NATURAL LANGUAGE PROCESSING

HATE SPEECH AND OFFENSIVE LANGUAGE DETECTION

DONE BY

ROHITH G [CB.EN.U4AIE19026]

ABSTRACT

With the improvement in technology and its ease of access to people around the globe, toxicity has also increased in text messages and social media. The increasing propagation of hate speech on social media and the urgent need for effective countermeasures have drawn significant investment from governments, companies, and researchers. A large number of methods have been developed for automated hate speech detection online. Automated hate speech detection is an important tool in combating the spread of hate speech, particularly on social media. Numerous methods have been developed for the task, including a recent proliferation of deep-learning-based approaches.

The detection of hate speech on social media is a crucial task. The uncontrolled spread of hate has the potential to gravely damage our society, and severely harm marginalized people or groups. Also in such speech, there is no proper grammar that can be followed or a pattern. They vary with languages and regions. People tend to showcase anger through different terms and symbols. This significantly increases the difficulty of automatic detection, as social media posts include paralinguistic signals like emoticons and hashtags, and their linguistic content contains plenty of poorly written text.

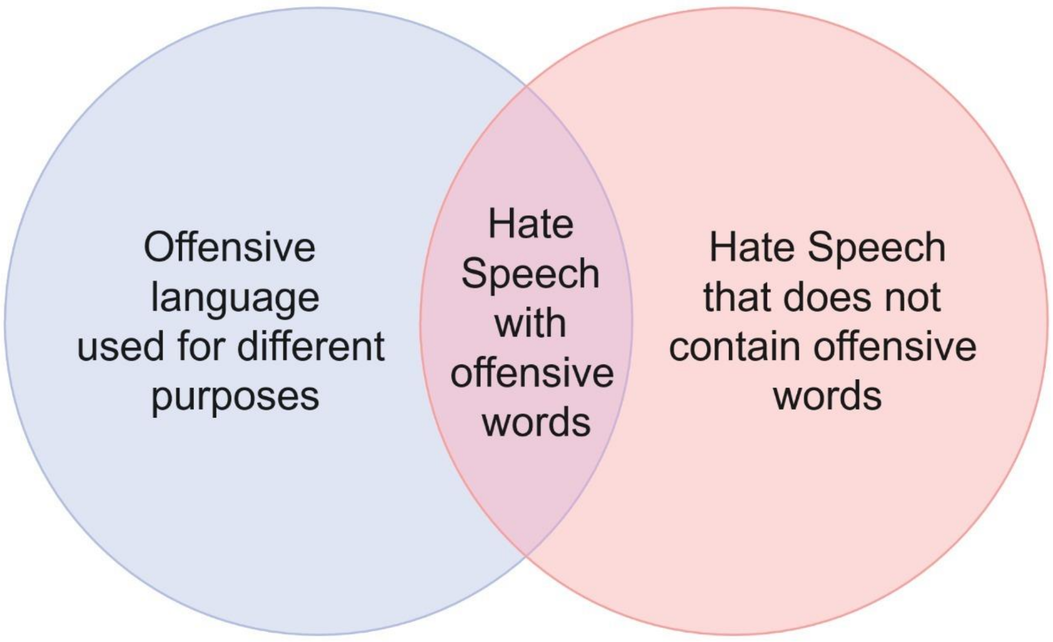
This project explores and tackles the problem of detecting hate speech and abusive content using multiple models and explores different languages. It comes across tackling multiple challenges due to language variations and achieves good accuracy based on the models used on them. Multiple machine learning models like logistic regression, linear SVM, random forests, and KNN were used to analyze and detect hate speech and offensive content in the Twitter dataset. This Linear SVM model performed the best and gave an accuracy of 89.41%. Then the Tamil dataset is analyzed with the term frequency-inverse document frequency [TF – IDF]. Tamil is one of the richest languages in this world and it's hard to analyze the different words and combinations used in it. Hence we first use TF – IDF which helps quantify the Tamil words from the given dataset and from that the data is preprocessed and cleaned and then fitted with the ML models. Even for this the Linear SVM model performs the best giving an accuracy of 70.35%, the highest among others. TF – IDF is also used to quantify other datasets taken as well.

