Face Emotion Classification and Recognition using Convolutional Neural Network

Chao Liu

Research School of Computer Science, Australian National University
u6293183@anu.edu.au

Abstract. Face emotion classification and recognition has been extensively studied and applied in recent years. Facial expression recognition is a very difficult and complicated problem, because different facial expressions are constantly changing. Convolutional neural network is widely used in various image classification projects because of its powerful image recognition and classification capabilities. In this report, the network structure design and parameter optimization of convolutional neural network are carried out on the PyTorch framework platform. First, we will process the data, including dividing training set and testing set. Second, build a convolutional neural network and train the network. Then we will get the correct accuracy of the network in classifying the facial emotion, and we will analyze the result.

Keywords: Face Emotion Classification, Convolutional Neural Network

1 Introduction

Image classification is a new technology developed in recent years. Its main research content is image classification and description. Since 2010, the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) image classification competition has become a well-known international competition, and it has achieved many excellent results. The competition uses a subset of the ImageNet (Deng et al., 2009) dataset, which contains millions of images divided into more than 1,000 categories. Then, Krizhevsk et al. (2017) proposed AlexNet using the convolutional neural network (CNN) model for the first time in large-scale image classification tasks, and successfully reduced the error rate to 16.4%, which was about 10% lower than the second. AlexNet is an eight-layer convolutional neural network whose first five layers are convolutional layers and the last three layers are fully connected layers. The last layer of its fully connected layer adopts Softmax classification method and uses Rectified Linear Units (ReLU) function as a nonlinear activation function. In addition, the model proposes the Dropout method to reduce the occurrence of overfitting.

Face emotion classification is more difficult. Various methods are used in order to recognize and classify human face emotions, but deep learning techniques are better than other methods because of its powerful functions and fast computing capabilities in different datasets (Hussain et al., 2020). Generally, the process of face recognition and classification involves various steps, such as preprocessing, detection, orientation, feature extraction, and emotion classification (Hussain et al., 2020).The advantage of the convolutional neural network is that it can directly compute convolution with the image pixels and extract image features from the image pixels. This processing method is closer to the processing method of the human brain.

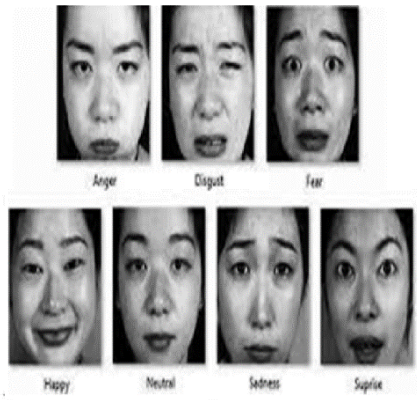

Fig. 1. Facial Expression

In this report, we build a CNN for face emotion classification problems. The dataset is comprised of 675 images that are divided into seven categories: Angry, Disgust, Fear, Happy, Neutral, Sad and Surprise. Fig.1 (Devi and Prabhu, 2020) shows facial expressions of seven different categories. Our purpose is to classify each facial picture into one of the seven

face emotion classes. The input of the CNN model is a facial image, and the output of the CNN is one of facial emotion labels.

2 Method

2.1 Dataset Preprocess

The dataset (Dhall et al., 2011) consists of 675 pictures in png format, and the size of the picture is 720 * 576 pixels. These images are divided into seven classes: Angry, Disgust, Fear, Happy, Neutral, Sad and Surprise. First, we used these 675 images as the original image, and randomly divided the 80% of images of each class into training set and 20% of images into testing set. For example, class Angry has 100 images, and 80 images will be divided into training set, and 20 images will be in testing set.

Next, we created a txt file that stores the images paths and their labels after dividing dataset. Our model would search for pictures based on the information on the txt, and would read the pictures and labels. We cannot use the original images to train our CNN models, because specific input format is required in our model. PyTorch can read pictures through the Dataset class, so we built a ProcessDataset class based on Dataset class to read the images and labels. The image would be resized to 48 * 48 pixels and converted to the grayscale image in ProcessDataset class after reading in. DataLoader is an iterable object, which stitches each data sample returned by the dataset into a batch and provides data shuffling operations. In DataLoder, the getiterm function in ProcessDataset would be triggered to read the image and the label and spliced into a batch, and the real input of the model would be returned.

