International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Improvement of Accuracy for Hate Speech Detection Using Modified Feature Extraction

Ishan Bansal[1], Mehak Sood[2]

[1,2]Student Researcher, Cluster Innovation Centre, University of Delhi, Delhi 110007

Abstract

The proliferation of toxic online content has become a significant concern in today's digital landscape, fueled by the widespread use of the internet among individuals from diverse cultural and educational backgrounds. One of the central challenges in the automated identification of harmful text content lies in distinguishing hate speech from offensive language. In this research paper, we undertake a comprehensive examination of two primary modeling approaches for hate speech detection. Leveraging the Twitter dataset, we conduct experiments that involve the utilization of n-grams as distinctive features, subsequently subjecting their term frequency-inverse document frequency (TFIDF) values to various machine learning models. A comparative analysis is conducted across 5 models among which, Logistic Regression and Gradient Boosting produce the best results.

Keywords: Twitter, Hate Speech, Feature Extraction, Logistic Regression, Naive Bayes, Random Forest, Decision Tree, Gradient Boosting

1. Introduction

Social Media platforms such as Twitter, Facebook, and Instagram have become widely popular among the digital population, transcending age, ethnicity, and interests. The exponential growth in content on these platforms exemplifies the vastness of big data. Researchers seeking insights into public opinions, user sentiments, and interests have gravitated toward this rich source of big data. Although these websites provide a public forum for people to express their ideas and beliefs, it is nearly impossible to police the type of content that is posted.

Exploiting this dynamic, individuals from diverse backgrounds and cultures often resort to aggressive and hateful language. In today's world, marked by the expansion of online social networks and escalating global conflicts, the issue of content censorship remains contentious, dividing opinions into two camps: supporters and opponents. It becomes even easier to propagate such trends, especially among younger generations, compared to more benign forms of expression. In light of these challenges, Burnap and Williams [1] contend that collecting and analyzing temporal data enables decision-makers to study the surge in hate crimes following "trigger" events. However, official records of such events are often sparse, as hate crimes frequently go unreported to law enforcement. Social networks, within this context, offer a more comprehensive but less reliable source of information, marred by noise.

To mitigate the noise and data unreliability, an efficient approach to detect both hateful and offensive posts within social network data is imperative. The first step in this process is to define "toxic language," which we broadly categorize into two distinct types: hate speech and offensive language. According to Wikipedia, "hate speech" is characterized as "any form of expression that targets individuals or groups

based on attributes such as beliefs, ethnic background, sexuality, gender identity, or disability." Offensive language encompasses text containing abusive slurs or derogatory expressions. Given the impracticality of manually filtering hateful tweets at scale, researchers have sought to identify automated alternatives. Most of the earlier work revolves either around manual feature extraction or the use of symbolic learning methods, which are then followed by a linear classifier.

In this paper, we use 2 experimental techniques for the classification of a tweet as hateful, offensive or neutral. It is a challenging task due to the inherent complexity of natural language constructs. Hatred takes on various forms, targeting diverse subjects, and can be articulated in multiple ways. We approach text classification for hateful content with the following key aspects:

- A language-agnostic solution devoid of pre-trained word embeddings.
- An empirical evaluation of the model's performance using a Twitter dataset for classification.

Our proposed solution leverages the Twitter dataset to train our classifier model, employing Term Frequency Inverse Document Frequency (TF-IDF) for feature extraction. Our findings demonstrate that, following hyperparameter tuning of TF-IDF features, Gaussian Naive Bayes emerges as the top-performing model.

The subsequent sections of this paper are organized as follows: Section 2 provides an overview of recent background work. Section 3 delineates our proposed methodology, while Sections 4 and 5 offer an in-depth exposition of the experiments and results. Finally, the paper concludes with insights into future research directions in Section 6.

2. Existing Research

Current methods map the problem to supervised document classification, which can be mainly resolved into 2 categories. The first category employs manual feature engineering which is then followed by standard classification algorithms such as Logistic Regression, Gaussian Naive Bayes and SVM. The second category employs deep learning models which use neural networks to automatically derive features from raw data.

