ECSE 4965/6965 Introduction to Deep Learning

Programming assignment 3

Introduction

So far, I have learned lots of network models, such as LeNet-5, GoogleNet and VGG. These modes are all designed for multi-dimensional data in which the attributes are largely independent of one another. However, certain data types such as time-series, text, and biological data contain sequential dependencies among the attributes. For these data, they might have certain order or they might have different length. Then if we use CNN cannot reach our goal. Therefore, a new method need to be found. The recurrent neural network provide an idea. It is a sequence of neural network blocks that are linked to each others like a chain. And from its name "recurrent", we can actually understand this model as multiple neural network and they pass information to next one. RNN is a good idea to handle sequences but it is also not perfect. Lots of issues in the RNN are found and they can be solved by another model called LSTM. In this project, I will combine the CNN and LSTM to train my model. The goal of this project is to classify the positions of human body. There are lots of joints on human body. Here, we simplify the condition. And I only use seven joints of human body. They are head, left wrist, right wrist, left elbow, right elbow, left shoulder and right shoulder. The dataset I use is the image extracting from the youtube video. For this particular problem, the CNN and LSTM give a good result.

Recurrent Neural Network

Traditional Neural Network usually can classify very well on statistic data. However, if the data have order or they have different length. Then the traditional Neural Network will fail. However, the RNN can address this issue. The RNN is a chain which it connect lots of units network and current one pass information to next one. The architect of the model as shown in figure (1).

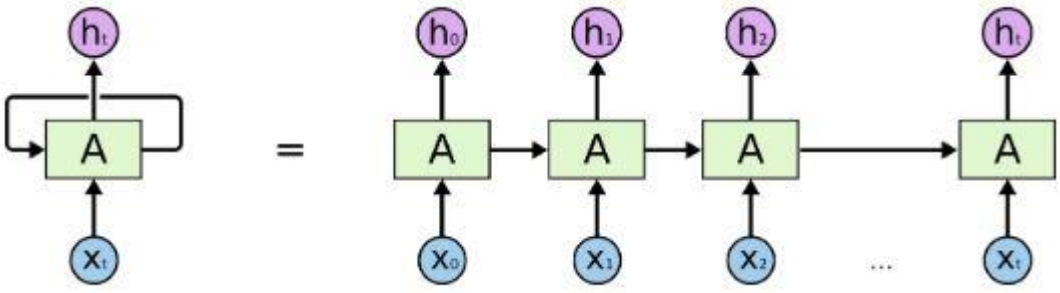

Figure (1) The structure of RNN.

For each unit network, it not only depends on the current input but also it depends on the previous state. Therefore, the function for hidden node and output are written as:

$$H_t = \tanh(W^X X_t + W^H H_{t-1} + W_0^H)$$

$$Y_t = g(W^Y H_t + W_0^Y)$$

The RNN is widely used on lots of field such as speech recognition, language modeling and translation. Although it is powerful, it still exist issues. When we build a RNN with a deep layer, the RNN usually fail. The problem is that when we train the model, we use the back propagation. And when we compute the gradients, we can find that we need to multiply the gradients of each layer. The function of back propagation chain rule as followed:

$$\nabla_{W^H} = \frac{\partial H^1}{\partial W^H}\left(\frac{\partial H^2}{\partial H^1}\frac{\partial H^3}{\partial H^2}\cdots\frac{\partial H^T}{\partial H^{T-1}}\right)\nabla H^T$$

It means that if the gradient larger than 1 or bigger than 1, then the result will too large or too small. It will cause the gradient explode or vanishing. In order to apply this model to a deep network. The LSTM is proposed and it can address this problem very well.

Long Short Term Memory

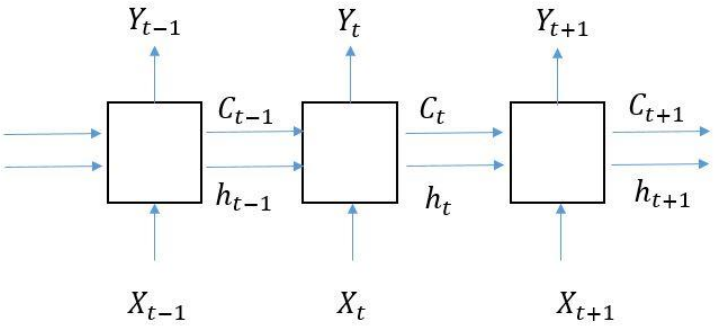

Figure (2) The structure of LSTM.

From its name, we know that the LSTM have both long memory and short memory. Compared with the RNN, the long memory can make the LSTM apply in a long length of sequence. From the structure shown in figure (2), we can see that the LSTM only add one term for the hidden nodes. However, the principle is not so easy. From the inner structure shown in figure (3) and (4), it indicates that the LSTM has a very complex inner structure.

