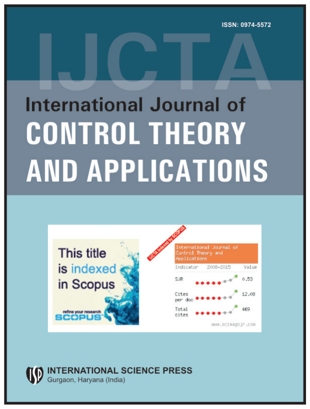
International Journal of Control Theory and Applications

ISSN : 0974-5572

© International Science Press

Volume 9 • Number 43 • 2016

Accelerating Learning Performance of Facial Expression Recognition using Convolution Neural Network

Umesh Chavan[a], Dinesh Kulkarni[b] and Shivanand Hiremath[c]

[a]*Assistant Professor, Department of Information Technology, Walchand College of Engineering, Vishrambag, Sangli, Maharashtra, India. Email: Umesh.chavan@walchandsangli.ac.in*
[b]*Professor, Department of Information Technology, Walchand College of Engineering, Vishrambag, Sangli, Maharashtra, India. Email: dinesh.kulkarni@walchandsangli.ac.in*
[c]*Asst. Professor, Department of Computer of Computer Science, National Defence Academy, Pune, Maharashtra, India. Email: skhnda@rediffmail.com*

Abstract: Facial expression recognition (happy, sad, disgust surprise, angry, fear expressions) is an example of advanced object detection, pattern recognition and classification task. Facial expression recognition techniques detect people's emotions using their facial expressions. Training of recognizing facial expression by machine is a challenging and time consuming task in machine learning. In this project, we have developed deep learning convolution Neural Network (CNN) for facial expression recognition. Experiments on representative facial expression dataset (FER 2013) by KAGGLE are performed to demonstrate the performance of the model. We developed model in Theano and exploited Graphics Processing unit (GPU) computation in order to accelerate the training process. The proposed architecture achieves 61% accuracy. Training on GPU performs 2 to 5 times faster than on the CPU. Furthermore the GPU version scales better than the CPU implementation with reference to network size.

Keywords: Graphical Processing Unit (GPU), Convolution Neural Network (CNN), Machine Learning, Facial Expression recognition (FER).

1. INTRODUCTION

The automatic recognition of facial expression presents a significant challenge to the pattern analysis and man machine interaction research community.it is imperative to develop innovative recognition methods that can detect facial expressions effectively and efficiently [1]. Although humans recognize facial expressions virtually without efforts or delay, reliable expression recognition by machine remains a challenge as of today. To automate recognition of emotional state, machine must be taught to understand facial gestures.

As per Wikipedia, the deep learning represents Deep architectures where both higher level and the lower features can define each other in a best possible way to have better learning [4]. Training the Deep learning network is little bit slow. Shallow structured architecture such as Support Vector Machines (SVMs), Gaussian

Mixture Model (GMMs), Logistic Regressions (LR), Multilayer Perceptron (ML) with single hidden layer are efficient in dealing with many simple problems but found in difficulty while dealing high dimensional complex problem like facial expression detection. It is inspired from neural network methodology. Convolution Neural Network (CNN) is a variant of Multilayer Perceptron (MLP) has two layers: convolution layers and sub-sampling layers. Here, several multilevel stages are present to train a model with feature map, where output feature map provides the some extracted features at all locations on the input. CNN [2] have been widely used for classification tasks. It is observed that CNN is shown better performance than other kinds of neural networks. CNN consists of several convolution layers and fully-connected layers. Each convolution layer extracts features from training set. In the highest convolution layer [5], convolution filter finally reflect objects' representative features. The output from several convolution layers is classified by fully-connected layers.

It is widely recognized within academia and industry that GPUs are the state of the art in training deep neural networks, due to both speed and energy efficiency advantages compared to more traditional CPU-based platforms. Deep Neural Network has been widely used for pattern recognition and classification tasks.

