Model Report

Michael Dowd, Napier Number: 40451589

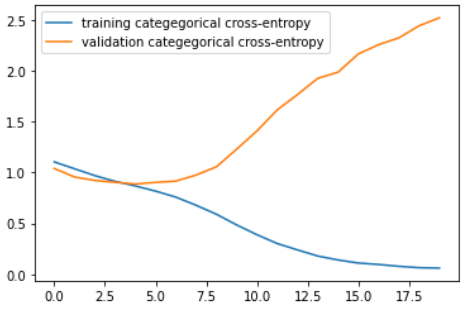
BERT is a neural network pre-trained on the Wikipedia and BooksCorpus data on the tasks of predicting missing words and sentence-matching. Pre-trained BERT models can be used simply as feature generators with excellent results. However the authors of the BERT paper achieved the best results on various NLP tasks after fine-tuning the model over a small number of epochs, this is the approach I took for this exercise.

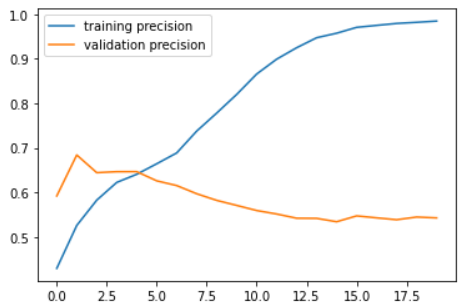
First, the pre-trained weights were downloaded as a keras layer from tensorflow hub using this link: https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/1. I then appended additional layers to the model to augment it for the abusive language classification task. I tried a number of configurations for the additional layers, however ultimately I achieved the best results using a configuration drawn from an example on the bert-for-tf2 github repo: https://github.com/kpe/bert-for-tf2/blob/master/examples/gpu_movie_reviews.ipynb. The model architecture is shown below, note the two dropout layers to avoid over-fitting:

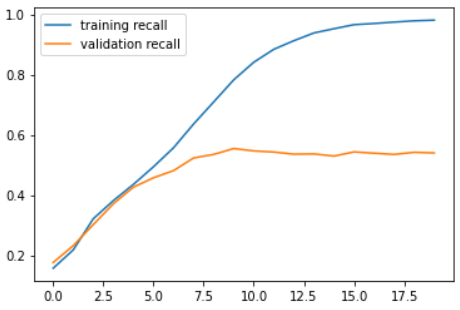
```
Model: "model_3"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_token_ids (InputLayer)    [(None, 128)]        0
__________________________________________________________________________________________________
input_token_masks (InputLayer)  [(None, 128)]        0
__________________________________________________________________________________________________
input_sent_segms (InputLayer)   [(None, 128)]        0
__________________________________________________________________________________________________
keras_layer_2 (KerasLayer)      [(None, 768), (None, 109482241   input_token_ids[0][0]
                                                                 input_token_masks[0][0]
                                                                 input_sent_segms[0][0]
__________________________________________________________________________________________________
lambda_3 (Lambda)               (None, 768)          0           keras_layer_2[0][1]
__________________________________________________________________________________________________
dropout_6 (Dropout)             (None, 768)          0           lambda_3[0][0]
__________________________________________________________________________________________________
dense_6 (Dense)                 (None, 768)          590592      dropout_6[0][0]
__________________________________________________________________________________________________
dropout_7 (Dropout)             (None, 768)          0           dense_6[0][0]
__________________________________________________________________________________________________
dense_7 (Dense)                 (None, 3)            2307        dropout_7[0][0]
==================================================================================================
Total params: 110,075,140
Trainable params: 110,075,139
Non-trainable params: 1
```

The BERT paper recommends some training parameters which the authors used to achieve good results during fine-tuning. Following these guidelines, I trained the model with an Adam optimiser and a small learning rate of of 2e-5 to minimise *Categorical Cross-Entropy*. I also used a batch size of 32. Finally the paper recommends training for just 2, 3 or 4 epochs, I implemented early stopping on the minimum value the validation loss, which occurred at epoch 5.