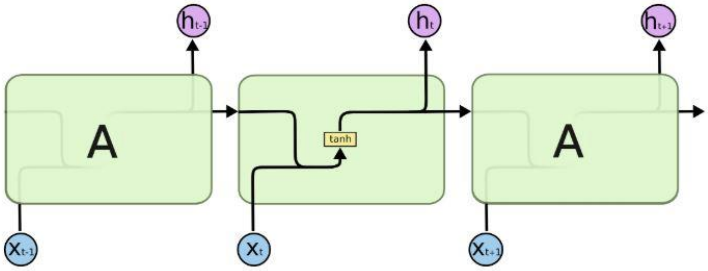

Figure (3) RNN inner structure.

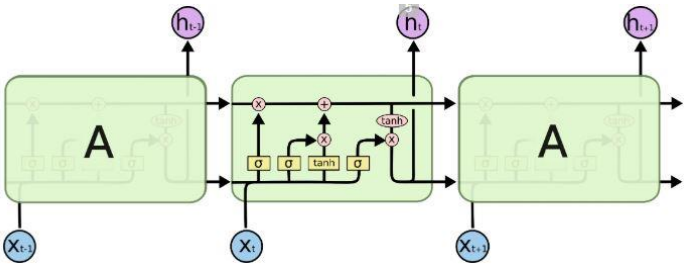

Figure (4) LSTM inner structure.

It is composed of three gates-Forget gate, Memory gate and output gate. The gate functions as followed:

$$f_t = \sigma(W^{hf} h_{t-1} + W^{xf} X_t + W_0^f)$$
$$i_t = \sigma(W^{hi} h_{t-1} + W^{xi} X_t + W_0^i)$$
$$o_t = \sigma(W^{ho} h_{t-1} + W^{xo} X_t + W_0^o)$$

These three gates are all sigmoid functions, which means that they are values from [0, 1]. The 0 and 1 control which information leave and pass. First, we need to decide which message we need to get rid of from the cell state. Thus, a forget gate show up and if it is 0, which means this message need to be thrown away.

Then, we decide which information we are going to store. The Memory gate works and for this step we need to introduce nonlinearity on current massage. Combining this two step, we get the long memory information C_t. After implementing the forget gate and memory gate, now we need to decide what we are going to output. Therefore, it is turn to output layer. It decide which part of information we want to pass to next one.

$$\widetilde{C_t} = \tanh(W^{hc} h_{t-1} + W^{xc} X_t + W_0^c)$$
$$C_t = f_t \otimes C_{t-1} \oplus i_t \otimes \widetilde{C_t}$$
$$h_t = O_t \otimes \tanh(C_t)$$

Architecture

In this project, I use CNN and LSTM to implement the classification. For the CNN part, I use three convolution layers and two pooling layers. For first two convolution layer, I use 16 5 by 5 filters and stride is one. I apply 32 3 by 3 filters for the last convolution layer. And after each convolution, I introduce the nonlinearity with ReLU function. The max pooling method is used in my CNN model. The input image is 64 by 64 three channels RGB image. When using the CNN to extract features, finally I get the fully connect layer which the length is 3872. Then these features are the input of the LSTM. In this project, I get 10 images for one video which means that the length of sequence of the LSTM is 10. Finally, I pick the mean squared error as the loss function. The CNN and LSTM structure as shown in figure (5).

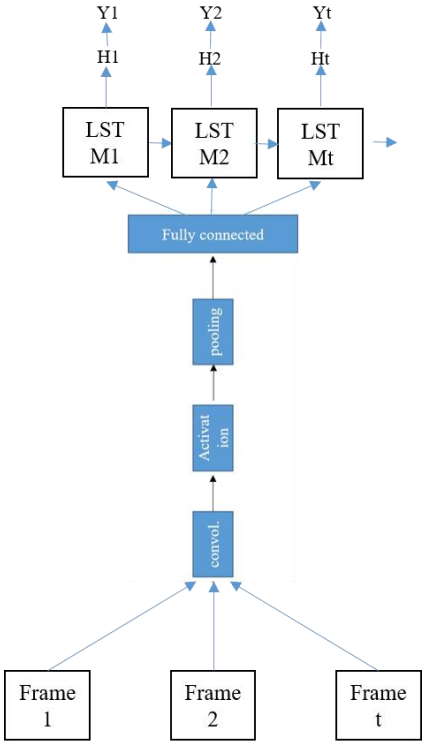

Figure (5) The architecture of model.

Result

For this project, my split data set into training data and validation data. The number of training data is 7800 and the left data for validation. The learning rate for this model is 0.0006. The numbers of nodes for each unit cell is 100. For the initialization, the weights are zero mean and 0.01 standard variance. The bias is zero.

