2022 Third International Conference on Intelligent Computing Instrumentation and Control Technologies (ICICICT) | 978-1-6654-1005-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICICICT54557.2022.9917752

Comparative Analysis of Different Machine Learning Methods for Hate Speech Recognition in Twitter Text Data

Abhay Tiwari
Department of Information Technology,
Indian Institute of Information Technology Allahabad,
Prayagraj, India
abhaytiwari455@gmail.com

Anupam Agrawal
Department of Information Technology,
Indian Institute of Information Technology Allahabad,
Prayagraj, India
anupam@iiita.ac.in

Abstract— **The purpose of this work is to solve the challenges faced by us in the field of hate speech recognition on various social media platforms, that is to get a better machine learning model that can detect hate speech with greater accuracy. As the reach of the internet and mobile phones has extended abruptly in the past few years, everyone has the power to share their opinions, but some use it as an opportunity to spread hate among one another. In this paper, we used the Davidson [10] dataset which is the most popular Twitter dataset for hate speech detection, and further we implemented various machine learning-based algorithms and compared them on the basis of various parameters such as accuracy, precision score, recall and F1 scores. After the study, we found out that XGBoost when used with TF-IDF transformer embedding gave us an accuracy of 94.43%, which is the maximum among these three models for the given benchmark dataset.**

Keywords— hate speech recognition, offensive Tweets, social media, twitter dataset, LSTM, XGBoost, multinomial Naïve Bayes.

I. INTRODUCTION

The study of human activities is an active research area, and the fast growth of digitalization and social networking websites has increased interaction between people from different psychological and cultural backgrounds. Hence, people have come closer to one another, which has increased "digital" conflicts between them. Consequently, hate speech is now the hottest topic in the field of NLP for researchers and the government. Social media companies are investing a huge amount of money in filtering negative or hateful contents to keep the platform safe for users. Therefore, to find a solution and to come up with a better approach, There is no scale defined to differentiate between hateful and offensive speech, nor is there any formal definition, but for the sake of simplicity, we can simply describe any group of words or sentences that hurt the feelings of any individual or group and cause them to harm in any form, be it emotionally or socially. Almost every country and community has defined certain rules and regulations in their beliefs, and everyone should act under those rules. And violating any of those rules may cause them huge fines or even send them to jail. Those rules and laws also extend to social media platforms and even those platforms, provide certain rules of action to prevent hate speech. Twitter along with Facebook has faced huge criticism for not doing enough to ensure the feelings of the users and communities are not hurt by any means. They are even forced to face the approved laws related to common people and their differences, like race, colour, gender, sexual orientation, etc. Based on the above discussion, hate speech is defined as any language or gestures in physical form that can deliver hatred toward a targeted group of people to humiliate and insult its members. It may get even worse in extreme scenarios and can even lead to violence, but for the sake of this paper, we will be restricting our discussion only to hate speech. It should be noted that cultural diversity should also be taken care of, because something offensive to one community may not be considered offensive in another. Like the word teenagers use in the everyday conversation cannot be posted on Facebook and Twitter. This term was considered in most of the previous works as well but there was still a lot of confusion and miscommunication. Due to such vast networks, it is impossible to detect and classify each and every piece of content on the internet. That's why we need automated methods to detect hate speech.

We have used the Davidson dataset [10] because each tweet in this dataset is manually classified as "hate", "offensive" or "neither" by the author, so it is pretty much accurate. The main player and opponent in the way of automatic hate speech detection on social media and other communication related platforms are to separate hate speech from other families of impressions and wordings nearly belonging to the same categories and hierarchies. We implemented three algorithms named LSTM, Naïve Bayes, and XGBoost. The latter model outperforms the other two models and gives the best results among the models. In the next part of this paper, we will discuss the related work that has been done in this field by many authors, followed by the details of the dataset we have used in this paper. In section four, we will discuss the models we implemented, followed by the results obtained by these models and a comparative study of these models on different parameters. In the last part, we will conclude our work with some possible future scope in this field.

