Portland State UNIVERSITY

CS 445/545: Machine Learning, Spring 2022

Programming Assignment #1

A. Rhodes

Note: This assignment is **due by Sunday, 4/24 at 10pm**; you will turn in the assignment by email to our grader, as instructed below.

Each student will turn in an individual assignment (so that we have something upon which to base your individual grade). However, you are encouraged to discuss and work through any issues related to this assignment with your instructor, TA and above all, other students in our class. Bottom line: you will code the assignment yourself; the assignment results/code will be the product of your individual work.

For this homework you will implement a two-layer neural network (i.e, one hidden-layer) to perform the handwritten digit recognition task of Homework 1. Please write your own neural network code; don't use code written by others, though you can refer to other code if you need help understanding the algorithm. You may use whatever programming language you prefer.

The dataset for this task is the MNIST dataset that you used in Homework 1.
https://www.kaggle.com/oddrationale/mnist-in-csv

Neural network structure: Your neural network will have 784 inputs, one hidden layer with n hidden units (where n is a parameter of your program), and 10 output units. The hidden and output units should use the sigmoid activation function. The network should be fully connected —that is, every input unit connects to every hidden unit, and every hidden unit connects to every output unit. Every hidden and output unit also has a weighted connection from a bias unit, whose value is set to 1.

Task: Each output unit corresponds to one of the 10 classes ('0' to '9'). Set the target value t_k for output unit k to 0.9 if the input class is the kth class, 0.1 otherwise.

Network classification: An example **x** is propagated forward from the input to the output. The class predicted by the network is the one corresponding to the most highly activated output unit. The activation function for each hidden and output unit is the sigmoid function.

Network training: Use back-propagation with stochastic gradient descent to train the network. Include the momentum term in the weight updates, as described in the lectures. Set the learning rate to 0.1 and the momentum to 0.9.

Preprocessing: Scale the data values to be between 0 and 1 by dividing by 255.

Initial weights: Your network should start off with small ($-.05 < w < .05$) random positive and negative weights.

Experiment 1: Vary number of hidden units.
Do experiments with n = 20, 50, and 100. (Remember to also include a bias unit with weights to every hidden and output node.)

For each value of n, train your network on the training set, as described above, changing the weights after each training example. After each epoch, calculate the network's accuracy on the training set **and** the test set for your plot. Train your network for 50 epochs. In your report, give a plot of both training and test accuracy as a function of epoch number (graph both of these in the same plot).

After training is complete, create a confusion matrix for each of your trained networks, summarizing results on the test set.

Discuss your results in a paragraph in your report. Include answers to the following questions:

(1) How does the number of hidden units affect the final accuracy on the test data?

(2) How does it affect the number of epochs needed for training to converge?

(3) Is there evidence that any of your networks has overfit to the training data? If so, what is that evidence?

(4) How do your results compare to the results obtained by your perceptron in HW 1?

Experiment 2: Vary the momentum value. Here, fix the number of hidden units to 100, and vary the momentum value during training. Use momentum values of 0, 0.25, and 0.5. Train networks using these values as in Experiment 1, and plot the results as in Experiment 1. (Include your plot for n=100 and momentum = 0.9 that you completed in Experiment 1.)

Create a confusion matrix for each of your trained networks, summarizing results on the test set.

Discuss your results in a paragraph in your report. Include answers to the following questions:

(1) How does the momentum value affect the final accuracy on the test data?

(2) How does it affect the number of epochs needed for training to converge?

(3) Again, is there evidence that any of your networks has overfit to the training data? If so, what is that evidence?

Experiment 3: Vary the number of training examples. In this experiment, fix the number of hidden units to 100 and momentum 0.9. Instead of using all of the training examples, train two networks, using respectively one quarter and one half of the training examples for training. Make sure that in each case your training data is approximately balanced among the 10 different classes. Plot the results, as in the previous experiments, plotting accuracy on both the training and test data at the end of each epoch.

Create a confusion matrix for each of your trained networks, summarizing results on the test set.

Discuss your results in a paragraph in your report. Include answers to the following questions:

(1) How does the size of the training data affect the final accuracy on the test data?

(2) How does it affect the number of epochs needed for training to converge?

(3) Again, is there evidence that any of your networks has overfit to the training data? If so, what is that evidence?

Report: Your report should include a short description of each experiment, along with the plots and discussion paragraphs requested above.

Here is what you need to turn in:

- Your report.
- Your well-commented code.

How to turn it in (read carefully!):

- Send these items in electronic format to our TA by the due date. No hard copy please!
- The report should be in pdf format and the code should be in plain-text format.
- Put "MACHINE LEARNING PROGRAMMING #1" in the subject line.

If there are any questions, don't hesitate to ask me or the grader.