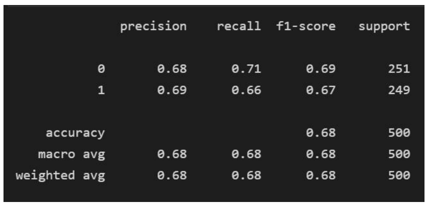
MSBD 5012 Machine Learning Homework 5 Report

Student: XXXXXXXX   ID: 12345678

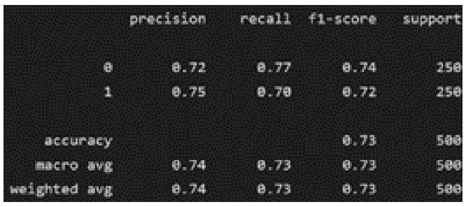
In this homework, I use the naïve BERT-ARCH model in the tutorial for sentiment analysis and change test different hyperparameters. I summarize the result in the coming chapters. **In the beginning I talk about the influence of extra parameters.** The first part talks about the variation of the learning rate, the second part is about the variation of the batch size, the third part shows the variation of the dropout rate, and the last part is about the variation of the maximum number of epochs.

Beginning: Influence of Maximum Sequence Length

I start from the naïve BERT-ARCH model. **I change the max_seq_len to 128 for a better performance.** Because if we use a smaller max_length, the sentence will be cut into more pieces. It’ s bad for model performance. The comparison of the two choice can be seen below. The left picture is the result with max_seq_len 25, and the right is the result with max_seq_len 128.



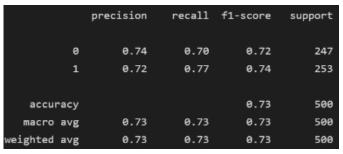
```
              precision    recall  f1-score   support

           0       0.68      0.71      0.69       251
           1       0.69      0.66      0.67       249

    accuracy                           0.68       500
   macro avg       0.68      0.68      0.68       500
weighted avg       0.68      0.68      0.68       500
```



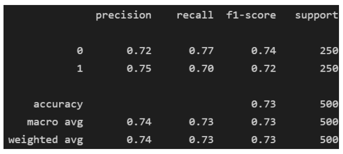
```
              precision    recall  f1-score   support

           0       0.72      0.77      0.74       250
           1       0.75      0.70      0.72       250

    accuracy                           0.73       500
   macro avg       0.74      0.73      0.73       500
weighted avg       0.74      0.73      0.73       500
```

Part One: Variation of the Learning Rate

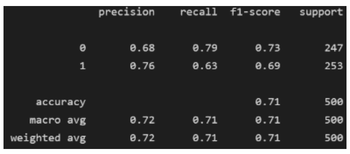
In this part I choose the learning rate to be 1e-2, 1e-3, and 1e-4. The result is shown below.



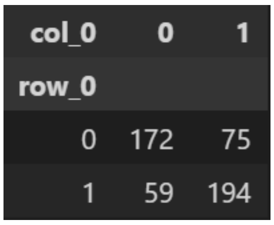
```
              precision    recall  f1-score   support

           0       0.74      0.70      0.72       247
           1       0.72      0.77      0.74       253

    accuracy                           0.73       500
   macro avg       0.73      0.73      0.73       500
weighted avg       0.73      0.73      0.73       500
```



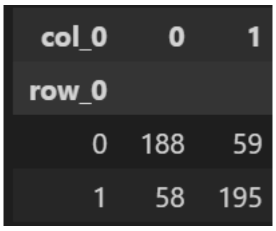
```
              precision    recall  f1-score   support

           0       0.72      0.77      0.74       250
           1       0.75      0.70      0.72       250

    accuracy                           0.73       500
   macro avg       0.74      0.73      0.73       500
weighted avg       0.74      0.73      0.73       500
```



