Automatic Hate Speech Detection using Machine Learning: A Comparative Study

Sindhu Abro[1], Sarang Shaikh[2], Zafar Ali[4]
Sajid Khan[5], Ghulam Mujtaba[6]
Center for Excellence for Robotics, Artificial Intelligence
and Blockchain, Department of Computer Science
Sukkur IBA University, Sukkur, Pakistan

Zahid Hussain Khand[3]
Department of Computer Science
Sukkur IBA University
Sukkur, Pakistan

Abstract—**The increasing use of social media and information sharing has given major benefits to humanity. However, this has also given rise to a variety of challenges including the spreading and sharing of hate speech messages. Thus, to solve this emerging issue in social media sites, recent studies employed a variety of feature engineering techniques and machine learning algorithms to automatically detect the hate speech messages on different datasets. However, to the best of our knowledge, there is no study to compare the variety of feature engineering techniques and machine learning algorithms to evaluate which feature engineering technique and machine learning algorithm outperform on a standard publicly available dataset. Hence, the aim of this paper is to compare the performance of three feature engineering techniques and eight machine learning algorithms to evaluate their performance on a publicly available dataset having three distinct classes. The experimental results showed that the bigram features when used with the support vector machine algorithm best performed with 79% off overall accuracy. Our study holds practical implication and can be used as a baseline study in the area of detecting automatic hate speech messages. Moreover, the output of different comparisons will be used as state-of-art techniques to compare future researches for existing automated text classification techniques.**

Keywords—Hate speech; online social networks; natural language processing; text classification; machine learning

I. Introduction

In recent years, hate speech has been increasing in-person and online communication. The social media as well as other online platforms are playing an extensive role in the breeding and spread of hateful content – eventually which leads to hate crime. For example, according to recent surveys, the rise in online hate speech content has resulted in hate crimes including Trump's election in the US [2], the Manchester and London attacks in the UK [3], and terror attacks in New Zealand [4]. To tackle these harmful consequences of hate speech, different steps including legislation have been taken by the European Union Commission. Recently, the European Union Commission also enforced social media networks to sign an EU hate speech code to remove hate speech content within 24 hours [1]. However, the manual process to identify and remove hate speech content is labor-intensive and time-consuming. Due to these concerns and widespread hate speech content on the internet, there is a strong motivation for automatic hate speech detection.

The automatic detection of hate speech is a challenging task due to disagreements on different hate speech definitions. Therefore, some content might be hateful to some individuals and not to others, based on their concerned definitions. According to [5], hate speech is:

"the content that promotes violence against individuals or groups based on race or ethnic origin, religion, disability, gender, age, veteran status, and sexual orientation/gender identity".

Despite these different definitions, some recent studies claimed favorable results to detect automatic hate speech in the text [21-32]. The proposed solutions employed the different feature engineering techniques and ML algorithms to classify content as hate speech. Regardless of this extensive amount of work, it remains difficult to compare the performance of these approaches to classify hate speech content. To the best of our knowledge, the existing studies lack the comparative analysis of different feature engineering techniques and ML algorithms.

Therefore, this study contributes to solving this problem by comparing three feature engineering and eight ML classifiers on standard hate speech datasets. Table I shows major concepts related to automatic text classification along with their explanations and references. This study holds practical importance and served as a reference for new researchers in the domain of automatic hate speech detection.

This rest of the paper is organized as: Section II highlights the related works. Section III discusses the methodology. Sections IV, V, and VI explain the experimental settings, results, and discussion. Finally, Section VII discusses the limitation, future work, and conclusion as well.

TABLE I. TEXT CLASSIFICATION (KEY CONCEPTS)

