LSTM Approach for Emotion Classification Based on Pupillary Response

Junxian Liu

Research School of Computer Science, Australian National University
u6686392@anu.edu.au

Abstract. This paper tries to use a Long Short-term Memory neural network to solve the emotion classification problem based on humans' pupillary response. This paper tested SGD, Adadelta and Adam optimizer. For baseline models, this paper considers a multilayer perceptron with drop out layers and a feed-forward neural network trained on a pre-processed version of the same dataset. As a conclusion, LSTM network performs better than both baseline methods.

Keywords: LSTM · Emotion detection · Pupillary response

1 Introduction and Related Work

In the early years, feed-forward neural networks have been applied to different kinds of tasks. It has been proved to be a very useful tool method to solve data-driven problems. Limited by its topological structure, it cannot deal with sequence inputs that has different length. To address this problem, people invented a deep learning technique called Recurrent Neural Networks(RNN). Long Short-term Memory(LSTM) [2] is one of the most popular RNN architecture used in problems that are related to sequences. In recent years, LSTM has become an architecture that many research try when dealing with sequence input. For example, natural language processing problems.

Observing people's emotion is always a very hard topic. People nowadays realized that pupillary response reflects anger very well. Chen et al. [1] collected pupillary response of different people while they were watching different videos. And based on that data, they can distinguish if a person is really angry. Liu [3] have tried to use a feed-forward neural network to solve anger classification problem according to pupillary response information and reached an accuracy of 83%. Liu has also tried to explain what the feed-forward network has learned. However, the method in that paper is not suitable for an LSTM network because the meaning of the inputs to the fully connected layer is not understandable once the LSTM part has processed the original data. Furthermore, the implicit knowledge extracted in that way cannot deal with series inputs with different length.

For time series baseline, Wang, Yan and Oates [4] proposed three baseline networks: multilayer perceptrons baseline, fully convolutional network baseline and residual network baseline. According to their research, these three methods form a very solid baseline for time series problems.

Based on the previous works, this paper constructs an LSTM network to solve the anger classification problem using data set from Chen et al. And then compare the result with a base-line methods in Wang, Yan and Oates' work and the method used in Liu's paper.

2 Data Set, Network and Method

2.1 Data Set

This paper uses the data set collected by Chen et al. The dataset is meant to use physiological signals to classify real and acted anger. It contains pupillary response in different time step and video label to differentiate genuine anger or acted anger. The reason for using this dataset is that it collected a sufficient amount of data from people from different parts of the world. The patterns recognized from this dataset can be considered universal for all human. Besides, it contains fewer missing data.

In this paper, the network will use the entire time series collected from both left eye and right eye. Before training, the pupillary diameter is normalized by the maximum and minimum diameter of the eye of the person. The reason for doing this is because different persons have different pupillary diameter in a natural state. If the data is not normalized, the network will not be able to figure out the implicit patterns easily.

In Liu's paper, the dataset is pre-processed, and for each person, there'are seven data which are Mean, Std, Diff1, Diff2, PCAd1, PCAd2, and Label. Among witch Mean is the mean pupillary diameter, Std is the standard division of pupillary diameter along the time series, Diff and PCA are the difference between the maximum and minimum pupillary diameter and principal component analysis result, respectively.

During the experiments, the data set is split into training, validation and testing set. Training set contains 273 patterns, validation and testing set contains 58 and 59 patterns, respectively.

2.2 Network

The topological structure of the LSTM network is pretty simple. It contains two LSTM layers, each LSTM layer contains 50 LSTM units. And the output of the second LSTM layer will be put into a fully connected layer which has two output indicates real anger and acted anger respectively. The classification result is calculated by taking the maximum of these two neurons.

The fully connected network used in Liu's work is a three-layered feed-forward network. The input layer has six neurons, the hidden layer contains 75 hidden neurons and use ReLU as activation function, the output layer use sigmoid as activation function and contains only 1 neuron.

