International Journal of Innovations in Engineering and Technology (IJIET)
http://dx.doi.org/10.21172/ijiet.161.07

Hate Speech Detection Using BERT and CNN Integration in Deep Learning-based Text Processing

Sanjay Dahiya
Assistant Professor, Computer Science & Engineering
Ch. Devi Lal State Institute of Engineering & Technology
Panniwala Mota, Sirsa, India

Abstract: Social networks are tools that allow you to interact with people from all over the world instantly to share ideas, opinions, and other information aspects such as users' personal and professional lives. Ideally, this should lead to talks or discussions on a positive turf, but the feeling of pseudo-anonymity offered by the Internet inflames the display of the negative side of the persona of social media users, in general, the use of foul language only to cause harm or discomfort to other people in particular. Quite often it is noticed that a person who is otherwise quite sober and restrained in tête-à-tête conversation may resort to the use of an otherwise unacceptable form of expression. Although social media platforms provide features to manage and reduce the onslaught of extreme reactions, severally, these features could not be used for giving an instant reprieve because of a lack of awareness amongst users or the non-availability of user-level permission to use such features. Making social media networks less fearsome and predatory and more democratic and freer places requires devoting continuous effort to improve and develop new techniques to detect offensive, toxic, and hate speech and expression. Such systems face multiple challenges such as large data volumes abound with informal text and dialect slang in a vocabulary that evolves very quickly. In this work, natural language processing techniques have been applied to develop models of supervised learning systems using Bidirectional Encoder Representations from Transformers (BERT) and Convolutional Neural Networks (CNN) which are capable of detecting offensive, toxic and hateful comments on social media networks. Looking at the model's accuracy score of 96.43% it could be claimed that the BERT with CNN has resulted in a qualitative leap in the field of natural language processing for detecting hate speech.

I. INTRODUCTION

The social media giants have managed to sign up nearly half of the world's population to their services. The total count of global social media users is so large that growth rates have started to naturally slow down, and now, harassment and abuse on these platforms is taking its toll on administrative, legal, and social structures [1]. Recently, social media have witnessed disturbing phenomena which sometimes turn into real violence, jeopardizing the physical safety and psychological well-being of the victims on the one hand, and destroying the efforts of the social networks to socialize and global cultural openness on the other hand [2]. Social media platforms are publicly accessible digital resources for online communication and collaboration. Despite its popularity and convenience, it is increasingly being used to spread hate speech. The level of anonymity granted by Twitter makes it conducive for the dissemination of hateful speech about people. Furthermore, a proportional relationship between hate speech propagation and the occurrence of hate-related crimes is highlighted in other studies [3]. Given the high volume and nature of messages posted on Twitter, it is imperative to develop ways to curb the dissemination of hateful messages

1.1 Hate speech and its characteristics

There is no single internationally accepted definition of hate speech. But in general terms, hate speech is communication that denigrates people because they belong to a particular group. This can include any form of expression, such as pictures, plays, and songs, as well as speech. Some definitions even extend the concept of hate speech to communications that foster a climate of prejudice and intolerance. The idea here is that these types of communications can later fuel discrimination, hostility, and violent attacks [4].

The task of identifying hate speech using text data written in Indonesian is presented in this research. Some research has been done on the same problem, however it is based mostly on feature engineering and employs a classical machine learning method. Hate speech is an expression of hostility towards an individual or social group on the grounds of membership in that group, which may be based on race, ethnic origin, national origin, religion, disability, gender, or sexual orientation. can be defined [5]. The following forms of hate speech are common on social media:

- Racism - the belief that race is the most important determinant of human traits and abilities and that racial differences lead to the intrinsic superiority of certain races.

- Gender Discrimination - Prejudice or discrimination based on gender. especially discrimination against women.
- Hate Speech – language (spoken or written) used to express hatred towards a target audience or intended to belittle, humiliate or offend members of a group.
- Verbal abuse – a type of verbal abuse that hurts, offends, or upsets someone. It's mostly aggressive and often very offensive.
- Harassment – a type of abuse intended to create confusion, foment or exacerbate conflict for entertainment purposes.
-

In the present age, the Internet – owing primarily to its global and instant reach – has become the primary "space" for expressing thoughts, exchanging ideas, working on collaborative projects, indulging in group discussions, and debating on issues of political and social importance. By its very nature, the Internet is democratic means of mass communication, and probably this is why it appeals to all social groups and individuals of all ages, races, and beliefs, but mainly appeals to young people. Without trying to demonize it, the anonymity and massiveness of the Internet and social media offer, make them convenient tools for the spread of hate speech [6].

To respond to the growing problem of online hate speech, in mid-2016, four social media giants namely, Facebook, Microsoft, Twitter, and YouTube came out with a "code of conduct to combat illegal online hate speech". Google, Instagram, Snapchat, and Dailymotion also joined to adopt the code of online conduct in the year 2018. However, social media companies are removing inappropriate content ever more proactively, yet at times the measures adopted in this direction prove inadequate and significant harm is caused before such harmful content receives the attention of social media houses. At other times, social media platforms behave in a partisan manner under the pressure of government agencies. Overall, the removal rate shows that the control carried out by companies still respects freedom of expression and companies also ask other users to flag or report hateful content. Hate Speech detection methods are left to the human factor and the general perception of what constitutes hate speech, so its detection is not always objectively correct. Especially in cases of sarcasm or the use of simply colloquial abusive language (without the intention of offending anyone), it is very easy to be led to wrong conclusions. Due to the massive scale of the web, the need for scalable, automated methods of hate speech detections has grown substantially. These problems have been attracting the Natural Language Processing (NLP) and Machine Learning (ML) communities quite a lot in the last few years. Despite this large amount of work, it remains difficult to compare their performance, largely due to the use of different data sets by each work and the lack of comparative evaluations [7].

