Hate Speech Detection

Akshar Joshi, Janam Vaidya, and Shobhit Gola

***Abstract*— In recent years, increasing amount of hateful information spreading on social media platforms encouraged ethical communities, governments and companies of digital media for taking countermeasures. A lot of research is being done in the area of automated hate speech detection online. Main objective of this research is to classify content in hateful and non hate speech in the area of racism, religion. However, we notice significant difference between the performance of the two (i.e., non-hate v.s. hate). Here we are comparing various ML and deep learning based models to evaluate performance on tweets classification on hate and non-hate full. Our methods are evaluated on the largest collection of hate speech datasets based on Twitter. We found that how simplest model of logistic regression can out perform deep learning based LSTM model for hate speech classification.**

I. INTRODUCTION

The rapid usage of social media platforms like twitter, Facebook leads to being popular for both political and personal communication. Now a days these communication are being well acknowledged with hatred and harassing speech. Which is constantly violating rules of social media. Hence it is a challenging problem in NLP community which motivates NLP and AI researches to put efforts for the same.

Term 'hate speech' was formally defined as 'any communication that disparages a person or a group on the basis of some characteristics such as race, colour, ethnicity, gender, sexual orientation, nationality, religion, or other characteristics'. In the European countries and America, after attack in London and Washington there is spike in hatred speech and posts towards Muslim migrants and other migrants.

Still hate speech identification process is labour intensive, and time consuming. There is huge demand of an algorithm which can automate process with higher accuracy for hate speech identification. Companies like Twitter, Facebook and Google are investing lots of resources for this objective and still being criticized for not having enough control for hate speech detection.

A focused research is going on across world wide for hate speech classification and detection in social and multi media domain. These research incorporates semantic analysis techniques which are build upon machine learning and natural language processing algorithms. Hate speech classification is another emerging domain where class of hate speech is being decided like racism, skin tone, demography etc. Current research is slightly towards detection of non-hateful content. But our objective is to detect and classify hateful content. Although current methods have reported promising results, we notice that their evaluations are largely biased towards detecting content that is non-hate, as opposed to detecting and classifying real hateful content.

II. OVERVIEW

Our end goal is to classify tweets into two categories, racist/sexist, and neither. This paper will be training datasets of tweets on some classic Machine Learning models and a Deep Learning model. We will observe these experiments and make a note of which model performs better in the specific lot. There will also be a pinch of NLP involved in the process since we will be dealing with raw textual data. Since there will be a lot of textual information, that is tweets; we will use text classifiers. The 'Methodology' in the upcoming sections will give a more in-depth view of our model's structure.

III. DATASET

All data used for training and testing has been retrieved from Kaggle[2]. As social media platforms are considered the place filled with all the hate speech, yet most of them, such as Twitter, have a very restrictive data distribution policy. These tweets were available publicly elsewhere. The tweets in the training dataset have three columns (id, label, and tweets). For simplicity, we say a tweet contains hate speech if it has a racist or sexist sentiment associated with it. The label, in the training dataset, will be '1' if the particular tweet is racist or sexist, and '0' if the tweet is neither. Also, the user's username in the tweets will be replaced with '@user.' An example of the data will be shown in the table below, i.e., the Training dataset. The dataset is not a very complex one. The training dataset and the testing dataset consist of 31,962 tweets and 17,197 tweets, respectively. The test dataset consists of id, target label (i.e., predicted after training the dataset on various models), and tweets.

Training Dataset		
id	label	tweet
1	0	@user when a father is dysfunctional.. .dysfunction.#run
2	0	@user @user thanks for #lyft credit...wheelchair vans in pdx. #disappointed
3	0	bihday your majesty
14	1	@user #cnn calls #michigan middle school 'build the wall' chant " #tcot

IV. METHODOLOGY

In this paper, we have, firstly, performed text cleaning and pre-processing, which will take out irrelevant data from the text (i.e. tweet), further elaborated upon in 'A'. The next

step is of text classification, this is a very crucial step as it categorizes text into organised groups. Text classifiers can automatically analyze text and then assign a set of pre-defined tags or categories based on its content. Some of the text classifiers are Bag of Words, Word2vec, TF-IDF, etc. In this paper, we have used TF-IDF, and has been talked about more in-depth in 'B'. Following this step, we have the training of the dataset on various models. The baseline models used are of (some classic ML models) Random Forest, SVM, and (neural-net) LSTM based model, which can be read about in the 'C'. Our paper shows that compared to these models, classic ML model of Logistic Regression gives better accuracy on this particular dataset.

