Assignment 4: Introduction to Neural Networks

To get our feet wet with neural networks, we are going to walk through portions of the process. You are do to this without using numpy.

You are given a data set with 5000 handwritten digits and their corresponding labels. Each training example is a 20 pixel by 20 pixel grayscale image of the digit. Each pixel is represented by a number indicating the grayscale intensity at that location. Thus, your neural network will have 400 inputs.

Your network will have 3 layers: an input layer with 400 inputs, output layer with 10 outputs (corresponding to the ten digits), and a hidden layer with 25 units. You will add bias units at the first and second layers. Thus between the first and second layers there are 401*25 = 10,025 weights. Between the second and third layers there are 260 weights. The total number of weights is therefore 10, 285. You are also provided with the set of weights to use for this assignment.

1) Implement a neuron unit: Write a function that takes as input the activations from the previous layer and the input weights for that that layer, and returns the activation value for that neuron.

2) Using the function in (1), write a function that takes input $\mathbf{x}^{(i)}$ of dimension 400 and the 10,025 weights and outputs the 10 output values of the neural network as a list.

3) Using the function in (2), write a function that classifies an image as a number between 0 and 9. Thus the output of the function is an integer between 0 and 9.

4) Use the function in (3) to classify all 5000 digits in the data set. What is the error rate?

5) Write down the MLE cost function for this neural network. Write a program to evaluate the cost with the given weights and with the 5000 test examples.

6) In order to find the optimal weights, we need to take the partial derivatives of the cost function. For this we use back propagation. Provide pseudo-code showing how we calculate the partial derivatives using back propagation. (No actual coding is needed here.) Be fully detailed with the equations.

Extra credit:

7) Write code to calculate the 10,025 partial derivatives. Calculate the mean, median, and standard deviation of the absolute value of the partial derivatives.

8) For each weight w obtain a random r number in the range [0.9w, 1.1w] and replace w with w+r. (You will need to make 10,025 replacements with 10,025

random numbers.) Using these new test examples, repeat problems (6) and (8). Comment on how your results changed.

9) Repeat (1)-(7) using numpy.