Sentiment Analysis

1st Li-Xin, Ng
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
nlx1451@gmail.com

2nd Hao-Yun, Yen
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
xyhy108062213@gapp.nthu.edu.tw

3rd Chang-Chi, Wu
Interdisciplinary Program of
Electrical Engineering and Computer Science
National Tsing Hua University
Hsinchu, Taiwan
sam54537@gmail.com

4th Nai-Yi, Hsu
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
vmsu11051105@gmail.com

5th Hui-Chee, Ong
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
edeline2001@gmail.com

6th Grace, Chan
Department of Computer Science
National Tsing Hua University
Hsinchu, Taiwan
gracechan668@gmail.com

***Abstract*—Based on the standard of a machine learning final project, after discussion, we chose the topic of movie emotion review classification and participated in one of the competition projects opened by the teacher. Sentiment analysis is a very important category in the field of NLP, and many people are doing research in this area. Sentiment analysis is the process of detecting positive or negative sentiment in text. It's often used by businesses to detect sentiment in social data, gauge brand reputation, and understand customers. However, we hope that this project can solve the problem of positive and negative sentiment classification.**

I. Introduction

We want to find a model that can get the highest f1 score in this competition. After our survey, these four models were selected. We tested four models which are distilbert-base-uncased-finetuned-sst-2-english(pretrained model), TFBertForSequenceClassification.bert-base-uncased, distilbert-base-uncased-finetuned-sst-2-english(finetuned model by our data), muppet-roberta-large-finetuned(finetuned model by our data). The best model of this project is muppet-roberta-large-finetuned(finetuned model by our data).

II. Model

A. Distilbert Model

This model is a fine-tune checkpoint of DistilBERT-base-uncased, fine-tuned on SST-2. According to the model, the finetuning parameter are shown below.

learning rate = 1e-5
batch size = 32
warmup = 600
max seq length = 128
num train epochs = 3.0

With the pretrained model above, we are able to get a f1 score of 0.8980578. However we are not satisfied with it and we decided to trained this model ourselves.To do so, we first obtained our training data from the Aldea website split it into training data and validation data. Both data are then tokenized before training. After that, we import the model using tensorflow with the hyperparameter below.

learning rate = 2e-5
batch size = 16
num warmup steps = 0
num epochs = 5
num train epochs = 1

After some research and testing, we ended up with the hyperparameter above. At first we tried a larger batch size and number of epochs, however the increase of batch size used up too much memory and one epoch alone requires about 22 hours to run in Google Colab. After some trial and error, we decided to use GPU teaching resource service of NCHC instead of Google Colab, and train the model using the final hyperparameter above. It took us 3 and half hours to finish training our model, which is able to reach a f1 score of 0.9304511.

B. TFBertForSequenceClassification.bert-base-uncased

TFBertForSequenceClassification [6] is a pre-trained Bert model. It is a Bert Model transformer with a linear sequence classification/regression layer on top of the pooled output. Moreover, bert-base-uncased used to predict in English. To finetune the model for our dataset, we use below hyperparameters.

learning rate = 3e-5
epsilon = 1e-8
epoches = 2
train data batch size = 32
valid data batch size = 32

With these hyperparameters we can reach 0.8728 accuracy after 953 steps for one epoch. However, it cost too much resource and time, we tried to set up steps per epoch = 400. After that we can spend about 8 hours to finish our finetune

Identify applicable funding agency here. If none, delete this.

instead of over 24 hours. Unfortunately, using 953 steps for one epoch could only get 0.87 accuracy, let alone 400 steps for one epoch.

C. Muppet-Roberta-Large Model

This model is finetuned by many dataset such as RTE, BoolQ and others. The author of this model use the method of **MUPPET [3]** to finetuned this model. We finetuned the Muppet-Roberta-Large model to get the best result. We use AutoNLP platform to finetune our work.

III. Dataset

Our dataset is provided by AIdea website. According to its' statement, data are included in **Large Movie Review Dataset [4]** and **Stanford Sentiment Treebank(SST) [5]**.
In train data, each data owns its unique id, review comment, and a label for classifying positive or negative. In test data, each data consists of a unique id and a piece of review comment.

IV. Result

We use 2 way to evaluate our model, one is accuracy and the other one is f1 score.

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{1}$$

In f1 score, precision means number of true positive data (True Positive) divided by number of predicted positive data (True Positive+False Negative). Recall means number of correctly predict as positive data (True Positive) divided by number of positive data (True Positive+False Positive).

$$precision = \frac{TP}{TP + FN} \tag{2}$$

$$recall = \frac{TP}{TP + FP} \tag{3}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{4}$$

We tested four models in our project. Distilbert-base-uncased-finetund-sst-2-english model without finetuned by out data can get a score of 0.8980578. After finetuned by our data we are able to reach a score of 0.9304511. With TFBertForSequenceClassification.bert-base-uncased pre-trained model we can get 0.8728 accuracy after finetuning with 953 steps of one epoch. The best model we tested is Muppet-Roberta-Large Model after we finetuned by out data it can reach a score of 0.9690994. We submit out best model to Aldea and get both first place in pubic ranking and private ranking. In private data we can get 0.9652367 points.