Much interest in the dissemination of hate speech is concentrated on Twitter, which is one microblogging social network. Microblogging differs from traditional blogging in terms of the length of the texts, which usually consist of short sentences. The reason for such a great appeal of this medium is the immediacy and simplicity of the messages. W. Barth, Taking Great Care: Defining Victims of Hate Speech Targeting Religious Minorities, Chicago Journal of International Law, Vol. 19, No. 1, 2018, pp.97-99, Its main feature is that the length of messages is limited to 280 characters (originally the limit was 140, which has been retained for some languages) [8].

From the foregoing discussion, it is evident that online hate speech is a stark and dark reality of social media. The ease of dissemination of information and its coverage, in many cases the anonymity, make this reality even more complex to grapple. Owing to the large variations in provisions of laws of lands, the criteria for characterizing an expression as hate speech are subjective. Accordingly, special attention and precision are required in order not to uphold the principles of freedom of expression and at the same time preserve the very nature of the Internet as a free and democratic space. This is where the present research effort fits in, for any effort to deal with hate speech shall start with its identification.

1.2 Language modelling

Language modelling is the use of various techniques for predicting the likelihood of a sequence of words appearing in a sentence. Generally, given a sequence of T words, $w_1, \ldots, w_T$, a language model assigns to follows with the probability as [9]:

$$P(w_1, \ldots, w_t) = \prod_{t=1}^{t} P(w_t | w_1, \ldots, w_t) \quad \ldots \quad (1)$$

The goal of language modelling is to provide sufficient information so that possible word sequences are more likely. Language models are useful in many Natural Language Processing (NLP) applications, especially those that have text as output. The simplest possible linguistic modelling can be affected through N-gram models. They create a probability distribution for a sequence, where the number determines the size of "gram" (one line of words). For example, if n = 4, one gram can be: "can you help me". Figure 1 shows different types such as unigrams (n = 1), bigrams (n = 2) and trigrams (n = 3) [10].

Figure 1. Examples of models with n = 1, 2, 3

To simplify the problem of estimating the language model from the data, the n-gram model assumes that each word depends only on the last $n-1$ words instead of the previous t–1 words. Thus, the probability of observing a word can be approximated by the possibility of its observation in the abbreviated historical context of the predecessors n−1 words. Therefore, the probability is calculated as follows (eq. 2):

$$P(w_1, \dots, w_t) \approx \prod_{t=1}^{t} P\left(w_t \middle| w_{t-(n-1),}, \dots, w_{t-1}\right) \dots (2)$$

where

$$P\left(w_1 \mid w_{t-(n-1)}, \dots, w_{t-1}\right) = \frac{count(w_{t-(n-1)}, \dots, w_{t-1}, w_t)}{count(w_{t-(n-1)}, \dots, w_{t-1})} \dots (3)$$

However, for large datasets, there is a data sparsity problem and its resulting model is not accurate. Data sparsity is a term that describes the phenomenon of many possible word sequences that have very few occurrences in a corpus, meaning that they are not observed enough frequently during training.

There are several ways to evaluate a language model. Perplexity (PP) is the inverse of the probability that the model assigns to the word count-normalized text corpus. The PP score of tests $W = w_1, w_2, \dots, w_N$ in an n-gram model is calculated as (eq. 4):

$$PP(W) = (P(w_1, w_2, \dots, w_N))^{-\frac{1}{N}} = \left(\prod_{t=1}^{n} P\left(w_{t-(n-1)}, \dots, w_{t-1}\right)\right)^{-\frac{1}{N}} \dots (4)$$

The better the model the lower the PP, as it indicates that the probability distribution or probability model is effective in its prediction sample. PP can also be interpreted as the weighted average coefficient branching of a language in predicting the next word. Note that the branching factor of a language is the number of possible words that can follow any word.

II. RELATED WORK

Machine learning techniques are the main approach used for automated hate speech detection. Hate speech detection can be modeled in machine learning as a dichotomous class or multiclass classification problems that can be addressed adequately using either classical learning algorithms or deep learning algorithms. Classical learning algorithms rely on manually engineered features while deep learning algorithms automatically learn features from the input data, instead of adopting handcrafted features. The unstructured nature of human language presents many intrinsic challenges to automated text classification methods. One key challenge faced by existing methods of hate speech detection is the failure to capture long-term dependencies. This leads to loss of contextual information, which is vital for semantic interpretation. Deep learning algorithms, particularly the recurrent neural network (RNN) algorithms, have been the de-facto methods in handling sequence data such as text [11, 12]. However, they have been limited in the length of sequences they can capture. Transformers are a promising way for capturing long-term dependencies in textual data.

