Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/5/20230583

Sentiment prediction based on neural network

Chen Kuangyu

City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong SAR

kuangchen5-c@my.cityu.edu.hk

Abstract. Recent years have seen a surge in interest in deep learning, and numerous new neural network models have appeared. In this study, we use CNN and LSTM, two popular deep learning models, to predict sentiment. We use the bidirectional model for the LSTM model. We make predictions by integrating the feature vectors of both sides after receiving sentences inputted in the reverse order from both ends. The recurrent neural network layer is replaced with the CNN model, which defines numerous one-dimensional convolution kernels to conduct convolution operations on the input. The two models are assessed and compared based on how well they anticipate and analyze sentence emotion tones. By comparison, we discover that the BiLSTM model's prediction accuracy on the test set is 10% greater than CNN's, despite requiring more training time.

Keywords: deep learning, sentiment prediction, neural network

1. Introduction

Artificial Intelligence (AI) has long been a popular issue due to the quick advancement of technology. Machine learning is the core of AI research. Expert systems, autonomous reasoning, natural language comprehension, pattern recognition, computer vision, intelligent robotics, and other disciplines of AI have all used it. For instance, one of the machine learning techniques, quantitative structure-activity relationship (QSAR) modeling, may quickly identify possible biologically active molecules from millions of candidate compounds. [1]. When the data become “bigger and bigger”, people tend to use deep neural networks to do some analysis and application which is known as deep learning. As a subset of machine learning, deep learning is used in many aspects because of its power and efficiency. The 2016 match between AlphaGo and Lee Sedol demonstrated was a classic example and made the tool familiar to the public. Research and application in recent years showed that deep learning has replaced previous related technologies, and extraordinary breakthroughs have been made in image recognition [2] and speech recognition. However, this does not imply that deep learning has been developed and matured, and it still needs further theoretical analysis and application practice.

In this paper, deep learning is utilized to the text to analyze and predict whether the emotional tone contained in the text is positive or negative. We pick a sentence from the book or classic celebrity quotes and transform it into vector form. Finally, the emotion analysis is carried out by a convolutional neural network.

Different types of neural network models have great differences in the processing and understanding of input text content. For example, RNN (Recurrent Neural Network) simply predicts and analyze the following text by the previous one or two words. It can handle certain short-term dependencies, but it

DOI: 10.54254/2755-2721/5/20230583

can't handle long-term problems. Such as "The Man comes from France, he speaks..." Preceding France is important for predicting the final language spoken. Because of long sentences, RNN cannot handle this type of prediction. However, the state value of the previously buried layer can be combined with the current input value using LSTM (Long Short-Term Memory), and use the sigmoid function to choose or reject information [3]. In this way, information can be saved and controlled to make correct judgments. LSTM can be understood as a kind of RNN with more complex neurons, but it has different effects on text processing. Therefore, the application and analysis of different models are crucial in sentiment analysis.

This essay will essentially be broken into five sections. The regression model for machine learning and the neural network model for deep learning will be introduced in the second section. The deep learning model and data processing technique employed in this paper will be the main topics of the third section. The fourth section will display the results of the predictions and highlight the benefits and drawbacks of various models. Finally, a summary and future application guidance are provided.

2. Overview

2.1. Machine learning and deep learning

The results and efficiency of machine learning depend greatly on how the raw data is represented. Researchers have long been trying to find features in raw data and use mathematical models to represent them [4]. Traditional machine learning usually uses regression models to fit data. Linear regression, polynomial regression, ridge regression, lasso regression, and elastic net regression are common regression models.

In essence, deep learning is a specific subset of machine learning. A vast number of matrix numbers are sent into the multi-layer artificial neural network as input. After the nonlinear activation method takes the weights, a data set is produced as the output. Deep learning performs far better at fitting specific data sets than regular machine learning. For instance, standard machine learning struggles to get a decent fit in the prediction of protein interaction because of the imbalance and nonlinearity of the data [5]. Specifically, we compare the following points:

2.1.1. Dependence on data. For accurate interpretation, deep learning algorithms frequently require a huge amount of data. Deep learning will therefore outperform machine learning when the volume of data is quite huge.

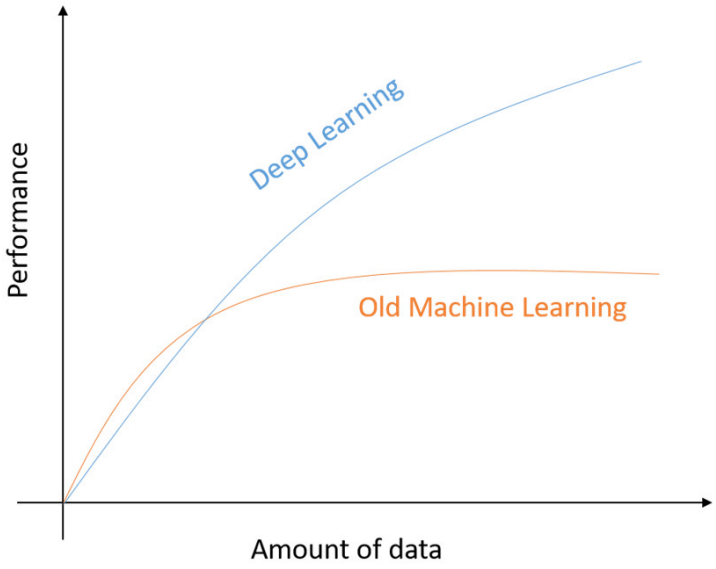

Figure 1. When the amount of data increases, deep learning outperforms machine learning.