S. No.	Concept	Acronym	Definition	References
1	Feature Extraction	FE	It is mapping from text data to real-valued vectors.	[6]
2	Bigram	-	It's a feature engineering technique which represents two adjacent words in a single numeric feature while creating master feature vectors for words.	[7]
3	Term Frequency - Inverse Document Frequency	TFIDF	It's a feature representation technique that represents "word importance" is to a document in the document set. It works in a combination of the frequency of word appearance in a document with no. of documents containing that word.	[8]
4	Word2vec	-	It is a technique used to learn vector representation of words, which can further be used to train machine learning models.	[9]
5	Doc2vec	-	It is an unsupervised technique to learn document representations in fixed-length vectors. It is the same as word2vec, but the only difference is that it is unique among all documents.	[10]
6	Machine Learning Classifiers	ML Classifiers	These are applied to numeric features vector to build the predictive model which can be used for prediction class labels.	[11]
7	Naïve Bayes	NB	It's a probabilistic based classification algorithm, which uses the "Bayes theorem" to predict the class. It works on conditional independence among features.	[12]
8	Random Forest	RF	It's a type of ensemble classifier consisting of many decision trees. It classifies an instance based on voting decision of each decision trees class predictions.	[13]
9	Support Vector Machines	SVM	It's a supervised classification algorithm which constructs an optimal hyperplane by learning from training data which separates the categories while classifying new data.	[14]
10	K Nearest Neighbor	KNN	It's a simple text classification algorithm, which categorize the new data using some similarity measure by comparing it with all available data.	[15]
11	Decision Tree	DT	It is a supervised algorithm. It generates the classification rules in the tree-shaped form, where each internal node denotes attribute conditions, each branch denotes conditions for outcome and leaf node represents the class label.	[16]
12	Adaptive Boosting	AdaBoost	It is one of the best-boosting algorithms, which strengthens the weak learning algorithms.	[17]
13	Multilayer Perceptron	MLP	It is a feedforward artificial neural network. It produces a set of outputs using a set of inputs	[18]
14	Logistic Regression	LR	It is a predictive analysis. It uses a sigmoid function to explain the relationship between one independent variable and one or more independent variables	[19]

II. RELATED WORKS

These days, hate speech is very common on social media. Therefore, in previous years, some of the researchers have applied a supervised ML-based text classification approach to classify hate speech content. Different researchers have employed different variety of feature representation techniques namely, dictionary-based [21-23], Bag-of-words-based [24-26], N-grams-based [27-29], TFIDF-based [30, 31] and Deep-Learning-based [31].

Peter Burnap et al. [20] employed a dictionary-based approach to identify cyber hate on Twitter. In this research, they employed an N-gram feature engineering technique to generate the numeric vectors from the predefined dictionary of hateful words. The authors fed the generated numeric vector to ML classifier namely, SVM and obtained a maximum of 67% F-score. Stéphan Tulkens et al. [22] also used a dictionary-based approach for the automatic detection of racism in Dutch Social Media. In this study, the authors used the distribution of words over three dictionaries as features. They fed the generated features to the SVM classifier. Their experimental results obtained 0.46 F-Score. Njagi Dennis et al. [21] used ML-based classifier to classify hate speech in web forums and blogs. The authors employed a dictionary-based approach to generate a master feature vector. The features were based on sentiment expressions using semantic and subjectivity features with an orientation to hate speech. Afterward, the authors fed the masters feature vector to a rule-based classifier. In the experimental settings, the authors evaluated their classifier by using a precision performance metric and obtained 73% precision.

Nonetheless, the combination of dictionary-based and ML approaches showed a good result. However, the major disadvantage of such type of approach is that it requires a dictionary, based on the large corpus to look for domain words. To overcome this drawback, many of the researchers have used a BOW-based approach which is similar to a dictionary-based approach but the word features are obtained from training data and not from the predefined dictionaries.

Edel Greevy et al. [23] used the supervised ML approach to classify the racist text. To convert the raw text into numeric vectors, the authors employed a bigram feature extraction technique. The authors used bigram features, with the BOW feature representation technique. They used the SVM classifier to perform experimental results. In their results, they achieved 87% accuracy. Irene Kwok et al. [24] employed an ML-based approach to the automatic detection of racism against black in the twitter community. In their research, they employed unigram with the BOW-based technique to generate the numeric vectors. The authors fed the generated numeric vector to the Naïve Bayes classifier. Their experimental results obtained a maximum of 76% accuracy. Sanjana Sharma et al. [25] classified hate speech on twitter. In their research, they employed BOW features. The authors fed the generated numeric vector to the Naïve Bayes classifier. Their experimental results showed a maximum of 73% accuracy.

Nevertheless, BOW showed better accuracy in social network text classification. However, the major disadvantage of this technique is, the word-order is ignored and causes misclassification as different words are used in different contexts. To overcome this limitation, researchers have proposed an N-grams-based approach [7].

Zeerak Waseem et al. [28] classify the hate speech on twitter. In their research, they employed character Ngrams feature engineering techniques to generate the numeric vectors. The authors fed the generated numeric vector to the LR classifier and obtained overall 73% F-score. Chikashi Nobata et al. [27] used the ML-based approach to detect the abusive language in online user content. In their research authors employed character Ngrams feature representation technique to represent the features. The authors fed the features to the SVM classifier. The results showed that the classifier obtained overall 77% F-score. Shervin Malmasi et al [26] used an ML-based approach to classify hate speech in social media. In their research, the authors employed 4grams with character grams feature engineering techniques to generate numeric features. The authors fed the generated numeric features to the SVM classifier. The authors reported maximum of 78% accuracy.