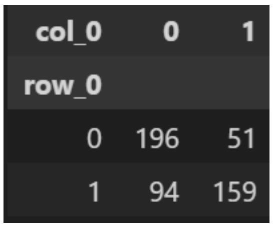
```
              precision    recall  f1-score   support

           0       0.68      0.79      0.73       247
           1       0.76      0.63      0.69       253

    accuracy                           0.71       500
   macro avg       0.72      0.71      0.71       500
weighted avg       0.72      0.71      0.71       500
```



col_0	0	1
row_0		
0	172	75
1	59	194



col_0	0	1
row_0		
0	188	59
1	58	195

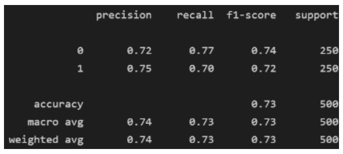


col_0	0	1
row_0		
0	196	51
1	94	159

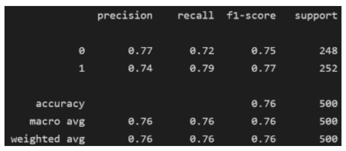
From the result above, we can find that when the learning rate varies, model performance also varies. A smaller learning rate may lead to a stuck in saddle point, and a larger rate may lead to a frustration around the local minimum. When the learning rate is small, we get a higher recall, but a lower precision. When the learning rate is large, we get a higher precision, but a lower recall. So we should use a proper learning rate to make tradeoff of these values. We can find that **best learning rate may be around 1e-3**.

Part Two: Variation of the Batch Size

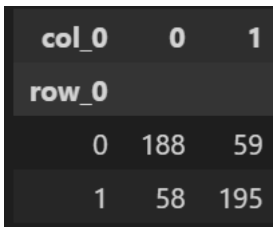
In this part I choose the batch size to be 32 and 64. The result is shown as follows.



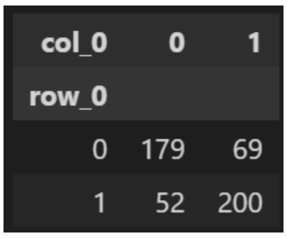
```
              precision    recall  f1-score   support

           0       0.72      0.77      0.74       250
           1       0.75      0.70      0.72       250

    accuracy                           0.73       500
   macro avg       0.74      0.73      0.73       500
weighted avg       0.74      0.73      0.73       500
```



```
              precision    recall  f1-score   support

           0       0.77      0.72      0.75       248
           1       0.74      0.79      0.77       252

    accuracy                           0.76       500
   macro avg       0.76      0.76      0.76       500
weighted avg       0.76      0.76      0.76       500
```



col_0	0	1
row_0		
0	188	59
1	58	195



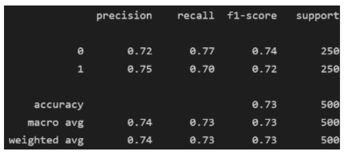
col_0	0	1
row_0		
0	179	69
1	52	200

After increasing batch size, we use more data simultaneously. The model will converge more precisely. From the result we can find that the precision and f1 score increase with the increase of batch size, but the recall becomes lower. From the confusion matrix we can also find that the number of misclassification increases with batch size increases. According to a study[1], increasing the batch size to a large scale may not lead to a better result. **A large batch size may converge to a sharp minimum, and a small batch size may converge to a flat minimum.**

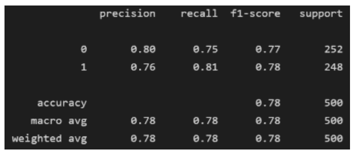
[1] Keskar N S, Mudigere D, Nocedal J, et al. On large-batch training for deep learning: Generalization gap and sharp minima[J]. arXiv preprint arXiv:1609.04836, 2016.

Part Three: Variation of the Dropout Rate

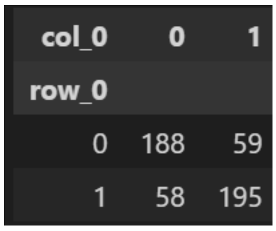
In this part I choose the dropout rate to be 0.1 and 0.2. The result is shown as follows.



