Two new methods for facial expression recognition using Convolutional Neural Networks

Jinfeng Wang1,2*, Xuegang Wang3

1School of Information Engineering, Institute of Disaster Prevention, Sanhe, Hebei, 065201, China
2School of Information Engineering, Bishop’s University, Sherbrooke, Quebec, J1M 1Z7, Canada
3PeopleTech, Amazon Canada, Vancouver, British Columbia, V6C2B3, Canada

*Corresponding author’s e-mail: wangjinfeng@cidp.edu.cn

Abstract. In this research, we propose two novel methods for facial expression recognition to improve the accuracy of recognition. The first of our novel approach is to add the Batch Normalization (BN) layer to the CNN model, and the second of the novel approach is to preprocess the image before image training, such as rotating image, cropping the image and adding Gaussian noise in the picture, especially it is beneficial for unbalanced classifications. Our model consists of 3 CNN layers, 3 BN layers, three average-pooling layers, and three fully-connected layers; our model has a satisfying performance on the prediction category after adopting the two methods mentioned above. Our CNN model is trained and tested with Kaggle facial expression recognition challenge databases. The implemented system can automatically recognize seven expressions in real-time: anger, disgust, fear, happiness, neutral, sadness, and surprise. The experimental results demonstrate the effectiveness of our proposed approach.

1. Introduction

Facial expressions contain a lot of information about human emotions and play an essential role in human communication. Within the past decade or two, automatic facial expression recognition (FER), which had become more and more popular, has become a hot spot in the computer vision and pattern recognition community. Recent advancements and many algorithms, such as PCA and LDA, have been introduced, which is still challenging for its relatively low discriminative rate and weak robustness to occlusions and corruptions.

There have been a lot of related works on the FER problem. For instance, Jinwoo Jeon et al. [1] adopted CNN to analyze the expression training sets and got 70% correctness on Kaggle [2] database; Diah AnggraeniPitaloka et al. [3] enhanced the CNN method to recognize six basic emotions by combining these processing techniques can improve the model’s accuracy to 97.06%; Wei-Lun Chao et al. [4], to enhance the connection between facial features and expression classes, class-regularized locality preserving projection (cr-LPP) is proposed.

The focus recently shifted from traditional machine learning techniques like Bayesian Classifiers and SVMs to Convolutional Neural Networks in the face and expression recognition. In 2016, A. Mollahosseini et al. used CNNs on seven datasets DISFA, CK+, FERA, SFEW, MMI, MultiPIE, and FER2013, to achieve state-of-the-art accuracies [5]. P.Barros et al. tackled challenges with illumination and face positioning using spatial-temporal hierarchical features [6]. Gil Levi and Tal Hassner
used Mapped Binary Patterns with CNNs to counter illuminations variances and trained an ensemble of VGG-16 networks on the CASIA-Web face dataset [7].

A motivating implementation of CNN for real-time expression detection is by S. Ouellet [8]. He improvised a game where a subject’s facial expressions were captured, and the video input stream was used with CNNs. Dan Duncan et al. used a pre-trained VGG-16 and fine-tuned on a home-brewed dataset to achieve 57.1% accuracy and created a real-time detection application [9]. Recently, Andre T. Lopes et al. focused on various image pre-processing techniques, including spatial normalization, intensity normalization, and downsampling to reach start-of-the-art accuracy with lesser training time [10].

This paper uses the new method of adding batch normalization layer and image pre-processing for facial expression recognition. The rest of this paper is organized as follows. Section 2 presents an overview of our facial expression recognition system and introduced our CNN model architecture. The image pre-processing techniques are introduced in section 3. The method of adding batch normalization layer is described in section 4. Section 5 describes extensive experiments of the above methods on FER. And some conclusions are given in section 6. The experiments are carried out on the Kaggle facial expression database.

2. System Overview and CNN model Overview

2.1. Facial Expression Recognizer System

The designed facial expression recognition system consists of four parts: facial image input module, a face detection module, facial feature landmark extraction module, and facial expression classification module. Figure 1 is the flowchart of this system. Facial expressions are divided into seven categories: anger, disgust, fear, happiness, neutral, sadness, and surprise. Here we focus on facial expression recognition, a face detection module, and a face feature landmark extraction module. We use the most commonly used methods.