In recent years, few researchers employed ML approaches to detect automatic hate speech. For example, Karthik Dinakar et al. [29] classified sensitive topics from social media comments or posts. In their research, they employed unigram with the TFIDF feature representation technique to generate the numeric feature vectors. The authors fed the generated features to four ML classifiers namely Naïve Bayes, rule-based, J48, and SVM. Their experimental results showed that the rule-based classifier outperformed NB, J48 and SVM classifiers by obtaining 73% accuracy. Shuhua Liu et al. [30] performed classification on web content pages into hatred or violence categories. In their study, they used trigram features, represented using TFIDF. The authors used the Naïve Bayes classifier. In their experimental settings, the Naïve Bayes classifier obtained highest accuracy of 68%.

The N-gram-based approach gives better results than the BOW-based approach but it has two major limitations. First, the related words may be at a high distance in sentence and finally increasing the N value, results in slow processing speed [32].

In recent years, authors employed deep learning-based NLP techniques to classify hate speech messages. Sebastian Köffer et al. [31] employed word2vec features and SVM classifiers to classify German texts hate speech messages and obtained 67% F-score. The word2Vec showed the lowest results because such approaches need enormous data to learn complex word semantics.

Recently, there has been a good attempt to construction and detection of hate speech as well as offensive language in other languages (i-e: Danish). An important research study [45] in 2019 worked on the construction of Danish dataset for hate speech and offensive language detection. The dataset contained comments from Reddit and Facebook. It also contained the various types and targets of the offensive language. The authors achieved the highest F1 score of 0.74 by using deep learning models with different features sets.

Schmidt et al. [46] conducted a survey on hate speech detection using natural language processing in 2017. The authors discussed in detail studies regarding various feature engineering techniques to be used for supervised classification of hate speech messages. The major drawback of this survey is that there were no experimental results for those mentioned techniques.

Previous studies showed that a variety of researchers from across the globe are working on hate speech recognition written in different languages such as German, Dutch and English. However, according to our information, no study provides a comparative study of various features and ML algorithms on the standard dataset that can serve as a baseline study for future researchers in the field of hate speech recognition. Hence, in this study, we compared three feature engineering and eight ML classifiers to evaluate which one best works on hate speech datasets (discussed in Section III).

III. Methodology

This section explains the proposed system which we have employed to classify tweets into three different classes namely, "hate speech, offensive but not hate speech, and neither hate speech nor offensive speech". Fig. 1 shows the complete research methodology. As shown in this figure, the research methodology is contained of six key steps namely, data collection, data preprocessing, feature engineering, data splitting, classification model construction, and classification model evaluation. Each of the step is discussed in detail in the subsequent sections.

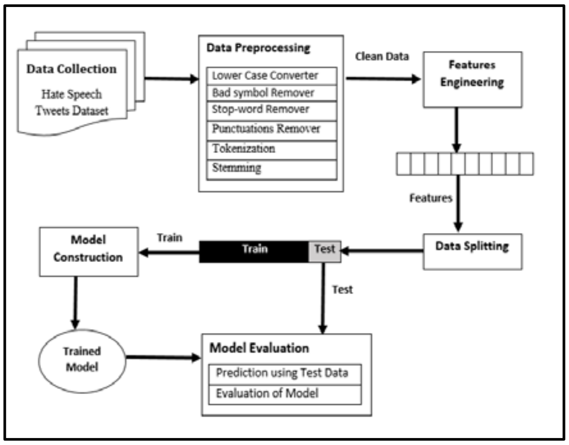


Fig. 1. System Overview.

A. Data Collection

In this research study, we collected publicly available hate speech tweets dataset. This dataset is compiled and labeled by CrowdFlower. In this dataset, the tweets are labeled into three distinct classes, namely, hate speech, not offensive, and offensive but not hate speech. This dataset has 14509 number of tweets. Of these, 16% of tweets belong to class hate speech. In addition, 50% of tweets belong to not offensive class and the remaining 33% tweets are offensive but not hate speech class. The details of this distribution are also shown in Fig. 2.

