Sentiment Analysis on Amazon Review

Final Project Group 10

Yongheng Zhang, Chenhao Li, Qihao Huang,

Lixiong Feng, Zihua Zhang

1. Abstract

In this project, we built and trained two different neural networks that predict the sentiment of sentences. There are several past papers succeeding in analyzing the sentiment of sentences. Our works are inspired by their works, but we built our own neural networks based on their basic ideas.

2. Introduction

We deem this subject as important and relevant because many industries--food, online shopping, movie review--need a fast and accurate way to determine the sentiment of a large number of comments and feedbacks, so that service can be improved. By using sentiment labeled data from Amazon comments, we were able to train a network that is able to predict the sentiment of an English sentence as either positive or negative.

CNN is a very popular network widely used for image classification and recognition, for it is good at extracting relevant information. The state of art also uses CNN as the architecture and ends up with high accuracy. Although CNN requires fixed length input, we can use zero padding to unified the input sentence. By Karakus, CNN gives a preferment that is close to RNN. We did two CNN model, one convolutional layer and two convolutional layers. Because the hyperparameters can affect the preferment of CNN greatly, we also try different hyperparameters, like filter number, kernel size, and batch size.

In dealing with a sequence of information, like speech, sentence, recurrent neural networks will also do a good job. In the work of Graves, Mohamed, and Hinton in “Speech recognition with deep recurrent neural networks”, they found that bidirectional Long Short-term Memory RNNs work very well in speech recognition. We view that it will also work in sentences analysis since we need to recognize both previous information (words) and those after so that we will know whether the sentence appears positively or negatively. Therefore, we built both single LSTM and bidirectional LSTM to see the different performance of them on analyzing the sentiment of sentences.

3.Data

3.1 Data Collection

We got our data from UCI Machine Learning Repository. The name of the dataset is *Sentiment Labelled Sentences Data Set*. The data has already been separated in sentences with number 0 and 1 at the end of each sentence, which indicates the sentences are negative or positive.

There are overall 1000 sentences for Amazon reviews with 500 sentences are negative and 500 positive sentences.

3.2 Data Preprocessing

Since the data we got are all strings, we need to convert them into digits so that we can put them into our neural networks. There have been several ways we can use to convert our data. The first method we used is very simple. The method is implemented by embedding function in Keras. We first lowered all words and assigned tokens for each of them. We then dropped all punctuations since we want to try some easy ways first, and this is how embedding function was built in Keras. However, each sentence may not have the same length. Therefore, we found the maximum length of all sentences and padded the rest of them with 0. In this case, we have 2815 vocabularies in the end.

Another method we used after that is Word2Vec, which will also convert words into vectors, but for those appear close to each other, they will have close vectors in distance. In this case, we leave ".", "!", and "?" since we thought these punctuations influence the sentiment of a sentence, and they are necessary to be used in the analysis. Furthermore, we drop those words appear less than 5 times. Then we ended up with 340 vocabularies.

4.Neural Networks

4.1 Convolutional Neural Networks

The architecture of our first CNN model is word embedding layer, converting input words to 100 long vectors, followed by a dropout layer with a 0.3 dropout rate. One convolutional layer with 128 filters and kernel size of 3, followed with Max pooling. Then the denser layer with 128 units, and one more dropout layer with a 0.3 rate. The last is the output layer. We using ReLu for hidden layer and sigmoid for the output layer, and using MSE as an error function.
The second CNN model we use adds one more convolutional layer, with 128 units and kernel size of 3, to our first model and other parameters remain the same. Architecture is shown in Appendix A.

4.2 Recurrent Neural Networks

First, we tried on a single LSTM neural network. It has one layer with embedding function which converts words into 100 units long vectors. Following that is a dropout layer with 10 percent dropout rate. Then, it has an LSTM layer with an output dimension of 15. After that is another dropout layer with dropout rate 10 percent. The last layer is a dense function with an output dimension of 1. The activation function here is sigmoid, and the loss function we use is binary cross entropy. We use Adam as our optimizer. We use 20 percent of data as our validation set. The batch size is 30. We trained it in 20 epochs.