Face detection is performed with a HOG feature descriptor combined with a linear classifier. This type of detector is general and suitable for not only human face detectors but also semi-rigid objects. So we take the HOG feature descriptor combined method in the face detection module.

After localizing the face, the facial part is cropped, and its dimension is reduced to 42px*42px, which is the input dimension of our CNN model. Features from the input image are extracted and classified by our CNN model. Then, finally, we get facial expression recognition results. See Figure 2 and Table 1 for our CNN model architecture.
Table 1. Our CNN model architecture.

| type              | kernel | stride | pad | output       | dropout |
|-------------------|--------|--------|-----|--------------|---------|
| Input             | 42*42*1 |        |     |              |         |
| Convolution1      | 5*5    | 1      | 2   | 42*42*32     |         |
| BatchNormalization1 |        |        |     |              |         |
| pooling1          | 3*3    | 2      |     | 21*21*32     |         |
| Convolution2      | 4*4    | 1      | 1   | 20*20*32     |         |
| BatchNormalization2 |       |        |     |              |         |
| Pooling2          | 3*3    | 2      |     | 10*10*32     |         |
| Convolution3      | 5*5    | 1      | 2   | 10*10*64     |         |
| BatchNormalization3 |       |        |     |              |         |
| pooling3          | 3*3    | 2      |     | 5*5*64      |         |
| Fully-connected1  |        |        |     | 1*1*2048     | 40%     |
| Fully-connected2  |        |        |     | 1*1*1024     | 40%     |
| Output            |        |        |     | 1*1*7        |         |

Figure 2. Our CNN Model Architecture.

2.2. CNN model Overview

The CNN model structure is also the third facial expression recognition module in the facial expression recognizer system. Our CNN model consists of a total of 3 CNN layers, 3 BN layers, three average-pooling layers, and three fully-connected layers. All images in the dataset are 48*48 pixels in size. After processing by our data augmentation module, we get 42*42-pixel size pictures as our input set, so the input volume dimensions (Width x Height x Depth, or W x H x D) is 42*42*1. Next comes our first layer of convolutional neural networks. The kernel is 5*5. The stride is 2. The padding is 2, next is our BN layer, next is our average-pooling layers. The second layer CNN, the third layer CNN and the first layer CNN are similar. The dropout of the first and the second fully connected layer is 40%. The last layer is our output layer.

In convolutional neural model Architecture, Kernel, stride, padding, Input, output, their relationship is shown in (1) (2) (3).

\[
\text{Input: } (N, C_{in}, L_{in}) \tag{1}
\]

\[
\text{Output } : (N, C_{out}, L_{out}) \text{ where} \tag{2}
\]
\[ L_{out} = \left\lfloor \frac{L_{in} + 2 \cdot \text{padding} \cdot \text{dilaton} \cdot (\text{kern}_{\text{size}} - 1) - 1}{\text{stride}} + 1 \right\rfloor \] (3)

2.3. Dataset for CNN model

Kaggle’s facial expression recognition challenge database is used for training and testing performance. This dataset has seven facial expression categories (angry, disgust, fear, happy, sad, surprise, and neutral), 28709 training images, 3589 validation images and every image is grayscale images, and 48px*48px (see Figure 4 sample Kaggle facial expression image). This dataset contains a human frontal face, poses, and domains, and even cartoon characters are included.

See Figure 3 for the number of each facial expression, 3995 photos are in the anger category, 436 photos are in the disgusting category, 4097 photos are in the fear category, 7215 photos are in the happiness category, 4830 photos are in the sadness category, 3171 photos are in the surprising category, 4965 photos are in the neutral category, Considering that there are averagely 4101 images for a single category, this imbalanced category is enough to cause classification error.

![Figure 3. The number of each facial expression](image)

![Figure 4. Sample Kaggle facial expression image (anger, disgust, fear, happiness, neutral, sadness and surprise).](image)

3. Data augmentation

When the amount of data is not large enough in deep learning, the following four methods are often used: Data augmentation. Manually increase the number of the training set. Create a batch of new data from existing data by translation, flipping, and adding noise.