Similarly, hate speech and offensive language detection were done for Arabic, German, and English. For these different models were used. RNN, LSTM, and Bi-LSTM were used to analyze the Arabic dataset. Bi – LSTM performed the best of all with an accuracy of 77.56%. For German and English, the dataset from HASOC was used and SVM was applied to it. Additionally, deep learning was done on the English dataset and the accuracy was analyzed.

KEYWORDS:
Artificial Intelligence, Deep Learning, Machine learning, Computational modeling, Hate speech, Speech recognition, Text analysis.

INTRODUCTION

This project tries to identify the hate speech and offensive content from social media platforms where they are highly used by introducing various machine learning and deep learning models in multiple languages. Different languages have unique structures and based on how they are phrased, the models should be carefully selected and used. The languages seen in this project are Arabic, English, German, and Tamil.


Offensive language used for different purposes
Hate Speech with offensive words
Hate Speech that does not contain offensive words

Their corresponding datasets were taken from different sources and analyzed and preprocessed. The data has to be cleaned first of all the junk characters like symbols, emojis, and spaces. Once the data cleaning has been done, they are then tokenized and split into their constituent words that help the analysis. There are 3 main classes to be analyzed – Hate speech, Offensive language, and neither. Then the ML and DL models are fitted to these cleaned data and the output is analyzed. The accuracies for various models are taken and then compared.

LITERATURE SURVEY

Hate-Speech and Offensive Language Detection written by Hammad Rizwan, Muhammad Haroon Shakee. So, in this paper, The task of automatic hate speech and offensive language detection in social media content is of utmost importance due to its implications in unprejudiced society concerning race, gender, or religion. Existing research in this area, however, is mainly focused on the English language, limiting the applicability to particular demographics. Despite its prevalence, Roman Urdu (RU) lacks language resources, annotated datasets, and language models for this task.

In this study, we: (1) Present a lexicon of hateful words in RU, (2) Develop an annotated dataset called RUHSOLD consisting of 10, 012 tweets in RU with both coarse-grained and fine-grained labels of hate speech and offensive language, (3) Explore the feasibility of transfer learning of five existing embedding models to RU, (4) Propose a novel deep learning architecture called CNN-gram for hate speech and offensive language detection and compare its performance with seven current baseline approaches on RUHSOLD dataset, and (5) Train domain-specific embeddings on more than 4.7 million tweets and make them publicly available. We conclude that transfer learning is more beneficial as compared to training embedding from scratch and that the proposed model exhibits greater robustness as compared to the baselines.

Automatic Hate speech detection is written by Mohiyaddeen and Dr. Sifatullah Siddiqi. So, in this paper Hate speech has been an ongoing problem on the Internet for many years. Besides, social media, especially Facebook, and Twitter have given it a global stage where those hate speeches can spread far more rapidly. Every social media platform needs to implement an effective hate speech detection system to remove offensive content in real-time. There are various approaches to identify hate speech, such as Rule-Based, Machine Learning based, deep learning-based, and Hybrid approaches. Since this is a review paper, we explained the valuable works of various authors who have invested their valuable time in studying to identify hate speech using various approaches.

1. Rule-Based Linguistic Approaches
In the Linguistic rule-based approach, Hate speech detection uses a linguistic engine that understands the grammar, morphology, and semantics of a specific language. Furthermore, the program adds rules that check for unique core semantic terms in the sentence in order to determine their potential meanings. For instance, if we input the keyword “bad.” The linguistic engine will automatically search for the terms “terrible/awful/unsatisfactory” as well.