Figure (6) is the loss curve, we can see that the loss decreases with the increase of iterations. And the difference between the training data and the validation data is small, thus, this model is fine. And from the accuracy curve for each position, we know that head, shoulders are very accurate under 10 pixels. The accuracy is over 80%. And all the positions converge at 20 pixels. It make sense because we know if the human do not move vertically, then the head and shoulders are almost fixed. From a sequence of a video, if human have different pose, then we know that the wrist and elbow move very frequently, thus, this shallow model cannot fit very well.

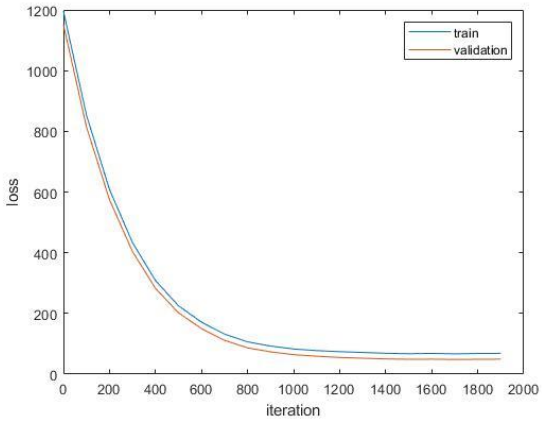

Figure (6). The loss curve for training and validation set.

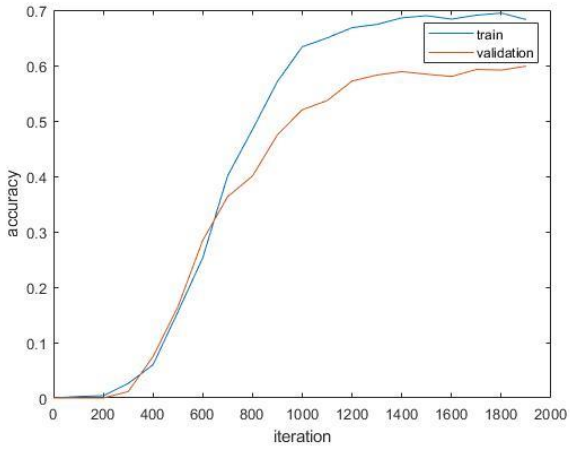

Figure (7). The accuracy curve for training and validation set.

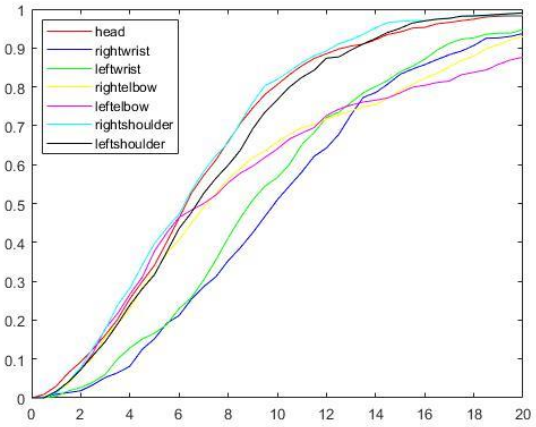

Figure (8). Prediction accuracy

From the figure (8), we see that the prediction positions are not very exact. When the distance below 5, the predictions are very terrible. I think that it might be the simple CNN structures and the little images that I use. Because the dataset is only 10 images as a sequence.

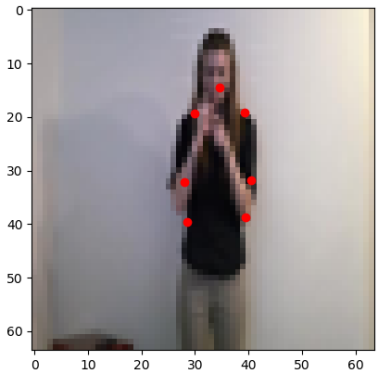

Figure (9) One example for prediction postions.

As shown in figure (7), the whole accuracy increases as the iteration increasing. The overfitting problem does not show up, which means that this model is not very complex. And the difference between training data and validation data is very small, thus, this model can be used to classify.

Conclusion

In the project, I implement CNN to extract features and using LSTM to predict the position of human body. This method provides a good result for a sequence. However, lots of parameter can influence the result. How to pick a proper parameter is the key. I found that using more filters to extra features and implement LSTM will increase the computation time obviously. And for little number of nodes in the LSTM, the speed also decrease. Thus, picking a good parameter can have a good approximation and it also improve the efficiency. Finishing this project, I have a deep understanding of the LSTM. And I know about that we can combine different methods to solve practical problem.