In order to conduct the classification job into two classes, Nobata et al. [5] employed syntactic, linguistic, n-gram, lexical, and pre-trained "comment2vec" and "word2vec" features and achieved an accuracy of 90. Other works have been carried out to identify hate text on Twitter. Kwok and Wang [6] used unigram features for the identification and binary classification of hateful racist tweets, which resulted in 76 percent accuracy. The hate speech was focused on a niche sexuality, ethnicity or race which related the collected unigrams to that specific niche group. Thus, this unigram glossary presented bias and cannot be employed to identify hate speech directed at other groups as reliably. Watanabe and Ohtsuki [12] distinguished hate speech using the relationship between words as well as "bag of words" (BoW) features.

With the emergence of hate speech and offensive language datasets, numerous studies have touched upon cross-dataset generalization since 2018. Grondahl et al. [7] trained a range of models, cross-applying them to four datasets. The models included LSTM—one of the most popular neural networks in text classification—and CNN-GRU (Zhang et al. [8]), which outperformed previous models on six datasets [9]. Grondahl et al.'s [7] experiments implied that using character-level features instead of word-level features would make the models systematically less prone to attacks. Adversarial training helps only to a certain extent for making the attacks less severe.

Current challenges in existing research are as follows:

1. It is difficult to analyze the semantic meaning of hate speech texts as typical datasets suggest that the language constructs are devoid of unique and discriminate features.
2. Due to the nature of the tweets, certain words are out-of-vocabulary (OOV) which are not encapsulated by the pre-trained embeddings. These words are excluded in the preprocessing to reduce the noise in the language.
3. Although tweets rarely contain two full sentences, splitting long tweets into two has been proven to cause a loss of linguistic information; hence, we will avoid doing so during the preprocessing phase.
4. Training based on domain-specific data is expected to increase performance on tasks like hate speech detection. However, the results from previous research showed that there were not any improvements, and it did not prove to show huge improvements in capturing features. Hence, we will not be training our model on domain-specific corpora.

3. Techniques Used

3.1 Feature Extraction

Initially, we preprocessed the dataset, making it cleaner for our experiment. We studied the bag of words model, tried experimenting with CountVectorizer but eventually used TF-IDF Vectorizer. N-gram was used to extract features from the input text and weigh them using TF-IDF. We then feed these features to different classification heads to compare their performances.

Let's first understand the logic behind TF-IDF. TF-IDF is an abbreviation for Term Frequency - Document Inverse Frequency. For a document "D", the number of times of a particular term "T" occurs is its frequency. Thus, the number of times a term occurs in a text defines its relevance, which is rational. We can assign a vector for describing the phrase in the bag of term models because the overall sequencing of the terms in the phrase is not relevant. For every term in the document, we calculate the term frequency. The weight of the term is proportional to the term's frequency.

$$TF(T, D) = \frac{Count\ (T)}{Total\ Number\ of\ Words\ in\ D} \qquad (1)$$

Coming to the Document Inverse Frequency, it is first empirical to understand the term Document Frequency. Document Frequency is similar to Term Frequency, but instead of finding out the frequency of a term in a given document, here we find the frequency of the term in the whole corpus collection. In mathematical formulation, the number of documents containing a particular term is the document frequency of that term.

$$DF(T, N) = Count\ (N) \in T\ Exists \qquad (2)$$

Therefore, Inverse Document Frequency (IDF) helps in identifying how relevant a word is. Term Frequency advises all words as equally significant and hence they can be used to study the weight of the term. IDF is based on the principle that a common word such as "the" are to be considered as "less significant". It is defined for a term as the number of documents in the corpus divided by the document frequency.

$$IDF(T,N) = \frac{N}{DF(T,N)} \quad (3)$$

We usually take the logarithm (base 2) as it dampens the effect of IDF and avoids harsh integers.

$$IDF(T,N) = \log_2\left(\frac{N}{DF(T,N)}\right) \quad (4)$$

We combined all the clean tweets into one giant corpus. then weighted each element in the corpus by its TF-IDF. We also tried a variation for feature extraction wherein, instead of extracting features directly from the corpus, we identify the hate level of each word present in the corpus and weigh TF-IDF against those features.

