NLP 110 Final Project - Team 10
Deep Neural-based Models for Emotion Detection in Online Chat Text

Kuan-Chieh Wu, Min-Yun Hsieh
National Yang Ming Chiao Tung University, Hsinchu, Taiwan
kuanchieh.cs09, minyunh.cs10@nycu.edu.tw

1 Introduction

Sentiment analysis refers to determining the nature of a text. Its main purpose is to identify the speaker's attitude on certain topics or polarized views of a text. Traditional studies on sentiment analysis usually aim to detect the polarity (positive, negative, or neutral) of a given text. Other kinds of studies exist that recognize multiple differentiated affective manifestations in texts, such as joy, anger, and fear, and explore sentences with compound emotions as well as the uses of language in expressing complex feelings.

Usually in a conversation, we can judge emotions from a person's tone of voice and expressions. However, with the advancement of technology, people can communicate through different forms of media, among which text messaging has gradually become the most mainstream form of communication, but when communicating only through text messages, it is difficult for the receiver to know the emotions of the speaker from the words alone.

In the past few years, many researchers established numerous methodologies to extract emotion from text however, surprisingly enough, a little effort is initiated in real-time messenger which will have the ability to analyze one's emotional state. There are many models that have successfully extracted emotions for our reference. For example word2vec, GloVe, CNN, LSTM, BERT, and so on.

Sentiment analysis systems have been applied to different application domains and on both long and short texts(conversation). Automatically recognizing a user's affective states can enhance the quality of interactions.

We participate in the Emotion Detection in Online Chat Text challenge, whose goal is to predict the emotion of the conversation. The challenge aims to predict the emotion label of the conversation by taking its conversational thread into consideration, including all textual and situations. We implement some popular deep neural-based models as our emotion classifiers, including LSTM and BERT-based models.

To analyze how each component of our training process influences the results, we construct several experiments during the training and practice phase to see the Score on the development set. We compare the results between training our models with different models and training strategies. Besides, we also compare the results between training with different parameters.

2 Related Work

As mentioned above, various methods have been proposed for sentiment analysis. These can be categorized into different types, depending on text level (including feature, sentence, or document), text length(long or short), and text type (formal or informal). In this section, we briefly review the word embedding, data preprocessing, and sequence that we've implemented in this challenge, and we also investigate some studies of emotion detection.

2.1 Word Embedding

Pre-trained words are very important and serve as a key factor in neural language understanding, its core concept is to convert text into numerical form. In the training of the neural network-like model, we have no way to directly bring the text into it for operation. Because the basis of the neural network is based on the output of neurons through weight operation.

Mikolov et al. (2013) invented a tool word2vec for word embedding, which trains vector space models faster than previous methods. Many emerging word embeddings are based on artificial neural networks rather than n-gram models and unsupervised learning of the past. Word2vec has achieved good results at the word level, but if you add up

the vectors of each word in a document and try to get the vector of this article, there will still be a problem of losing the order.

In the work of Pennington et al. (2014), 300-dimensional vectors can be used to represent 2.2 million words, which can effectively solve the above-mentioned dimensional explosion problem and save a lot of computing and storage costs. They also introduced a new global log-bilinear regression model for the unsupervised learning of word representations that outperforms other models on word analogy, word similarity, and named entity recognition tasks.

2.2 Squence Model

Dealing with sequential data is crucial in the field of natural language processing. Lots of works tackle this problem by utilizing various kinds of deep learning frameworks. Zhang (2021) proposes a Long-ShortTerm-Memory (LSTM) model for the text classification task. The model transforms text into low-dimensional word vectors by word embedding and uses CNN to extract the local features of the text. Combined with LSTM's characteristics of preserving historical information in text sequences, it makes up for CNN's deficiency in extracting context-related semantics.

However, these RNN-based models still have limited ability to capture the relations between long-distance words in inputs with very long sequences. Besides, the RNN-based model takes a large amount of time to train since it requires the inputs to be processed in sequential order, on the other hand, the attention mechanism allows the model to be trained in parallel, and obtain outputs after looking at the whole input sequences.

Therefore, the development of state of the art pre-trained systems such as BERT (Devlin et al., 2018) is proposed. BERT is conceptually simple and empirically powerful. It obtains new state-of-the-art results on eleven natural language processing tasks.

2.3 Emotion Detection

Several research teams have been endeavored to detect emotion in online chat text. For instance, Dahiya et al. (2020) build a chat analyzer and successfully applied on WhatsApp Chats, this model classifies text into one of 6 emotions while taking into consideration the emojis used by the person. Basu et al. (2017) combine deep Convolutional Neural Network (CNN) and Long-ShortTerm-Memory (LSTM) model, proposing a LSTM-base model to recognize emotions. In addition, Kodiyala and Mercer (2021) utilize a BERT to classify the emotion from different posts on Twitter and achieve competitive result comparing to LSTM.

