Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/6/20230830

Sentimental analysis for clothing dataset

Zhezhong Ding[1, 4], Zhuoheng Du[2], Qingqi Yang[3]

[1]Wuhan Britain-China School, Wuhan, 630000, China, zhezhongding@163.com
[2]Chongqing Nankai Secondary School, Chongqing, 404100, China, 1924479827@qq.com
[3]The High School Attached to Hunan Normal University, Changsha, 410000, China 423198@qq.com

[4]zhezhongding@163.com

Abstract. In recent years, sentiment analysis has been applied to various fields, including politics, business, education, etc. Simultaneously, with the development of science and quality of life, clothes have become an increasingly important part of people's daily lives, especially women. In this research, using Natural Language Processing, we interpret a dataset about clothing by trying different approaches, which include other models (CNN, RNN, and LSTM), different optimizers (Adam, RMSProp, and SGD), and different hyper-parameters (Epochs, Batch Sizes, and Learning Rate). Then, we analyze the influences of those approaches by all kinds of valuation standards. Ultimately, we improve our accuracy by 8.0 %.

Keywords: sentimental analysis, machine learning, CNN, RNN, encoding.

1. Introduction

Sentiment Analysis [SA) [1] has become an essential tool for many companies and organizations to recognize people's opinions accurately. Specifically, using Machine Learning, SA helps them tell the emotional attitude of each review - positive or negative, for example - and then prepare for their next operation. For the government [2]. SA aids in identifying the attitudes of political opinions, so the politicians can collect information about their support rate and make a better plan for the future; of online shopping websites [3]. SA gives sentimental details on the product reviews, so the staff can determine whether a product should be pulled off and then follow the current trend. SA has also been widely used in finance [4,5], healthcare [6], education [7,8], etc. [1,9].

Due to the ever-increasing technology and life quality, people's, significantly women's, pursuit of beauty is growing stably. Among the products for decoration, clothes have become an indispensable part. Thus, clothing stores need to analyze consumers' comments to plan for their future products and regulate their sales products properly. Fortunately, Natural Language Processing (NLP), which includes Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM), can be used to help train the models for sentiment interpretation.

In this research, we try to interpret a Women's Clothing E-Commerce dataset revolving around customer reviews. The dataset consists of 23486 rows and ten feature variables, but only the parts about Review Text and Rating are used in the research. In the course of the study, we used different models,

DOI: 10.54254/2755-2721/6/20230830

optimizers, and hyper-parameters, to analyze their influences of mainly f1 score and accuracy of the result and improve the accuracy by 9.0 %.

2. Preliminaries

This section introduces several kinds of text representation methods.

2.1. One-hot encoding

In one-hot encoding, each word w in the corpus vocabulary is given a unique integer ID (wid) between 1 and |V|, where V is the set of the corpus vocabulary. A V-dimensional binary vector of 0s and 1s then represents each word. This is done via a |V| dimension vector filled with all 0s barring the index, where index = wid.

2.2. Bag of N-grams

In order to capture some of the text, the bag-of-n-grams (BoN) technique divides the text into groups of n connected words (or tokens). Each unit is referred to as an n-gram. The unique n-grams found in the text corpus make up the corpus vocabulary, or V. Then, a vector with a length of |V| is used to represent each document in the corpus. If |V|=0, there are no n-grams in the text. This vector displays the frequency counts of the n-grams that are present in the article.Take the example of "The cat sat on the mat."

For Bag of 1-gram (unigram), the sentence can be represented as [2 1 1 1 2 0], in which Unigram Vocabulary maps for given sample corpus: {'the': 4, 'cat': 0, 'sat': 3, 'in': 2, 'hat': 1, 'with': 5}

For Bag of 2-gram (bigram), the sentence can be represented as [1 0 1 1 1 1 0], in which Bigram Vocabulary maps for given sample corpus: {'the cat': 4, 'the cat sat': 0, 'sat in': 3, 'in the': 2, 'the hat': 5, 'cat with': 1, 'with the': 6}

For Bag of 3-gram (trigram), the sentence can be represented as [1 0 1 1 0 0 0], in which Trigram Vocabulary maps for given sample corpus: {'the cat sat': 4, 'cat sat in': 0, 'sat in the': 3, 'in the hat': 2, 'the cat with': 5, 'cat with the': 1, 'with the hat': 6}

Note: the bags are sequenced alphabetically

2.3. TF-IDF

According to the TF-IDF approach, a word must be extremely important to the sentence if it appears multiple times in sentence S1 but not frequently in the other sentences Sn in the corpus. As n's frequency in S1 increases, so should its significance (how many times that word occurs in sentence S1). Additionally, its magnitude need to diminish in line with how often the term appears in other Sentences Sn in the corpus. In order to calculate the TF-IDF values, two quantities—TF and IDF—are first multiplied to provide a TF-IDF score.

