Sentiment Prediction through Deep Neural Network

Guankai, Sang
Boston University

Yifan, Zhang
Boston University

August 11, 2021

Abstract

Nowadays, people rarely have time to finish all movies, even the latest one, therefore movie reviews become a choosing standard. Having a high accuracy on sentiment prediction of movie reviews, people can save their time even more by not reading each single review and still can know others' attitude on those movies. We trained a Multi-layer neural network with IMDB data imported from Keras. Even the dataset has been already processed, we still need to do word embedding and split validation dataset from training dataset. For increasing the accuracy of our model without overfitting, we changes parameters such as learning rate, epoch, and also adding more hidden layers with proper dropout rates between them. Finally, two-hidden-layer neural network with adjusted parameters not only gets a higher model accuracy around 95% but also achieve a higher validation accuracy around 88% before overfitting.

1 Dataset Preparation

1.1 Review Length Limitation

To do a text classification, the first step of building a model is word embedding, which maps words in movie review to a high dimensional vector. Since all neural network requires to have input with same size, we need to truncate or pad the dataset before we do word embedding.

Shown as Figure 1, the mean of review length in IMDB dataset is 234.76 words and the standard deviation is 172.911. Also, we can see from the boxplot, that the maximum of the dataset is around 500. Therefore, even there are multiple outliers that have over 500 word length, we still can set the value of maxlen in pad_sequences function as 500 to cover the most of dataset. Then, this function will take the original training and testing dataset and truncate reviews longer than 500 or pad reviews shorter than that with 0.

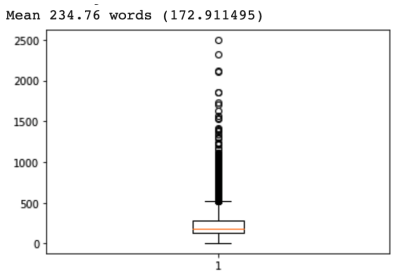


Figure 1: Distribution of review length in words

1.2 Distribution of dataset

In order to check the power of our model, we need to have a standard. The most common one is accuracy, but if our dataset is imbalanced, then we cannot use it, since each category with different number of data may cause the bias of the result, model might give a better accuracy result for the category with more training data, so the value of accuracy will not be stable and reliable. For our training dataset shown in Figure 2, it is balanced, so accuracy is available as the standard for our future tests.

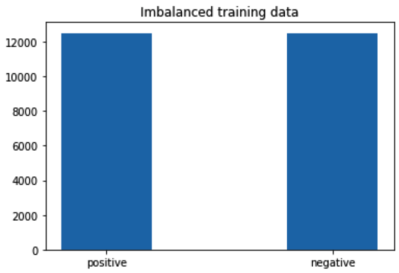
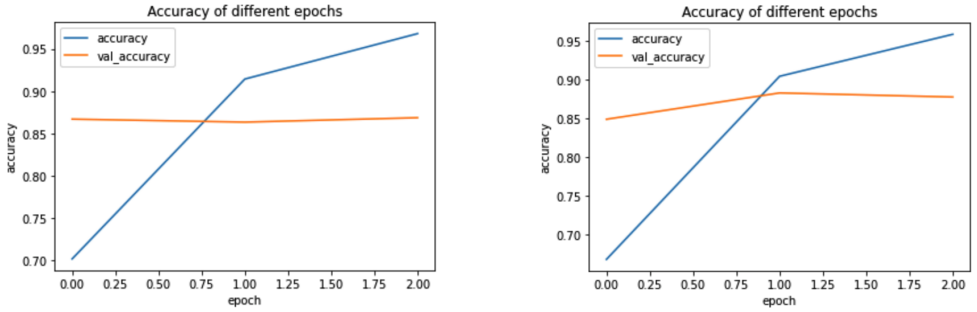


Figure 2: Distribution of training dataset

1.3 Validation Dataset

A major challenge of training a neural network is overfitting, which means that a model acts so well on training dataset, but not on new coming dataset. One of way to solve it is through validation dataset in order to update parameters of model so that model will act well on general data.



(a) Test dataset as validation

(b) With extra validation dataset

Figure 3: Accuracy of models with or without validation dataset

By training a multi-layer models with extra validation dataset or not, we can see the accuracy difference between them. Figure 3 shows that the validation accuracy of model with no extra validation decreases from the first epoch when the accuracy of the model continuously increases. However, the validation accuracy of model with extra validation dataset decreases after the first epoch. This means without extra validation dataset, model will become overfitting earlier. Also, shown as Table 1, model with extra validation has the highest model accuracy. Therefore, in order to have enough training and validation dataset as the same time, we separate a half of training dataset as validation dataset after shuffling.

Table 1: Accuracy of models

	No validation	Test dataset as validation	With extra validation
Accuracy	86.94	86.85	87.14

2 Modeling and Optimization

The starting model that we build has one word embedding layer, a flatten layer, and two dense layers. Output of embedding layer is 32×500, since we would like to have word vector in 32 dimensions and the input length at 500 as discussed previously. Activation for the hidden layer is Relu, and that for the output layer is Sigmoid in order to get results in percentage. Optimizer is Adam with learning rate 0.0005. Training this model in three epochs with extra validation dataset, we get model accuracy around 87% and early overfitting.

2.1 Learning rate

One of the way to avoid overfitting is picking a proper learning rate, which can help model get higher accuracy before overfitting appears. Shown as Figure 4, learning rate in 0.005, which is our starting learning rate, it has highest starting accuracy, and it can even reach closely to 1 at the end. However, from Table 2, we can see that even model with learning rate 0.005 has highest model accuracy at the beginning, it gets overfitting earlies as well, and when the accuracy at that time is just around 80%. Among all the learning rate value, 0.0001 gives the best result. Its model accuracy before overfitting is 94.42%, which is the highest score than others. Therefore, we changes learning rate to 0.0001.