II. RELATED WORK

Hate speech detection is one of the main challenges for social media giants like Reddit, Facebook and Twitter [1]. They have invested billions to keep their network free from hatred and offensive speech. Twitter uses a real time mechanism to filter hate speech, and they have built a dashboard that keeps checking every tweet which is posted online, and it flags or takes down any tweet depending on the severity of the tweet [2]. Facebook also has a state of the art advanced filtering system for classifying hate speech. Facebook has its own machine learning system that detects hate or offensive content in real time and takes down the content. Several deep learning methods are also used to cope with this challenge and with Recurrent Neural networks engineers and researchers have done a tremendous job [3]. Apart from this hate speech detection is only built for English but it is also built for many languages like Hindi, French, Spanish, Arabic and many people have contributed to this challenge [4]. There is a multilingual model for the detection of hate speech. This field is still advancing, and researchers are trying to get more accurate and robust models to detect hate speech.

III. Dataset

The Davidson dataset [10], also known as the hate-speech-and-offensive-language dataset, is a huge collection of words, phrases, and sentences that are considered and identified as hate speech by internet users, compiled by Hatebase.org. They collected a huge sample of tweets from 33,458 users using the Twitter API and extracted the timeline further, which resulted in 86 million tweets. The author than, took a sample of 25 thousand tweets and make them coded through CrowdFlower (CF) workers (Paid service provider available on http://faircrowd.work/platform/crowdflower/, in this dataset they categorised the tweets). Those tweets were then divided into three categories: “0” for Hate speech, “1” for offensive language, and “2” for neither we will use only two classes that are hate and non-hate, offensive language will be considered as hate. As per the basic definitions of each class, tweets were cross-checked while categorizing. Similar work was done by the Waseem dataset [11], but it has only 15k tweets, which is not ideal for these models, so we continued with the Davidson dataset. The data is in the csv format and is of 4 MB and data contain the following important columns:

Column Count: The count is the number of people who analyzed that tweet.

Column Hate speech: Number of users who think this is hate speech as per its basic definitions.

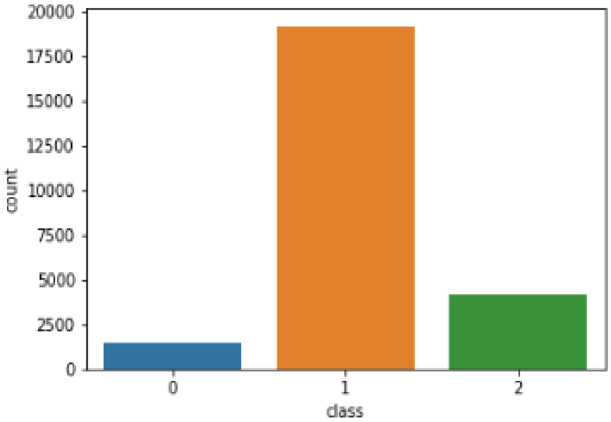
Column Offensive Language: Number of users who think this is offensive speech as per its basic definitions.

Column Neither: Number of users who think this is neither hate nor offensive speech as per its basic definition.

Column Class: It contains the class number of the class that has the maximum votes.

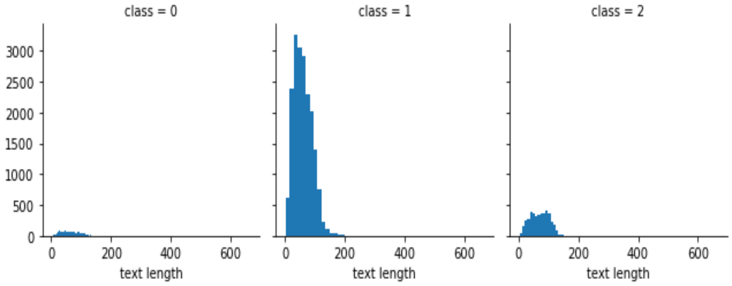
Column Tweet: Column tweet is a column that contains the actual tweet.

We plotted different classes 0,1 and 2 on a graph to see number of tweets in each class as shown in Figure 1 to see if there is any imbalance in the datset.



Graph: 1
Figure 1: Tweet count of different classes

As we can see, there is a huge imbalance in the dataset between class 0 that contains hate words and class 1 that contains offensive words. In the real world also, we will get similar types of problems, as very small portion of the tweets are hate speech.



Graph: 2
Figure 2: Tweet length of different class of tweets

The length of the tweets was around 0 to 200 in all the three classes as shown in Figure 2. Where, y-axis represents number of tweets. This is a good length and will be used while tokenizing the words from the tweets.

