NLP 110 Final Project (Team 25) - Emotion Text Classification with CNN, BERT, RoBerta and Hugging Face Pre-Trained Models

Bo-Cyuan Lin
NYCU / IDS / 310554012
bclin1833@gmail.com

Cheng-En, Cai
NYCU / ICO / 310551068
chengencainycu@gmail.com

1 Introduction

Empathy is an important human trait that not only helps us understand each other, but is also an ability to survive in society. Since the process from understating the meaning behind the word to responding with empathy is a complex task, it is a big challenge for AI to learn the skill.

This work is trying to find the latent variables behind the sentences to predict the emotions among the conversations. We use CNN, Bert, Bert+CNN, and RoBerta as our models. First, the data will be pre-processed and transformed into new dataframe. Then the data are plugged into models to predict the label. And finally, we use confusion matrix to evaluate models' performance.

2 Related Work

Emotion detection is one task of text classification. There are multiple models had been proposed. Feed-forward networks are the simplest deep learning models for text presentations. It view the sentences as a bag of words, embedding them as vector inputs by word2vec or Glove. All these vector input passing multi-layers, and will be classified by the last layers such as Linear Regression or SVM. Rnn-based models is a short-memory model, which could memorize the previous inputs and take them into account with new input. The power of a recursive neural network is that it allows the input and output data to be not just a single set of vectors, but a sequence of multiple sets of vectors. However, Rnn-based models do not perform as well as expected in long term memory. Among many variants to RNNs, Long Short-Term Memory(LSTM) is designed to improve the shortcomings of RNNs in long term memory.

2.1 CNN

A Convolutional Neural Network (CNN) is composed of three kinds of layers: convolutional layer, Pooling layer and fully connected layer. The operations of the convolutional layer is that sliding a learn-able weight kernel on the input data to capture the spatial information of the data. And we usually implement the padding on the result of the convolution to avoid the information vanish. After the operations the output of the convolutional layer will pass through an activation function.The pooling layer is used to conduct the down sampling on the output of the convolutional layer and we usually use the max-pooling and the average-pooling.

2.2 BERT

In 2018 Google developed BERT based on bidirectional transformer. BERT is trained by 3.3 billion words, has 340 million parameters. Google pre-trained BERT to perform two tasks simultaneously. First, model should fill in the blanks with missing words.

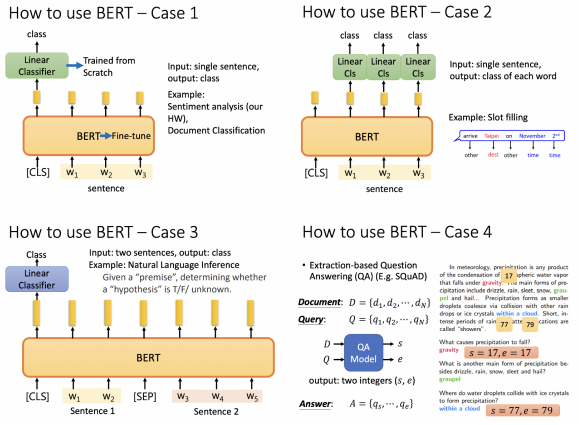

Figure 1: BERT for Different Task

The second task is to determine whether the second sentence is connected to the first sentence in the

original text(Next Sentence Prediction). There are some common applications with BERT model: (1) One-Sentence Classification (2) Token Classification (3) Two-Sentence Classification (4) Question Answering as shown in Figure 1.

2.3 RoBERTa

The full name of the RoBERTa method is Robustly Optimized BERT Pretraining Approach. The target of this paper is to optimize the training of BERT method.

3 Data

The data is split into 3 part: train data, validation data and test data. Train data is mainly used in the training phase for model fitting and directly involved in the process of model parameter tuning. Validation data is used to evaluate the initial capability of the model and basis for hyper-parameter tuning during the training process. Test data is used to evaluate the generalization ability of the model. To evaluate the true capability of the model, the test data should not be selected in training phase.