B. Text Preprocessing

Several research studies have explained that using text preprocessing makes better classification results [33]. So, in our dataset, we applied different preprocessing-techniques to filter noisy and non-informative features from the tweets. In preprocessing, we changed the tweets into lower case. Also, we removed all the URLs, usernames, white spaces, hashtags, punctuations and stop-words using pattern matching techniques from the collected tweets. Besides this, we have also performed tokenization and stemming from preprocessed tweets. The tokenization, converts each single tweet into tokens or words, then the porter stemmer converts words to their root forms, such as offended to offend using porter stemmer.

C. Feature Engineering

The ML algorithms cannot understand the classification rules from the raw text. These algorithms need numerical features to understand classification rules. Hence, in text-classification one of the key steps is feature engineering. This step is used for extracting the key features from raw text and representing the extracted features in numerical form. In this study, we have performed three different features engineering techniques, namely, *n*-gram with TFIDF [8], Word2vec [9] and Doc2vec [10].

D. Data Splitting

Table II shows the class-wise distribution of the overall dataset as well as data set after splitting (i.e. Training set and Test set). We have used the 80-20 ratio to split the preprocessed data (i.e. 80% for Training Data and 20% for Test Data). The training data is used to train the classification model to learn classification rules. Moreover, the test data is further used to evaluate the classification model.

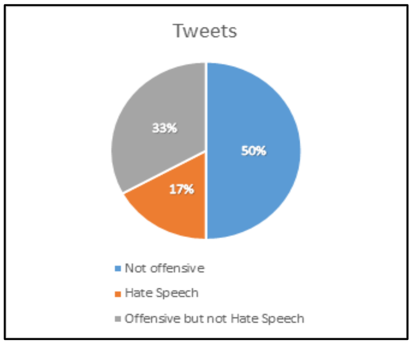


Fig. 2. Class wise Data Distribution.

TABLE II. DETAILS OF DATA SPLIT

	Class	Total Instances	Training instances	Testing instances
0	Hate Speech	2399	1909	490
1	Not offensive	7274	5815	1459
2	Offensive but not Hate Speech	4836	3883	953
	Total	**14509**	**1607**	**2902**

E. Machine Learning Models

According to “no free lunch theorem” [34], there is no any single classifier which best performs on all kinds of datasets. Therefore, it is recommended to apply several different classifiers on a master feature vector to observe which one reaches to the better results. Hence, we selected *eight* different classifiers NB [12], SVM [14], KNN [15], DT [16], RF [13], AdaBoost [17], MLP [18] and LR [19].

F. Classifier Evaluation

In this step, the constructed classifier predicts the class of unlabeled text (i.e. “hate speech, offensive but not hate speech, neither hate speech nor offensive speech”) using test set. The classifier performance is evaluated by calculating true negatives (TN), false positives (FP), false negatives (FN) and true positives (TP). These four numbers constitute a confusion matrix as in Fig. 3. Different performance metrics are used to assess the performance of the constructed classifier. Some common performance measures in text categorization are discussed briefly below. The more details of performance metrics can be found in [35].

1) Precision: Precision is also known as the positive predicted value. It is the proportion of predictive positives which are actually positive. Refer to “(1)”.

$$Precision = \frac{TP}{(TP+FP)} \tag{1}$$

2) Recall: It is the proportion of actual positives which are predicted positive. Refer to “(2)”.

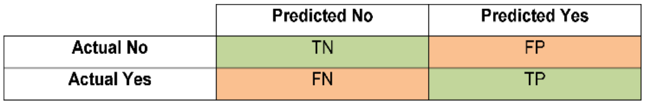
$$Recall = \frac{TP}{(TP+FN)} \tag{2}$$

3) F-Measure: It is the harmonic mean of precision and recall (as shown in Equation 3). The standard F-measure (F1) gives equal importance to precision and recall. Refer to “(3)”.

$$F - measure = \frac{2 \times (precision \times recall)}{(precision + recall)} \tag{3}$$

4) Accuracy: It is the number of correctly classified instances (true positives and true negatives). Refer to “(4)”.

$$Accuracy = \frac{(TP+TN)}{TP+FP+TN+FN} \tag{4}$$



	Predicted No	Predicted Yes
Actual No	TN	FP
Actual Yes	FN	TP

Fig. 3. Confusion Matrix.

IV. Experimental Settings

As mentioned in section C, we used three types of features namely *n*-gram (bigram) with TFIDF, Word2vec and Doc2vec. Hence, we have a total of three different master feature representations. In addition, eight different ML algorithms were applied to the created three master feature vectors. Hence, overall 24 analyses (3 master feature vectors x 8 ML algorithms) were evaluated to check the effectiveness of classification models.