Regularization. A small amount of data will lead to over-fitting [11] of the model, making the training error very small and the test error very large. The over-fitting can be restrained by adding a regular term after Loss Function.

Dropout: This is also a regularization method, but unlike the above, it is achieved by randomly nul-

ling the output of some neurons.

Unsupervised Pre-training. In this paper, we use method one and method 4 to achieve enough data. Data augmentation mainly includes the following specific methods: Noise perturbation, random perturbation of each pixel of the image, the standard noise modes are Gauss noise (See Figure 4); Flip, flip an image horizontally or vertically; Shift transform, the image is translated in a certain way on the image plane. Random or manufactured definitions can be used to specify the translation range and the translation step length along the horizontal or vertical direction of the translation. Change the location of the image content.

Dropout is a technique for addressing the overfitting problem. The key idea is randomly dropping units (and their connections) from the neural network during training, preventing teams from co-
adapting too much. During training, dropout samples from an exponential number of different “thinned” networks. At test time, it is easy to approximate the effect of averaging the predictions of all these thinned networks by simply using a single unthinned network with smaller weights. Significantly reduces overfitting and gives significant improvements over other regularization methods. In this paper, our dropout is 40%.

To train more data into our CNN model, we executed the data augmentation. We use a 42px × 42px random crop image Training data input (see Figure 5 and Figure 6). By this method, ten times or more, Generate data for training and introduce spatial invariance.

In Kaggle facial recognition challenge training dataset, 436 images are in the disgusting category. Because an average of 4101 photos for a single type, this imbalanced category size is enough to cause classification error.

4. Batch Normalization layer
To reduce this problem of internal covariate shift, Batch Normalization [12] adds Normalization ‘layer’ between each layer. An important thing to note here is that normalization has to be done separately for each dimension (input neuron), over the mini-batches, and not altogether with all measurements. Hence the name ‘batch’ normalization.

Due to this normalization of ‘layers’ between each fully connected layer, the range of input distribution of each layer stays the same, no matter the changes in the previous layer, given x inputs from the kth neuron.

\[ x^{\wedge{(k)}} = \frac{x^{(k)} - E[x^{(k)}]}{\sqrt{Var[x^{(k)}]}} \] (4)

Normalization brings all the inputs centered around 0. This way, there is not much change in each layer input. So, layers in the network can learn from the back-propagation simultaneously, without waiting for the previous layer to learn.
The calculation diagram of the BN layer is as follows: x is the input data, that is, mean-variance normalization, and the latter that to y is a typical linear transformation, similar to full connection but without crossover. See this linear transformation and the latter network as a whole. If there is no BN layer, x is directly inputted into the back network, the transformation of X distribution in the training process will inevitably lead to the latter network to adjust the mean and variance of X since learning, which is reflected in the BN layer. That is normalized data. The cost is that there is a linear layer y in the network, but the performance of the former is more excellent, so it accelerates.

5. Experiments
“A Real-time Facial Expression Recognizer using Deep Neural Network” was published at the 2016 International Conference on Ubiquitous Information Management and Communication. The dataset used in that article is the same as the dataset used in our paper, both of which are the Kaggle dataset. After continuous adjustment of parameters and data processing methods, we finally reproduced the experimental results in that paper, achieving an accuracy of 69.2%. Table 2’s data comes from that paper.

It can be seen from Table 2 that the average accuracy for all categories was 69.2%, accuracy for the happy and surprise classification was higher than the others. Still, accuracy for the fear category was poor. The reason for the fear category's low testing accuracy is that the number of images for the type is tiny compared to other facial expressions; the imbalanced class is enough to cause classification error.

