Intent Classifier: Oracle coding test

Abdul Karim

17 Oct, 2020

1 Abstract

This report is related to BERT fine tuning for intent classification. It is a part of ORACLE 24 hours coding challenge. I fine tuned BERT model using TITAN Xp GPU machine and achieved in scope accuracy of 0.77 whereas out of scope Precision and Recall of 0.85 and 0.29 respectively on test set in 100 epochs of training. These results can be improved by better fine tuning with regularizations , early stopping methods and hyper-parameters. The results we obtained are reasonable and shows the potential performance of BERT model on imbalanced dataset for intent classification.

2 Data Collection

The data set is collected from the [1] for training and testing the model. In the original paper, the authors talks about the using this dataset for measuring in scope and out of scope accuracy of intent classification model. The imbalanaced dataset is accessed from the GitHub repository `https://github.com/clinc/oos-eval/blob/master/data/data_imbalanced.json` using Jupyter Notebook. The dataset consists of 6 sets (train, validation, test, train_oos, validation_oos, and test_oos). The size of each of these data sets is shown in the Table 1.

Table 1: Imbalanced data sets sizes

train	val	test	train_oos	val_oos	test_oos
10525	3000	4500	100	100	1000

The out of scope refers to a situation where the intent is not known. It should also be noted that the data in highly imbalanced as number of sentences in some intents is larger then the others. For instance, report fraud intent class has 25 sentences whereas gas type intent has 100 sentences in the train data set. The number of sentences for each intent class is 20 in validation and 30 in test. For out of scope datasets, the intent class is always the same which is oos. For

3 Data Preprocessing

The data in all data sets is first checked for null values and it is found that there is no null value in any of the dataset. As per the three strategies of training for dealing with the out of scope sentences [1], i trained the BERT model by considering the oos intent as 151th intent in the training data. The data was converted into pandas data-frames and respective sets were concatenated for this purpose. Thus three data sets were prepared named train, test and validation. Each set consisted of 151 intent classes by considering the oos as one of the intents.

4 Data Tokenization

As discussed earlier, I chose BERT model to be fine tuned for this purpose. A small uncased version pre-trained BERT model was downloaded from `https://github.com/google-research/bert`. It comes with its vocabulary and trained weights. Each of the sentence in all three datasets were tokenized using BERT tokenizer and converted into numerical values (each word). The length of the longest sentence in all datasets is found to be 33. The smaller sentences were padded and start and end identification token were included in each sentence. Thus we obtain a train data set with shape (10625,33), validation data set with shape (3100,33) and test data set with a shape (5500, 33). It should be noted that each intent class is also converted into a numerical label.

5 Data Imbalance

Each intent class was assigned with a class weight using sklearn class weight function `https://scikit-learn.org/stable/modules/generated/sklearn.utils.class_weight.compute_class_weight.html`. Thus major classes were assigned with less weight and minor classes were assigned with more weight. For instance, direct_deposit class intent was assigned a weight equal to 1.40 whereas oos class intent was assigned a weight equal to 0.70. Details of each class intent is given in the jupyter notebook with this report.

6 BERT model fine tuning

I chose BERT model because of its superior performance on various NLP tasks [1]. BERT stands for Bidirectional Encoder Representations from Transformers , which was introduced in the the paper by google [2] in 2018. It uses transformer architecture and bi-directional training and contextual word embeddings. Google has made the source code and the trained weights open source for the use of various NLP tasks `https://github.com/google-research/bert`.

I downloaded the one of the open source simple BERT model from `https://github.com/google-research/bert` originally published by Turc in arXive

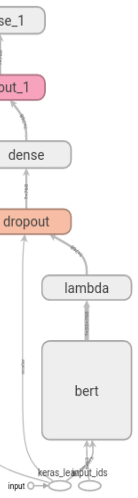

Figure 1: BERT fine tuned

[3]. The model was appended with one fully connected layer sandwitched by drop-out layer. Then a softmax function was used as a multi-class output. It should be noted that I used TensorFlow Version: 2.3.0 and Keras Version: 2.4.0 as a deep learning framework for this task. Model was compiled with an Adam optimizer with a learning rate value of 1e-5, categorical cross entropy as loss function with accuracy as a metric. The batch size was chosen to be 16 and the data was shuffled. It should be noted that the model was trained with the class weights which catered for the class imbalance problem. Validation data was used to validate the model and test was use to test the model. Total of 100 epochs were used with no early stopping. I used Titan xp (I love my machine), to fine tune the BERT Model for this task. A tesnorboard generated model structure is shown in Figure 1.