A. Text Cleaning and Pre-processing

This is a very important feature whenever dealing with textual information, as it does make changes in the overall accuracy and performance of any particular model. This aspect of our paper brings in Natural Language Processing (NLP). Text data needs to be cleaned and encoded to numerical values before giving them to Machine Learning and Deep Learning models, this process of cleaning and encoding is called as text cleaning and pre-processing. Text cleaning[3,4] is basically an amalgamation of-

i removing punctuations, unwanted tags, i.e. noise removal
ii expanding contractions
iii tokenisation
iv converting all text to lowercase
v removing stopwords
vi performing stemming
vii performing lemmatization

Tokenisation is the process of breaking down strings into tokens, which in turn are small structures. For instance, bigrams are tokens of two consecutive words, and trigrams are tokens of three consecutive words. A token is the technical name for a sequence of characters that we want to treat as a group. When we count number of tokens in a text, we are counting occurences of these sequences.
In stemming, we normalise words into its base form or root form (i.e. by cutting off the end and the beginning of the word). Some of the types of stemmer's are Lancaster, Snowball, Porter. While using Snowball stemmer, we have to provide the language we are using for text cleaning. Stemming does reduce the vector dimension since after performing it, we don't consider all similar words.
Now for lemmatization, it takes into consideration the morphological analysis of the word. It generally does under perform compared to stemming. . Lemmatization does the following steps-

i It groups together different different infelcted forms of a word, called lemma.
ii It is somehow similar to stemming, as it maps several words into one common root.
iii The output of lemmatization is a proper word.
iv For instance, it maps 'gone','going', and 'went', into 'go'.

In code, lemmatization requires a dictionary which the algorithm can look into to link back to the form in its original lemma. Since the output is a perfect word, thus it needs to have a dictionary.
When performed all of these aspects of text cleaning, we will have transformed our raw data into clean data, which will be easier for the training models to understand.

B. Text Classifier: TF-IDF

Considered one of the best text classifiers[5,6,7], TF-IDF significantly enhances the performance of all the models. The main purpose of text classification is to convert open-ended text into categories. By vectorising texts, we can further perform multiple tasks such as finding relevant documents, ranking, clustering, and so on.
TF-IDF stands for **Term Frequency - Inverse Document Frequency.** TF-IDF has very large dimensionality. The main idea behind TF-IDF is text mining and information retrieval. TF-IDF as a text classifier is used to calculate **weight** or **score** of words in the document or in a set of texts.
Term-Frequency(TF) of a word is more if the word has been used more number of times. TF for a word can be calculated by the formula:

$$tf(w,d) = \frac{count\ of\ w\ in\ d}{number\ of\ words\ in\ d}$$

where w=word and d=document.
Inverse Document Frequency(IDF) is the inverse of the document frequency which measures the informativeness of word w. We need to weigh down the frequent terms, while scaling up the rare ones. IDF for a word can be calculated by the formula:

$$idf(w) = \frac{N}{occurence\ of\ w\ in\ documents}$$

where N=count of corpus(set of texts).
When we multiply tf(w,d) and idf(w), we will get the **importance** of a word in a text, relative to the set of texts. The TF-IDF model contains information on more important words and less important ones as well. TF-IDF vectors were generated using Scikit's TF-IDF vectorizer on the pre-processed tweets.

C. Baseline Models

1) Multinomial Naive Bayes:

One of the probabilistic algorithms of machine learning is the Multinomial Naive Bayes[5,8] algorithm. It is the most traditional method of text categorization, also includes training of multi-dimensional data sets and information retrieval. Some examples are document classification, span filtration, and sentimental analysis. In NB, the class conditional probability is modeled as n-dimensional gaussian distribution. This technique is a generative model. It is a probabilistic algorithm based on the assumption that features of a class are independent. Bayes theorem calculates probability P(c—x) where c is the class of the possible

outcomes and x is the given instance which has to be classified, representing some certain features.

$$P(c|x) = P(x|c) * P(c)/P(x)$$

Tag of a text is predicted by Naive Bayes, and it calculates the probability of every tag in a given text and its output is the tag which has the highest probability. NLP problems are one area where Multinomial Naive Bayes are generally and most used.

2) Random Forest:

Random forest is a supervised learning algorithm. It builds an ensemble of decision trees with the help of the bagging method. The general idea of the bagging method is that many relatively uncorrelated decision trees operating as a committee will outperform any individual constituent models. Random forest[9,10] builds multiple decision trees and merges them to get a more accurate and stable result. One advantage of the Random forest model is that it is used for classification and regression problems; these problems form the majority of the current machine learning system.

There are two things we need for our random forest to give accurate predictions:

i Some predictive power should be allowed to the features of this model.

ii The decision trees and their predictions need to be uncorrelated.

3) Long Short-Term Memory:

It[11,12] is a special neuron for memorizing long term dependencies and is much more effective than basic Recurrent Neural Network (RNN). Repeating the same basic networks, RNNs can be thought of as multiple copies, each passing a message to a successor, where the gap between relevant information and place that it's needed is small, RNNs can learn to use past information. As the gap grows, RNNs become unable to learn information; thus, LSTM[13] comes into the picture. LSTM overcomes the vanishing gradient problem. To preserve long term information, LSTM uses multiple gates (i.e., forget gate, store gate, input gate, and output gate) to regulate the amount of information. It uses activation functions like sigmoid and tanh. Although LSTM is considered to give a good and accurate performance, there is a high chance of it under-performing due to the problem of vanishing gradients.