2. RELATED WORK

Otkiri Gupta, Rajiv and Ramesh Raskar [3] used semi-supervised paradigms in convolution neural network for classification of facial gestures. In their paper they listed some of researchers who significantly contributed in developing automatic expression classifiers. Machine learning strategies (like SVMs combined with local binary features) have been applied for facial expression recognition in the past [Kotsia and Pitas2007; Michel and El Kalio by 2003; Shan et. al., 2005; Dhal et. al., 2011; Walekie et. al., 2015; Presti and Casica 2015; Vieru et. al., 2015]. Other methodologies include emotion recognition through speech [Nwe et. al., 2003; Schuller et. al., 2004]. Majority of these systems involves multiple phases - face detection, face alignment, feature extraction and landmark localization followed by classification of labels as the last step. Recently Deep neural networks have proven to classify high dimensional data such as images [Krizhevsky et. al. 2012; Szegedy et. al. 2015], audio and videos [Karpathy et. al., 2014; Tran et. al. 2014]. With development in convolution neural network (CNN) we have seen neural nets applied for face detection [Taigm et. al., 2014; Zhao et. al., 2015] and expression recognition [Abidin and Harjokko 2012; Gargesha and Kuchi 2002; He et. al., 2015] but these networks were shallow or used other feature extraction techniques like PCA. In recent years researchers have made significant progress in developing automatic expression classifiers [11]. Some of them contain drawback of recognition rate or timing. Usually to achieve accurate recognition two or more techniques can be combined.

3. METHODS

Feature extractor is base for the conventional approach for two-dimensional pattern recognition. The input to a neural network is fed from output of feature extractor which is static, independent of neural network. It is challenging task to select a "suitable" feature extractor because it is not part of training procedure and therefore it cannot adapt to network configuration.

CNN makes this challenging task as a part of network and act as a trainable feature extractor with some degree of shift, scale and deformation invariance. CNN are composed of three different types of layers: convolutional layers (CL), pooling layers (PL) (optional) and fully connected layers (FCs). These layers are arranged in feed-forward structure as shown in Figure 1.

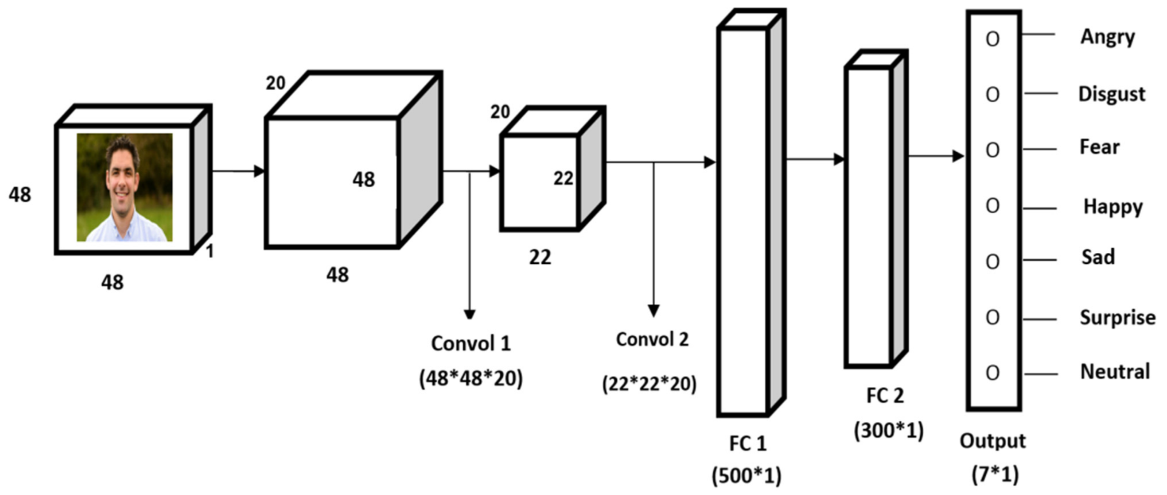

Figure 1: CNN Model Architecture

Our CNN model consists of two convolution layers with max-pooling/sub-sampling layers and two fully connected layers. Detail specification of parameters is listed on Table 1. Implementation of CNN was done in python deep learning library Theano.

