Comparison of face emotion classification among neural network, bi-directional neural network and bi-directional recurrent neural network

Yiran LI

Research School of Computer Science, Australian National University
u6920084@anu.edu.au

Abstract: Nowadays, face emotion classification has been widely used. In this study, I try to use BDNN and BRNN which may have better performance to classify facial emotion data sets, and compare the classification accuracy of these two models with that of ordinary neural network, so as to get the best result of which NN model has the best classification effect. When constructing BDNN, I use forward neural network and reverse neural network to realize the combination. The two neural networks need to update and transfer the weight. For BRNN, I use the LSTM method. After comparison, it is found that BRNN has the highest classification accuracy, but the degree of over fitting is also the highest. BDNN has slightly higher classification accuracy than NN, and the degree of over fitting of these two models is relatively low.

1. Introduction:

Nowadays, with the development of neural network and deep learning, emotion recognition for face image is applied in more and more fields. After we extract data information from facial expression images, we use neural network to classify them and get different emotions, which can be applied in human-computer interaction, lie detection, anxiety detection, pain analysis and other fields [1]. Therefore, I have a great interest in face emotion data set, and choose this data set to study several neural network technologies. In terms of technology, I chose to study the performance of bidirectional neuron network technology in face emotion classification. In order to draw a conclusion, I need to establish a common neural network and a BDNN to compare the classification effect of these two neural networks. And for the optimization of BDNN technology, I chose to use bidirectional recurrent network for optimization. Because in general, convolutional neural network is used to classify image data, but RNN also has a good effect on image classification. And BDNN is also applied to two-directional neural network for information transmission.

1.1 data set introduction:

The data set I applied is the face emotion data set, in which the data used for analysis is the SFEW data. In this paper, I learned that SFEW data set is a static facial expression database extracted from the temporary data set AFEW. It covers unconstrained facial expressions, varied head poses, large age range, different face resolutions, occlusions, varied focus and close to real world illumination. And it’s a data set that can truly reflect the real world image [1].

In this paper, the author uses the methods of local phase patterns (LPQ) and pyramid of histogram of gradients (PHOG) to extract and classify the data in the image. LPQ is based on calculating the short-term Fourier transform (STFT) on a local image window. PHOG is an extension of the HOG method which counts the appearance of the local gradient direction of the image [1].

The report also shows that JAFFE is a database of static facial expressions obtained under laboratory conditions. For seven kinds of experiments, the classification accuracy of GAFFE is 69.01% for LPQ and 86.38% for PHOG, while for SFEW, LPQ is 43.71% and 46.28% for PHOG. Therefore, we can see that for SFEW, LPQ and PHOG, the classification accuracy is very low, which may be due to some real conditions that make the image features more fuzzy [1].

In the SFEW data set I need to use, only the first 5 principal components of LPQ and PHOG are included. When using, I need to select five attributes of LPQ or five attributes of PHOG to classify it.

2. Methods:

2.1. Data Preprocessing:

Before using neural network to process data, we need to preprocess the data to eliminate some possible errors.

First of all, because LPQ and PHOG are two methods to extract eigenvalues, their values can only be used separately to classify. So I created two new tables, SFEW_ LPQ and SFEW_ PHOG, in order to facilitate the classification of these two kinds of data using neural network. SFEW_ LPQ stores first 5 principal components of LPQ and labels. The first 5 principal components of PHOG and labels are stored in SFEW_PHOG. I ignored the column of picture name.

For SFEW_PHOG, There is a row of data missing in PHOG, five data are NaN, missing data will affect the accuracy, so I filled them with 0. And Data in SFEW_LPQ does not have a null value.

And the original labels in the two tables are {1, 2, 3, 4, 5, 6, 7}. I changed them to {0, 1, 2, 3, 4, 5, 6} to prevent problems in the loss of training calculation.

Normalization is also an indispensable step in data preprocessing. Normalizing the data can prevent the attributes with large value range from having large weight in training, thus improving the performance of neural network. I want to use the min max normalization method to achieve, that is (x − min) / (max − min).

