Machine Learning Project 6

Ailiyasi Ainiwaer (Elyas)
Xiaoli Qiu

We used Matlab built-in convolutional neural network function and MNSIT handwritten digits data from Lecun et al. (http://yann.lecun.com/exdb/mnist/) to train and test our network. We also tested our network with Elyas's and Xiaoli's handwritten digits.

1. Data

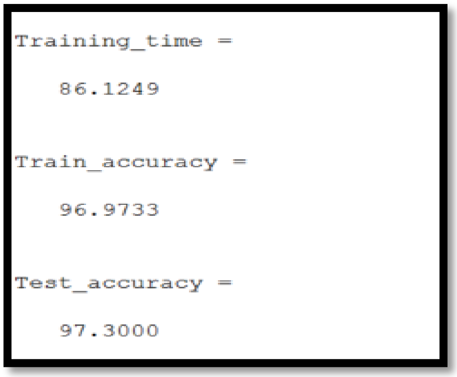
We used 60000 images to train our network, then used 10000 images for testing.

2. Network Architecture

We used a modified version of LeNet-5 as our network architecture:
Input: 28×28
Convlution1: 6, 5×5 filters (output size: 24×24×6) with a stride of 1 plus 6 bias so 1st layer total number of learning parameters = 5×5×6+6=156
Relu activation
Max pooling1: 2×2 filters with a stride of 2 (output size: 12×12×6)
Convolution2: 16, 5×5 filters(output size: 8×8×16) with a stride of 1 plus 16 bias so 2nd layer total number of learning parameters = 5×5×6×16+16=2416
Relu Activation
Max pooling2: 2×2 filters with a stride of 2 (output size: 4×4×16)
Flatten layer: an input for the fully connected layer: 4×4×16=256
Fully connected layer with 10 output: total number learning parameters = 256×10 +10= 2570
Softmax and classification