Below are 2 figures of the most frequently occurring words obtained from our dataset, classified as hate or non-hate, generated using the WordCloud library in Python:

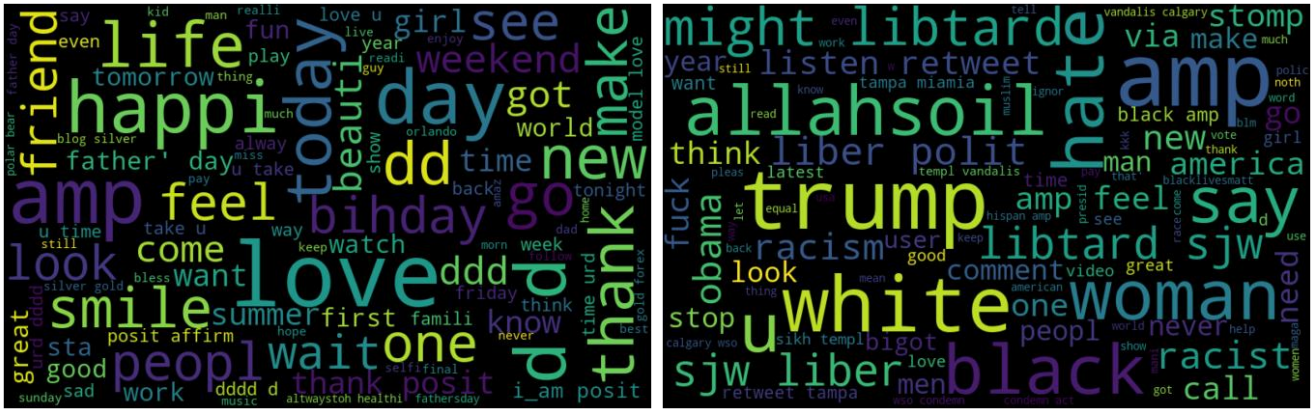

Figure 1: Non-hate words (left). Hate words (right). We observe that the non-hate words obtained indicate optimism while hate words represent societal bias based on gender, race and color.

3.2 Algorithms

We take a brief look at the 5 algorithms whose performances will be compared in the paper.

Table 1: Algorithms Overview.

Logistic Regression	Naïve Bayes	Decision Tree	Gradient Boosting	Random Forest
It is a linear model primarily used for classification rather than regression	Based on Bayes theorem, and naively assumes that every pair of features is conditionally independent for a given class variable.	Predicts target by learning simple decision rules from data features.	Makes it possible to optimize arbitrary differential loss functions.	Utilizes a randomly chosen portion of the training set to generate an assortment of decision trees.

4. Methodology

4.1 Cleaning the Dataset

Our dataset contains 31962. After running a quick Pandas profile, we got the following results: the "label" column represents the categorization of tweets. 0 represents a clean tweet, whereas 1 represents a hateful tweet. The ratio of hateful tweets to not-hateful tweets is 2242:29720, or approximately 1:13, which means every 1 in 14 tweets is offensive. The dataset structure is as follows:

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

Figure 2: Raw Dataset.

The Pandas Profiling report can be seen in Figure 3, which provides with us detailed statistics about the dataset.

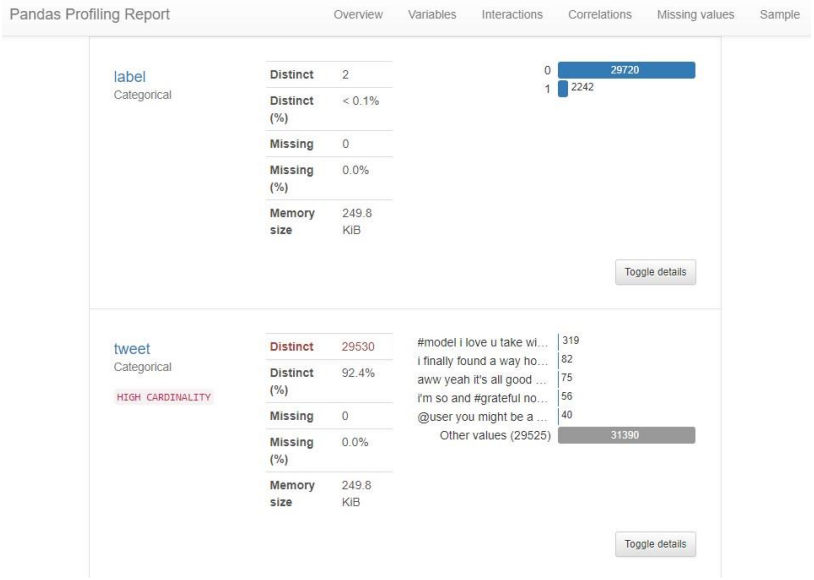

Figure 3: Pandas Profiling Report.