IV. Methodology

The conceptual schema/algorithm of our model is pre-processing contains the four units, detailed in this section and the flowchart is shown in Figure 3.

Data Cleaning:

In this phase, we have cleaned the data. In particular, we have cleaned the tweet column and, in the cleaning, we have removed the URLs, hashtags, or any unwanted characters as they do not have any meaning in the sentences, and as we can observe, many people like to put extra symbols and words in the text. We have also used NLP techniques like stemming. We used the NLTK library to construct POS tag unigram, bigram, and trigram in order to extract the information about the syntactic structure.

Stop Words:

We have also removed the stop words using the NLTK library. Stop words are those words that are not important in our work as they are just there for grammatical purposes and carry no useful information.

Stemming:

Stemming [8] is an NLP based technique that converts words to their root forms. In laymen's terms, we can say that as there are many forms of a word which can be used but actually, they are pointing to the same thing, so we change all of them to their base form. Both in stemming and in lemmatization, we try to reduce a given word to its root word. Both the methods just have a slight difference which is irrelevant in our dataset because of its size, so we used stemming.

Model Building:

After analysis, we have done tokenization on the data which means essentially reducing a tweet into smaller units and we have also done pad sequence i.e. making all sequence of same length. After that we split the data into train and test for model training. Furthermore, to implement these three models, we will import the corresponding libraries of these models.

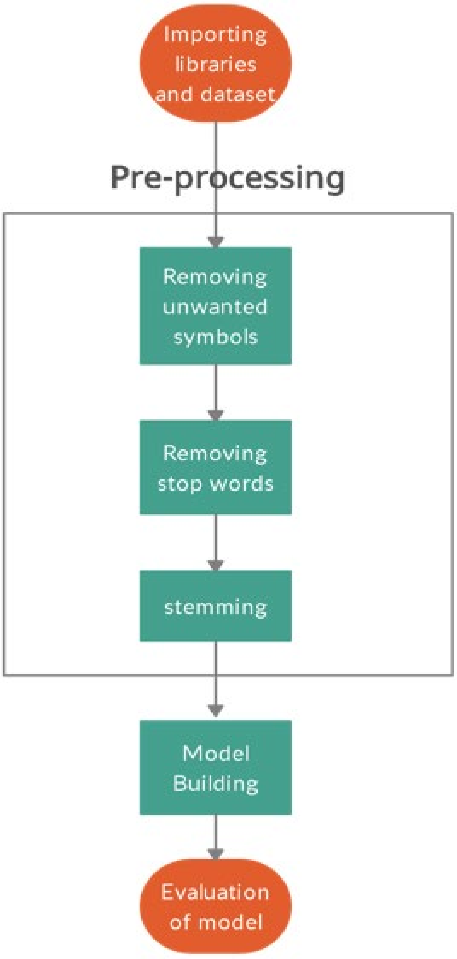


Figure 3: Flow chart of proposed methodology

Model Training:

LSTM:

We have trained our model on LTSM which stands for long short-term memory. These are recurrent neural network and can be termed as the pioneer of analysing order dependence in the problems involving sequence prediction. This model can be used to cope with the challenges like machine translation and speech recognition. The overall work was a bit complex especially in relating the areas like bi-directional sequence to sequence to the respective fields. LSTM [7] it is capable of analysing the sentence as a whole rather than just a word or phrase, which makes it more efficient. The model diagram is shown in Figure 4 where x(i) represents input and h(i) represents output. The output in Figure 4 of the previous state is taken as an input to next state Which makes this model understand the whole sentence in the context of the preceding words also.

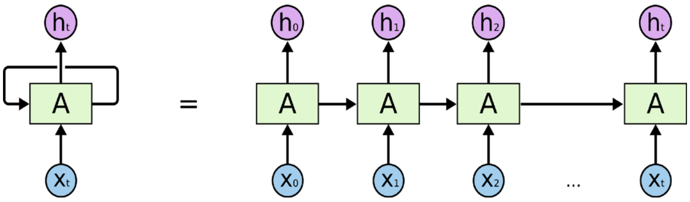


Figure:4 LSTM Model used in our work [7]

We have used TensorFlow keras API's for model training, and we have used RNN based approach to train the model. Model was trained with one Embedding Layer, Dropout Layer, and LTSM Dense Layer.