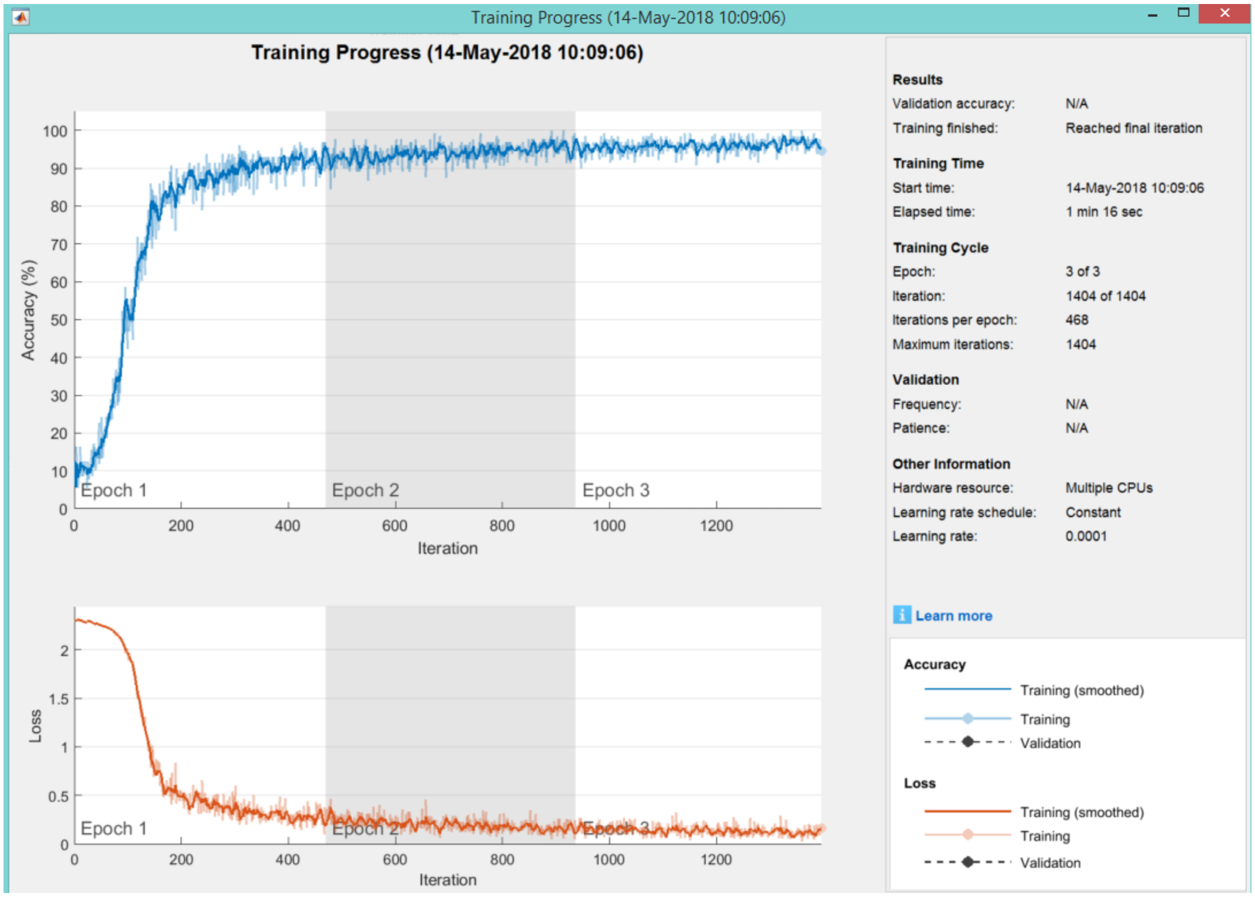
3. Training time

In optimization, we used stochastic gradient descent with momentum , multiple cpu cores used with Matlabs parallel computing function. So, Training time = 86.1249sec (Fig.1 , output from Matlab). We also checked convergence of algorithm by plotting loss and accuracy versus iteration (Fig.2, Matlab built -in plotting feature).



```
Training_time =

    86.1249


Train_accuracy =

    96.9733


Test_accuracy =

    97.3000
```

Figure 1. Screen shot of training time, training accuracy, and test acuracy.



Results

Validation accuracy: N/A

Training finished: Reached final iteration

Training Time

Start time: 14-May-2018 10:09:06

Elapsed time: 1 min 16 sec

Training Cycle

Epoch: 3 of 3

Iteration: 1404 of 1404

Iterations per epoch: 468

Maximum iterations: 1404

Validation

Frequency: N/A

Patience: N/A

Other Information

Hardware resource: Multiple CPUs

Learning rate schedule: Constant

Learning rate: 0.0001

Figure 2. Screenshot of training progress (Matlab built-in feature)

4. We used the trained network to calculate prediction then compare with labels, then calculate accuracy using equation,

   accuracy = (number of corrected labeled samples/ total number of samples) *100

   so, Training accuracy = 96.9733% and Testing accuracy = 97.3000%.

   To visual check performance of network recognition on the test set, we plotted 10 correctly labelled images(Fig.3) and 10 mislabeled images(Fig.4).
5. Finally, we tested our trained network with Elyas's (Fig.5) and Xiaoli's (Fig.6) handwritten digit samples. We used various Matlab built-in image processing functions to produce input images similar to LeCun's dataset.

