Toxic text classification using Bi-directional LSTM with Attention *

Rabindra Khadka *NLP,DSSC, University of Trieste*

Abstract

Online toxic comments have posed a great challenge for many websites and as a whole to the psychological well being of our societies.Humans along can not filter those comments manually and it is not a practical choice in dealing with the huge volume of comments.This project is based on developing the tool to automatically identify and classify toxic comments into `six` classes. Unidirectional LSTM architecture without attention mechanism was used as the baseline model which gave a mean ROC AUC of `0.968` on the test set. Then Bidirectional LSTM with attention mechanism was implemented and gave the improved accuracy result of `0.9735` on the test set.

1.Introduction

In today's internet world, comment sections should be carefully handled by companies. This section can be full of toxic and intimidating comments in the absence of moderator. Human moderator along is not a practical solution for gate keeping the toxic comments. This leads us to take help of NLP tools to detect, classify and stop the diffusion of those toxic comments in our societies.

In this project, attempt was made to improve the classification of toxic online comments in the dataset from **kaggle competition**. The dataset consists of **159,571** comments from wikipedia talk pages labeled manually by humans.The provided six classes of toxicity namely are: **toxic, severe toxic, obscene, threat, insult, and identity hate.**.Some of the comments can be seen in the **fig 1** below. At first bi-directional LSTM architecture was used to carry out the classification and then attention mechanism was added to see how the performance would improve.Finally, the results of evaluation of the model on the provided dataset were visualized.

id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0005300084f90edc	"\nFair use rationale for Image:Wonju.jpg\n\nT...	0	0	0	0	0	0
00054a5e18b50dd4	bbq \n\nbe a man and lets discuss it-maybe ove...	0	0	0	0	0	0
0005c987bdfc9d4b	Hey... what is it..\n@ | talk .\nWhat is it.....	1	0	0	0	0	0
0006f16e4e9f292e	Before you start throwing accusations and warn...	0	0	0	0	0	0
00070ef96486d6f9	Oh, and the girl above started her arguments w...	0	0	0	0	0	0

Fig1: A subset of the training set showing the comments and the labels.

2. Related previous work

Recent work based on this particular topic is done by the group who created the dataset used in the kaggle challenge [1]. They used binary identification of toxic comments without fine-grained classification where they applied simple n-gram NLP method and suggested future work on complex

*Dataset link: https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge.

methods like LSTM.Since then, there have been number of different attempts by incorporating CNN,RNN with word level embedding.With the release of BERT by google in 2018,this particular pre trained model has also been used to take on this problem set.

For this project, the approach taken was to observe how the performance could be improved by the application of attention mechanism in bi-directional LSTM.

3.Approach

3.1 Preprocessing of the data

Some of the preprocessing steps applied to the dataset included lower casing of all the words, removal of stop words, filtering out numbers and non alphanumeric characters.Then the comment sentences were tokenized using keras `tokenizer` function after which the token list from each comment was padded. The padding method involved filling in a sentence with empty indicators if the comment was too short and cutting off a sentence if it was too long.

For representing each token word in the input comments, word embedding was performed using the 50 dimensional pretrained global vectors for word representation(GloVe), which is trained on word-word co-occurence statistics on 6 billion tokens from Wikipedia and Giga word corpora with a vocabulary size of 400k [2].Then the input vectors of 'float32' were created by mapping word to id. Then these vectors were used to form the embedding matrix which then was used for the embedding layer in the model.

The dataset exhibited high imbalance in the number of examples for each label as shown in **fig 2**. The data set was divided randomly into training and validation set in the ratio of 70:30.

