# CSE 151 Homework 4

Erin Werner

March 23rd, 2019

## 1 Homework 4 Response: Option 1

Supervised Kaggle Competition Username: **ErinWerner**

**Comparing Deep vs Shallow Neural Nets**

System 1 (Deep):

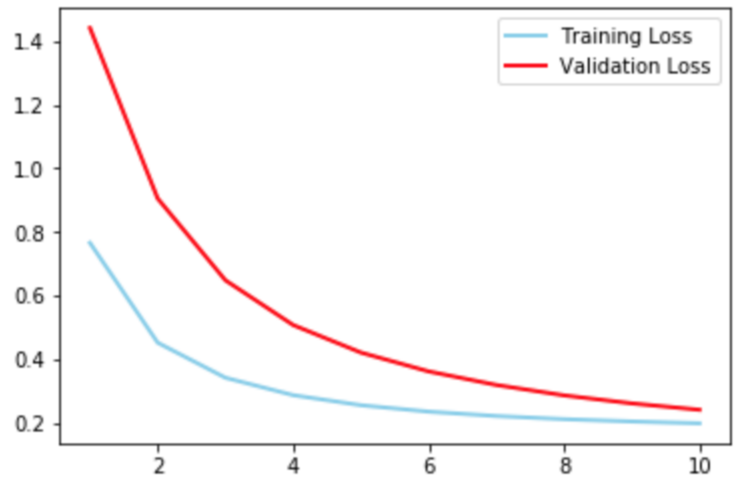
Hyperparameters that worked best:
numEpochs = 10
numClasses = 10
batchSize = 100
learningRate = 0.001

Test Accuracy = 96%

For my deep neural net model, I used three convolution layers and a fully connected layer. Each convolution layer is followed by ReLU activation function and a max pooling layer. My first layer implements a convolution that takes 1 input, uses 32 filters, has a kernel size of 5, with a stride of 1 and padding of 2. Then, it implements a ReLU and a max pooling layer with a kernel size and stride of 2. My second layer implements a convolution that takes 32 inputs, uses 32 filters, has a kernel size of 5, with a stride of 1 and padding of 2. Then, it also implements a ReLU and a max pooling layer with a kernel size and stride of 2. My third layer implements a convolution that takes 32 inputs, uses 64 filters, has a kernel size of 5, with a stride of 1 and padding of 2. Once again, it then implements a ReLU and a max pooling layer with a kernel size and stride of 2. My output is reshaped before implementing the fully connected layer, which takes a 3*3*64 input and has 10 possible classification outputs.

This resulted a very accurate MNIST classifier (96% accurate). Each epoch generated results with a loss lesser than the epoch before it. So, the model improved with each cycle of forward/back propagation. The loss values will eventually converge, making the model a stable one. As both the training and validation sets have a low loss and the test set has a high accuracy, it shows that my model is not overfitting to the data. Therefore, my model is generalizable.

Training and Validation Loss Results of 4-layer CNN:




System 2 (Shallow):

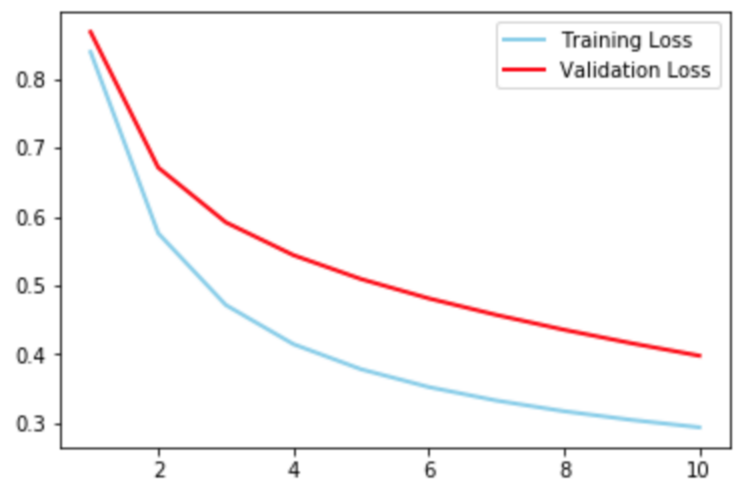
Hyperparameters that worked best:
numEpochs = 10
numClasses = 10
batchSize = 100
learningRate = 0.001

Test Accuracy = 84%

For my shallow neural net model, I used one convolution layer and a fully connected layer. The convolution layer is followed by ReLU activation function and a max pooling layer. My first layer implements a convolution that takes 1 input, uses 32 filters, has a kernel size of 5, with a stride of 1 and padding of 2. Then, it implements a ReLU and a max pooling layer with a kernel size and stride of 2. My output is reshaped before implementing the fully connected layer, which takes a 14*14*32 input and has 10 possible classification outputs.

This resulted a somewhat accurate MNIST classifier (84% accurate). Each epoch generated results with a loss lesser than the epoch before it. But the training and validation loss decreased at different rates, getting farther apart with each epoch, as shown in the plot below. Still, the model improved with each cycle of forward/back propagation. As both the training and validation sets have a low loss and the test set has a somewhat high accuracy, it shows that my model is not overfitting too much to the data. Therefore, my model is pretty generalizable, but not as accurate as it could be.

Training and Validation Loss Results of 2-layer CNN:




It is clear from the Training/Validation Loss plots and the Test Accuracies that the deep Convolutional Neural Network (CNN) performs better than the shallow CNN. The test accuracy represents the results of the model's predictions compared to the true data. The deep net with four layers, two more than that of the shallow net, creates a model that is 12% more accurate than the shallow net on the test data. This is a significant increase in accuracy, indicating that the deep CNN was much better than the shallow CNN at distinguishing the different classifications of the MNIST dataset.

Additionally, the Training/Validation curves reveal a lot about the deep vs shallow net behavior. The plots reveal the training and validation loss after each epoch, or iteration, of the model. The lower the loss, the better a model, unless the model has over-fitted to the training data. Both models have a low training and validation loss and relatively high accuracies, meaning that the models did not over-fit to the training data. Unlike accuracy, loss is not a percentage. It is a summation of the errors made for each example in training or validation sets. For the deep net, the validation loss is initially much larger than that of the training loss. Yet, the loss values grow closer with each epoch. The two loss curves converge together, indicating that the 4-layer CNN is very stable. However, for the shallow net, although the loss is reduced with each iteration, the loss values of the training and validation set grow farther apart. This means that the model slightly over-fitted to the training data and will then take longer to converge. As a result, it is not as stable or accurate as the deep net.