Table 1
Our CNN Model Architecture parameters

Type	*Kernel*	*Output Size*
Input		48x48x1
Convolution 1	5x5	48x48x20
Pooling 1	2x2	22x22x20
Convolution 2	5x5	22x22x20
Pooling 2	2x2	9x9x20
Fully connected 1		1x1x500
Fully connected2		1x1x300
Fully output		1x1x7

4. OVERVIEW OF CNN MODEL

The deep network is sequences of convolution layers, pooling layers known as convolution Neural Network. The network has main components-convolution layer (s), pooling Layers (PLs), Rectified Linear Unit (ReLUs), Fully connected layer (FCs), output layer and soft-max layer.

I. Convolutional Layer (CL)

Convolution Layer performs convolution over the input. It consists of several two-dimensional planes of neurons. These planes are called as feature maps. Each neuron of a feature map is connected to small subsets of neurons inside the feature maps of previous layer

First the convolution between each input feature map and the receiptive kernel is computed. Corresponding to the connectivity between the convolution layer and its preding layer these convolution outputs are then summed up together with a trainablke scaler, known as the bias term. Finally the result is passed through an activation function (e.g. ReLU). An illustration is given in Figure 2.

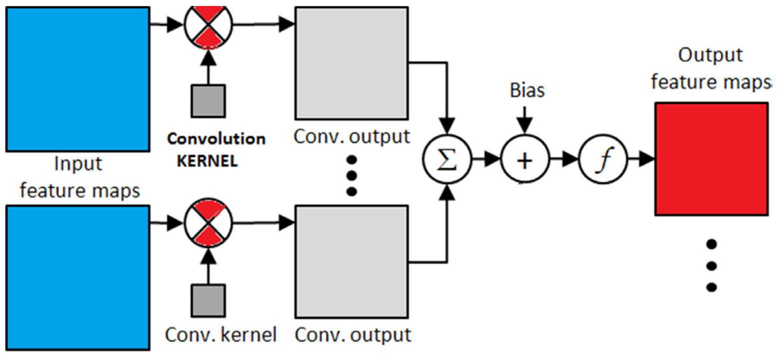

Figure 2: The convolutional Layer of a CNN (from [9])

We refreed article [10] for the mathematical modeling of CNN which is presented here.

The output $y_n^{(l)}$ of feature map n in a convolution layer l is given by

$$y_n^{(l)}(x,y) = f^{(l)}\left(\sum_{m \in M_n^{(l)}} \sum_{(i,j) \in k^{(l)}} w_{mn}^{(l)}(i,j).y_m^{(l-1)}\left(x.h^{(l)}.+i, y.v^{(l)}+j\right) + b_n^{(l)} \right)$$

where, $k^{(l)} = \{(i,j) \in N^2 \mid 0 \le i < k_x^{(l)}; o \le j < k_y^{(l)}\}$ (1)

$k_x^{(l)}$ and $k_y^{(l)}$ are the width and height of the convolution kernels of $w_{mn}^{(l)}$ of layer l. The set $M_n^{(l)}$ contains the feature maps in the preceding layer $(l-1)$ that are connected to feature map n of layer l. The values $h(l)$ and $v(l)$ describe the horizontal and vertical step size of the convolution in layer while $f(l)$ is the activation function of layer l.

II. Pooling/sub-sampling Layer (PL)

The pooling layer is usually placed after the Convolution layer. Its primary utility lies in reducing the spatial dimensions (Width x Height) of the Input image for the next Convolutional Layer. It does not affect the depth dimension of the Volume. The operation performed by this layer is also called sub-sampling, as the reduction of size leads to loss of information as well.

