Optimizing Deep Convolutional Neural Network for Facial Expression Recognition

Umesh B.Chavan and Dinesh Kulkarni

***Abstract*—Facial expression recognition (FER) systems have attracted much research interest in the area of Machine Learning. We designed a large, deep convolutional neural network to classify 40,000 images in the data-set into one of seven categories (disgust, fear, happy, angry, sad, neutral, surprise). In this project, we have designed deep learning Convolution Neural Network (CNN) for facial expression recognition and developed model in Theano and Caffe for training process. The proposed architecture achieves 61% accuracy. This work presents results of accelerated implementation of the CNN GPUs. Optimizing Deep CNN is to reduce training time for system.**

***Index Terms*—Deep Learning,Convolutional Neural Networks,Graphical Processing Units(GPUs).**

I. Introduction

FACIAL expression recognition have found applications in technical fields such as Human-computer-Interaction (HCI) which detect people's emotions using their facial expressions and security monitoring. Use of Machine learning is powerful approach to detect and classify images[1]. To improve their performance, it is necessary to collect larger data-sets, as well as need to build powerful models. The weakest point of machine learning is that it can't do feature engineering. The biggest drawback is it is time consuming for learning with large data sets with powerful model. GPU deep learning is new model where deep neural networks are trained to recognize patterns from massive amount of data. This has proven to be effective

A. Motivation

Deep learning refers to machine learning techniques to automatically learn hierarchical representation in deep architecture for classification. The requirement for deep learning is that it requires availability and compute power to process on it. The successes of ML solution are not dependent on accuracy but also on the time require to train and scale accordingly. To reduce the training time GPU based Deep Learning proved to be the powerful method for Deep learning[2]. The reason was the wide of GPUs that make parallel processing even faster, cheaper and more powerful.

B. Problem statement

The objective is to predict facial expressions from the gray scale image of person's face. Our evaluation metric is the accuracy and training time of the machine. It's visualized with support of confusion matrix and speedup chart. In short

- Input: 48x48 gray scale image of face.
- Output: Face expression class.

DOI: http://dx.doi.org/10.24018/ejers.2020.5.2.495
Published on February 25, 2020.
Umesh Chavan, Walchand College Of Engineering, Sangli, India (e-mail: umesh.chavcan@walchandsangli.ac.in).
Dinesh Kulkarni, Walchand College Of Engineering, Sangli, India (e-mail: d_b_kulkarni@yahoo.com).

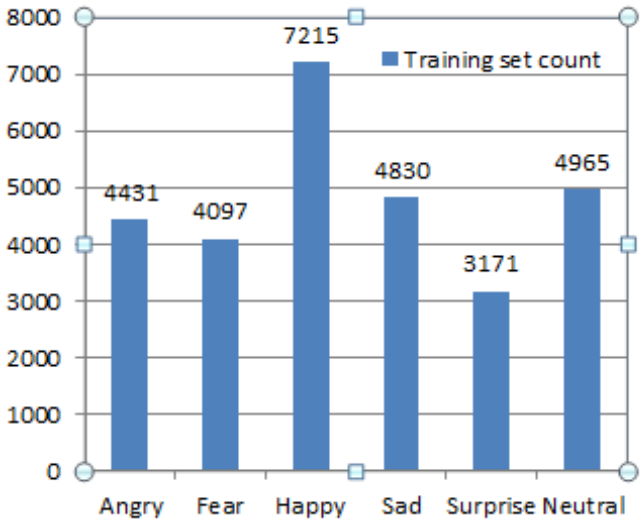

Fig. 1. Distribution of emotions in Training set

C. Data Set

We imported data set from Kaggle competition: "Challenges in Representation Learning: Facial Expression recognition Challenge"[3]. The set is grouped into 28,809 samples as training set and 10% for test and validation set. Data set consists of images and classes of seven types –Anger, Disgust, Fear, happy, sad, surprise, Neutral. The class emotion distribution can be found in Fig. 1

