NLP Coursework Report

Hongji Wu
hw2121

Yanting Pan
yp1221

Yik-Kwan Chan
yc4421

1 Introduction

The use of patronizing and condescending language in the general media has long been proved to have negative consequences, even when it is done unintentionally and with good intentions. To predict whether a document contains patronising or condescending language to vulnerable communities (e.g. refugees, homeless people, poor families), we created a binary classification model. All code behind this report is available at this repo: *https://github.com/HongjiWu/DontPatronizeMeTask1* Our tame name is CPW, and the user name appears on the leaderboard is hjwu0420.

2 Data analysis of the training data

We employed the dataset *Don't Patronize Me!*, there are currently 10,637 paragraphs concerning potentially vulnerable social groupings. The presence of PCL at the text span level has been indicated in these paragraphs.

The labels of datasets are subjective because only three annotators annotate sentences, and the agreement after adjustment of two annotators is only 61%.

Label 0 has a far higher number of instances than the other labels. Label 0-1 has a total of 9476 in the dataset, but label 2-4 only has a total of 993, indicating that the dataset is unbalanced.

Table 1: performance

Label	Numbers	Percentage
0	8529	81.46%
1	947	9.05%
2	144	1.38%
3	458	4.37%
4	391	3.73%

3 Modelling

3.1 Model Comparison

According to the research *RoBERTa: A Robustly Optimized BERT Pretraining Approach*(**?**), RoBERTa outperforms XLNet. For proving that statement holds for this task, we put RoBERTa and XLNet to the test. And after hyper-parameter tuning, the results in Table 2 also demonstrates that Roberta is more effective in this task. And for comparing the cased RoBERTa model with an similar uncased model, we choose DistilBERT-uncased, and the result shown in the table below indicates that cased model is performing better than uncased one in this task. Therefore, we choose cased RoBERTa model as our major pretrained model for all following research.

Table 2: f1 score of different models

	RoBERTa	XLNet	DistilBERT
Official Dev Set	0.5782	0.5250	0.5162
Test Set	0.5469	0.5063	0.4969

3.2 Hyper-parameter Tuning

In order to do hyper-parameter tuning, based on the Roberta model, we changed the number of epochs from 1 to 15 times. The precision, recall and f1 score are shown in the figure below, we found that the model suffered from Overfitting after about 13 epochs.

Table 3: early_stopping

delta	0.01
metric	mcc
metric_minimize	True
patience	2

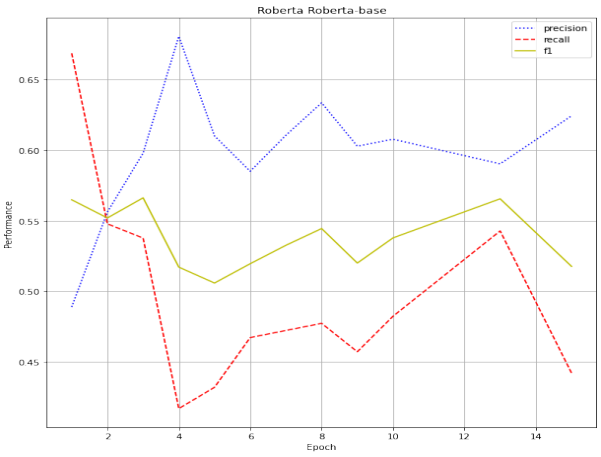

Figure 1: Roberta performance

Then we analyzed the situation with *early stopping*,

as shown in the following figure:

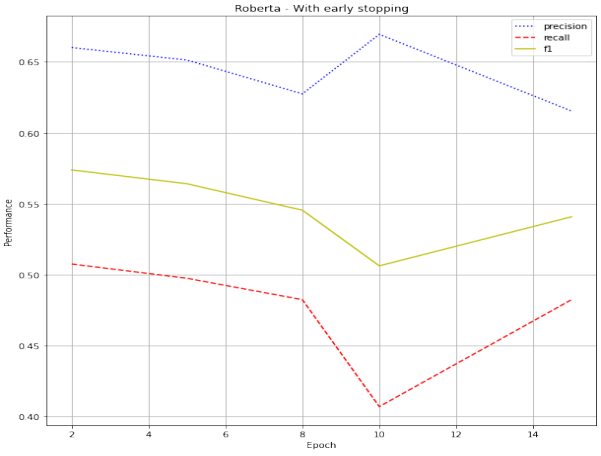

Figure 2: Roberta performance - with early stopping

We could see that with *early stopping*, the performance of f1 score is better than before. Thus, in the following comparison, we applied *early stopping*.

