Racism detection using deep learning techniques

Sukanya L[1*], *Aniketh* J[1*], *Abhiman Sathwik,* E[1], *Sridhar Reddy* M[1], and *Hemanth Kumar* N[1]

[1]Department of Information Technology, GRIET, India

Abstract. With the pervasive role of social media in the socio-political landscape, various forms of racism have arisen on these platforms. Racism can manifest in various forms on social media, both concealed and overt. It can be hidden through the use of memes or exposed through racist comments made using fake profiles to spread social unrest, violence, and hatred. Twitter and other social media sites have become new settings in which racism and related stress appear to be thriving. Racism also spread based on characteristics including dialect, faith, and tradition. It has been determined that racial animosity on social media poses a serious threat to political, socioeconomic, and cultural equilibrium and has even put international peace at risk. Therefore, it is crucial to monitor social media as the primary source of racist opinions dissemination and to detect and block racist remarks in a timely manner. In this study, we aim to detect tweets containing racist text by performing sentiment analysis using both ML and DL algorithms. We will also build a webpage using Flask framework and SQLite for users to interact with the model.

1 Introduction

Social media has become a powerful force in the realm of socio-political issues, exerting a significant influence over our thoughts and actions in a variety of ways. The widespread use of social media platforms worldwide, coupled with the freedom of expression, has given rise to various negative trends in recent years, with racism being among the most significant. Presently, 22% of American adults are active on Twitter, while the platform boasts three hundred thirty-six million users in use and a total of 1.3 billion profiles worldwide, 90% of which have a public profile. Racism is frequently subtly represented on these platforms, such as through memes, but it is also overtly stated in racist tweets, which are frequently uploaded under false identities. Deep learning has shown great potential in automated categorization tasks. LSTM models were specifically designed to accelerate text processing and perform text classification tasks. We have developed an LSTM model to classify randomly selected predefined textual data. This model will help to reduce the time required for detecting racism. The algorithm's time efficiency is much better than existing approaches, and it provides high-performance speed and works efficiently.

*Corresponding author: anikethjinda114@gmail.com

2 Literature survey

K.R. Kaiser, ET. AL[4], in his article titled "Using social media to understand and guide the treatment of racist ideology," discusses how social media sites like Facebook, Twitter, and Instagram have turned into havens for racist ideology, exposing the flaws in American society. Social media, according to the authors, can provide useful insights into the thoughts of racists who fiercely guard their tribal identities and irrationally despise people who are different. One can learn more about the underlying mechanisms that can free society from its segregationist beliefs and work towards the abolition of racism by researching social media. The paper's main goal is to examine social media posts to better understand the causes of racism and devise first approaches for combating racist ideology. The authors did qualitative research that comprised a content analysis of 600 American Facebook postings to look for trends in cognition, problem-solving, personality structures, belief systems, and coping methods. A descriptive account of the data and an interpretive analysis are combined in the content analysis to give a framework for using social media to understand and guide the treatment of racist ideology.

CARLOS GERSHENSON in his study describes "Artificial Neural Networks". This training package's objective is to give those who are unfamiliar with artificial neural networks (ANNs) a brief introduction to them. We briefly introduce network models first, and then we give a comprehensive overview of ANNs. We provide an explanation of the backpropagation algorithm as an example because it is a frequently used algorithm from which many others are built. Every layer is described, along with how it works and how the outputs are organized.

J.S.MALIK.ET.AL.[3], in their paper "Deep Learning for Hate Speech Detection: A Comparative Study," emphasize the value of automated hate speech detection as a strong tool for halting the spread of xenophobic remarks, particularly on social networking platforms. To approach this task, numerous techniques have been created, including a profusion of deep learning-based solutions. Additionally, a number of datasets have been created that emphasize various aspects of the hate-speech identification issue. The study's main goal is to assess the field's advancement and pinpoint the advantages and disadvantages of the most cutting-edge practices currently available. The authors put a strong emphasis on indicators of actual performance, such as domain generalization, detection accuracy, and computing efficiency.

P.Zhou.ET.AL[6], in his paper "Text classification improved by integrating bidirectional LSTM with two-dimensional max pooling", describes how RNN and LSTM have benefits over other networks to classify text. These use 2-Dimensional max pooling to get more meaningful features for sequence modeling tasks. Due to its recurrent nature, which is well suited to processing variable-length text, recurrent neural networks (RNNs) are among the most often used designs in NLP activities.