The base-line method is a five-layered feed-forward network with 500 neurons in each hidden layer. All three hidden layers use ReLU as activation function. To prevent overfitting, this method added a dropout layer between each layer. The dropout rates are 0.1, 0.2 and 0.3 for the input layer to the hidden layer, between hidden layers, and hidden layer to output layer, respectively.

2.3 Method

Long short-term memory Long short-term memory is an RNN architecture proposed by Hochreiter and Schmidhuber. When an input sequence is too long, a normal RNN architecture will be affected more by the inputs that are close to the end. However, LSTM can 'remember' important inputs and consider them. This feature makes LSTM have a better performance than normal RNN architecture when handling longer series.

An LSTM unit contains four components: memory cell, forget gate, input gate and output gate. Memory cell stores important information this LSTM unit has seen before. This information is called cell state c_i. Cell state will become hidden state h_i when the LSTM unit decide to do output. The hidden state from other LSTM cell h_{t-1} will participate in calculations in the LSTM unit alone with the input in the current time step x_t. The result of forget, input and output gate is like a controlling signal. Forget gate controls if some information in the memory cell will be removed in the current step. Input gate decides whether to store some information to the current input into the memory cell. Similarly, output gate decides whether to access certain information when doing output. Result signal s of forget, input and output gate is calculated by Equation 1

$$s = sigmoid(W_i x_t + b_i + W_h h_{t-1} + b_h) \tag{1}$$

Where W_i and b_i is the weight and bias for input, W_h and b_h is the weight and bias for the information in the previous memory cell. Note that here for different gates the weight and bias are different.

New cell state c_t is updated using Equasion 2

$$c_t = f_t \odot c_{t-1} + i_t \odot tanh(W_g x_t + b_g + W_h h_{t-1} + b_h) \tag{2}$$

Where f_t and i_t is the output of forget gate and input gate, W_g and b_g is the weight and bias for input x_t while updating cell state, W_h and b_h is weight and bias for hidden state from previous time step. $\odot$ is Hadamard product, which is elementwise product between two matrix.

Optimizer and loss This paper considers three different optimizers:

- SGD optimizer: This is a very basic optimizer. After the network is trained on a mini-batch, the parameters will be updated using the current gradient and learning rate.
- Adadelta optimizer [5]: Adadelta is an extension of Adagrad optimizer. This optimizer will automatically change the learning rate to improve the effect of gradient descent.
- Adam optimizer [6] Adam optimizer is an RMSprop optimizer with momentum. Which can make it converge faster. And RMSprop can be seen as a Adadelta optimizer that has $\rho = 0.5$. Adam optimizer will also change the learning rate automatically.

This paper will use cross-entropy loss for all the methods. This method is widely used in all kinds of binary classification problem. Cross entropy loss is calculated using Equation 3 [8].

$$loss(x, class) = -log(\frac{exp(x[class])}{\sum_j exp(x[j])}) \tag{3}$$

Where x is a tensor in shape $(batchsize, nclasses)$, $x[class]$ is the true label, and $x[j]$ is the predicted label. Note that here $x[class]$ is in one-hot coding form and $x[j]$ is not.

Experiment settings The LSTM network and base-line network are trained and tested on the normalized series dataset. The network from Liu's work is trained and tested on the processed dataset that convert the series data into six non-series inputs.

This paper uses different optimizer and loss settings to train LSTM network. By comparing the results, this paper wants to find out the best result an LSTM network can produce.

According to Wang, Yan and Oates' paper, the base-line network use Adadelta optimizer with learning rate 0.1, $\rho = 0.95$ and $\epsilon = 1e - 8$. The model will be evaluated with cross-entropy loss.

This paper will choose the training set that has the highest average accuracy on the validation set. To make the result more reliable, the average accuracy is computed from results of a 10-fold cross-validation. Furthermore, a 95% confidence interval is also calculated according to cross-validation result to capture a most possible performance on the test set. The network will then be retrained on both the training and validation set. After the final training, this paper reports the performance of this trained network on the test set.