2.2. Neuron Network:

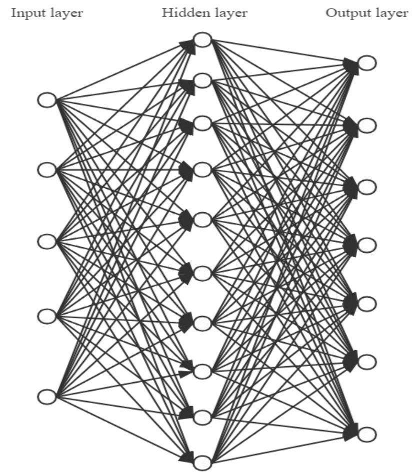
I chose three kinds of neural networks for classification. One is common neural network for comparison, one is bidirectional neural network (this is the method I choose), the other is bidirectional recurrent neural network (as the optimization method of BDNN).

For all models, the Adam optimizer and cross entropy loss are used in reverse transfer optimization, because they are effective and suitable for classification problems.

2.2.1. Common Neuron Network:

The common neural network consists of an input layer, several hidden layers and an output layer. In my neural network, I choose to use only one hidden layer to simplify the operation. The number of neurons in the input layer is equal to the number of attributes, and the number of neurons in the output layer is equal to the number of labels. For the number of neurons in the hidden layer, I tried many times to find out the parameters with better classification effect. In the process of trying, I found that when the number of neurons in the hidden layer exceeds 15, the over fitting degree of the model will be stronger, and when the number of neurons in the hidden layer is less than 10, the classification accuracy will be too low, so I chose n = 12 which has a good performance as the number of neurons in the hidden layer.

The common neural network model is shown as follows:

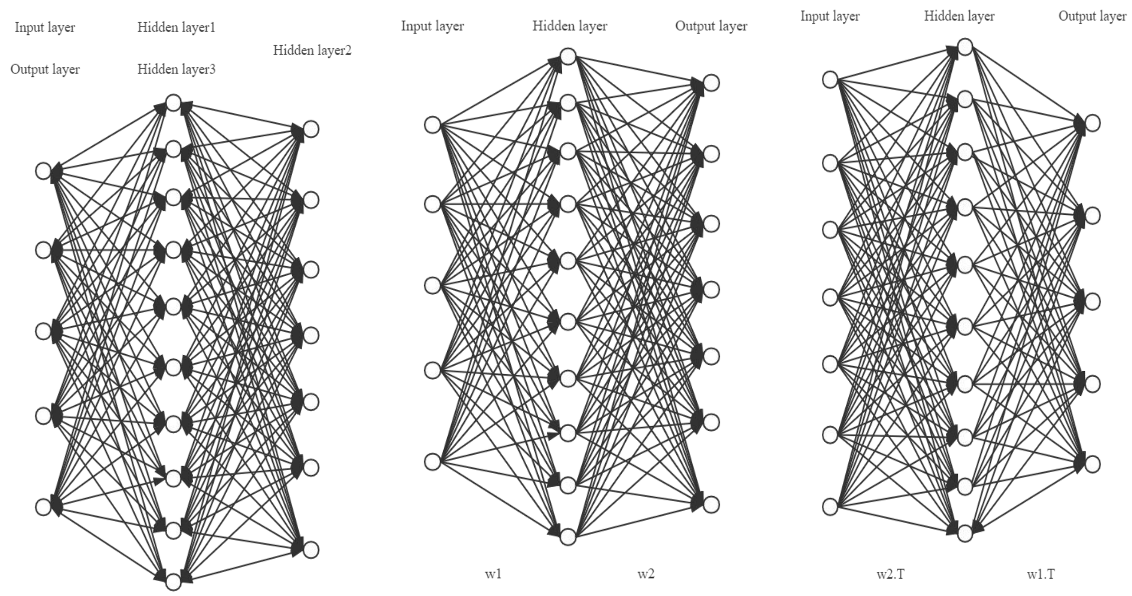


2.2.2. Bidirectional Neuron Network:

Bidirectional neural network is to optimize the neural network by using error back propagation technology in both forward and reverse directions, and adjust the weight matrix of the network in both directions. Therefore, compared with the general neural network, the bidirectional neural network does not need to use more hidden layer neurons, which can reduce the network generalization error to a certain extent [2].

