Toxic Comment Classification

Author: Jampala, Sai Krishna Anudeep

Contents

Introduction

This project is based on Kaggle Competition, “Toxic Comment Classification”. We built a model to classify the comments on Social Media based on the Toxicity. We are considering six types of Toxicity. They are Toxic, Severe Toxic, Obscene, Threat, Insult, Identity Hate. Each comment could belong to none, one or more classes. So, the problem will be multiclass classification. Two Datasets, Training and Testing are provided. Goal is to build a model to classify the comments based on toxicity. We train the model on the Training Dataset and test it on the test dataset. After trying several methods, we built our final model with Long term Short term Memory (LSTM).

Data

Our Data consists of Two Data sets, Train and Test Data sets. We train our model on Train Data-set and tested it on Test Data-set.

Train Data set contains 95851 observations and 8 columns. The columns are Id, Comment_text, Toxic, Severe Toxic, Obscene, Threat, Insult, Identity Hate. The following is a snap shot of the Training Dataset.

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
0	22256635	Nonsense? kiss off, geek. what I said is true...	1	0	0	0	0	0
1	27450690	"\n\n Please do not vandalize pages, as you di...	0	0	0	0	0	0
2	54037174	"\n\n ""Points of interest"" \n\nI removed the...	0	0	0	0	0	0
3	77493077	Asking some his nationality is a Racial offenc...	0	0	0	0	0	0
4	79357270	The reader here is not going by my say so for ...	0	0	0	0	0	0

Test Data set contains 226998 observations and 2 columns. The columns are Id, Comment_text. The following is a snap shot of the Testing Dataset.

	id	comment_text
0	6044863	==Orphaned non-free media (Image:41cD1jboEvL. ...
1	6102620	::Kentuckiana is colloquial. Even though the ...
2	14563293	Hello fellow Wikipedians,\nI have just modifie...
3	21086297	AKC Suspensions \nThe Morning Call - Feb 24, 2...
4	22982444	== [WIKI_LINK: Talk:Celts] ==

Each Comment could belong to None, One or More classes. There are no missing values in the data.

Data Preprocessing

I have created a function **"txtprepros"** to preprocess. It preprocesses text in following ways:

- Converting all letters to Lowercase.
- Replacing some symbols with space. We have used regular expressions to remove some symbols and replace them with space.
- Removing the Stop words from text. I have downloaded NLTK stop words corpus. Using this corpus, common stop words have been removed.

Tokenizing

- I have used "Keras Tokenizer" to tokenize the preprocessed text into words

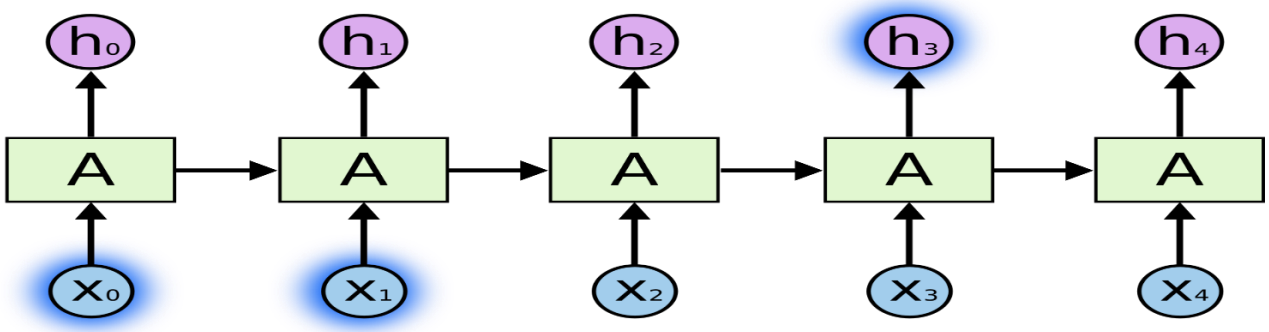
Model:

We have built an LSTM Model for this Classification Problem.