During training, 25% of the *training* data was held out for validation and early-stopping. From the charts and output below you can see that the validation precision and recall essentially level out after a number of epoch. The validation loss function, categorical cross-entropy, decreases to a minimum at epoch 5 and increases thereafter while training loss continues to decrease, suggesting that after epoch 5 the model is overfitting to the training data.







```
Epoch 1/10
225/225 [==============================]     val_categoricalcrossentropy: 1.0389 - val_categoricalaccuracy: 0.4533 - val_precision: 0.5919
Epoch 2/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9555 - val_categoricalaccuracy: 0.5108 - val_precision: 0.6839
Epoch 3/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9209 - val_categoricalaccuracy: 0.5337 - val_precision: 0.6441
Epoch 4/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9034 - val_categoricalaccuracy: 0.5638 - val_precision: 0.6463
Epoch 5/10
225/225 [==============================]     val_categoricalcrossentropy: 0.8861 - val_categoricalaccuracy: 0.5758 - val_precision: 0.6464
Epoch 6/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9022 - val_categoricalaccuracy: 0.5675 - val_precision: 0.6258
Epoch 7/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9151 - val_categoricalaccuracy: 0.5779 - val_precision: 0.6151
Epoch 8/10
225/225 [==============================]     val_categoricalcrossentropy: 0.9745 - val_categoricalaccuracy: 0.5679 - val_precision: 0.5968
Epoch 9/10
225/225 [==============================]     val_categoricalcrossentropy: 1.0551 - val_categoricalaccuracy: 0.5713 - val_precision: 0.5819
Epoch 10/10
225/225 [==============================]     val_categoricalcrossentropy: 1.2290 - val_categoricalaccuracy: 0.5654 - val_precision: 0.5706
```

For evaluation, a test set of 20% (or 2400 samples) of the total cleaned data was held apart from the training and validation set. First, baseline predictions were generated by randomly predicting labels for these test samples. As expected, given 3 labels, a random predictor achieves a weighted average precision, recall and f1-score of 33%.

The trained model significantly outperforms the baseline and achieves a weighted average f1-score of 58%. Interestingly the model achieves the best results on the Non-Aggressive samples (NAG, f1-score: 70%), second best results on the Covertly Aggressive samples (CAG, f1-score: 52%) and the worst results on the Overtly Aggressive samples (OAG, f1-score of 45%).

Classification Report - Baseline

	precision	recall	f1-score	support
CAG	0.33	0.32	0.33	817
NAG	0.41	0.33	0.37	1041
OAG	0.22	0.32	0.26	542
accuracy			0.32	2400
macro avg	0.32	0.32	0.32	2400
weighted avg	0.34	0.32	0.33	2400

Classification Report - Model

	precision	recall	f1-score	support
CAG	0.47	0.58	0.52	817
NAG	0.70	0.69	0.70	1041
OAG	0.57	0.37	0.45	542
accuracy			0.58	2400
macro avg	0.58	0.55	0.55	2400
weighted avg	0.59	0.58	0.58	2400

The confusion matrix provides more detail:

```
Confusion Matrix
['CAG', 'NAG', 'OAG']
[[475 228 114]
 [281 721  39]
 [265  78 199]]
```

- Of the 817 CAG samples, 228 were incorrectly predicted as being NAG while 114 were incorrectly predicted as being OAG
- Of the 1041 NAG samples, only 39 of them were incorrectly predicted as being OAG, while 281 were incorrectly predicted as being CAG.
- Of the 542 OAG samples only 78 of them were incorrectly predicted as being NAG, while 265 of them were incorrectly predicted as being CAG.

The model clearly has some difficulty differentiating overt aggression from covert aggression. Also, for non-aggressive comments the model is much more likely to mis-label them as covertly aggressive than overtly aggressive. This is inline with our understanding of the labels – covert aggression is a subtle concept even for a human to understand in some cases.