B551 Artificial Intelligence: Assignment 4 Part 2 Report

Team members: Piyush Rai and Yee Sern Tan

Step 1. Nearest Neighbor

The square of the Euclidean distance is used against all images in the training set to compute the nearest neighbor. Minimizing a positive distance is same as minimizing its square. The detected orientation is taken as the nearest neighbor's. Using the z-scores of each entry the success rate increases. Z-score can be explained by taking the entry, subtract from it the mean, and then divide over the standard deviation of the image.

Step 2. AdaBoost

- The stumps are chosen without replacement for 192*191/2 possible pairs of distinct elements from a (192, 192)-sized tuple. The first and second elements of the pair correspond to coordinates of elements in the upper diagonal of a 192 x 192 matrix.
- The difference in the value of the image evaluated at these two elements are computed. The difference is divided over the standard deviation of the image.
- The results over all training images are collated into a statistic, where the optimal split is computed to minimize the entropy. The split with the minimal entropy is selected for the hypothesis at each step. Hypotheses weights are computed and image weights are updated
- This process is computationally independent for each orientation, but to save time the statistic is only computed once for all orientations.
- The weighted majority – the maximum of the sum of hypothesis weights where classification is correct – is taken to be the classification result.
- To compensate for time limit, the size of training set scales inversely with stump count keyed in at the input.

Step 3. Neural Nets

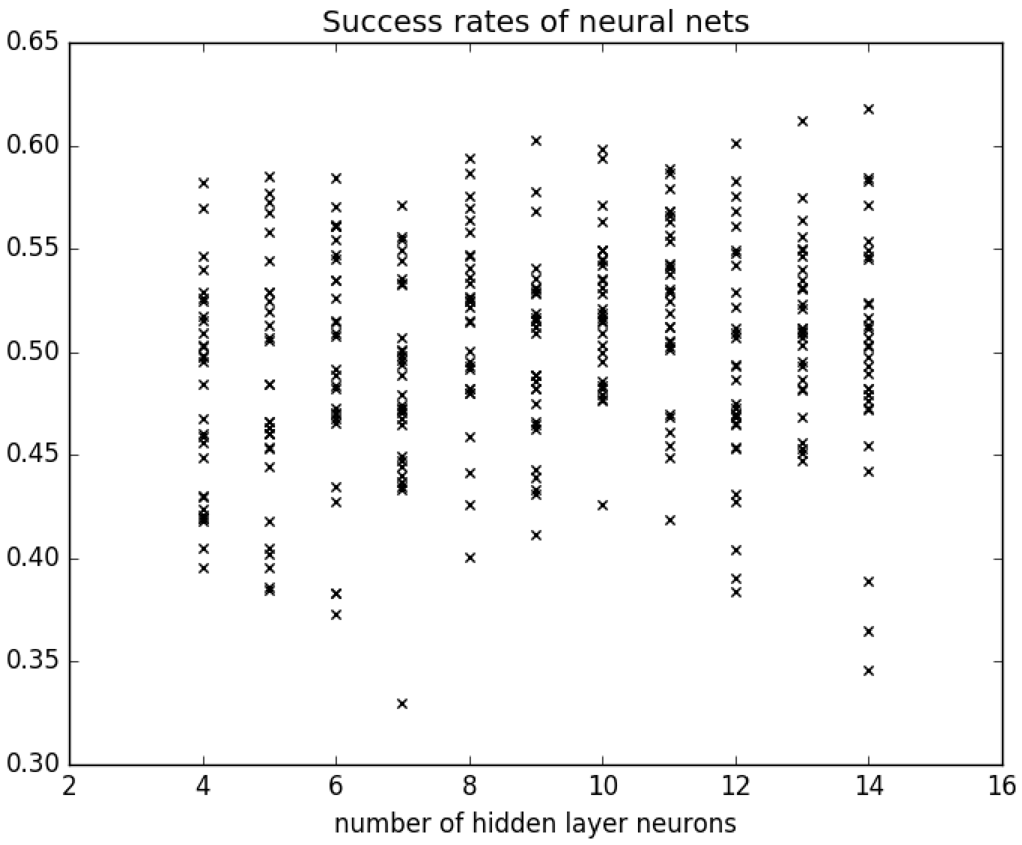
- Help is consulted from: Efficient BackProp, Y. LeCun et al., 2002, in Neural Networks: Tricks of the Trade.
- We assume that brightness is statistically independent of orientation. Therefore the brightness is scaled by subtracting the mean in the image, and then dividing over the standard deviation of the image.
- The activation function is chosen to be tanh.
- The weights are chosen according to a normal distribution with mean 0 and standard deviation m^(-0.5), where m is the number of neurons in the layer.
- Stochastic gradient descent is used because it typically converges faster in training.
- Lower learning rates can converge closer in the end, but start slower. Starting out with small number of iterations, the optimal learning rate is to try to be minimal while ensuring early convergence. It is obtained with an interactive input of learning rate parameters. The resulting learning rate is 0.5/t for the hidden layer and 0.35/t for the output layer, where t is the time step of iterations.
- The target values are empirically tested to yield best results at -0.99 and 0.99 for 10,000 gradient descents (the tanh activation function asymptotes at -1 and 1). For higher iteration

