Neutral network to judge stress face with compression and Characteristic Input Method

Zhaoming Wang(u6623146)

Abstraction

In this report, I will show the methods I have tried to train a neutral network to classify the face is stressful or calm.

In the paper Thermal Super-Pixels for Bimodal Stress Recognition [1], the authors used SVM to do the work, and get an accuracy about 95%.

The purpose of this report is to give the ways I tried to achieve the similar accuracy in paper Thermal Super-Pixels for Bimodal Stress Recognition [1]. And how I used the technique in progressive image compression [3] and the technique used in the paper Explaining student grades predicted by a neural network [4] to improve the efficiency of training and avoid the overfitting.

Introduction

In the paper Thermal Super-Pixels for Bimodal Stress Recognition [1]. The authors try to analyze the face is stressful or calm based on two images, RGB image and thermal image. In this paper, it uses VJ face detection algorithm to get the data in RGB images and applying the LSC super-pixel algorithm on the thermal images to get the thermal data. Then the use SVM on both data, and filtering them. Finally, they use the following equation:

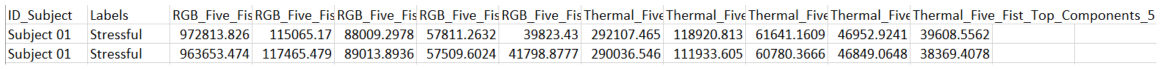
$$S_{Modal} = \tanh(\gamma .(\omega_1 * S_{RGB} + \omega_2 * S_T + Threshold))$$ [1].

To fuse the both data.

This topic is quite interesting. Nowadays more and more people are feeling stressful, so if we can see the stress on one’s face, it will be easier to make a contact-free monitoring, [1] and make relevant reaction.

Based on this paper, I wondered whether I can achieve the similar result by training a NN. This is obvious that the stress recognition is a classification problem. So I trained a classification NN to judge the face is calm or stressful using the data set provided by ANUStressDB.

In the process of training, in order to improve the accuracy of the NN, I implement two method used in paper Progressive image compression [3] and Explaining student grades predicted by a neural network [4] and successfully improve the test accuracy to about 60% which is better than guess.



ID_Subject	Labels	RGB_Five_Fis	RGB_Five_Fis	RGB_Five_Fis	RGB_Five_Fis	RGB_Five_Fis	Thermal_Five	Thermal_Five	Thermal_Five	Thermal_Five	Thermal_Five_Fist_Top_Components_5
Subject 01	Stressful	972813.826	115065.17	88009.2978	57811.2632	39823.43	292107.465	118920.813	61641.1609	46952.9241	39608.5562
Subject 01	Stressful	963653.474	117465.479	89013.8936	57509.6024	41798.8777	290036.546	111933.605	60780.3666	46849.0648	38369.4078

Image 1: two exmaple data from dataset from ANUStressDB

Methods

I. Basic Idea

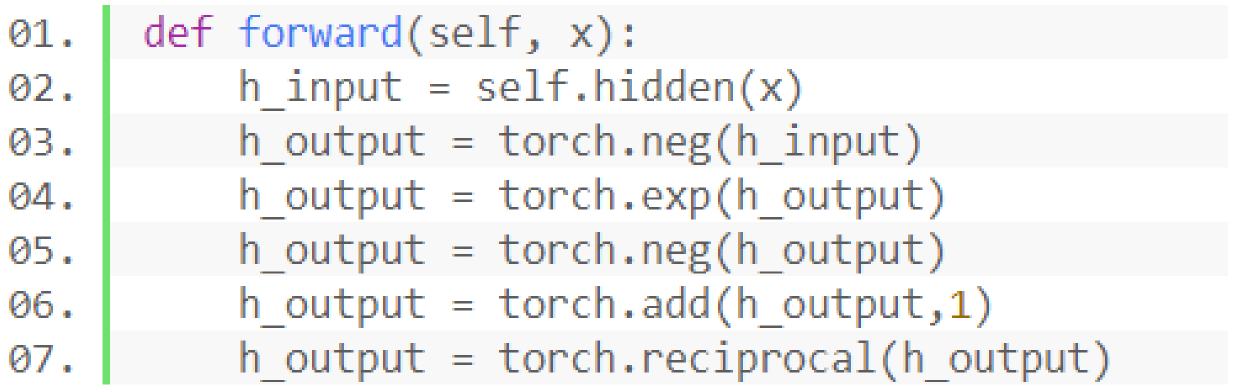
The dataset provided by ANUStressDB has ten float data for each state of each subject (image 1). This is obviously the binary classification. So, I use the basic binary classification method.
In this method, I first divided the dataset into training set and test set at the rate of 8:2. Then, I defined a customized neural network structure with 1 hidden layer and used sigmoid as the activation function. I also used Cross Entropy as loss function.

II. Multiple optimizer [2]

Base on the search on google to improve the training accuracy, the first way is to change the optimizer. Therefore, I tried to change the optimizer, for example Adadelta, Adagrad, Adam, SDG. After compared the result of each optimizer, I finally choose the SDG as my optimizer.
Then, I tried to change the study rate, number of hidden neurons and the number of epochs, and defined the study rate at 0.0095, and the number of hidden neurons as 900.