The Skip-Gram model of [13] predicts the context of the word. The input level of the model is of size $(1 \times V)$, where V the number of words in the vocabulary. The network entry is the unique representation of the target word. This input vector is transformed by the weight matrix W, of size $(V \times E)$, and passed through a hidden layer of size

$(1 \times E)$, where E the desired embedding dimension is. The higher the value of this hyper-parameter, the more information the embeddings capture, but the harder it is to do so. Finally, the weight matrix W' of size $(E \times V)$ transforms the hidden layer into the output layer of size $(1 \times V)$, since the predictions are supposed to be the words encoded with the one-hot technique. The skip-gram model learns with training to predict the context given a target word. Once the entire vocabulary is trained, a weight matrix $W_{V \times E}$ connecting the input to the hidden layer is to be produced to give the embeddings. This representation ideally encapsulates the semantics and similar words are close to each other in the vector space. Skip-gram performs better with a small amount of data and is found to represent rare words well.

The Continuous Bag-of-Words (CBoW) model presented in [14] is similar to the opposite process of the skip-gram model since it predicts the current word based on the surrounding words. The frame consists of a window of words around the current (middle) word, i.e. a few words before and after the centre word. The input layer of size (1×V) consists of the words of the one-hot coded frame and for each such word the weight matrix $WV \times E$ results in the hidden layer.Global Vectors (GloVe) proposed in [15] is an unsupervised learning algorithm for obtaining vector representations of words. Like most unsupervised algorithms, it is based on measures such as word frequency and the number of times it occurs with other words. GloVe models are trained using a word co-occurrence metric that indicates how many times each word pair is used within a given corpus and minimizes the least squares error. This creates a word vector space where distance between words is related to semantic similarity. In the model presented in [15], the output of the self-perceived layer is fed into a feedforward neural network, from which the new output is fed into a new encoder with the same layers as before. Otherwise, the decoder has the same layers, but with an attention layer in between. Ultimately, the decoder has a linear layer and a softmax layer, where the linear layer is a simple neural network that represents the vectors formed by the decoder as larger vectors called logit vectors. The output of the linear layer is fed to the softmax layer, which converts the output to probability values.

Transformers, [16] and [17], allow parallel computation using attention models. It contains two separate mechanisms: an encoder that reads text input and vectorizes it, and a decoder that decodes the vectors to output. The basis of the encoder/decoder architecture is an encoding component, which is a stack of encoders, and a decoding component with a corresponding number of decoders. The input to the encoder first passes through a layer called self-awareness. This helps the encoder consider other words in the input when encoding a particular word. BERT, developed by the Google AI team, has been trained on the English Wikipedia dataset containing almost 2.5 billion words and the Books Corpus dataset containing 800 million words. The bi-directionality of the model is manifested in the fact that it reads all inputs simultaneously. Because of this, the model can learn the context of a word with the help of all the words surrounding it [18].

Research reported in [19] has depicted that the post attempts to categorize text pulled from Twitter into hate speech or offensive language or neither. It is important to mention that for the pre-processing of the text, each comment (Tweet) was converted to lowercase and a technique known as stemming was used, specifically the Porter stemmer. Also, a sentiment dictionary designed to assign a sentiment score to each social media comment was used. The models used were: logistic regression, Naïve Bayes, decision trees, random forests, and linear SVM. The conclusion the researchers were led to was that verbal methods are effective in identifying offensive language but lag in accuracy in detecting hate speech. The most essential, however, is the report that due to the difficulty of detecting hate speech, the people who use it must be studied by analyzing both their personality and their motivations so that other parameters can be added to the context.

Work reported in [20] extracted sentences from some major “hate sites” in United States. They annotated each of the sentences into one of three classes: “strongly hateful (SH),” “weakly hateful (WH),” and “non-hateful (NH).” They used semantic features and grammatical patterns features, run the classification on a test set and obtained an F1-score equal to 65.12%. The research reported in Word2Vec, proposed in [21] was the first technique used for word embedding. Its working mechanism could encode the text through two main approaches: the skip-gram model or the Common Bag of Words (CBOW) model. While the latter predicts a word on the basis of the words within the surrounding context, the former predicts the words within the surrounding context starting from the current word. These mechanisms map words into vectors that are closer when words are similar and often close together.

The work [22] has presented HaSpeeDe task, which detects HS in Italian texts from Twitter and Facebook, is described in the study. In addition to confirming the challenge of cross-platform HS detection, the results showed

great promise in tasks utilizing the same social network's data for both training and testing. Research work in [23] deploy deep learning to detect hate speech in Tweets. They used three neural network techniques, where the word embeddings were initialized with either random embeddings or GloVe embeddings. The following methods were deployed: (1) Convolutional Neural Network (CNN), (2) Long short-term memory (LTSM) and (3) FastText. The experiments conducted showed that CNN performed better than LSTM which was better than FastText. Moreover, they concluded that embeddings learned from deep neural network models when combined with gradient boosted decision trees led to best accuracy values, which significantly outperforms the existing methods.

The research reported in [24] authors used nine publicly available datasets that focused on different abusive language types such as aggressive behavior, hate speech, and toxic comments. These datasets have different labels, however, the authors binarized the labels on all datasets into positive (abusive language) and negative (non-abusive language). To assess the model generalization, the authors trained a support vector machine (SVM) classifier using a training split of one dataset and then tested it on the testing splits of other datasets. Results showed, expectedly, that the performance on out-domain data declined by more than 50% of F1 score in most cases.