2.2 Data Augmentation and Data Normalization

Convolutional neural network needs a large amount or even massive data to train model, otherwise it is very likely to fall into the dilemma of overfitting. However, the dataset of our project cannot provide massive training samples. Therefore, data augmentation is very important in our model training. Effective data augmentation can not only expand the number of training samples, but also increase the diversity of training samples. On the one hand, it can avoid over-fitting, and on the other hand, it can improve our model performance. We used random horizontal flip and random rotation to expand the data, so that the input data would be almost different during every training epoch.

In image processing, each pixel information of the image can be regarded as a feature. It is very important to subtract the average value of each feature to centralize the data. This normalized processing method is called centralized normalization. The data preprocessing of convolutional neural network is usually to calculate the average pixel value of the training set image, and then subtract the average value when processing the training set and test set images. The theory of the operation is that our image is a type of stable data distribution, that is, the statistics of each dimension of the data obey the same distribution. Therefore, the statistical average of the data is subtracted from each sample can remove common parts and highlight individual differences. Fig. 2 shows two final input images.

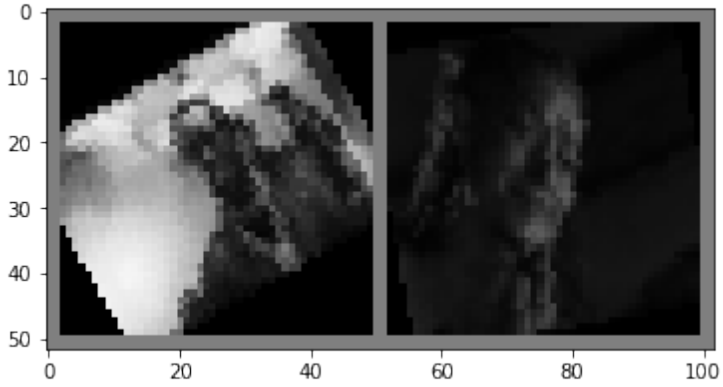

Fig. 2. The input images

2.3 Build Convolutional Neural Network

We built the following network architecture in our project:

[Convolution-ReLU-Max-pool-ReLU]2 -[Fully Connected-ReLU]3-Softmax Loss Function

The first one is convolution and pooling layer. Convolution is a local operation and the local information of the image is obtained by applying a convolution kernel of a certain size to the local image area. The input of this layer is a three-channel picture, and the length and width of the picture are 48 pixels. The convolution kernel size is 5, so the size after convolution is (48 - 5 + 1) = 44. We set the in_channels = 3, and out_channels = 6, so the picture changed from three channels to six channels in this convolution process. We also used max pooling in this layer, and the length and width of the picture is changed from 44 to 22 after pooling. The size of the image after the pooling is smaller than its input. In fact, the pooling is actually a down sampling method.

The activation function layer is also called the non-linearity mapping layer. The activation function is introduced to increase the expressive capacity of the entire network. Otherwise, the stacking of several linear layers will still only play the role of linear mapping and cannot form complex functions. To avoid the occurrence of gradient saturation, Nair and Hinton (2010) introduced the ReLU into the neural network in 2010. The ReLU function is one of the most popular activation functions used in convolutional neural networks. The ReLU function is actually a piecewise function, which is defined as:

$$\mathrm{ReLU}(x) = \max\{0, x\}$$

The second layer is another convolution and pooling layer. The input of this layer is 6-channel data, and the output is 16-channel data. The kernel size of this layer is 5, so the size after convolution is (22 - 5 + 1) = 18. In addition, we also used max pooling in this layer, and the length and width of the picture is changed from 18 to 9 after pooling.

The third to fifth layers are fully connected layers. The input size is 18 and the output size is 9. The fully connected layer acts as a classifier in the entire convolutional neural network. The fully-connected layer can map the learned feature representation to the label space of the sample.

The role of the objective function is used to measure the error between the predicted value and the true sample label. In the current convolutional neural network, the cross-entropy loss function (also known as Softmax loss function) is the most commonly used objective functions in classification problems.

2.4 Train and Test the Model

When training convolutional neural networks, we can randomly disrupt the training dataset before each epoch of training, to ensure that the data in the same batch is different in different epochs of training, even if the training data is fixed. This kind of processing can not only increase the model convergence rate, but at the same time, this can slightly improve the prediction result of the model on the test set.

Convolutional neural networks usually use stochastic gradient descent (SGD) type optimization algorithms for model training and parameter solving. Thus, we used SGD as our optimizer.