Table 1: LSTM hyper parameters after tuning

Parameters	**Values**
Batch Size	64
Loss function	categorical_crossentropy
Optimizer	rmsprop
Activation function	Sigmoid
Epoch	6
Dropout	0.1

Naïve Bayes:

Naive Bayes (Multinomial) [5] algorithm is a supervised learning algorithm. We have used TF-IDF word embeddings in this model to increase its performance to make it perform better. This model is just like Bayes theorem which is used for solving classification problems and here we want to classify hate speech. The Naïve Bayes classification is faster as it can train machine learning methods at a quick speed. Hence, solve the problem of LSTM's slow speed. It is basically based on prediction, so it is more effective and the output of every state is given as the input of every state as shown in Figure 5. Variables in the box are represented for values of variable (i) is varied from 1 to n. The square nodes represent discrete variables, that are feature indicators Xij and class indicators Ci. Circular nodes represent the continuous variables, i.e. latent parameters Ψ and Θ and their hyper-parameters α and β as detailed in [12].

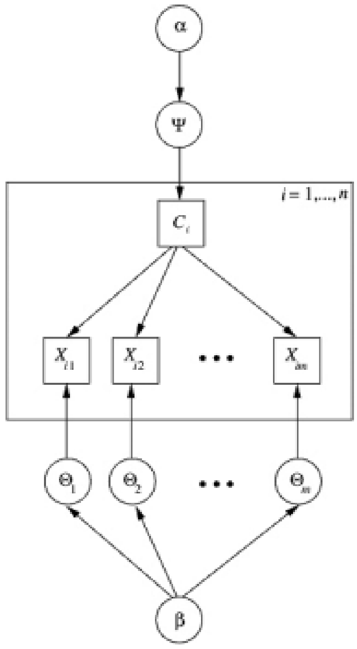


Figure 5: Naïve Bayes Train Structure [12]

For the testing purpose, we do not need many values of x and y instead we take a tweet/paragraph one by one and then test if it is containing hate or not as shown in Figure 6 the variables are same as mentioned in Figure 5.

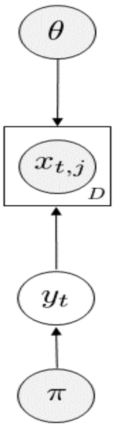


Figure 6: Naïve Bayes Test Structure [12]

Naïve Bayes is named after "Bayes" because it is based on Bayes theorem, by the help of this theorem we can find the probability of an occurrence of event, provided that some dependent event has already happened is:

$$P(R|S) = P(S|R) * P(R) / P(S)$$

Where we have to find the probability of event R such that S has already happened. In this work's context, Naïve Bayes counts the occurrences of each word and by the help of which it calculates the probability of word R given that word S has occurred in the tweet before.

XGBoost:

Our last model is XGBoost [6]. Commonly known as Extreme Gradient Boosting algorithm or model, this is a standard and scalable boosted decision tree prominently used as a machine learning library. We have used TF-IDF word embedding. It is one of the top leading machine learning libraries for ranking problems and classification, and offers parallel tree boosting. As it is based on decision trees it converts tweets into various trees, where each tree is a sentence in the tweet and the predictions are made as shown in Figure 7.

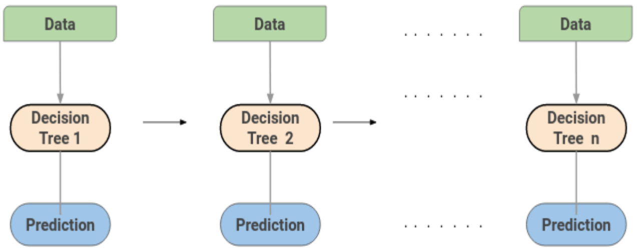


Figure 7: XGBoost Model Structure [6]

It is a very helpful tool to make things way easier for working, but one must be master of decision trees, supervised learning and ensemble learning to make the best possible outcomes out of it.

V. Results

We used different algorithms and machine learning techniques to analyse and extract the best possible results and solutions. Of all the techniques and machine/deep learning algorithms, Table 2 showed the best solutions. XGBoost appeared to be slightly better than Naïve Bayes because of its tree structure the sentences can be divided in several sentences. Whereas Naïve Bayes will consider it as a whole sentence and may lead to incorrect conclusion. XGBoost and Naïve Bayes are better than LSTM as we can see in the results table 2 because of TF-IDF embedding, and LSTM faces the issue of overfitting in big datasets. To make it more interesting, appealing and tantalizing, the results in table 2 say it all. The results of LSTM model may vary 1% due to different splitting of dataset every time.

