

# HW4 — MNIST Classification using PyTorch

## 1. MLP for MNIST Classification

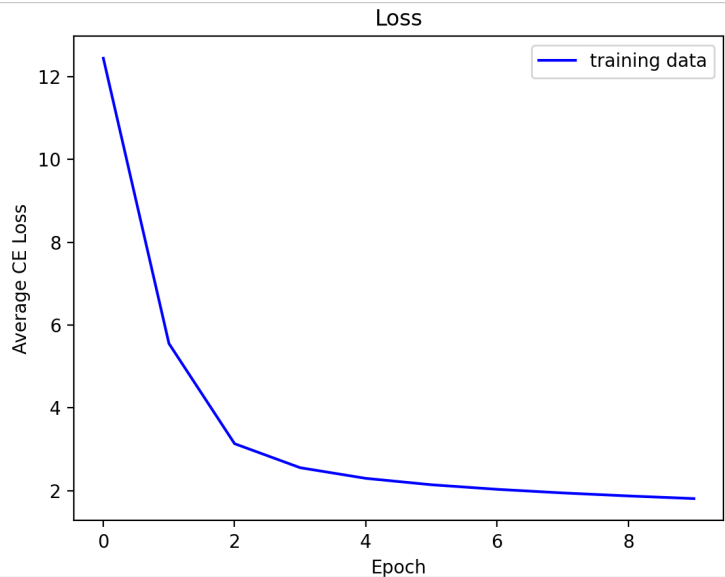
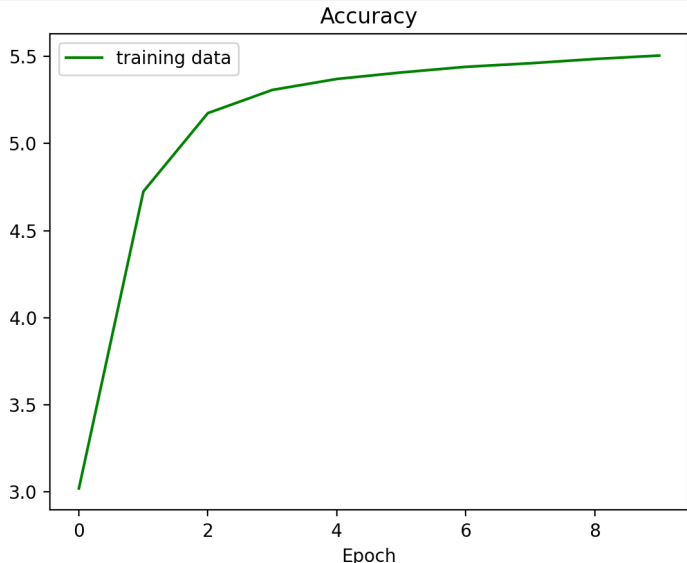
A MLP was trained on the MNIST dataset with the following architecture:

- **FC → ReLU → FC → ReLU → FC → Softmax → Cross-Entropy Loss**
- The two hidden layers have dimensions 456 and 128 respectively

The training parameters used are listed below:

- batch\_size = 100
- max\_epoch = 10
- learning\_rate = 0.001
- weight\_decay = 0.005

The final training and test performance is recorded below

	Training Data	Test Data
<b>Loss</b>	<b>0.301</b> 	n/a
<b>Accuracy</b>	<b>0.9160</b> 	<b>0.9177</b>