Model	Score / Accuracy
distilbert-base-uncased-finetund-sst-2-english (pretrained)	0.8980578 (Score)
distilbert-base-uncased-finetund-sst-2-english (finetuned by our data)	0.9304511 (Score)
TFBertForSequenceClassification.bert-base-uncased (finetuned by our data)	0.8728 (Accuracy)
muppet-roberta-large-finetuned (finetuned by out data)	0.9690994 (Score)

TABLE I
Grade by each model

Ranking	Team name	Grade
1	Team11	0.9690994

Fig. 1. Aldea Public Ranking

Ranking	public排名	Team name	Grade
1	1	Team11	0.9652367

Fig. 2. Aldea Private Ranking

V. Bonus

For the bonus part we use word cloud to investigate which word showed a lot of times in the negative and positive part. By doing so, we can find where we can improve our model.

First, we have to delete the word that doesn't affect the sentiment like 'movie' and 'film' to make the result more clearly.

```
stopwords.add("br")
stopwords.add("movie")
stopwords.add("film")
```

Fig. 3. deleted word example

In the train data part, the result is not clear. Most of the words are the same in the positive and negative part. The only difference between them is 'great' in the positive part and 'bad' in the negative part. The rest of words are the same.

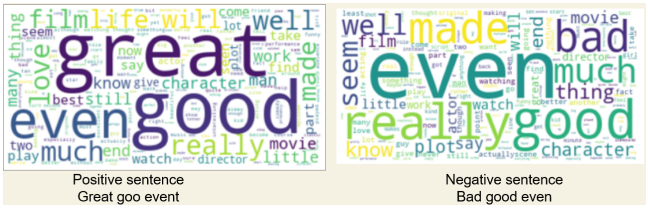

Fig. 4. result in train data

In the test data part, we don't have answer, so can we only test it with our prediction. We try our two models. But the Distilbert Model's result is also not clear enough to identify whether it is positive or negative. However, the Muppet-Roberta-Large Model's result is very clear and we can easily to identify the result. In the positive part, we have 'good', 'well' and 'great', most of the words are positive. And we also have 'good', 'bad' in the negative part, but the rest of the words are meaningless. It shows that we still have to work harder in the negative part.

There is still one question that both part have the word 'good'. We research the test data, and find the possible answer

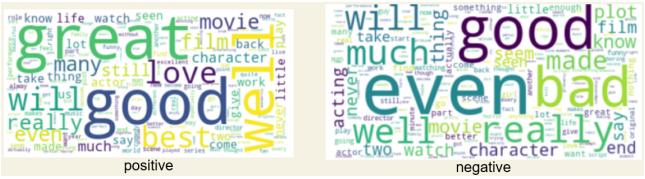

Fig. 5. result in test data

is that 'It is not good enough ' and 'I don't think it is good'. Hence, both positive and negative part have the word 'good'.

And there is also a fun fact that most people may think that the word 'not' will show in the negative part. But, there is no word 'not' is the negative part.

VI. Discussion

- We use word cloud method to improve our model because this method can check the correctness roughly but we think that this method is not a best method as the word cloud can't check the correctness of the sentence.
- The current result is close to 97%, and it is difficult to improve the accuracy. The benefit after the improvement is not so great, because there are only two emotions, positive and negative.If we want to keep doing it, we should do it with multiple emotions as the main axis.

VII. Conclusion

After we tested these four models, our finetuned Muppet model achieved a f1 score of nearly 97%, which is the highest f1 score in the class, proving that our model brings the greatest benefit to the data of this competition.

VIII. Author Contribution

Li-Xin, Ng: (16.67%) research survey, finetuned the Muppet Model, report writing

Hao-Yun, Yen: (16.67%) train and finetune the TFBert model, presentation, report writing

Chang-Chi, Wu: (16.67%) train the finetune the TFBert model, report writing

Nai-Yi, Hsu: (16.67%) bonus, presentation, report writing

Hui-Chee, Ong: (16.67%) finetune and train the Distilbert Model, report writing

Grace, Chan: (16.67%) finetune and train the Distilbert Model, report writing

References

[1] https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english

[2] https://huggingface.co/docs/transformers/custom_datasets

[3] Armen Aghajanyan et al. "Muppet: Massive Multi-task Representations with Pre-Finetuning". In:CoRRabs/2101.11038 (2021). arXiv: 2101 . 11038.URL:https://arxiv.org/abs/2101.11038

[4] A. L. Maas et al. "Learning word vectors for sentiment analysis," in *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies.* Portland, Oregon, USA: Association for Computational Linguistics, June 2011, pp. 142–150. [Online]. Available: http://www.aclweb.org/anthology/P11-1015

[5] R. Socher et al. "Recursive deep models for semantic compositionality over a sentiment treebank," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing.* Seattle, Washington, USA: Association for Computational Linguistics, Oct. 2013, pp. 1631–1642. [Online]. Available: https://aclanthology.org/D13-1170

[6] Google AI Language Team Authors and The HuggingFace (2018) PyTorch BERT model [Source code]. https://github.com/huggingface/transformers/blob/v4.15.0/src/transformers/models/bert/modeling_bert.py#L1501

[7] https://towardsdatascience.com/sentiment-analysis-in-10-minutes-with-bert-and-hugging-face-294e8a04b671