LSTM:

LSTM stands for "Long Short-term Memory". These units are a building unit for layers of a Recurrent Neural Network(RNN). A RNN composed of LSTM units is often called an LSTM network. A common LSTM unit is composed of a Cell, an Input Gate, an output Gate and a Forget Gate. The cell is responsible for "remembering" values over arbitrary time intervals; hence the word "memory" in LSTM. Each of the three *gates*can be thought of as a "conventional" artificial neuron, as in a multi-layer (or feedforward) neural network: that is, they compute an activation(using an activation function) of a weighted sum. Intuitively, they can be thought as *regulators* of the flow of values that goes through the connections of the LSTM; hence the denotation "gate". There are connections between these gates and the cell.

The expression *long short-term* refers to the fact that LSTM is a model for the Short-term memory which can last for a *long* period of time. An LSTM is well-suited to Classify, Process and Predict time series given time lags of unknown size and duration between important events. LSTMs were developed to deal with the exploding and vanishing gradient problem when training traditional RNNs.



Steps of Model Building:

- First, we have limited the maximum length to 99th percentile of the length of the comments. Here it was 312 words.
- In next step, we pad the comments to the maximum length i.e., We truncate the comment length to 312 words, if it is greater than 312. If comment length is less than 312 words, null values would be given to empty slots.
- Next, we create an embedding layer. This layer reduced the dimensions given to the words. Here we chose output dimensions to 50.
- In next step, we fix the number of memory cells in LSTM to 32.
- Then we instantiate the Model and in the next step, we compiled the Model. We have used "RMSPROP" as optimizer, "Accuracy" as metric and have give same weight to all classes.

Layer (type)	Output Shape	Param #	Connected to
main_input (InputLayer)	(None, 312)	0	
embedding_1 (Embedding)	(None, 312, 50)	9105750	main_input[0][0]
lstm_1 (LSTM)	(None, 32)	10624	embedding_1[0][0]
toxic_output (Dense)	(None, 1)	33	lstm_1[0][0]
severe_toxic_output (Dense)	(None, 1)	33	lstm_1[0][0]
obscene_output (Dense)	(None, 1)	33	lstm_1[0][0]
threat_output (Dense)	(None, 1)	33	lstm_1[0][0]
insult_output (Dense)	(None, 1)	33	lstm_1[0][0]
identity_hate_output (Dense)	(None, 1)	33	lstm_1[0][0]

Total params: 9,116,572
Trainable params: 9,116,572
Non-trainable params: 0

None

- In the Final step, we fitted the model with the data. We have used one epoch and batch size was 32.

```
Epoch 1/1
159571/159571 [==============================] - 3526s 22ms/step - loss: 0.8547 - toxic_output_loss: 0.3167 - severe_toxic_outp
ut_loss: 0.0571 - obscene_output_loss: 0.2080 - threat_output_loss: 0.0224 - insult_output_loss: 0.1975 - identity_hate_output_
loss: 0.0528 - toxic_output_acc: 0.9042 - severe_toxic_output_acc: 0.9901 - obscene_output_acc: 0.9470 - threat_output_acc: 0.9
969 - insult_output_acc: 0.9507 - identity_hate_output_acc: 0.9909
```

Results

- Here area under ROC curve is considered as a Metric, and my Result was 0.5107 which was low.

Recommendations to Improve

- We could use more Epochs to increase the model's performance.
- Instead of using same weights to every class, we could assign weights depending on the frequency of each class.
- We can change the Hyper parameters to maximize the performance.
- Using advanced pre-processing can improve results significantly.

References:

- https://en.wikipedia.org/wiki/Long_short-term_memory
- https://www.google.com/search?biw=1536&bih=734&tbm=isch&sa=1&ei=65_3Wo2GA-rp_Qb8yanQDQ&q=lstm+network&oq=lstm&gs_l=img.3.1.0j0i67k1j0l5j0i67k1j0l2.1714.2644.0.4494.4.4.0.0.0.0.73.264.4.4.0....0...1c.1.64.img..0.4.263....0.ol4AhpuveYI#imgrc=IrwppVe-yiuXjM:
- https://github.com/SaiKrishnaAnudeepJ/Toxic-Comment-Classification-Using-LSTM
- https://www.kaggle.com/c/jigsaw-toxic-comment-classification-challenge