DOI: 10.54254/2755-2721/5/20230583

2.1.2. Hardware requirements. Given the numerous matrix operations needed for deep learning, GPU is crucial. GPU constantly optimizes the model in deep learning. In this sense, deep learning is more GPU-dependent. Therefore, deep learning often requires more time for training. The ResNet algorithm, for example, takes two weeks to complete a training session, whereas machine learning algorithms typically take only a few seconds.

2.1.3. Feature processing. In machine learning, features usually need to be determined by experts and then encoded. Deep learning, on the other hand, extracts features directly from data sets. For instance, convolutional neural networks attempt to learn low-level features (boundaries, lines) in the front layer, then learn partial faces, and then high-level face descriptions [6].

2.2. Deep architectures

Neural network technology was developed in the 1950s and 1960s. It has the name perceptron. There are three types of layers: hidden layers, input layers, and output layers. To get the output layer's classification result, a hidden layer will transform the input features and send to the output layer. But a single-layer perceptron is unable to deal with slightly complex functions. The multi-layer perceptron was not found until the 1980s. Rumelhart Williams was the biggest contributor to the invention. The backpropagation BP method developed by Werbos is employed in the training procedures, and the multilayer perceptron simulates the response of neurons to excitation using continuous functions like Sigmoid or Tanh. From this, We can infer that a neural network's ability to represent reality is directly correlated with the number of layers it has, with each layer using fewer neurons to accommodate increasingly complicated functions [7].

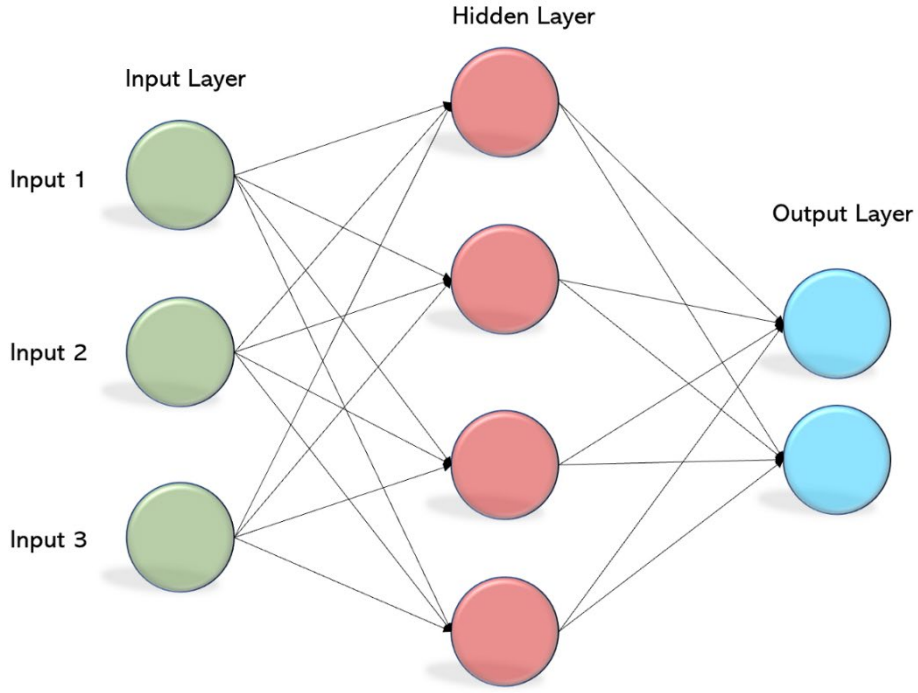

Figure 2. A neural network in which all the upper and lower neurons are connected – a multilayer perceptron.

The optimization function is more prone to fall into the local ideal solution as the layers of the neural network get deeper, and this "trap" deviates more and more from the actual global optimal solution. Deep networks perform worse than shallow networks when trained on little amounts of data. At the same time, another significant issue is that the phenomena of "gradient disappearance" gets worse as network layers increase. Particularly, sigmoid is frequently used to describe the input and output

DOI: 10.54254/2755-2721/5/20230583

functions of neurons. When BP backpropagates the gradient for a signal of magnitude 1, the gradient degrades to 0.25 of the original for each layer transmitted. When there are more layers, the gradient exponential decay practically prevents the lower layers from receiving the effective training signals.

In order to solve the local optimal solution problem, Hinton employed the pre-training method in 2006 and increased the number of hidden layers to 7 [8], which made the neural network "deep" in a real sense, thus unveiling the boom of deep learning.

At present, deep learning models are constantly developing and updating. New models emerge one after another, and we will introduce several common neural network models here.

2.2.1. Convolutional neural networks (CNN). Convolutional Neural Network (CNN) is a type of feedforward neural network that performs exceptionally well for processing large-scale images because its artificial neurons may respond to a portion of the nearby units in the coverage range [9]. It has a pooling layer and a convolutional layer.

