Classifying handwritten digits in MNIST Dataset using CNN
Trilokinath Modi

1 Methodology

The idea behind the given task is to train two different Convolution neural network with the objective of predicting the image labels on a untrained test data with high accuracy. The train, test and valid data sets were given as part of exercise. Stochastic gradient descent algorithm is used in both networks which samples from the input data distribution uniformly randomly, however, a batch size of 8192 was maintained to speed up learning. The weights are initialised from a normal distribution $N(0, 0.01^2)$ in all layers of the entire network, where ever applicable. The network architecture was also given as part of exercise. All the input images were divided by 255 to make it $\in$ [0,1]

2 Results

The classification errors obtained on all three data-sets is presented in table 1. Figure 1 represents the learning curve for both the network. Clearly, network 2 achieve better accuracy. Network 2 has much more feature maps which are primarily responsible for capturing the local structural properties of images implying the network gets greater freedom diving into fine intricacies while learning the training data. The multiple convolution layer in network - 2 helped in capturing more abstract features. The pooling layer summarises the outputs of neighbouring feature maps to single values. In other words, the combination of convolution layers greatly reduces the input size, yet, preserving most of the information that can be extracted from images. In the end, the refined data is fed to deep neural network which in principal is same as multilayer perceptron. In the figure 1, it can also be observed that the network - 2 learns very quickly, which is primarily because of having multiple convolution layers with higher number of feature maps.

Table 1: *Classification error in percentage.*

Network	Training Data	Validation Data	Test Data
1	9.74	10.38	9.22
2	1.10	1.50	1.28

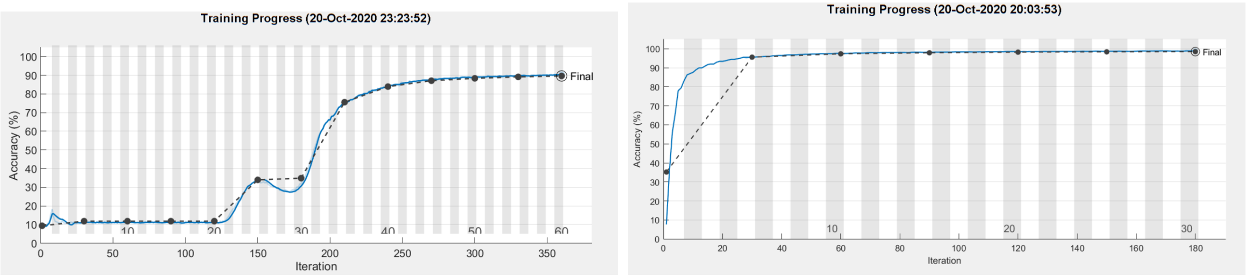

Figure 1: *Learning curve produced by networks 1 and 2 respectively from left(a) to right(b). The x-axis contains information about iterations and epochs. The black spots imply epochs. The y-axis displays validation accuracy. Single GPU is used to compute the plots and roughly it took 2-2.5 minutes to complete the training. Each epoch contains 6 iterations of batch 8192.*