## 2. CNN for MNIST Classification

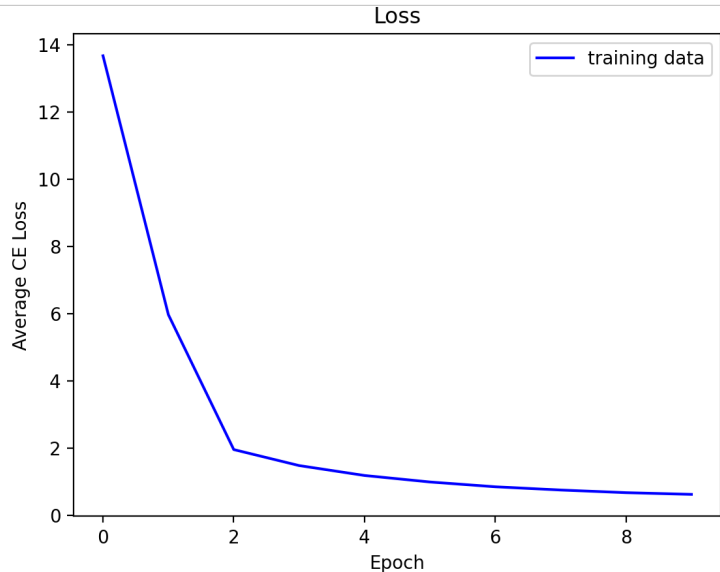
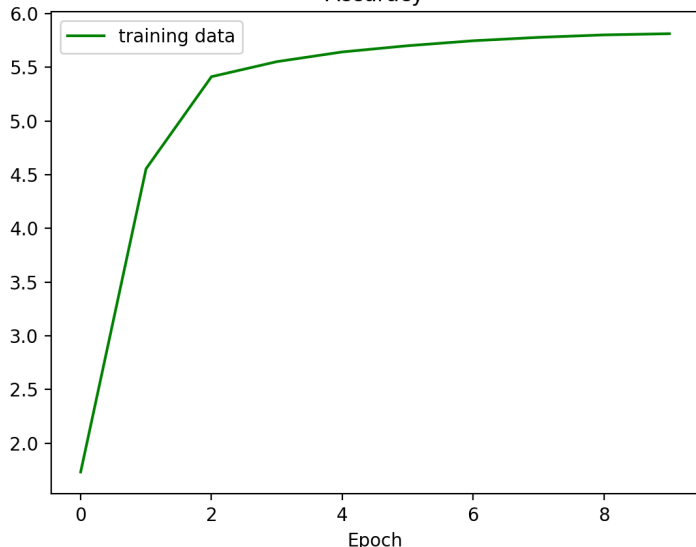
A CNN was trained on the MNIST dataset with the following architecture:

- **Conv → ReLU → MaxPool → Conv → ReLU → MaxPool → Flatten → FC → ReLU → FC → Softmax → Cross-Entropy Loss**
- The output channel sizes for the two convolution layers are 8 and 16 respectively
- A square 5x5 kernel was used for valid convolution (using padding of size 2)
- The single hidden layer has size 128

The training parameters used are listed below:

- batch\_size = 100
- max\_epoch = 10
- learning\_rate = 0.001
- weight\_decay = 0.005

The final training and test performance is recorded below

	Training Data	Test Data
<b>Loss</b>	<b>0.088</b> 	n/a
<b>Accuracy</b>	<b>0.9716</b> 	<b>0.9732</b>

### 3. CNN vs MLP

Let us compare the performance between a 2-layer multilayer perceptron and the convolutional neural network. For both architectures I have used the same optimizer as well as training parameters. From the performance data listed above, it is apparent that the CNN outperforms the MLP. Now let us compare the performance of each model with respect to the aspects below:

#### 3.1 Number of Layers

Increasing the number of FC/Conv layers in both the CNN and MLP slightly increase the performance of the models. This could be a result of the models having more power to model the complexities of the model. However, if the number of layers is increased significantly then the model actually performs slightly worse. This could be a result of vanishing gradients. A solution to this problem is proposed in the paper for ResNet.

#### 3.2 Kernel Size

For the CNN, I have tested two different kernel sizes (3 and 5). The larger kernel size (5) outperformed the smaller kernel size in terms of both training and test accuracy, and also converged at a faster rate. An intuitive reason for this is that a small kernel size ( $=1$ ) would be equivalent to a regular FC rather than a Convolution layer, and since we know CNNs outperform MLPs a larger kernel size would outperform a smaller one. However, there is probably some threshold where increasing the kernel size would produce no further gains.

#### 3.3 Activation

I have tried using sigmoid activations for both models and they both produced lower training and test accuracy. This is again related to the vanishing gradient problem. These results are corroborated by homework's 2 and 3.

#### 3.4 Loss

Softmax Cross-Entropy outperforms Euclidean Loss. See previous homework 2 for more details.