Table 2. Test accuracy confusion matrix of our CNN model without BN layer and data augmentation.

|       | Anger | Disgust | Fear | Happy | Sadness | Surprise | Neutral |
|-------|-------|---------|------|-------|---------|----------|---------|
| Anger | 61.1  | 1.0     | 6.3  | 3.5   | 15.3    | 1.6      | 11.2    |
| Disgust | 23.6  | **67.3** | 1.8  | 0.0   | 1.8     | 1.8      | 3.6     |
| Fear   | 10.6  | 0.2     | **49.5** | 3.0   | 19.7    | 8.1      | 8.9     |
| Happy  | 1.4   | 0.0     | 0.8  | **88.7** | 3.9     | 1.6      | 3.6     |
Sadness 6.1 0.0 8.1 5.1 **65.2** 0.8 14.8
Surprise 2.2 0.0 5.8 3.6 2.9 **83.2** 2.4
Neutral 3.6 0.3 3.7 7.3 14.4 1.5 **69.2**

The average of accuracy is 69.2%

There are six pre-processing steps to enhance CNN performances (a) The method mentioned in the paper [1] (b) cropping (c) adding BN Layer (d) adding Gaussian noise (e) d + c (f) b + d (g) b + c + d, the method of data augmentation (b)(d)(e)(f)(g) only to images that are fear category, increase the number of fear images by ten times, so that the number of fear images reaches the average, to solve the problem of unbalanced category size leading to misclassification.

| Pre-processing step | Anger | Disgust | Fear | Happy | Sadness | Surprise | Neutral | Average |
|---------------------|-------|---------|------|-------|---------|----------|---------|---------|
| (a)                 | 61.1  | 67.3    | 49.5 | 88.7  | 65.2    | 83.2     | 69.2    | 69.2%   |
| (b)                 | 80.2  | **69.4**| 68.9 | 100   | 70.4    | 88.1     | 72.4    | 78.5%   |
| (c)                 | 61.1  | 67.3    | **49.5**| 88.7 | 65.2    | 83.2     | 69.2    | 69.2%   |
| (d)                 | 88.2  | 78.5    | 77.4 | 100   | **74.2**| 100      | 78.8    | 85.3%   |
| (e)                 | 88.2  | 78.5    | 77.4 | 100   | **74.2**| 100      | 78.8    | 85.3%   |
| (f)                 | 88.8  | 74.5    | 78.9 | 100   | 78.4    | **100**  | 79.2    | 85.7%   |
| (g)                 | 88.8  | 74.5    | 78.9 | 100   | 78.4    | **100**  | 79.2    | **85.7%**|

From the results of Table 3, when we use the (b) method to enlarge the image classification of fear expressions by ten times, the accuracy of fear is improved, and the accuracy of each expression is improved accordingly. In particular, the category of happy expressions has reached 100%, and the average accuracy has increased to 78.5%. After using method (c), the accuracy has not been improved. From the results of Table 3, using the (b)(d) approach alone can improve the accuracy, when using (b) (d) method together has the highest accuracy, and using the (C) method will not improve the accuracy.

Based on the above experiment, we have expanded the image of all classifications by ten times. Then used the same (b)(c)(d)(e)(f)(g) data augmentation method. The experimental results are shown in Table 4.

| Pre-processing step | Anger | Disgust | Fear | Happy | Sadness | Surprise | Neutral | Average |
|---------------------|-------|---------|------|-------|---------|----------|---------|---------|
| (b)                 | 81.2  | **70.4**| 65   | 100   | 72.8    | 89.1     | 74.4    | 79%     |
| (c)                 | 61.1  | 67.3    | **49.5**| 88.7 | 65.2    | 83.2     | 69.2    | 69.2%   |
| (d)                 | 89.2  | 78.1    | 77.0 | 100   | 75.2    | 100      | 79.8    | 85.6%   |
| (e)                 | 89.2  | 78.1    | 77.0 | 100   | **75.2**| 100      | 79.8    | 85.6%   |
| (f)                 | 87.8  | 78.5    | 74.9 | 100   | 79.7    | **100**  | 80.2    | 85.9%   |
| (g)                 | 87.8  | 78.5    | 74.9 | 100   | 79.7    | **100**  | 80.2    | **85.9%**|
Table 4 shows the accuracy of all the category images after expanding ten times. Comparing Tables 3 and 4, we can draw the following conclusions. After quantitatively expanding all the images, the accuracy can still be improved. But the rate of promotion is low.

