Twitter Sentiment Analysis for Hate speech detection using a C-BiLstm model

Vivek Giradharbhai Nathani (0698871), Harshil Ketan Patel (0698350)

I. ABSTRACT

Due to users' freedom and anonymity, as well as the absence of regulation imposed on social media platforms, the spread of hate speech and harassment in online communication is increasing as the volume of internet information expands. The hate speech focuses on the offensive, sexism, racism, religion, and other topics, and it poses a threat to public safety in the communities. Even though most social networks and micro-blogging websites forbid the use of hate speech, the size of these networks and websites makes it almost impossible to control all of their content. Therefore, the necessity arises to detect such speech automatically and filter any content that contains hateful language.

With the expansion of deep learning, quite a lot of researchers have inclined toward using their deep neural networks for abundant discipline. Even for natural language processing (NLP)-based tasks, deep networks, specifically recurrent neural networks (RNN), and their types are lately being considered over the traditional shallow networks. In the present paper, we study this complex problem by following a more holistic approach, which considers the various aspects of abusive behavior. We focus on Twitter, due to its popularity, and analyze tweets from different angles of abusive behavior.

We propose a LTSM based classification system that differentiates between forms of hateful speech. This system describes a contemporary approach that employs the C-BiLstm [16] model, a combination of the Bidirectional long short-term memory (BiLSTM) and the CNN architecture. Long-term sentence dependencies are recorded by the BiLSTM before CNN is used to extract features. With a constant initial learning rate of 0.8 and a dropout ratio of 0.5, the model achieved an accuracy of 87% with an early stopping criterion based on loss function during training.

Keywords: Sentiment analysis · NLP · Deep learning · Hate speech · Offensive language · Bi-LSTM · LSTM · Twitter · CNN

II. INTRODUCTION

Online social networks (OSN) and micro-blogging websites are attracting internet users more than any other kind of website. Their contents are rapidly growing, constituting a very interesting example of the so-called big data. Big data has been attracting the attention of researchers, who have been interested in the automatic analysis of people's opinions and the structure/distribution of users in the networks, etc. While these websites offer an open space for people to discuss and share thoughts and opinions, their nature and the huge number of posts, comments, and messages exchanged makes it almost impossible to control their content. Furthermore, given the different backgrounds, cultures, and beliefs, many people tend to use aggressive and hateful language when discussing with people who do not share the same backgrounds.

Words have energy and power with the ability to help, to heal, to hinder, to hurt, to harm, to humiliate, and to humble." [15] Indeed, words are the most powerful of all. Often social media services such as Facebook and Twitter were criticized for not having a strict measures concerning the usage of toxic speech.

EU defines hate speech as: [13]

> All conduct publicly inciting violence or hatred directed against a group of persons or a member of such a group is defined by reference to race, color, religion, descent or national or ethnicity.

Hate speech and offensive language differ in the subjectivity of attack. Also, different countries have different legislative norms when dealing with this issue. From the past decade, several sentiment analysis studies have been conducted related to hate speech in different social networking and micro-blogging sites like Facebook, Twitter, Reddit and YouTube. In this work, a sentiment analysis classification system is developed that utilizes the concept of deep learning to study the occurrence of hate speech on Twitter.

Text classification models have been heavily utilized for a slew of interesting natural language processing problems. Like any other machine learning model, these classifiers are very dependent on the size and quality of the training dataset. Insufficient and imbalanced datasets leads to poor performance.

In [23], the author used ConceptNet and Wikidata to improve sexist tweet classification by two methods **(1.) text augmentation** and **(2.) text generation** in order to increase the size of the training set, preserve the label, increase diversity, and boosts the performance of the classifier.

In our study we first try to improve the dataset by text augmentation method, and improve the classification model by the use of a multi-step classifier, and word-embeddings with a deep learning based neural network.

The study contributes to the field in a four-fold manner as:

- Augment and generalize the dataset by combining multiple annotated datasets available online for tweets.
- Propose a methodology that employs deep neural networks for textual data to learn deep features and can later be used for multi-class classification of tweets as hateful, offensive, abusive, spam, or normal.
- Investigate the suitability of adapting pre-trained word-embeddings (GloVe word-embeddings) to individually convert every single token into vectors better known as an embedding matrix. Detection of Hate Speech and Offensive Language in Twitter Data
- Experiment with stacked CNN, BiLSTM and a BiLSTM-CNN networks and delineate their effect on results.

III. Motivation and Related Work

A. *Motivation*

Hate speech is a particular form of offensive language where the person using it is basing his opinion either on segregative, racist or extremist background or on stereotypes. *Merriam-Webster* defines hate speech as a **"speech expressing hatred of a particular group of people.'**, whereas an online dictionary defines hate speech as

> A speech that attacks a person or a group based on protected attributes such as race, religion, ethnic origin, national origin, sex, disability, sexual orientation, or gender identity.

Hate speech is considered a world-wide problem that many countries and organizations have been standing up against. With the spread of the internet, and the growth of online social networks, this problem becomes even more serious, since the interactions between people become indirect leading to an enormous amount of opinionated data available on the web. This data serves well in the area of sentiment analysis, text analysis, big data or data mining. Many a time, people post hate speech or use offensive language to express their views. These kinds of posts on social media may be hurtful to some people of certain religion or race or gender. The problem of unfiltered hate speech has also encouraged turning up group-based hatred against some minorities.

For such reasons, websites such as Facebook, Youtube and Twitter prohibit the use of hate speech. However, it is always difficult to control and filter all the contents. So, hate speech identification has become a significant task of sentiment analysis. Once hate speech is detected, the respective organization can then decide how to deal with them.

