Face Expression Recognition using Deep Neural Network

Mingze Li[1, *, †], Qi Xie[2, †], Zhaohan Yang[3, †], and Aijun Zhang[4, †]

[1]Data science and big data technology, Xi'an Jiaotong-liverpool University, Suzhou, China

[2]Faculty of business and economics, University of Melbourne, Melbourne, Australia

[3]Data science and big data technology, Zhengzhou University of Light Industry, Zhouzhou, China

[4]Data science and big data technology, University of Shanghai for Science and Technology, Shanghai, China

*Corresponding author: mingze.li19@student.xjtlu.edu.cn, qxxie010329@gmail.com, 1925348711@qq.com, zaj153350609@hotmail.com

†All these authors are equally contributed.

Abstract. The research topic concerned in this paper is creating a machine learning model for facial expression recognition (FER). It is a technology to which uses biometric markers to detect emotions in human faces. FER is important because of human-computer interaction, pattern recognition and image recognition. Three components are containing in our method, including dataset construction, model building and Emoji generation. First, the authors aim to build a hand-crafted Convolutional Neural Network to recognize the emotion and designed a GUI in which generates the corresponding emoji derived from the facial expression put in. This paper tested our model in the popular FER2013 dataset and our model achieves 89% accuracy on the dataset, which validates the satisfying performance of our method. Our method can both achieve satisfying accuracy on the dataset and show its strong application ability in most of the tasks. The method can be applied to the real-world scenes for expression recognition and emoji generation.

Keywords: Facial Expression Recognition, Convolutional Neural Network, Emoji Generation.

1. Introduction

In this paper, we learned how to create a machine learning model for Emoji creation. We designed a deep learning-based model to classify different images' facial expressions and analyze the live video feeds in real-time to capture the face and find out the expression. In specific, a convolution neural network architecture is conducted and trained on dataset for facial recognition, thus distinguishing facial expressions from the images. In this paper we introduce facial expression recognition from static images using two-dimensional approach, specifically extracting facial features such as eyes, mouth, eyebrows which are always present in the image. The major advantage of such a robust approach is relatively low computational requirements.

Numerous methods have been worked in this area. Since Darwin made his work in 1872, many behavioral scientists were attracted to this new field of science and the first attempt regarding to the facial expression recognition was completed by Suwa et al. who analyzed facial expressions from image sequences in 1978. However, most of the progress were made during the last decade. There have been various techniques developed to deal with facial expressions, the main stream ways including Convolutional Neural Network [1,2,3] which is also implemented in this paper, Support Vector Machine (SVM) [4], Bayesian Network (BN) [5], and rule-based classifiers [6,7,8]. Furthermore, several scientists achieved a millstone in this topic. Lyons et al used Latent Dirichlet Allocation (LDA) to analyze the principal components of the feature vectors from his training images in order to form discriminant vectors [9]. Cohen et al. made a comparison between different Bayesian Network and discovered that the Tree-Augmented-Naive (TAN) Bayes classifiers was the best performer [10].

The process of our study is that firstly based on convolutional nerval network we constructed the dataset. The dataset used in this project is FER2013 face expression dataset, the size of each image is

fixed as 48×48 grayscale image, containing 7 expressions in total, corresponding to 0-6 numerical labels. Then the network structure is articulated and presented, with GUI interface framework formed as well. We trained the data and tested the experiments several times until experimental results are then presented and discussed. Our model achieves 89% accuracy on FER2013 dataset, which validates the effectiveness.

2. Method

2.1. Introduction of CNN

CNN is a deep neural network that can be used in computer vision widely. CNN can extract specific features from different images. The word "convolution" in CNN stands for mathematical function. Simply, two images can be represented by the product of a matrix to obtain output features extracted from different images. This article will focus on the structure and function of CNN. As a kind of neural network, CNN is composed of the input layer, the hidden layer and the output layer respectively. There are three types of CNN hidden layers, convolutional layer, pooling layer and fully connected layer. As these layers are added, the architecture of CNN is formed. In addition to these three layers, the CNN has two more important parameters, namely the dropout layer and the activation function. The fully connected layer completes the mapping from the input image to the label set, that is, classification. In this process, the most important work is how to further adjust the weights of the neural network by iterating the training data, that is, the backpropagation algorithm. The fully connected layer consists of weights and biases and is used to connect neurons between two different layers. These layers are usually placed before the output layer to form the final layers of the CNN architecture. In this process, the input image from the previous layer is fed back to the FC layer. And then the plane vectors go through the FC layer. The classification process begins at this stage. Classification means that the mapping from input image to label set is completed by using the fully connected layer. CNN will move on to the Fully Connected layer. In this process, the most important work is how to adjust the weight continuously through the training data. This is known as a backpropagation algorithm. The Fully Connected layer includes the weights and deviations of the neurons connected between different layers. These layers are usually located at the end of CNN. In this process, the input image of the upper layer is flattened and fed back to the FC layer. The plane vector then typically performs function operations in these layers and passes through several FC layers. The classification process also starts at this period. CNN will enter the exit after that. In general, when all features are connected to the fully connected layer, overfitting will occur in some training datasets. Overfitting can occur when a particular model performs well on the training data. However, the dropout layer can be used to remove some neurons from the neural network during training, thereby reducing the size of the model to solve this problem. For example, when the exit value is 0.3, 30% of the nodes randomly exit from the neural network.