Table 2: Results of three Machine Learning Techniques

Model	Precision	Recall	F1 score	Accuracy
LSTM	89%	72.40%	90.37%	89.39%
Naive Bayes TF-IDF	95%	94%	93%	94.33%
XGBoost TF-IDF	93%	94%	93%	94.43%

As the Naïve Bayes and XGboost methods are giving very similar results, we further investigated these methods for confusion matrix. We didn't include LSTM because it was already way behind these two methods.

Table 3: Confusion Matrix for Naive Bayes

	0	**1**
0	**7008**	**0**
1	**421**	**6**

0: Hate speech
1: Non-Hate

Table 4: Confusion Matrix for XGBoost

	0	**1**
0	**6929**	**79**
1	**335**	**92**

0: Hate speech
1: Non-Hate

From the above results shown in table 3 and table 4, we can see that XGBoost leads the metrics as there are more true positives than Naïve Bayes and less false negatives which is crucial to our system. As we cannot allow any hate speech comments to stay on the platform, which makes it better than Naïve Bayes., we are now in a state to conclude that XGBoost is better.

VI. Conclusion And Future Scope

In our work using the benchmark dataset, XGboost and Naïve Bayes appeared to be key players when it comes to precision score, recall value, F1 score and accuracy, when used with TF-IDF transformer embedding otherwise in similar embeddings LSTM is better. Although we tried hard and gave even more time and effort to all the systems and methods mentioned in the methodology section to increase the performance, the results shown by XGboost and Naïve Bayes outperformed LSTM, and overall analysis suggests that XGBoost is the best method among these.

There is a lot of work going on in this field but still, due to the rapid expansion of hate speech victims, there is always space for improvement here. We can implement more advance techniques that are based on transformers, for better results. We can further extend this work to generate an automated API that can extract text from the social media platforms and figure out the culprits behind those hateful texts. This will be a good step towards the betterment of the digital world.

References

[1] F. Del-Vigna, A. Cimino, F. Dell-Orletta, M. Petrocchi, and M. Tesconi, "Hate me, hate me not: Hate speech detection on facebook," in First Italian Conference on Cybersecurity, 2017.

[2] J. Jacobs and K. Potter, Hate crimes: Criminal law & identity politics. Oxford University Press on Demand, 1998.

[3] M. Bouazizi and T. Ohtsuki, "Multi-class sentiment analysis on twitter: Classification performance and challenges," Big Data Mining and Analytics, vol. 2, no. 3, pp. 181–194, Sep. 2019.

[4] G. Jalaja and C. Kavitha, Sentiment Analysis for Text Extracted from Twitter. In : Springer, Singapore, 2019, pp. 693–700.

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

[6] J. Han, S. Wu, and X. Liu, "Identifying and categorizing offensive language in social media," in Proceedings of the 13th International Workshop on Semantic Evaluation, 2019, pp. 652–656.

[7] P. K. Roy, A. K. Tripathy, T. K. Das and X. -Z. Gao, "A Framework for Hate Speech Detection Using Deep Convolutional Neural Network," in IEEE Access, vol. 8, pp. 204951-204962, 2020

[8] S. Al-Saqqa, A. Awajan and S. Ghoul, "Stemming Effects on Sentiment Analysis using Large Arabic Multi-Domain Resources," *2019 Sixth International Conference on Social Networks Analysis, Management and Security (SNAMS)*, 2019, pp. 211-216

[9] Kanis, J., Skorkovská, L: Comparison of Different Lemmatization Approaches through the Means of Information Retrieval Performance . In: Sojka, P., Horák, A., Kopeček, I., Pala, K. (eds) Text, Speech and Dialogue. TSD 2010. Lecture Notes in Computer Science(), vol 6231. Springer, Berlin, Heidelberg, 2010

[10] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in Proceedings of the 11th International AAAI Conference on Weblogs and Social Media, ser. ICWSM '17, 2017.

[11] Z. Waseem 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, pp. 88–93.

[12] Kohonen J, Talikota S, Corander J, Auvinen P & Arjas E: A Naive Bayes classifier for protein function prediction.: In Silico Biol, 2009