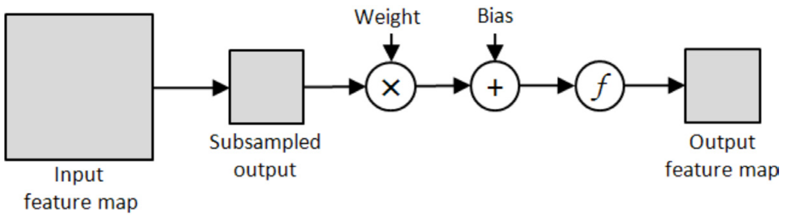

Figure 3: The subsampling Layer of CNN (From [9])

Output ${}_n^{(l)}y$ of a feature map n in subsampling layer l is calculated according to

$${}_n^{(l)}y(x,y) = f^{(l)}\left({}_n^{(l)}w\,k \,. \sum_{(i,j) \in S^{(l)}} {}_n^{(l-1)}y\left(x.s_x + i.y.s_y + j\right) + {}_n^{(l)}b \right)$$

where, $s^{(l)} = \{(i,j) \in N^2 \mid 0 \le i < s_x^{(l)}; o \le j < s_y^{(l)}\}$ (2)

${}_x^{(l)}s$ and ${}_y^{(l)}s$ define width and height of the subsampling kernel of layer l and ${}_n^{(l)}b$ is the bias of feature map n in layer l. The value ${}_n^{(l)}w$ is the weight of feature map n in layer l and $f^{(l)}$ is the activation function of layer l.

Figure 3 illustrates the process that is performed by a subsampling layer.

III. Fully Connected Layer (FC):

The Fully Connected layer is configured exactly the way its name implies: it is fully connected with the output of the previous layer. Fully connected layers are typically used in the last stages of the CNN to connect to the output layer and construct the desired number of outputs. Let the input be x with size k and l be the number of neurons in the FC layer. This results in a Matrix W_{lxk}.

$$f(x) = \sigma(W \times x) \tag{3}$$

σ is the activation function.

IV. Output Layer (OL):

The output layer is one hot vector representing the class (expression) of the given input image. It therefore has the dimension of seven as seven facial expression classes. The resulting class for output vector x

$$C(x) = \{i \mid \exists i \forall j \neq i : x_j \leq x_i\} \tag{4}$$

CNNs are usually trained with a variant of the gradient based back propagation method. All training patterns along with the expected output are fed into the network. Afterwards the network error is back propagated through the network and used to compute the gradient of the network error w.r.t. the weights, this gradient then used to update the weight values according to specific rules.

5. DATASET

We referred a dataset provided by Kaggle [6] website for Facial Expression Recognition Challenge. This consists of 37,000- 48x48 pixel gray-scale images of faces. Each image is labeled with one of 7 expression categories: Angry, Disgusts, Fear, Happy, Sad, Surprise and Neutral. We used a training set of 36,000 examples, a validation set of 1,000 examples. The emotions are labeled in each image. Network is trained on the data set, which comprises 48-by48-pixel gray-scale images of human faces each labeled with one of 7 emotions. Some samples images with labeled expression are shown in Figure 4 There are variations in the data set considerably in scale, rotation and illumination.