The research conducted by [25] uses a deep learning method which is an amalgamation of CNN and LSTM models. The built model is named ConvLstm. The ConvLstm architecture replaces the pooling layer contained in CNN by using the LSTM layer. The LSTM layer is worn to conserve information from relatively long input word sequences. The ConvLstm model was then tested on two different datasets, namely the IMDB movie review dataset and the SSTB dataset. This study also uses Word2Vec to form word vectors from the dataset used. The best accuracy results obtained using the ConvLstm model are 88.3%. In the research reported in [26] to get the optimal model for text classification, an experiment is carried out trial and error using LSTM with word embedding feature Word2Vec 300 dimensions. By tuning hyperparameters and making a comparison of eight proposed LSTM models with large-scale datasets, and to demonstrate that LSTM with Word2Vec feature can achieve good performance in text classification. The results showed that the text classification using LSTM with the Word2Vec feature obtains the highest accuracy on the fifth model with 95.38%, while the average value of precision, recall, and F1-score is 95%. In another multi-class classification approach, authors [27] in trained support vector machine employing character and word *n*-gram-based features and classified the hate, offensive, and neutral text. The authors concluded that the character 4-gram-based model performs best

Investigation work in [28] collected 450,000 tweets for their research. N-grams (1-5) word features were extracted from tweets and the supervised model was used to classify them. Three classifiers (i) Bayesian logistic regression, (ii) Support Vector Machine (SVM) and (iii) voted ensemble classifier were tested. The best results were obtained using the Voted Ensemble Classifier where the accuracy, recall and F1-scores were 0.89, 0.69, 0.77, respectively. [29] worked on English-Hindi mixed tweets for HS detection. They developed their embeddings with the large corpus of code-mixed data to build the model. The experimental results confirmed that the developed code-mixed embedding provided better performance compared to the pre-trained word embedding. Several classifier models such as SVM, Random Forest, CNN-1D, LSTM, and Bi-LSTM models were used for the experiment. The best performance was obtained with CNN-1D model, where the precision, recall, and F1-score value were 83.34, 78.51, 80.85 respectively.

III. HATE SPEECH DETECTION USING DEEP LEARNING

The formulation of the problem that has been discussed in this study is how to classify multi-class and multi-label hate expressions on Twitter by using Natural Language Processing and Deep Learning (CNN and BERT) mechanisms into five classes namely, toxic, hate, insult, threat and obscene; and examine the performance of the same in classifying multi-class and multi-label natural language tokens. The work reported herein was aimed at (i) proposing a CNN and BERT based approach to classify multi-class and multi-label textual hate content on Twitter, and (ii) training (and testing) the proposed classifier and choose the best feature set to optimize its classification accuracy.

In recent years, artificial neural networks based on structure and functions are increasingly attracting the interest of researchers with their usefulness. Areas of network design include the computational speed of nodes, the ability to handle complex nonlinear operations, and the ability to identify relationships between quantities that are difficult to model. The use of artificial neural networks in their various applications gives ease of implementation, quite high reliability in their operation, and also immediate response if they have been implemented in hardware. Several of these applications already exist on the market and are particularly widespread in their use. Some of the capabilities

of neural networks include pattern recognition, function calculation, analysis, prediction, optimization, and automatic control. The diagram shown in +Fig. 1 depicts the mechanism of hate speech diagnosis using CNN and BERT.

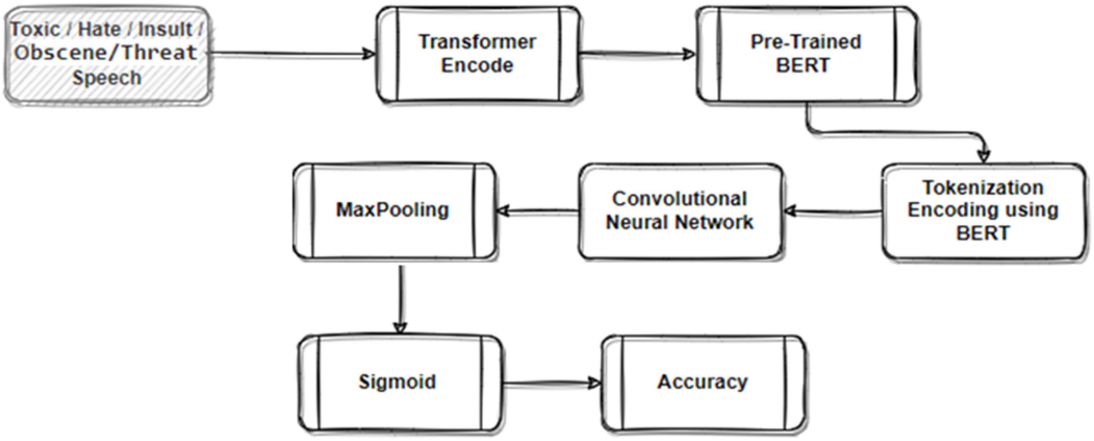

Figure 1. The architecture of the proposed hate-speech detection model

3.1 Dataset

The first goal was to gather insights from user tweets. More specifically, it was important to build a neural network that performs hate speech analysis of tweets. This involved producing six bits for each tweet – each bit corresponding to one of the five classes the tweet may belong to. The six possible classes are toxic, hate, insult, threat, and obscene. The quintet of bits <00001> for a tweet means that the tweet belongs to <obscene> class .i.e the fifth class in order (where the quintet bit is a '1'. After a thorough search the following dataset was identified for the present research effort:

https://www.kaggle.com/code/dhirendra73/toxic-comment-glove-and-gru-with-97-accuracy/data.