Then, we built another one with bidirectional LSTM RNN that has the first layer with embedding function which converts each word into 20 units long vector. Then, the next layer is the bidirectional LSTM with each direction having an output dimension of 30. Following a dropout layer with a dropout rate of 10 percent. The last layer is a dense layer with an output dimension of 20 by using softmax activation. The loss function is calculated by sparse categorical cross entropy. We use Adam as our optimizer. We use 20 percent of data as our validation set. The batch size is 30. We trained it in 20 epochs. Both architectures are shown in appendix A.

5.Result

5.1 Convolutional Neural Networks

The test accuracy of one convolutional layer model after 10 epochs is around 83% and the two-layer model gives the test accuracy around 80%, but training accuracy is both near 100%, which in overfitting, shown in Appendix B.

Then we change some hyperparameters of one convolutional layer CNN and see how accuracy change. The first thing we modify is the filter number, and the test accuracy decreasing as the filter number decrease. For batch size, it does not affect the accuracy. And the kernel size does affect accuracy either. Changing those three hyperparameters does not affect the training accuracy, which remains overfitting.

5.2 Recurrent Neural Networks

The results of single LSTM with using the naive word embedding method and Word2Vec are shown in Appendix B. It is not so efficient at the beginning of training that losses of both test and training data do not decrease too many, while the accuracy does not change and is low at the start of training. Both methods end with around 80% test accuracy.

Figure 10 and 11, in Appendix B, show the result of using bidirectional LSTM with Word2Vec. It gets trained in each epoch, but it seems not good enough since test accuracy varies a lot in each epoch. Still, the test accuracy remains at nearly 80% while the training accuracy is going up to 100%. Overfitting problem is still kind of existing after 10th epoch.

6.Conclusion and Discussion

After using the convolutional neural networks and the recurrent neural networks, our accuracy is approximately 83%. Both single LSTM and bidirectional LSTM end up with similar accuracy, while their performances are different. Bidirectional LSTM truly gets to work on training start at the beginning while single LSTM works well only after several epochs. We found out that

Word2Vec method could reduce the overfitting issue. However, overfitting still exists. In order to solve this problem, we could try to use a more complex model and see if this approach helps. It is also worth it to try to change the number of hidden layers, the activation functions, the number of epochs, and the number of neurons. Moreover, we think it is necessary to find more data for training purpose while by using Word2Vec, we drop out a lot of words, and maybe there are some words are important in detecting the sentiment of each sentence.

7.Reference

A. Graves, A. Mohamed and G. Hinton, "Speech recognition with deep recurrent neural networks," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, 2013, pp. 6645-6649. doi: 10.1109/ICASSP.2013.6638947

Britz, Denny. “Understanding Convolutional Neural Networks for NLP.” WildML, 10 Jan. 2016, www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/.

Karakuş, Betül Ay, et al. “Evaluating Deep Learning Models for Sentiment Classification.” *Concurrency and Computation: Practice and Experience*, vol. 30, no. 21, 2018

Kotzias, Dimitrios, et al. "From group to individual labels using deep features." *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015.

M. Denil, A. Demiraj, and N. de Freitas. Extraction of salient sentences from labelled documents. Technical report, University of Oxford, 2014.