V. Results

This section explains the overall results of 24 analyses. Tables III to Table VI shows the precision, recall, F-measure and accuracy of all 24 analyses, respectively. The bold values represented are the maximum and minimum result values. All the tables are showing performance for different features representation and classification techniques applied in experimental settings. In all 24 analyses, the lowest precision (0.58), recall (0.57), accuracy (57%) and F-measure (0.47) found in MLP and KNN classifier using TFIDF features representation with bigram features. Moreover, the highest recall (0.79), precision (0.77), accuracy (79%) and F-measure (0.77) were obtained by SVM using TFIDF features representation with bigram features. In feature representation, bigram features with TFIDF obtained the best performance as compared to Word2vec and Doc2vec. However, there was a fringe difference between the result observed in bigram, and Doc2vec. In text-classification models, the SVM classifier best performed among all the eight classifiers. However, the AdaBoost and RF classifiers results were lesser than SVM results and were better than LR, DT, NB, KNN, and MLP results.

Furthermore, Fig. 4 and Fig. 5 show the confusion matrix of best-performing analyses. Fig. 4 shows the SVM classifiers' confusion matrix using bigram with TFIDF features. As shown here, out of 490 tweets belonging to hate speech class, only 155 were correctly classified. However, the 335 instances were incorrectly classified. Of these 335 instances, 54 were falsely classified as not offensive and 281 were falsely classified as Offensive but not Hate Speech. The 1459 instances belong to the second class, the 1427 tweets were correctly classified as not offensive speech. The remaining 32 instances were misclassified, 5 were incorrectly classified as hate speech and 27 were falsely classified as an offensive language but not hate speech. The remaining 953 instances out of 2902 test set belonging to offensive language but not hate speech class. Here, the SVM classifier correctly classified the 698 tweets as an offensive language but not hate speech. The 122 and 133 instances were misclassified into hate speech and not offensive speech, respectively.

However, Fig. 5 shows the confusion matrix of the Adaboost classifier using bigram with TFIDF features. As shown here, the overall performance of the Adaboost classifier is lower than the SVM classifier while using bigram with TFIDF features. The Adaboost only performed well in offensive language but not hate speech class.

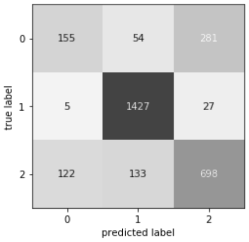


Fig. 4. Confusion Matrix (Features: Bigram (TFIDF), Classifier: SVM).

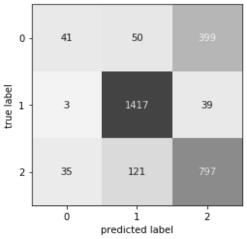


Fig. 5. Confusion Matrix (Features: Bigram (TFIDF), Classifier: ADABOOST).

TABLE III. Precision of all 24 Analysis

Features	**LR**	**NB**	**RF**	**SVM**	**KNN**	**DT**	**AdaBoost**	**MLP**
Bigram	0.72	0.71	0.73	**0.77**	0.61	0.71	**0.75**	**0.58**
Word2vec	0.69	0.66	0.66	0.70	0.64	**0.62**	0.65	0.69
Doc2vec	0.70	0.65	0.65	0.70	0.69	**0.61**	0.66	0.71

The bold marked values represented are the higher and lower result values.

TABLE IV. Recall of all 24 Analysis

Features	**LR**	**NB**	**RF**	**SVM**	**KNN**	**DT**	**AdaBoost**	**MLP**
Bigram	0.75	0.73	0.75	**0.79**	**0.57**	0.73	**0.78**	0.70
Word2vec	0.72	0.67	0.68	0.73	**0.61**	0.63	0.68	0.71
Doc2vec	0.72	**0.62**	0.67	0.72	0.65	0.63	0.67	0.71

The bold marked values represented are the higher and lower result values.

TABLE V. F-MEASURE OF ALL 24 ANALYSIS

Features	LR	NB	RF	SVM	KNN	DT	AdaBoost	MLP
Bigram	0.72	0.68	0.74	**0.77**	**0.47**	0.71	0.73	0.63
Word2vec	0.69	0.66	0.66	0.70	0.61	**0.60**	0.65	0.65
Doc2vec	0.70	0.63	0.66	0.72	0.65	**0.61**	0.66	0.66

The bold marked values represented are the higher and lower result values.