Table 2: Tweet Column Overview.

Length		Characters and Unicode		Unique	
Max Length	274	Total Characters	2708448	Unique	28836
Median Length	136	Distinct Characters	163	Unique (%)	90.2
Mean Length	84.73962831	Distinct Categories	19		
Min Length	11	Distinct Scripts	2		
		Distinct Blocks	2		

We processed our dataset by removing "@" tags and "words" and making the sentences all lowercase to achieve our final dataset (Figure 4).

	label	clean_tweet_final
0	0	when a father is dysfunctional and is so selfi...
1	0	thanks for credit i can't use cause they don't...
2	0	bihday your majesty
3	0	i love u take with u all the time in urd+-!!! ...
4	0	factsguide: society now

Figure 4: Final Clean Tweet Dataset.

4.2 Experimental Setup

We used the scikit-learn library in Python for training and experimentation. We used Jupyter Lab to execute our experiments.

4.3 Modifying Feature Extraction

We already know that TFIDF is used for feature extraction from a document. Now, let us apply TF-IDF Vectorizer on the text in 2 different ways, by modifying the underlying feature extraction technique.

4.3.1 Experiment A

Presented below is the corpus of tweets used for feature extraction using TF-IDF.

```
['homicides rose in most big cities this year - the wall street journal', "we might be going to hell, but we're not surprised
at", 'reject news - my latest aicle on', "', ugly' comments on the embarrass new yorkers", 'you forgot because vetting people
from terrorist is .dumb', '(advanced value chain videos at )', '+ young + - relevant read for all women in journalism', 'both
&amp;', 'you might be a libtard if...', 'those replies. is dead tho. d', 'girls in the world nude yong girl', 'finally gets s
upremely called out on via', "no it doesn't, germans, even fascists rape too. so how should help?", 'the "monster law" made b
y republican lawmakers that disenfranchised black voters "with near surgical precision.a|', 'no,we can see they have been ins
trumental in promoting false narrative &amp; abetting divisive blm agenda.', 'women nude anal anal t girls', 'how liberal ins
titutionalized white supremacy', 'runs his mouth endlessly like both have lost. and yet, ppl only take glee in her losing. pe
rhaps?', '(advanced value chain videos at )', "this's insane acts by they will trample national flag of other nation. it's cl
early .", 'girls in the world sexy mature housewife', 'sad how people feel they have to treat others like scum just because o
f their religion, race, sexuality, ect.', 'leader reveals 5 of his most horrifying hopes for america', 'you might be a libtar
d if...', 'the for', '10 hours of walking in as a woman', 'yup. i concur. as do nearly 3 million voters.', "aren't protesting
because a won-they do so because trump has fuhered &amp;a|", "aren't protesting because a won-they do so because trump has fu
hered &amp;a|", "why can't women be defenders if they want?", 'boob camron diaz naked pics', 'why the nazis studied american
race laws for inspiration a', 'read and understand how berniebros chose sanders over hillary', 'hitler called...', 'american
taxpayers have also lost their homes by the millions a| a|', "i couldn't end without mentioning", 'if you are pa of the 64.2
mil who rejected his business. &amp; must be stoppea|', 'black professor makes assumptions about an entire race whilst speaki
ng for entire race. next week the jews!a|', '&amp; the keeps telling me that but i see &amp; every other in', 'the problem is
that thereas no space for alternative explanations. a|', '&amp; joseon people in japan, will abuse the for claims of own righ
```

Figure 5: Corpus Head.