Appendix A

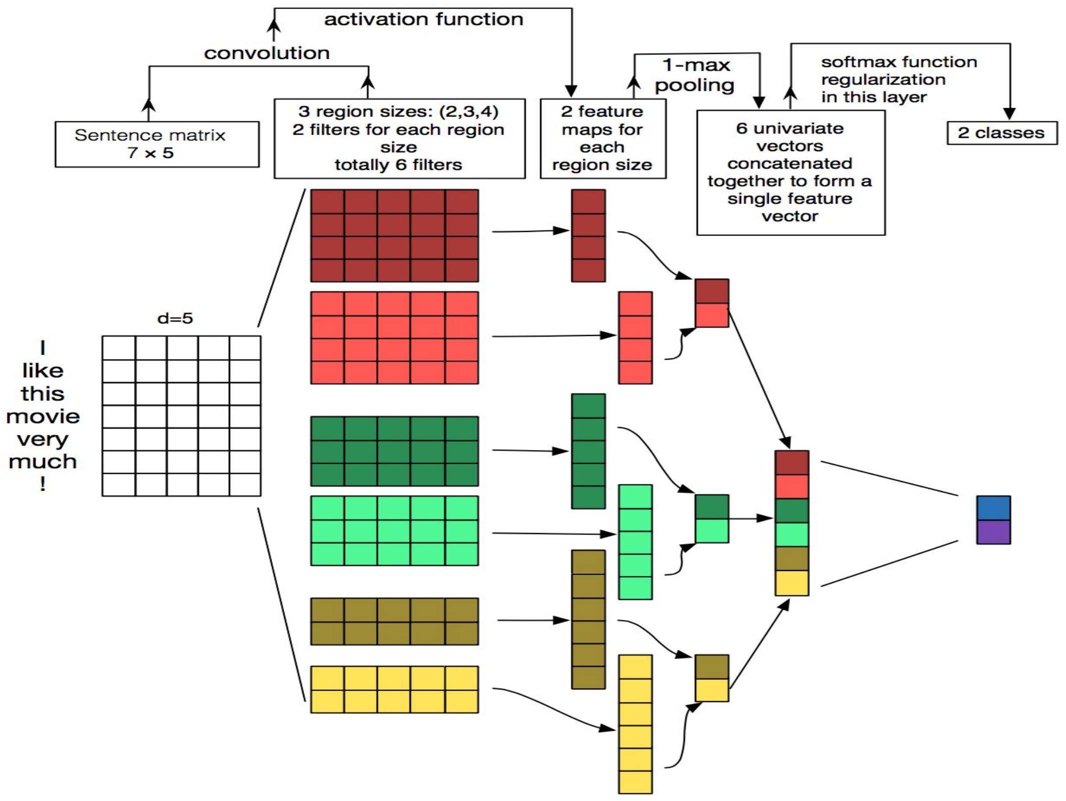
activation function
convolution
1-max pooling
softmax function regularization in this layer
Sentence matrix 7 x 5
3 region sizes: (2,3,4) 2 filters for each region size totally 6 filters
2 feature maps for each region size
6 univariate vectors concatenated together to form a single feature vector
2 classes
d=5
I
like
this
movie
very
much
!

Fig.1 Convolutional neural network Architecture (Britz)