Generally, bidirectional neural network requires the same number of input neurons and output neurons. For our data set, we need to build two neural networks to realize bidirectional neural network.

We need two common neural networks to represent the forward neural network and the reverse neural network respectively. The input of the forward neural network is five attribute data, the output is seven categories, the input of the reverse neural network is seven categories, and the output is five columns of data. The output of the forward neural network is directly passed in as the input of the reverse neural network. So the input and output of the bidirectional neural network are all five columns of data. But the classification result is actually the output of the forward neural network, and the calculation accuracy is also calculated by it.



When using error back propagation to optimize, two neural networks need to exchange weight information[3]. After an optimization update of the forward neural network, the weight matrix of input layer to hidden layer of the forward neural network is transposed to the weight of hidden layer to output layer of the reverse neural network. And the weight matrix of hidden layer to output layer of the forward neural network is transposed to the weight of input layer to hidden layer of the reverse neural network. Then, after the reverse neural network is updated, the two weight matrices are passed to the forward neural network as well. In this way, the two neural networks update and transfer weights in turn, and then the whole bidirectional neural network can be optimized. It is assumed that the two weight matrices of the forward neural network are W11 and W12, and the two weight matrices of the reverse neural network are W21 and W22. So the process is: update the forward neural network, W22 = W11. T and W21 = W12. T, and then update the reverse neural network, W11 = W22. T and W12 = W21. T.

In addition, for the reverse neural network, the activation function also needs to become the inverse function of the corresponding function in the forward neural network, because the input and output of the activation function are completely reversed compared with the forward neural network. Because I can't implement the inverse function of sigmoid function or softmax function, and the relu function has no inverse function (corresponding to multiple values at the place equal to 0), I directly omit the activation function in the neural network, and use the linear function as the activation function.

In the optimization of the reverse neural network, the loss gradient used is still the loss of the forward neural network, because the output of the reverse neural network does not represent the loss size.

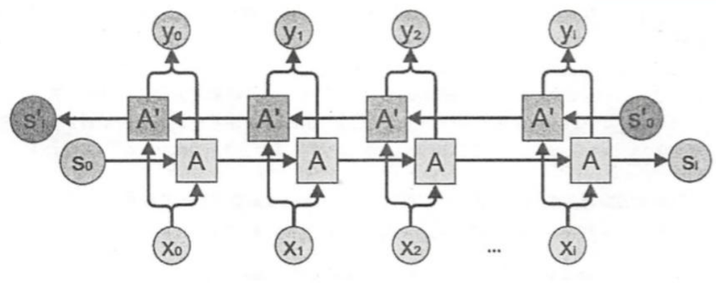
2.2.3. Bidirectional Recurrent Neuron Network:

The recurrent neural network (RNN) is a kind of recurrent neural network which takes sequence data as input, recursion in the evolution direction of sequence and all nodes are linked by chain[4].

BRNN is composed of two RNN models which are superposed up and down. The output is determined by the state of the two RNN models. For each time t, the input will be provided to two RNN with opposite directions at the same time, and the output is determined by the two unidirectional RNN.[5].

It also contains two hidden layers, and one for the forward pass and the other for the reverse pass. Both these layers are used when optimizing the model. And compared with bidirectional neural network, it has cyclic information transmission. So it may have the better accuracy for reducing transmission error.

In my model, I use the LSTM to build BRNN. And I use the number of hidden neurons for each layer as n=6, as it has two hidden layers, so that it will have the same hidden neurons number, which make the result more fair.