III. Using the technique mentioned in the paper Progressive image compression [3]

In the paper Progressive image compression [3], it set a model use basic logistic activation function $y=(1-e^{-x})^{-1}$. So, I change the forward function in the following way:



```python
def forward(self, x):
    h_input = self.hidden(x)
    h_output = torch.neg(h_input)
    h_output = torch.exp(h_output)
    h_output = torch.neg(h_output)
    h_output = torch.add(h_output,1)
    h_output = torch.reciprocal(h_output)
```

Based on the paper [3], we can produce a network with one fewer unit which requires no further training by dropping one of paired vectors that angle is less than 15 degree or larger than 165 degree. In order to achieve this, I add this section of code in the forward function to achieve the same effect as the nn.dropout function does:

```
tensor_list = y_pred.tolist()
        for v1 in tensor_list:
            for v2 in tensor_list:
                if (tensor_list.index(v1) >= tensor_list.index(v2)):
                    continue
                #check the angle
                l1 = np.sqrt(v1.dot(v1))
                l2 = np.sqrt(v2.dot(v2))
                cos_v1v2 = v1.dot(v2)/(l1*l2)
                angle = np.arccos(cos_v1v2)
                angle = angle*360/2/np.pi
                if (angle <= 15 or angle >=165):
                    tensor_list.remove(v2)
        y_pred = torch.Tensor(tensor_list).float()
```

This section uses vectors to get angle [3]. And drop the vector so that the model can train faster with more hidden neurons.

IV. Using the technique mentioned in the paper Explaining student grades predicted by a neural network [4]

In the paper Explaining student grades predicted by a neural network [4], it divided the training set with some certain rules. So, in the test part, the NN model can first filter the input by their characteristic pattern and give the network's next most likely output. So, I divided the training set and the test set in the following way:

```
train_data1 = train_data[train_data["RGB_Five_Fist_Top_Components_1"]>1000000]
train_data2 = train_data[train_data["RGB_Five_Fist_Top_Components_1"]<=1000000]
test_data1 = train_data[train_data["RGB_Five_Fist_Top_Components_1"]>1000000]
test_data2 = train_data[train_data["RGB_Five_Fist_Top_Components_1"]<=1000000]
```

Results and Discussion

The result is shown in the following table:

The method used in the code	The train accuracy	The loss	The test accuracy
I	51.09%	0.69	38.26%
I and II	55.45%	0.68	41.22%
I, II and III	61.39%	0.66	48.70%
All the method	68.25%	0.58	63.53%
The code in paper Thermal Super-Pixels for Bimodal Stress Recognition [1].	None	None	89%

As is shown in the table the method II improved the accuracy very little. That’s because the model I trained is still overfitting. So, if I can’t find a better optimizer, this problem will still be there.

As for the method III, we can come to the conclusion that this method is useful. As the pruning is applied, I can improve the hidden layers and the number of hidden neurons without using too much time. However, since the model still hasn’t chosen the best structure. So, the improvement is not enough. Therefore, the technique used in part III of methods improve the efficiency and decrease the overfitting when the hidden neurons number is much high. So, this will also work in the future.

As for the method IV, we can find that the improvement is huge. My analysis is that the face of different people may show quite differently when they are stressful. This significantly block the training process and make it harder for a NN to be well-trained. Therefore, divided the training set and test set base on their characteristic pattern will greatly help the training and testing process. However, the accuracy is still far less than that shown in the paper Thermal Super-Pixels for Bimodal Stress Recognition [1]. I think it is because the division is still too rough. If I can find a better way to divide base on the characteristic pattern of different subject in the dataset, the accuracy may improve further.

Conclusion and Future Work

My model in this report doesn’t perform well, but I tried lots method to improve it. And by the result, though the test accuracy is still far less than the result in the paper Thermal Super-Pixels for Bimodal Stress Recognition [1]., it still shows some improvement in the training and test accuracy.

So, the first priority in my future work is to find the suitable NN definition, for example the optimizer to make avoid the overfitting problem. Or trying to do some other pre-operation on the data set.

Then is to find another way to pruning the training step and try to do the same job in the test part. Each method is aim to improve the efficiency of the NN model, since the current efficiency is not high enough.

Reference

[1] R. Irani, K. Nasrollahi, A. Dhall, T. B. Moeslund and T. Gedeon, "Thermal super-pixels for bimodal stress recognition," 2016 Sixth International Conference on Image Processing Theory, Tools and Applications (IPTA), 2016, pp. 1-6, doi: 10.1109/IPTA.2016.7821002.

[2] https://github.com/baifengbai/PYTORCH_LEARNING/tree/master/chapter6-%E5%AE%9E%E6%88%98%E6%8C%87%E5%8D%97

[3] T. D. Gedeon and D. Harris, "Progressive image compression," [Proceedings 1992] IJCNN International Joint Conference on Neural Networks, 1992, pp. 403-407 vol.4, doi: 10.1109/IJCNN.1992.227311.

[4] T. D. Gedeon and H. S. Turner, "Explaining student grades predicted by a neural network," Proceedings of 1993 International Conference on Neural Networks (IJCNN-93-Nagoya, Japan), 1993, pp. 609-612 vol.1, doi: 10.1109/IJCNN.1993.713989.