Two sections make up the convolutional layer. The feature extraction layer is the first one. Each neuron's input is linked to the local receptive field of previous layers, from which the local characteristics are retrieved. After the local feature has been retrieved, its link to other features in terms of position is also established. Multiple feature maps make up each of the network's computational layers. All neurons have identical weights on the plane of each feature map. The activation function of the convolution network is the sigmoid function with a tiny kernel of impact function, which gives the feature map displacement invariance. Additionally, because neurons on a mapping surface share weight, there are fewer network-free parameters. A computational layer for the local average and secondary extraction follows each convolutional layer in a convolutional neural network. The feature resolution is decreased by this original two-time feature extraction structure.

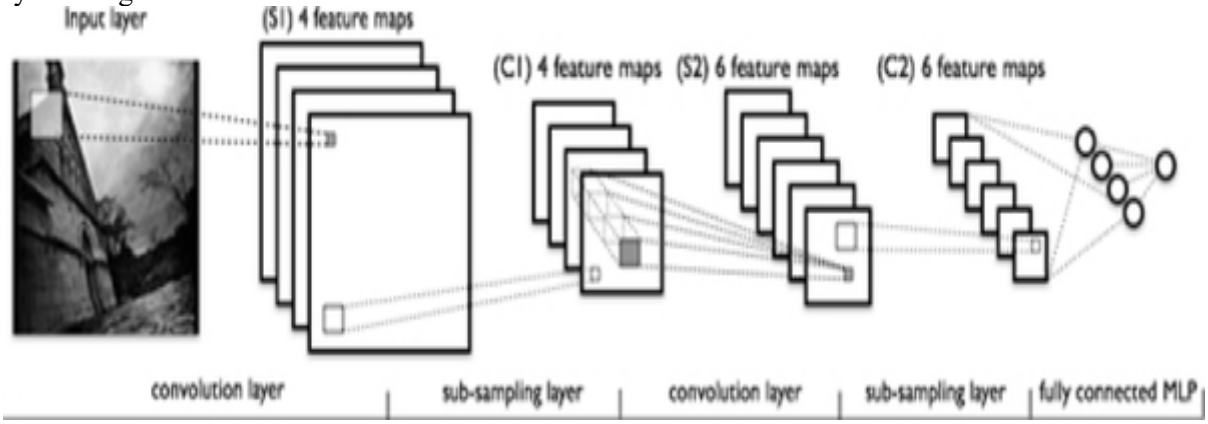

Figure 3. An illustration of reading features from the picture via CNN model.

CNN is mostly used to locate two-dimensional graphs with scaling, the variance of displacement, and other types of distortion. The pooling layer mostly performs this aspect of the function. When utilizing CNN, explicit feature extraction is avoided because the CNN feature detection layer learns from the training data. Additionally, implicit learning is done using the training data. Moreover, a major advantage of the convolutional network over the network of neurons is that because the weights of neurons on the same feature map surface are the same, the network can learn in parallel [10]. Neural network convolution weights with local Speech recognition and image processing have a distinct advantage thanks to a shared specific structure. Its design is more reminiscent of actual biological neural networks, and weight sharing makes the network less complex. In particular, the multidimensional network input vector image can directly input this feature to prevent the complexity of the reconstruction caused by data in the feature extraction and classification process.

2.2.2. Recurrent Neural Networks (RNN). Compared with the input and output of CNN, which are independent of each other, the recurrent neural network can connect the preceding and subsequent texts. For example, the prediction of the words in the sentence mentioned before is what CNN can't do [11]. In essence. RNN gives machines human-like memories.

DOI: 10.54254/2755-2721/5/20230583

RNN has input units, hidden units, and output units. The bulk of the work is done by these hidden units. As shown in figure 3, there is a one-way information flow from the input unit to the hidden unit and a second one-way flow from the hidden unit to the output unit. In some circumstances, RNNs overcome the latter limitation by performing "Back Projections," which involve sending data from output units back to hidden units. Additionally, the preceding hidden layer's state is included in the input of the hidden layer, which means that the nodes in the hidden layer may be interconnected or self-connected.

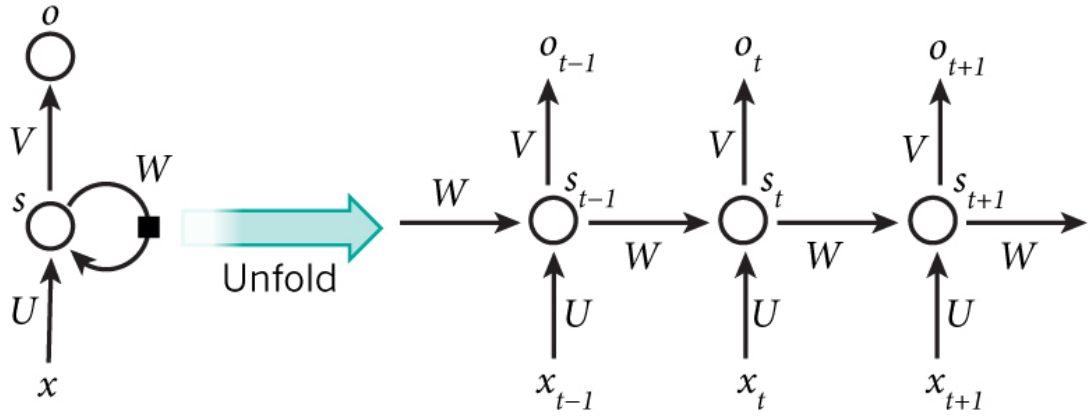

Figure 4. A typical RNN structure.