This dataset contains comments from microblog messages using Twitter having 159751 rows. The generated dataset is in .csv (comma-separated values) format. The first attempt was to collect tweets for toxic and hate speech over a while, but we encountered various problems due to the Twitter API policy which did not allow us to download a large number of Tweets in a short time since it has set an upper limit. For this reason, it was deemed necessary to search the internet to find a dataset containing comments-tweets from users for various topics over hate speech.

The data considered was in the form of tweets comprising 159751 rows in *csv* (comma-separated values) format and containing the keywords that are often used by netizens to spread toxic, severe toxic, obscene, threat, insult, and hate speech. The screenshot of the code below (Fig. 2) shows the process of mounting Tweets' dataset using the Pandas library.

```
1   #Loading the dataset and show a basic description
2   df = pd.read_csv("/content/drive/MyDrive/HateSpeech/train.csv")
3   print("Toxic Hate Speech Dataset Loaded!")
```

Figure 2. Mounting Tweets using Pandas Library.

3.2 Data Labeling

data items were labelled to divide the comments into six classes, namely, toxic, severe toxic, obscene, threat, insult, and hate. The presence and absence of a particular type of hate speech were marked by a 1 and 0 respectively. The dataset has been tokenized and annotated by five individual annotators. The results of the labelling carried out by each annotator were compared with each other and tested to ensure the appropriateness of the category allotted to each tweet. A screenshot of the tokenization process and the labelled dataset is shown in fig. 3 and fig. 4 below.

```
#List of the labels in the dataset
target_columns = ["toxic", "severe_toxic", "obscene", "threat", "insult", "identity_hate"]
#Establish batch size and sequence length for tokenization
batch_size = 25
valid_len = 256
```

Figure 3. Tokenization code

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	b5f57f317fc270d0	More like Madcocksuckingboy er faggot. 68.171...	1	0	1	0	1	0
1	766997bfc92e84cc	I hate you? I hate you!	1	0	0	0	0	0
10	d72bb7f6b9efd310	"Connor is a canadian citizen who loves life a...	1	0	1	0	1	0
29	29ebcbb5f1f24fcd	Go use your drugs, insignificant nobody. Erase...	1	0	0	0	1	0
45	19cb1a382c6e6dc0	"\n\n Stop lying about my edits \n\nMy good fa...	1	0	0	0	0	0

Figure 4. Dataset Labeling

3.3 Word Cloud of Hate Speech

One of the easiest ways to identify or perform an analysis of feelings is to use a word cloud because within this graphic one can identify which are the most used words within a text or concrete corpus. This can allow us to have a fairly clear idea about its polarity. Within Python, the construction of a word cloud is done with the help of the WordCloud library which has several tools that allow us to visualize the feelings (Hate, Toxic, Abusive, etc.) within our text. The construction of the definitive word cloud was not achieved at first because several words do not add value to our analysis.

The illustration (fig. 5) shows the first result of the word cloud corresponding to toxic and hate tweets in the present dataset. This illustration or analysis, allows us to have a first approximation of what our word cloud could be.

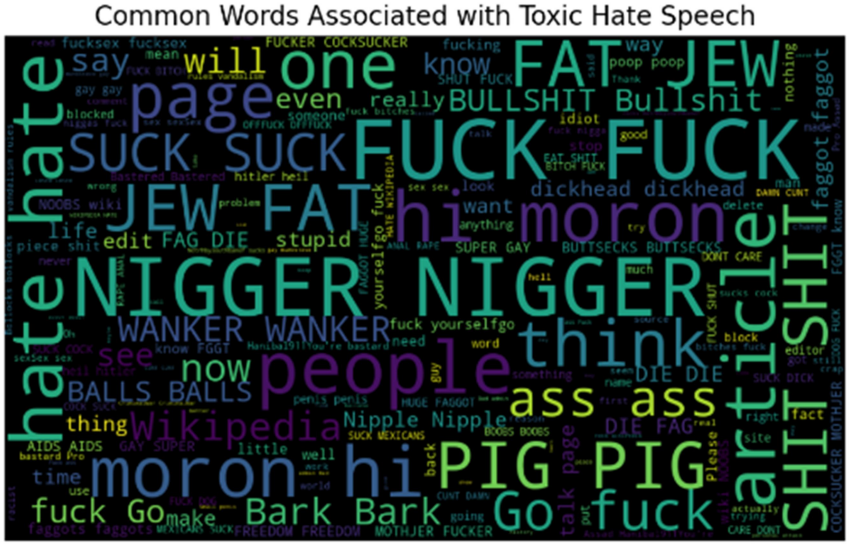

Figure 5. Word Cloud of Hate and Toxic Tweets

3.4 Data Visualization

The purpose of data visualization is to recognize and classify samples according to the class they belong to. A well-known example here is which tweet lies in which class or category. In practice, these cases i.e. the number of classes is known in advance and can be from 2 to N numbers for further classification below is the bar graph (fig. 6) that depicts tweets which lie under the category of toxic, severe toxic, obscene, threat, insult and hate.

