International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

Evaluation of Convolutional Architectures for Offline Handwritten Digit Recognition

Amit Adate[1], Rishabh Saxena[2]

Abstract: ***Deep learning implementations have resulted in significant performance improvements in several application domains and as such several network architectures have been developed to facilitate their methods. This paper presents a comparative study of two architectures among those which are implemented for handwriting recognition, Highway CNN and LeNet-5. The evaluation is performed on two separate machines for both CPU(Intel-i5 3250M) and a GPU(Nvidia GTX-1060). We compared them not only on the basis of their accuracy, but also their training time, recognition time and their memory requirements. Our experiments demonstrate the advantage of global training and feature mapping on the MNIST dataset.***

Keywords: CNN, Highway CNN, LeNet-5

1. Introduction

Deep neural networks have become the backbone of image processing and classification [1]. These networks employ deep layers of different operations to classify and predict images and are also used for natural language processing [2]. Image classification began with Convolutional Neural Networks which laid the foundation for further advances in the field of computer vision [3]. But as CNNs go deeper, we start to hit a bottleneck where adding more number of layers doesn't affect the accuracy too much. To avoid this, several new architectures were formed including cascading CNN by Lin et al [4] which uses several CNNs using the rejection method which passes on the rejected data points to the CNN after it or a voting committee which decides on the final prediction, VGGnet by Simoyan et al [5] which had a 19 layer CNN that strictly used 3x3 filters with stride and pad of 1, along with 2x2 maxpooling layers with stride 2 and AlexNet by Krivhezsky et al [6] which used 8 learned layers to give the highest accuracy ImageNet LSVRC-2010 with a top-1 error rate of 37.5%. Another useful network in this list is a highway network which uses an information highway to pass the information to the layers below using information highways [7]. This network is considerably more accurate than a regular CNN which uses just one channel of information to predict objects. We have evaluated its accuracy in contrast with another popular architecture used for classification, the LeNet-5.

A. Highway Convolutional Neural Networks

In a general neural network, we have the function for a linear neuron as

$$y = f(x)$$

Where f(x) is our convolution, matrix multiplication, or activation for a fully connected layer etc. When the signal is sent backward, the gradient always must pass through f(x), which can cause trouble due to the non-linearities which are involved. To solve this, Residual Networks [8] involve

$$y = f(x) + x$$

Which allows the layers below to directly alter the layers above by sending the data directly. Building on this, we have Highway Networks which apply the equation

$$y = H(x, W_h) \cdot T(x, W_t) + x \cdot (1 - T(x, W_t))$$

In this equation we can see an outline of the previous two kinds of layers discussed: y = H(x,Wh) mirrors our traditional layer, and y = H(x,Wh) + x mirrors our residual unit. What is new is the T(x,Wt) function. This serves at the switch to determine to what extent information should be sent through the primary pathway or the skip pathway. By using T and (1- T) for each of the two pathways, the activation must always sum to 1.

B. LeNet - 5

To solve the problem that small networks are unable to learn the training set, and large networks which are over parameterized are, a new network architecture was introduced by [9]. Designed for the specific purpose of recognizing two dimensional shapes such as digits, while removing variability and the distortions. These notions lead us to the idea of a variant of convolutional neural network mentioned above To perform the task of feature detection, in every CNN the units take their data inputs from the receptive field a layer below, essentially extracting their local feature. And such units in different locations of the image are grouped together to generate a feature map [10].This operation is equivalent to convolution followed by a feature representation function.

2. Methodology

The model currently used uses various python libraries and multiple deep learning packages for a lowered down version of the highway convolutional neural network. The proposed network was created in a flavor of TensorFlow called TensorFlowLearn [11] which uses specialized convolutional, max pooling and fully connected layer functions along with batch normalization [12] functions and a list of models as well as training functions.

