060011). AIP Publishing LLC.