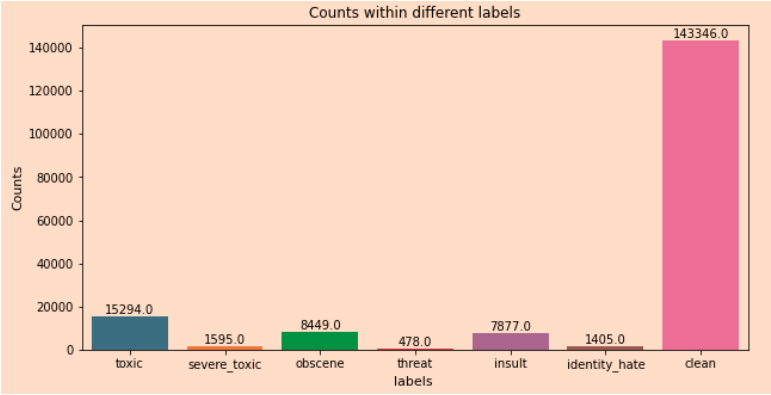

Fig 2: Counts of comments under each label. A single comment can have multiple labels.

3.2 Baseline Model & Architecture.

Uni-directional LSTM was chosen as the baseline model and later attention mechanism was added to the `Bi-directional LSTM` model to observe improve in performance. The inputs of the model are the average of 50d GLove embedding of all the tokens in each comment.Then the tensor from the embedding layer was fed into `uni-directional lstm` with the output dimension set at 50. The entire unrolled sequence of results was returned from the layer.The LSTM architecture allows

information to flow a longer path across the unrolled network i.e the previous state keeps flowing across the network.The recursive run for the LSTM model was set to 100. To prevent overfitting, dropout layer was added to the model with probability of dropout at 0.5. Finally, a fully connected layer with sigmoid activation was added to convert the output of the dropout layer to probabilities of each label.

From the baseline model, the performance measured based on ROC curves gave us the mean ROC AUC of 0.96 .The accuracy over the test set for the baseline model stood at 0.9706.

As our dataset consisted of high number of non-toxic comments(true negatives), this affected ROC curves plot as it is given by the true positive rate is `TPR=TP/TP+FN` and the false positive rate i.e `FP/FP+TN`; but this metric is used to compare the results in the kaggle competition. `Precison andRecall`' performance was also observed to see the performance of the models which indicates that labels with higher training examples exhibits a better PR performance curve.

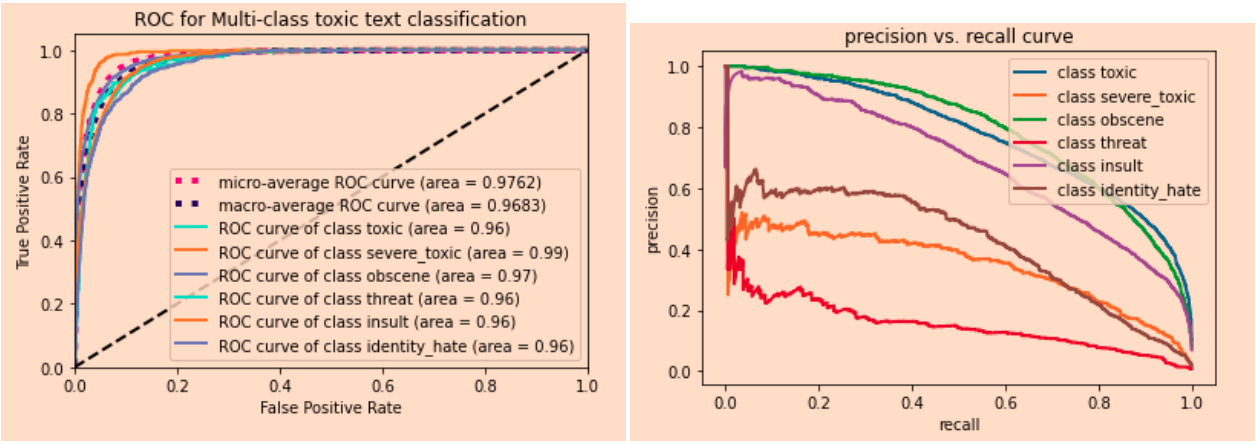

Fig 3: (L) ROC with AUC value for each label along with the average AUC curves.(R) Precision Recall curves for each labels from the baseline model performance.