To view our experiments, refer to [13]

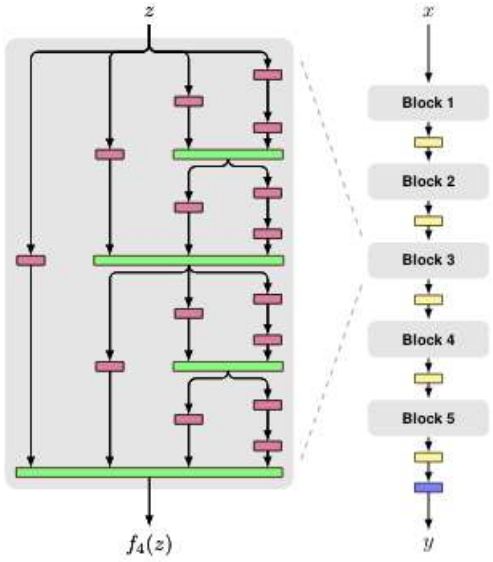

Illustration 1: The highway network block in long chain of CNN layers. Notice thatthere are layers which extend straight below to the final fully connected layeras well. This helps the highway network to perform better without increasing depth

The model uses the MNIST dataset [14] which is widely available online as well. This dataset is used for digit classification and contains 28 x 28 = 784 pixel images of hand written digits which have 60,000 training examples and 10,000 test case examples. The model was trained on an i5 3250M processor with 2.6GHz which took 28 minutes to train.

We first clear the dataset and import it into the program by using the tflearn input function that decides on the height, width and batch size of the given input images. The model uses the capabilities of the highway networks and implements them by training multiple layers of highway convolutional layers and max pooling layers before training the fully connected layers to give the output. The highway layers provide a single pathway to the data between the layers while normal data can pass through the max pooling and fully connected layers. The highway layers help to transmit the error directly to the top of the given CNN. This improves the performance against normal CNNs as it provides with a direct as well as indirect error correction methodology which helps train the network better [7]. The activation function used is the Exponential Linear Unit [15] which uses the following condition for it's activation:

$$f(x) = \{ x , x \geq o\}$$

and

$$f(x) = \{\alpha.(exp(x)-1), x<o\}$$

The function used to train the highway convolutional layer is the highway conv 2D method which takes the parameters as the given network, number of convolutional filters and the activation function. Batch normalization potentially helps in two ways: faster learning and higher overall accuracy. The improved method also allows you to use a higher learning rate, potentially providing another boost in speed [12]. The regression in the final layer provides the final output for the CNN that allows us to predict the objects based on the given dataset. The process of feature mapping can be performed by implementing the feature map as a plane of units that share a single weight vector. That is there is a constraint while performing the operation across the image. The weight sharing methodology helps in reducing greatly the number of free parameters, as large number of units share the same weight. We extract multiple feature maps, extracting various feature type.

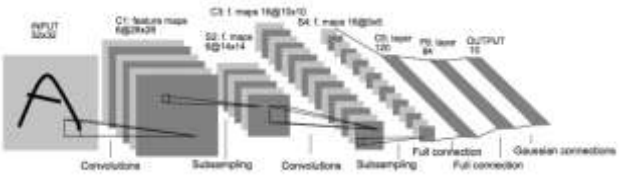

illustration 2: The LeNet Architecture

LeNet 5 is an architecture similar to the one mentioned above. But has more feature maps, a bigger fully convoluted layer, and its representation format to encode the categories at the output layer is distributive. LeNet 5 has a total of about 340,000 connections and 60,000 free parameters, most of them are in the last two layers.

In our training procedure we implemented a module that distorts the input images during training using small randomly picked affine-transformations, applied through shifting, skewing, scaling and rotation.

3. Results

The given Highway Convolutional Neural Network performs with the following metrics, The total time taken for each epoch under verbose3 command is 611.92 seconds on the current CPU training model. This model was run for 14 epochs with a batch size of 1000 and a learning rate of 0.0001.

