Investigating Hate Speech and Abusive Language on Twitter with Deep Learning

Emiliano Rodrigues
CS230 - Fall 2018 - Stanford University

Motivation

Using Natural Language Processing (NLP) to categorize text extracts into different groups is a very common practice in Deep Learning. Sentiment Analysis is a field that has evolved a lot and gained a lot of momentum over the past years with more processing power and newer techniques to analyze text and keep track of the contribution of meaning from each word to the sentence as a whole. In this project, I explore the use of these techniques and apply it to a field that has gained a lot of attention lately: hate speech on Twitter.

Problem Definition

The dataset I used consisted of 24,783 tweets. Each tweet was reviewed by three different people, each of whom classified them as containing hate speech, containing abusive language, or neither. The most voted category for each tweet was the classification given to that specific tweet. It is important to note that given the tweets were classified by different people, what these individuals consider as hate speech, abusive language and neither is subject to their own interpretations of these terms.

After preprocessing tweets, which entails cleaning them from unhelpful sentence markers, user handles and other Twitter-specific key words, like *RT* (Re-tweet), I move on to build models that are able to classify the tweets as containing either hate speech, abusive language or neither of these. Then, I explore what these Neural Networks are doing behind the scenes: Why does the model classify a specific sentence as hate speech? What key words from the extract contribute the most to the classification? Can we trick the Neural Network into classifying something as hate speech that is actually not?

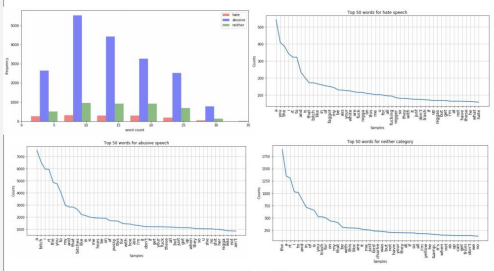
I extracted the relevant columns from the CSV file using Pandas, and proceeded to perform basic preprocessing that involved removing links and user handles (@person). Then, I used Keras's Tokenizer class in order to remove all punctuation, numbers and make all the text samples lowercase.

From that, I broke apart each sentence by the words they were composed of, encoding each word using Keras Tokenizer's $texts\_to\_sequences$. Each text extract was then padded at a maximum of 25 words, which is the average word count for a tweet containing 140 characters, which used to be the platform's word limit before being recently increased to 280 characters earlier this year. Since our dataset doesn't contain any tweet that recent, this was not a problem and didn't incur in lost data.

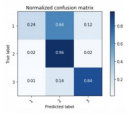
Approaches

In this project, I implemented three different Neural Network structures in order to see how they compare and which one would be most appropriate for this specific classification task. All models were trained with 300d GloVe embeddings. The first model was a Recurrent Neural Network, where I used a Bidirectional LSTM Layer. The second one consisted of a Convolutional Neural Network, where I used a Convolutional 1-dimensional layer, as well as a Max Pooling layer. Finally, I implemented a model mixing the Convolutional Layer with the Bidirectional LSTM Layer.

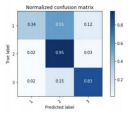
Data Analysis



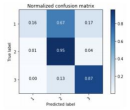
Modeling



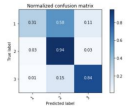
CNN with Conv1D Layer



RNN with Bi-LSTM Layer



RNN with Bi-LSTM Layer, Early Stopping activated



NN with Conv1D Layer + Bi-LSTM Layer

Discussion

Looking at the confusion matrix for the CNN, we see that the model is very successful at classifying tweets marked as abusive language, with a 96% accuracy overall for this category, while it does poorly for tweets classified as hate speech, classifying only 24% of these correctly, and mistakenly labeling them as abusive language instead. This is likely due to the similarities pointed out earlier between abusive language and hate speech, which seem to have very similar vocabulary frequencies. When it comes to tweets classified as neither hate speech nor abusive language, the model also performs well and is able to classify 84% of these correctly.

Looking at the confusion matrix for the RNN, this model performed worse than the CNN for both the abusive language and the neither classes, but was better by 10% compared to the CNN in classifying tweets that were labeled as hate speech.

The RNN with Early Stopping performed better than the previous two models when classifying tweets as neither, and performed just as well as the non-early stopping RNN for abusive tweets, but performed much worse than the two previous models for tweets classified as hate speech, only classifying 16% of these correctly.

Finally, the R + C NN performed better than the CNN, with a higher accuracy for tweets classified as hate speech (31% vs. 24%), but the same accuracy percentage in testing for the neither category (84%). Despite fairly good accuracy distributions for the three classes compared to the previous models, it didn't outperform the first RNN trained for 30 epochs, that had better accuracies for hate speech and abusive language labeled tweets.

It is interesting to look at the sentences that were miss-classified by the CNN model in the test set, so I went ahead and did that. For example, these are two samples that were classified as abusive language, but were labeled originally as hate speech: "*gucci mane in jail and dropping mixtapes every month and you hoes cant even text back*", "*im feeling pretty fuckin ghetto smh*". In these examples and personally, I believe these sentences would be better classified as abusive/offensive language as the model predicted, but that is not what they were labeled as.

Challenges & Future Work

- Size of Dataset → there were not enough hate speech samples, which ended up compromising how well the model could learn the patterns for these.
- Number of epochs trained for → I trained the models for 30 epochs but they could've been trained for much longer.
- Testing deeper networks → The networks I built were fairly straightforward in their composition, but I wonder if more complex structures would help at all.
- Binary vs. ternary classification → It would be interesting to see how well the model would learn the hate speech patterns if there were no abusive language category.
- Choice of Vector Embeddings → Could try fastText