II. Related Work

Lots of significant research is being carried out in developing automatic expression classifiers. The state of facial movement can be represented in Facial Action Coding System (FACS). It is a system to classify human facial movement by their appearance on the face using Action Units (AUs). Advanced methodology used for emotion detection includes neural networks and the multilevel Hidden Markov Model (HMM), Bayesian Networks. The problem of classifying emotions from facial expressions in images is widely studied. One of the first papers to apply neural nets to this end is the EMPATH paper from 2002 [4]. It proceeds by performing Gabor filtering on the raw images followed by various transformations and PCA before applying a 3 layer neural net. More

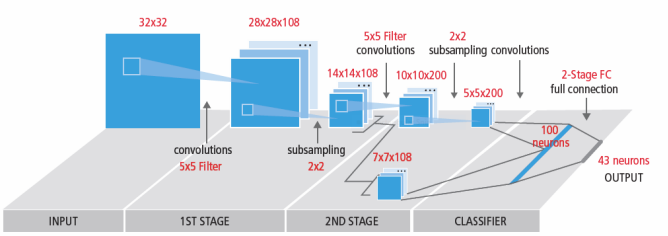

Fig. 2. Typical CNN architecture

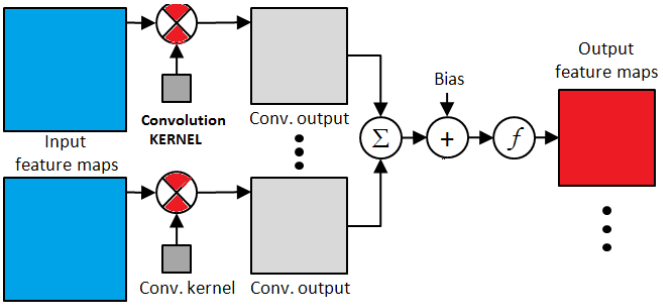

Fig. 3. Convolution Layer

recently, papers such as[5],[6],[7] have used deep neural and convolutional neural networks to classify emotions. Most of these papers focus on classifying emotions in video footage or based on audiovisual data (mixing speech recognition and video techniques). Many papers seek to recognize and match faces), but most papers do not use convolutional neural networks to extract emotions from still images. An exception to this is a paper by Kahou et al. which [4] actually trains a deep convolutional neural network on a set of static images, but then applies this to video data. In addition to the Kaggle Competition, there is also another competition for emotion recognition in static images and videos, the 'Emotion Recognition In The Wild Challenge' [4].

III. Methodology

A. Proposed work

The main focus in this work to improve over the existing approach is to use deep learning algorithms to automatic extractions of features. It also performs classification in the outer layer using supervised learning. The convolutional neural network (CNN) is the first and the more important deep learning implementation. The main blocks of CNN are Convolutional Layer (CL), Pooling Layers (PLs), Rectified Linear Unit Layers (ReLU), fully connected layers (FC), and Loss Layer (LL). A typical CNN architecture can be seen as shown in Fig 2

B. Convolutional Layers(CLs)

The convolution is the computationally intensive operation in CNN. Convolution Layer performs convolution over the input. The convolution is also known as a filtering operation. The filter is represented in Fig. 3 as Convolution Kernel. The neurons in a given kernel share the same weight and bias parameters. This is what it allows the filter to look for the same patterns in different sections of the input image. By arranging the neurons in the kernel in the pattern of grid, we ensure that entire image will get scanned. Convolution of image in CL illustration is shown in Fig. 3 The output is a feature Map (FM).

Different features from input are extracted by applying convolution operation. Edges, lines, and corners are some type of Lower-level features. If we have an image x of size M x N and kernels with size of m x n ,the convolution can be expressed as

$$z_{ij}^{(k)} = \sum_{s=0}^{m-1} \sum_{t=0}^{n-1} w_{st}^{(x)} x_{(i+s)(j+t))}^{(c)} \tag{1}$$

Here w is the weight of the kernel that is model parameter c denotes the channel of image. If the numbers of kernels are K and the number of channels is c. The size of convolved image will be $(M-m+1) \times (N-n+1)$. After convolution all convolved images are activated by the activation function. We have

$$a_n^{(l)} = h(z_{ij}^{(k)} + b_k) \tag{2}$$

Here b denotes the bias of i and, that is $b \in R^{(K)}$ a one dimensional array.