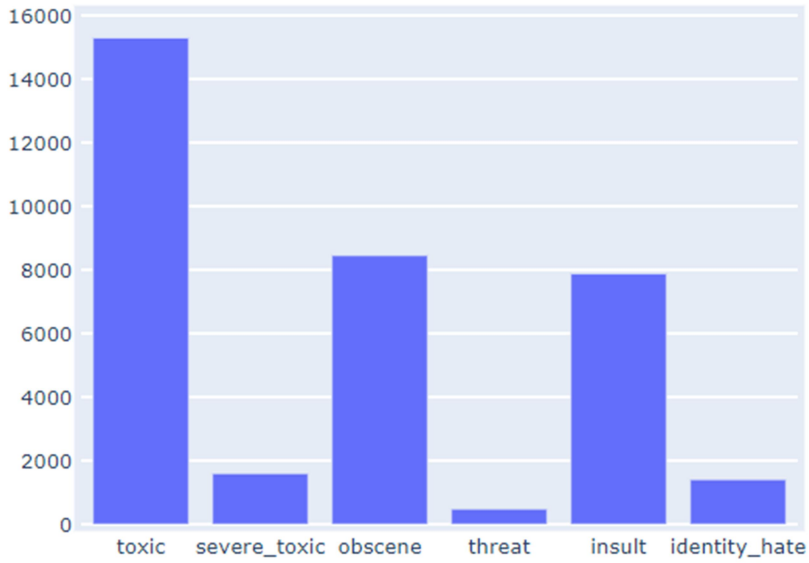

Figure 6. Bar Graph of Tweets based on Categories.

3.5 BERT Implementation

For modelling a hate speech prediction engine, the dataset was divided into two parts. The first data was used for model formation and training. Thereafter, the second data was used for testing the predictive model. The data resulting from the above normalization is then divided into test data and training data. The distribution of the data used is 80:20. Where 80% for data for training data and 20% is for test data. The distribution of training data and test data above aims so that the learning algorithm can learn from the patterns that have been obtained from the results of the training process which was implemented in the testing data. The process of training and testing using the BERT method continued until the optimal model was obtained.

Google Inc. proposes a series of methods whose purpose is to study the use of attention in networks of Transformers such as BERT. In each head of BERT, generate attention coefficients among all token representations, and based on these attentions generate a new representation for each token. With this in mind, the scheme has based on the premise that attention coefficients indicate how important is each token for the formation of the resulting representation. With this, the scheme developed a syntactic analysis of attention and general analysis. During the syntactic study, each head of the model is analyzed separately, in a quest to find the linguistic knowledge that the scheme developed. Since the analysis is on the syntax of the words in the input, it is necessary to transform the coefficients of attention from one at the level of tokens to one at the level of words.

```
X_Train (Token) Tensor Example: tensor([[-0.2521,  0.4818,  0.2623,  ..., -0.5040,  0.2138, -1.0409],
        [ 0.1345,  0.0670,  0.2133,  ...,  0.4435,  0.7186,  0.0524],
        [-0.2951,  0.5972,  1.2183,  ..., -0.2713,  0.3426, -0.0500],
        ...,
        [ 0.5530,  0.1956,  0.8139,  ..., -0.9231, -0.0652, -1.1645],
        [ 0.4888,  0.1615,  0.8827,  ..., -0.8471, -0.1268, -1.2246],
        [ 0.5984, -0.0179,  0.9008,  ..., -0.9343, -0.0702, -1.5759]],
       device='cuda:0')
```

Figure 7. Process Tokenization Encoding using BERT

At this stage, the data has gone through the pre-processing and feature extraction phase. Next through the process of modelling, BERT pre-trained which aims to enter the dataset into the model to be tested and trained. Figure 7 is a result of the BERT pre-trained model setup.

Therefore, to do this, when in the case of attention coming from a word with multiple tokens, it was considered as the attention of the word the average of the attention of all its tokens. On the other hand, when a multi-token word is serviced, the sum of the service is considered received in their tokens as the attention on the word. These decisions manage to have the property that the attention coming from each word adds up to 1, independent of how many tokens were original. This process is described more formally in equations 5 and 6, below.

$$\alpha_i, word_j = \sum_{t \in word_j} \alpha_{i,t} \ \ldots \ (5)$$

$$\alpha_{word_i}, word_j = \sum_{t \in word_i} \frac{\alpha_{t,word_j}}{|word_i|} \ \ldots (6)$$

In the context of word-level attention coefficients, it is said that given a head h and a word p_1, the word p_2 is predicted if p_1 mostly attends p_2 with the attention coefficients of h. Under this definition, it is searched if any 'un'head predicts any syntactic relationship. To do this, we use a dataset of English sentences that contains their labelled syntactic relationships. Later, for each head, it was verified if the predictions of the words correspond to their heads' syntactic for sentiment analysis, obtaining a recognition percentage for each relationship. To support the results found for toxic and hate speech detection, the predictive capabilities of the BERT heads were compared with the offsets more common to each relationship.