The resource of data is from Hannah Rashkin et al[?]. The following are descriptions of features.

Features	Descriptions
idx	The sample index
conv_id	Certain conversation
utterance_idx	The order in one conversation
prompt	Description the situation under the conversation
utterance	One dialogue by one of speakers
label	Emotion under the conversation

In all conversation, there are totally 32 labels(classes) and each label's count is almost 696 in data as shown in Figure 2.

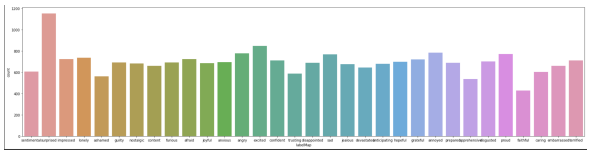

Figure 2: Counting distributions of labels in train.csv and valid.csv

4 Method

4.1 PreProcess

The prompt and utterance field contains some special patterns, like __comma__. We replace it with ",".

Before we training our model, we have tried four kinds of data conversion as input of the model, the first one is only Prompt, the second one is to string all the same conversation indexes together as input, the third one is to use every two conversations as new data. The fourth is to combine the utterance and prompt of the same conversation index together to form a new data set.

Methods	Training Accuracy	Validation Accuracy
Prompt Only	0.5831	0.5617
Utterance Only	0.5740	0.5458
Two Dialogues	0.4024	0.3854
Group Prompt and Full Conversations	**0.6287**	**0.5917**

Figure 3: Comparison for the different processing methods.

4.2 Data Augmentation

To increase the data number and make data balanced, we resample the training data. Here we set our random_state is 42, and each class number of data is 2000. To make sure the new data contains the original one, how we resample data is that creating (2000 - original_class_numbers) numbers of data. Eventually, we have 64000 numbers of data as our training set.

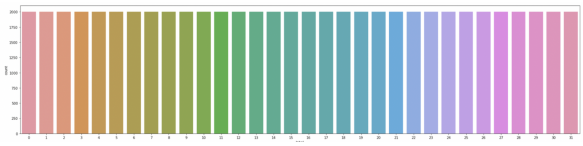

Figure 4: The structure of the CNN Text Classification

4.3 CNN

For the CNN text classifications, first, we use the embedding layer to conduct the word embedding. Second, the output of the embedding layer will pass through the bi-gram to six-gram convolutions with Relu activation function, then the output of them will be concatenated together. The fully-connected part has two dense layers and the final output will pass through the softmax.

4.4 Bert Model

To predict the emotion with entire conversation, we can view this task as a One-Sentence Classification task in BERT model. In first step, we would need to add [CLS] in the beginning of the sentence. [CLS]

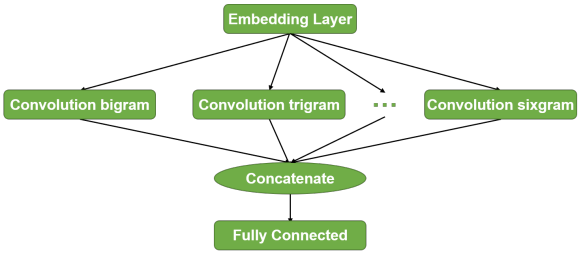

Figure 5: The structure of the CNN Text Classification

is a special token and the last hidden state of BERT corresponding to this token (h[CLS]) is used for classification tasks.

And then, BERT would use word-piece embedding input for tokens(words), use position embedding to present the information of position for each token, use segment embedding to separate the two sentence. The embedding structure is shown as Figure 3.