2. Machine Learning Approaches
Machine learning creates a mathematical model based on training data to make predictions or decisions without being explicitly programmed. The aim of Machine learning is to make a classifier or regression model through learning the training data set and then using the test data set to evaluate the performance of the classifier or regression model. Machine learning can be classified into the following categories based on the nature of the training data. e.g. Supervised learning, Unsupervised learning, Semi-supervised learning.

3. Deep Learning Approaches
The deep learning approach uses neural networks to solve complex problems in an innovative way. When you feed a neural network a series of examples, such as pictures of humans, It can recognize the features that are shared by those pictures. When we use layers of neural networks side by side, these layers recognize every detail of the picture to create an effective model. After sufficient training, a neural network becomes refined and capable of classifying unlabeled pictures

METHODOLOGY

In this project, since we use multiple datasets, the codes have been split into five parts. Each part focuses on a dataset from the mentioned languages and the corresponding model that was fit on them and analyzed.

The performance of the learning algorithms is affected by how the data is represented. Several algorithms have been proposed for feature extraction. TF-IDF has shown prominent results; therefore, we used it for feature extraction in this study. It combines both TF, which is the frequency of the word in the document, and IDF is the distribution of the word in the documents. It is calculated as below,

$$\mathrm{TF(T,D)} = \frac{\text{Frequency of termTin documentD}}{\text{Total no of terms in documentD}}$$

$$\mathrm{IDF(T)} = \log \frac{\text{Total number of documents}}{\text{Frequency of TermTin All docs}}$$

$$\mathrm{TF-IDF(T,D)} = \mathrm{TF(T,D)} \times \mathrm{IDF(T)}$$

CLASSIFICATION:

Both conventional machine learning and deep learning models were employed for classification.

1. SUPPORT VECTOR MACHINES (SVM):

Support Vector Machine is one of the most commonly used supervised learning algorithms for classification and Regression. Generally, SVM is designed to deal with linearly separable data for binary classification problems. However, it can be adopted for non-linear data by applying kernels on the input data to transform the input data into a higher-dimensional space. This higher dimensional space is assumed to separate the data linearly. Similarly, several approaches can be adopted for multi-class classification problems, such as one-vs-one, one-vs-all, etc. The main objective of the SVM is to maximize the margin of separation between the classes. After training, it only keeps the data points at the border of the margin. Such a method is helpful as it only selects a subset of the training data that can have an advantage in situations where training data is small. The SVM classifier should allocate the data to the correct side of the margin using the following condition for the input data χ.

$$y_i\left(\mathbf{w}^T\mathbf{x}_i - b\right) \geq 1, \text{ for all} 1 \leq i \leq n$$

2. RANDOM FOREST:

Random Forest is an ensemble of classifiers that is constructed by integrating several decision trees. It can be used for both classification and regression problems. It is more efficient than a decision tree classifier because several trees can result in a more robust decision and improve the overall accuracy. The splitting of the data can be performed using various strategies such as gabbing or bootstrap aggregation. In such a strategy, random samples are selected with repetition and fit trees to the selected sample. The predictions are then made on the validation data by averaging the predictions from each individual tree.

3. LOGISTIC REGRESSION

Logistic Regression, also known as the logit model, is a statistical model that employs a logistic function to model a single binary dependent variable. It is a classic machine learning algorithm that has been widely used in a variety of applications. The l1 regularized Logistic Regression for any problem can be defined as

$$\min \sum_{i=1}^{N} \log\left(1 + \exp\left(-y_i \left(\mathbf{w}^T \mathbf{x}_i + \theta\right)\right)\right) + \lambda \|\mathbf{w}\|_1$$

where w is the weight vector and θ is the LR model parameters, and $\lambda > 0$. The classifier can then be used for the classification of the test data with labels $y \in \{-1,1\}$ as

$$\Pr(y = 1 \mid \mathbf{x}) = \frac{1}{1 + \exp(-(\mathbf{w}^T \mathbf{x} + \theta))}$$

$$\Pr(y = -1 \mid \mathbf{x}) = \frac{1}{1 + \exp(\mathbf{w}^T \mathbf{x} + \theta)}$$

