Model Report

Michael Dowd, Napier Number: 40451589

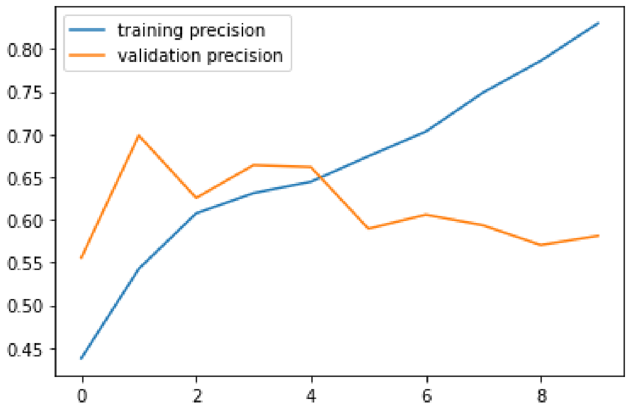
BERT is a neural network pre-trained on the Wikipedia and BooksCorpus data on the tasks of predicting missing words and sentence-matching. Pre-trained BERT models can be used simply as feature generators with excellent results. However the authors of the BERT paper achieved the best results on various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

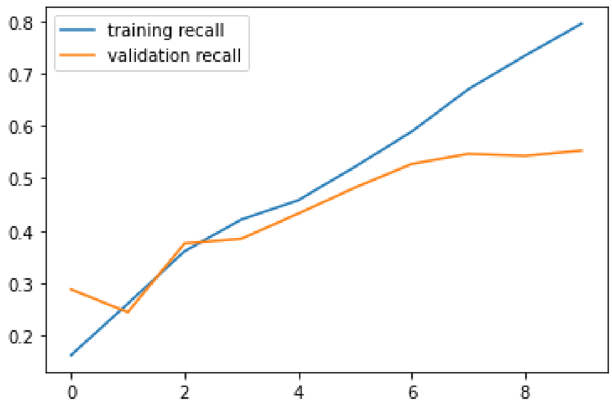
First, the pre-trained weights were downloaded as a keras layer from tensorflow hub using this link: https://tfhub.dev/tensorflow/bert_en_uncased_L-12_H-768_A-12/1. I then appended additional layers to the model to augment it for the abusive language classification task. I tried a number of configurations 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, however I achieved better results by training for up to 10 epochs.

During training, 25% of the *training* data was held out for validation purposes. From the charts and output below you can see that the validation precision and recall are essentially decreasing and increasing respectively along with number of epochs, the task therefore being then to find the epochs which maximise f1-score. After 4 epochs the recall on the validation set is only 38.46%, however at epoch 10 the validation precision and recall are 58.10% and 55.29%.






```
Epoch 1/10
225/225 [==============================]    val_precision: 0.5555 - val_recall: 0.2879
Epoch 2/10
225/225 [==============================]    val_precision: 0.6988 - val_recall: 0.2446
Epoch 3/10
225/225 [==============================]    val_precision: 0.6255 - val_recall: 0.3758
Epoch 4/10
225/225 [==============================]    val_precision: 0.6640 - val_recall: 0.3846
Epoch 5/10
225/225 [==============================]    val_precision: 0.6616 - val_recall: 0.4325
Epoch 6/10
225/225 [==============================]    val_precision: 0.5894 - val_recall: 0.4821
Epoch 7/10
225/225 [==============================]    val_precision: 0.6058 - val_recall: 0.5271
Epoch 8/10
225/225 [==============================]    val_precision: 0.5934 - val_recall: 0.5467
Epoch 9/10
225/225 [==============================]    val_precision: 0.5702 - val_recall: 0.5429
Epoch 10/10
225/225 [==============================]    val_precision: 0.5810 - val_recall: 0.5529
```

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 close to 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: 69%), second best results on the Overtly Aggressive samples (OAG, f1-score: 50%) and the worst results on the Covertly Aggressive samples (CAG, f1-score of 48%).

```
Classification Report - Baseline
              precision    recall  f1-score   support

         CAG       0.33      0.33      0.33       817
         NAG       0.46      0.35      0.39      1041
         OAG       0.23      0.34      0.27       542

    accuracy                           0.34      2400
   macro avg       0.34      0.34      0.33      2400
weighted avg       0.36      0.34      0.35      2400

Classification Report - Model
              precision    recall  f1-score   support

         CAG       0.45      0.52      0.48       817
         NAG       0.69      0.69      0.69      1041
         OAG       0.56      0.45      0.50       542

    accuracy                           0.57      2400
   macro avg       0.57      0.55      0.56      2400
weighted avg       0.58      0.57      0.58      2400
```

The confusion matrix illustrates this in more detail. Of the 1041 NAG samples, only 48 of them were incorrectly predicted as being OAG, while 277 were incorrectly predicted as being CAG. Of the 542 OAG samples only 71 of them were incorrectly predicted as being NAG, while 229 of them were incorrectly predicted as being CAG. Clearly the main difficulty the model had is in differentiating covertly aggressive comments from non or overtly aggressive comments.

```
Confusion Matrix
['CAG', 'NAG', 'OAG']
[[422 256 139]
 [277 716  48]
 [229  71 242]]
```

This is inline with our understanding of the labels – covertly aggressive text would be more difficult even for a human to identify.