Dataset	Number of conversations
Training set	19533
Validation set	2770
Testing set	2547

Table 1: Number of conversations in each dataset.

3 Dataset

The dataset used in this project is empathetic_dialogues (Rashkin et al., 2018), presented in Towards Empathetic Open-domain Conversation Models: a New Benchmark and Dataset. The dataset is built by thousands of conversations. Every conversation has a speaker and a listener, one conversation contains about 3 to 6 utterances, and every conversation has a prompt and an emotion label.

In this dataset, a row of data is separated by utterance. Every data contains id, conv_id, utterance_idx, prompt, utterance, and label. Identical conv_id indicate conversations from the same group, utterance_idx indicates the number of utterances in a certain round of conversation, prompt is the description of the speaker's situation and the label is the emotional label of a whole group of conversation. There have 32 emotion label, the following dictionary is used to map emotion class to its label. *'sad': 0, 'trusting': 1, 'terrified': 2, 'caring': 3, 'disappointed': 4, 'faithful': 5, 'joyful': 6, 'jealous': 7, 'disgusted': 8, 'surprised': 9, 'ashamed': 10, 'afraid': 11, 'impressed': 12, 'sentimental': 13, 'devastated': 14, 'excited': 15, 'anticipating': 16, 'annoyed': 17, 'anxious': 18, 'furious': 19, 'content': 20, 'lonely': 21, 'angry': 22, 'confident': 23, 'apprehensive': 24, 'guilty': 25, 'embarrassed': 26, 'grateful': 27, 'hopeful': 28, 'proud': 29, 'prepared': 30, 'nostalgic': 31*

The dataset is divided into training set, validation set, and testing set. We calculated the number of conversations in each dataset, the results are shown in Table 1.

We also count the number of labels in training set, the results are shown in Table 2.

Emotion	Count	Emotion	Count
0	667	16	606
1	515	17	667
2	629	18	614
3	527	19	615
4	598	20	576
5	377	21	641
6	602	22	695
7	585	23	616
8	619	24	462
9	1004	25	622
10	493	26	563
11	634	27	645
12	621	28	619
13	525	29	686
14	569	30	592
15	750	31	599

Table 2: Number of labels of a group of conversation in training set.

4 Method

The design of the system can be divided into three parts: data preprocessing, word embedding, classification models. The detailed work of each part is shown in the following subsections. Additionlly, we discuss word embedding and classification models with two categories below:

- RNN-based model: LSTM
- BERT-based models: BERT-cased, BERT-uncased

4.1 Data Preprocessing

First, since the data is divided by different utterances, we concatenate the conversations sequentially. By doing this, we will get a full conversation, prompt and the emotion label of the conversation. Then utilize a text preprocessing package, text-hammer, to process text in prompt and utterances:

- Convert all the letters into lowercase.
- Change the personal pronoun + be abbreviation back to the original.
- Remove emails.
- Remove HTML tags.
- Remove special characters.
- Remove accented characters.
- Convert into root words.

By counting the number of labels (see Table 2), we can see the data is imbalanced. The minimum number of labels is only 337 for emotion 5 (faithful), and the maximum is 1004 for emotion 9 (ashamed), a difference of about 3 times. We originally wanted to down-sampling the data to the same sample rate, but considering that there is not a lot of data for each label. Take into account that if the data is reduced, it will cause important emotional words to be removed from the data, we decided not to down-sampling the data and maintain the original number of data.

4.2 Word Embedding

In this task, we implement two kind of model. For RNN-based model, the embedding are initialized with weights using pre-trained GloVe, a model with 6B tokens and 300 dimension word vectors.

4.3 Models

The goal of this challenge is to detection the emotion in conversations. Since we have 32 type of emotions in this challenge, the task is modeled as a multi-label classification problem. To tackle this task, we implement two models, including LSTM and BERT.

For RNN-based model, we use cross entropy as the loss function, Adam as optimizer, and accuracy as evaluation metrics.

For BERT-based model, the model framework includes two phases, enhancing the pre-trained language model and fine-tuning the classification model. For BERT-cased and BERT-uncased, to be in line with the input format, we add the special token [CLS] at the beginning of the sentence and add the special token [SEP] both between the prompt and utterances and at the end.

5 Experiment

In this section, we present the detailed experiment settings used for constructing our models, and show the experiment results and competition results of challenge.

5.1 Implementation Details

During the training phase, we train the model on the training set and observe the testing results on the validation set.

Hyper-parameters	LSTM
Batch size	256
Optimizer	Adam
Loss	cross entropy
Dropout	0.2
Epoch	25
Hidden dimension	256
Number of layers	3
EarlyStopping	val loss

Table 3: Hyper-parameters of RNN-based model.