We directly extracted features from the above corpus using TF-IDF. After feature extraction, we obtained the feature dataframe. We split the feature dataframe into 2 categories: hate features with label = 1 and non-hate features with label = 0. After splitting the features, our training set contained 90 percent of all the hate features and 40 percent of all the non-hate features. We kept the proportions as such so that the train set overall contained a balanced proportion of hate to non-hate tweets.

4.3.2 Experiment B

Instead of directly extracting features from the corpus, we further split the corpus into words. Then, using a counter, we obtained the 1000 most frequently occurring words in our dataset. Now, we assign a hate level to each of these words. Since each tweet was classified using labels, that classification lost its semantic sense when we split up the tweets into words.

For example, say the word "the" occurs in a hate tweet with label = 1, and it also occurs in a non-hate tweet with label = 0. To decide whether the word "the" can be classified as hate or not, we bring up the concept of hate level.

Figure 6: Head of the top 1000 most frequently occurring words.

Each word was assigned a hate level relative to the maximum hate level possible. So, for example, if a word occurs in 100 tweets, with 80 of them being hate tweets, the word will be classified as having a hate level greater than 0.5. We assigned 0.5 as the breaking point. Any word having a hate level greater than 0.5 will be classified as a hate word, and any word having a hate level of 0.5 or less will be classified as a non-hate word. Based on the hate levels assigned to the words, we extracted features using TF-IDF.

5. Results

5.1 Experiment A

We obtained the following ROC scores:

Table 3: Results from Experiment A.

Classifier	Accuracy Score	ROC AUC Score
Logistic Regression	0.804962311557789	0.785750482625482
Gaussian NB	0.547110552763819	0.667702895752896
Random Forest	0.759108040201005	0.754898648648648
Decision Tree	0.668341708542736	0.691638513513513
Gradient Boosting	0.791771356783913	0.710569498069498

We plot the ROC Curve for the above algorithms using their ROC scores. Receiving Operating Characteristic Curve is a graph that displays performance of a classification model. AUC represents area

under the ROC curve which measures performance across all potential classification thresholds. Algorithms whose curves more closely represents an inverted L shape graph give better performance. As the curve approaches the y=x line, the test accuracy decreases.

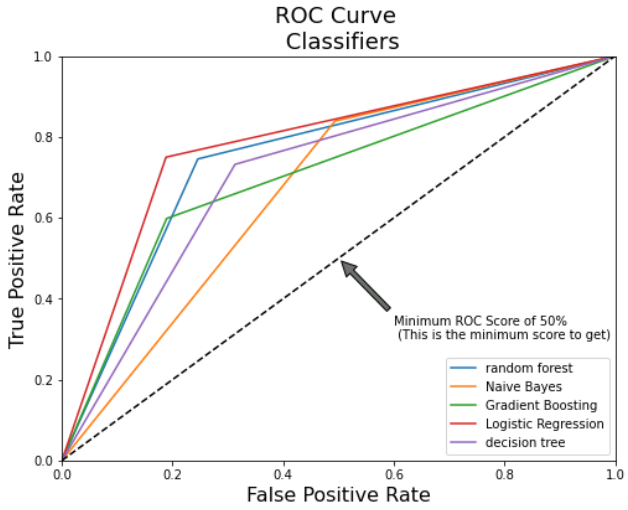

Figure 7: ROC Curve for Experiment A.

As we can see from both the ROC scores and ROC curves, logistic regression produces the best results for hate speech detection when the features are derived from the corpus as a whole, consisting of phrases and sentences.

5.2 Experiment B

We obtained the following ROC scores:

Table 4: Results from Experiment B.

Classifier	Accuracy Score	ROC AUC Score
Logistic Regression	0.7418341708542714	0.785750482625482
Gaussian NB	0.8709170854271356	0.839792471042471
Random Forest	0.7478015075376885	0.717869208494208
Decision Tree	0.8011934673366834	0.831177606177606
Gradient Boosting	0.8998115577889447	0.837400772200722

Using the above ROC scores, we plot the ROC curve for the classifiers.