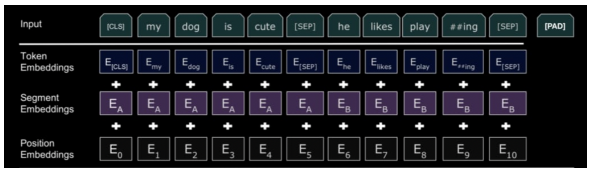

Figure 6: The embedding of BERT

As we prepared the input for the BERT model, they will pass multiple layers called self-attention shown as Figure 5. Like bidirectional RNN, it can see the information of the whole input sequence at the same time Parallelization is possible. Each embedding input would be multiplied by three different transformation matrices and produce three vectors: query, key, and value. Query and key should be dot-produced and we can get the attention scores to present how query and key are matched. And attention scores will be dot-produced again with value to extract the information. Each output of self-attention would consider each relation of the words. And at the last layer, the first output, which is (h[CLS]), would help us to classify the emotion from the entire sentence.

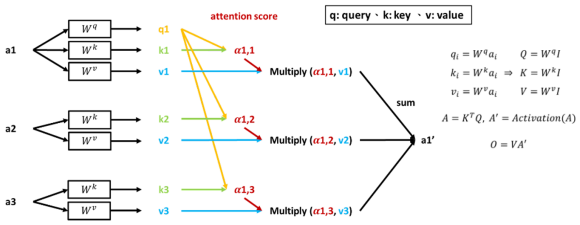

Figure 7: Structure of the self-attention

4.5 BERT(base) + CNN

For the BERT + CNN , first, we replace the embedding layer with pre-trained BERT base model. Second, the output of the embedding layer will pass through the bi-gram to four-gram convolutions with Relu activation function and the maxpooling, then the output of them will be concatenated together and pass through the density layers.

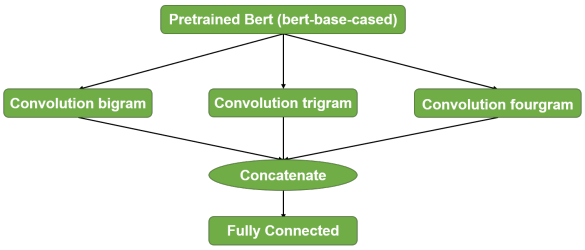

Figure 8: The structure of the BERT + CNN Text Classification

4.6 RoBERTa Model

RoBERTa has similar structure with the BERT. To improve the original BERT, RoBERTa changes the following three things: 1. Removing the Next Sentence Prediction (NSP), 2. Training with bigger batch sizes and Longer sequences, 3. Dynamics mask.

5 Experiment

5.1 CNN and BERT(base) + CNN

```
Epoch 1/5
    loss: 2.8016 - accuracy: 0.2118 - val_loss: 1.9885 - val_accuracy: 0.4119

Epoch 2/5
    loss: 1.7503 - accuracy: 0.4692 - val_loss: 1.6919 - val_accuracy: 0.4877

Epoch 3/5
    loss: 1.2905 - accuracy: 0.5868 - val_loss: 1.6734 - val_accuracy: 0.5011

Epoch 4/5
    loss: 0.8998 - accuracy: 0.7035 - val_loss: 1.7658 - val_accuracy: 0.4859

Epoch 5/5
    loss: 0.5991 - accuracy: 0.8022 - val_loss: 1.9657 - val_accuracy: 0.4870
```

Figure 9: The train information of the CNN

We train the CNN model with TensorFlow Keras on the GPU of the Google Colab with the Adam optimizer with learning rate $1e-3$ and the categorical cross entropy. And train the BERT + CNN model with Pytorch on the GPU of the Google Colab with the Adam optimizer with learning rate $2e-5$ and the cross entropy.

From the training logs above, we can find that there is the problem of the overfitting for the CNN and BERT(base) + CNN method. Both of their

Figure 10: The train information of the BERT + CNN

training accuracy can achieve over 0.8, but the validation accuracy of them is still under the 0.5.

