Hate speech detection using deep learning based artificial neural networks

Jaykumar Naik[1] and **Nikki Gaikwad**[2]
[1]jkn5454@mavs.uta.edu
[2]nbg6335@mavs.uta.edu

Abstract

In a modern world where online presence is a necessary more and more people have been interacting with each other now than before. This means on many of the online platforms you are open to interact with a person you don't know or have never met before. This creates a tendency to sometimes receive hate which counts as cyber bully and is not something anyone should go through. Our project aims to create a deep learning model to learn to detect such hate in text form by creating two models and comparing their accuracy and speed. So, in future we can stop such comments before the receiving user even sees it.

1 Introduction

In this project we try and test different models to detect text-based hate speech. The objective is to identify negative sentiment and hate directed towards someone. We also aim to make this detecting as fast as possible to be able to handle high influx of real time data in the online world where platforms like Twitter, Facebook, Instagram are receiving millions of messages and comments every minute.

We achieve this using natural language processing techniques to perform this mere second to make if a fast process while decreasing the probability of false positives and false negatives. In a perpetual online world where people get notifications of new interactions right aways in real-time it is imperative that our model to detect hate speech is also as fast as the real-time notification.

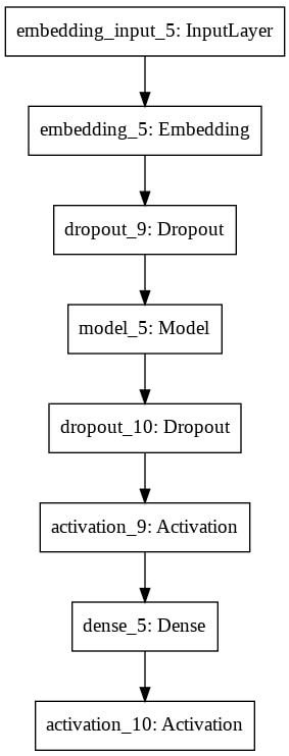
Natural language processing play an important role on online platforms these days to analyze the ingest data at scale. This report provides a comprehensive overview of deep learning approach in natural language processing. In addition, we will do accuracy comparison and performance analysis of two deep learning models on same data sets.

2 Project Approaches

2.1 First approach

As detailed below for the first methodology we create and experiment with a Convolutional Neural Network (CNN). CNN is a popular and common in many deep learning approached for natural language processing as well as image/video-based data in machine learning.

In this approach we started with an embedding layer. Embedding layer convers each word into a fixed length vector of numbers. The fixed length of word vectors helps us to represent words in a better way along with reduced dimensions.



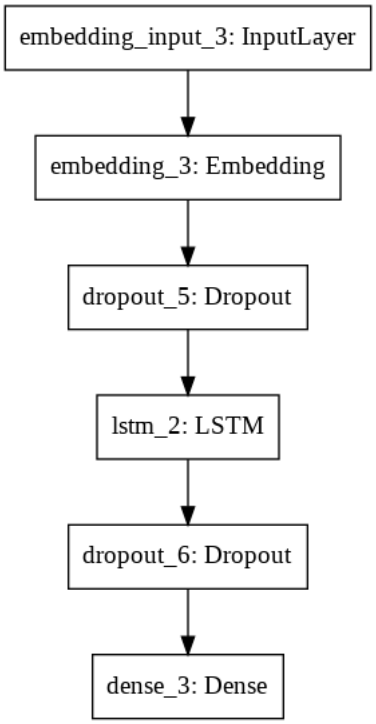
Next, we use a dropout layer at 25% drop probability which means this layer will turn off 25% of the units. This is a regularization method to reduce overfitting on a large dataset.

Next is where our main CNN layer comes named model. In our CNN layer we have three 1-dimentinal convolutional layers all of size 1 with relu activation and global max polling after each of them.

We add another dropout layer now with 50% drop probability before adding a dense layer with softmax activation which dense the network output in two classes and with the help of softmax activation function we can convert that into probabilities for output lables.

2.2 Second approach

LSTM is a special kind of RNN model which does not suffer from short term memory loss. Long short-term memory provides the advantage of considering the gradient during the backpropagation. In RNN, the gradient vanishes as it



back propagates through time which in consequence does not contribute too much learning.
The first layer is the embedding layer which has the same inputs as the CNN model's embedding layer has for comparison purposes. Next, dropout layer also has the dropout probability of 25%. Then we have our main LSTM layer, which has the basic RNN activation filter ReLu along with the addition of Input Gate, Output Gate, Cell State and the Forgot Gate.
The first gate is the Forgot Gate which is responsible for deciding which information is important and the rest is thrown away. Input to the Forgot Gate is the current input and the input from previous hidden state.
The Input Gate is a multiplicative gate which protects the memory content saved in cell t-1 states. Similarly, the Output Gate is also a multiplicative gate.
Next, we have the dropout layer which again has the same dropping probability as the 2nd dropout layer of CNN model.
The last layer is the Dense layer, which is a fully connected one, in which each neuron receives input from all neurons of the previous layer, which gives output of 50 dimensionality space.

