Jeter Gutierrez, Ronuel Diaz
CSC448: Artificial Intelligence
Due: May 21st, 2018
Prof. Lucci

Program 4 Chinese Perceptron

The Purpose of this assignment is to program a single layer perceptron and teach it to classify six Chinese characters. There were a lot of different ways to get this assignment done. We decided to build everything from the ground up to get a better understanding behind the concept of supervised learning. Our language of choice is called *Processing*.

The program is made up of 4 main functions:

- *train()*: The main function that trains the perceptron for a number of iterations that has been specified by the user.
- *learn*(): The implementation of the perceptron learning rule. This function updates the weights associated with each output node when called.
- *testing()*: The function that allows users to input their own data to have it classified by the neural network.
- *convert()*: Converts an image into a binary array.
- *sigmoid()*: The activation function.

We started by loading our training data and our learning data into the program. From here, our code loops through each image of our training data and calculates the dot product between the binary vector and the weight vector that is associated with the output node. The bias is also added on to the resulting sum. After this step, we send the result into our activation function which returns an activation value for the specific output node. If this image correlates to the image from the learning data then the activation function should output a value close to 1.

Afterwards our code will update the weight vector and this will continue for many iterations specified by the user.

The training data was hand drawn and saved into the training folder. The training data folder contains around 430 images of sample data. We chose to hand draw our sample data using different styles and different character placement on the screen. This led to an increase in our accuracy from less than 30% to around 70%. We would even say that it reached 95% accuracy.

Another way to optimize our results would be to have both more hidden layers and a larger dataset. The time it took to train our neural network would end up increasing if we were to implement the above optimization strategies.