3. Results and Discussion:

3.1. Results:

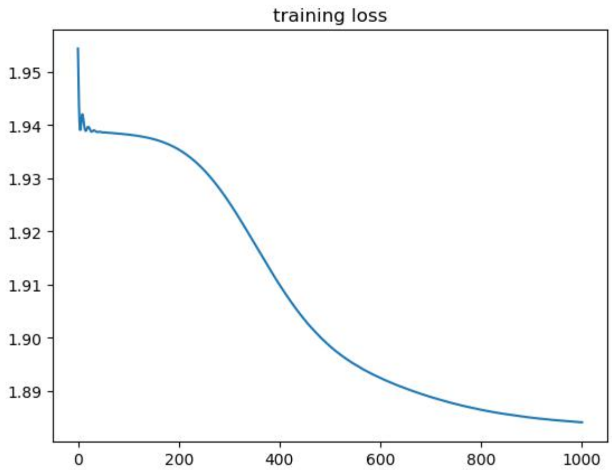
For each model, loss and accuracy and the confusion matrix are applied to evaluate its classification effect. I use 1000 epochs in the code to train and print the training results six times in the training process to observe the change of classification effect in the training process. I use 500 data as training data and others as testing data.

3.1.1. NN:

```
Epoch [1/1001] Loss: 1.9544  Accuracy: 17.00 %
Epoch [201/1001] Loss: 1.9354  Accuracy: 19.40 %
Epoch [401/1001] Loss: 1.9100  Accuracy: 20.40 %
Epoch [601/1001] Loss: 1.8925  Accuracy: 21.40 %
Epoch [801/1001] Loss: 1.8865  Accuracy: 21.80 %
Epoch [1001/1001] Loss: 1.8841  Accuracy: 24.00 %

Confusion matrix for training:
[[38.  3.  1.  5. 18. 14.  4.]
 [16.  9.  4.  1. 16.  8.  2.]
 [29.  5.  4.  5. 16.  4.  7.]
 [27.  3.  3.  5. 13. 10.  5.]
 [18.  4.  3.  5. 36. 11.  4.]
 [19.  7.  2.  5. 19. 15.  5.]
 [21.  3.  3.  3. 17. 12. 13.]]
Testing Accuracy: 18.29 %

Confusion matrix for testing:
[[ 7.  3.  1.  1.  3.  1.  1.]
 [ 3.  5.  2.  2.  3.  2.  2.]
 [ 8.  1.  3.  2.  7.  5.  4.]
 [10.  2.  2.  4.  7.  6.  3.]
 [ 4.  1.  1.  0.  8.  3.  2.]
 [12.  2.  0.  0. 11.  3.  0.]
 [12.  2.  2.  0.  6.  4.  2.]]
```



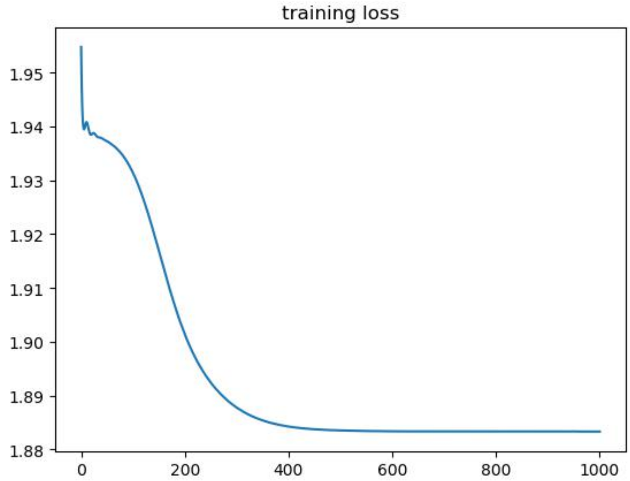
We can see that the training accuracy is 24% and the test accuracy is 18.29%. Training accuracy is higher than test accuracy, so it’s kind of over fitting. And the accuracy is not so good. For these two confusion matrix, we can see that each label has been predicted wrongly. And it predict most data in three label that have more probability. From the plot, we can see that it has some volatility in a small epoch range. And when the loss is nearly constant,the loss is still large. This model doesn’t have a good performance.