In the research field, hate speech has been subject to some studies, trying to automatically detect it. Most of these works on hate speech detection have goals such as the construction of dictionaries of hate words and expressions [20] or the binary classification into **"hate" and "non-hate"** [25]. However, it is always difficult to clearly decide on a sentence whether it contains hate or not, in particular if the hate speech is hiding behind sarcasm or if no

clear words showing hate, racism or stereotyping exist.

This makes the task of hate speech detection quite different and more challenging than sentiment analysis: not only is it context-dependent, but also, we should not rely on simple words or even n-grams to detect it. In a related context, writing patterns have proven to be effective in text classification tasks such as sarcasm detection [6], [7], multi-class sentiment analysis [5] or sentiment quantification [4]. Through this work, we try to extract patterns of hate speech and offensive texts using a deep learning approach, and use these, along with other word-embedding features to detect hate speech in short text messages on Twitter.

B. Related Work

The extraction and analysis of text-based data have emerged to be an active research field. Owing to the global availability of such data, text analytic has acquired a lot of attention. Several studies are being conducted on these data over the past decade; the only differences are the methods used and the targeted domain. Table I presents a brief investigation of previous work carried out for hate speech detection.

As seen in Table I, all the studies were implemented on Twitter data. The reason is the effortless availability of tweets that can be crawled using the Twitter API. Out of all, the majority of the research focuses on the identification of hate speech and differentiating them with non-hate (or offensive) texts.

Citations	Classes	Model	Dataset used
Waseem [26]	3 classes (Sexism, Racism, None)	Empirical	Twitter
Park et al. [18]	3 classes (Sexism, Racism, Neither)	CNN	Twitter
Watanabe et al. [28]	3 classes (Sexist, Racist, Neither)	SVM, CNN + LSTM	Twitter
Mathur et al. [17]	2 classes (Hate speech, Abusive)	CNN-LSTM	Twitter
Wiedemann, et al. [29]	2 classes (Offensive, Other)	BiLSTM – CNN	Twitter

TABLE I: Existing research in Hate speech Detection.

The authors in this study [2] addresses the problem of hate speech hovering on social media by combining multiple datasets and generating an experimental dataset with balanced classes to avoid unbiased training. The goal of the present work is to overcome the limitations of the baseline models: (1) effect of imbalance data on the results and (2) improving accuracy for LSTM and BiLSTM classifiers. They developed a classification model to evaluate the data and categorize them into different classes allowing the model to understand the sentiments of a variety of sentences. Hate speech detection is a prominent application of sentiment analysis. Thus they experimented with a variety of LSTM and BiLSTM models — simple deep neural networks and stacked networks to achieve the desired goal and their best models acquired an **"accuracy of 86% for LSTM and 80.1% for BiLSTM."**

In some cases, CNNs, when used with word embeddings, have also emerged as a possible solution for the classification of toxic content. Thus Following the work by Collobert et al. (2011) [21], the authors of this study [3] employs a Convolutional Neural Network (CNN) model with the word vectors that are merged with a set of extracted features, downsized using max-pooling, and brought together with character n-grams (4-grams) fed to the neural network model to predict the categories of each tweet (racism, sexism, both, and non-hate)

They experimented 4 approaches to hate-speech classification, based on different feature embeddings and the dataset created by Waseem (2016) [27]. The average 10-fold cross-validated results for all four models were shown, and compared to the Logistic Regression (LogReg) model used by Waseem and Hovy (2016) [27]. It was observed that all CNN models convincingly outperformed Logistic Regression in terms of both precision and F 1 -score, while the LogReg model achieved better recall than all the neural network models. **"The model developed with combination of word2vec + character n-grams showed better precision (86.61%), and recall (70.42%) with an F-score of 77.38%."**

In a subsequent study [28] the authors proposed an approach to detect hate expressions on Twitter using an approach based on collecting unigrams and patterns from the training set. These patterns and

unigrams are later used, among others, as features to train a machine learning algorithm. To evaluate the performance of classification, they used 4 different key performance indicators (KPIs) which are the percentage of true positives, the precision, the recall and the F1-score. **The results showed an accuracy equal to 87.4% on detecting whether a tweet is offensive or not (binary classification), and an accuracy decreased to 78.4% on detecting whether a tweet is hateful, offensive, or clean (ternary classification).**

The authors [9] of this study states that metadata information of a tweet like punctuation, hashtags, and tagged users are really important. So they followed a more **"holistic"** approach to treat **a.)** raw text, and **b.)** domain specific metadata, separately at first, and later combining them into a single model. After experimenting with several choices for the Recurrent Neural Networks (RNN) architecture (Gated Recurrent Unit or GRUs, Long Short-Term Memory or Long Short-Term Memory Networks (LSTMs), and Bidirectional RNNs), they found out that simple GRUs are performing better than a complex units.The best performance is reached when all attributes are used.

This demonstrates the fact that the metadata information does not overlap the information that can be extracted from raw text, and this is why the proposed model can be quite powerful and outperforms the state-of-art models for these tasks. They achieved an accuracy of 93% and an AUC of 89%.

The work present in this paper is more in line with the citations of Table II. This study will try to identify the tweets with hateful language and classify them into offensive, abusive, normal, (hateful - racism and sexism) classes.