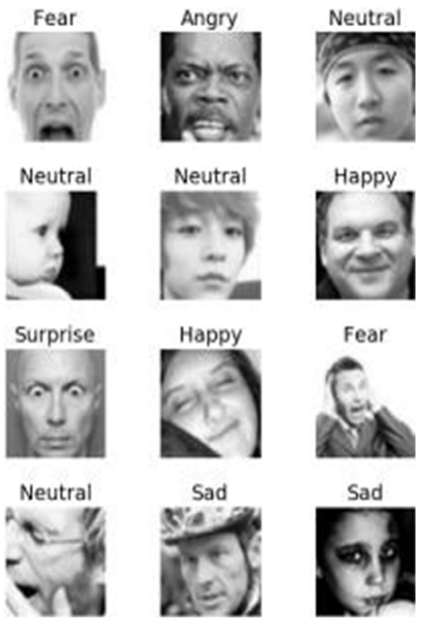

Figure 4: Example images from Kaggle dataset

6. ANALYSIS

A. Experiments

We built a CNN that had two convolution layers and two fully connected (FC) layers. In the first convolution layer, we had 20, 5x5 filters with pooling. In the second convolution layer, we had 20, 5x5 filters and also pooling. In all convolution layers ReLU activation function is used. In the first FC layer we had 500 neurons and in second FC layer we had 300 neurons. In both FC layers same as in the convolution layer we used ReLU activation function. Also we used softmax as our loss function. Figure 1 shows the architecture of this deep network. We trained the network for varying number of epochs on each run (for 2, 10, 30, 50, 70 epochs) and with batch size of 30 samples. We cross-validated the hyper-parameters.

Table 2
The hyper-parameters for model

Parameters	*Value*
Learning Rate	0.00001
Regularization	0.0000001
Decay	0.9999
Epsilon	0.001
Batch size	30

To make the model training faster, we exploited GPU accelerated deep learning facilities on Theano [7] library in using Python.

B. Benchmark and Results

All benchmarks in this paper were performed in machine having with specification-AMD Phenom(tm)II X4 B97 - processor and the GPU is GeForce GTX520, compute capability 2.1,48 cores. The operating system is Ubuntu 14.04. The KAGGLE Database [6] is selected as the training set which contains 37,000 images. Figure 6 shows the confusion matrix on the Kaggle FER 2013 dataset. The performance in GPU speedup over kaggle FER 2013 data set is shown in Table 3. We can see the performance improvement in speed of execution time of CPU and GPU training with different number of epochs which is shown in Figure 5. The average speedup gain is approximately 5 times. The loss history plot for epoch 10 is shown Figure 6.

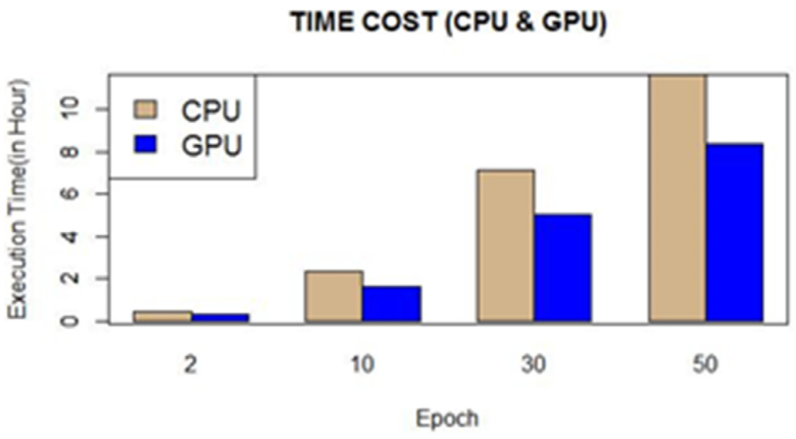

Figure 5: Execution Time and Corresponding speedup with GPU

Table 3
Performance: Execution Time (in hours) and Accuracy

Epochs	*CPU/hour*	*GPU/hour*	*Speedup*	*Accuracy (%)*
2	0.469	0.091	5.15	39%
10	2.346	0.458	5.12	55%
30	7.113	2.344	5.09	59%
50	11.761	2.344	4.95	61%

