**Hate Speech Detection with Machine Learning**

**Name: Raiyan, Mohd. Yah-ya ID: 19-41752-3**

**Name: Saha, Shoibal ID: 19-41802-3**

**Name: Hasan, Jakir ID: 18-38795-3**

**Abstract**

Humanity has benefited greatly from the growing usage of social media and information sharing. However, this has also resulted in a number of issues, such as the dissemination and sharing of hate speech messages. Therefore, recent research used a range of feature engineering approaches and machine learning algorithms to automatically recognize the hate speech messages on diverse datasets in order to address this developing problem on social media sites. To our knowledge, however, no study has been done to contrast the various feature engineering approaches and machine learning algorithms to determine which feature engineering methodology and machine learning algorithm perform best on a common dataset that is made publicly available. This study compares the performance of three feature engineering approaches and eight machine learning algorithms in order to assess how well they perform on a publicly accessible dataset with three different classes. According to the experimental findings, the support vector machine method performed best when combined with bigram features, with an overall accuracy rate of 79%. Our research has real-world applications and can serve as a benchmark in the field of automatically identifying hate speech texts. Additionally, the results of various comparisons will be utilized to evaluate upcoming studies for current automated text categorization approaches using state-of-the-art methods.

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

1. **INTRODUCTION**

Hate speech has been more prevalent both outside and online in recent years. The social media and other internet platforms are heavily contributing to the creation and dissemination of inflammatory information, which finally results in hate crimes. For instance, recent studies indicate that the increase in online hate speech has contributed to hate crimes like Trump's election in the US, the attacks in Manchester and London, as well as terror attacks in New Zealand. The European Union Commission has taken a number of actions, including legislation, to address these negative effects of hate speech. The European Union Commission has mandated that social media platforms sign an EU hate speech policy and delete any hate speech within 24 hours. However, it takes a lot of time and effort to manually find and remove hate speech information. There is a tremendous drive for automatic hate speech identification because of these issues and the prevalence of hate speech content online. Due to conflicts over various definitions of hate speech, automated identification of hate speech is a difficult undertaking. Consequently, depending on their own definitions, some people may find some things to be hostile while others may not. Hate speech is defined as "the content that advocates violence against individuals or groups based on race or ethnic origin, religion, disability, gender, age, veteran status, and sexual orientation/gender identity" by. Despite these various criteria, some recent research  claimed to have had success in automatically detecting hate speech in the text. The suggested approaches used ML algorithms and various feature engineering techniques to categorize information as hate speech. Despite all of this effort, it is still challenging to compare how well any method for classifying hate speech material performs. To the best of our knowledge, no research have been done that compare various feature engineering methods with ML algorithms. Therefore, by contrasting three feature engineering and eight ML classifiers on common hate speech datasets, our work helps to solve this problem. The key terms for automated text classification are included in Table I, along with their definitions and sources. This work is useful in real-world applications and serves as a guide for emerging experts in the field of automatic hate speech recognition. The rest of the essay is structured as follows: The connected works are highlighted in Section II. The approach is covered in Section III. The experimental conditions, findings, and discussion are described in Sections IV, V, and VI. The limitation, future work, and conclusion are all covered in Section VII.

**RELATED WORK**

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 [24-26], N-grams-based [27-29], TFIDF-based [30, 31] and Deep-Learning-based [31]. Peter pet 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. 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. 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. 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]. Waseem et al. [28] classify the hate speech on twitter. In their research, they employed character 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. et al. [27] used the ML-based approach to detect the abusive language in online user content. In their research authors employed character 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 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 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. 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 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.

1. **METHODOLOGY**

The suggested technique used to categorize tweets into three groups—"hate speech, offensive but not hate speech, and neither hate speech nor offensive speech"—is explained in this section. The whole research technique is shown in Fig. 1. Data collecting, data preprocessing, feature engineering, data splitting, classification model development, and classification model assessment are the six main processes of the study technique as indicated in this picture. The next section goes into great depth about each stage.