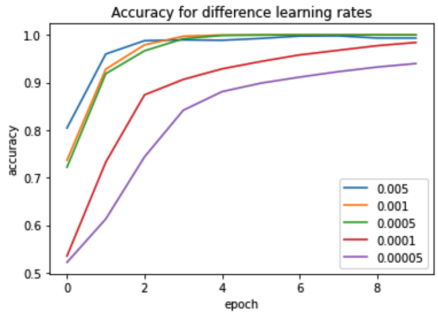


Figure 4: Accuracy of different learning rate

Table 2: Accuracy and validation accuracy of different learning rate

Learning rate	Overfitting appears at	Accuracy
0.005	Epoch 2	0.8042
0.001	Epoch 3	0.9281
0.0005	Epoch 3	0.9184
0.0001	Epoch 7	0.9442
0.00005	Epoch 10	0.9393

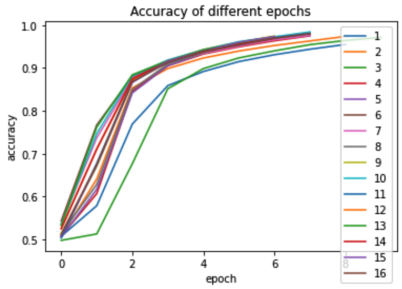
2.2 Hidden layer

Another way to increase the accuracy of the model is having deeper layer. However, having more layers also means more parameters and more complex model, which could cause overfitting. Therefore, choice on neuron number and balance between neuron and depth of the model becomes important.

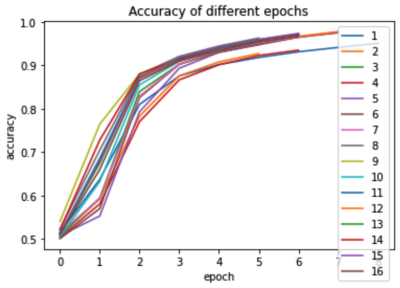
By using EarlyStopping function with patience value 2, we will get the accuracy of the model before validation accuracy decreases too much, which caused by overfitting. The result is shown as Figure 5. From part (a), we can clearly see that for the first layer, 10 neurons gives the highest accuracy before overfitting, and for the second layer, seeing from part (b), we get the highest accuracy with 12 neurons. In part (c), models with 14 neurons gives the best result. However, shown in Table 3, the validation accuracy of the model with either only one hidden layer or three hidden layers is not as good as the model with two hidden layers. Therefore, we will only use two hidden layers, one with 10 neurons and the other with 12 neurons.

Table 3: Accuracy of different depth

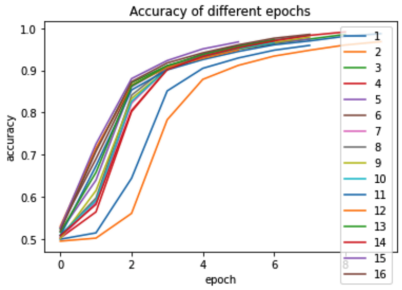
layer number	neuron number	Model accuracy before overfitting	val_accuracy
1	10 neuron	0.9614	0.8695
2	12 neuron	0.9526	0.8762
3	14 neuron	0.9701	0.8618



(a) Hidden layer 1



(b) Hidden layer 2



(c) Hidden layer 3

Figure 5: Accuracy depended on neuron number in each layer

2.3 Dropout

Adding a dropout layer after dense layer helps prevent overfitting as well. Since it drops a percent of neurons due to the setting up dropout rate and then continue training the model, this add some randomness during the training process. Seeing from the Figure 6, we can see that our model achieve highest accuracy when dropout rate equals to 0.3, and the accuracy of the model is 0.9445 and validation rate is 0.8717. Since this does not give a higher model accuracy or validation accuracy, we will not use any dropout layer for our final model.

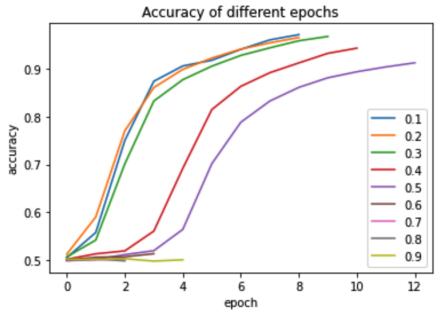


Figure 6: Accuracy with various dropout rate

3 Conclusion

The major challenge in our model is the decision on the proper review length as the input of word embedding layer and on the parameter in the deep neural network to avoid overfitting while increasing the model accuracy. Through the analysis of the distribution of general review length, we decide the input of word embedding layer as 500. Shown as Table 4, by manipulating learning rate, we increase the model accuracy significantly, and then by adding hidden layer, changing number of neurons and trying various dropout rate, we improve the validation accuracy while keeping the model accuracy around. Our final model has learning rate in 0.0001 with two hidden layer. The first hidden layer with 10 neurons and the second with 12 neurons. Even though we get high model accuracy as 0.9526 and validation model as 0.8741, it still can be further improved. In our model, we try to set each dropout layer with the same dropout rate, and it does not give us a better result, but for future improvement, we can make them different according to the specific output of the previous dense layer. We could also try to use 1D CNN based on our final model.

Table 4: The accuracy of all four methods

Changes	Model accuracy	Validation accuracy
Basic Model	0.7054	0.8709
learning rate	0.9595	0.8741
Depth of the model	0.9526	0.8762
Dropout rate	0.9445	0.8717