It should be noted that the processing of natural language by computer requires the processing of natural language into symbols that can be recognized by the machine, and it needs to be numerically processed in the process of machine learning. Words serve as the building blocks for understanding and processing natural language, hence it must be numerically generated, and it is a feasible and effective method to generate Word Representation [12]. What is a word vector? That is, a vector v of real numbers of a specified length is used to represent a word. Using One-hot vector to represent words is one of the simplest techniques, namely according to the number of words | V | to generate a | V | * 1 vector. When One for the other bits is zero, then this vector is on behalf of a word. World embedding will also be an important technology when we do follow-up sentiment analysis.

2.2.3. Long Short-Term Memory (LSTM). A type of temporal recurrent neural network called LSTM is suitable for digesting and forecasting significant events in time series with relatively large intervals and delays. In essence, LSTM is one kind of RNN models, although it has a more complicated structure.

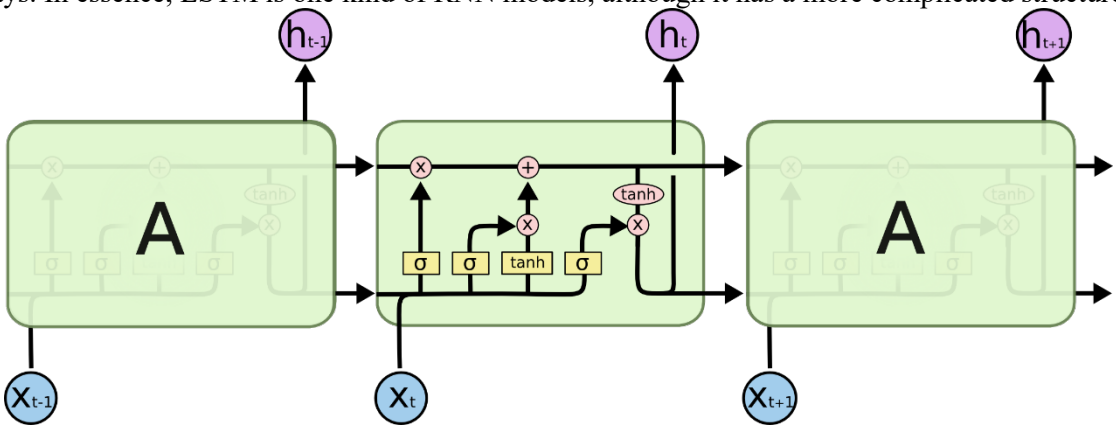

Figure 5. The structure of LSTM.

DOI: 10.54254/2755-2721/5/20230583

The primary distinction between LSTM and RNN is the addition of a "processor" to the algorithm to determine whether the information is relevant or not. A cell is the name of the processor's internal structure.

A cell contains three gates: an input gate, a forget gate, and an output gate. When a message enters the LSTM network, it may be determined by applying the rules if it is valuable or not. The forgetting door will be used to erase any conflicting information, leaving just the information that satisfies the algorithm authentication. As a result, it can retain relevant information from the prior text to affect decisions and results in the future, which is crucial for language modeling [13].

3. Sentiment analysis

Next, we will apply deep learning to the sentiment prediction of text. We used quotes from famous people, classic movie dialogue, books, etc., as the original dataset. These statements were divided into two categories, one positive and one negative. Positive ones we label as 1, and negative ones we label as 0.

We will process and import these statements in a series of steps. Finally, the model was used to predict and record the correct rate.

3.1. Processing of statements

We put positive and negative statements into two separate CSV files. Import statements through Python. We store the statements separately in an array and label each statement with a 0 and a 1. The next step is to separate the statement from the label and cut the statement.

We cut a sentence into phrases or prepositions. For example, we will cut “I need to go to the after-sell service as well.”into” “I”, ”need to”, “go to”, “the”, “after-sell service”, ”as well”, “.”. Then we split it into a training set and a test set by a self-defined “splitTrain_test” function with a ratio of 9:1. Because the computer will not be able to recognize the language directly, the next step was to transform the tokens of statements into tensor types through the torch.

Due to deep learning's ongoing self-optimization, if we traverse all the data sets once to compute the loss function, we should then compute the gradient of each function parameter and update the gradient. This method does not support online learning since every time the parameters are adjusted, it must walk over every sample in the data set. This results in high computation costs and sluggish computation speeds (This method is called Batch gradient descent [14]). On the other way, you can calculate the loss function and then assess the gradient update parameters each time you look at data (This is called stochastic gradient descent [15]). Although this approach is rather quick, it has poor convergence performance. Maybe around the optimal point, hit less than the optimal point. The goal function may oscillate sharply as a result of the two parameter updates cancelling one another.