Citations	Classes	Model	Dataset used
Gröndahl et al. [12]	3 classes (Hate, Offensive, Ordinary)	LR, MLP, CNN + GRU, LSTM	Twitter
Gao and Huang [11]	3 classes (Hate, Offensive, Clean) 6 classes (Toxic, Obscene, Insult, Hate, Severe Toxic, Threat)	CNN, LSTM, Bi-LSTM, Bi- GRU	Twitter, Wikipedia
Davidson et al. [8]	3 classes (Clean, Offensive, Hateful)	RF, LR, SVM, NB	Twitter

TABLE II: Existing research in Hate speech and offensive language Detection.

IV. Research Methodology

Deep learning technology is increasingly and extensively used for text classification, gradually taking the role of traditional machine learning techniques. Deep learning can automatically learn the characteristics of objects from vast amounts of data and more correctly express them. When classifying texts using a deep learning approach, convolutional neural networks (CNN) and long short term memory networks (LSTM) are widely used. One type of multi-layer neural network is convolutional neural networks designed to enhance error back propagation.

Thus we are using sentiment analysis on Twitter data for Hate speech detection. Various steps of the analysis are briefly explained below. Later, in Figure 1 we present a system architecture with BiLSTM-CNN based classifier used in this study. Also an overall structure of the model is mentioned in Figure 4, that describes the input layers, hidden layers, activation function, and output layer.

A. *Dataset source and description*

Data Name	Classes	Approx tweets (in thousands)
WZ-LS [27]	racism, sexism, both, and neither	15
DT [8]	offensive, hate, and neither	24
FOUNTA [10]	normal, spam, hate, and abusive	90

TABLE III: Data Source

B. *Data Gathering*

Dataset quality is mandatory with proper labels that can train the network with better learning. To avoid wastage of time, the study is inclined towards pre-annotated Twitter datasets that are publicly available, we are just combining and generalizing the labels.

The dataset from Crowdflower website [10] contains tweets crawled from hatebase.org. It contains

approximately 100K tweets with four labels as—abusive, normal, hate, and spam.But, after analyzing the entire dataset, it was found that the dataset has a strong class imbalance for "hate speech" tweets. On average, only 6% of the entire dataset has tweets with the label "hate" compared to the other three classes. For that reason, here another dataset is also considered which is available on github [8] by Davidson and Thomas, which contains Tweets that are labeled 3 major classes as – hate speech, offensive, and neither.

Adding onto this, we gathered some tweets using APIs from the tweet ID given in the data [27] also we merge the above data with the dataset provided in SemEval2019 [1] which are annotated and labeled into 3 classes as – Hate speech, Targeted, Aggressive.

After combining all the datasets and removing the tweets labeled **"Targeted"**, we generalized the labels into fewer classes as – **offensive, abusive, normal, hateful, and spam**. After a lot of manipulation, the three data sets were combined to make a bigger data set and with generalized labels, to make the dataset balanced and perform the task of classification.

After pre-processing, the final dataset consists of approx 94K samples with label distribution described in

label	total tweets	distribution in data
abusive	27150	29%
hateful	9422	10%
normal	16048	17%
offensive	20429	22%
spam	21205	22%

TABLE IV: Data Distribution

C. *Data Pre-processing*

The data that is collected might contain a lot of junk and irrelevant data. So, the pre-processing of the data is required to enhance the data. The very first step of pre-processing was to remove redundant data as we merged 3 data and created new data for this study.

While cleaning the tweets data, we followed the below steps,

1) Removal of URLs from the tweets (which starting either with "http://" or "https://")
2) Removing tags (i.e., **"@user"**) and irrelevant expressions (words written in languages that are not supported by ANSI coding).
3) Removing stop-words, punctuation's, and mapping contractions
4) Converting the tweets to lower-case
5) Decomposing the hashtags into words
6) **Tokenization of data** - Before feeding any text to the network, we need to transform each sample to a sequence of words.
7) **Stemming of words** - Group together with the words that root out from a similar word. For example, "presented" and "presenting" are all the stems of "present."

As neural networks are trained in mini-batches, every sample in a batch must have the same sequence length (number of words). Tweets containing more words than the sequence length are trimmed, whereas tweets with fewer words are left-padded with zeros (the model learns they carry no information). Ideally, we want to avoid outliers as they waste resources (feeding zeros in the network), making the training of the network slower.

Therefore, we take the **95th percentile** of length of tweets (with respect to the number of words) in the input corpus as the optimal sequence length. For tweets, this results in **sequences of 21 words** (in effect, 5% of tweets that contain more than 21 words are truncated).

We additionally remove any words that appear only once in the corpus, as they are most likely typos and can result in over-fitting. Once pre-processed, the input text is fed to the network for learning.

D. *Feature Extraction*

Operations on single strings like dot products or back propagation cannot be done. Therefore, instead of a string, they are converted into vectors (typically 25-300 dimensions) and fed to them for the task. The most popular pre-trained word-embeddings are Word2Vec [22] and GloVe (global vectors).

We used pre-trained word embeddings from GloVe [14], which is constructed on more than 2 billion tweets. We choose the highest dimension embeddings (128) available.