A phrase or word's frequency within a certain document is measured by TF (term frequency).The mathematical Expression of TF is shown below:

$$TF(t,d) = \frac{(Number\ of\ occurrences\ of\ term\ t\ in\ document\ d)}{(Total\ number\ of\ terms\ in\ document\ d)}$$

IDF (inverse document frequency) measures the importance of the term across a corpus. In computing TF, all times are given equal importance. However, it's a well-known fact that stop words like is, are, am, etc., are not significant to the document despite their high frequency. IDF weighs down the general terms across a corpus and the rare terms to account for such cases. IDF of a term t is calculated as follows. The formula is shown below:

$$IDF(t) = log_e \frac{(Total\ number\ of\ documents\ in\ the\ corpus)}{(Number\ of\ documents\ with\ term\ t\ in\ them)}$$

The TF-IDF score is a product of these two terms. Thus, TF-IDF score = TF * IDF.

Let's consider an example.

Sentence A = "The Car is Driven on the Road."

Sentence B = "The Truck is Driven on the highway Computation of TF-IDF."

Scores are shown in Table 1.

DOI: 10.54254/2755-2721/6/20230830

Table 1. The calculation process of TF-IDF value.

Word	TF*IDF A	TF*IDF B
The	0	0
Car	0.043	0
Truck	0	0.043
Is	0	0
Driven	0	0
In	0	0
The	0	0
Road	0.043	0
Highway	0	0.043

For TF-IDF, the biggest problem is that the feature vectors are high-dimensional representations. The dimensionality increases with the size of the vocabulary.

3. Algorithms

3.1. CNN

Data will be analyzed in a model to make predictions based on the dataset after being tokenized and embedded. There are many different models, and Convolutional Neural Network is one of the tools we employ in this paper (CNN). This neural network belongs to a particular class of feed-forward neural networks used in computer vision, recommender systems, natural language processing, etc. In this essay, we will use CNN to perform NLP tasks.

The model consists of convolutional layers and pooling layers to a fully connected layer with some activation functions. Convolution layers are used to extract features of a particular area of data, the different size of the kernel used refers to the other extractors of the input data. Then the outputs of the layers will go through pooling layers to reduce the resolution of features, which significantly decreases the number of parameters and increase the model's robustness to noise and distortion by only taking parts of the output data to the next layer. Our model uses max pooling to take the maximum data from each layer's output. In the end, fully connected layers combine all the activated data streams from pooling layers to predict people's sentiments (positive or negative).

3.2. RNN (LSTM, and BI - LSTM)

The LSTM model contains three gates: the forget gate, the input gate, and the output gate. Each gate has a specific ability, such as storing information, omitting information, etc. By the LSTM, we can preliminarily solve the long-term dependence problem since LSTM can forget the early lead and store some data.

Bi - LSTM (Bidirectional LSTM), has the purpose of making the feature data obtained at time T have context information. It has two independent LSTM models and shares the same embedding vector list. The input list is put into two separate LSTM models in positive and negative sequence, then extract the feature.

4. Experiment

In this section, we try optimizers, the number of epochs, learning rates, and batch sizes to find the combination with the highest accuracy.

DOI: 10.54254/2755-2721/6/20230830

4.1. CNN

4.1.1. Optimizer. Adam and RMSprop in Figure 1 do not show significant differences in the dataset of women's Clothing sentiments. However, Adam does have more converged accuracy because it's the combinations of RMSprop and SGD, which can effectively adapt to different neural network structures.

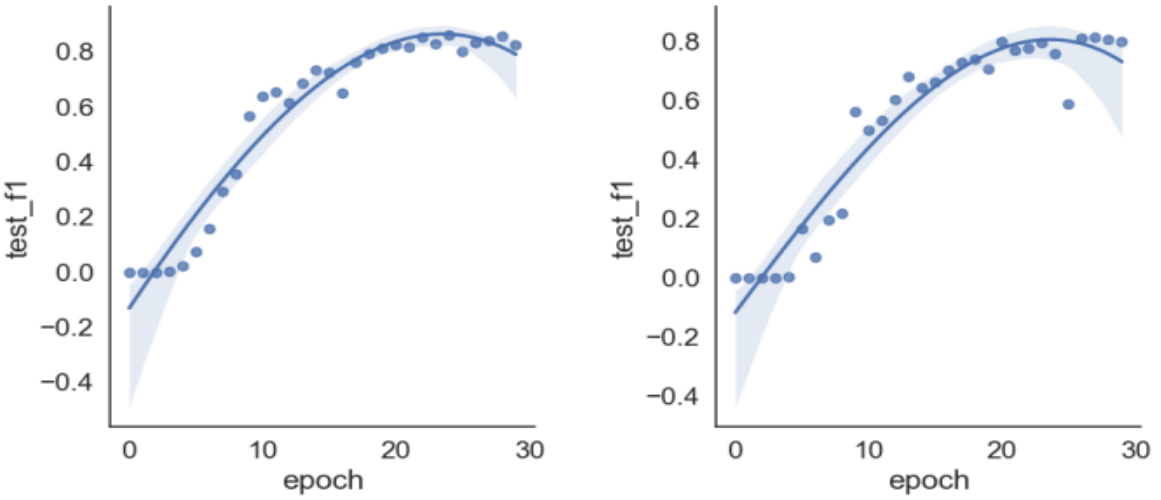

Figure 1. Model's training f1 score with Adam(left) and RMSprop(right) as optimizer.