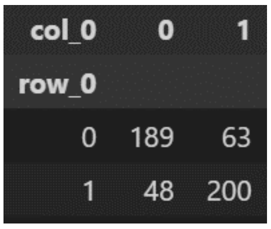
```
              precision    recall  f1-score   support

           0       0.72      0.77      0.74       250
           1       0.75      0.70      0.72       250

    accuracy                           0.73       500
   macro avg       0.74      0.73      0.73       500
weighted avg       0.74      0.73      0.73       500
```



```
              precision    recall  f1-score   support

           0       0.80      0.75      0.77       252
           1       0.76      0.81      0.78       248

    accuracy                           0.78       500
   macro avg       0.78      0.78      0.78       500
weighted avg       0.78      0.78      0.78       500
```



col_0	0	1
row_0		
0	188	59
1	58	195

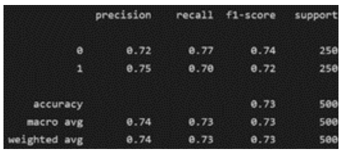


col_0	0	1
row_0		
0	189	63
1	48	200

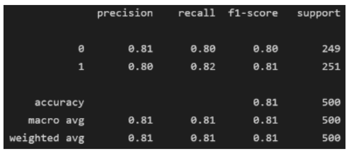
After increasing dropout rate, the neural network will temporarily dropout more neurons when training. This mechanism can alleviate the inner joint relationship of the neurons. So we can improve the generalization ability of our model. The result also demonstrates this. Although the recall decreases, the number of misclassification decreases.

Part Four: Variation of the Number of Epochs

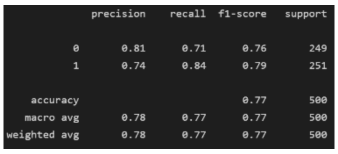
In this part I choose the number of epochs to be 10, 20, and 30. The result is shown as follows.



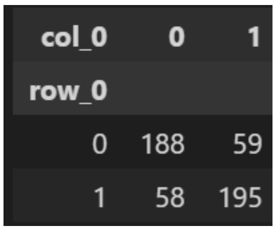
```
              precision    recall  f1-score   support

           0       0.72      0.77      0.74       250
           1       0.75      0.70      0.72       250

    accuracy                           0.73       500
   macro avg       0.74      0.73      0.73       500
weighted avg       0.74      0.73      0.73       500
```



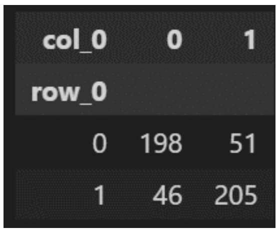
```
              precision    recall  f1-score   support

           0       0.81      0.80      0.80       249
           1       0.80      0.82      0.81       251

    accuracy                           0.81       500
   macro avg       0.81      0.81      0.81       500
weighted avg       0.81      0.81      0.81       500
```



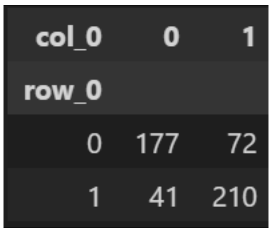
```
              precision    recall  f1-score   support

           0       0.81      0.71      0.76       249
           1       0.74      0.84      0.79       251

    accuracy                           0.77       500
   macro avg       0.78      0.77      0.77       500
weighted avg       0.78      0.77      0.77       500
```



col_0	0	1
row_0		
0	188	59
1	58	195



col_0	0	1
row_0		
0	198	51
1	46	205



col_0	0	1
row_0		
0	177	72
1	41	210

From the result above, we can find that after increasing the number of epochs to 20, the evaluation scores (including precision, recall, and f1 score) all increase. This is because **more epochs of training will make our model fit the data better.** However, we can see that after we set epochs 30, the recall and f1 score both decrease. And from the confusion matrix we can also find that the result seems to be worse. I think this is because **after training more, the model is going to be overfitted.**

Conclusion

In this experiment I start from the naïve BERT model and finetuned it with some hyperparameters. These hyperparameters may influence the model performance to a different extend. When we use the model afterwards, we should carefully search these hyperparameters to get a better result. However, **when we train a model with the same parameters twice, it may not get identical results.** This may also be an important thing for us to notice.