E. Proposed System Architecture

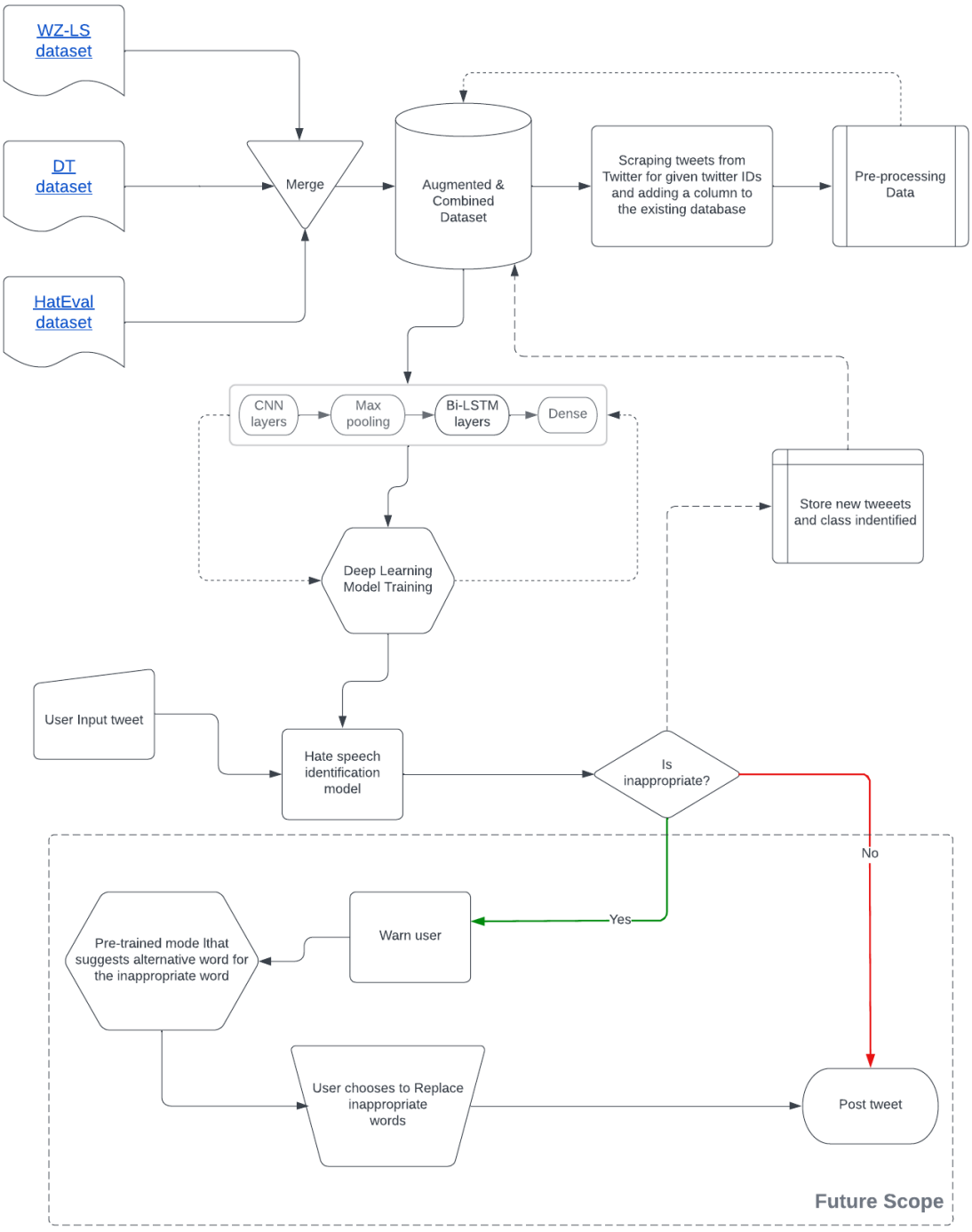

Fig. 1: System Model

F. Deep Learning for Text Analysis

The classification model can be trained using a machine learning algorithm or a deep neural network. This study inspired from [19] utilizes the concept of deep learning for training and classification. For sequential data, in general, RNNs are used which can further be an LSTM model. Given below is a brief understanding of RNN and LSTM deep neural networks.

*1) **Convolution Neural Network (CNN):*** CNN is a multi-layer feed-forward neural network which improves the error in back-propagation network (BP) and reduces computation time and complexity of BP. It is recently used for sentiment classification because it can recognise local features by using convolution kernel, and automatically learns these features for classification solution.

CNN model consists of three main layers; **convolution layer, pooling layer, and fully connected layer** [30]. Figure 2 shows the stages of CNN structure for text classification. Sentences are converted into a matrix of numbers and input to the convolutional layer. Each sentence consists of words or tokens, and each token is corresponded to a row or vector on the matrix table. These vectors are typically generated by embedding techniques such as the Word2Vec and GloVe model.

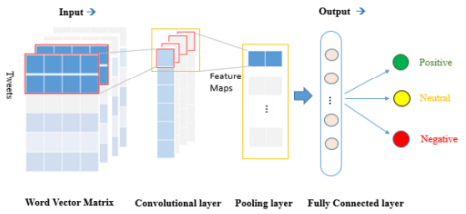

Fig. 2: CNN Architecture

CNN model takes the input of vectors and extracts local feature using filters. The most computations of features are performed in convolutional layer which is the most important layer in CNN. **Convolutional layer** produces feature maps using a function called convolution kernel. After the convolution operation, pooling layer extracts the most important features.

The **Pooling layer** calculates local sufficient statistics. This process allows the pooling layer to reduce feature dimensions, makes CNN achieve computational time and cost reduction, and prevents the model from overfitting problem. Lastly, the **Fully connected layer** produces a probability distribution to classify sentiment results.

*2) **Recurrent Neural Network (RNN):*** In NLP, word and sentences are analyzed, where each word in a sentence depends upon the word that comes before and after it. For such dependency, we have a recurrent neural network (RNN). The RNN is slightly different from the long-established feed-forward NN we know about. We know that a feed-forward network comprises input nodes, hidden units and output nodes. RNN differs from the feed-forward neural network because of its temporal aspects. In RNNs, every word in an input sequence will be related to a definite timestamp. And the total number of timestamp will be the maximum length of the sequence.