D. Proposed Model

Since the dataset can be classified as racist/sexist and neither using '1' and '0', it can be seen as a basic classification problem. Thus, **Logistic Regression** would be a better way to predict probabilities. This has been tested in this paper. The reason why Logistic Regression has outperformed some state of the art techniques, is because sometimes other factors such as the complexity of the dataset plays a huge role in the training of models. Logistic Regression uses the log odds ratio rather than probabilities and an iterative maximum likelihood method. The way Logistic Regression works is that it fits a single line to divide the space into two and so performs better than a decision tree when the data is distributed in a fashion such that it can be linearly classified. The model[14] uses L2 regularisation form, thus to solve the penalties, the solver's used are 'newton-cg', 'lbfgs', 'sag' and 'saga'. The regularisation is done to reduce the variance of a model, which then takes care of overfitting in the training model. Between the two regularisation methods (i.e., L1 and L2), the difference is only of the penalty term. It is using the Logistic Regression implementation available in scikit-learn. About Logistic Regression, it constructs linear decision boundaries and assumes variables are independent. It can also be easily extended to multiple classes (i.e., Multinomial Regression) instead of binary. It also uses sigmoid functions to achieve better numbers while training. The logistic regression model is a classification model and, on the other hand, also gives you probabilities. This is a big advantage over models (the baseline ones) that can only provide the final classification. When solving a classification problem, the activation function on the final output layer should be sigmoid. The results of all the models will be shown in the next segment. Representation of Logistic-

$$Logistic(\eta) = 1/(1 + exp(-\eta))$$

V. Results and Analysis

After performing our experiments, we received the following results (Table). We obtained the best result for Logistic Regression (LR) classifier, i.e., 96.07 percent accuracy, as it can be seen that the classic ML model of Logistic Regression has outperformed other techniques, including LSTM technique. Performing text cleaning (i.e., basic NLP technique) and text classifiers have also enhanced the performance. Thus, the source code for the models gave the following F1-scores and accuracies.

Results		
Experiments	F1-Score	Accuracy
Multinomial NB	19.56	93.51
Random Forest	48.91	95.00
LSTM	58.66	94.99
Logistic Regression	64.92	**96.13**

VI. Conclusions

With the help of this paper, we detect racist/sexist tweets on twitter. We have observed that it's not always that state of the art deep learning models outperforms all other models. Sometimes classic Machine Learning models perform better than deep learning models. The less complex the dataset, using classic machine learning models give better results. Also, when the TF-IDF text classifier was used, it gave better performances combined with Logistic Regression. A logistic regression model for binary classification can be far better than a neural network model because neural networks are more challenging to train and are more prone to overfitting than logistic regression. Logistic regression is a neural network with no hidden layers.

REFERENCES

[1] S.MacAvaney, H.Yao, E.Yang, K.Russell, N.Goharian, O.Frieder. *Hate speech detection: Challenges and solutions*. 2019.

[2] Provided by Analytics Vidhya : Twitter Sentiment Analysis.
https://kaggle.com/arkhoshghalb/twitter-sentiment-analysis-hatred-speech
Accessed 27 July 2020

[3] Hamza, Ali: Effectively Pre-Processing the Text Data Part 1: Text Cleaning.
https://towardsdatascience.com/effectively-pre-processing-the-text-data-part-1-text-cleaning-9ecae119cb3e.
Medium, 31 January 2019

[4] B.Pahwa, S.Taruna, N.Kasliwal. *Sentiment Analysis- Strategy for Text Pre-Processing*. 2018.

[5] K.Kowsari, K.J.Meimandi, M.Heidarysafa, S.Mendu, L.Barnes, D.Brown. *Classification Algorithms: A Survey*. 2019.

[6] Abusalah, Mustafa: Twitter Hate Speech Sentiment Analysis.,
https://medium.com/@muabusalah/twitter-hate-speech-sentiment-analysis-6060b45b6d2c
Medium, 14 September 2019

[7] Scott, William: TF-IDF for Document Ranking from Scratch in Python on Real World Dataset.
https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-n-real-world-dataset-796d339a4089
Medium, 21 May 2019

[8] M.Abbas, K.Ali, S.Memon, A.Jamali, A.Ahmed. *Multinomial Naive Bayes Classification Model for Sentiment Analysis*. 2019.

[9] Yiu, Tony : Understanding Random Forest.
https://towardsdatascience.com/understanding-random-forest-58381e0602d2
Medium, 14 August 2019

[10] M.Denil, D.Matheson, N.Freitas. *Narrowing the Gap: Random Forests in theory and in practice*.

[11] S.Hochreiter, J.Schmidhuber. *Long Short-Term Memory*. 1997.

[12] Colah's Blog : Understanding LSTM Networks.
https://colah.github.io/posts/2015-08-Understanding-LSTMs/
Accessed 27 July 2020

[13] Phi, Michael : Illustrated Guide to LSTM's and GRU's: A Step by Step Explanation.
https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
Medium, 28 June 2020

[14] C.J. Peng, K.L.Lee, G.Ingersoll. *An Introduction to Logistic Regression Analysis and Reporting*. 2002.

[15] P.Badjatiya, S.Gupta, M.Gupta, V.Varma. *Deep Learning for Hate Speech Detection in Tweets*. 2017.