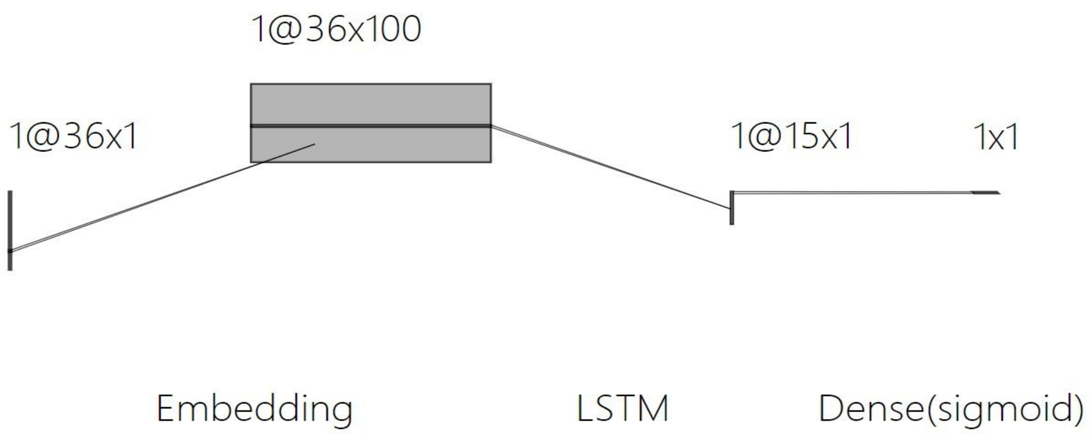

Fig.2 Single LSTM Architecture

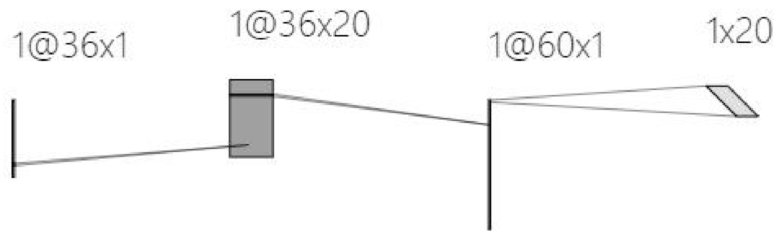

Fig.3 Bidirectional LSTM Architecture

Appendix B

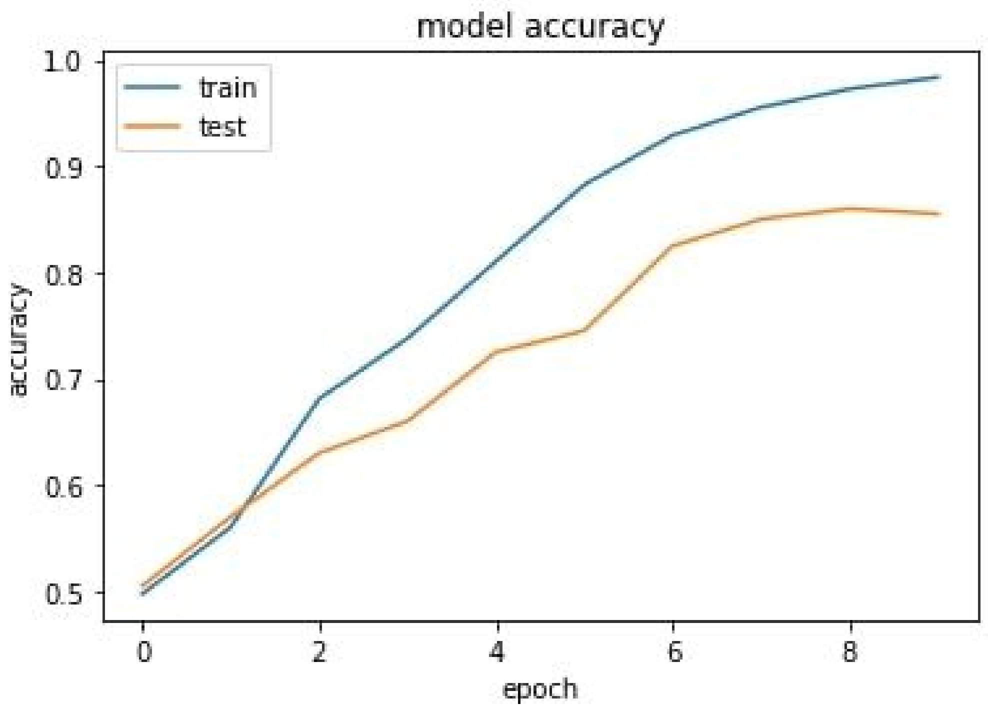

Fig.4 Accuracy of one convolutional layer CNN

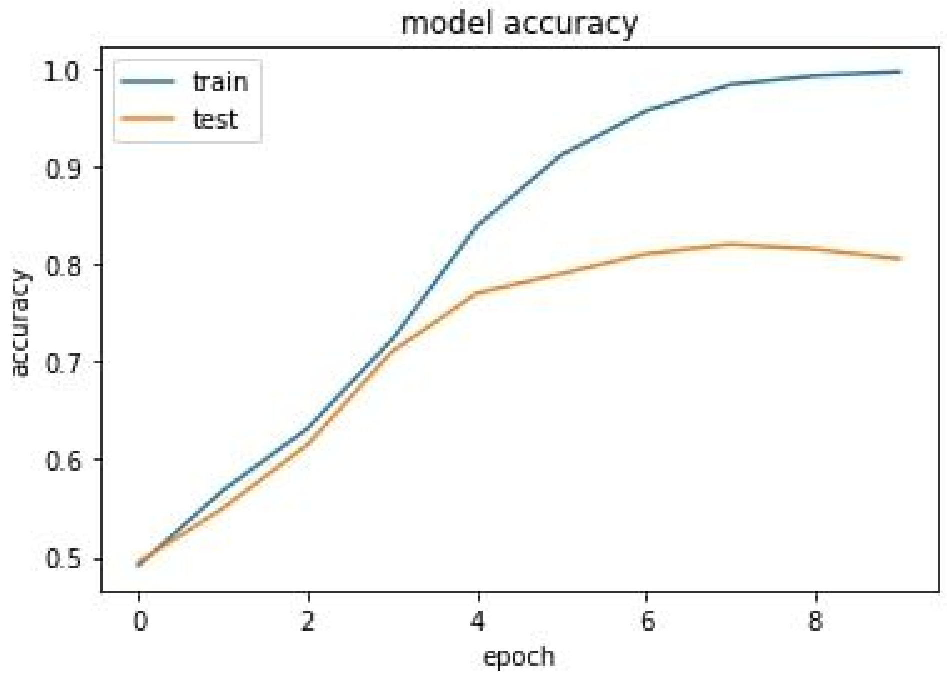

Fig.5 Accuracy of two convolutional layers CNN

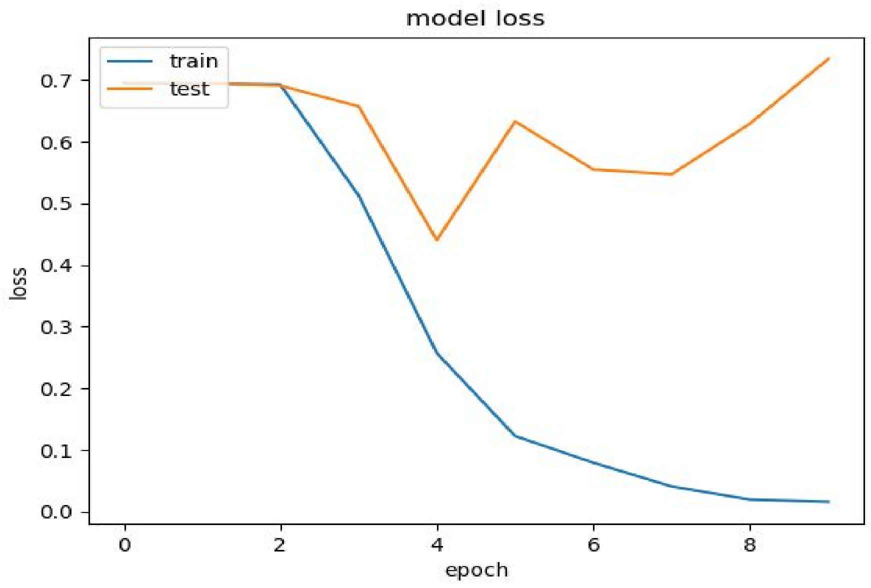

Fig.6 Loss of single LSTM by using the naive word embedding

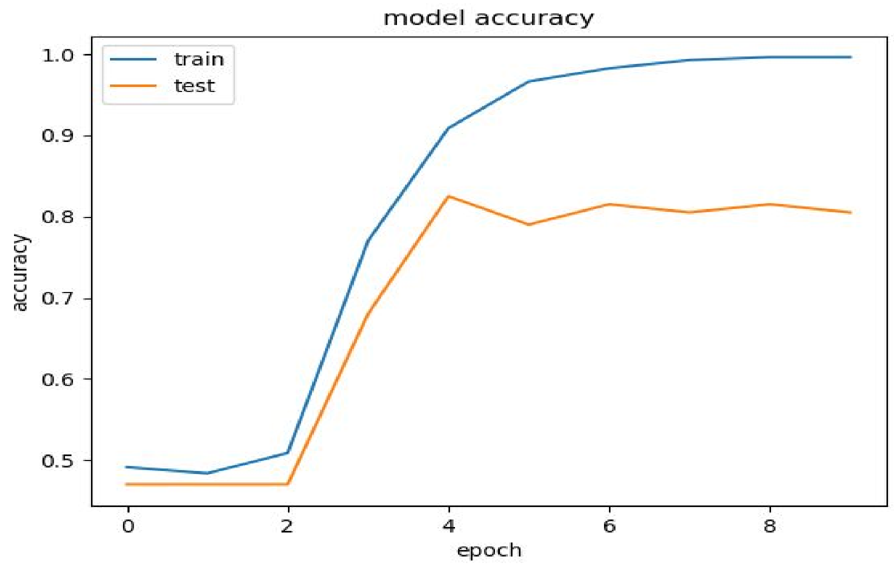