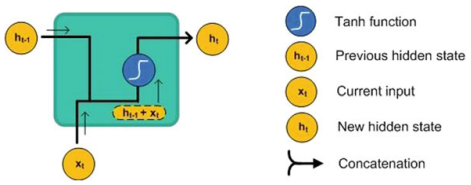

Fig. 3: RNN Cell

Each timestamp(RNN cell) shown in Figure 3 is associated with a hidden state vector h_t and and the input value x_t . This vector summarizes and encapsulates the data from the previous timestamp. This hidden state functions as a current state word vector and the previous timestamp has hidden state vector as well.

Although RNNs seem to work brilliantly for short sequences, for long sequences its other variants are considered. The long short-term memory (LSTM) and grated recurrent unit (GRU) are two variants of RNN created to resolve the problem of short-term memory and vanishing gradients. Both of them have gates as an internal mechanism that helps in regulating the flow of the network.

*3) **Long Short-Term Memory Units (LSTM)**:* Long short-term memory (LSTM) networks are a particular kind of RNN capable of analyzing and learning long-term dependencies. LSTMs are explicitly considered to deal with the long-term dependency problem. They can easily remember any piece of information as long as it is required, best for while working on a sequence of sentences. Long short-term memory units are modules that can be used inside of recurrent neural networks. At an advanced level, it makes sure that h_t can encapsulate information about long-term dependencies in the text.

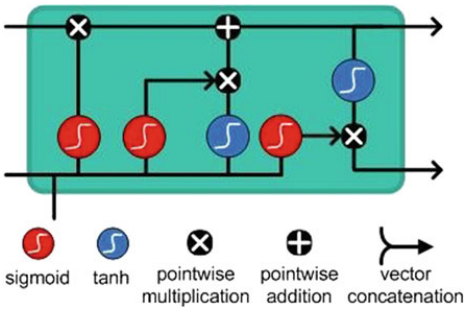

Fig. 4: LSTM Cell

An LSTM cell Figure 3 is similar to the RNN except for the gates and cell state. LSTM has three gates—input gate, forget gate and output gate. Each gate has its purpose and computes different operations. The gates in LSTM contain sigmoid activation function.

As described earlier, tanh activation squishes the value between -1 and 1. Similarly, a sigmoid activation squishes the value between 0 and 1. This helps in deciding whether to update or forget data. This helps LSTM in overcoming the vanishing gradient and exploding gradient problem which appears in the back-propagation.

*4) **Bi-directional Long Short-Term Memory Units (Bi-LSTM)**:* BiLSTM is a variant of LSTM network that works both in the direction of the network. In this, the network flows in feed-forward as well as in a feedback loop for the units. This is quite beneficial when dealing with natural languages as in a sentence the meaning of each word depends on its neighboring words as well. A meaning of a word can change on account of its surrounding words and the overall sense of the sentence. This is why it is better that the network is trained in both ways.

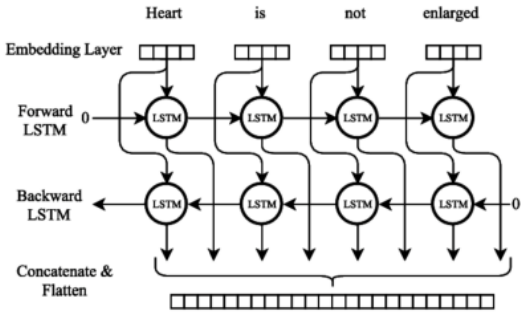

Fig. 5: BILSTM Architecture

G. Model Analysis

The first four layers of the text categorization model based on CNN and LSTM or its version can be regarded as the input layer, convolutional network layer, LSTM or its variant layer, and softmax classifier layer. In this study, both subjective and objective text data are used. The basic model structure is shown below:

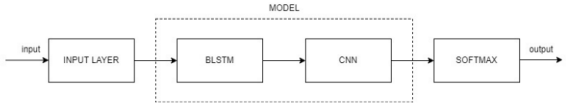

Fig. 6: Short Hand Model Architecture

A classification model is developed to evaluate the data and categorize them into different classes. This allows the model to understand the sentiments of a variety of sentences. In this proposed classification system, we have developed a selection of CNN combined with BiLSTM-based classifiers [24]. Our classifier has five base layers shown in Figure 7 that are:

- **Sequence Input Layer** - This layer takes the sequential input value, in our case, the embedding matrix of dimension 128.

- **Convolution Layer** - CNN model is good at extracting the most important words from tweets or sentences and the convolution layer is the main step in CNN model. The word vectors matrix from word embedding layer are fed into one dimensional convolution layer. In one-dimensional convolution layer, the convolution word vector matrix is calculated through N filters and width q of convolution kernel to construct the local feature of n-gram.The second layer is the LSTM or BiLSTM layer.

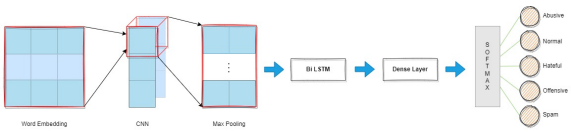

Fig. 7: CNN-BILSTM Architecture

- **Max Pooling Layer** - Once convolution operation produces feature maps, pooling layer then extracts the most important features to calculate the local sufficient statistics. One-dimensional max-pooling converts each kernel size of input into a single output of the maximum number to reduce or down-sample version of the input. This is the reason why CNN model effectively reduces the number of features to prevent overfitting, also reduces time and complexity of parameters.
- **BILSTM Layer** - Bi-LSTM allows the information to flow in both directions; backward to forward and from forward to backward by using two hidden states. This structure helps the network to retain preceding and succeeding information. The sequence output of the first layer in Bi-LSTM is the input of the second layer, and the sequence output of the second layer is the concatenation of the last unit output of forward and backward layers
- **Dense Layer and Result** - Dense layer is used in the model to connect each input with every output by using weights. Softmax is a function used in the final layer to produce the output. It calculates the possibilities of each possible class. The result of tweet is classified into either of the classification by using binary cross-entropy. The output layer is mapped with the classification layer that provides the predicted value as hate speech, offensive language or spam, normal, abusive.