TABLE VI. ACCURACY OF ALL 24 ANALYSIS

Features	LR	NB	RF	SVM	KNN	DT	AdaBoost	MLP
Bigram	0.75	0.73	0.75	**0.79**	**0.57**	0.73	0.78	0.70
Word2vec	0.72	0.67	0.68	0.73	**0.61**	0.63	0.68	0.71
Doc2vec	0.72	**0.62**	0.67	0.72	0.65	0.63	0.67	0.71

The bold marked values represented are the higher and lower result values.

VI. DISCUSSION

In the experimental work, we have evaluated eight classifiers over three different feature engineering techniques, giving 24 different analyses over hate speech dataset containing three classes. Our experimental results showed that the SVM algorithm with the combination of bigram with TFIDF FE techniques showed the best results. The theoretical analysis is discussed in subsequent sections.

A. *Feature Engineering*

The selection of feature engineering is important in text classification. In this study, we compared three distinct feature extraction techniques namely, Bigram with TFIDF, word2vec and doc2vec. The experimental results exhibited that from these three techniques, bigram with TFIDF outperformed. Conversely, the Word2vec and Doc2vec showed lower results. The possible reason for the outperformance of bigram and TFIDF is that bigram maintains the sequence of words compared to word2Vec and doc2vec [36]. Moreover, several studies showed that the TFIDF representation technique is better than the binary and term frequency representation [6].

The possible reason for the lower performance of Word2vec is because it is unable to handle OOV (out-of-vocabulary) words specially in the domain of Twitter data. Moreover, Word2Vec requires a huge amount of training set to learn the complex relationship between the words [37]. However, as shown in Table I (data collection table), our dataset has approximately 15000 tweets, which might be not enough to train effectively to word2vec for eliciting the complex word relationship.

In our experimental results, Doc2Vec also showed lower performance. This might be because it performs low in case of very short length documents [38] and the tweets which we used in our dataset often having 280 character length.

B. *Machine Learning Classifier*

Several studies proved that no single ML algorithm performed better on all kinds of data. Therefore, the comparison of various ML algorithms is required to discover which one is best performing on the given dataset. Hence, on our dataset, we used eight different ML algorithms as discussed in Section 3.E i.e. ML Models.

The experimental results proved that SVM and AdaBoost classifiers achieved the best performance possibly because SVM uses threshold functions to separate the data, not the number of features based on margin. This shows that SVM is independent upon the presence of the number of features in the data [7, 15]. In addition, SVM has the capability to best perform on non-linear data apart from the linear data because of its kernel functions. The possible reasons behind the outperformance of AdaBoost are that it uses adaptive algorithms to learn the classification rules iteratively [39] and it focuses on the reduction of the training error. The results obtained with RF and LR classifiers are a little lower than SVM and AdaBoost results but are somewhat higher than the results of NB, DT, KNN, and MLP. The low performance of RF might be due to the unavailability of informative features which leads to incorrect predictions [40]. It is possible that the performance of LR might be lower because its decision surface is linear in nature and cannot handle nonlinear data adequately [41].

The lowest performance was obtained amongst the NB, DT, MLP and KNN classifiers. The NB classifier works on conditional independence among features. Thus, the performance of the NB classifier is negatively affected as the conditional dependence becomes more complicated due to the increase in the number of features [12]. The DT showed lower performance in predicting hate speech because the features inside the master features vector are represented as continuous data points that make it difficult to find the ideal threshold values that are required to build a decision tree [42]. The reason behind the poor performance of the MLP classifier is due to not having enough training data that's why it is considered as complex "black box" [43]. The KNN had the worst performance due to laziness of the learning algorithm and it does not work adequately for noisy data [44]. Hence the KNN is not suitable for detecting hate speech tweets.

C. *Classwise Performance*

As discussed in Section 3.A we have three classes name "hate speech", "offensive but not hate speech" and "neither hate speech nor offensive speech". The results show that all features and classifiers performed well for two classes (i.e. offensive but not hate speech, and neither hate speech nor offensive speech). Our experimental results showed that the 24 combinations performed lowest for class hate speech. According to Table I, the class “Hate Speech” has the lowest training instances as compared to other classes, but the major reason for misclassification of class “Hate Speech” (as shown in Fig. 3 and Fig. 4) might be overlapping of different bigram words with higher frequency in other classes than hate speech class. For example, bigrams like “lame nigga, white trash, bitch made” are more frequently appearing in class “Offensive but not Hate Speech” as compared to class “Hate Speech”. Hence, it might be possible that the classifier learned weak learning rules.