Fig.7 Accuracy of LSTM model by using the naive word embedding

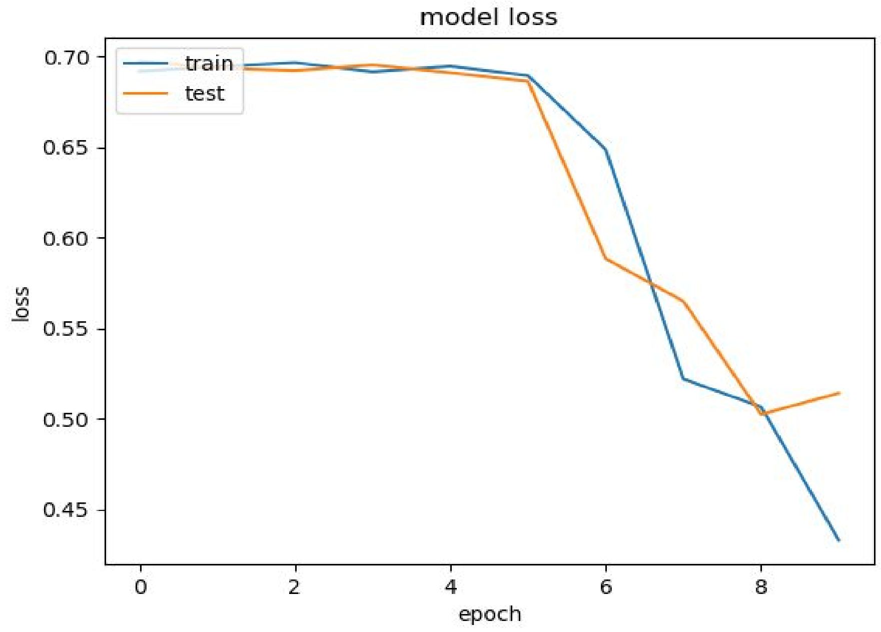

Fig.8 Loss of LSTM by using Word2Vec

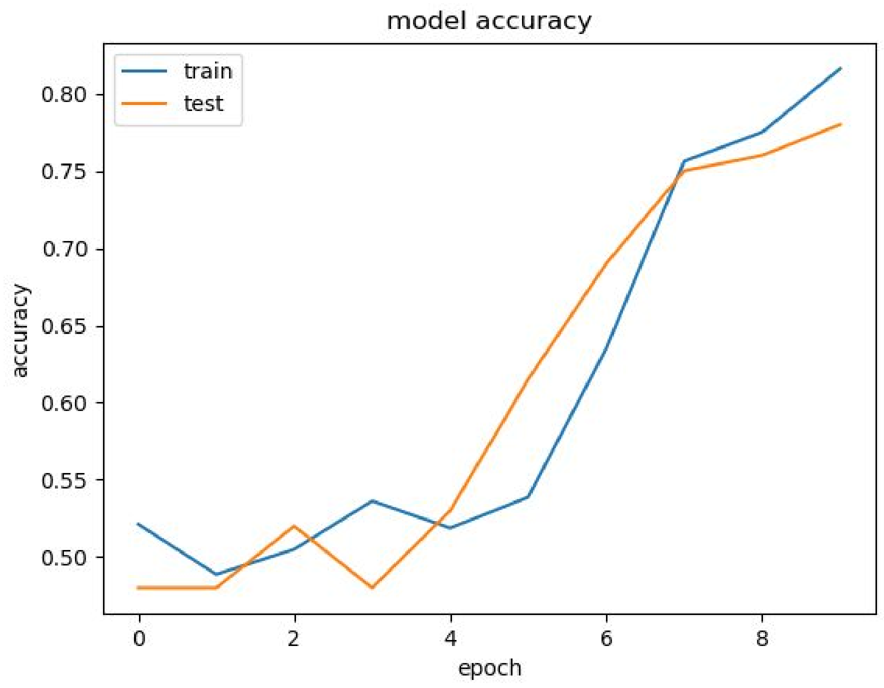

Fig.9 Accuracy of LSTM by using Word2Vec

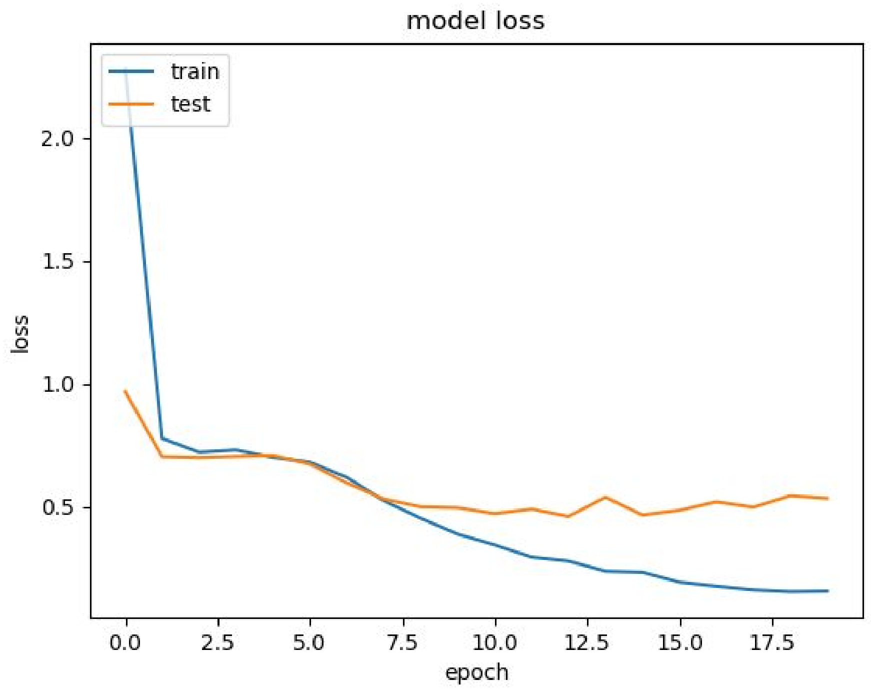

Fig.10 Loss of bidirectional LSTM with using Word2Vec

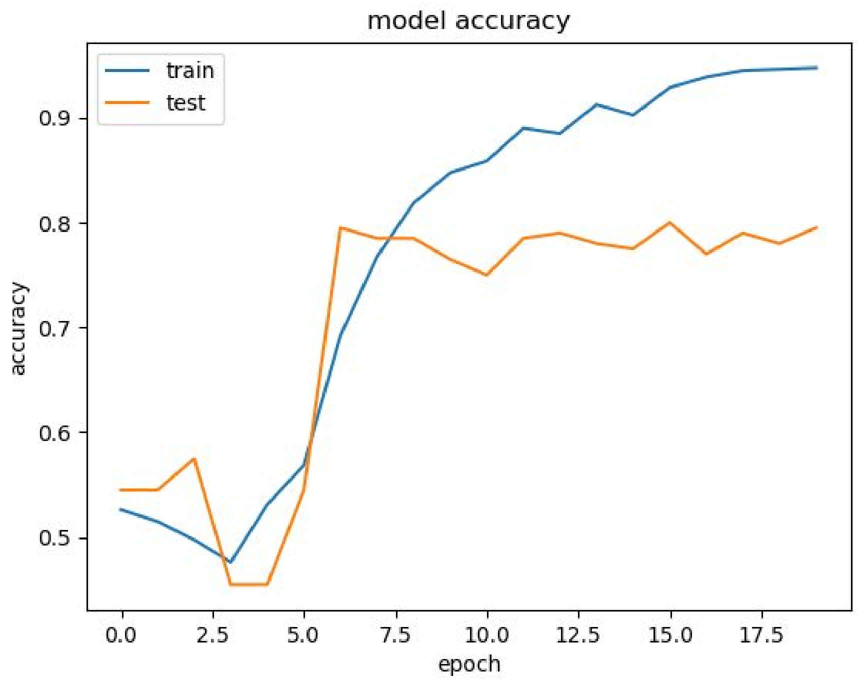

Fig.11 Accuracy of bidirectional LSTM with using Word2Vec

Member Contribution

Yongheng Zhang:

Did relevant research about this project, including the state of art and papers related to the sentiment analysis. Analyzed the final result of this project and wrote future improvements.

Chenhao Li:

Built the recurrent neural network and wrote the relevant part (architecture and result) in the report. Rearrange the report in order like a formal paper.

Qihao Huang:

Did data preprocessing, provided ideas to improve the performance, and checked and edited the report.

Zihua Zhang:

Built the convolutional neural network, wrote the architecture and the result of CNN in the report.

Lixiong Feng:

Initiated the team, invited all members in--all but one group member did not know each other before the project. Encouraged members to communicate with each other, constructed the group chat. Called for all the group meetings, forced teammates to talk and share their ideas during group meetings. Helped to develop the CNN, and investigated together with teammates in Word2Vec/Doc2Vec. Prepared the team for the presentation, and created the first prompt of powerpoint.