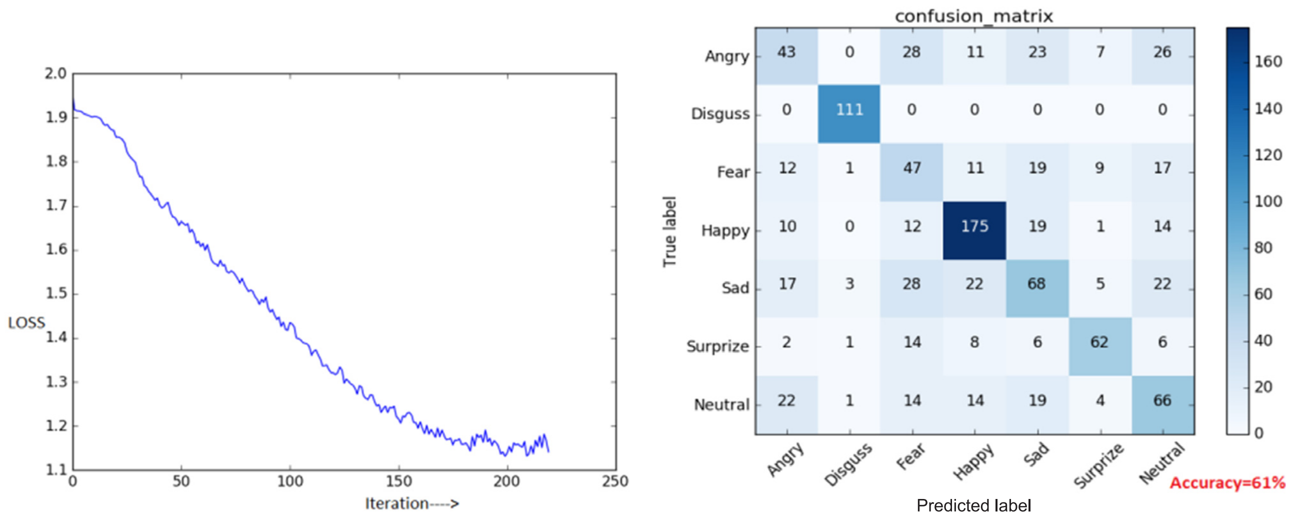

Figure 6: Training Loss History of the Model and Confusion Matrix for Epoch = 10

7. CONCLUSION

The results in this work show that GPUs are just as fast and efficient for deep learning In this work We built convolution neural network (CNN)to recognize facial expression from gray-scale images of faces. We evaluated their performance using different performance measurement and visualization techniques. To accelerate speed of training, we supported GPU to it. The result demonstrated that Deep CNN's are capable of learning facial characteristics and with satisfactory accuracy for facial emotion detection. The validation accuracy obtained is 61% as shown in Figure 6 for epoch = 10.Some of the difficulties with improving this is that images are very small and some cases it is difficult to distinguish which emotion is on each image. The result shows that training on GPU performs 2 to 5 times faster than on the CPU.

Acknowledgment

We thank NVIDIA Corporation for providing support to our research work with the donation of GPU card. We are also thankful to the developers of Theano [7] and Kaggle team [6] for sharing their work with the community.

REFERENCES

[1] Lajervedi and Wu,Member IEEE, Facial expression recognition in Perceptual color space, IEEE transaction on Image processing, Vol. 21,No.8,Aug.2012.

[2] A.Krizhevsky, I.Sutskever, and G.Hinton, ImageNet Classification with Deep Convolutional Neural Network July 2012, NIPS.

[3] Gupta, Otkrist and Raviv, Dan and Raskar, Ramesh, Deep video gesture recognition using illumination invariants, journal arXiv preprint arXiv:1603.06531, 2016

[4] Panda, Mrutyunjaya, Deep Learning in Bioinformatics., CSI Communications, 2012.

[5] Jeon, JinwooandPark, Jun-CheolandJo, YoungJooandNam, ChangMo and Bae, Kyung-Hoon and Hwang, Youngkyoo and Kim, Dae-Shik,, Proceedings of the 10th International Conference on Ubiquitous Information Management and

[6] https://www.kaggle.com/c/challenges-in-representationlearning-facial-expression-recognition-challenge/data.

[7] https://github.com/Theano/Theano.

[8] Burkert, Peter, et. al.,DeXpression: Deep Convolutional Neural Network for Expression Recognition.,arXiv preprint arXiv:1509.05371 (2015).

[9] S.L.Fung and A Bouzerdoum,"MATLAB Librray for CNN" ICT Research Institute,Visual and Audio Processing Laboratory. University of Wollongong Tech. Rep. 2009.

[10] Strigl, Kilfer, Podlipnig,"Performance and Scalability of GPU-based Convolution Neural Networks",IEEE Explorer,Feb 2010.