H. Parameters Optimization

In many cases, the model may produce less accuracy or even produce overfitting or underfitting. To obtain high model performance, conducting hyper-parameters tuning is very critical. Therefore, the randomised search strategy was used to tune hyper-parameter and optimise the accuracy.

For the model training, the dataset was divided into 75% and 25% in training and validation respectively. The maximum length of tweets was set to 21.

Parameters	**Values**
Embedding Dimension	128
Kernel & Pool size	3
BILSTM Output size	128
Activation	Softmax & Relu
Recurrent Dropout	0.75
Batch size	32
Vocab size	69000
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	0.01

TABLE V: Hyper Parameter settings

The experiment is built using the TensorFlow framework. The tuned model with these parameters defined in the Table V resulted in 73.5% of validation accuracy.

V. Experimental Results

The comparison tests in this study largely use the classification methods of Simple BiLSTM, CNN and a CNN-BiLSTM model. The results of the classification are displayed in the Table VI. The following are the primary measures for evaluating text classification: accuracy and loss rates. All models are trained over 50 iterations.

Model	Comparison of Experimental Results	
	Validation Accuracy	Validation Loss
Simple BiLSTM	73.84%	0.8659
CNN	72.81%	1.1763
C-BiLSTM	73.51%	0.9203

TABLE VI: Accuracy measures from different models

*1) **Results from Simple BiLstm**:* From the table above, we can see the Simple BiLSTM model has the highest validation accuracy and lowest validation loss, which makes BiLstm our best model but while looking at the training graph presented in Figure 8, we can infer that with the increase in epochs, **the model is being underfitted which makes the solution not so desirable**.

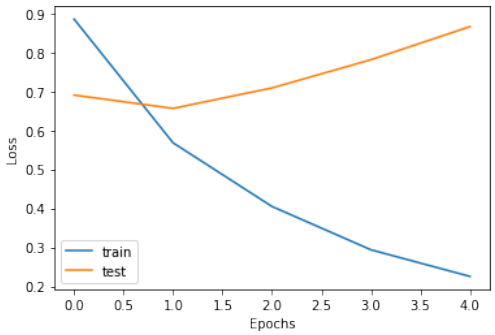

Fig. 8: Learning graph for BiLSTM

*2) **Results from CNN**:* From the table above, we can see the CNN model has the lower validation accuracy and highest validation loss, which makes CNN **underfit** the training samples as represented in Figure 9

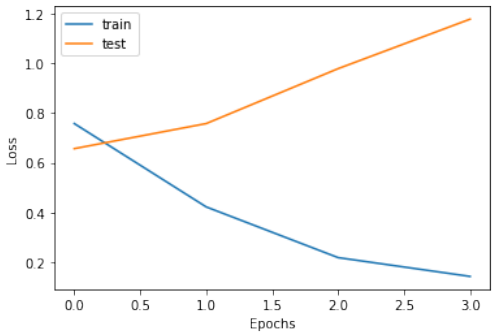

Fig. 9: Learning graph for CNN

*3) **Results from C-BiLstm**:* From the table above, we can see the C-BiLstm model has the lower validation accuracy compared to Bi-Lstm model and higher validation loss, but while looking at the learning graph we can see that our model has learnt well as compared to others as in this the validation loss remains constant after 5th Epoch. The same can be represented in Figure 10

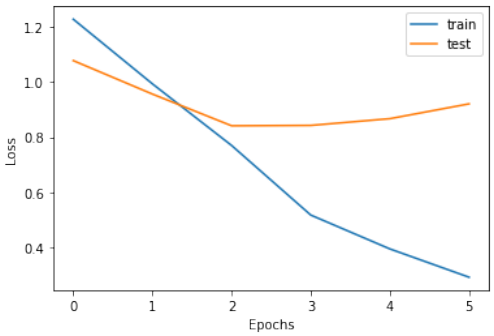

Fig. 10: Learning graph for C-BiLstm

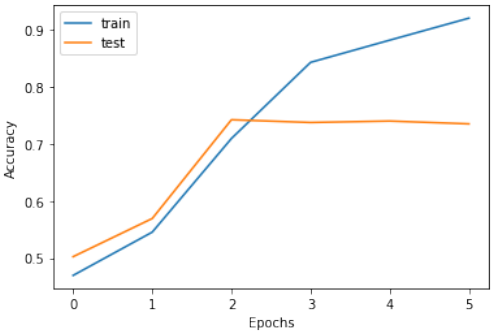

Fig. 11: Accuracy graph for C-BiLstm

Cross-validation is carried out utilising a range of test items and training objects in order to prevent overfitting during the experimental phase. The line graphs above show the loss rate of the loaded pre-training model during training for each of the three models. The accuracy graph demonstrated in Figure 11, model has learnt a good amount of information until a particular epoch, after that the accuracy remains constant and model was being overfitted, so to stop that from happening, we have applied an Early stopping function which stops the

training when the loss rate is steadily decreasing and the accuracy of the model used is steadily rising.