5.2 BERT

To predict the emotion by BERT model, we use pre-train model from ktrain, which is a lightweight wrapper for the deep learning library TensorFlow Keras. In the experiment, the main structure, shown as Figure 9, includes embeddings layers, self-attention layers, fully connected layer, and so on. The number of total parameters are 109,506,848. To train these model, the learning-rate we set is 2e-5. The optimizer is Adam. the loss function is categorical_crossentropy.

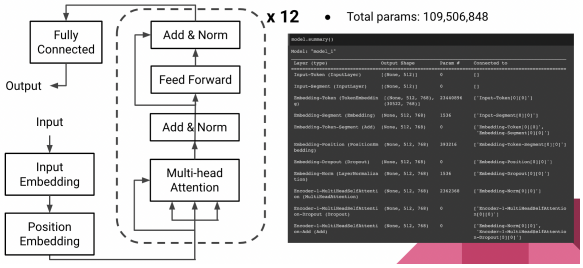

Figure 11: The struct of BERT model

After finishing, we use the model to valid.csv to validate our model. The figure 10 is the result. And the figure 11 is our result from kaggle competition.

accuracy			0.62	2770
macro avg	0.61	0.61	0.60	2770
weighted avg	0.62	0.62	0.61	2770

Figure 12: The validation score of BERT model

Figure 13: The test score of BERT model

5.3 RoBERTa

We use the TFAutoModelForSequenceClassification function of tensorflow to obtain the pre-trained

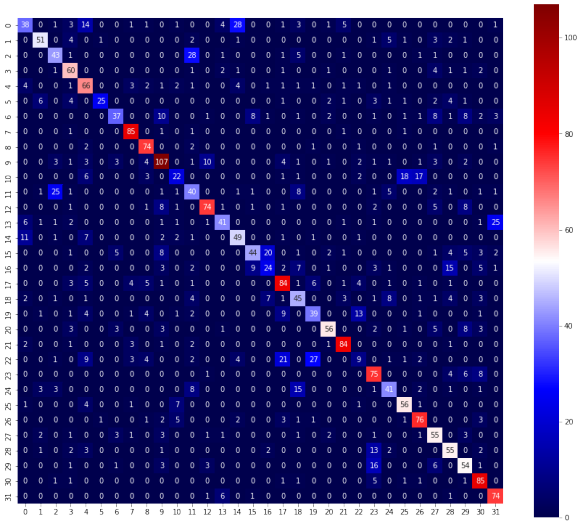

Figure 14: The confusion matrix of the RoBERTa

model of RoBERTa-base. And train the model on the TPU of the Google Colab with the Adam optimizer with learning rate $1e-5$ and the sparse categorical cross entropy. Figure 10 and Figure 11 are the results of the RoBERTa base model for the emotional text classification. The figure 12 and figure 13 is the result of the RoBERTa, it achieve similar accuracy with the BERT(ktrain) and it just cost 1.5 minutes per epoch.

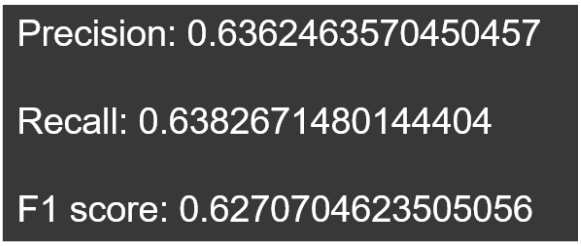

Figure 15: The Precission, Recall and F1 score of the RoBERTa

5.4 Submission

When we adjust the hyper-parameter(the epoch number, learning rate, amounts of re-samples) and get the best accuracy of valid data, we would append the valid data as part of the training data and re-train the model. The amount of the training data would increase about 15

6 Try Other ML methods and Pre-Trained Models in Hugging Face

In addition to the base BERT and base RoBERTa pre-trained models, we also try other ML methods and pre-trained models in the hugging face/text

classification, total 33 kinds. And the performance of using the pre-trained model Nakul24/RoBERTa-emotion-classification is the best.