3.3 Bidirectional Architecture

For a bidirectional LSTM architecture,everything remains same as unidirectional architecture as described above but there will be one set of parameters for a forward unrolled LSTM and another set for a backward unrolled LSTM that reads input token lists starting at the end rather than the beginning.So, for each token position, there will be two output states at each time step. Also here, to prevent overfitting dropout layer with $p_drop = 0.5$ was also added.

3.4 Attention Mechanism

Word attention mechanism will allow us to keep track of important informative words which might be helpful in solving classification problem correctly. As RNN tend not to keep information from previous steps of the sequence, this would lose some useful informative relevant words.So, to solve this problem attention mechanism can be implemented which gives all encoded states equal importance.Weighted sum of the encoded states can be used in the attention mechanism which will improve the classification.

The attention mechanism used in this project is simple which just computes attention weights by stacking fully connected layer on the top of each encoded state and followed by `tanh` layer.So, the hidden vectors of each word h_t from the recurrent time step t is fed into a single *tanh* non linear function which returns a new word representation. The attention score is computed by applying tanh non-linearity function to the sum of features vector from the dense layers .Then the attention weights is computed by normalizing the attention score of all the time steps using the softmax function.Then finally the attention context vector is given by the sum of the original hidden states h_t of each word weighted by the attention weights.

4 Experiments

The uni-directional LSTM model and bi-directional LSTM model with attention mechanism were ran using google colab machine. At first, the model was run for 10 epochs and it was observed that the model over-fitted just after 2 epochs as can be seen in fig 4 below.Therefore, the model was run for two epochs only. The dropout probability for the dropout function was also manipulated and when fixed at 0.5 gave a better performance result.

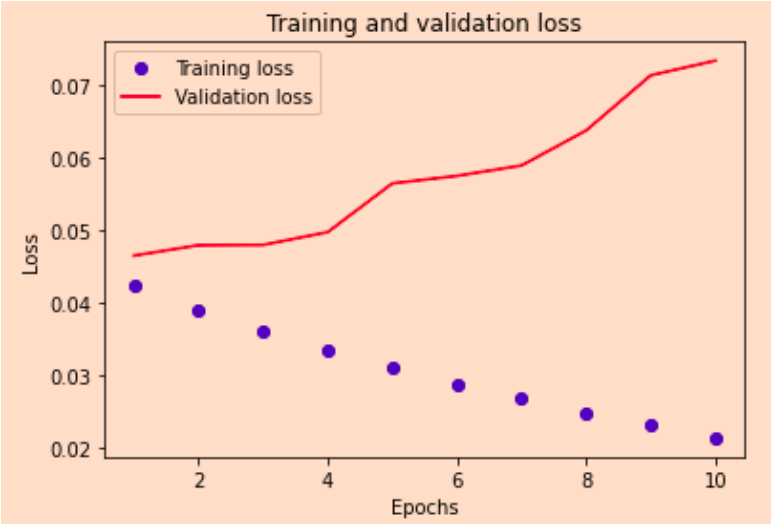

Fig 4: Loss function from the training session.It can be noticed that the validation loss after 2 epochs starts increasing which is a clear indication of overfitting problem.So, the training was carried out with just two epochs.

The experiment carried out on the model by changing the hidden state size, directionality of LSTM showed that the bi-directionality performs better than unidirectional LSTM and with the addition of attention mechanism to bidirectional architecture further improved the ROC AUC performance metric value. The performance results have been tabulated below.

	Directionality	#layers	Hidden Size	Test(ROC AUC)
0	Unidirectional	1	50	0.9683
1	Unidirectional	1	100	0.9691
2	Bidirectional	1	100	0.9695
3	Birdirectional+Attention	1	100	0.9734

Fig 5: Results from the search of directionality type,hidden state size of the RNN model with LSTM cell with and without attention.