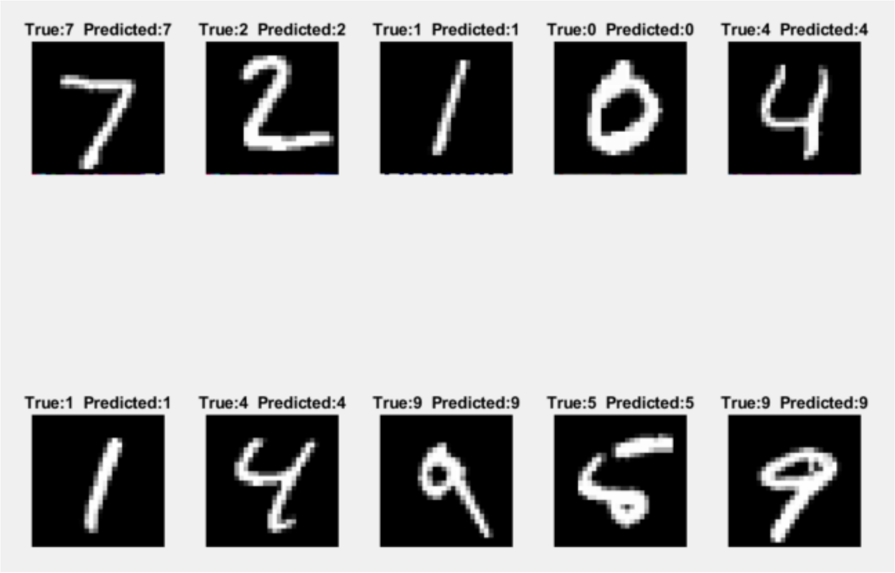


Figure 3. 10 correctly labeled images from test data

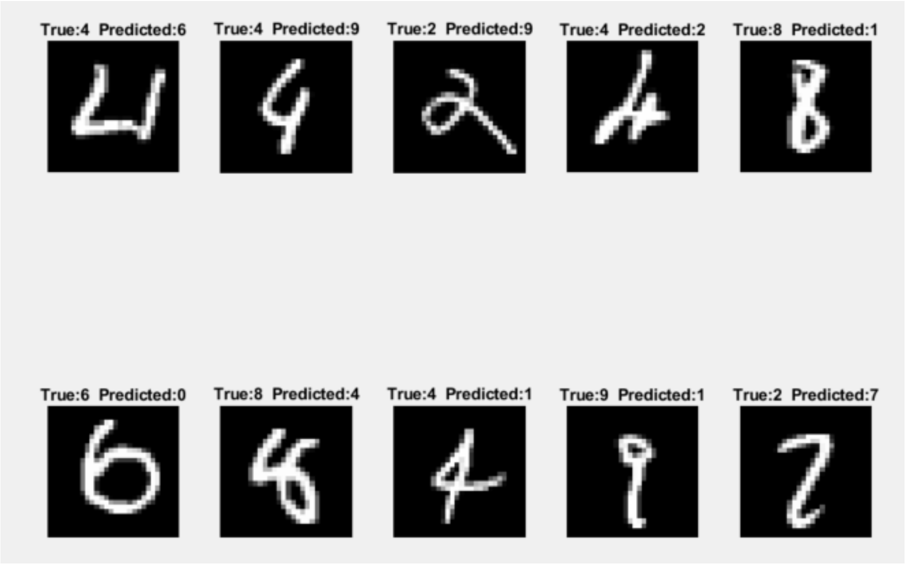


Figure 4. 10 incorrectly labeled images from test data

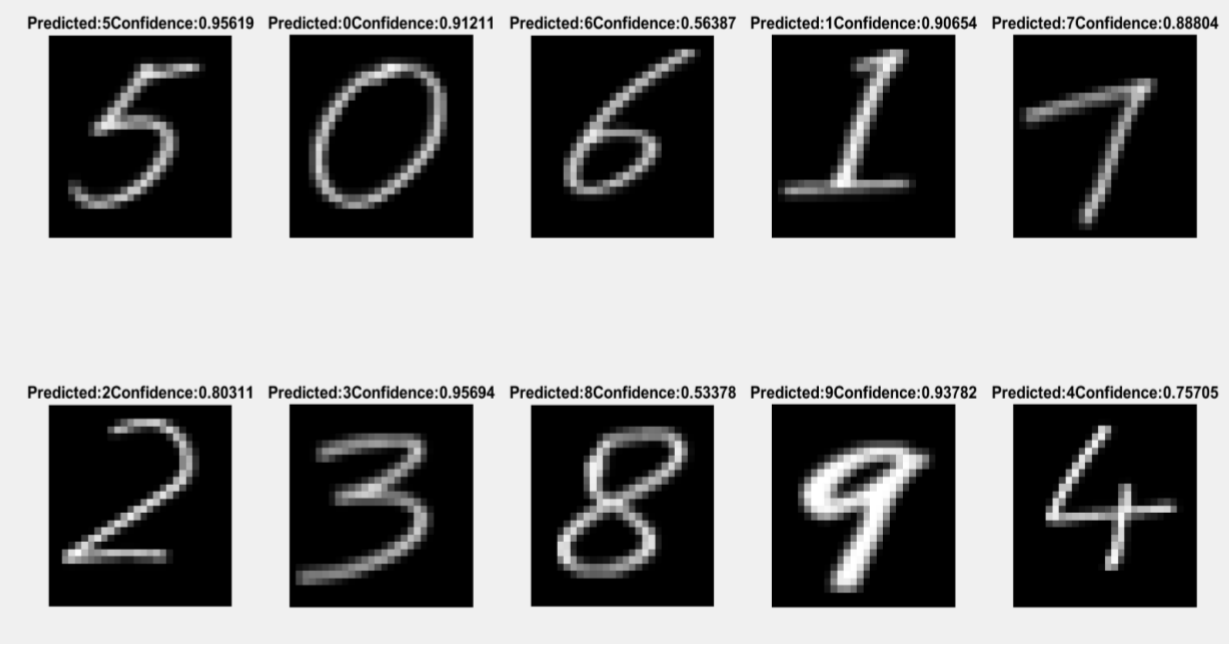

Predicted:5Confidence:0.95619
Predicted:0Confidence:0.91211
Predicted:6Confidence:0.56387
Predicted:1Confidence:0.90654
Predicted:7Confidence:0.88804
Predicted:2Confidence:0.80311
Predicted:3Confidence:0.95694
Predicted:8Confidence:0.53378