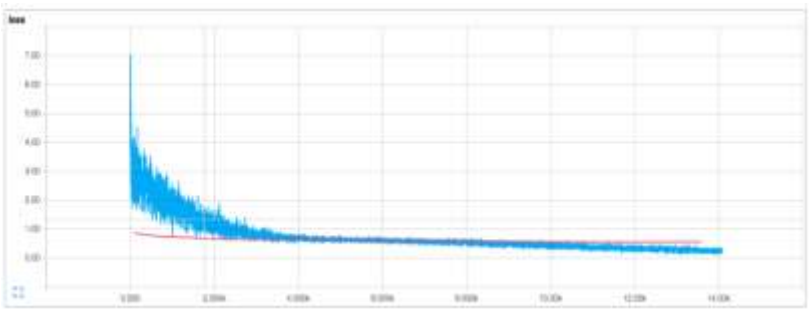

Illustration 3: Loss vs Epochs for Highway CNN

For illustration 2, the red line indicated the test data whereas the blue line indicates the training data. We see that the training data loss on the Highway CNN goes down fast but the crawls as the number of epochs increase. The loss at the end of the epoch 14 averages around 0.58. For the testing dataset, the values remain mostly constant as the model has been fit well with less epochs, hence the loss for the entire testing dataset averages around to about 0.77.

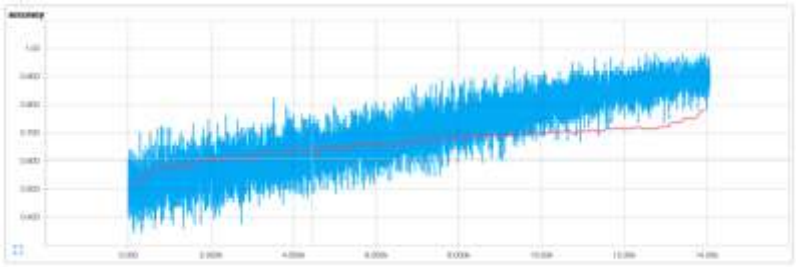

Illustration 4: Accuracy vs Epochs for Highway CNN

For illustration 3, the red line indicates the test data whereas the blue line indicates the training data. We see that the accuracy of the model increases steadily with greater variations at the start. The training set variations become more even as we go through the training, finally stopping at an average of 91.65%. At the same time, the testing set accuracy increases at a lower pace but take a sharp turn at the end when the accuracy starts to stagnate, allowing a much better fit of the data.

The illustration 4 describes how the Highway Model and a normal CNN model based on a paper by Klaus et al [7], which differ as the number of epochs increase overtime, allowing for a better loss to epochs graph. We trained with MNIST set with image size 28x28. To match the size of LeNet 5, the first convolution layer applied padding. We also used Relu instead of Sigmod as activation function. And we Applied dropout in the FC layer.

The functional requirements for the LeNet that we considered are as following, Input Image is 28x28x1 and is converted to 32x32x1. First Convolution layer has the output shape window of 28x28x6. All the Activation functions applied can be random. The first Pooling layer should be of the output shape window of 14x14x6. The second convolution layer has the output shape window of 10x10x16. The second Pooling layer has the output shape window of 5x5x16. We added a layer to flatten the output of the output shape of the final pooling layer such that it is read as a 1 dimensional instead of 3 dimensional. Our first fully connected layer has 120 output nodes and our second fully connected layer has 10 output nodes. The LeNet function mentioned here [9] is applied to the result of the second fully connected layer.