Finally we changed the *ClassificationModel* to *distilbert-base-uncased* model:

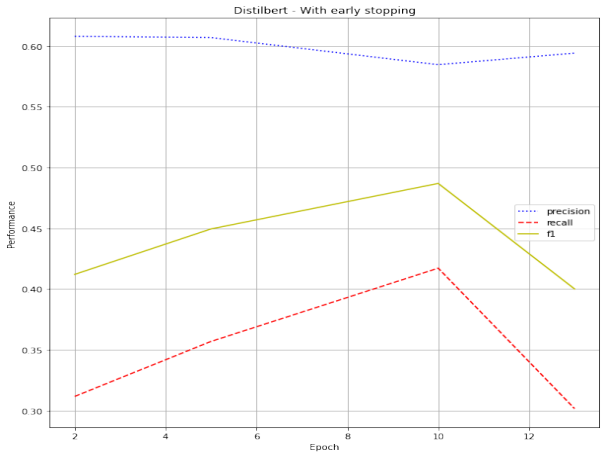

Figure 3: Distilbert-base-uncased

The performance is not satisfying. We turned back to *Roberta* model.

The figure of performance with different training batch size(8/32/64/128) is uploaded to github *https://github.com/HongjiWu/DontPatronizeMeTask1* due to the page limits.

To conclude, when we fixed our model to *Roberta* model, with *early stopping*, epoch time fixed to 8 and training batch size fixed to 64, we obtained our best performance.

3.3 Creative Direction for Improvement

For speeding up the training process in order to try more techniques and analysis, we only applied most of the following methods with the relatively simple RoBERTa baseline model. We believe similar results could be expected to appear on larger pre-trained models.

After analyzing the original dataset. The most challenging features we found is the relative small size and unbalanced distribution of the original database. There is only around 10000 datapoint for training, and only 10% of them have a label of 1. Therefore, for further improving the performance of our model, we believed preprocessed the dataset is more important than training on a larger transformer with even more parameters. The most effective techniques we applied is data augmentation. Since the original dataset has only about 10% positive datapoints, we try to upsampling its size to similar amount as the negative ones. We copied all positive datapoints 8 times so the amount of positive datapoints increases from **794** to **6352**, and the overall size of training dataset grow up to **12704**. Furthermore, we have a much evenly distributed dataset. By purely adopting this tech-

niques, we improve the performance of the baseline model from **48%** to **54%**. We also considered adding back-translation and data sampling into this model. However neither of them appears to have such propounding influences on the performance of our model. Nevertheless, we believe we could add back-translation into our previous process of data augmentation since translating text back and forth would naturally generate repetitive datapoints while keep the diversity of our input dataset.

4 Analysis

4.1 Different Levels of Patronizing Content

For testing out whether our model is better at predicting content with higher patronizing level, we store the original label of the official dev set, and classify them into different groups in order to calculate our model's predicting accuracy with different patronizing level. As we can see from the result table above, there is a clear trend to have higher predicting accuracy for the group that has higher patronizing level (and higher predicting accuracy for negative sample with lower patronizing level.)

Label	Predicting Accuracy(%)	Size
0	95.07	1704
1	76.96	191
2	33.33	18
3	59.55	89
4	80.43	92

Table 4: Patronizing Level

However, we should notice that the distribution of our dev dataset is very uneven, since over 1700 datapoints have label value = 0 while only dozens of datapoints with positive(greater or equal to 2) labels.

4.2 Length of Input Sequences

For testing out whether the length of input size affects our predicting accuracy, we calculate the median length of our input text and divide all our input sample into 2 separated training set then train RoBERTa model for each group. The result demonstrates that the length of input sequence don't have substantial influence over model performance.

4.3 Influence of Categorical Data

For analyzing the influence of categorical data provided by the dataset, we divide our training dataset into separate groups according to their keywords. Then comparing the performance between each model. As the result shown in the table below, the categorical data provided with input text would generate great influence over the overall performance of our model.

keyword	f1_score	Size
refugee	0.4652	1254
hopeless	0.3184	1365
vulnerable	0.4405	1151
disabled	0.3854	1176
migrant	0.4155	966
women	0.2721	967
poor-families	0.4052	1415
in-need	0.3799	1753
immigrant	0.3742	860
homeless	0.3796	1797

Table 5: Influence of Categorical Data

5 Conclusion

The final f1 score of our fine-tuned RoBERTa-cased model is 0.5469 on test set, and 0.5782 on the official dev dataset. Through researching over this topic, we find that data pre-processing might be more effective on improving model's overall performance than model selection and hyperparameter tuning for this specific test since the given training dataset is severely unbalanced. Through analyzing the following questions, we find that categorical data and different level of patronizing content in training dataset could have profound affect on model's performance. In the next step, we would like to research on how specifically did the difference in the mechanism of pre-trained model(like difference between XLNet and BERT), collaborate with the features of this task, dataset,and pre-processing techniques we applied, have influence the performance of the model.