As the training dataset is moderately sized and a batch size of 256 was used, a jagged profile for loss and training accuracy was observed. Figure 7 shows the loss history as well as training and validation set accuracies which were calculated after every ten epochs. The training was stopped at 2000 epochs, and the final test accuracy was nearly 100%, validation accuracy was almost 85.9%, and loss was below 10e-5.

![Validation Accuracy During Training](image)

Figure 7. Validation accuracy during Training.

Figure 7 compares the situation (1) enlarging the number of fear expressions ten times without training on the BN level. (2) Enlarging the number of fear expressions by ten times and training with the BN layer. From Figure 7, we can conclude that the CNN model with the BN layer takes less time to converge and can quickly converge in the initial stage of training; however, the model Without the BN layer, CNN model convergence takes longer.

From the different benefits of Batch Normalization, what is most important is to provide a bit of regularization. You can do less dropout on your network because the BN adds “noise” and adds robustness. Every data point goes through the network with different data points in separate batches — and now, with BN, it will look at the same data point with other normalized “glasses” for each batch.

The time and memory optimization are apparent. “Because the batch normalization already has terms for scaling and shifting” [12]. And thus, the time it takes for the network to converge is shorter. They are putting it before the activation layer is considered best practice and not mandatory.

6. Conclusion

In this paper, we have successfully implemented a system that can automatically recognize seven expressions in real-time. The designed facial expression recognition system consists of four parts; one of the most important parts is the CNN module used to classify images. After we took the image preprocessing method and added the Batch Normalization layer method, the classification accuracy was improved to 85.9%. The experimental results show that image preprocessing enhances the accuracy up to +16.7%. Therefore, in the future, for the image classification problem, we should adopt the method of image preprocessing to improve the generalization ability of the CNN model, especially for unbalanced data; Data augmentation could increase the performance of our CNN model. Batch normalization reduced training time and boosts the stability of a neural network, which means the CNN model with BN layer takes less time to converge and can quickly converge in the initial stage of training. This effect applies to both the sigmoid and ReLU activation functions.
Acknowledgement
This research was supported by: Hebei Province Higher Education Teaching Reform Research and Practice Project under Grant no. 2018GJJG469. We thank the authors for providing Kaggle facial expression recognition challenge databases.

References
[1] Jeon, J., et al. (2016) A Real-Time Facial Expression Recognizer Using Deep Neural Network. In: Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication. New York. pp.1-4.
[2] Kaggle facial expression recognition challenge dataset. https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge/data
[3] Pitaloka, D.A., Wulandari, A., Basaruddin, T., Liliana, D.Y. (2017) Enhancing CNN with preprocessing stage in automatic emotion recognition. Procedia Comput. Sci. 116: 523–529.
[4] Chao, W.L., Ding, J.J., Liu, J.Z. (2015) Facial expression recognition based on improved local binary pattern and class-regularized locality preserving projection. Signal Process, 117: 1–10.
[5] Mollahosseini, A., et al. (2016) Going deeper in facial expression recognition using deep neural networks. In: IEEE Winter Conference on Applications of Computer Vision. Santa Rosa. pp. 1-10.
[6] Barros, P., et al. (2015) Emotional expression recognition with a cross-channel convolutional neural network for humanrobot interaction. In: 2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids). Seoul. pp. 582–587.
[7] Levi, G., et al. (2015) Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In: Proceedings of the 2015 ACM on International Conference on Multimodal Interaction. New York. pp. 503–510.
[8] Ouellet, S. (2014) Real-time emotion recognition for gaming using deep convolutional network features, CoRR, vol. abs/1408.3, pp. 6
[9] Duncan, D., et al. (2009) Facial Emotion recognition in Real Time. Evol. Psychiatr., vol. 74, no. 1, pp. 123-135.
[10] Lopes, A.T., et al. (2015) A Facial Expression Recognition System Using Convolutional Networks. In: 2015 28th SIBGRAPI Conference on Graphics, Patterns and Images. Salvador, pp. 273-280.
[11] Srivastava, N., et al. (2014) Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 15.1: 1929-1958.
[12] Ioffe, S., Szegedy, C. (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on Machine Learning. Lille, pp. 448-456.