IV. RESULTS

At the modelling stage, the author used a batch size of 25 referring to the suggestion rendered in BERT literature. For Epochs, 30 epochs were determined due to testing and ensuring that there was no underfitting. There are several parameters to determine whether the model is underfitting or overfitting. The determination of all the parameters above is based on "hit-and-trial error" to get the expected performance. Among all parameters, Epoch plays a significant role related to functional performance using a batch count of 300. The model was trained with a learning rate of 9.625e-06. Passing 25 Epochs reduces the learning rate by a factor of 10. Therefore, these data and values allow the final optimal training of the model including validation using a batch size value of 25.

Beginning training: lr = 9.625e-06, Batch Count = 300, Batch Size = 25, Epoch Count = 30

```
Epoch: 27, Training loss: 1.623893, Validation loss: 1.902306
Epoch: 28, Training loss: 1.552954, Validation loss: 1.880662
Epoch: 29, Training loss: 1.534881, Validation loss: 1.956535
Epoch: 30, Training loss: 1.555934, Validation loss: 1.873683
```

Figure 8. Model Optimization and Epoch Evaluation

Based on the proposed model of deep learning using BERT and CNN above the scheme concludes that the 30th epoch can achieve a validation loss: of 2.36% for toxic and hate tweets and a training loss: of 1.87% for the same. The plot of fig. 9 elaborates the scenario for ready reference.

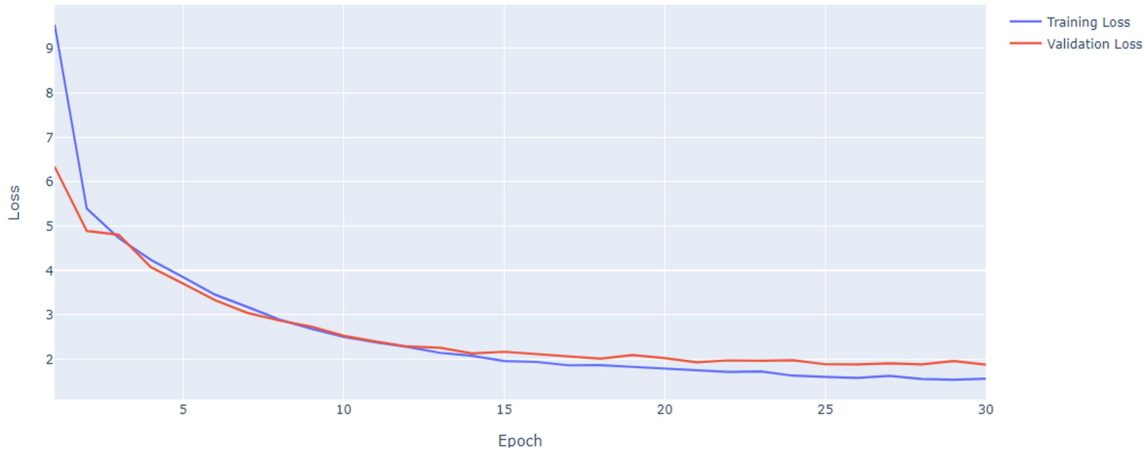

Figure 9. Epoch Evaluation

4.1 Accuracy

Finally, in the case of BERT with CNN integration, it was found that for text classification tasks it is possible to use the vector representation of the tokens obtained from the output of the last layer of the transformers using 1D CNN together with a classifier of max, where {p1, . . . , p_{he}} are the predictive probabilities. For the test text, the model accuracy was found to be 96.43%.

Based on the dataset used, which is in the form of a *csv* file, the pre-trained model from the BERT-base was used with 1D CNN. BERT-base is proven to be useful not only for short sentences to be classified, but can also conduct training so well that the results obtained were as accurate as 96.43%. This accuracy has been proven to be quite good, effective and reliable for the classification of social media (Tweets) postings as hate speech or otherwise as also for assigning a class signifying the degree of hatred of speech.

V. CONCLUSION

In an already polarized world, social networks prove to be a double-edged sword with the appearance of phenomena such as hate speech. In the present work, the presence of hate speech tokens in the Twitter database was detected and analyzed. For this, a base algorithm, BERT, has been carried out with the inculcation of CNN which fulfils the initial objective of the research. In addition, the presence of hate speech on the social network Twitter has been analyzed through an extensive study that has served to extrapolate its essential characteristics of it. For this, it has elaborated a procedure for the extraction and manipulation of said characteristics of CNN which has been shown with a BERT of 96.43% of accuracy that provides valuable data for the classification of toxic, severe toxic, obscene, threat, insult, and hate-speech categories.

These conclusions have served to include more information to the textual classification through BERT, a final multimodal model that combines categorical and numerical variables of the social network with the text entry of tweets, offering not only a new way of understanding the hate speech in social networks in general but a contribution of value that shows that the context of the social network improves the problem of textual classification using Deep Learning and Natural Language Processing. Finally, since it has been encouraged to find relationships with the spread and vitality of hate in the network, it would be relevant to study aspects such as reviewing a history of hate, both trends and public figures and anonymous users affected by hate, or the aggressors themselves. After for this, one could study how they behave with each other with an extension of attributes in CNN layers.

REFERENCES

[1] C. Baccarella, T. Wagner, J. Kietzmann, and I. McCarthy, "Social Media? It's Serious! Understanding the Dark Side of social media," European Management Journal, vol. 36, pp. 431-438, 2018. DOI: 10.1016/j.emj.2018.07.002.

