John Geddes

CS 460G

Homework 4 Report

March 23, 2018

**Implementation Decisions**

I decided to use Python 2.7 in this project. There was no particular reason other than it was already installed on my computer.

For this project, I implemented a multilayer perceptron to predict a handwritten binary number. I decided to only use one hidden layer with 3 hidden nodes just to keep it simple. I also implemented a bias for the input layer and the hidden layer, even though they weren’t needed for this dataset. I decided to use lists as my data structures, despite knowing that the *numpy* library is much faster. My program works perfectly, albeit a little slow, so I didn’t want to go through the effort to convert my code from lists to *numpy*.

These lists are explicitly described in the code, but I will briefly explain some of them here. The training data is stored in two arrays: one that is a list of all the expected outputs, and one that is a 2D list storing all the rows in the dataset. The testing data is stored the same way as the training data. The weights of the entire network are stored in a 3D array, with network layer, node, and weights to the next layer as indices. Before beginning training, all of the weights are randomly assigned between -1.0 and 1.0. I used the sigmoid function as my activation function.

I used the *tqdm* library to implement progress bars when running for-loops on the 12000+ training data points and 2000+ test data points. I did this because the loop was very slow, and I couldn’t tell if the code was hanging or actually running. I made two different stopping criteria for the training of the network: a maximum number of epochs to train over, and a minimum percent of predictions that need to be correct. Those criteria are 10 epochs and 99% accuracy. I also picked a learning rate of 0.005.

After running the network on the training data, these were the results:

Current epoch: 1

10795 out of 12665

85.23 %

Current epoch: 2

12480 out of 12665

98.54 %

Current epoch: 3

12532 out of 12665

98.95 %

Current epoch: 4

12552 out of 12665

99.11 %

Number of epochs taken: 4

Running test

2101 out of 2115

99.34 %

As you can see, the training took 4 epochs to reach at least 99% accuracy when predicting the training data. Then the program ran the test data and predicted 99.34% correctly.