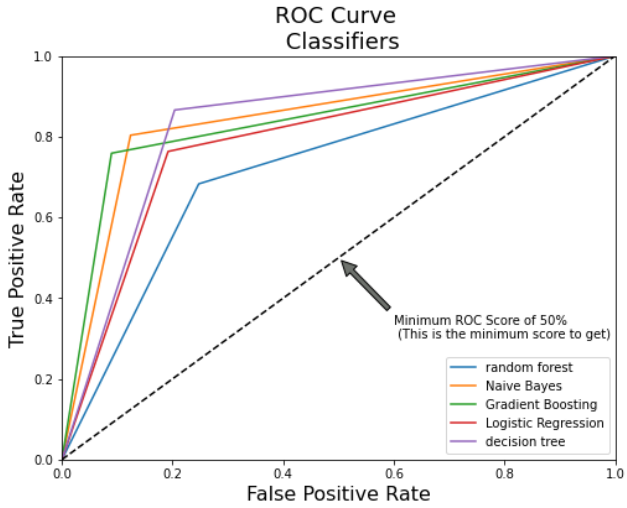

Figure 8: ROC Curve for Experiment B.

As we can see from both the ROC scores and ROC curves, Gradient Boosting produces the best results for hate speech detection when the features are derived directly from individual words in the corpus.

6. Discussion

Existing hate speech detection models perform adequately on new, previously unseen datasets, with their accuracy hovering between 70 to 80 percent. This is due to the limitations of existing NLP methods, the difficulty of constructing datasets, and the nature of online hate speech, which are frequently interrelated. The behavior of social media users, and particularly haters, poses an added challenge to existing NLP approaches. Feature engineering and the extraction of a dataset play a major role in determining the final outcome.

In the future, we plan to experiment with state-of-the-art deep learning architectures like LSTM and GRU to increase accuracy. We also intend to use larger datasets from across the web to train and test the classifiers, as well as to investigate the rise and fall of cyber-hatred on online social media platforms.

7. References

1. Burnap, P. Burnap and M. L. Williams (2015), Cyber Hate Speech on Twitter: An Application of Machine Classification and Statistical Modeling for Policy and Decision Making, Policy & Internet, 7: 223-242. https://doi.org/10.1002/poi3.85
2. W. Warner and J. Hirschberg (2012), Detecting Hate Speech on the World Wide Web, In Proceedings of the Second Workshop on Language in Social Media, pages 19–26, Montréal, Canada, Association for Computational Linguistics.
3. N. Djuric, J. Zhou, R. Morris, M. Grbovic, V. Radosavljevic, and N. Bhamidipati (2015), Hate Speech Detection with Comment Embeddings, In Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion), Association for Computing Machinery, New York, NY, USA, 29–30. https://doi.org/10.1145/2740908.2742760
4. N. D. Gitari, Z. Zuping, H. Damien, and J. Long (2015), A Lexicon-based Approach for Hate Speech Detection, International Journal of Multimedia and Ubiquitous Engineering, 10, 215-230. https://doi.org/10.14257/ijmue.2015.10.4.21

5. C. Nobata, J. Tetreault, A. Thomas, Y. Mehdad, and Y. Chang (2016), Abusive Language Detection in Online User Content, In Proceedings of the 25th International Conference on World Wide Web (WWW '16), International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE, 145–153. https://doi.org/10.1145/2872427.2883062
6. Kwok, I., & Wang, Y. (2013), Locate the Hate: Detecting Tweets against Blacks, Proceedings of the AAAI Conference on Artificial Intelligence, 27(1), 1621-1622. https://doi.org/10.1609/aaai.v27i1.8539
7. Grondahl, T., Pajola, L., Juuti, M., Conti, M., and Asokan, N. (2018), All You Need is "Love": Evading Hate Speech Detection, In Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security (AISec '18). Association for Computing Machinery, New York, NY, USA, 2–12. https://doi.org/10.1145/3270101.3270103
8. Zhang, Z., Robinson, D. and Tepper, J. (2018), Detecting hate speech on Twitter using a convolution-GRU based deep neural network, The semantic web, ESWC 2018, 03-07 Jun 2018, Heraklion, Greece. Lecture Notes in Computer Science, 10843, Springer Verlag, pp. 745-760, ISBN 978-3-319-93417-4. https://doi.org/10.1007/978-3-319-93417-4_48
9. Yin, W., & Zubiaga, A. (2021), Towards generalisable hate speech detection: A review on obstacles and solutions, ArXiv.