Hyper-parameters	BERT
Batch size	16
Optimizer	Adam
Learning rate	1e-5
Decay	1e-6
Loss	cross entropy
Dropout	0.2
Epoch	3
Hidden layer	1

Table 4: Hyper-parameters of BERT-based model.

RNN-based model The experiment details are shown in Table 3. Additionally, the max sequence of input sentences are set to 300, and the embedding layer have been fine-tuned during training phase as well. We only take prompts in dataset as the input of LSTM model.

BERT-based model For BERT, because the GPU memory is limitation, we set the maximum sequence as 256 and if the sequence length exceed 256, the second sequence will be truncate. The detail of hyper-parameters are list in Table 4

5.2 Experiment results

In this task, we will analyze the validation accuracy of each epoch at training time for different model. Test scores from the kaggle challenge.

We tried medium-based model and hard based model, the best result of each model is shown in Table 5. The details of each model's experiment will be introduced later.

For LSTM, we can see the results shown in Figure 1, because we set the epoch to 25, we use early stop to prevent overfitting. Training stop in epoch 12, training accuracy reach nearly 0.7, while the validation accuracy reach about 0.45.

Result	Medium	Hard
Model	LSTM	BERT
Epoch	12	3
Val_acc	0.4560	0.7926
Score	0.59719	0.59719

Table 5: Best experiment results of LSTM and BERT model.

Input Feature	Test score
Prompt only (Fig 2(a))	0.55035
Prompt + utterance (Fig 2(b))	0.54078
Prompt + conversation (Fig 2(c))	0.59464

Table 6: Results of different input for BERT-model.

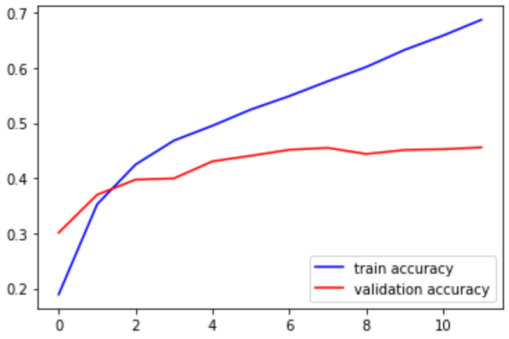
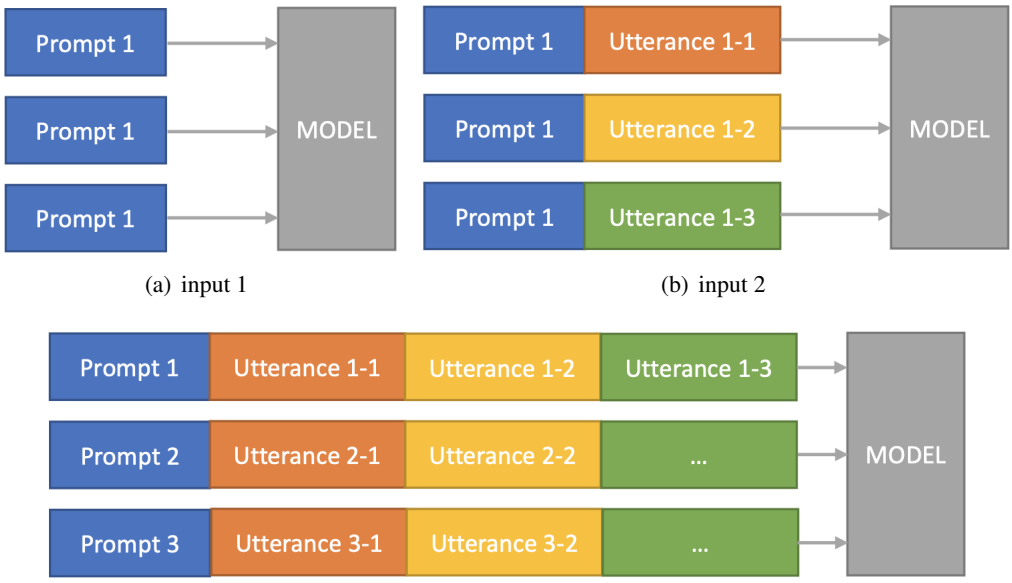


Figure 1: LSTM training result.

For BERT, in the beginning, we tried different input for the same model, the model used is BERT-cased with one hidden layer, batch size 16, learning 1e-5, decay 1e-6 and epoch 5, the details of each input is shown in Figure 2. And the results of different input are shown in Table 6. We can see the best input is prompt with conversation, we will adjust the parameters of the model for this input.