3 Result and Discussion

3.1 Different optimizer

Table 1. Cross validation result: 300 epoch

Optimizer	Max accuracy on validation	Min accuracy on validation	95% confidence interval
SGD	0.5254	0.4576	0.4695+/-0.0481
Adadelta	0.9153	0.8475	0.8729+/-0.0435
Adam	0.9322	0.5763	0.7864+/-0.2224

Table 1 shows the cross-validation result of different optimizers. Learning rate is initialized to make these optimizers to perform best. Adam optimizer shows the best performance in all aspects when initial learning rate is 0.001. SGD optimizer does not perform very well, the network performance is lower than a random guess. Because the problem is complicated, the gradient of the loss space may have many local minima. SGD optimizer cannot deal with local minimums efficiently and result in a very poor performance. Adadelta optimizer performances the best among these three. A possible reason is because this optimizer converges fast. At the end of the training, the loss for Adadelta optimizer keeps on shaking around a certain value. This is a symbol that the training is stuck in a local minimum.

3.2 Network prediction result

Table 2 shows the result of different fully traind networks on test set.

Table 2. Accuracy on test set

Netowrk	accuracy on test set
Baseline	0.81
Liu's work	0.83
LSTM with Adam	0.91

In final training we choose Adam optimizer. According to the previous experiment, Adam optimizer can reach a higher maximum accuracy. And 95% confident interval indicates that the result still has space for improvement because the standard deviation is larger than SGD and Adadelta optimizer.

Figure 1 shows how accuracy on training and validation set change as epoch number changes. According to the figure, 200 epoch is the best choice. However, previous test have proven that Adam optimizer can still be improved at 300 epoch. The assumption is that Adam optimizer failed to adjust the learning rate to a proper number. So during training, this paper manually changed learning rate for Adam optimizer using Equation **??**

$$lr = (1 - \alpha) * lr_{begin} + \alpha * lr_{end} \tag{4}$$

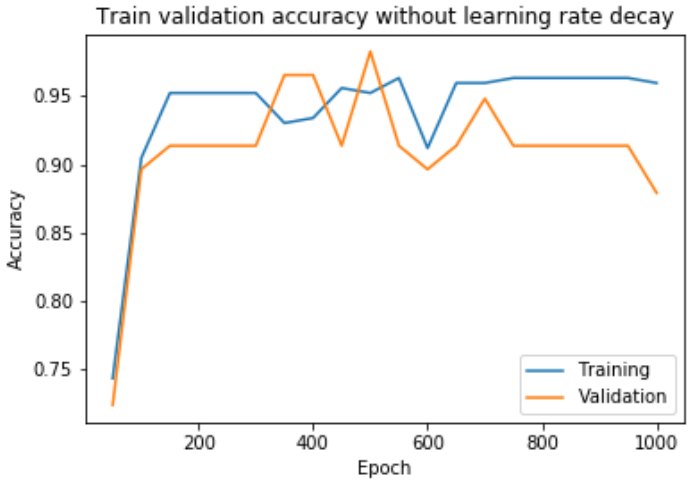

Fig. 1. Accuracy on training and validation set without manually specified learning rate.

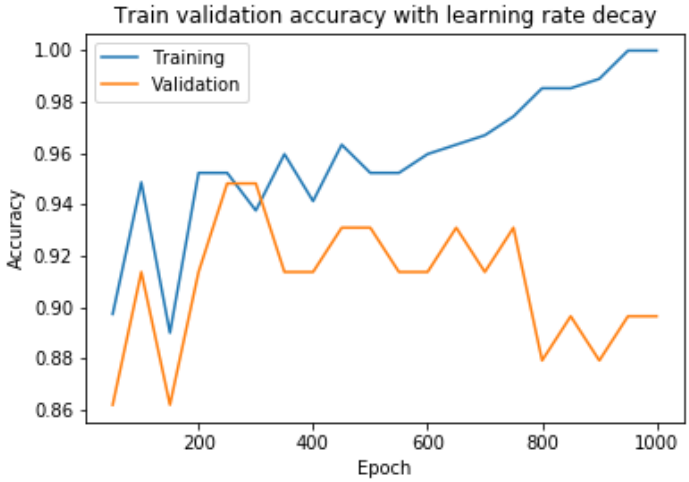

Fig. 2. Accuracy on training and validation set with manually specified learning rate.