Mini-batch gradient descent is being used as a compromise method to address the drawbacks of the two methods. This technique separates the data into multiple batches and modifies the settings for each batch. In this method, a batch of data dictates the gradient's direction collectively, making it difficult to veer off course throughout the descent. There is less randomness. On the other hand, the computational load is not too heavy because the batch contains a lot fewer samples than the entire dataset. This experiment's batch size is 64 and its epoch_num is 5.

3.2. Selection of deep learning models

After the data work is done, the next step is to build and introduce the model. In this experiment, we chose BiLSTM and a one-dimensional convolutional neural network.

The forward LSTM combines with the backward LSTM to form BiLSTM. Simply put, it is an LSTM structure with two hidden layers. In the experiment, we input forward text into the forward LSTM and reverse text into another layer. Finally, the forward and backward hidden vectors are concatenated. An output of 1 is a positive prediction, and 0 is a negative prediction.

For CNN, using one-dimensional convolution and maximum time convergence, the CNN model takes a single pre-trained word representation as input and then obtains and transforms a sequence

DOI: 10.54254/2755-2721/5/20230583

representation for downstream applications. The width, height, and quantity of channels of the input tensors for a single text sequence with n words represented by d dimensional vectors are N, 1, and d, respectively. Input is transformed into output by the CNN model. Create several one-dimensional convolution kernels, and then carry out individual convolution operations on the input. Local features between varying numbers of neighboring words can be captured by convolution kernels of various widths. Concatenate all scalar sink outputs into vectors after applying a maximum time sink layer to all output channels. The joined vector is transformed into the output category using the fully connected layer. Overfitting can be decreased via dropout. In comparison to the bidirectional LSTM model above, two embedding layers are additionally utilized, one with trainable weights and the other with fixed weights.

4. Outcome and analysis

Due to the large dataset, we cannot show the results of each batch. We take the first five batches of each epoch for comparative analysis.

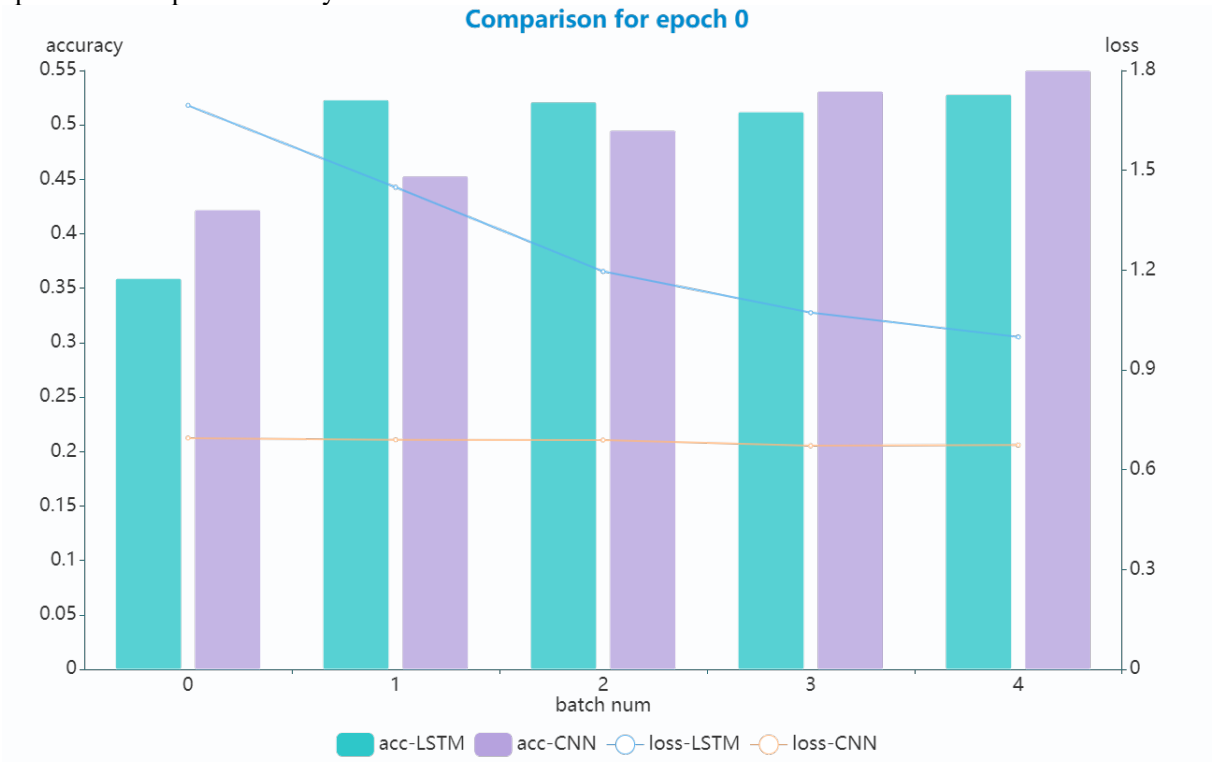

Figure 6. The result of epoch0.

We can see that both of the two deep learning models' prediction accuracy for the first epoch is not very good. The highest value of BiLSTM is 0.528 while that of CNN is 0.550. However, the loss value of CNN is much lower than that of BiLSTM. However, due to the continuous optimization of the model, the accuracy of the two models is increasing.