ANTIGONI-MARIA FOUNTA.ET.AL [9], in their paper "Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior", Emphasise that there has been an increase in sexist, racist, and other types of abusive and cyberbullying behaviour throughout time, which is frequently conveyed through rude, abusive, or hateful language. The majority of prior research has been devoted to examining these abusive practises in well-known online social networks, such Facebook and Twitter. To build on earlier research, the authors provide an eight-month study that looks at various abusive behaviours on Twitter using a variety of labelling schemes that encompass numerous abusive behaviours.

ADITYA GAYDHANI. ET. AL [4], in their paper "Detecting Hate Speech and Offensive Language on Twitter using Machine Learning" cite statistics showing that more people from

varied backgrounds are using the internet and that poisonous online content has grown to be a major problem today. The computerized detection of hazardous text content faces a significant issue in distinguishing between hate speech and offensive language. The authors propose a technique for classifying tweets on Twitter automatically into three categories: inflammatory, clean, and hateful. On a Twitter dataset, they conduct tests using n-gram features and TFIDF values given to several machine learning algorithms. The authors compare the models while accounting for various n-gram n values and TFIDF normalization methods. Additionally, they create a module that acts as a go-between between the user and Twitter.

3 System architecture

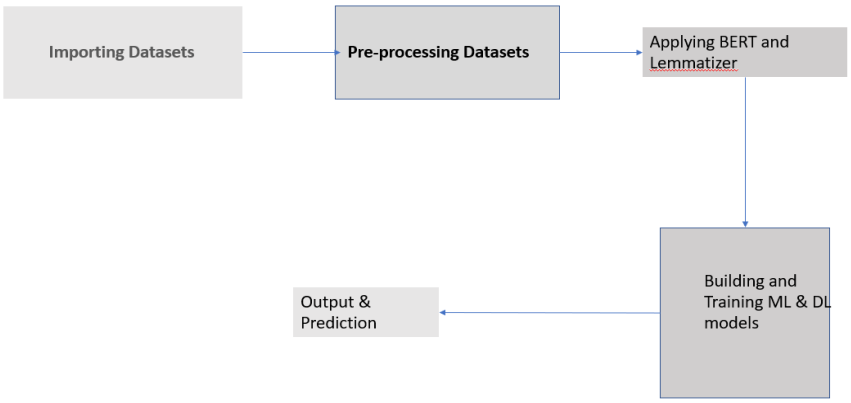

Fig. 1. Functioning of our system

4 Methodology

4.1 Dataset

The process starts by gathering datasets from Kaggle. This can be obtained by visiting Kaggle website and signing in using Gmail account. After completion of signing in, you can search for racist-nonracist tweets. Download the datasets you need. The datasets will be in excel format.

4.2 Data pre-processing

Once the data has been acquired from the Kaggle, it needs to be pre-processed. Since the raw dataset is in the form of texts. There are certain pre-processing techniques which can be applies to clean the dataset.

4.2.1 For ensemble models

We apply pre-processing steps using various NLP techniques. The idea is to simply remove the words that are common to all documents in the corpus. It removes special characters, converts them into lower case, removes mail, tags etc. It is then easier for the model to learn the information.

4.2.2 Tokenization

The NLP pipeline starts with tokenization. This has significant effects on the remainder of the pipeline. Tokenization divides texts written in natural language and unstructured data into discrete information chunks. Tokenization is a method for dividing clauses, words, characters, and subwords. Tokenization is done in the following sequence:

- Tokenize the text into sentences.
- Tokenize sentence into words.
- Tokenize regular expression sentences.
- The output can be used as input to other algorithms.

After applying the following pre-processing techniques, all the unnecessary data is removed like retweets, links, user id's, symbols, numbers, hashtags etc. All the uppercase words are converted to lowercase for further simplification. The dataset is transformed from unstructured data to a stream of textual data containing string that holds valuable data. The next step in the process is feature extraction.

4.3 Applying BERT, Lemmatizer

Feature Extraction techniques can be used to reduce the number of features in a dataset by creating new features from the currently existing features and then replacing them. The new reduced features contain most of the crucial information of the original set of features. We applied 2 feature extraction methods here:

- BERT for Mixed models
- Lemmatizer for Deep Learning models.