VII. CONCLUSION

This study employed automated text classification techniques to detect hate speech messages. Moreover, this study compared three feature engineering techniques and eight ML algorithms to classify hate speech messages. The experimental results exhibited that the bigram features, when represented through TFIDF, showed better performance as compared to word2Vec and Doc2Vec features engineering techniques. Moreover, SVM and RF algorithms showed better results compared to LR, NB, KNN, DT, AdaBoost, and MLP. The lowest performance was observed in *KNN*. The outcomes from this research study hold practical importance because this will be used as a baseline study to compare upcoming researches within different automatic text classification methods for automatic hate speech detection. Furthermore, this study also holds a scientific value because this study presents experimental results in form of more than one scientific measures used for automatic text classification. Our work has two important limitations. First, the proposed ML model is inefficient in terms of real-time predictions accuracy for the data. Finally, it only classifies the hate speech message in three different classes and is not capable enough to identify the severity of the message. Hence, in the future, the objective is to improve the proposed ML model which can be used to predict the severity of the hate speech message as well. Moreover, to improve the proposed model’s classification performance two approaches will be used. First, the lexicon-based techniques will be explored and assessed by comparing with other current state-of-the-art results. Secondly, more data instances will be collected, to be used for learning the classification rules efficiently.

REFERENCES

[1] Hern, A., Facebook, YouTube, Twitter, and Microsoft sign the EU hate speech code. The Guardian, 2016. 31.

[2] Rosa, J., and Y. Bonilla, Deprovincializing Trump, decolonizing diversity, and unsettling anthropology. American Ethnologist, 2017. 44(2): p. 201-208.

[3] Travis, A., Anti-Muslim hate crime surges after Manchester and London Bridge attacks. The Guardian, 2017.

[4] MacAvaney, S., et al., Hate speech detection: Challenges and solutions. PloS one, 2019. 14(8): p. e0221152.

[5] Fortuna, P. and S. Nunes, A survey on automatic detection of hate speech in text. ACM Computing Surveys (CSUR), 2018. 51(4): p. 85.

[6] Mujtaba, G., et al., Prediction of cause of death from forensic autopsy reports using text classification techniques: A comparative study. Journal of forensic and legal medicine, 2018. 57: p. 41-50.

[7] Cavnar, W.B. and J.M. Trenkle. N-gram-based text categorization. in Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval. 1994. Citeseer.

[8] Ramos, J. Using tf-idf to determine word relevance in document queries. in Proceedings of the first instructional conference on machine learning. 2003. Piscataway, NJ.

[9] Mikolov, T., et al. Distributed representations of words and phrases and their compositionality. in Advances in neural information processing systems. 2013.

[10] Le, Q. and T. Mikolov. Distributed representations of sentences and documents. in International conference on machine learning. 2014.

[11] Kotsiantis, S.B., I.D. Zaharakis, and P.E. Pintelas, Machine learning: a review of classification and combining techniques. Artificial Intelligence Review, 2006. 26(3): p. 159-190.

[12] Lewis, D.D. Naive (Bayes) at forty: The independence assumption in information retrieval. in European conference on machine learning. 1998. Springer.

[13] Xu, B., et al., An Improved Random Forest Classifier for Text Categorization. JCP, 2012. 7(12): p. 2913-2920.

[14] Joachims, T. Text categorization with support vector machines: Learning with many relevant features. in European conference on machine learning. 1998. Springer.

[15] Zhang, M.-L. and Z.-H. Zhou, A k-nearest neighbor based algorithm for multi-label classification. GrC, 2005. 5: p. 718-721.

[16] Abacha, A.B., et al., Text mining for pharmacovigilance: Using machine learning for drug name recognition and drug–drug interaction extraction and classification. Journal of biomedical informatics, 2015. 58: p. 122-132.

[17] Ying, C., et al., Advance and prospects of AdaBoost algorithm. Acta Automatica Sinica, 2013. 39(6): p. 745-758.

[18] Gardner, M.W. and S. Dorling, Artificial neural networks (the multilayer perceptron)—a review of applications in the atmospheric sciences. Atmospheric environment, 1998. 32(14-15): p. 2627-2636.

[19] Wenando, F.A., T.B. Adji, and I. Ardiyanto, Text classification to detect student level of understanding in prior knowledge activation process. Advanced Science Letters, 2017. 23(3): p. 2285-2287.

[20] Burnap, P. and M.L. Williams, Us and them: identifying cyber hate on Twitter across multiple protected characteristics. EPJ Data Science, 2016. 5(1): p. 11.