*4) **Comparison to Baseline model**:* The first base model is of [7] which has performed hate speech detection using a variety of techniques on parts of this dataset. Davidson and other authors have incorporated several statistical classifiers, namely logistic regression, linear SVM, Naïve Bayes and Random Forest in their work. The authors created the sentiment classifier using 25K tweets that were annotated by the workers of Crowdflower.com. Although the author has not used any deep learning models in their work, the research still serves as suitable as a performance baseline model for this work. The model achieved an overall F1 score of 91%, but a huge drawback in their work is the strong imbalance for data of class "hate speech" as compared to the other classes. Their results showed how imbalance data heavily affect their results.

Hence, the objective of this model is to improve hate speech accuracy. For that reason we created a dataset with equal data in each class. Our C-BiLSTM model performs admirably on three of the five classes with an accuracy of 73.51%, as indicated in the table and figures above. Our C-BiLSTM model performs at its best with the supplied dataset and preset custom parameters. Even if our model is not as sophisticated as others, it nonetheless produces a respectable result, demonstrating the model's versatility with various labels and lesser miss-classifications.

VI. Cost and Challenges for the model

A. Cost of the Approach

Here, the impacts of numerous elements on the performance of our model are investigated.

*1) **The length of the maxlen**:* The length of 30 words, which is thought to be the one that most closely resembles the average tweet length in the dataset, yields the best result. The accuracy will sharply decline after the maxlen exceeds the length of the tweet because the number of zero vectors in the article's vectors will rise significantly.

*2) **GPU processing cost**:* The execution time to train an epoch was approx 1500 seconds on 12 GB NVIDIA Tesla K80 GPU on TensorFlow, where early stopping was achieved at 5th epoch.

B. Challanges Encountered

Most of the challenges for text classification problems were faced during data gathering and pre-processing phase, as the data was quite imbalanced so we applied methods to gather, generalize and merge multiple datasets to produce a final data for the experiment.

Negation permits to change a word's meaning to its inverse meaning. Therefore during the extraction of features it is essential to represent the process whether or not a word is negated.

Short informal text is one of the challenges in sentiment analysis. They are restricted in length generally spanning one or less than one sentence. They tend to have several slang phrases, misspellings and shortened word forms. They also have special markers namely hashtags that are used to facilitate search but also represent a sentiment or topic. To deal with this issue, we have deleted the tweets which are classified as spam or have just one word and hence avoiding overfitting of the data.

While training the model for Bi-Lstm and CNN, we ran into the problem for underfitting, to resolve this we had to follow either of the below approach

- **Adding more data**
- **Removing features from the training sample**
- **Increasing model complexity**

We decided to go with the last step, as adding more data to our sample was not practically possible for us without scraping the twitter and manually labeling the data.

So to solve this issue, we increased the model complexity by combining Bi-Lstm with CNN to capture features more precisely and with lesser noise. This model also improved the classification accuracy of tweet content and made it more generalized since which can effectively convey text semantics and maintain more contextual information. **Thus making the C-BiLstm the best model for the given dataset under given hyper parameters.**

VII. Future Work

Through this study, we tried to work towards reducing the hate-speech content from the internet

but as just detecting hate-speech with the help of deep learning model isn't enough, we have to build a system that detects and helps user in replacing words with lesser offensive or hateful making the internet more respectful and also giving the users controlled freedom of speech.

A good system that can help in maintaining the internet hate-speech can be obtained by building a chrome extension that can possibly detect any hateful or offensive words while the user is typing it. Giving user a warning when any such text is detected before they publish it. Suggesting them with lesser offensive or hateful words to replace that with it. The same can be seen in the Figure 1, this extension will open doors to various new findings and also help in building dictionary for hateful/offensive words which will again help in improving the accuracy of the existing model.

VIII. Conclusion

Due to users' freedom and anonymity, as well as the absence of regulation imposed on social media platforms, the spread of hate speech and harassment in online communication is increasing as the volume of internet information expands. It is difficult to control and filter all the contents. So, hate speech identification has become a significant task of sentiment analysis. Most of the present works on hate speech detection have goals such as the construction of dictionaries of hate words and expressions or the binary classification into **"hate" and "non-hate"**. In this study, we aimed to improve the accuracy of hate speech detection systems with the help of deep learning concepts.

The present work tries to improve the classification by the use of a multi-step classifier and word-embeddings with a deep neural-based network. The primary goal of the present work is to develop a C-BiLstm model (Combination of Bi-direction LSTM and Convolution layer). The paper also illustrates the ability to extract characteristics and learn information from both past and future scenarios. Throughout the study, we created a class balanced dataset which was later used to train various models like CNN, LSTM and Bi-LSTM to overcome the limitations of the baseline models: effect of imbalance data on the results.

Utilising a suitable window to identify the features and setting the maximum length to a specific length(21) lead to increased performance. Later, with more such specific hyper setting parameters and the dataset, we found out that our C-BiLstm model performed better compared to others when classifying the tweets. This model achieved an accuracy of 73% and also outperformed the miss-classification for the class labelled "hate speech" compared to the baseline model.

The above work contributed to the study and analysis of the occurrence of toxic content (hate speech, offensive languages) over social media. Consequently, drawing off the curtains and increasing public transparency and making world safer and better where the users can freely direct their thoughts and opinions, detection and removal of toxic contents are reasonably essential.

References

[1] Valerio Basile, Cristina Bosco, Elisabetta Fersini, Debora Nozza, Viviana Patti, Francisco Manuel Rangel Pardo, Paolo Rosso, and Manuela Sanguinetti. SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 54–63. Association for Computational Linguistics, June 2019.