4.1.2. Epoch. For 10 epochs, f1_score is concave up (Figure 2). For 30 epochs, however, the whole graph (Figure 3) is open down. When epoch=25，f1of the test is approximately 75%， and f1 of the train shown in Figure 3 is about 80%±5%. Therefore, for 10 epochs, the model does not reach the training completion state, and the fitted function is also very different from the nearly converged model at 30 epochs. Therefore, the model used for our dataset cannot be effectively identified in 10 epochs, but it can be effectively placed in 30 epochs.

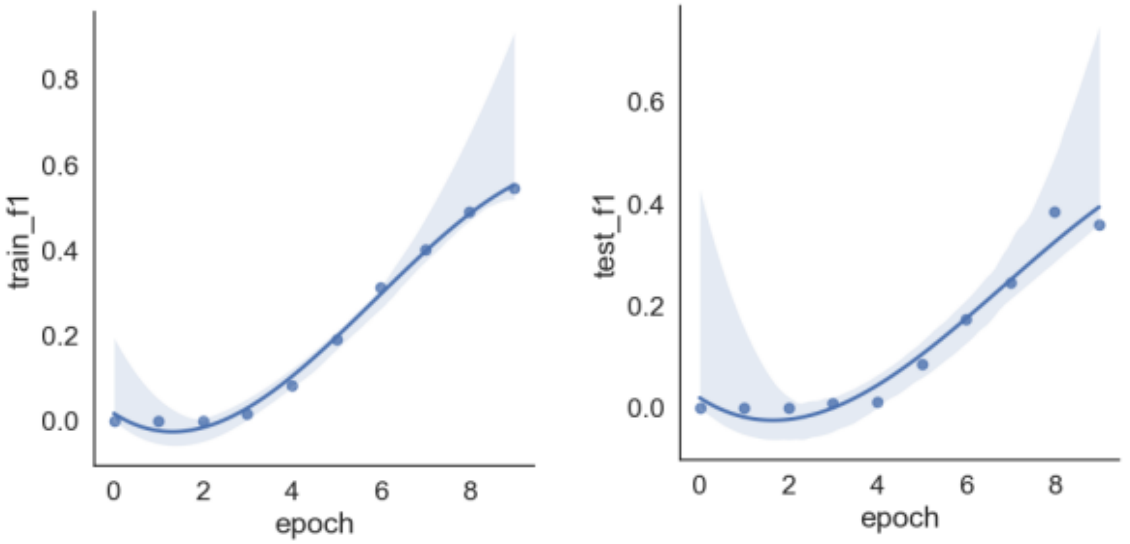

Figure 2. The f1 score of train set(left) and test set(right) after 10.

DOI: 10.54254/2755-2721/6/20230830

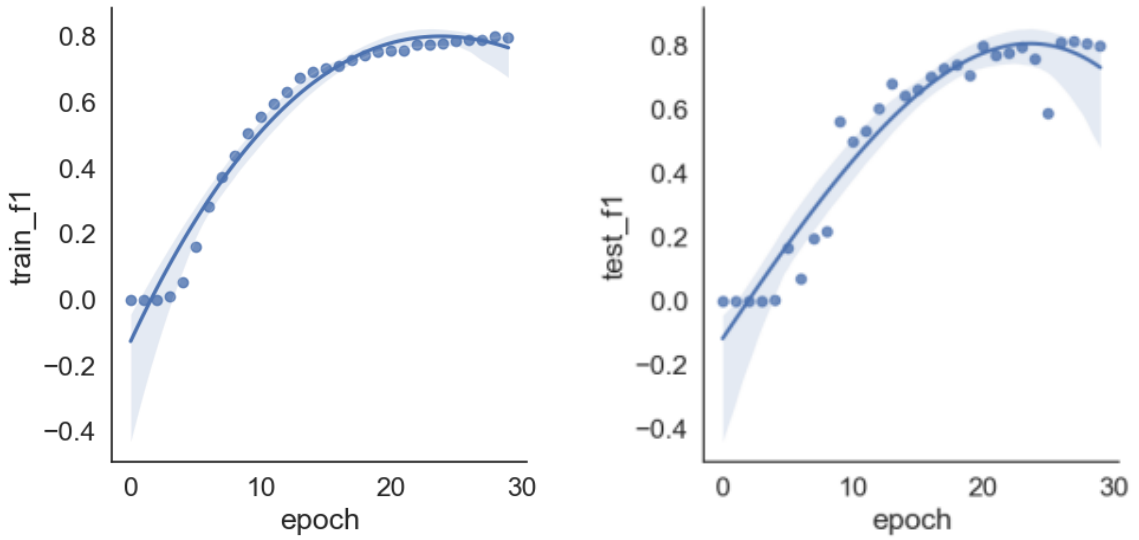

Figure 3. The f1 score of train set(left) and test set(right) after 30 epochs of training.