C. Pooling Layer

It is also referred as subsampling layer. In this case there are several layer options; with max-pooling being the most applied. It reduces spatial dimension of the input data.It basically takes a kernel of size 2x2 and then apply it to input volume and outputs the maximum in every sub-block that the kernel convolves around. Output $y_{ij}^{(k)}$ of a feature map n in subsampling layer k is calculated according to

$$y_{ij}^{(k)} = max(a_{(l_1 i+s)(l_2 j+t)}^{(k)}, 1) \tag{3}$$

Here l_1 and l_2 are size of pooling filters and $s \in [0, l_1]$ and$t \in [0, l_2]$. Usually l_1 and l_2 are set to same value $2 \sim 4$.

IV. Experiments and Results

All benchmark in this paper were performed in machine having computation platform with (1) CPU: AMD Phenom (tm) II X4 B97 - processor; (2) GPU is GeForce GTX520, compute capability 2.1,48 cores. The software platform is composed of: Ubuntu 14.04 Operating system, CUDA 7.5, Python with Theano. All training and testing are in single precisions

A. Program Code

In the part of code snippet; the model is created in Python having 2 convolutional layers and one FC layer. The first CL has 32 filters of size 32x32. Second CL has 64 filters of 3x3 sizes. The object CNN is instantiated with parameters for CNN layers.

```
def main():
    X, Y = getImageData()
    model = CNN(convpool_layer_sizes=[(32, 3,3), (64, 3,
3),(96,3,3),(128,3,3)], hidden_layer_sizes=[200],
    )
    model.fit(X, Y,epochs=3,batch_sz=30)

if __name__ == '__main__':
    main()
.
```

Fig. 4. Program code

Fig. 5. Images from Kaggle data set

B. Data Set

The experiment was conducted on the dataset provided by Kaggle (4) website for Facial Expression Recognition Challenge. This dataset consists of 37,000- 48x48 pixel gray-scale images of faces. Each image is labeled with one of seven expression categories:- Fear, Happy, Sad, Angry, Disgusts, Surprise and Neutral. We used a training set of 36,000 samples, a validation set of thousand 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 seven expressions. Some samples images with labeled expression are shown in Fig.8. There are variations in the data set considerably in scale, rotation and illumination.

C. Experiment

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

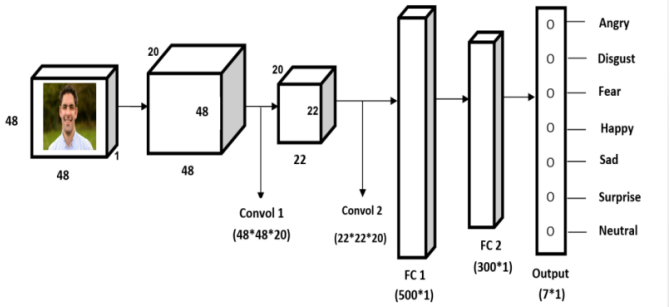

Fig. 6. FER CNN Architecture

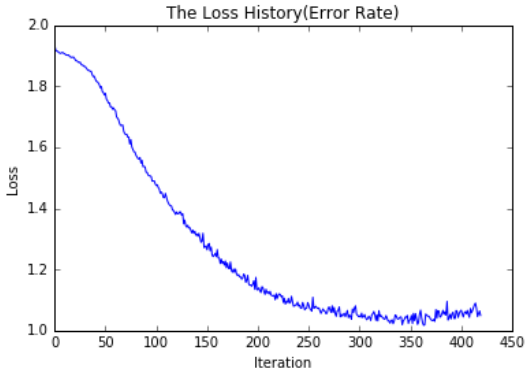

Fig. 7. Training loss

as our loss function. Fig. 9 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 (Learning rate, regularization, decay, epsilon, Batch size) as shown in Table 1. To make the model training faster, we exploited GPU accelerated deep learning facilities on Theano (3) library in using Python.