Another key point during model training is the learning rate. The network will converge more quickly when using a good learning rate. The initial learning rate should not be too big, so we set the learning rate be 0.001. We would train 200 epochs, and we can see that the loss reduces and the training accuracy increases, and finally the training accuracy is about 90.00%.

3 Results and Discussion

Table 1 below shows the classification results of our CNN model.

Table 1. The result of test set

Class	Total num	Correct num	Accuracy
Angry	20	7	35.00%
Disgust	15	4	26.67%
Fear	20	12	60.00%
Happy	20	7	35.00%
Neutral	20	5	25.00%
Sad	20	9	45.00%
Surprise	20	4	20.00%
Total	135	48	35.56%

Note that the result of each run is different, because we randomly divided the training set and test set. The range of the test accuracy is between 31.00% and 40.00%. Training accuracy is about 90.00% but test accuracy is only 35.56%. The probably reason is that the model is overfitting, as we do not have enough data to train .

Fig.3 shows the accuracy of neural network for face emotion classification that I built in assignment 1. The performance of the neural network is much better than CNN model. This is because the dataset of neural network is not raw images,

which means feature extraction has been done before we trained neural network. However, in this project, we used raw images as our input, so we should extract features from given images.

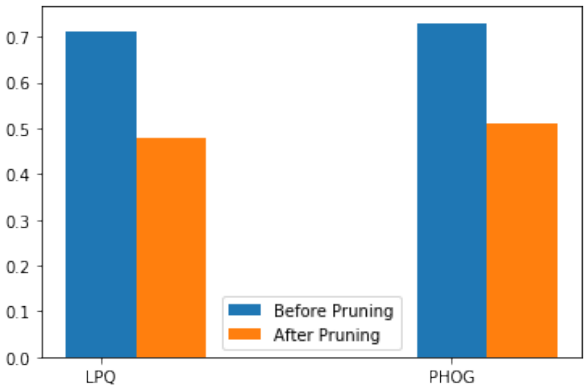

Fig. 3. The accuracy of neural network

The accuracy of face emotion classification of Alizadeh and Fazel (2017) is much higher than mine. This is because they trained CNN models with different depth using gray-scale images from the Kaggle website, which consists of almost 30 thousand 48*48 pixels grayscale images of faces. The faces have been automatically registered so that the face is more or less centered and occupies about the same amount of space in each image. However, there are only 675 images in our dataset, and the images have not been cropped which means that the face is not centered and occupies different amount of space in each image. The result could be better if we can crop the images and put the face in the center of the image before training. Fig. 4 is an example of our dataset.

Fig. 4. An example of the dataset

4 Conclusion

In this report, we have demonstrated how to build a 5-layer convolutional neural network model to classify face emotions. We processed the raw images and used data augmentation to expand data. Our CNN model can basically recognize and classify face emotions. We do not have enough images, so our model maybe overfitting. In addition, our dataset is not good because the face is not centered and occupies different amount of space in each image. Therefore, the accuracy is not good compared to other people's model.

The future work can focus on training convolutional neural network model with more data. Using enough data can avoid overfitting and improve the accuracy. Moreover, we will try to change the number of convolution and fully connected layers.

5 Reference

Deng, J., Dong, W., Socher, R., Li, L., Li, K. & Fei-Fei, L. 2009, "ImageNet: A large-scale hierarchical image database", IEEE, , pp. 248.

Krizhevsky, A., Sutskever, I. & Hinton, G. 2017, "ImageNet classification with deep convolutional neural networks", *Communications of the ACM*, [Online], vol. 60, no. 6, pp. 84-90.

Hussain, S.A. & Salim Abdallah Al Balushi, Ahlam 2020, "A real time face emotion classification and recognition using deep learning model", *Journal of Physics: Conference Series*, vol. 1432, pp. 12087.

Devi, M.K. & Prabhu, K. 2020, "Face Emotion Classification using AMSER with Artificial Neural Networks", IEEE, , pp. 148.

Dhall, A., Goecke, R., Lucey, S. & Gedeon, T.(. 2011, "Static Facial Expression Analysis in Tough Conditions: Data, Evaluation Protocol and Benchmark", IEEE Computer Society, .

Nair, V., & Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)* (pp. 807-814).

Alizadeh, S., & Fazel, A. (2017). Convolutional Neural Networks for Facial Expression Recognition, 1704.06756.