[2] Bhadauria H. S. Virmani Jitendra Kriti Bisht Akanksha, Singh Annapurna. Detection of hate speech and offensive language in twitter data using lstm model. *Springer Singapore*, 2020:243–264, 04 2020.

[3] Utpal Kumar Sikda Björn Gambäck. Using convolutional neural networks to classify hate-speech. *Association for Computational Linguistics*, 6:85–90, 08 2017.

[4] Mondher Bouazizi and Tomoaki Ohtsuki. Sentiment analysis in twitter: From classification to quantification of sentiments within tweets. In *2016 IEEE Global Communications Conference (GLOBECOM)*, pages 1–6, 2016.

[5] Mondher Bouazizi and Tomoaki Ohtsuki. A pattern-based approach for multi-class sentiment analysis in twitter. *IEEE Access*, 5:20617–20639, 2017.

[6] Mondher Bouazizi and Tomoaki Otsuki Ohtsuki. A pattern-based approach for sarcasm detection on twitter. *IEEE Access*, 4:5477–5488, 2016.

[7] Dmitry Davidov, Oren Tsur, and Ari Rappoport. Semi-supervised recognition of sarcastic sentences in twitter and amazon. *CoNLL 2010 - Fourteenth Conference on Computational Natural Language Learning, Proceedings of the Conference*, 01 2010.

[8] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. *Proceedings of the 11th International AAAI Conference on Web and Social Media*, pages 512–515, 2017.

[9] Antigoni Maria Founta, Despoina Chatzakou, Nicolas Kourtellis, Jeremy Blackburn, Athena Vakali, and Ilias Leontiadis. A unified deep learning architecture for abuse detection. *Proceedings of the 10th ACM Conference on Web Science*, 2019:105–114, 06 2019.

[10] Antigoni-Maria Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. Large scale crowdsourcing and characterization of twitter abusive behavior. *11th International Conference on Web and Social Media, ICWSM 2018*, 2018.

[11] Lei Gao and Ruihong Huang. Detecting online hate speech using context aware models, 2017.

[12] Tommi Gröndahl, Luca Pajola, Mika Juuti, Mauro Conti, and N. Asokan. All you need is "love": Evading hate speech detection. In *Proceedings of the 11th ACM Workshop on Artificial Intelligence and Security*, AISec '18, page 2–12, New York, NY, USA, 2018. Association for Computing Machinery.

[13] A. Hern. Facebook, youtube, twitter and microsoft sign eu hate speech code. [Accessed 15-Jul-2022].

[14] Richard Socher Jeffrey Pennington and Christopher D. Manning. Glove: Global vectors for word representation. *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[15] KatieMae. Words are the most powerful force available to humanity. https://medium.com/@katiemaeonline/words-are-the-most-powerful-force-available-to-humanity-5a42f09dd1ac. [Accessed 15-Jul-2022].

[16] Yue Li, Xutao Wang, and Pengjian Xu. Chinese text classification model based on deep learning. *Future Internet*, 10(11), 2018.

[17] Puneet Mathur, Rajiv Shah, Ramit Sawhney, and Debanjan Mahata. Detecting offensive tweets in Hindi-English code-switched language. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 18–26, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[18] Ji Ho Park and Pascale Fung. One-step and two-step classification for abusive language detection on twitter, 2017.

[19] Juan Carlos Pereira-Kohatsu, Lara Quijano-Sánchez, Federico Liberatore, and Miguel Camacho-Collados. Detecting and monitoring hate speech in twitter. *Sensors*, 19(21), 2019.

[20] Uritsky Razavi Amir, Inkpen Diana and Stan Sasha Matwin. Offensive language detection using multi-level classification. pages 16–27, 05 2010.

[21] L´eon Bottou Michael Karlen Koray Kavukcuoglu Pavel Kuksa Ronan Collobert, Jason Weston. Natural language processing (almost) from scratch. 2011.

[22] Xin Rong. word2vec parameter learning explained. *CoRR*, abs/1411.2738, 2014.

[23] Matwin Stan Sharifirad Sima, Jafarpour Borna. Boosting text classification performance on sexist tweets by text augmentation and text generation using a combination of knowledge graphs. *Association for Computational Linguistics*, 6:107–114, 10 2018.

[24] Sakirin Tam, Rachid Ben Said, and Ö. Özgür Tanriöver. A convbilstm deep learning model-based approach for twitter sentiment classification. *IEEE Access*, 9:41283–41293, 2021.

[25] William Warner and Julia Hirschberg. Detecting hate speech on the world wide web. In *Proceedings of the Second Workshop on Language in Social Media*, pages 19–26, Montréal, Canada, June 2012. Association for Computational Linguistics.

[26] Zeerak Waseem. Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter. In *Proceedings of the First Workshop on NLP and Computational Social Science*, pages 138–142, Austin, Texas, November 2016. Association for Computational Linguistics.

[27] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter, June 2016.

[28] Hajime Watanabe, Mondher Bouazizi, and Tomoaki Ohtsuki. Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection. *IEEE Access*, 6:13825–13835, 03 2018.

[29] Gregor Wiedemann, Eugen Ruppert, Raghav Jindal, and Chris Biemann. Transfer learning from lda to bilstm-cnn for offensive language detection in twitter. 2018.

[30] Ashima Yadav and Dinesh Kumar Vishwakarma. Sentiment analysis using deep learning architectures: A review. *Artif. Intell. Rev.*, 53(6):4335–4385, aug 2020.