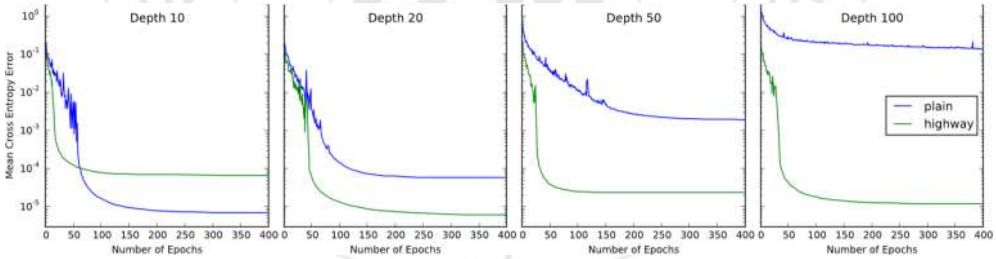

Illustration 5: Highway Model Vs Normal Model

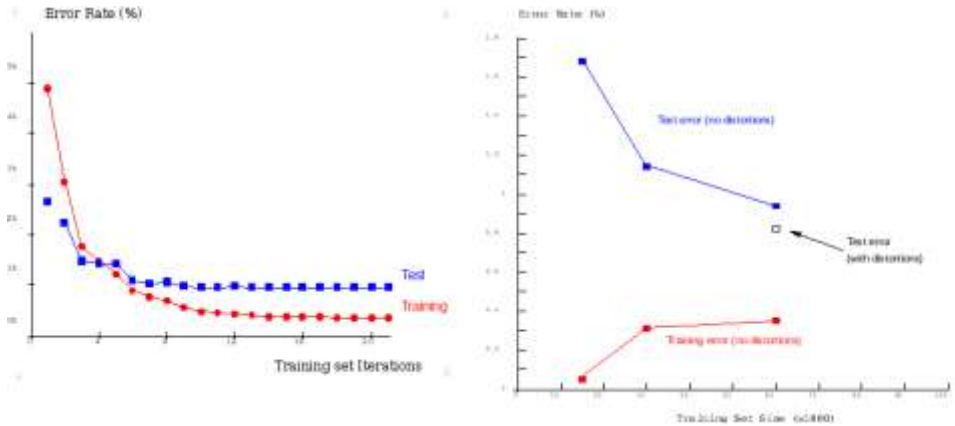

Illustration 6: Training vs Testing for LeNet-5

Our implementation of LeNet yields 99.1% correct rate on the MNIST test set.

References

[1] Dan C. Ciresan, Ueli Meier, and J¨urgen Schmidhuber. Multi-column deep neural networks for image classification. CoRR, abs/1202.2745, 2012.
[2] Marc Moreno Lopez and Jugal Kalita. Deep learning applied to NLP. CoRR, abs/1703.03091, 2017.
[3] Dan C. Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and J¨urgen Schmidhuber. High-performance neural networks for visual object classification. CoRR, abs/1102.0183, 2011.
[4] Ilya Kalinowski and Vladimir Spitsyn. Compact convolutional neural network cascade for face detection. CoRR, abs/1508.01292, 2015.
[5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014.
[6] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. pages 1097–1105, 2012.
[7] Rupesh Kumar Srivastava, Klaus Greff, and J¨urgen Schmidhuber. Highway networks. CoRR, abs/1505.00387, 2015.
[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. CoRR, abs/1512.03385, 2015.
[9] Yann Lecun, Lon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In Proceedings of the IEEE, pages 2278–2324, 1998.
[10] Yann Lecun, L. D. Jackel, Harris A. Eduard, N Bottou, Corinna Cartes, John S. Denker, Harris Drucker, Eduard Sackinger, Patrice Simard, and Vladimir Vapnik. Learning algorithms for classification: A comparison on handwritten digit recognition. In Neural Networks: The Statistical Mechanics Perspective, pages 261–276. World Scientific, 1995.
[11] Tensorflow learn.
[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. CoRR, abs/1502.03167, 2015.
[13] Github repository https://github.com/amitadate/convolutional-architectures-mnist
[14] Yann LeCun. The mnist database of handwritten digits.
[15] Djork-Arn´e Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). CoRR, abs/1511.07289, 2015.