[2] T. Chakrabarty, K. Gupta, and S. Muresan, "Pay 'Attention' to Your Context When Classifying Abusive Language," pp. 70-79, 2019. DOI: 10.18653/v1/W19-3508.

[3] Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on twitter using a convolution-gru based deep neural network," in European semantic web conference, 2018: Springer, pp. 745-760.

[4] K. Hairsine, "Reporting Hate Speech -- Practical Tips for Journalists," 2016. [Online]. Available: https://akademie.dw.com/en/reporting-hate-speech-practical-tips-for-journalists/a-19152896.

[5] E. Sazany and I. Budi, "Deep Learning-Based Implementation of Hate Speech Identification on Texts in Indonesian: Preliminary Study," 2018 International Conference on Applied Information Technology and Innovation (ICAITI), Padang, Indonesia, 2018, pp. 114-117, doi: 10.1109/ICAITI.2018.8686725.

[6] M. Kostic and V. Vilic, "Hate Speech on The Internet," Law and Politics, 14(1), 31-40, 2016. https://www.researchgate.net/publication/324721041_HATE_SPEECH_ON_THE_INTERNET.

[7] Ziqi Zhang, David Robinson, and Jonathan Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. 2017.

[8] J. Pereira-Kohatsu, L. Quijano-Sanchez, F. Liberatore, and M. Camacho-Collados, "Detecting and Monitoring Hate Speech in Twitter," Sensors, vol. 19, 2019.

[9] L. Voita, "Language Modeling," 2019. [Online]. Available: https://lena-voita.github.io/nlp_course/language_modeling.html.

[10] G. A. Fink, "n-Gram Models," in Markov Models for Pattern Recognition, Advances in Computer Vision and Pattern Recognition, Springer, London, 2014, https://doi.org/10.1007/978-1-4471-6308-4_6.

[11] X. Wang, W. Jiang, and Z. Luo, "Combination of convolutional and recurrent neural network for sentiment analysis of short texts," in Proceedings of COLING 2016, the 26th international conference on computational linguistics: Technical papers, 2016, pp. 2428-2437.

[12] M. S. Islam, S. S. S. Mousumi, S. Abujar, and S. A. Hossain, "Sequence-to-sequence Bangla sentence generation with LSTM recurrent neural networks," Procedia Computer Science, vol. 152, pp. 51-58, 2019

[13] C. Zhang, X. Liu, and D. Bis, "An Analysis on the Learning Rules of the Skip-Gram Model," in The 2019 International Joint Conference on Neural Networks.

[14] Q. Wang, J. Xu, H. Chen, and B. He, "Two Improved Continuous Bag-of-Word Models," pp. 2851-2856, 2017. DOI: 10.1109/IJCNN.2017.7966208.

[15] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," in EMNLP, vol. 14, pp. 1532-1543, 2014. DOI: 10.3115/v1/D14-1162.

[16] N. Kalchbrenner, L. Espeholt, K. Simonyan, A. van den Oord, A. Graves, and K. Kavukcuoglu, "Neural Machine Translation in Linear Time." 2016. [Online]. Available: https://arxiv.org/abs/1610.10099

[17] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need." 2017. [Online]. Available: https://arxiv.org/abs/1706.03762

[18] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[19] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated Hate Speech Detection and the Problem of Offensive Language." Proceedings of ICWSM 2017. [Online]. Available: https://arxiv.org/abs/1703.04009

[20] N. D. Gitari, Z. Zuping, H. Damien and J. Long, "A lexicon-based approach for hate speech detection", *Int. J. Multimedia Ubiquitous Eng.*, vol. 10, no. 4, pp. 215-230, Apr. 2015.

[21] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G.S.; Dean, J. Distributed Representations of Words and Phrases and their Compositionality. In Proceedings of the Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013, Lake Tahoe, NV, USA, 5–8 December 2013; pp. 3111–3119.

[22] Bosco, Cristina, et al. "Overview of the evalita 2018 hate speech detection task." Ceur workshop proceedings. Vol. 2263. CEUR, 2018.

[23] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. 2017.

[24] Karan, M.; Šnajder, J. Cross-Domain Detection of Abusive Language Online. In Proceedings of the 2nd Workshop on Abusive Language Online (ALW2), Brussels, Belgium, 31 October–1 November 2018; Association for Computational Linguistics: Brussels, Belgium, 2018; pp. 132–137.

[25] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, 2014. DOI: 10.3115/v1/D14-1181.

[26] R. Ganda and A. Mahmood, "Deep Learning Approach for Sentiment Analysis of Short Texts," pp. 705-710, 2017. DOI: 10.1109/ICCAR.2017.7942788.

[27] Malmasi, S., Zampieri, M., 2017. Detecting hate speech in social media, in: Proceedings of the Recent Advances in Natural Language Processing, ACL, Varna, Bulgaria. pp. 467–472.

[28] P. Burnap and M. L. Williams, ‘‘Cyber hate speech on Twitter: An application of machine classification and statistical modeling for policy and decision making,’’ Policy Internet, vol. 7, no. 2, pp. 223–242, Jun. 2015.

[29] S. Kamble and A. Joshi, ‘‘Hate speech detection from code-mixed Hindi English tweets using deep learning models,’’ 2018, arXiv:1811.05145. [Online]. Available: http://arxiv.org/abs/1811.05145