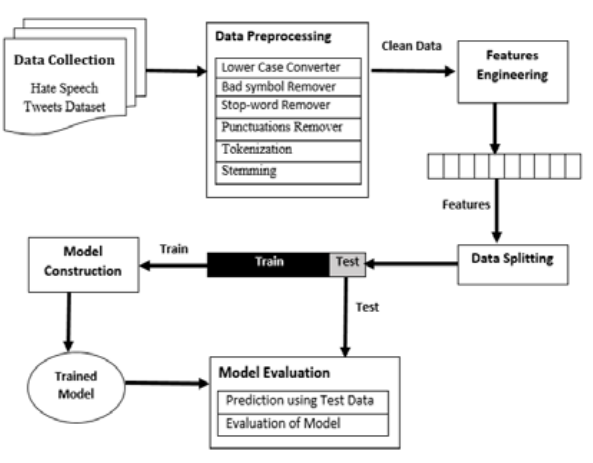


Fig 1. System Overview

**A. Data Collection**

We compiled a dataset of hate speech tweets that were made available to the public for this study. CrowdFlower assembled and annotated this dataset. The tweets in this dataset have been classified into three groups: offensive but not hate speech, not offensive, and offensive but not offensive. There are 14509 tweets in this dataset. 16% of these tweets fall under the category of hate speech. Additionally, 33% of tweets are offensive but not hate speech, while 50% of tweets fall into the non-offensive category. Fig. 2 also displays this distribution's specifics.

**B. Text preprocessing**

Preprocessing of Text preparation produces improved categorization results, according to a number of studies [33]. Therefore, in our dataset, we used several preprocessing approaches to remove irrelevant and noisy information from the tweets. Additionally, using pattern matching algorithms, we eliminated all the URLs, usernames, white spaces, hashtags, punctuation, and stop-words from the gathered tweets. In addition, we have tokenized and stemmed tweets that have already been analyzed. Each tweet is first tokenized into words or tokens, and then each word is then converted to its root form using a Porter stemmer, such as outraged to insult.

**C. Data Splitting**

Table II displays the whole dataset's class-wise distribution as well as the data set following splitting (i.e. Training set and Test set). The preprocessed data was divided according to the 80-20 rule (i.e., 80% for training data and 20% for test data). The classification model is trained using the training data in order to learn the classification rules. The categorization model is also evaluated using the test data.

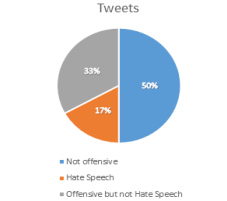


Fig 2. Class wise Data Distribution

**D. Equations**

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 testset. 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 = 𝑇𝑃/(𝑇𝑃+𝐹𝑃) (1)

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

𝑅𝑒𝑐𝑎𝑙𝑙 = 𝑇𝑃/(𝑇𝑃+𝐹𝑁) (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.

𝐹 − 𝑚𝑒𝑎𝑠𝑢𝑟𝑒 = 2 ×(𝑝𝑟𝑒𝑐𝑖𝑠𝑖𝑜𝑛 ×𝑟𝑒𝑐𝑎𝑙𝑙)/(𝑝𝑟𝑒𝑐𝑖𝑠𝑖𝑜𝑛+𝑟𝑒𝑐𝑎𝑙𝑙 ) (3)

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

𝐴𝑐𝑐𝑢𝑟𝑎𝑐𝑦 = (𝑇𝑃+𝑇𝑁)/𝑇𝑃+𝐹𝑃+𝑇𝑁+𝐹𝑁 (4)

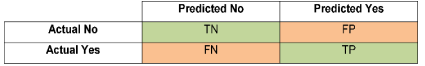


Fig 3: Confusion Matrix

**E. Mistake or Challenges of Detecting**

The problem of automatically identifying hostile and/or harmful communication, particularly on social media, has multiple dimensions. Some of these challenges are directly connected to the drawbacks of keyword-based techniques. For example, words can be obscured in a variety of ways, either intentionally to circumvent automatic content moderation [60] or as a result of the usage of social media for communication (see the trend in certain messages to substitute letters with similar appearing numerals).

Furthermore, many idioms are not intrinsically offensive, but they can be in the correct context. However, even in the case of slurs, not only can various slurs have varying degrees of offense, but the offense may also change based on time (as formerly harmless phrases can become slurs over time), as well as varied usage of the same term, different users, and different audience members. One example is the difference in slur usage between in-group and out-group speakers. When this element is ignored, it can lead to bias in hate speech detection corpora, and hence bias in hate speech detection.

Another example of subjectivity in the annotation of abusive or obscene speech is the categorization of tweets using the word f-word. Specifically, the difference in labeling when the term is used as part of a hashtag vs when it is used outside of a hashtag. This means that, while the usage of the "f-word" in and of itself greatly increases the likelihood of a tweet being tagged as hostile or offensive, tweets containing it in a hashtag are just slightly more likely than any other tweet to be branded as such.

