Jigsaw Unintended Bias in Toxicity Classification

Author: Shashwata Sourav Roy Samya
Email: shashwata@student.sust.edu

ABSTRACT:

This was a competition on Kaggle which wanted the participants to deal with toxic comments containing various racial, religious, homophobic slurs, etc. We were given a large dataset of about 1.8 million comments and with various parameters about gender, religion, race, disabilities and so on. The task in hand was to find a solution to classify if a comment is toxic or not. The challenge was to deal with words that can have varied meanings (e.g. 'gay') and reduce the biases towards those communities or identities while classifying a toxic comment. We tried to implement a Bidirectional LSTM approach for which we achieved a private Kaggle score of 0.91383.

METHODOLOGY:

We were provided with train and test datasets where train datasets have 18,04,874 rows with 45 columns and test contained 97,320 rows with 2 columns. The train dataset had comments under the 'comment_text' column and a toxicity level ranging from 0 to 1 under the 'target' column. The dataset also contained other addition and sub type toxicity attributes.

Since the comments came in with different repetitive and special characters, our first task was to clean them up. We converted them into lower case at first, then converted each and every contraction to their original word. Next, we went on removing the punctuations from our comments and also repetitive and common words using stopwords.

This whole process took a considerable amount of time, thus after applying this on our train dataset comments, we saved in to a new dataset for easier calling. The same had been done for the test dataset.

We then proceeded to look into the different subtype of toxicity where we tried to find the number of null values in them. Due to an overwhelming percentage of the null values in them, we produced distribution plots of those sub toxic features in their own categories such as race, ethnicity, gender, religion, etc. without the null values to get a clearer idea of how they varied throughout our dataset. The null values would skew the distributions to the left.

Next, we created a word cloud with 'target' greater or equal to 0.5 as that is classified as toxic is this challenge. It yielded us with words that were prevalent in our comments.

To train our model we had to convert our comments into machine readable numbers. This was done using the Tokenizer from TensorFlow where we tokenized all the comments and padded all of them up.

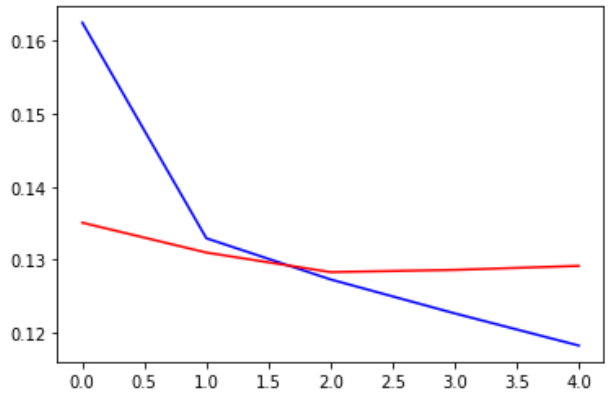
For much better representation of our comments, we also used word embedding where we generated a word vector matrix using the GloVe word embedding of 300 dimensions. Lastly our dataset was split into 70% training set and 30% as test set.

For our model, we had an Embedding Layer, Bidirectional LSTM Layer, an Average Pooling Layer, a Dropout followed by a Dense Layer. This was created by referencing different notebooks of other participants. The Adam optimizer was chosen with a learning rate of 0.001. The model was trained for 5 epochs on GPU for faster training.

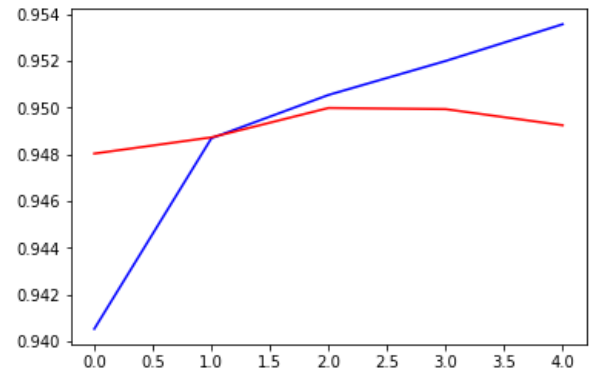
RESULT ANALYSIS

We achieved a private score of 0.91383 on Kaggle upon submission. Our accuracy on our training dataset was 0.9538 and upon validation we had 0.9492. We think that we may have overfitted our model as presented in the plots below between i) Training accuracy vs Validation accuracy and ii) Training loss vs Validation loss.

```
plt.plot(model.history.history['loss'], color='b', label="Training accuracy")
plt.plot(model.history.history['val_loss'], color='r', label="Validation  accuracy")
plt.show()
```




```
plt.plot(model.history.history['accuracy'], color='b', label="Training accuracy")
plt.plot(model.history.history['val_accuracy'], color='r', label="Validation  accuracy")
plt.show()
```



The plots show that our validation accuracy increased then at one point it started decreasing. The reverse can be said for our validation loss as well. We have looked into various options and tried adjusting some parameters but to no avail. We hope to investigate and study it in depth to get a better understanding of what may have been the cause for our model to overfit.

CONCLUSION:

Further higher accuracy may be achieved by using approaches such as Convolutional Layers along with Bidirectional GRU. Even our Bidirectional LSTM can be made more robust by using proper parameters and accurate layers. Our comments could also be further processed by using lemmatization which can produce a much better representation of our words to work with in our model. To implement these techniques in our model properly, we need to gain further knowledge on these approaches and their corresponding topics which we hope to do in the near future.