In general, activation function is one of the most important core parameters in CNN model. Their role is to learn and simulate the relationships between various continuous and complex variables in the network. Simply put, it determines which information about the model should be forwarded. Meanwhile, it also increases the nonlinearity of the network. Common activation functions include ReLU, Softmax, tanH, Sigmoid, etc. These functions serve a specific purpose. The activation function also improves the number of nonlinearities in CNN.

In traditional image recognition and other different application areas (such as audios), CNN is superior to ordinary neural networks. CNN is an excellent feature extractor in a completely new task. This shows that by using the input data at each layer and fine-tuning CNN in conjunction with the assigned task, it is possible to extract useful attributes from a trained CNN and its trained weights. A classifier with a task-specific tag can be added after the last layer of CNN, which is called pre-training. Compared with ordinary neural networks, CNN can accomplish such tasks more effectively. Another benefit of this pre-training is that we can save more memory and time. The only thing to train is the final tag classifier. CNN can capture effective spatial features from images. The spatial features is the

arrangement which is pixels in the image and the relationship between them. Therefore, CNN can identify the object and its relationship with other objects in images more accurate.

2.2. Dataset construction

The data set used in this project is the FER2013 Facial Expression dataset. There are 7 images in total, among which each image is composed of fixed grayscale images with a size of 48×48, corresponding to digital labels 0-6 respectively. The dataset is divided into two files, train and test, where train is used for training and test for testing. Their local distribution is as follows, with seven folders, each containing images of the corresponding category.

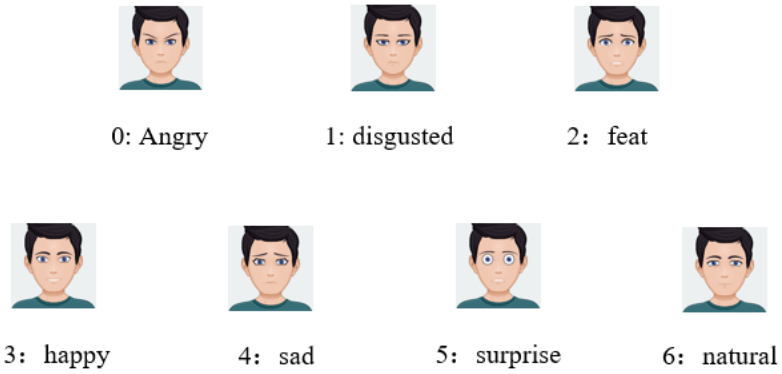

Fig. 1 Category of FER2013.

Therefore, ImageDataGenerator is chosen as iterator in this paper to generate multiple batches of tensor image data through real-time data enhancement, and send the data to the network in a continuous cycle (batch by batch). Together with flow_FROm_directory (directory): generates data after data promotion or normalization based on the folder path. Batch data is continuously generated in an infinite loop. The following parameters are used:

1. Directory: indicates the destination folder path. For each class, the folder must contain a subfolder. Any JPG, PNG, BNP, PPM image in the subfolder will be used by the generator.

2. Target_size: integer tuple, default is (256, 256). The image will be resized to this size, which is used as (48,48) in this paper.

3. Color_mode: color mode, which is "gray_framescale", one of "RGB ", default is" RGB ". Represents whether the images will be converted to single-channel or three-channel images. This article uses gray_framescale.

4. Batch_size: batch size of the data. The default value is 32. This article uses 64.

"Shuffle" : indicates whether to shuffle data. The default value is True. The training set is True and the validation set is False.