3.1.2. BDNN:

```
Epoch [1/1001] Loss: 1.9547  Accuracy: 14.40 %
Epoch [201/1001] Loss: 1.9014  Accuracy: 19.80 %
Epoch [401/1001] Loss: 1.8843  Accuracy: 23.00 %
Epoch [601/1001] Loss: 1.8834  Accuracy: 23.00 %
Epoch [801/1001] Loss: 1.8833  Accuracy: 22.80 %
Epoch [1001/1001] Loss: 1.8833  Accuracy: 23.00 %

Confusion matrix for training:
[[34.  4.  1.  5. 17. 14.  8.]
 [15.  9.  1.  1. 16.  9.  5.]
 [30.  4.  1.  5. 16.  4. 10.]
 [25.  2.  2.  6. 15. 10.  6.]
 [14.  6.  2.  8. 36. 10.  5.]
 [18.  6.  1.  4. 20. 16.  7.]
 [20.  3.  3.  2. 16. 15. 13.]]
Testing Accuracy: 20.57 %

Confusion matrix for testing:
[[ 7.  3.  0.  0.  3.  1.  3.]
 [ 2.  5.  1.  2.  3.  2.  4.]
 [ 7.  1.  0.  2.  8.  4.  8.]
 [ 8.  2.  1.  6.  8.  5.  4.]
 [ 4.  1.  1.  0.  9.  2.  2.]
 [11.  2.  0.  0. 10.  3.  2.]
 [11.  2.  0.  0.  5.  4.  6.]]
```



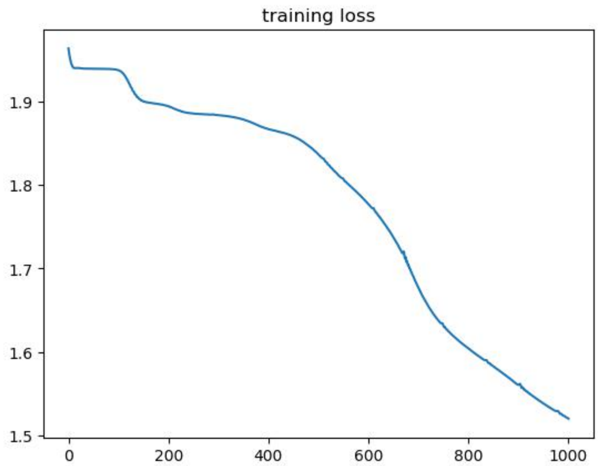
We can see that the training accuracy is 23% and the test accuracy is 20.57%. For BDNN model, the training data is slightly higher than test accuracy. So it is not that over fitting. And it’s accuracy is higher than NN model. But it’s not a stable result. Actually, it has a worse accuracy than NN sometimes. Also, it is tend to be more likely to over fitting than NN in several times attempts. And it have a better accuracy than NN in most of attempts. From the confusion matrix, we can also see that it predicted wrong in each label. And it tend to predict many data as label 0. From the plot, we can see that it has some volatility in a small epoch range. And when the loss is nearly constant,the loss is still large.

3.1.3. BRNN:

```
Epoch [1/1001] Loss: 1.9632  Accuracy: 11.20 %
Epoch [201/1001] Loss: 1.8941  Accuracy: 21.00 %
Epoch [401/1001] Loss: 1.8665  Accuracy: 22.00 %
Epoch [601/1001] Loss: 1.7767  Accuracy: 31.00 %
Epoch [801/1001] Loss: 1.6040  Accuracy: 37.00 %
Epoch [1001/1001] Loss: 1.5201  Accuracy: 40.60 %

Confusion matrix for training:
[[31.  7.  7. 13.  5. 11.  9.]
 [11.  7.  9.  6.  7.  8.  8.]
 [ 4.  4. 38.  6.  6.  8.  4.]
 [10.  3.  8. 28.  6.  8.  3.]
 [11.  6. 14.  5. 30.  7.  8.]
 [ 8.  3.  8.  5.  7. 37.  4.]
 [10.  6.  5.  5. 10.  4. 32.]]
Testing Accuracy: 25.14 %

Confusion matrix for testing:
[[6. 1. 3. 0. 3. 1. 3.]
 [1. 2. 2. 7. 2. 2. 3.]
 [1. 3. 8. 4. 7. 2. 5.]
 [8. 1. 6. 8. 2. 6. 3.]
 [1. 1. 2. 3. 4. 2. 6.]
 [4. 2. 3. 3. 3. 7. 6.]
 [2. 1. 7. 3. 3. 3. 9.]]
```