DOI: 10.54254/2755-2721/5/20230583

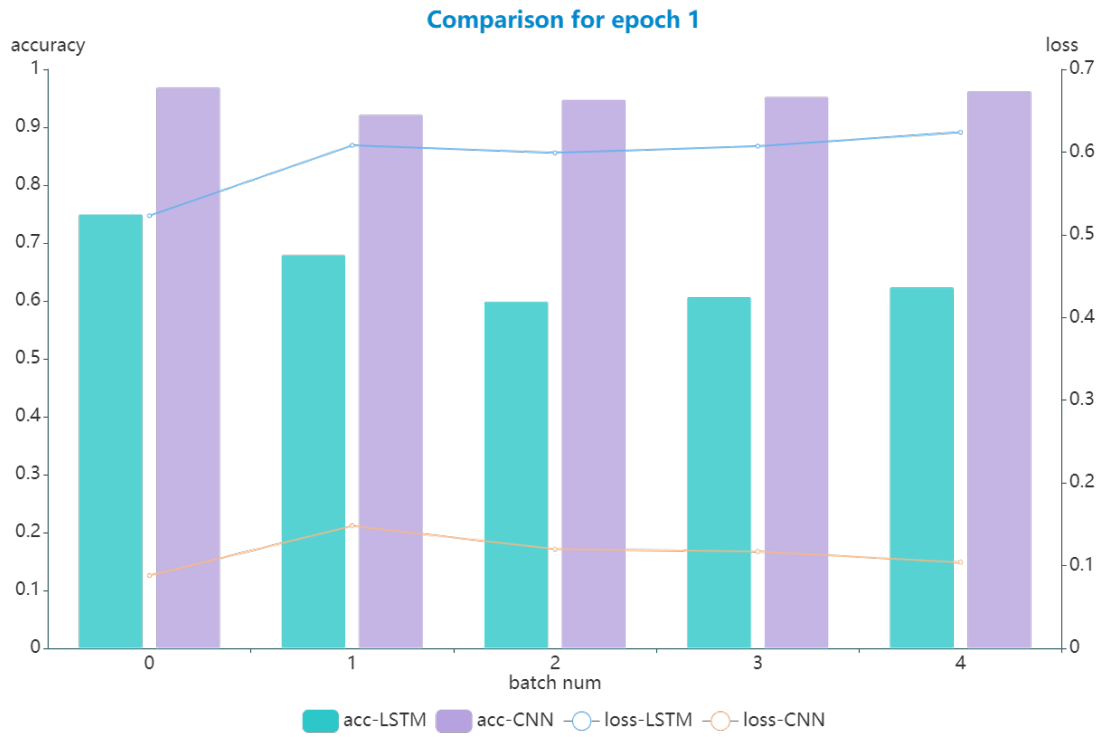

Figure 7. The result of epoch1.

After the second epoch, the correct rate of CNN changed qualitatively, reaching 0.9. At this time, BiLSTM is 0.6. Moreover, we notice that although BiLSTM has improved compared with the first epoch, it has a downward trend in the current epoch.