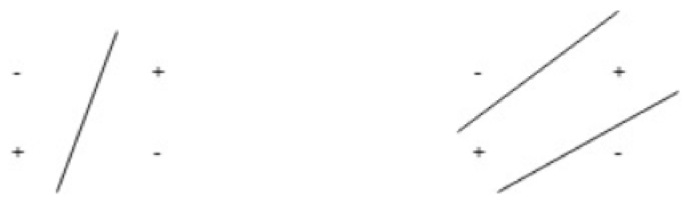
4. KNN:

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on the Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a good suite category by using K- NN algorithm. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for Classification problems. K-NN is a **non-parametric algorithm**, which means it does not make any assumptions on underlying data. It is also called a **lazy learner**

algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

5. MULTILAYER PERCEPTION:

Multilayer Perceptron is a Neural Network that learns the relationship between linear and non-linear data. The perceptron is very useful for classification data sets that are linearly separable. They encounter serious limitations with data sets that do not conform to this pattern as discovered with the XOR problem. The XOR problem shows that for any classification of four points that there exists a set that is not linearly separable.



The Multilayer Perceptron (MLPs) breaks this restriction and classifies datasets that are not linearly separable. They do this by using a more robust and complex architecture to learn regression and classification models for difficult datasets.

1. RECURRENT NEURAL NETWORK (RNN):

Recurrent neural networks are powerful techniques that can deal with sequential data efficiently. Since text can be considered a sequence of words and sentences, RNN is suitable for processing textual data. In this study, we particularly focused on a specific type of RNNs called Bidirectional RNNs. It adds an additional hidden layer to the network so that it passes information in the backward direction apart from the forward direction. For an input X at time step t and hidden units h, the forward and backward states can be updated as,

$$H(t)_f = \phi\left(\mathrm{X}_t W_{xh} + H(t)_{t-1} W_{hh} + b_h\right)$$

$$H(t)_b = \phi\left(\mathrm{X}_t W_{xh} + \mathrm{H(t)}_{t-1} W_{hh} + b_h\right)$$

H(t)f and H(t)b represent the hidden state in the forward and backward directions, respectively. W represents weight, and b represents bias. Bidirectional RNN forms concatenate both. H(t)f and H(t)b and obtains a hidden state *H(t)*. The final output is then obtained using,

$$O(t)=H(t)W+b$$

Here *O(t)* is the output obtained at time *t*.

All models used require tuning of some parameters. The performance of the model is dependent on parameter tuning. In this study, the parameters for all models were empirically calculated, and the optimal values were then used for testing.

IMPLEMENTATION:

EXPERIMENTAL CODE 1

Here the Twitter dataset is taken and analyzed for the English language for its hate speech and offensive content. Since NLTK packages are used, they are first downloaded and unpacked. From the CSV dataset file, after loading it, each line/tweet is read and printed into the data frame that is classified into one of the three classes - Hate speech, Offensive language, and neither. Also, it tells the number of harmful words detected in each tweet.

This is the raw data. This is then cleaned and stripped of all the unnecessary characters like symbols, emojis, and gibberish words so that the models can easily work better and efficiently on the data when fit, providing better results. Hence, after preprocessing the data, it is then quantified of the words using TF – IDF, and then the output split data is printed visually to show the distribution of safe, hate, and offensive words. Now the model is split into test and train data. Then the ML models namely Linear SVM, logistic regression, multilayered perceptrons, K Nearest Neighbours (KNN), and random forests are applied and their individual accuracy scores are analyzed from the output. The precision, recall, and f1 – scores give the final best accuracy for each model, and from the results obtained it is safe to say that the Linear SVM model performed the best out of all the models fit, with an accuracy rate of 89.41%.

