Task 2: Classification of Handwritten digits

Sanjeev Narayanan

17th February 2018

This task aims to recognize handwritten digits input as images using Convolutional Neural Networks (CNN) [1]. The data set was used from the MNIST database of handwritten digits [2]. The original images are from NIST and they were normalized to fit a 20×20 pixel box. The normalization algorithm converts the images to a gray scale format and centers it in a 28×28 image.

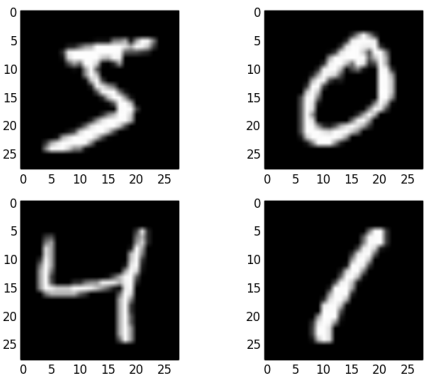

Figure 1: Examples from the MNIST dataset

A two-layer convolution network is employed with a max pooling layer for each and two fully connected layers for the classification. This experiment also varies certain parameters of the network in order to view the effects of changing the parameters on loss computed in each epoch of the categorical-cross validation (10 epochs).The training is done with the first 60,000 samples and the testing performed with the remaining 10,000. The network topology [1] can be summarized as follows:

- Convolutional Layer with F1 number of filters and feature maps of size $S1 \times S1$
- Max pooling layer for the above covering 2×2 patches
- Convolutional Layer with F2 number of filters and feature maps of size $S2 \times S2$

- Max pooling layer for the above covering 2×2 patches
- Dropout layer with probability of 20%. Dropout consists of randomly setting a fraction rate of input units to 0 at each update during training time, which helps prevent overfitting.
- A Flatten layer to flatten the input images.
- Fully connected layer with N1 neurons and rectified linear unit layer activation
- Fully connected layer with N2 neurons and rectified linear unit layer activation
- Output layer

The process used was a straightforward method. The libraries and the classes are loaded with the training and testing data initialized. The above mentioned model is then run and the classification error rates are observed.

In order to vary the parameters, the following are considered:

- F1 = 10,30,96
- F2 = 5,15,48
- S1 = 3,5,11
- S2 = 2,3,5
- N1 = 56,128,256
- N2 = 28,50,128

For each of these cases, the loss percentages in each epoch are plotted.
The following observations are immediately deduced from the figures 2,3 and 4

- As filter size increases, the losses reduce and converge to a very small value.
- For a bigger filter size, accuracy tends to be better.
- Larger the size of dense units in the fully connected layers, better is the accuracy of prediction of digits.

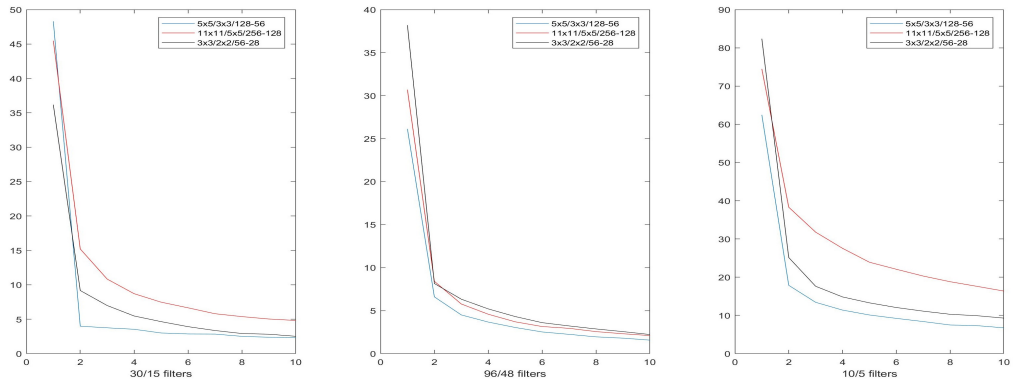

Figure 2: Plot of Loss percentage for varying number of filters

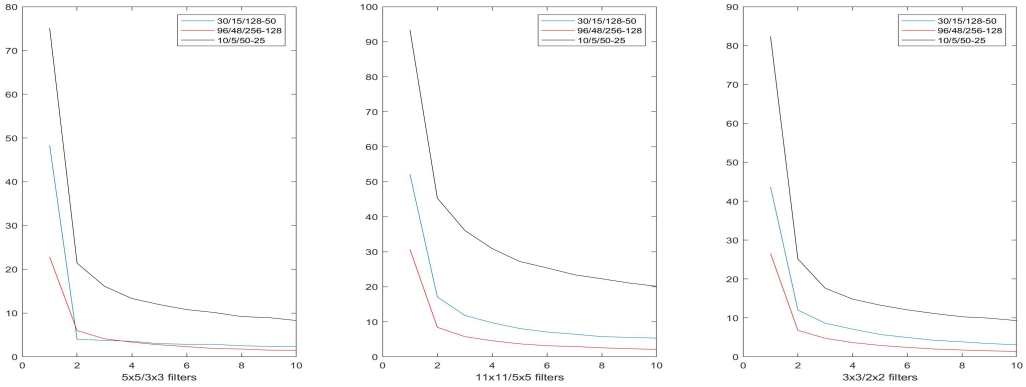

Figure 3: Plot of Loss percentage for varying size of filters place holder

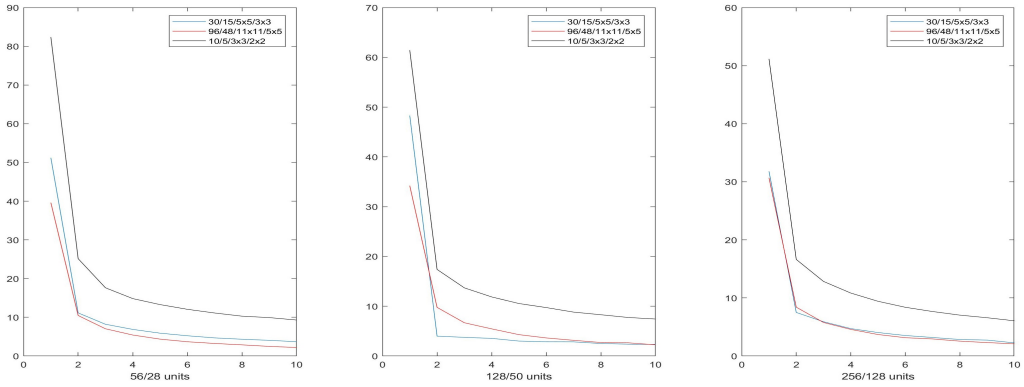

Figure 4: Plot of Loss percentage for varying number of dense units

However, there is a huge trade-off in terms of computation time versus size of filters, number of filters or number of dense units. For a trivial problem like predicting the digit input, we can go easy on the accuracy and focus more on the computation time. Since this code is run on a CPU, the computation time varies from 1 minute to 10 minutes per epoch. However, on a titan GPU this process is much faster as it lasts for a maximum of 1 minute in the most time consuming epoch. What we can conclude from this task is that we can use CNN for a better classification of handwritten digits over shallow learning techniques. By varying the parameters, it is clear that using 96 and 48 filters with 11×11 and 5×5 feature space matrices, employing 256 and 128 dense units produce the best performance in terms of predicting the digits with an overall error of 0.65%.

References

[1] F. Chollet *et al.*, "Keras," 2015.

[2] Y. LeCun, C. Cortes, and C. Burges, "Mnist handwritten digit database," *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, vol. 2, 2010.