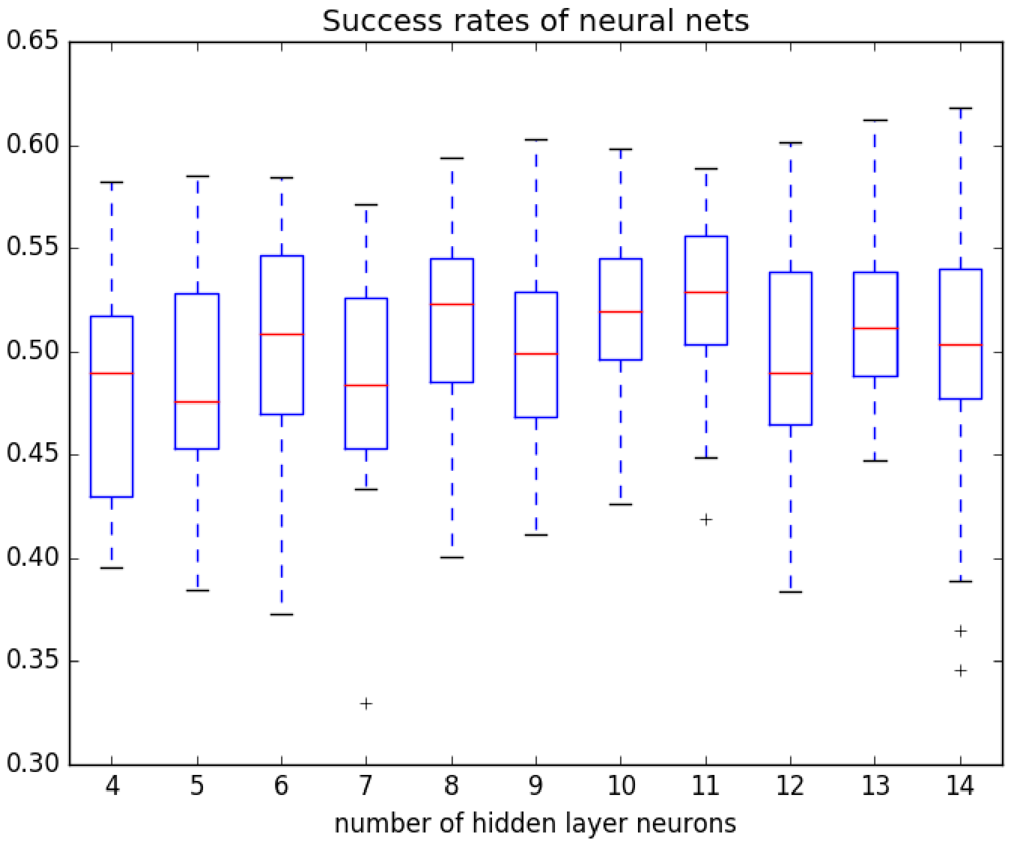
of gradient descents, it did not work well, but can be patched by adjusting the target values to -0.995 and 0.995.

Step 4. Analysis and Improvement

Now that although nearest neighbor has the highest classification score, it can hardly learn a model that can be represented economically. So, it is excluded. In our preliminary analysis neural nets have the potential to beat AdaBoost; so it is chosen.

- Keeping the training time roughly constant, the number of iterations varies inversely proportional to the "hidden count" (the number of neurons in the hidden layer). The product of number of iterations with hidden count is fixed at 100,000 to have time for generating statistics. For each value of the hidden count, ranging from 4 to 14, the algorithm is trained and tested 30 times. (Training set and testing set are the whole collection provided.) The proportions of correct classifications are plotted with a scatter plot and box plot (see plots in next page).
- It is found that at 11 the output is optimal, that is, with high mean and low standard deviation. Low values of hidden count have lower mean, while high values have high standard deviation, which could possibly be due to lower number of iterations.
- When the resulting mean and standard deviation of classification scores are equally preferable, simpler models with lesser number of hidden counts is preferred.


Success rates of neural nets
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
2
4
6
8
10
12
14
16
number of hidden layer neurons


Success rates of neural nets
0.65
0.60
0.55
0.50
0.45
0.40
0.35
0.30
4
5
6
7
8
9
10
11
12
13
14
number of hidden layer neurons