7 Results

A tensorboard was used monitor the loss and accuracy for each epoch for test and train set as shown in Figure 2. The accuracy for both train and validation increases with each epoch and the loss is decreasing till it becomes saturated. It is evident from the plots in Figure 2 that there is over-fitting of the model as the train accuracy is much higher the validation accuracy. The train accuracy reaches to 0.98 where the validation accuracy flattens at o.85. This over fitting can be reduced by early stopping, regularization techniques or using smaller model. The orange colour represents validation set and the grey colour represents the train set.

Besides accuracy, I also computed out of the scope (oos) precision, recall and

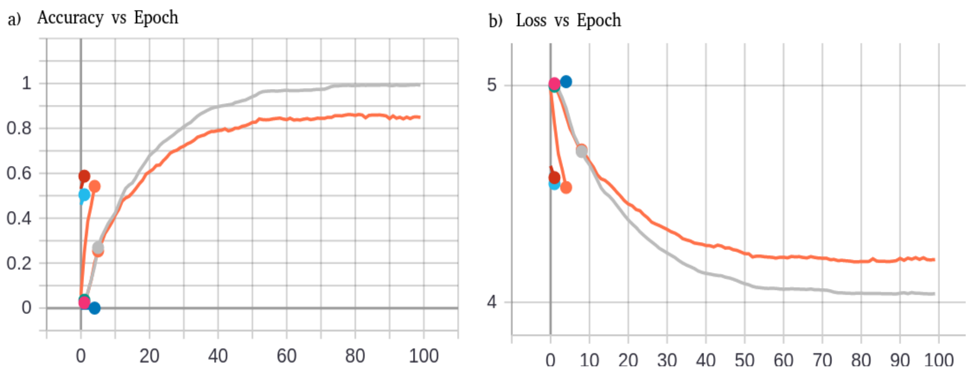

Figure 2: Accuracy and Loss

F1 score by considering oos as 151th intent class as described in the reference paper given for this task [1] as shown in the Table 2.

Table 2: In scope accuracy and out of scope Precision and Recall on the test set

Model name	In scope accuracy	oos precision	oos recall	oos F1 score
in this task	0.77	0.85	0.29	0.43
in reference paper [1]	0.96	NA	0.44	NA

The results can be improved with better fine tuning, using various types of kernal and layer regularizers, using more fit hyper-parameters search and early stopping. The model we trained under-performs the state of the art in all the metrics for test data as shown in Table 2. The possible reason is because for this task, I just wanted to see the potential of the BERT fine tuned without further optimization or regularization or early stopping. The higher precision shows that our classifier is very exact for oos intent class. On the other hand, recall value is low for oos intent class which shows that our classifier is not very sensitive for oos intent class as compared to the one trained by the authors of the reference paper [1]. F1 score is the combined measure of precision and recall.

8 Conclusion

As mentioned before, the accuracy both in scope and out of scope can be improved using regularizers, early stopping, various drop out layer and smaller architecture. BERT is overall very impressive transformer based model but it is over trained in this task. Even though a simple version of BERT is used, still it contains millions of parameters (109,596,823) in our case which might create delays in inference. For that purpose, we can implement pruning techniques because among the many parameters in the network, some are redundant and don't contribute a lot to the output, so they can be dropped. Thus model inference can be made less compute intensive and efficient. We can rank the neurons in the network according to how much they contribute, we could then remove the low ranking neurons from the network, resulting in a smaller and faster network. Furthermore, half precision training substantially will also help in reducing the model training memory and time.

References

[1] Stefan Larson, Anish Mahendran, Joseph J Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K Kummerfeld, Kevin Leach, Michael A Laurenzano, Lingjia Tang, et al. An evaluation dataset for intent classification and out-of-scope prediction. *arXiv preprint arXiv:1909.02027*, 2019.

[2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[3] Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-read students learn better: On the importance of pre-training compact models. *arXiv preprint arXiv:1908.08962v2*, 2019.