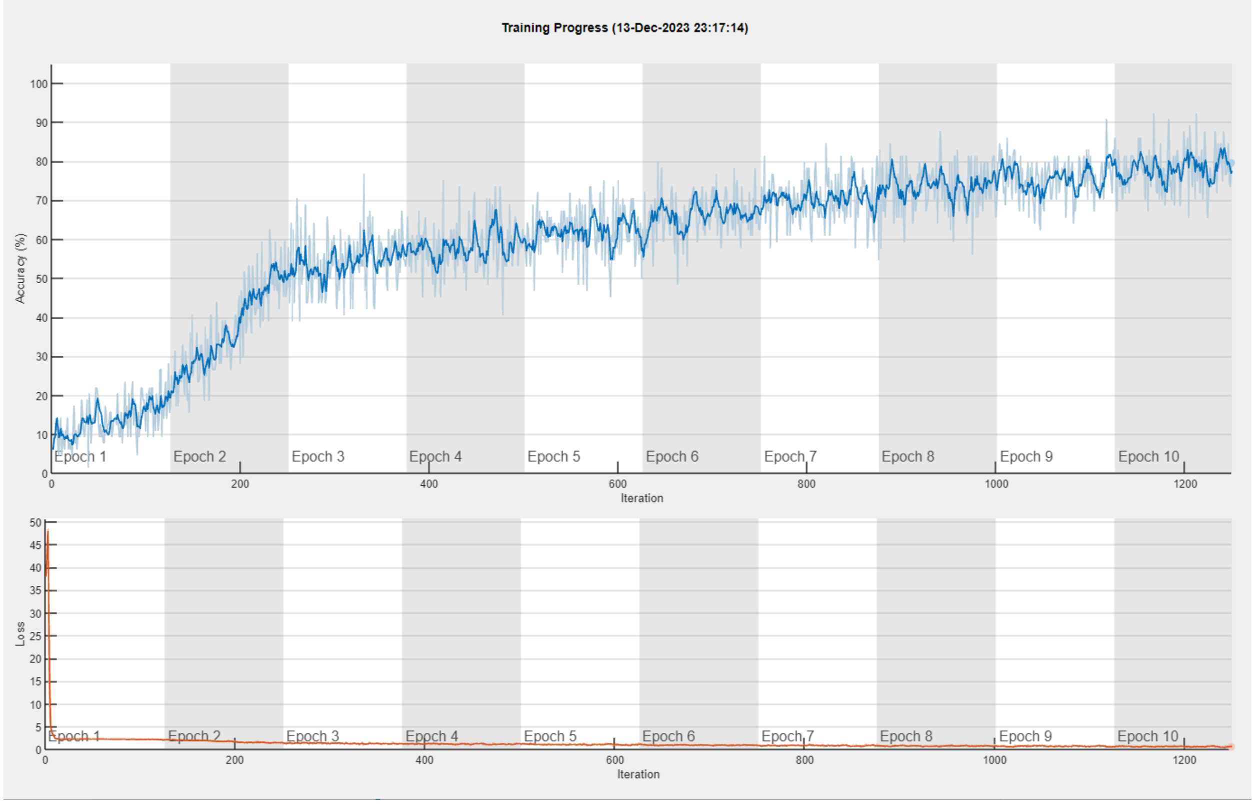
I imported the MNIST dataset. I wasn't sure if we could've just used the previous data from the mat files provided for previous assignments, so I started from scratch by importing it:

```
fullfile(matlabroot,'toolbox','nnet','nndemos','nndatasets','DigitDataset');
imds = imageDatastore(dataFolder, ...
   'IncludeSubfolders',true,'LabelSource','foldernames');
```

I then scripted for the dataset to be trained and saved as a classifier:

```
fullfile(matlabroot,'toolbox','nnet','nndemos','nndatasets','DigitDataset');
imds = imageDatastore(dataFolder, ...
   'IncludeSubfolders',true,'LabelSource','foldernames');
rng(1);
[trainData, testData] = splitEachLabel(imds, 0.8, 'randomized');
layers = [
   imageInputLayer([28 28 1])
   convolution2dLayer(3, 8, 'Padding', 'same')
   reluLayer
   maxPooling2dLayer(2, 'Stride', 2)
   convolution2dLayer(3, 16, 'Padding', 'same')
   reluLayer
   maxPooling2dLayer(2, 'Stride', 2)
   fullyConnectedLayer(10)
   softmaxLayer
   classificationLayer
];
options = trainingOptions('sgdm', ...
   'MaxEpochs', 10, ...
   'MiniBatchSize', 64, ...
   'InitialLearnRate', 0.001, ...
   'Shuffle', 'every-epoch', ...
   'Verbose', true, ...
   'Plots', 'training-progress');
% Train
net = trainNetwork(trainData, layers, options);
% Save classifier
save('mnist_classifier.mat', 'net');
% Make predictions on the test set
predictedLabels = classify(net, testData);
% Compare predicted labels with actual labels
accuracy = sum(predictedLabels == testData.Labels) / numel(testData.Labels);
fprintf('Accuracy on the test set: %.2f%%\n', accuracy * 100);
```



I’m not sure if I had to use apps like classification learner or deep network designer as I couldn’t find a way to export the dataset as a variable in the workspace for those apps to use it. I used matlab’s documentation for the script above: (https://www.mathworks.com/help/deeplearning/ug/create-simple-deep-learning-network-for-classification.html)