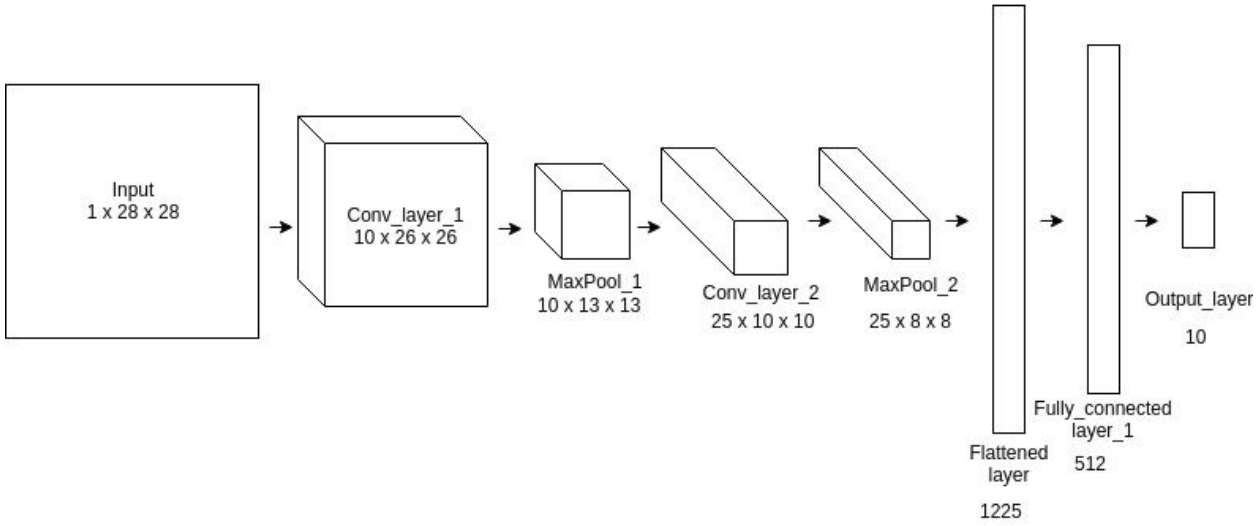
Problem 2

1. The CNN has the following architecture:

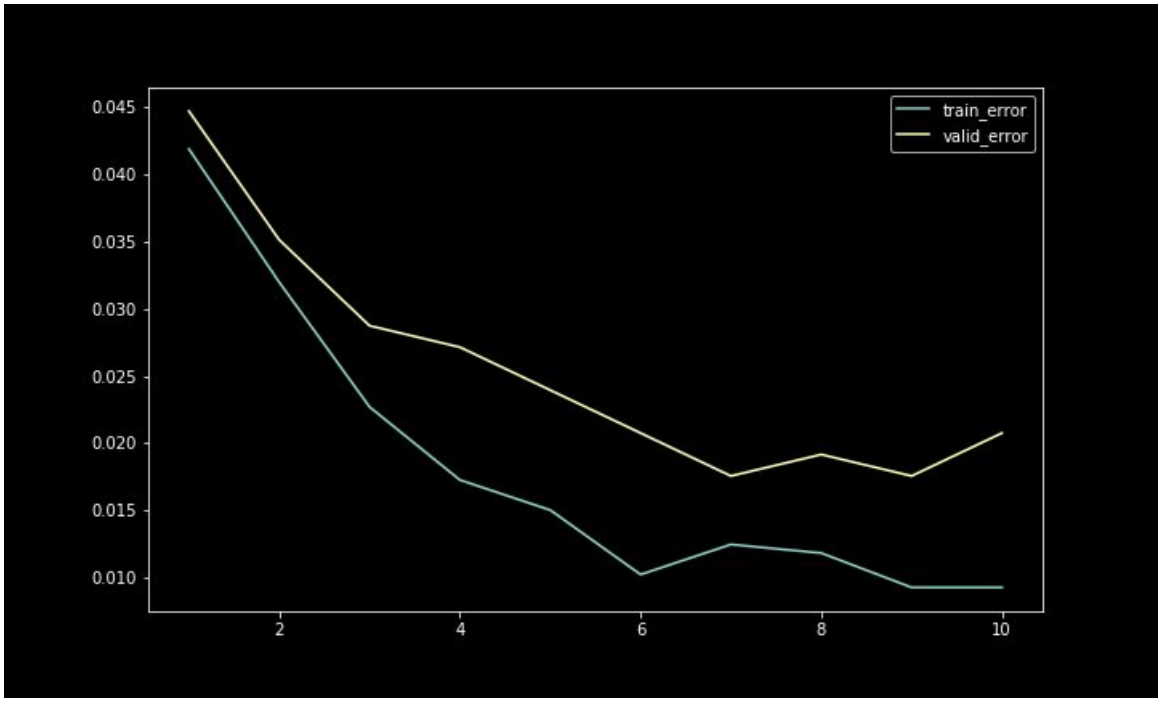


- Input image convolves on 10 3x3 kernels
- Conv_layer_1 go through a max pooling with window_size=2, stride=2
- Then it convolves on 25 4x4 kernels
- Conv_layer_2 go through a max pooling with window_size=3, stride=1
- Flatten the feature maps to 1-d vector
- The flattened layer then connects to one hidden fully-connected layer
- Fully_connected_layer_1 fully connects to the output layer

- Activation functions for hidden layers: ReLU
- Loss function: cross entropy

- # of parameters = 3x3x1x10 + 4x4x10x25 + 25x8x8x512 + 512x10 + 512 + 10
  = 828,932

2. Training and validation error curve



x-axis is the number of epoch, y-axis is the error

For the MLP and CNN we used the same hyper-parameters:

- learning_rate=0.01
- batch_size=32
- activation functions for hidden layers: ReLU
- loss function: cross entropy

As we can see in the graph, the errors on both training and validation set decrease very quickly with CNN. After 1 epoch of training, the validation error got 4.5%. After around 7 epochs, it reached the smallest validation error 1.76%. Then it starts to overfit after 7 epoches.

Compare to the MLP we did in Problem 1(NN structure: 512 * 256, with epoch: 10, minibatch: 32, learning rate: 0.01, activation function:ReLU), we see that CNN converges much faster than MLP. After the first training epoch, the validation error of MLP is 8.6%. The validation error of MLP at epoch 10 is 3.07% and it curve is still decreasing which means it still learns from the training set without overfitting. From this comparison we can see the advantage of using CNN over MLP for image classification problem, as the convolution kernel can preserve the structure of the image but MLP cannot.

Reference:

The validation accuracy curve of MLP