4.3.1 BERT

BERT is trained utilising a multi-task learning approach and a huge corpus of text data, such as Wikipedia and the Book Corpus. By anticipating the input text's hidden words and figuring out how two sentences relate to one another in each situation, it learns to represent a sentence's meaning. One of the BERT's key characteristics is that it is bidirectional, which means that it considers the context of both the words that come before and after a sentence. This leads to a more accurate representation of the sentence's meaning.
It has 2 main stages:

- Pre-training
- Fine-Tuning

4.3.2 Lemmatizer

A lemmatizer is a piece of software that breaks down words into their lemma, or root, form. Lemmatization entails identifying a word's canonical form, which is the one that is commonly found in a dictionary or glossary, by studying the morphology of the word. In natural language processing (NLP) applications, lemmatization is frequently used to enhance text

analysis and information retrieval. Lemmatization can assist standardise and normalise text data by breaking down words into their simplest forms, making it simpler to handle and analyse.

4.4 Algorithms

We applied GCN with BERT, LSTM+GCN with BERT, LSTM, CNN, RNN, GRU, KNN,SVM, Voting Classifier, Random Forest, Decision Tree algorithm to train our model and later on predict with it. At last we got LSTM as highest accuracy model and we predict with it connecting it to front end using Flask Framework.

4.4.1 GCN

The processing of data with a graph structure, citation networks, protein-protein interaction networks, and social networks, can be done using the Graph Convolutional Network (GCN) model, a kind of neural network. A convolution operation is carried out on a graph structure using GCN, much like a convolutional neural network (CNN) does with image input. However, GCN can accommodate asymmetric graph architectures, in contrast to CNNs, which work with a normal grid structure. The GCN model has several layers, each of which does a graph convolution process. A feature vector is used to represent each node in the graph, and it is updated in each layer based on the features of the nodes around it. A weighted sum of the characteristics of the neighboring nodes, with the weights obtained during training, is commonly used to update the feature vectors. The ability of GCN to learn representations of graph nodes that include both local and global information about the network topology is one of its main features. Multiple GCN layers are stacked to do this, with each layer being able to capture progressively more intricate characteristics of the graph. It has demonstrated promising performance in various tasks, particularly when the underlying data heavily relies on the graph structure as shown in figure 2.

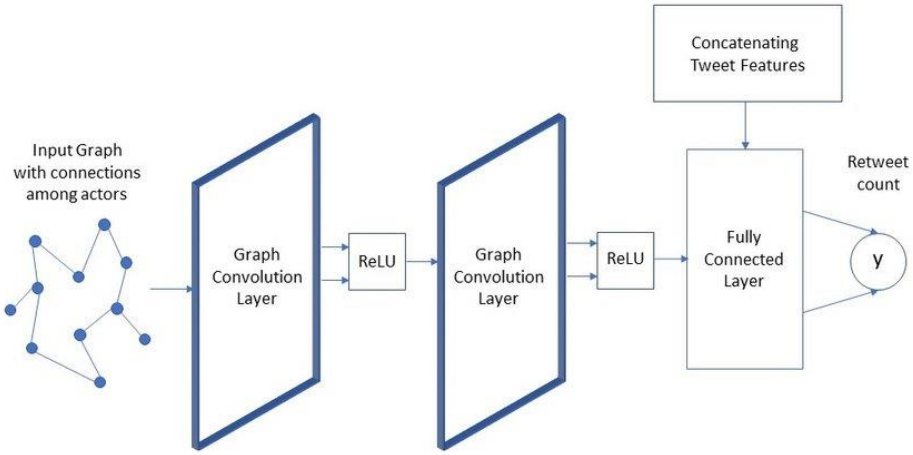

Fig. 2. Graphical Convolutional Network

4.4.2 Voting classifier

An example of an ensemble learning technique used in machine learning is the voting classifier, which combines the predictions of various separate classifiers to provide a final prediction. Each classifier in a voting classifier is trained using a separate algorithm or a different subset of the features on the same dataset. Each classifier outputs a class label or a probability distribution across the class labels when making a prediction. The voting classifier then uses a vote system to aggregate these predictions and arrive at a final prediction. A voting classifier works on the principle that by combining the predictions of various classifiers, we may lower the variance of the predictions and raise the model's overall accuracy. It is of 2 types:

- Hard Voting
- Soft Voting

4.4.3 LSTM

Information can remain thanks to LSTM, also known as an improvised RNN or sequential network. It can resolve the RNN's vanishing gradient issue. RNNs are employed for permanent memory. Like how RNNs function, they retain the prior knowledge and use it to process the incoming data. Due to declining gradients, RNN has the disadvantage of being unable to recall long-term dependencies. Long-term dependency issues are specifically avoided when building LSTMs.

