Intelligent Systems HW3 Part 2

# Problem Description

Given a data set of 5000, 28x28 pixel hand written digits, develop a neural network to classify the digits. Use 4000 images for training, and 1000 images for testing.

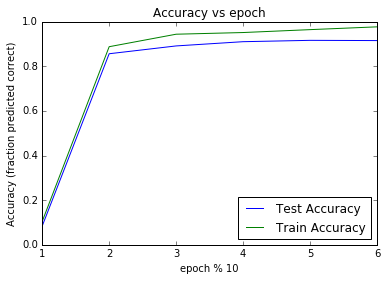
# Network Code Description

The neural network code was developed so that an arbitrary number of layers can be used. New layers are added to the model using the add() function, and can specify size, learning rate, and momentum. The ability to use a different loss function and optimizer for the network was added, but is currently set to stochastic gradient descent and a least mean square loss function. Each layer is represented by a “Layer” class, and contains a numpy matrix representing the input weights to each neuron, as well as parameters defining the layer such as learning rate and momentum. Each layer is independent of the other layers, and the model class controls the propagation of data through the layers to classify, as well as to backpropogate errors. The initial weights are chosen at random between -0.05 and 0.05 for each layer.

Through much trial and error, the final neural network was set with an input of 784 inputs, 200 neurons in the hidden layer, and 10 neurons in the output layer. The learning rate for the first layer is .01, and the rest of the layers have a learning rate of 0.005. All layers have a momentum of 0. The total number of training epochs was set at 60 epochs, and the batch size for training was set as a random sampling of 2000 images. The image labels 0 through 9 are encoded in a one-hot encoding for training. The predictions given by the network are also in a one-hot encoding form. As an example, 1 is encoded as [0, 1, 0, 0, 0, 0, 0, 0, 0, 0]. It was decided to stop training at 60 epochs because this was where the training and test accuracy started to diverge. The training error continued decreasing, while the testing error started increasing, indicating overtraining.

# Results

The final network error on the training set was 97.77% and the final network error on the test set was 91.8%. It took 8 minutes to train the network on an 8 core machine.



# Analysis of Results

Describe, discuss, and interpret the results you got, and why you think they are as they are.

Initially, the network was not changing accuracy enough, and continually had an accuracy of 10%. So, the learning rate was decreased. The initial momentum was also decreased to 0. As the learning rate was decreased, the accuracy started to change more, but didn't adjust enough. So, the network was made more expressive through adding more layers and larger layers. The data input size for training was also reduced and shuffled during every epoch. After adjusting the network size, the training and testing error started to increase. The network with one hidden layer maxed out at 91.8%. The network performance could potentially be made better by increasing the number of layers and reducing the size of each layer, while also reducing the number of connections between neurons in consecutive layers. This would allow for more local feature learning, as well as less of a chance of over training. The current network architecture is limited by the issue of over training. To reduce this concern, the number of parameters in the model could be decreased, along with further optimization of the network parameters.

# Code