4.1.3. Learning rate. 0.1: The Learning rate is too large, so the step forward is too long. The hyperparameter oscillates back and forth. Both test_f1 and test_acc shown in Figure 4 are 0, meaning that the model cannot converge.

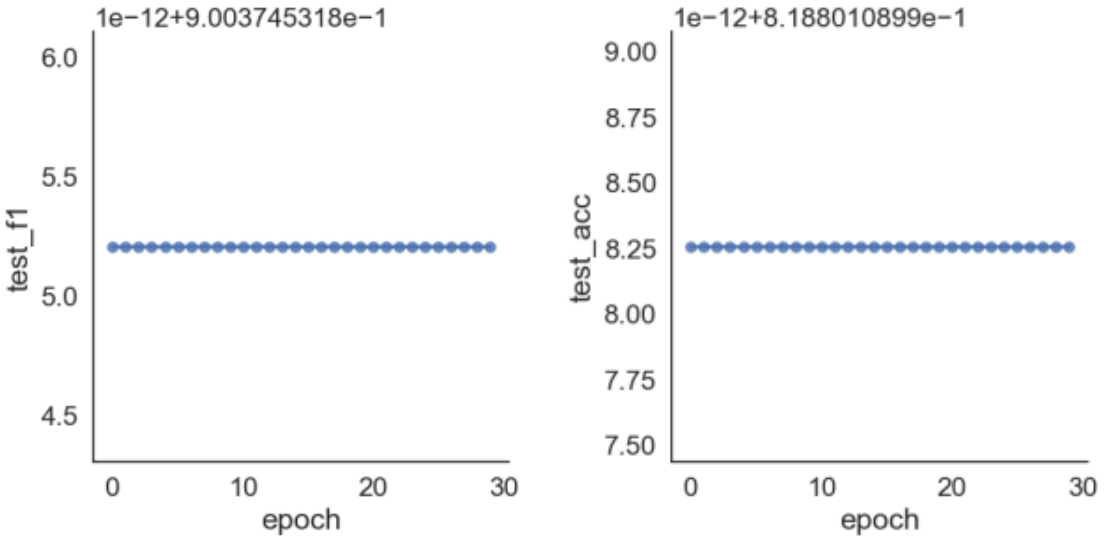

Figure 4. The test_f1 score(left) and the test_acc(right) with 0.1 learning rate.

0.0001: The learning rate is too low. The learning rate is too low, so the loss will not converge. Here, the model starts to converge from the last five epochs shown in Figure 5 (left), and the convergence speed is too slow. According to Figure 5(right), the model has not yet started to update.

DOI: 10.54254/2755-2721/6/20230830

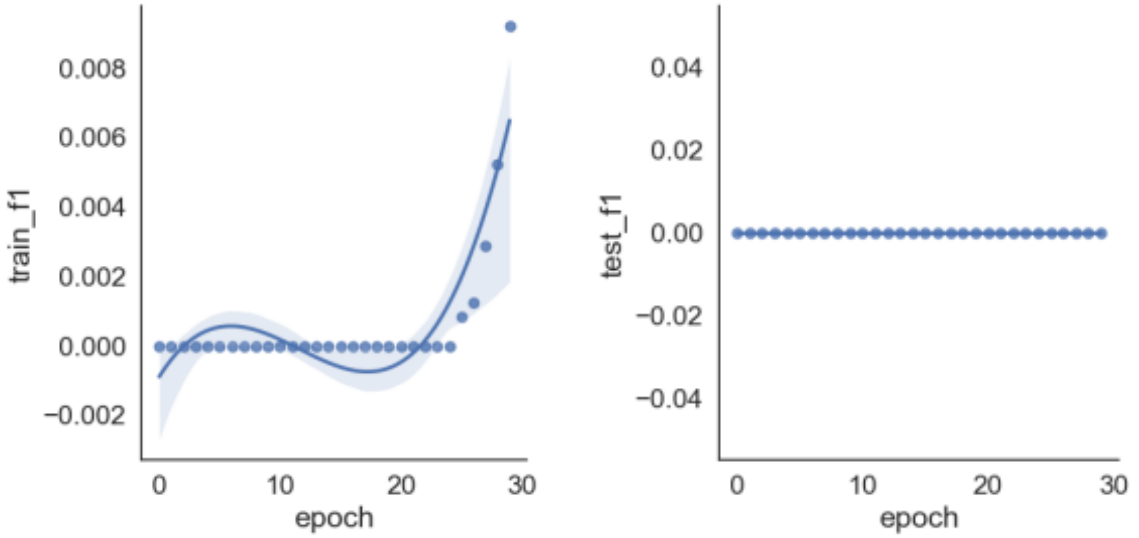

Figure 5. Low converge speed of training data set with 0.0001 learning rate(left); the test f1 score has not even been updated yet.

0.001： This is the most optimal learning rate of the model. From Figure 6, the maximum f1_score can be identified with the value of 0.75, and train_acc converges to 0.95 in 30 epochs.