We tried BERT-cased and BERT-uncased, different hidden layer and different parameters, the results is shown in Table 7. We have tried many combinations of parameters, but we only pick a few and put them in the table for discussion. In the experiments of adjusting the parameters, we set the epoch to 5, but we observe the validation accuracy of each epoch in each experiment, the validation accuracy is best when it is about epoch 3 or 4.

Under this observation, we decided to increase the epoch to 10, and after running epoch 10 with both BERT-cased and BERT-uncased , since the score of these two model is close. We set the parameter according to the results in Table 7 with one hidden layer, learning rate 1e-5 and decay 1e-6. And we will determine the number of epochs according to the validation accuracy result of this



(a) input 1

(b) input 2

(c) input 3

Figure 2: Input used for BERT model.

time. In this experiment, we see that validation accuracy is the best when using BERT-Uncased and epoch is set to 3. Here we make two attempts. One is to directly adjust the epoch to 3 according to the original training data. The other is to add validation data to the training data. In the experiment of adding validation data into training data, we got the best results, which can be seen in Table 8. It achieves 0.59719 on score.

6 Conclusion

In this paper, we present the details and experiment results to accomplish the Emotion Detection in Online Chat Text challenge. We implement different model (LSTM and BERT) and training strategies to carry out the experiments. In the experiment results we can see BERT has displayed its great advantage of text representation. In this challenge, both LSTM model and BERT model we implemented pass the baseline, and in the end of the challenge, we achieves 0.59719 on challenge score, which gets 29th place in the challenge.

We believe that the set maximum sentence length limits the performance of our model to a certain extent. If the complete conversation can be used to train the model, the model can get more emotional information.

7 Work Division

Kuan-Chieh Wu Processing data, Implement and perform experiments on LSTM and BERT(cased, uncased).
Min-Yun Hsieh Perform experiments on BERT(cased, uncased).

8 Question and Answer

Q1. Is the reason for choosing to implement early stopping to prevent overfitting?

A1. Yes, in LSTM part we set early stopping to prevent overfitting.

Q2. Do you have any special adjustments to the dropout ratio?

A2. No, we only try dropout ratio in 0.2.

Q3. If you set max_length to 256, how do you deal with the extra length?

A3. We just cut off the words that exceed the length.

Q4. Is the embedding weight of LSTM randomly initialized? Or are there pretrained embedding weights such as GloVe or fasttext?

A4. We initialized the weight using pre-trained GloVe.

Q5. Have you tried using the AdamW optimizer?

Model	Hidden layer	Learing Rate	Decay	Epoch	Val_acc	Score
BERT-cased	1	1e-5	1e-6	5	0.6102	0.59619
	2	2e-5	2e-6	5	0.6022	0.58288
BERT-uncased	1	1e-5	1e-6	5	0.6113	0.59615
	2	2e-5	2e-6	5	0.6030	0.59373

Table 7: Experiment result of BERT model with different parameters

BEST	BERT-Uncased
Learning Rate	1e-5
Decay	1e-6
Epoch	3
Score	0.59719

Table 8: BEST Experiment result in challenge

A5. No, we only use Adam.

Q6. Why the results of val acc and test are somewhat different?

A6. We add validation data into training data in our best result, but we forgot to remove validation data when training model, thus the validation accuracy and test are different. In this case, the validation accuracy can be ignore. As for the adjust parameters table is we put the wrong number which has been corrected in this report.

Q7. Have you tried steplr?

A7. No.

Q8. Why the valid score is 0.7 but the score is only more than 0.5?

A8. The answer for Q8 is the same as for Q6.

Q9. Have you ever used Bert + LSTM?

A9. No, we haven't tried to combine BERT and LSTM.

Q10. Have you tried the ensemble method?

A10. No, we haven't try the ensemble method.

References

Saikat Basu, Jaybrata Chakraborty, and Md. Aftabuddin. 2017. Emotion recognition from speech using convolutional neural network with recurrent neural network architecture. In *2017 2nd International Conference on Communication and Electronics Systems (ICCES)*, pages 333–336.

Sonika Dahiya, Astha Mohta, and Atishay Jain. 2020. Text classification based behavioural analysis of whatsapp chats. In *2020 5th International Conference on Communication and Electronics Systems (ICCES)*, pages 717–724.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Vishwa Sai Kodiyala and Robert E. Mercer. 2021. Emotion recognition and sentiment classification using bert with data augmentation and emotion lexicon enrichment. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 191–198.

Tomas Mikolov, Kai Chen, G.s Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Proceedings of Workshop at ICLR*, 2013.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. I know the feeling: Learning to converse with empathy. *CoRR*, abs/1811.00207.

Yanbo Zhang. 2021. Research on text classification method based on lstm neural network model. In *2021 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, pages 1019–1022.