V. Results & Evaluation

The final validation accuracy we obtained is 61% for training with epochs=10.Loss history plot for epoch 10 is shown in Fig. 7. The confusion matrix for classification is shown in Fig. 8. The performance in GPU speedup over kaggle (4) FER 2013 data set is shown in table I 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 Fig. 9. The average speedup gain is approximately 5 times

TABLE I
Performance: Execution Time (in hours) and Accuracy

Epochs	CPU	GPU	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%

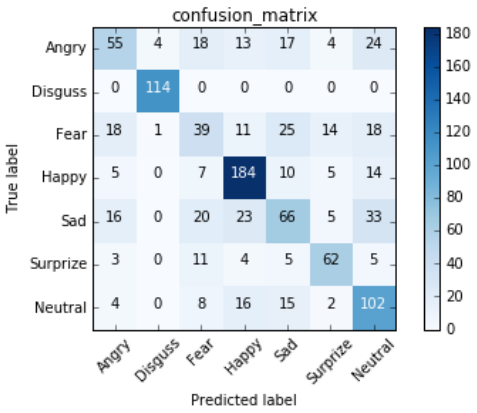

Fig. 8. Confusion matrix

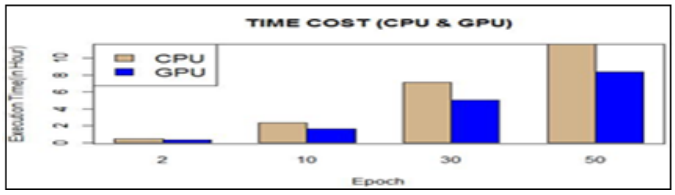

Fig. 9. Speedup with GPU

VI. FUTURE SCOPE

To accelerate speed of training of network, work can be extended by distributing training of CNN on multi core architecture.CNN training can be parallelized across model parallelism or data parallelism.

VII. CONCLUSION

Although topology structure of convolution neural network is simple, it still needs a huge amount of work in calculation. NVIDIA GPU based on hardware architecture of stream processor has significant improvement in face expression recognition based on convolution neural network in support of programming model in CUDA. Compared with CPU, it has amazing advantages. Experiments show that stream processor is suitable for convolution neural network. The results in this work show that GPUs are just as fast and efficient for deep learning. In this work, 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. 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.

ACKNOWLEDGMENT

The authors would like to thank NVIDIA corporation for donating NVIDIA GPU card for this research work.

REFERENCES

[1] Hinton, Geoffrey E.,Osindero, Simon, Teh, Yee-Whye,*A fast learning algorithm for deep belief nets*,Neural Computation ," 2006.
[2] LeCun, Yann, Cortes, Corinna and Burges, Christopher J.C,*The MNIST database of handwritten digits*,http://yann.lecun.com/exdb/mnist/.
[3] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins,*Theano: A CPU and GPU Math Compiler in Python*,PROC. OF THE 9th PYTHON IN SCIENCE CONF. ,2010.
[4] source,*https://www.kaggle.com/c/challenges-in-representation-learning-facial-expression-recognition-challenge*, www.kaggle.com. [Online] ,2013.
[5] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey.("Imagenet classification with deep convolutional,Advances in neural information processing systems, 2012.
[6] Rothkrantz, M. Pantic and J.M., *"Facial action recognition for facial expression analysis from static face images*,IEEE Transactions on Systems,Vol. 34(3),2004
[7] Vinod Nair, Geoffrey E. Hinton,*Rectified Linear Units Improve Restricted Boltzmann Machines*,Toronto : Department of Computer science,University of Toronto

Umesh Chavan obtained his Bachelor's Degree in Computer Engineering from Shivaji University,Kolhapur Then he obtained the Master's Degree in Computer Science and Engineering from Shivaji University,Kolhapur, India and currently doing Ph.D in Computer Science and Engineering at Walchand College of Engineering, Sangli. His specialization includes image processing, High Performance computing, Machine Learning

Dinesh Kulkarni obtained his Bachelor's Degree in Electronics Engineering from Marathwada Univrsity, Aurangabad.He obtained the Master's Degree in Electronic Engineering from Shivaji University, Kolhapur. Also He obtained M. Tech in Computer Science Engineering from IIT, Bombay. He received Ph.D. degree from Shivaji University, Kolhapur. He has published more than 18 papers in National and International Journals. His specialization includes High Performance Computing, Computer Algorithm.