**VI. DISCUSSION**

Decision Tree is a Supervised learning approach that may be used for both classification and regression issues, however it is most commonly employed for classification. It is a tree-structured classifier in which internal nodes contain dataset attributes, branches represent decision rules, and each leaf node represents the result. A Decision tree has two nodes: the Decision Node and the Leaf Node. Decision nodes are used to make decisions and have numerous branches, whereas Leaf nodes represent the results of those decisions and do not have any more branches. The judgments or tests are based on the characteristics of the provided dataset. It is a graphical depiction of all possible solutions to a problem/decision given certain parameters. It is named a decision tree because, like a tree, it begins with the root node and then branches out to form a tree-like structure. The CART algorithm, which stands for Classification and Regression Tree algorithm, is used to form a tree. A decision tree simply asks a question and divides the tree into subtrees based on the answer (Yes/No). The figure below depicts the general structure of a decision tree:



Because there are several algorithms in machine learning, the essential thing to remember when developing a machine learning model is to select the appropriate method for the given dataset and issue. The two arguments for utilizing the Decision tree are as follows:

* Decision trees are designed to mirror human decision-making abilities, making them simple to grasp.
* Because the decision tree has a tree-like form, the rationale behind it is simply comprehended.

In a decision tree, the algorithm begins at the root node and works its way up to predict the class of a given dataset. This algorithm checks the values of the root property with the values of the record (actual dataset) attribute and then follows the branch and jumps to the next node depending on the comparison.

The algorithm checks the attribute value with the other sub-nodes and moves on to the next node. It repeats the procedure until it reaches the tree's leaf node. The following algorithm will help you better understand the entire process:

Step 1: Begin the tree with the root node, which includes the whole dataset,

Step 2: Using the Attribute Selection Measure, find the best attribute in the dataset (ASM).

Step 3: Subdivide the S into subsets containing potential values for the best qualities.

Step 4: Create the decision tree node with the best attribute.

Step 5: Create new decision trees recursively using the subsets of the dataset obtained in step 3. Continue this procedure until you reach a point where you can no longer categorize the nodes and refer to the last node as a leaf node.

Python was used to create the Decision tree. We utilized the dataset "twitter.csv" from prior categorization models for this. We may compare the Decision tree classifier against other classification models using the same dataset, such as KNN SVM, Logistic Regression, and so on.

The following steps will also stay unchanged:

* Data Preparation step
* Adapting a Decision-Tree method to the Training dataset
* Predicting the outcome of a test
* The result's accuracy was tested (Creation of Confusion matrix)
* Visualizing the outcome of the test set.

**V.CONCLUSION**

Warner and Hirschberg were among the first projects to automate the identification of hate speech on the World Wide Web from the beginning. Their study focused on detecting anti-Semitism, and their work encompassed a variety of problems that should have been addressed by now. Kwok and Wang were the first to do racial research, following in the footsteps of Warner and Hirschberg and developing a supervised algorithm to detect racist tweets. They created a supervised classifier to differentiate between hateful and non-hateful tweets. This study served as a foundation for many subsequent researchers that examined predictive variables for hate speech identification. They granted access to their massive corpus of 16K tweets dedicated to hate speech study in English.

To detect hate speech communications, this study used automated text categorization approaches. Furthermore, three feature engineering strategies were compared in this study. Furthermore, we employed a Decision tree classifier, which was efficient enough to provide us with a decent accuracy result of close to 98%. The project's results are important because they will be used as a baseline study to evaluate future experiments using different automatic text categorization algorithms for automatic hate speech identification. Furthermore, this work is scientifically valuable since it gives experimental data in the form of many scientific measures utilized for automatic text classification. Our research has two significant drawbacks. First, in terms of real-time prediction accuracy for the data, the suggested ML model is inefficient. Finally, it merely categorizes hate speech messages into three categories and is incapable of determining the intensity of the message. As a result, the goal for the future is to develop the presented ML model, which can also be used to forecast the intensity of the hate speech message. Furthermore, two techniques will be applied to improve the classification performance of the suggested model. To begin, the lexicon-based methodologies will be investigated and evaluated by comparing them to other existing state-of-the-art outcomes. Second, more data examples will be collected and used to efficiently learn categorization rules.

**VI. REFERENCE**

Auto matica Sinica, 2013. 39(6): p. 745-758.

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

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

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

[4] 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.

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

[6] Greevy, E. and A.F. Smeaton. Classifying racist texts using a supportvector machine. in Proceedings of the 27th annual international ACMSIGIR conference on Research and development in information