3 Dataset description

For preparing the data we used two datasets from Kaggle since the first[1] dataset had a bad ratio of positive labels at only 2,242 positives labels in 24,783 tweets. Introducing the second[2] dataset at 31,962 values with little processing to merge both dataset we prepare our final dataset at 56,745 tweets with 22,862 positive labels and 33,883 negative labels. We used python to merge these two datasets.

In our dataset we have a tweet column which represents the actual text of the tweet as it is taken from the platform and a label column with values either 0 or 1 used to represent whether the given tweet is a hate speech or not.

To get most out of our data we need the tweet data to be in its original form with images, links, hashtags, retweet quote mention, all of these will help us better understand the data and make better predictions. Twitter data is also a good candidate for our project compared to other platforms since Twitter ensures the tweet us only up to 240 characters long so we get the data in fixed size range.

4 Project description

4.1 Description

The machine learning algorithm works with numbers and our imported dataset is in textual format which even before we vectorize we had to clean up the dataset. We converted the hashtags, image and external URLs, emoticons into common tags recognizing them for what they are instead of leaving them in their raw format which would've contributed less to the data.

Both the models were fed the same data and were trained using same training parameters to get accurate comparison of the models.

4.2 Main references

In our main reference [1] published in world wide web companion 2017 experiments with multiple deep learning architectures to learn semantic word embeddings and aims to classify the tweets in to racist, sexist or neither.

Experimenting with various deep learning models and embeddings average accuracy achieved in deep learning approaches without any additional classifiers is 81.3%.

[1] https://www.kaggle.com/datasets/mayurdalvi/twitter-sentiments-analysis-nlp

[2] https://www.kaggle.com/datasets/mrmorj/hate-speech-and-offensive-language-dataset

Second reference [2] published in ArXiv: Computation and Language and accepted in ICWSM-2018 we used focuses on classifying hate speech on whether it's directed or generalized in addition to detecting hate speech in the tweet.

The paper explores the hateful content and attempts to organize them into different types (race, religion, threat, etc.)

Third reference [3] published in ArXiv: Computation and Language and accepted in the Semantic Web Journal used Word2Vec embeddings and trains a CNN model on multiple publicly available twitter datasets and compares their performance.

Fourth reference [4] published in The Semantic Web. ESWC 2018, develops support vector machine and a combination of CNN + LSTM model and compares their result on multiple twitter databases. They used random weight initialization for the first embedding layer in the model.

4.3 Difference in approach

First difference
Rather than trying to classify the tweets into whether it's directed hate or generalized hate we focused first only on classifying the tweet into whether it's a hate speech or not.

Second difference
The reference paper used pre downloaded GloVe file for words embedding which is later used to initialize weights for the embedding layer of the deep learning model. We took the approach to initialize the weights of the embedding layer in randomize manner.

Third difference
One of the reference paper goes into developing a model by combining two deep neural network architectures and testing their performance on different publicly available Twitter datasets while we aim to compare of two different architectures on same dataset for fast iteration purpose and to gain better understanding of the individual architectures.

4.4 Difference in performance

Our CNN model with random weight initialization performed 8.1% better on F1 score at 88% accuracy, 9.47% better on precision at 98% and 7.8% better on recall at 88%.

Our LSTM models F1 score accuracy came out at 76% while the reference paper was able to get an accuracy of 80.4%

The long short term memory also takes longer to train compared to convolutional neural network because of the sequential computation for the hidden layers.