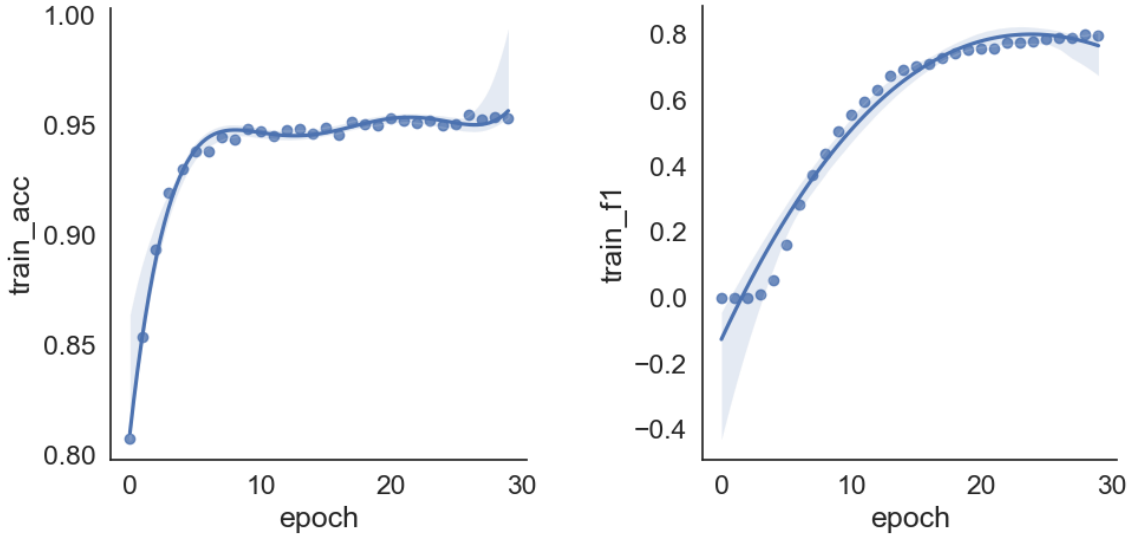

Figure 6. The train set accuracy(left) and the f1 score(right) with the suitable 0.001 learning rate.

Model conclusion:

Using Adam with 0.01 learning rate and 30 epochs is the most effective way to optimize our CNN model in this dataset.

4.2. LSTM

In our experiment in LSTM, we use 30 epochs, 64 batch size, 0.001 learning rate. These two standards shown in Figures 7 and 8 are not an ideal number; each of them has high loss and low accuracy and f1_score. We'll change each variable to make it perform better next.

DOI: 10.54254/2755-2721/6/20230830

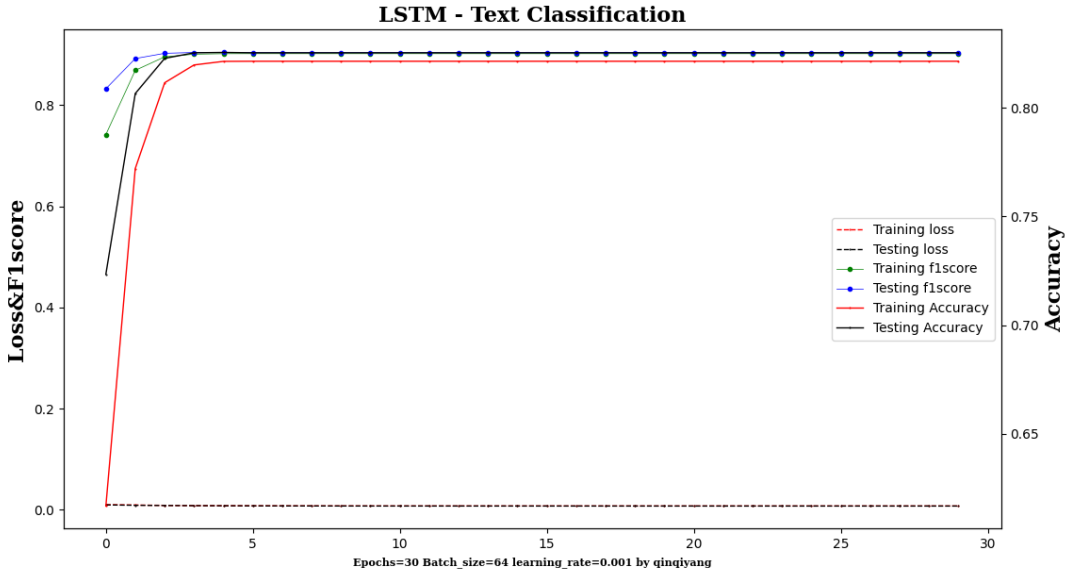

Figure 7. Standard LSTM model using parameters listed above.

