CS583A: Course Project

Yujie Zhou

May 19, 2019

1 Summary

I participated in an active Kaggle competition, Jigsaw Unintended Bias in Toxicity Classification. The goal of the competition is to detect the toxicity across a diverse range of conversation. The model I tried which produced the best result is a LSTM model. I implement the LSTM model using Keras and run the code on Kaggle kernel using its GPU. For the evaluation, the competition uses a newly developed metric and the details could be found on competition page. In the public leaderboard, the score of model is 0.93296. I rank 1033 among the 2010 teams upon submission. The result on the private leaderboard is not available until June 19, 2019.

2 Problem Description

Problem. The competition asks to help with detecting toxic comments and minimize unintended model bias. The problem found by the conversation AI team was that the models incorrectly learned to associate the names of frequently attacked identities such as gay with toxicity. Though sometimes the comments containing these identities are not toxic. The competition is at `https://www.kaggle.com/c/jigsaw-unintended-bias-in-toxicity-classification`.

Data. The main component of data set includes a column of comment text for training. It also has several additional toxicity sub-type attributes. A subset of comments have identity labels which represent the identities mentioned in these comments.

Challenges. The challenge is to build a model which could recognize toxicity and minimizes its unintended bias with respect to the mentioned identities. It could be challenging because the optimal model should identify whether the mentioned identities are related to toxicity or not.

3 Solution

Model. The final model I use is a simple LSTM model which contains two bidirectional LSTM layers.

Implementation. I implemented the LSTM model using keras. The text pre-processing stage before training has several steps. The first step is to remove the punctuation. The second step is to clean the contractions to reduce the unknown word counts. Meanwhile, I also include the calculation of sample weights which aims to reduce the unintended bias by using the identity parameters. The code is available at `https://github.com/JackZhou303/CS-583-Deep-Learning/tree/master/Deep%20Learning%20Final%20Project`. I committed the code on Kaggle kernel. It took 1.67 hours to train the model.

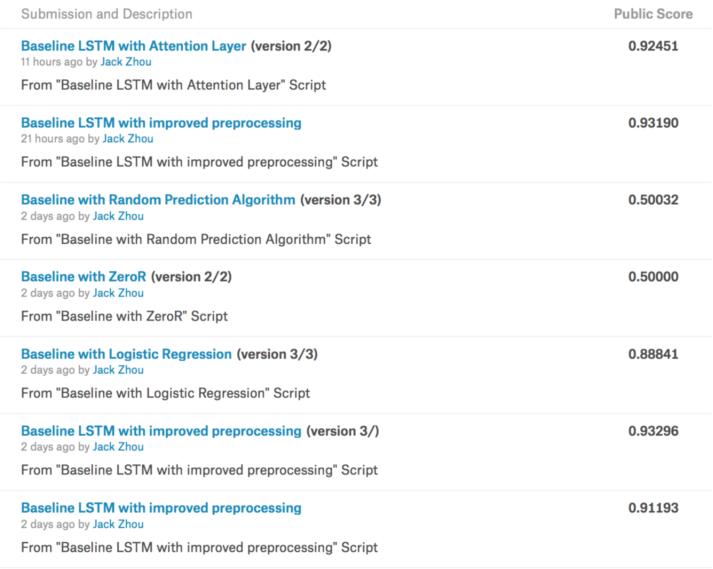
Settings. There are two loss functions used. The first one is custom loss function which targets the sample weight. The second one is binary cross-entropy. The chosen optimizer is Adam. The number of LSTM hidden units is 128. The learning rate is initialized as 0.001 and adjusted by learning rate scheduler over each iteration. The dropout layer takes 0.33 rate. The data is trained under two models. For each run of model, there are 4 global epochs. The batch size is 512. The max length for padding sequence is set to 220. After comparison, there is no limit over the number of max features to have a higher accuracy.

Tricks for improvements. I did not attempt any advanced trick, but several improvements over the base model. The first attempt was to reduce the unknown words by using the stemming words with NLTK. However, because of its long running time, this trick was not used. Instead I slightly reduced the number of unknown works by cleaning the contractions. On the other hand, I also considered to calculate the weights of identity parameters. It helps to reduce the potential bias.