2.3. Our network

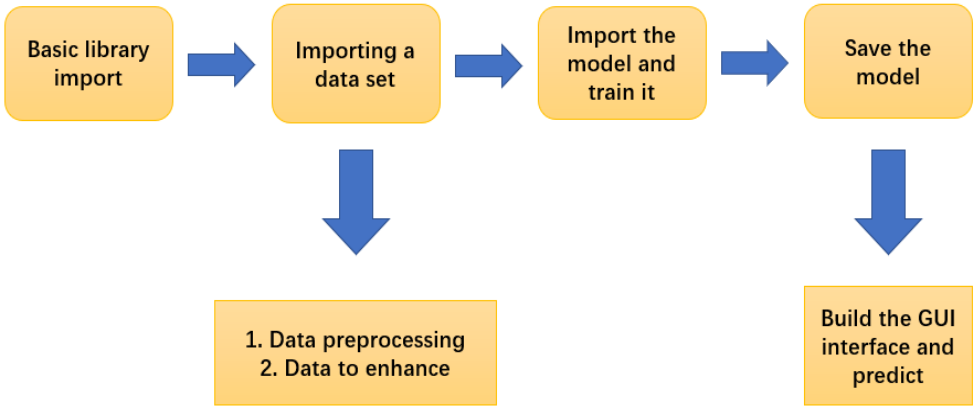

Fig. 2 Framework of Our model.

The network structure is shown in the following figure, which is a linear structure, mainly composed of convolution, pooling and dropout layers in turn. The number of convolution channels is increasing, the size of convolution kernel is (3*3), the size of pooling kernel is (2*2), and the probability of layer loss is 0.5.

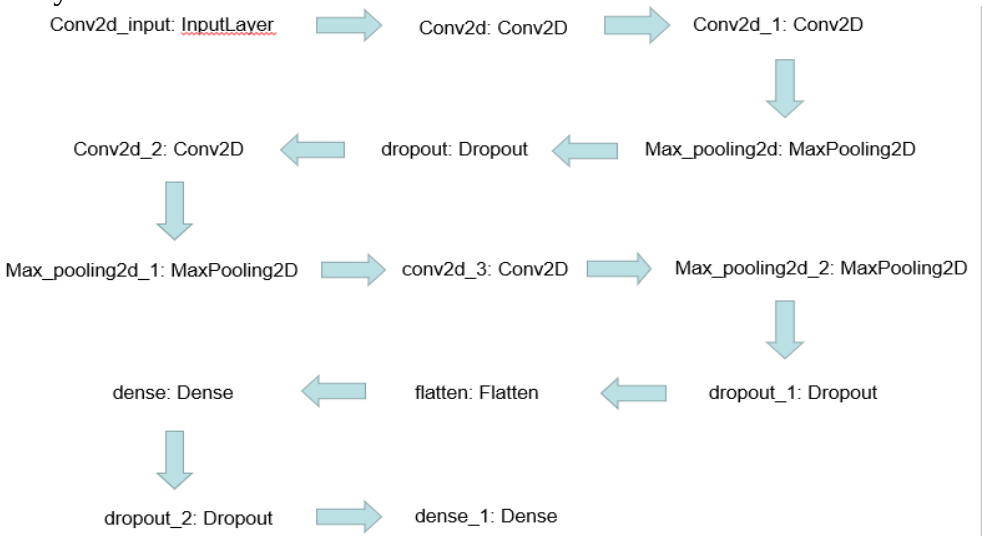

Fig. 3 Structure of the network.

2.4. Emoji Generation GUI

The GUI interface framework uses Tkinter, by importing CV2 library, using Numpy and Keras library, and then operating the camera, shooting human facial expressions in a limited area range, and putting the facial expressions into the trained neural network for classification. The labels that the neural network has learned and output are mapped to different types of local expressions, and the corresponding emoji pictures of the facial expressions are displayed in the defined GUI interface box. This completes the project from face input to expression output content.

3. Results

3.1. Comparison result

The loss and accuracy in the following line charts are calculated using the validation set after each iteration. The loss decreased from 1.788 to 0.363 with 50 iterations. The accuracy increased from 26.6% to 86.8% with 50 iterations. And neither of them showed obvious smooth phenomenon, so there is still room to improve accuracy.