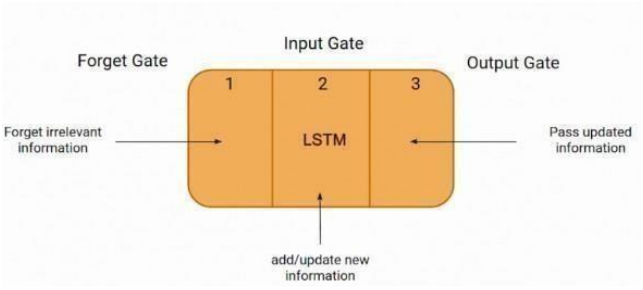

Fig.3. Long short term memory

It contains 3 gates:

- Forget Gate
- Input Gate
- Output Gate

4.4.3.1 Forget Gate

An LSTM network cell's first action is to decide whether to keep or throw away the information from the previous timestamp. Below is the forget gate equation.

Forget Gate:

$$f_t = \sigma\,(x_t * U_f + H_{t-1} * W_f) \qquad (1)$$

X_t = Timestamp's input
U_f = input's weight
H_{t-1} = Previous timestamp's hidden state
W_f = Hidden's state weight matrix

A sigmoid function is then applied. Consequently, ft will become a number between 0 and 1. As seen below, this ft is later multiplied by the cell state of the preceding timestamp.

$$C_{t-1} * f_t = 0 \ldots.\text{if } f_t = 0 \text{ (forget everything)} \quad (2)$$
$$C_{t-1} * f_t = C_{t-1} \ldots\text{if } f_t = 1 \text{ (forget nothing)} \quad (3)$$

4.4.3.2 Input Gate

The input gate's job is to retain crucial information. The value of the fresh information is measured by the input gate. The input gate's equation is shown below:

Input Gate :

$$i_t = \sigma (x_t * U_i + H_{t-1} * W_i) \quad (4)$$

X_t = Timestamp's input
U_f = input's weight
H_{t-1} = Previous timestamp's hidden state
W_f = Hidden's state weight matrix
A sigmoid function is applied and consequently, l will have a value between 0 and 1 at timestamp t.

4.4.3.3 Output Gate

Now the model is learned based on the training data we have passed. Based on that, the model predicts the output for the input when passed by the user. Output's gate equation is:

Output Gate:

$$o_t = \sigma (x_t * U_o + H_{t-1} * W_o) \quad (5)$$

Due to sigmoid function, it will also have a value between 0 and 1. Now, using Ot and the updated cell state's tanh function, we will derive the current hidden state. as displayed below:

$$H_t = o_t * \tanh(C_t) \quad (6)$$

It becomes out that the concealed state depends on both the present output and long-term memory (C_t). Simply activate SoftMax on hidden state H t if you need to take the output of the current timestamp.

$$\text{Output} = \text{SoftMax}(H_t) \quad (7)$$

Prediction is the token in this case having the highest score in the output. We predict the output based on the SoftMax function i.e., the token with maximum score (probability) will be shown in the output.

4.5 Workflow and Algorithm

1. First, the necessary libraries are imported, including TensorFlow, NumPy, Pandas, and Matplotlib.
2. The datasets are loaded.
3. The datasets are first pre-processed for ensemble models and once done, we build the GCN with BERT, LSTM + GCN with BERT architectures and train them.
4. Then we repeat the process for DL algorithms ie.; LSTM, CNN, RNN, GRU. We use Lemmatizer here to convert the words into vectors.
5. Then we repeat the process for ML algorithms ie.; KNN, SVM, Voting Classifier, Random Forest.
6. Then we display a bar graph by comparing the accuracy of models. We get highest accuracy for LSTM model.
7. We save the LSTM model and then connect it to front end using Flask framework, where the user can register, sign-in and interact with the model to predict his input.

5 Results Analysis

5.1 Pre-Processing

Displays the pre-processed datasets. It doesn't contain any special characters, mails, uppercase etc.

```
#Removes Punctuations
def remove_punctuations(data):
    punct_tag=re.compile(r'[^\w\s]')
    data=punct_tag.sub(r'',data)
    return data

#Removes HTML syntaxes
def remove_html(data):
    html_tag=re.compile(r'<.*?>')
    data=html_tag.sub(r'',data)
    return data

#Removes URL data
def remove_url(data):
    url_clean= re.compile(r"https://\S+|www\.\S+")
    data=url_clean.sub(r'',data)
    return data
```