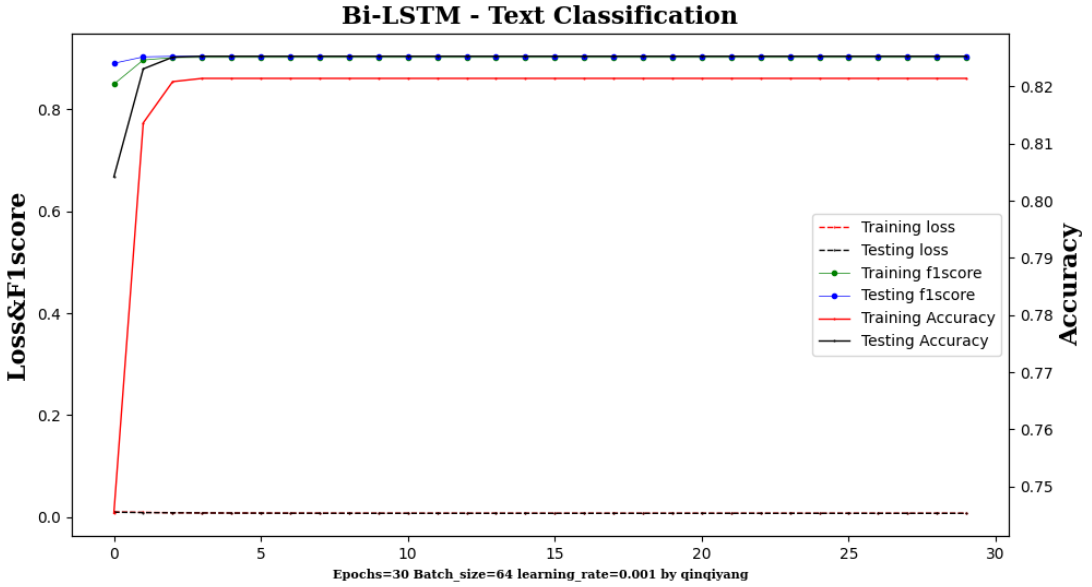

Figure 8. Standard bi-LSTM model with parameters listed above.

4.2.1. Change in epoch: from 30 to 10. The change is not apparent. Comparing Figure 7 and Figure 9, we can only see that the initial training f1_score is a little smaller than the standard in LSTM model. Compare Figure 8 and Figure 10, the initial training f1_score is more significant than standard in bi-

DOI: 10.54254/2755-2721/6/20230830

LSTM model, and the change finished in epoch 1. In that case, the epoch increase can increase the speed of change but cannot increase or decrease any value.

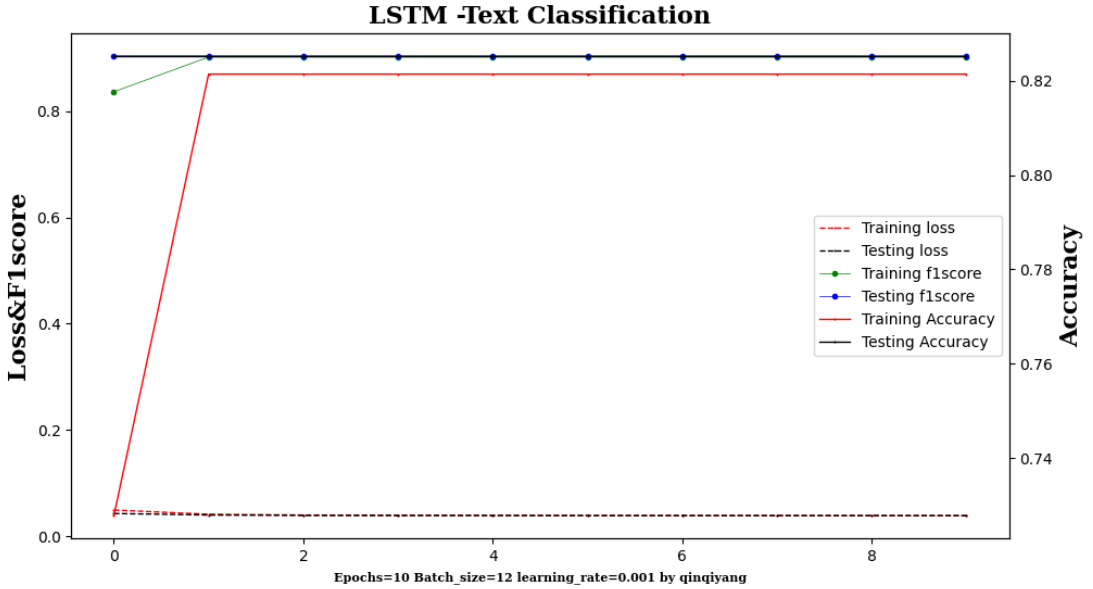

Figure 9. LSTM model with ten epochs.