Predicted:9Confidence:0.93782
Predicted:4Confidence:0.75705

Figure 5. Cropped and predicted images (Elyas)

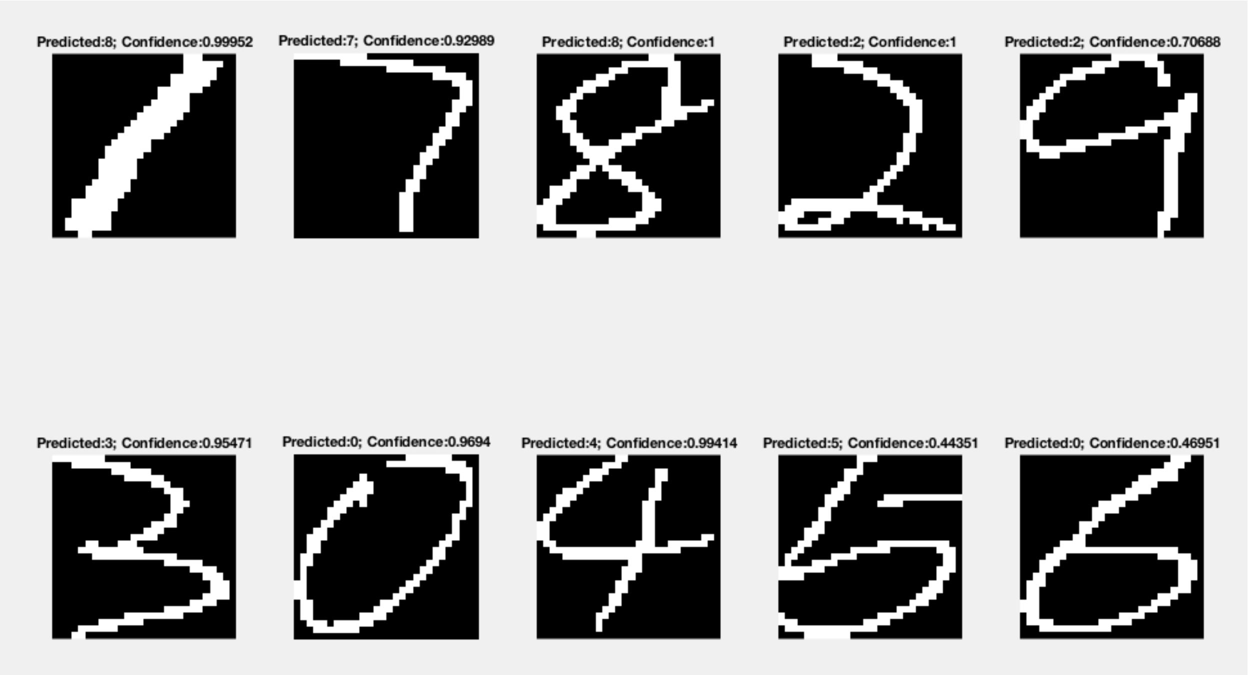

Predicted:8; Confidence:0.99952
Predicted:7; Confidence:0.92989
Predicted:8; Confidence:1
Predicted:2; Confidence:1
Predicted:2; Confidence:0.70688
Predicted:3; Confidence:0.95471
Predicted:0; Confidence:0.9694
Predicted:4; Confidence:0.99414
Predicted:5; Confidence:0.44351
Predicted:0; Confidence:0.46951

Figure 6. Cropped and predicted images (Xiaoli)