Fig. 4. Pre-processing datasets

5.2 Accuracy Prediction

```
import matplotlib.pyplot as plt2
plt2.barh(y_pos, score, align='center', alpha=0.5,color='blue')
plt2.yticks(y_pos, classifier)
plt2.xlabel('Accuracy Score')
plt2.title('Model Performance')
plt2.show()
```

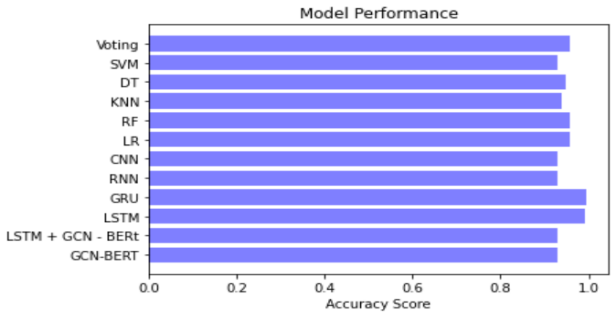

Fig. 5. Comparing accuracy of models

5.3 Output

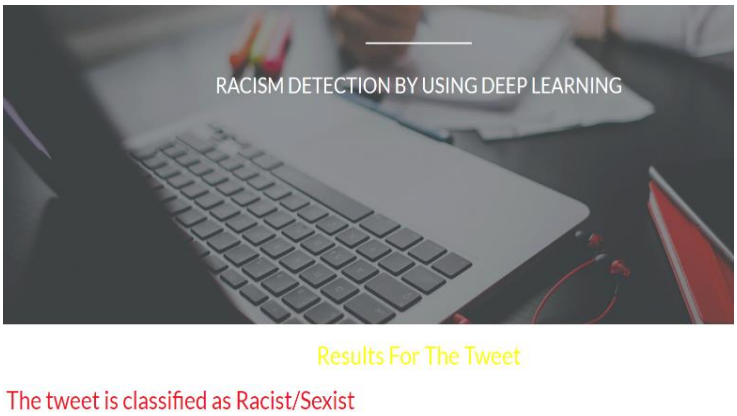

Fig.6. Output predicted for user input

6 Conclusion

On social media platforms like Twitter, racist comments are more prevalent, and it is important to automatically identify and block them in order to halt their spread. In this study, racism is detected using sentiment analysis to identify tweets that include racist content by identifying unfavorable feelings. Social media now dominates socio-political outlooks and influences our thoughts and behaviours in a variety of ways. Racism is one of the most prevalent vices to have evolved in recent years due to the widespread usage of social media platforms around the globe and the freedom of speech. Social media platforms like Twitter provide a brand-new environment where racism and its associated tension appear to be thriving.

We have developed a model that divides tweets into racist and non-racist categories. Models for the Voting Classifier included CNN, RNN, LSTM, GRU, RF, SVM, KNN, DT, and GCN with BERT. As the model with the highest accuracy, we chose LSTM and saved it for prediction. The Lemmatizer technique was used to pre-process the data sets after they were downloaded from Kaggle. We use VS code and Jupyter Notebook as our technology in this case. The layout is efficient and useful, so everyone will gain from it. We are happy with the results of our application so far and appear to have achieved our objectives.

References

1 Aggarwal and C. Zhai, "A survey of text classification algorithms" in Mining Text Data, New York, NY, USA:Springer, 2012
2 Mousa and B. Schuller, "Contextual bidirectional long short-term memory recurrent neural network language models: A generative approach to sentiment analysis", Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1 Long Papers, pp. 1023- 1032, 2017
3 S. Perone, "Machine learning: Text feature extraction (tf-idt) - Part 1", pyevolve.sourceforge.net, September 2011

4 Huang Wenliang, Li Shijian and Liu Juxin. A large-scale online spam short message filtering system. Beijing Youdian Daxue Xuebao/Journal of Beijing University of Posts and Telecommunications, v 31, n 3, June 2008, p 33-37
5 PuniÅ¡kis, R. Laurutis and R. Dirmeikis, An Artificial Neural Nets for Spam e-mail Recognition electronics and electrical engineering, vol. 5, no. 69, 2006, ISSN 1392-1215
6 Sebastiani, "Machine learning in automated text categorization", ACM computing surveys (CSUR), vol. 34, no. 1, pp. 1-47, 2002.
7 J.R.Nolan, "Estimating the true performance of classification-based nlp technology", Workshop On From Research To Commercial Applications: Making NLP Work In Practice, pp. 23-28, 1997.
8 S.Ramasundaram, “Text Categorization by Backpropagation Network”, , International Journal of Computer Applications, Vol., no.6, October 2010.