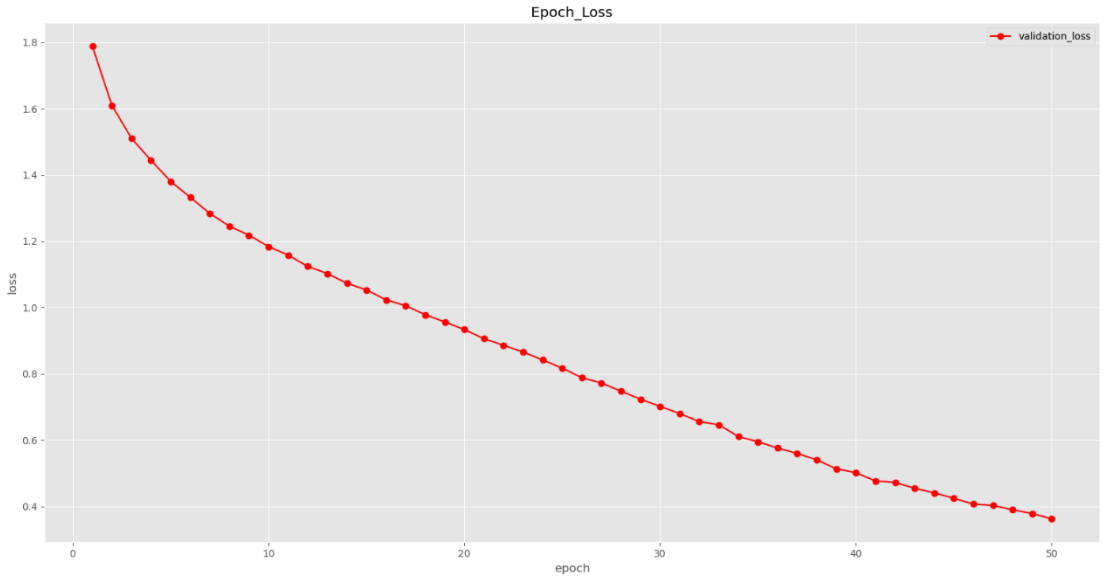

Fig. 4 Loss of validation set

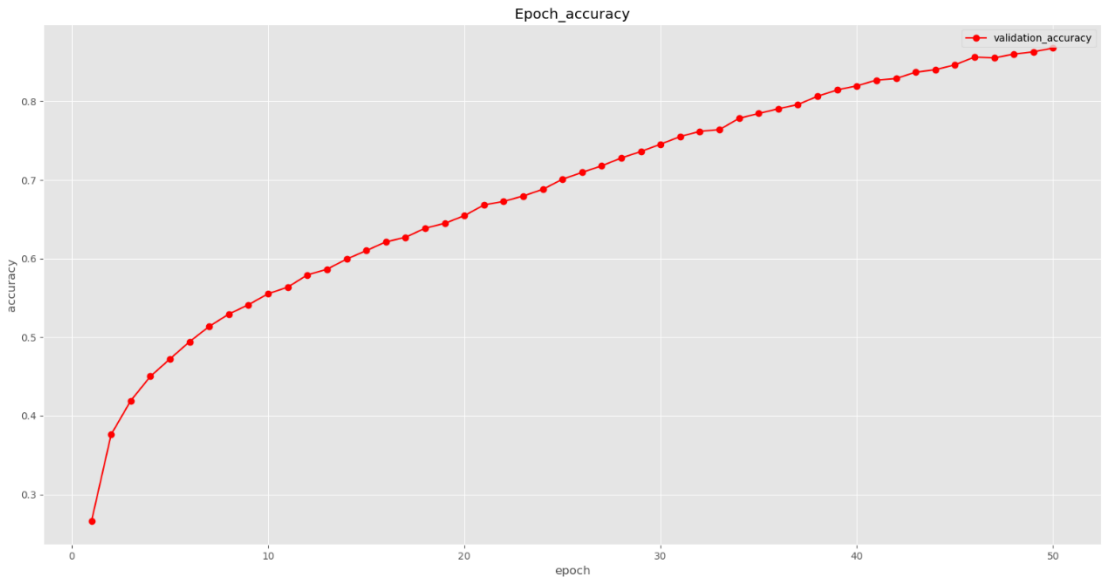

Fig. 5 Accuracy of validation set.

3.2. Real-world Demo

Finally, a photo of the user will be taken. The photo is put into the model for analysis to get its expression label. Then output the emoji image for its label. The following pictures are our display results. As we can see in these three pictures, the first and second pictures are the results of accurate recognition, and the third chapter is the result of recognition failure.