[21] Gitari, N.D., et al., A lexicon-based approach for hate speech detection. International Journal of Multimedia and Ubiquitous Engineering, 2015. 10(4): p. 215-230.

[22] Tulkens, S., et al., A dictionary-based approach to racism detection in dutch social media. arXiv preprint arXiv:1608.08738, 2016.

[23] Greevy, E. and A.F. Smeaton. Classifying racist texts using a support vector machine. in Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval. 2004. ACM.

[24] Kwok, I. and Y. Wang. Locate the hate: Detecting tweets against blacks. in Twenty-seventh AAAI conference on artificial intelligence. 2013.

[25] Sharma, S., S. Agrawal, and M. Shrivastava, Degree based classification of harmful speech using twitter data. arXiv preprint arXiv:1806.04197, 2018.

[26] Malmasi, S. and M. Zampieri, Detecting hate speech in social media. arXiv preprint arXiv:1712.06427, 2017.

[27] Nobata, C., et al. Abusive language detection in online user content. in Proceedings of the 25th international conference on world wide web. 2016. International World Wide Web Conferences Steering Committee.

[28] Waseem, Z. and D. Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. in Proceedings of the NAACL student research workshop. 2016.

[29] Dinakar, K., R. Reichart, and H. Lieberman. Modeling the detection of textual cyberbullying. in fifth international AAAI conference on weblogs and social media. 2011.

[30] Liu, S. and T. Forss. Combining N-gram based Similarity Analysis with Sentiment Analysis in Web Content Classification. in KDIR. 2014.

[31] Köffer, S., et al., Discussing the value of automatic hate speech detection in online debates. Multikonferenz Wirtschaftsinformatik (MKWI 2018): Data Driven X-Turning Data in Value, Leuphana, Germany, 2018.

[32] Chen, Y., Detecting offensive language in social medias for protection of adolescent online safety. 2011.

[33] Shaikh, S. and S.M. Doudpotta, Aspects Based Opinion Mining for Teacher and Course Evaluation. Sukkur IBA Journal of Computing and Mathematical Sciences, 2019. 3(1): p. 34-43.

[34] Ho, Y.-C. and D.L. Pepyne, Simple explanation of the no-free-lunch theorem and its implications. Journal of optimization theory and applications, 2002. 115(3): p. 549-570.

[35] Seliya, N., T.M. Khoshgoftaar, and J. Van Hulse. A study on the relationships of classifier performance metrics. in 2009 21st IEEE international conference on tools with artificial intelligence. 2009. IEEE.

[36] Chaudhari, U.V. and M. Picheny, Matching criteria for vocabulary-independent search. IEEE Transactions on Audio, Speech, and Language Processing, 2012. 20(5): p. 1633-1643.

[37] Li, Y. and T. Yang, Word embedding for understanding natural language: a survey, in Guide to Big Data Applications. 2018, Springer. p. 83-104.

[38] Wang, Y., et al. Comparisons and selections of features and classifiers for short text classification. in IOP Conference Series: Materials Science and Engineering. 2017. IOP Publishing.

[39] Schapire, R.E., The boosting approach to machine learning: An overview, in Nonlinear estimation and classification. 2003, Springer. p. 149-171.

[40] Xu, B., Y. Ye, and L. Nie. An improved random forest classifier for image classification. in 2012 IEEE International Conference on Information and Automation. 2012. IEEE.

[41] Eftekhar, B., et al., Comparison of artificial neural network and logistic regression models for prediction of mortality in head trauma based on initial clinical data. BMC medical informatics and decision making, 2005. 5(1): p. 3.

[42] Dreiseitl, S., et al., A comparison of machine learning methods for the diagnosis of pigmented skin lesions. Journal of biomedical informatics, 2001. 34(1): p. 28-36.

[43] Singh, P.K. and M.S. Husain, Methodological study of opinion mining and sentiment analysis techniques. International Journal on Soft Computing, 2014. 5(1): p. 11.

[44] Bhatia, N., Survey of nearest neighbor techniques. arXiv preprint arXiv:1007.0085, 2010.

[45] Sigurbergsson, G. I., & Derczynski, L. (2019). Offensive language and hate speech detection for Danish. arXiv preprint arXiv:1908.04531.

[46] Schmidt, A., & Wiegand, M. (2017, April). A survey on hate speech detection using natural language processing. In Proceedings of the Fifth International workshop on natural language processing for social media (pp. 1-10).