---

4 Evaluation

Evaluation was done based on the ROC AUC curves which was the official metric of the kaggle challenge. Also, precision and recall curve was also observed along with the test accuracy. The macro average of ROC AUC suitable for this kind of imbalanced dataset was also taken into account while plotting the ROC curve. The summary of the result from different models has been tabulated below. We can see in the `fig 6` below , the ROC AUC curves are very close to the upper left corner indicating a near perfect classification. The PR curve points out that the classes with few training examples(`threat`,`severe_toxic`,`identity_hate`) have a weaker performance than those with higher number of training examples.

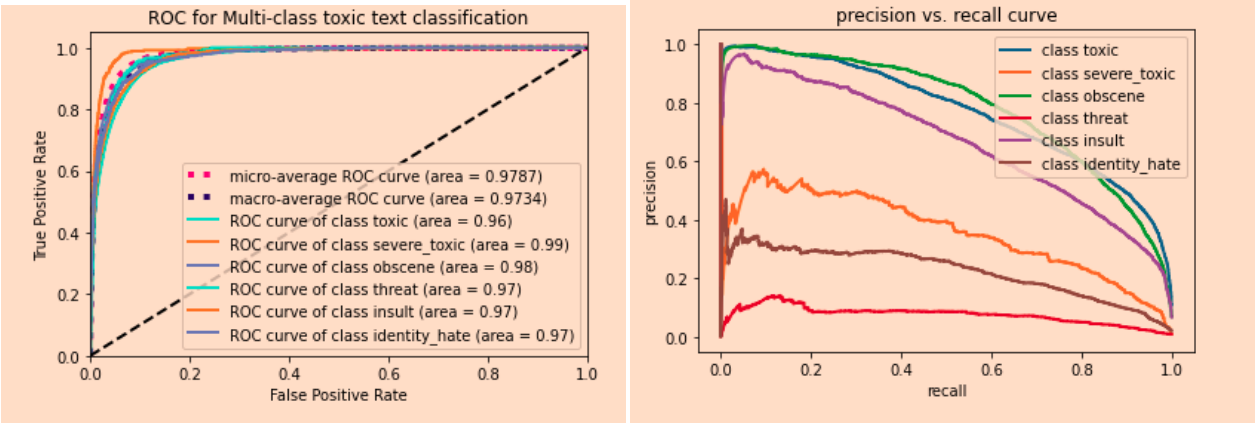

Fig 6: (L) ROC with AUC value for each label along with the macro and micro average AUC curves.(R) Precision Recall curves for each labels from the baseline model performance.

	Model	Test(ROC AUC)
0	Baseline(Unidirectional LSTM)	0.9683
1	Bidirectional LSTM	0.9695
2	Birdirectional+Attention	0.9734

Fig 7: Tabulating the average AUC ROC from different models on the test set.

The results show a good improvement of performance with Bi-directional LSTM with attention mechanism .

5 Conclusion

Implementing the tensorflow framework for Bidirectional LSTM with attention yielded a remarkable result with accuracy around .9976 and mean ROC AUC of 0.9734 which placed this result within top 1% of the results in kaggle entry.

The performance of the model is mainly limited by the embeddings of those rarely used vulgar words. In the future work, this particular issue can be addressed with the help of CNN or other methods which can help us to identify toxic character pattern. In addition data augmentation, feature engineering and fine tuning of pretrained model such as "BERT" can also be tried to boost the performance of the model.

References

[1]. Wulczyn, E., Thain, N., & Dixon, L. (2017) Ex Machina: Personal Attacks Seen at Scale.
[2]. Pennington, J., Socher, R., & Manning, C. (2014) GloVe: Global Vectors for Word Representation. Pro-ceedings of the 2014 Conference on Empirical Methods in Natural Language Processing.
[3]. Tensorflow implementation of attention mechanism.https://github.com/uzaymacar/attention-mechanisms
[4].https://www.tensorflow.org/tutorials/text/nmt_with_attention