For BRNN, the training accuracy is 40.68% and the test accuracy is 25.14%. It has a very high training accuracy and a good test accuracy. Most of the time, it has the best accuracy in these three models. But it's also the easiest model to over fit. Its training accuracy is always very high and is much more higher than the testing accuracy, so it always has a very high degree of over fitting. From the confusion matrix, we can also see that it predicted wrong in each label. And it tend to predict many data as label 0. From the plot, we can see that the loss is still decreasing when there are 1000 epoch. So it will fit the training data very well.

3.2. Discussion：

Because BDNN is equivalent to a common neural network plus a reverse neural network, which optimizes and updates the weights together, BDNN will have better performance and higher accuracy than NN in theory. And because of the increase of the reverse neural network, the whole neural network will have less possibility to have some network generalization errors, so the accuracy will be higher, and sometimes the over fitting degree of training will be lower than NN. However, in some cases, the wrong changes can not offset the influence of over fitting brought by one layer network, so sometimes the degree of over fitting is higher than NN. For BRNN, RNN can retain some information of the data before training, so it can have better classification effect in image processing. BRNN is equivalent to the optimized version of BDNN, so it generally has the best performance. But at the same time, BRNN training time will be relatively long. And because of the storage and transmission of information, it is more likely to lead to the problem of over fitting. It is the most serious over fitting problem among the three models.

And because the classification accuracy in the report of data set is not high, the classification accuracy of these three models is very low. Moreover, because the data set is too small, the model can not be trained better, and the performance is not good enough, even has certain contingency when comparing the results.

4. Conclusion and Future Work:

4.1. Conclusion:

In this study, NN, BDNN and BRNN models are constructed and applied to face emotion data set classification. In the comparison of the results, we find that BDNN optimizes the accuracy of NN to a certain extent by adding a layer of reverse neural network. And the bi-directional neural network can speed up the training efficiency and improve the performance of NN by alternating the positive and negative directions. BRNN is also a kind of neural network suitable for image classification. On the basis of BDNN, it changes the neural network into recurrent neural network. By retaining more information of pre order, the accuracy can be greatly improved, but at the same time, it also faces a more serious over fitting problem.

4.2. Future work:

(1)We can study the influence of the changes of various hyperparameters in the three models on the model performance, and find out the better choice of hyperparameters.
(2)Study the data set more deeply, understand the method of getting the eigenvalue data, extract the eigenvalue data on the more complex data set and then apply it to the model.
(3)Learn how to implement the inverse functions of those activation functions to realize a more complex BDNN model.

Reference

[1] Dhall A, Goecke R, Lucey S, et al. Static facial expressions in tough conditions: Data, evaluation protocol and benchmark[C]//1st IEEE International Workshop on Benchmarking Facial Image Analysis Technologies BeFIT, ICCV2011. 2011.

[2] Nejad A F, Gedeon T D. Bidirectional neural networks and class prototypes[C]//Proceedings of ICNN'95-International Conference on Neural Networks. IEEE, 1995, 3: 1322-1327.

[3] Siwen Ji, Shengye Yan, Meng Wang. Image classification algorithm based on bidirectional neural network [J]. Computer engineering and design, 2018,39 (10): 3113-3117

[4] Yuanhang Su,C.-C. Jay Kuo. On extended long short-term memory and dependent bidirectional recurrent neural network[J]. Neurocomputing,2019,356.

[5] Blog.csdn.net . 2020. [principles of AI mathematics] probabilistic machine learning (IV): an example of Tan algorithm based on semi naive Bayes_ Artificial intelligence_ [Online] available at:< https://blog.csdn.net/leviopku/article/details/80206809 >

[6] Fengnian Zhao,Zhiming Ruan,Zongyu Yue,David L.S. Hung,Sibendu Som,Min Xu. Time-sequenced flow field prediction in an optical spark-ignition direct-injection engine using bidirectional recurrent neural network (bi-RNN) with long short-term memory[J]. Applied Thermal Engineering,2020,173.