where lr_{begin} and lr_{end} are two manually set numbers. According to experiment, $lr_{begin} = 0.001$ and $lr_{end} = 0.00001$ is the best setting. It turns out that this method do imporves the result(Figure 2).

According to Figure 2, the best epoch number is 300. By looking at Figure 3, the loss do settles at epoch 300.

4 Conclusion and future work

As a conclusion, LSTM network out-performs the baseline method and Liu's network on the pre-processed dataset. The accuracy of a fully trained two-layered LSTM network can reach 91%. However, limited by the hyper-parameters and optimizer tested in this work, the upper bound of an LSTM network on this dataset is still not clear. For future works, more optimizer and hyper-parameter settings would be a good idea. Also, attention model [7] can be a good alternative since in recent years this method is booming in different areas.

Acknowledgements

The author thanks Peiqing Yang for the helpful discussions.

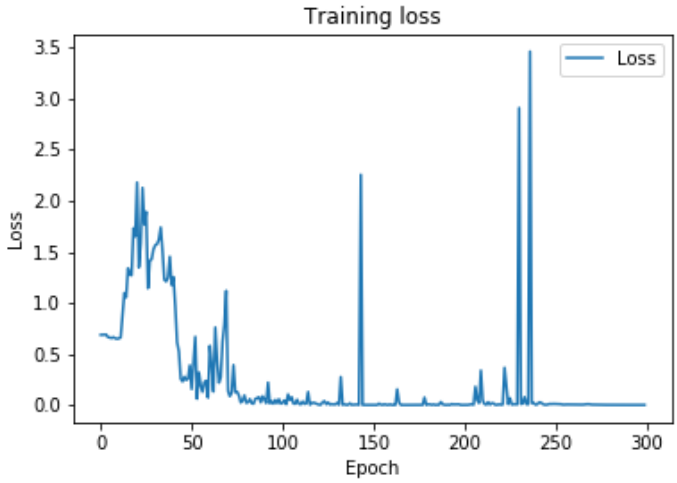

Fig. 3. Loss change with manually specified learning rate.

References

1. Chen, L., Gedeon, T., Hossain, M. Z., and Caldwell, S.: Are you really angry?: detecting emotion veracity as a proposed tool for interaction. In Proceedings of the 29th Australian Conference on Computer-Human Interaction 2017, pp. 412–416. ACM, New York, NY, USA https://doi.org/10.1145/3152771.3156147
2. Hochreiter, S., Schmidhuber, J.: LONG SHORT-TERM MEMORY. Neural Computation 1997, pp. 1735–1780
3. Liu, J.: Comparing the Ability of Interpreting Neural Networks of Characteristic Input Method and Decision Tree Method, ABCs2020 2020
4. Wang, Z., Yan, W., Oates, T.: Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline, arXiv:1611.06455, vol.4, 2016
5. Zeiler, M.: ADADELTA: AN ADAPTIVE LEARNING RATE METHOD, arXiv:1212.5701, vol.1, 2012
6. Kingma, D., Ba, J.: Adam: A Method for Stochastic Optimization, arXiv1412.6980, vol.9, 2017
7. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A., Kaiser, L., Polosukhin, I.: Attention Is All You Need, arXiv:1706.03762, vol.5, 2017
8. PyTorch api, https://pytorch.org/docs/stable/nn.html?highlight=crossentropyloss#torch.nn.CrossEntropyLoss. Last accessed 31 May 2020.