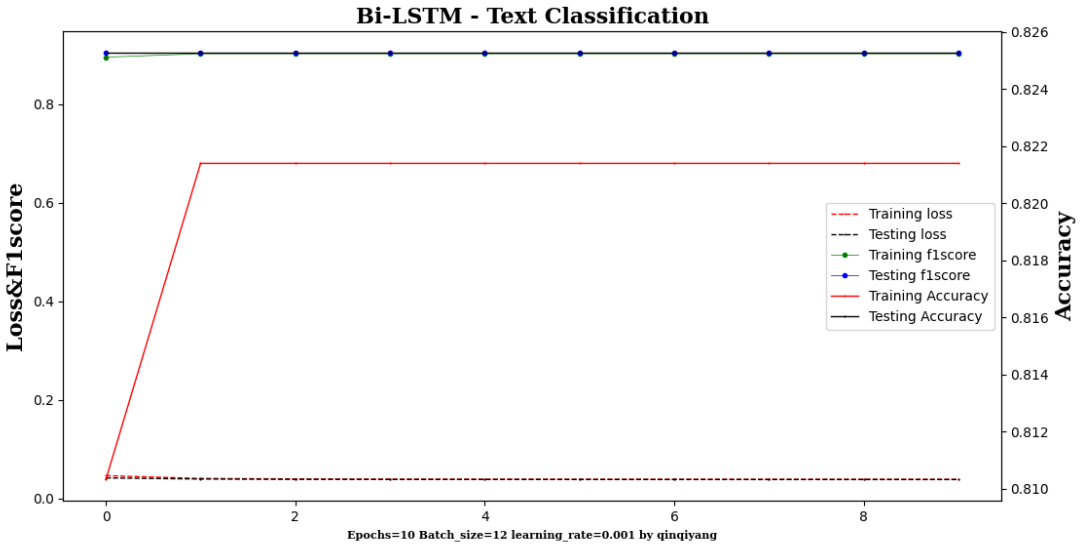

Figure 10. Bi-LSTM model with ten epochs.

4.2.2. Change in learning rate: from 0.001 to 0.1(batch size: 64). In this part, the batch size is 64 since we know that the bigger the batch size is, the more pronounced the results are.

Figures 11 and 12 show that the learning rate change is noticeable. Although the fluctuation is big, we can see that the ending of each accuracy is high, with three up to around 88% and one up to 85%-86%. However, the line between training accuracy and testing accuracy in Figures 11 and 12 is not a constant, which means that the higher the learning rate, the less stable accuracy. Also, in each figure, we can see that f1_score have a slight fluctuation at epoch 10 and 19 in figure 11 represent LSTM model, and epoch 9 and 18 in figure 12, representing bi-LSTM model. These fluctuations follow the training

DOI: 10.54254/2755-2721/6/20230830

accuracy, according to f1_scores formula. According to the results, the increasing learning rate can raise the accuracy but increase the change.

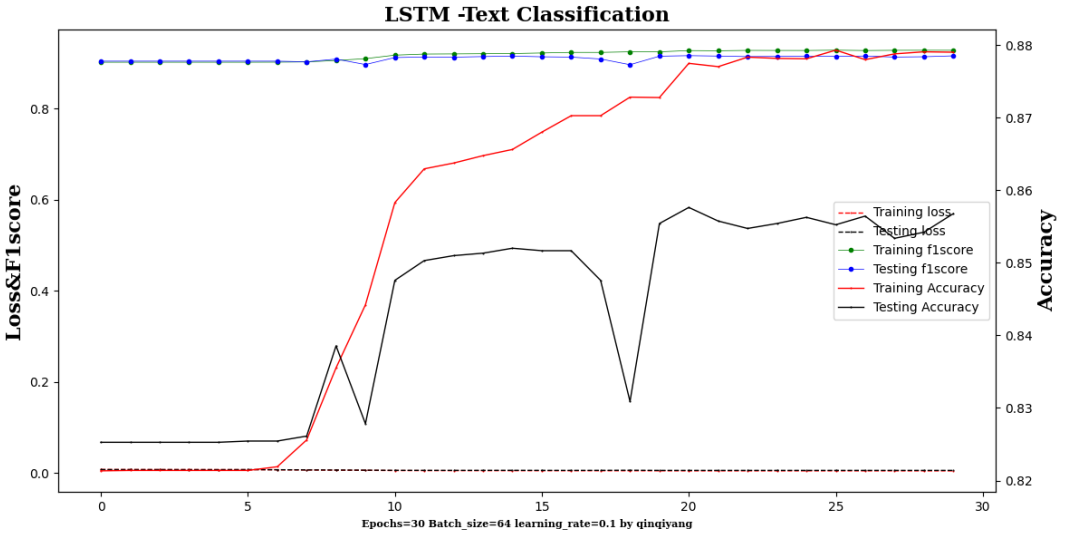

Figure 11. LSTM model with 0.1 learning rate.