EXPERIMENTAL CODE 2

In this code, the Tamil dataset is used. Its corresponding dataset CSV file is loaded. After loading it, each line/tweet is read and printed into the data frame that is classified into different classes. The null entries and unknown tags are filled and removed respectively. Tamil is one of the languages with rich grammar and words. Hence it's a bit of a tedious task to classify the negative words and comments from this Tamil dataset. The dataset is then cleaned of unnecessary characters, emojis, and gibberish words. Also Tamil is a language that is heavily influenced by English words. Hence those words are also filtered and segregated in the preprocessing and cleaning stage.

The classes identified in the Tamil corpus are Counter-speech, homophobia, misandry, misogyny, not – Tamil, transphobia, xenophobia, and hope-speech. After this identification, the clean data is then finally vectorized and quantified using TF – IDF after which it is split into train and test data, and then the ML models are fitted and their individual accuracy scores are analyzed from the output. The precision, recall, and f1 – scores give the final best accuracy for each model, and from the results obtained it is safe to say that the Linear SVM model performed the best out of all the models fit, with an accuracy rate of 70.35%.

EXPERIMENTAL CODE 3

For the third part, the Twitter dataset for the Arabic language was taken. Its CSV file was loaded and read. This has 3 classes – Normal, abusive, and hate. The tweets were then split into these three identified classes and then analyzed for the number of occurrences and shown visually. For this code instead of using the ML models used so far, RNN and LSTM models were used. Three models namely RNN, LSTM, and Bi – LSTM were used and their output accuracies were compared. Similar to the previous data used and analyzed, this Arabic dataset was also first pre-processed and then cleaned and stripped of all the extra characters and emojis, and spaces. Also, the Arabic language is written backward so the data has to be preprocessed carefully to avoid losing valuable information.

After data cleaning and tokenizing, the models are applied. Reinforcement Learning takes in the data and updates the weights by back-propagating the updated error values and updating the node weights. Then LSTM and Bi – LSTM are used. These are updates for the RNN model and prevent the vanishing gradient problem. This uses memory units called cell states that store information as memory. This creates feedback connections and allows the model to not only process single data points such as images, but also entire sequences of data such as speech or video. Bi – LSTM traverses and stores the information of the previous and subsequent values of a particular data, making it more efficient in learning and providing accurate outputs. Like so, Bi -LSTM is the model that works best here, providing an accuracy of 77.56%

EXPERIMENTAL CODE 4

This code takes in German and English languages from the HASOC dataset and is evaluated. Two tasks are evaluated in this model. Task 1 is a binary classifier model and task 2 is a multi-class classifier model. The binary class model takes in inputs for ‘Offensive’ and ‘Not Offensive’ words whereas the multiclass takes in input for ‘Hate’, ‘Offensive’, ‘None’, and ‘PrFn.’ Based on these classes the model is trained and accuracy is obtained. In task 1 the SVM F1 score of 93.70% and in task 2 it comes up to 93.98%

EXPERIMENTAL CODE 5

This code part is similar to that done in Code 1 where the English twitter dataset was analyzed. Here a Deep Learning model called simple RNN is used to evaluate the accuracy. It provided a test accuracy of 96.73%

RESULT AND DISCUSSION

In the experimental codes 1 and 2, we used classic machine learning algorithms such as random forests, KNN, linear SVM, etc., to classify and identify hate speech and offensive languages. Out of these, for both the English and Tamil datasets used, the Linear SVM algorithm performed better than others giving the best accuracies.

The models were evaluated in terms of accuracy (Ac), precision (Pr), recall (Rc), and F1-Measure (F1). These standard metrics are calculated as

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

$$Precision = \frac{\text{True Positive}}{\text{True Pos.} + \text{False Pos.}}$$

$$Recall = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

$$F - measure = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Where TP is truly positive, TN is a true negative, FP is a false positive, and FN represents false negatives.