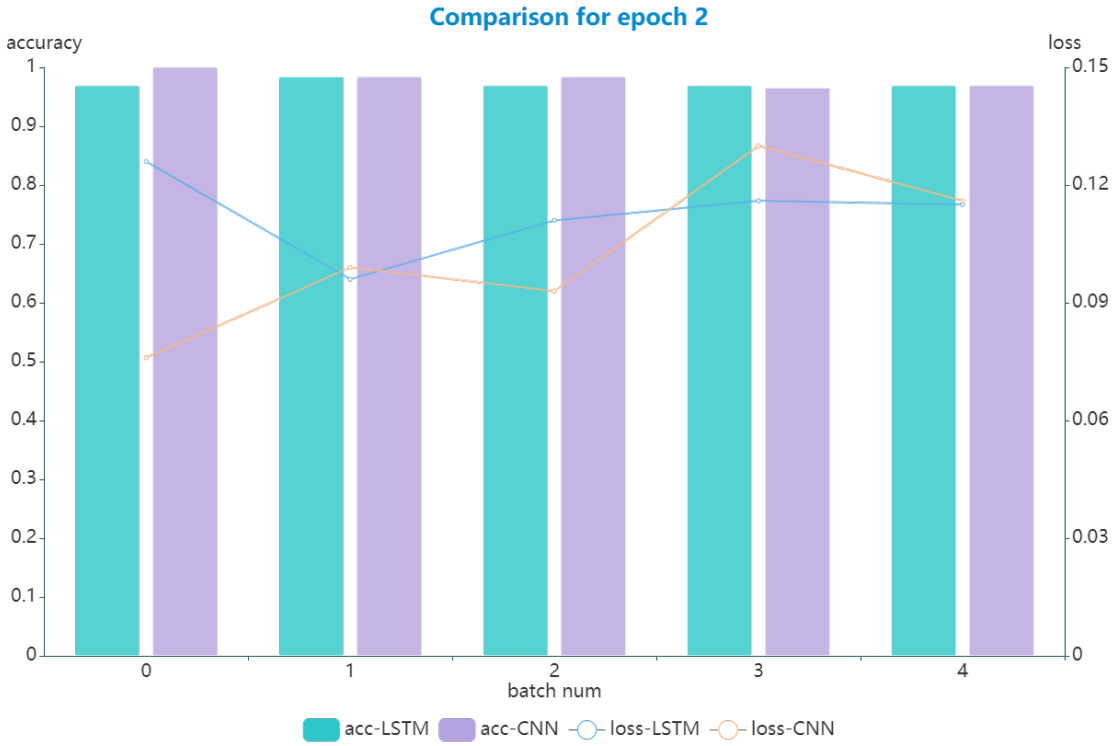

Figure 8. The result of epoch2

DOI: 10.54254/2755-2721/5/20230583

This time the situation is reversed for the last epoch. The correct rate of BiLSTM has been significantly improved. The CNN model pulled back a bit after hitting a stunning 100% in the first batch, hovering above the 0.95 range.

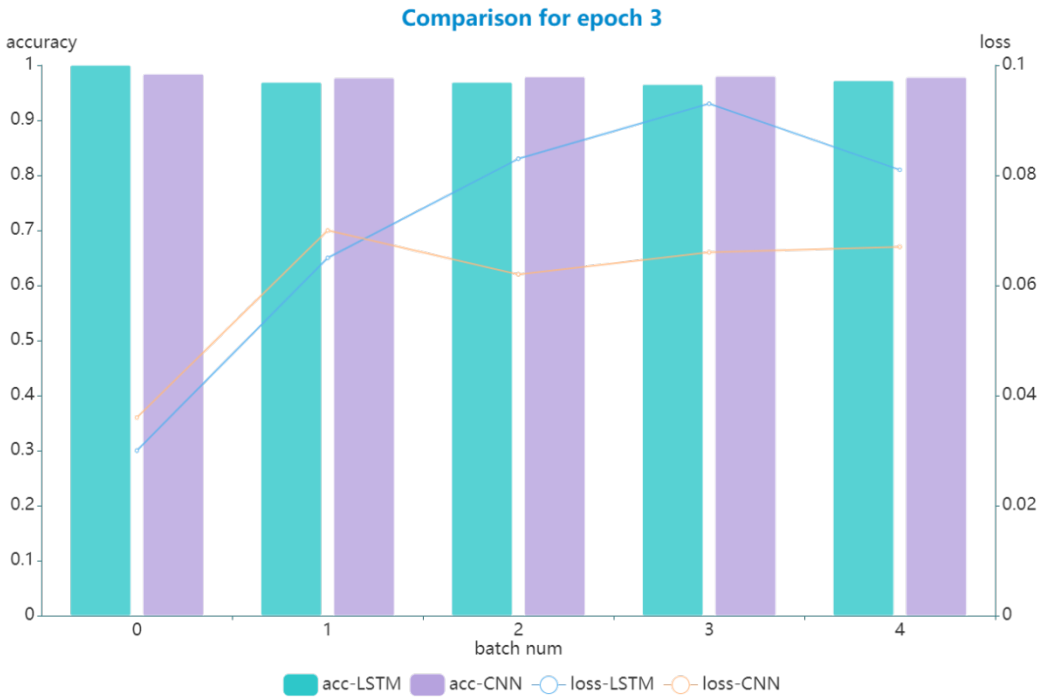

Figure 9. The result of epoch3.

History is always surprisingly similar. BiLSTM is in the same situation as CNN in the last epoch. It hit 100 percent before falling back. CNN, on the other hand, is relatively stable at around 98%.

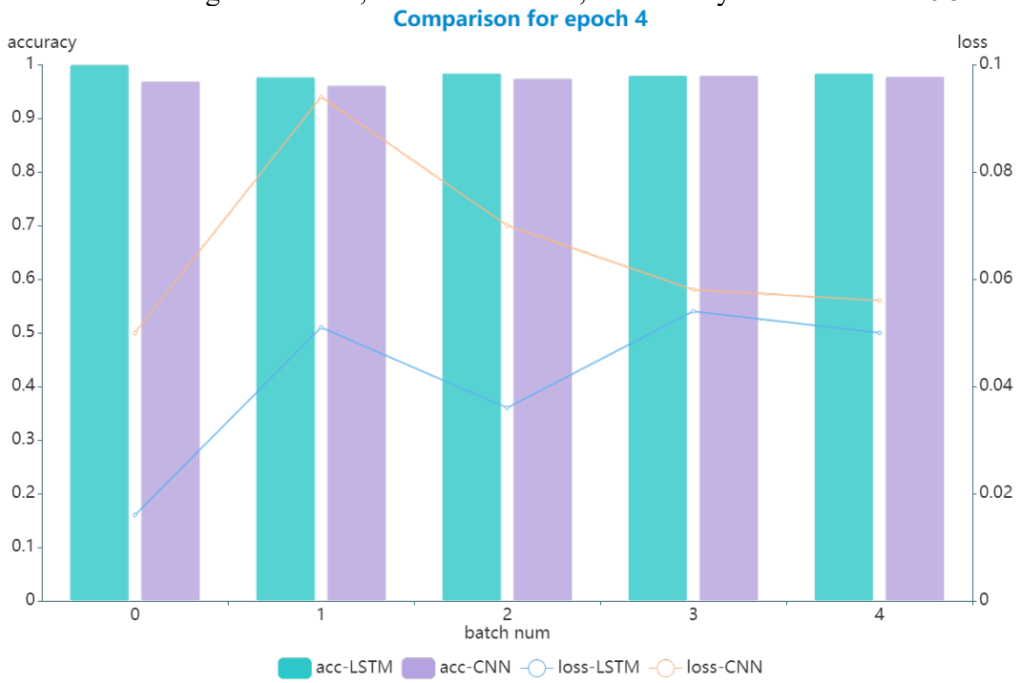

Figure 10. The result of epoch4.