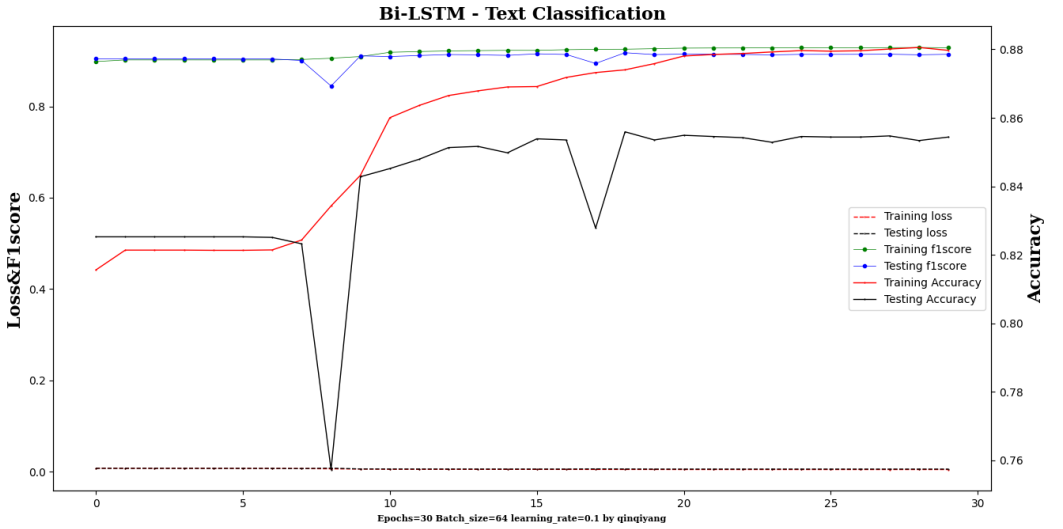

Figure 12. Bi-LSTM model with 0.1 learning rate.

Model conclusion:

Now, we already know that the increase of epoch can increase the speed of changing; the smaller the batch size is, the faster the speed is, and the increasing learning rate can raise the accuracy but increase the fluctuation. Thus, we choose to use 50 epochs and a 0.01 learning rate to get a successful graph of this dataset. Each model improves the accuracy and reduces the loss. In our advanced LSTM model and bi-LSTM model, the highest accuracy is up from 82.0% to 91.0%.

5. Conclusion

In this paper, we mainly introduce the different methods for text representation, use other methods to analyze the given dataset, and try to improve the accuracy. Firstly, we briefly introduce the current

DOI: 10.54254/2755-2721/6/20230830

development in NLP and clarify our purposes for writing the paper. Then, we present four methods for embedding: One-hot encoding, Bags of N-grams, TF-IDF, and Word2Vec. Besides, we explain the algorithms of CNN and RNN (including LSTM and BI-LSTM). We also try three optimizers (Adam, RMSProp, SGD), three different epochs (10,25,30), and three kinds of learning rates (0.01, 0.001, 0.0001) in CNN, thus increasing the accuracy by 5.0%. Similarly, we also try two different epochs (10,30) and two learning rates (0.01, 0.001) in LSTM and BI-LSTM, thus increasing the accuracy by 9.0%.

Acknowledgment

Ding Zhezhong*: Conceptualization; model evaluation - CNN; writing - review, and editing.
Du Zhuoheng*: Methodology in text representation; writing- original draft preparation; typography.
Yang Qingqi*: Conceptualization; model evaluation - LSTM & BI - LSTM.
All authors contributed equally to this work and should be considered co-first authors.

References

[1] Dang, N. C., Moreno-García, M. N., & De la Prieta, F. (2020). Sentiment analysis based on deep learning: A comparative study. Electronics, 9(3), 483.
[2] Balahur, A., Steinberger, R., Kabadjov, M., Zavarella, V., Van Der Goot, E., Halkia, M., ... & Belyaeva, J. (2013). Sentiment analysis in the news. arXiv preprint arXiv:1309.6202.
[3] Agarwal, A., Xie, B., Vovsha, I., Rambow, O., & Passonneau, R. J. (2011, June). Sentiment analysis of twitter data. In Proceedings of the workshop on language in social media (LSM 2011) (pp. 30-38).
[4] Mouthami, K., Devi, K. N., & Bhaskaran, V. M. (2013, February). Sentiment analysis and classification based on textual reviews. In 2013 international conference on Information communication and embedded systems (ICICES) (pp. 271-276). IEEE.
[5] Thet, T. T., Na, J. C., & Khoo, C. S. (2010). Aspect-based sentiment analysis of movie reviews on discussion boards. Journal of information science, 36(6), 823-848.
[6] Mite-Baidal, K., Delgado-Vera, C., Solís-Avilés, E., Espinoza, A. H., Ortiz-Zambrano, J., & Varela-Tapia, E. (2018, November). Sentiment analysis in education domain: A systematic literature review. In International conference on technologies and innovation (pp. 285-297). Springer, Cham.
[7] Zhou, J., & Ye, J. M. (2020). Sentiment analysis in education research: a review of journal publications. Interactive learning environments, 1-13.
[8] Çoban, Ö., Özyer, B., & Özyer, G. T. (2015, May). Sentiment analysis for Turkish Twitter feeds. In 2015 23nd Signal Processing and Communications Applications Conference (SIU) (pp. 2388-2391). IEEE.
[9] Balahur, A., Steinberger, R., Kabadjov, M., Zavarella, V., Van Der Goot, E., Halkia, M., ... & Belyaeva, J. (2013). Sentiment analysis in the news. arXiv preprint arXiv:1309.6202.