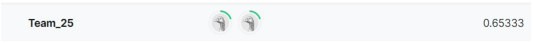

Figure 16: Prediction of using Pre-Trained RoBERTa: Nakul24/RoBERTa-emotion-classification

Model Name	Performace (Valid acc)	Run On
cardiffnlp/twitter-roberta-base-sentiment	X (58 <val acc < 61)	TPU (Colab)
finiteautomata/bertweet-base-emotion-analysis	X (58 <val acc < 61)	TPU (Kaggle)
bhadresh-savani/bert-base-go-emotion	X (55 <val acc < 61)	TPU (Kaggle)
H2O Package - AutoML	X (val acc < 55)	GPU (Colab)
RoBERTa with 1 v.s. ALL (Binary Tree Structure)	X (val acc < 55)	TPU (Colab)
CNN with 1 v.s. ALL (Binary Tree Structure)	X (val acc < 55)	GPU (Colab)
RoBERTa with 1 v.s. ALL	X (val acc < 55)	TPU (Colab)
joeddav/distilbert-base-uncased-go-emotions-student	X (55 <val acc < 61)	TPU (Kaggle)
distilbert-base-uncased-finetuned-sst-2-english	X (55 <val acc < 61)	TPU (Colab)
cross-encoder/ms-marco-MiniLM-L-12-v2	X (val acc < 55)	TPU (Kaggle)
ProsusAI/finbert	X (55 <val acc < 61)	TPU (Kaggle)
roberta-large-mnli	X (val acc < 55)	TPU (Kaggle)
tals/albert-xlarge-vitaminc-mnli	X (val acc < 55)	TPU (Kaggle)

Figure 17: Tried Pre-Trained Models or Methods (1/3)

Model Name	Performace (Valid acc)	Run On
avichr/heBERT_sentiment_analysis	X (val acc < 55)	TPU (Kaggle)
bhadresh-savani/distilbert-base-uncased-emotion	X (val acc < 55)	TPU (Kaggle)
j-hartmann/emotion-english-distilroberta-base	X (57 <val acc < 59)	TPU (Kaggle)
mmillet/rubert-tiny2_finetuned_emotion_experiment_augmented_anger_fear_no_emojis	X (val acc < 55)	TPU (Kaggle)
vortixhead/distilbert-base-uncased-finetuned-emotion	X (val acc < 55)	TPU (Kaggle)
j-hartmann/emotion-english-roberta-large	X (58 <val acc < 63)	TPU (Kaggle)
Nakul24/RoBERTa-Goemotions-6	X (58 <val acc < 61)	TPU (Kaggle)
Jorgeutd/sagemaker-roberta-base-emotion	X (57 <val acc < 59)	TPU (Kaggle)
aarnphm/finetune_emotion_distilroberta	X (58 <val acc < 59)	TPU (Kaggle)
Nakul24/RoBERTa-emotion-classification	X (60 <val acc < 65)	TPU (Kaggle)
BERT + KNN	X very poor	CPU
cardiffnlp/twitter-roberta-base-sentiment-latest	X (60 <val acc < 63)	TPU (Kaggle)

Figure 18: Tried Pre-Trained Models or Methods (2/3)

Model Name	Performace (Valid acc)	Run On
anvay/finetuning-cardiffnlp-sentiment-model	X (60 <val acc < 63)	TPU (Kaggle)
bert-base-multilingual-cased	X (val acc < 55)	TPU (Kaggle)
microsoft/codebert-base	X (val acc < 55)	TPU (Kaggle)
arpanghoshal/EmoRoBERTa	X (58 <val acc < 61)	TPU (Kaggle)
facebook/dpr-question_encoder-single-nq-base	X (58 <val acc < 61)	TPU (Kaggle)
wietsedv/bert-base-dutch-cased-finetuned-sentiment	X (val acc < 55)	TPU (Kaggle)
vinai/bertweet-base	X (60 <val acc < 63)	TPU (Kaggle)
cardiffnlp/bertweet-base-sentiment	X (60 <val acc < 63)	TPU (Kaggle)