DOI: 10.54254/2755-2721/5/20230583

Things didn't seem to change much in the last epoch. The accuracy of both models hovered above 95%. To sum up, BiLSTM is slightly higher than CNN.

Table 1. Comparison of BiLSTM and CNN models in average loss, accuracy of training set and testing set, training speed and training device.

neural network	avg loss	train acc	test acc	speed(examples/sec)	device
BiLSTM	0.047	0.986	0.566	66.8	CPU
CNN	0.064	0.982	0.467	368.4	CPU

Both models are trained and predicted on the CPU. In terms of time, BiLSTM trains 66.8 examples per second much slower than CNN's 368.4. In the training set, the correct rate of BiLSTM is almost equal to that of CNN. The difference between the two is only 0.04. However, BiLSTM is nearly 0.1 higher than CNN in the test set.

In general, BiLSTM lags behind the CNN model in the speed of training and optimization, but it is better than the CNN model in accuracy.

5. Conclusion

Two neural network models, CNN and LSTM, are applied to the problem of text sentiment analysis. After preprocessing the data set using Token and word embedding, training and analysis will be conducted. Although the CNN model's training speed is significantly faster than the BiLSTM model's, the LSTM model has a higher accuracy than the CNN model. The CNN model outperforms the BiLSTM model in terms of optimization. 90% accuracy can be attained by optimizing it with a modest amount of data.

Sentiment analysis is a hot topic in recent years, the text is one of the carriers of emotion. In the face of a growing number of text messages, and text contained in the feelings of great research value, how to efficiently and accurately analyze text emotion is the problem our current and future have to face, also is the direction of future development. Here is a long way to go. The application of deep learning to text sentiment analysis is a breakthrough to deal with text sentiment classification, which is also the direction of in-depth discussion and research by researchers in the future.

References

[1] Zhang, L., Tan, J., Han, D., & Zhu, H. (2017). From machine learning to deep learning: progress in machine intelligence for rational drug discovery. *Drug Discovery Today*, *22*(11), 1680–1685.

[2] Agrawal, A., Lu, J., Antol, S., Mitchell, M., Zitnick, C. L., Parikh, D., & Batra, D. (2016). VQA: Visual Question Answering: www.visualqa.org. International Journal of Computer Vision, 123(1), 4–31.

[3] Yu, Y., Si, X., Hu, C., & Zhang, J. (2019). A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures. Neural Computation, 31(7), 1235–1270.

[4] Pouyanfar, S., Sadiq, S., Yan, Y., Tian, H., Tao, Y., Reyes, M., Shyu, M.-L., Chen, S.-C., & Iyengar, S. S. (2019). A Survey on Deep Learning: Algorithms, Techniques, and Applications. ACM Computing Surveys, 51(5), 1–36.

[5] Tian, K., Shao, M., Wang, Y., Guan, J., & Zhou, S. (2016). Boosting compound-protein interaction prediction by deep learning. Methods (San Diego, Calif.), 110, 64–72.

[6] Kumar, S., & Singh, S. K. (2020). Occluded Thermal Face Recognition Using Bag of CNN (BoCNN). IEEE Signal Processing Letters, 27, 975–979.

[7] Bengio, Y. (2009). Learning deep architectures for AI. *Foundations and trends® in Machine Learning*, *2*(1), 1-127.

[8] Hinton G E, Salakhutdinov R R. Reducing the Dimensionality of Data with Neural Networks[J]. Science, 2006, 313(5786):504-507

[9] Potluri, S., Fasih, A., Vutukuru, L. K., Al Machot, F., & Kyamakya, K. (2011, July). CNN based

DOI: 10.54254/2755-2721/5/20230583

high performance computing for real time image processing on GPU. In *Proceedings of the Joint INDS'11 & ISTET'11* (pp. 1-7). IEEE.

[10] O'Shea, K., & Nash, R. (2015). An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*.

[11] Barman, P. P., & Boruah, A. (2018). A RNN based Approach for next word prediction in Assamese Phonetic Transcription. *Procedia computer science*, *143*, 117-123.

[12] Hinton G E. Learning Distributed Representations of Concepts[C]. Proceedings of the 8th Annual Conference of the Cognitive Science Society. 1986, 1: 12.

[13] Sundermeyer, M., Schlüter, R., & Ney, H. (2012). LSTM neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*.

[14] Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.

[15] Bottou, L. (2012). Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade* (pp. 421-436). Springer, Berlin, Heidelberg.