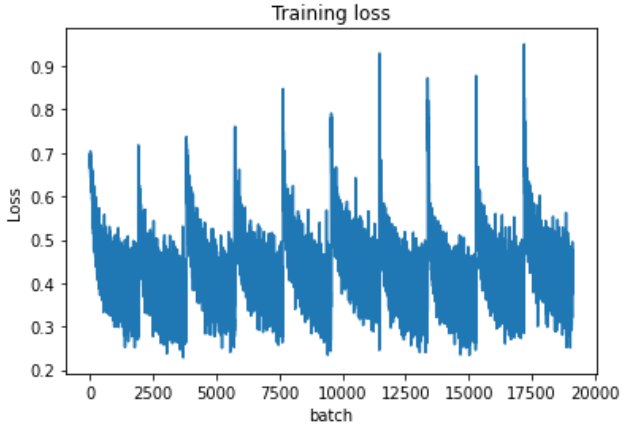


Figure 1 Loss of CNN model during training across 10 KFolds

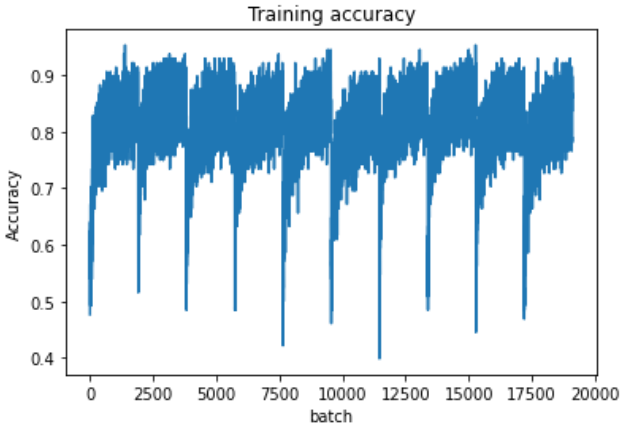


Figure 2 Accuracy of CNN model during training across 10 KFolds

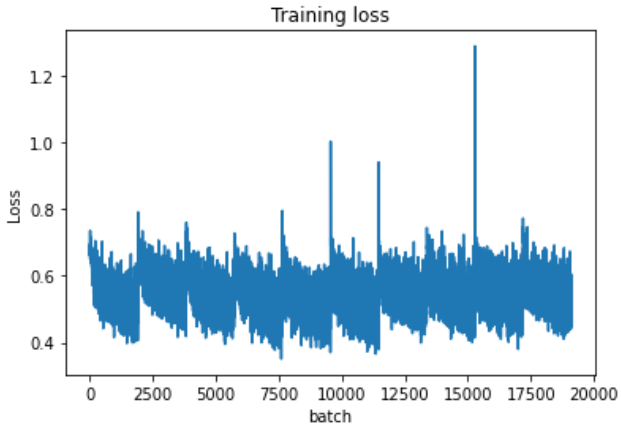


Figure 3 Loss of LSTM model during training across 10 KFolds

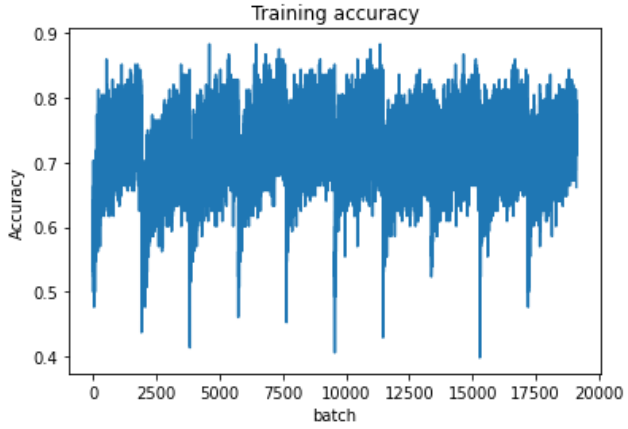


Figure 4 Accuracy of LSTM model during training across 10 KFolds

5 Analysis

The performance improvement in the CNN model mostly came from the enhancing the 1-dimentional convolutional filters of the layer. And experimenting with the hyperparameters tweaking and testing which brought the CNN model from initial 63% of accuracy to 88% this is also 8.1% more accurate than the main reference paper [1].

Working more on LSTM could've also improved its accuracy to at least 85% for what we are only getting 76% at current model architecture. We can also use GloVe embeddings in the future to feed the deep neural network better weights for particular weights.

An interesting architecture we can try in future is to combine the CNN and LSTM into one single model to take advantage of the state machine from LSTM into CNN and compare the performance with the previous models. We can enhance the project furthermore and train to model to classify hate speech further into generalized or directed hate speech where we can classify if the detected hate is generalized towards a community, organization, etc. or towards a singular person. If the hate is directed towards a generalized entity it could incite mob mentality and mobilize large number of people from the online platform into taking illegal violent actions.

Conclusion

In this project an artificial neural network architecture design is implemented using deep learning in order to classify Twitter tweets between hate speech or not. There is a large room for improvement of future purposes for this project and design of more deep neural networks architecture. Here is huge scope of improvement of this project for the future and we can also work on a web interface to flag tweets in the Twitter web interface and hide the tweets automatically which are flagged as hate speech.

The next step for the more effective accuracy for this project is to merge the CNN and LSTM models into one and use GloVe embedding to initialize the weights to achieve higher accuracy. With more computational power and time we can use larger data sets which will make the model produce better results for the real world deployment.

References

[1] Badjatiya, Pinkesh and Gupta, Shashank and Gupta, Manish and Varma, Vasudeva, 2017. "Deep Learning for Hate Speech Detection in Tweets". In Proceedings of the *26th International Conference on World Wide Web Companion (WWW '17 Companion)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 759–760. https://doi.org/10.1145/3041021.3054223

[2] Mai ElSherief, Vivek Kulkarni, Dana Nguyen, William Yang Wang, Elizabeth Belding, Hate Lingo: A Target-based Linguistic Analysis of Hate Speech in Social Media 2018, *arXiv:1804.04257*.

[3] Ziqi Zhang, Lei Luo, "Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter", 2018, *arXiv:1803.03662*.

[4] Zhang, Z., Robinson, D., Tepper, J. (2018). Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network. In: , et al. The Semantic Web. ESWC 2018. Lecture Notes in Computer Science(), vol 10843. Springer, Cham. https://doi.org/10.1007/978-3-319-93417-4_48