Figure 19: Tried Pre-Trained Models or Methods (3/3)

7 Advanced

For the advanced competition, we use the extra data (https://huggingface.co/datasets/emotion) which contains sad, joy, love, anger, fear, and surprise emotions. In the begging, to train the model with two datasets, we mapped the new label into the original label. And we also remove ”love” label since there is no label corresponding to the original data. In this data, there are totally 5 labels(classes) shown as Figure 14.

We use three ways to make the usage of the extra data. The first is appending the extra data into the origin training data. The second is warming up the model first, then train the model with the origin training data. The Third way is to conduct the data argument with the new data. The results show that the third method is the best. The first method has serious over fitting And the performance of second and third methods are similar with using only using the origin data, too. We reckon the reason is that the extra data only contains six emotions which may break the balance of the origin data.

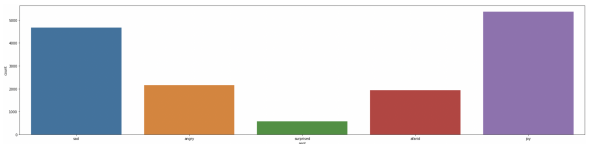

Figure 20: Counting distributions extra data from huggingface

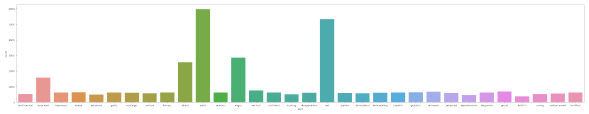

Figure 21: Appending the extra data from huggingface to the origin data may break the balance

Figure 22: Score of RoBERTa with extra data and data argumentation.

8 Conclusion

For this work, we find that the Bert and the RoBerta methods have better performance than the CNN methods. And the overfitting is the main problem for those models due to the data is not enough. With multiple tests, the data in the form of prompt with one line sentence of utterance has better performance. And we found that if we don’t consider the prompt information the result will be poor. For this project we had learnt how to use the multiple tools (Pytorch, Tensorflow with Keras, H2O, Scikit-Learn) with platforms (GPU/ TPU of Google Colab, GPU/ TPU of Kaggle) to conduct the text classification.

9 Work Division

We have a meeting every week, discussed and implemented each methods together.

Items	Cheng-en, Cai	Bo-Cyuan Lin
Data Processing	V	V
CNN	V	V
BERT	V	V
BERT(base) + CNN	V	V
RoBERTa	V	V
Other Pre-Trained/Method	V	V
Advanced	V	V
Slide & Report	V	V

Figure 23: Work Division

10 Question and Answer

Teacher: Have you tried anything to reduce over fitting?

Reply: We reduce the number of the dense layers which conducting the classification. (3 dense layer: number of parameters $>$ 300 million (origin amount $<$ 150 million), Reduce to 1 Dense layer: keep origin number of the parameters.)

11 Reference

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Yinhan Liu , MyleOtt, NamanGoyal, JingfeiDu, MandarJoshi, DanqiChen, OmerLevy, MikeLewis, LukeZettlemoyer, and VeselinStoyanov. 2019b. RoBERTa: A robustly optimized BERT pretraining approach. arxiv preprint arXiv:1907.11692.

https://www.youtube.com/watch?v=gh0hewYkjgo&ab_channel=Hung-yiLee

https://github.com/Rayryu/CNN-based-sentimental-analysis-multiclass-classification

https://blog.csdn.net/lch551218/article/details/117249357

https://colab.research.google.com/github/dipanjanS/nlp_workshop_odsc19/blob/master/Module05%20-%20NLP%20Applications/Project07B%20-%20Text%20Classification%20Deep%20Learning%20CNN%20Models.ipynb

https://huggingface.co/datasets/emotion