Performing these five experiments gave me a deep understanding of all the algorithms used to train the models. From all the codes taken the models that used LSTM, SVM, and RNN models performed the best and gave the highest accuracies. From using these models for the Arabic dataset in experimental code 3, RNN gave an accuracy of 63.88%. The LSTM models performed better than this RNN algorithm and gave better accuracies of 68.45% and Bi -LSTM gave 77.56%.

The experimental code 5 uses the simple RNN model after we used TF – IDF to quantize the data and then tokenized the cleaned data. This helped improve the accuracy a lot. Hence we get an accuracy of 96.73% for this model. Hence it is safe to say that since the LSTM model performs better than the average RNN models, using that here might provide even better accuracies that might amount to around 99%.

CONCLUSION

This project helped give a deep insight into the best models to use to identify hate speech and offensive language from a given dataset. The machine learning algorithms performed well and gave good accuracies for the datasets used in experimental codes 1 and 2. But overall Bi - LSTM is the robust data-driven technique to identify and classify the offensive words from the data taken.

LSTM in general works really well for time series data analysis like these word classifications and in sentiment analysis problems because they use memory/cell states that store the word information of the data for a specified amount of time. This helps enhance the accuracy and makes the algorithm more efficient in training the model. Linear SVM can also be trusted and used in multiple classification problems such as these.

With moving averages, the SVM and LSTM models both perform significantly better on the combined dataset over the standard base dataset. This is because of the smoothing effect of the moving averages on the data which helps in learning the influence of the external parameters on the base stock price in a much better manner.

REFERENCES

1. *F. E. Ayo, O. Folorunso, F. T. Ibharalu, and I. A. Osinuga, "Machine learning techniques for hate speech classification of Twitter data: State-of-The-Art future challenges and research directions", Comput. Sci. Rev, vol. 38, pp. 100311, 2020.*

2. *P. Kapil and A. Ekbal, "A deep neural network-based multitask learning approach to hate speech detection", Knowledge-Based Syst, vol. 210, pp. 106458, 2020.*

3.*S. Safari, S. Sadaoui, and M. Mouhoub, "Hate and offensive speech detection on Arabic social media", Online Soc. Networks Media, vol. 19, no. March, pp. 100096, 2020.*

4.*P. Fortuna and S. Nunes, "A Survey on Automatic Detection of Hate Speech in Text", ACM Comput. Surv, vol. 51, no. 4, pp. 1-30, Sep. 2018.*

5.*H. Karayiğit, Ç. Inan Aci and A. Akdağll, "Detecting abusive Instagram comments in Turkish using convolutional Neural network and machine learning methods", Expert Syst. Appl, vol. 174, pp. 114802, Jul. 2021.*

6.*S. A. Shah, D. Z. Seker, M. M. Rathore, S. Hameed, S. Ben Yahia, and D. Draheim, "Towards Disaster Resilient Smart Cities: Can the Internet of Things and Big Data Analytics Be the Game Changers?", IEEE Access, vol. 7, pp. 91885-91903, 2019.*

7. *F. E. Ayo, O. Folorunso, F. T. Ibharalu, I. A. Osinuga, and A. Abayomi-Alli, "A probabilistic clustering model for hate speech classification in Twitter", Expert Syst. Appl, vol. 173, pp. 114762, Jul. 2021.*

8.*J. Dhillon, V. Gupta, R. Govil, B. Varshney and A. Sinha, "Crowdsourcing of Hate Speech for Detecting Abusive Behavior on Social Media", 2019 International Conference on Signal Processing and Communication (ICSC), pp. 41-46, 2019.*

9. *A. P. Jain and P. Dandannavar, "Application of machine learning techniques to sentiment analysis", 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), pp. 628-632, 2016.*

10.*H. Sahi, Y. Kilic, and R. B. Saglam, "Automated Detection of Hate Speech towards Woman on Twitter", 2018 3rd International Conference on Computer Science and Engineering (UBMK), pp. 533-536, 2018.*