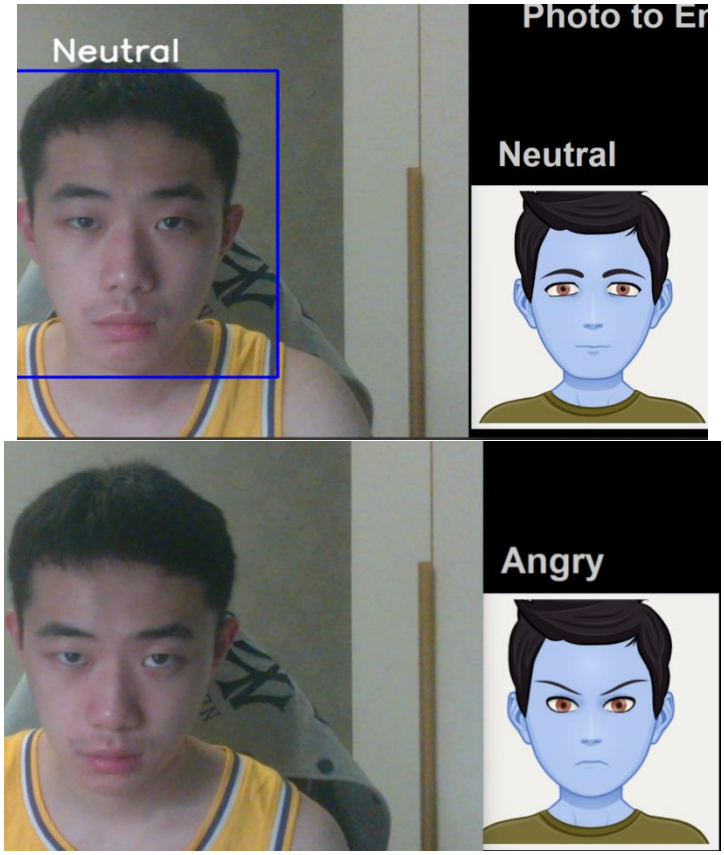

Fig. 6 Result of the application.

4. Conclusion

In this paper, the authors aim to create a machine learning model for facial expression recognition and Emoji generation and train on the data by the convolutional neural network structure, perform emotion recognition on the portrait, complete facial expression recognition, and use numbers 0-6 as the predicted label to accurately classify the recognized facial expressions.

In terms of methods, this research uses Hand-Crafted CNN network to complete the data prediction, efficiently and effectively obtains the 86.8% accuracy of facial emotion classification on FER2013 dataset.

According to the usability and practicality about this study, facial expression recognition technology can realize real-time, accurate analysis of expressions, intuitively and clearly gain the changes or expressions of human emotions, and has wide application potential in many fields such as finance, aerospace, electricity, education and medical care. In addition, the downstream application market continues to expand, promoting key technologies such as 3D recognition and live detection to lead the development of facial expression recognition industry and realize facial expression recognition innovation.

References

[1] C. Padgett, G. Cottrell, Representing face images for emotion classification, in: Advances in Neural Information Processing Systems (NIPS), 1997.

[2] Z. Zhang, M.J. Lyons, M. Schuster, S. Akamatsu, Comparison between geometry-based and Gabor-wavelets-based facial expression recognition using multi-layer perceptron, in: IEEE International Conference on Automatic Face & Gesture Recognition (FG), 1998

[3] Y. Tian, Evaluation of face resolution for expression analysis, in: CVPR Workshop on Face Processing in Video, 2004.

[4] M.S. Bartlett, G. Littlewort, M. Frank, C. Lainscsek, I. Fasel, J. Movellan, Recognizing facial expression: machine learning and application to spotaneous behavior, in: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2005.

[5] I. Cohen, N. Sebe, A. Garg, L. Chen, T.S. Huang, Facial expression recognition from video sequences: temporal and static modeling, Computer Vision and Image Understanding 91 (2003) 160–187.

[6] M. Pantic, L. Rothkrantz, Expert system for automatic analysis of facial expression, Image and Vision Computing 18 (11) (2000) 881–905.

[7] M. Pantic, L.J.M. Rothkrantz, Facial action recognition for facial expression analysis from static face images, IEEE Transactions on Systems, Man, and Cybernetics 34 (3) (2004) 1449–1461.

[8] M. Pantic, I. Patras, Dynamics of facial expression: recognition of facial actions and their temporal segments from face profile image sequences, IEEE Transactions on Systems, Man, and Cybernetics 36 (2) (2006) 433– 449.

[9] M.J. Lyons, J. Budynek, S. Akamatsu, Automatic classification of single facial images, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (12) (1999) 1357–1362.

[10] I. Cohen, N. Sebe, A. Garg, L. Chen, T.S. Huang, Facial expression recognition from video sequences: temporal and static modeling, Computer Vision and Image Understanding 91 (2003) 160–187.