Cross-validation. I attempted to tune the parameters by using 5-fold cross-validation. In reality, since the data set is quite big, the program will take too long to run. Meanwhile, because the competition requires the run-time to be within 2 hours, it is less desirable to include cross-validation to help with the prediction. So I tried simple validation split to help adjusting the drop out layer slightly. On the other hand, when trying the LSTM model with Attention layer, I included a 4-fold cross-validation and 2 epochs for each fold. This is the best I could do to run the code within 2 hours.

4 Compared Methods.

I have tried a total of four other methods for comparison. I tried three baseline models. The first models uses zero rule algorithm. Its accuracy is at 0.500. The second model uses random prediction algorithm with accuracy of 0.50032. The third model is a simple logistic regression model with the accuracy of 0.88841. It is clear that my final result has beaten these three baseline models. One the other hand, I also tried to add an attention layer to the original LSTM model. The result accuracy is 0.92451 without using the sample weights from identity parameters. The list of my attempts is shown in Figure 1.

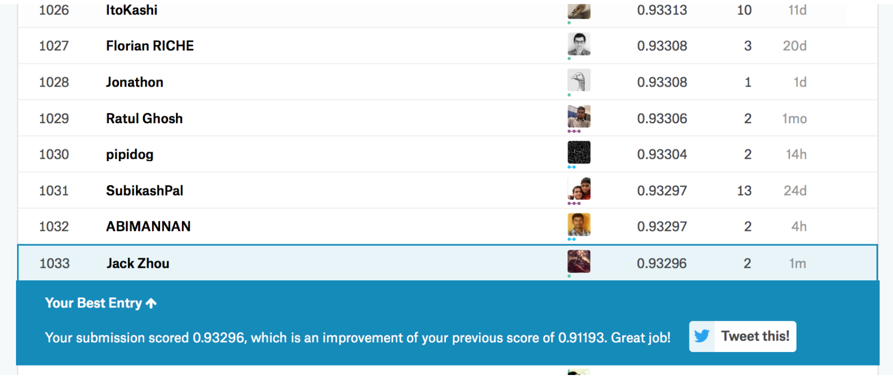


Submission and Description	Public Score
Baseline LSTM with Attention Layer (version 2/2) 11 hours ago by Jack Zhou From "Baseline LSTM with Attention Layer" Script	0.92451
Baseline LSTM with improved preprocessing 21 hours ago by Jack Zhou From "Baseline LSTM with improved preprocessing" Script	0.93190
Baseline with Random Prediction Algorithm (version 3/3) 2 days ago by Jack Zhou From "Baseline with Random Prediction Algorithm" Script	0.50032
Baseline with ZeroR (version 2/2) 2 days ago by Jack Zhou From "Baseline with ZeroR" Script	0.50000
Baseline with Logistic Regression (version 3/3) 2 days ago by Jack Zhou From "Baseline with Logistic Regression" Script	0.88841
Baseline LSTM with improved preprocessing (version 3/) 2 days ago by Jack Zhou From "Baseline LSTM with improved preprocessing" Script	0.93296
Baseline LSTM with improved preprocessing 2 days ago by Jack Zhou From "Baseline LSTM with improved preprocessing" Script	0.91193

Figure 1: The list of attempted methods and performance

5 Outcome

I participated in an active competition. My best accuracy is 0.93296 in the public leaderboard. I rank 1033/2010 in the public leaderboard upon submission. The screenshot is in Figure 2.



1026	ItoKashi	0.93313	10	11d
1027	Florian RICHE	0.93308	3	20d
1028	Jonathon	0.93308	1	1d
1029	Ratul Ghosh	0.93306	2	1mo
1030	pipidog	0.93304	2	14h
1031	SubikashPal	0.93297	13	24d
1032	ABIMANNAN	0.93297	2	4h
1033	Jack Zhou	0.93296	2	1m

Your Best Entry ↑

Your submission scored 0.93296, which is an improvement of your previous score of 0.91193. Great job!

Figure 2: The ranking in the leaderboard.

6 Reason for the Change of Competition

I picked the PetFinder competition in my initial proposal. As I was working on the project, I noticed the competition was closed and stopped allowing any submission. So I decided to immediately change the focus to another active competition, Jigsaw Unintended Bias in Toxicity Classification.