CS 445/545

Machine Learning

Winter 2016

Homework 2: Neural Networks

Due Thursday, January 28, 2016

For this homework you will write code to implement a two-layer neural network (i.e, one hidden-layer) to perform the letter recognition task of Homework 1. Please write your own neural network code; don't use existing 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 **Letter Recognition** data set from the UCI machine-learning repository: http://archive.ics.uci.edu/ml/datasets/Letter+Recognition

Neural network structure: Your neural network will have 16 inputs (the attributes used in the letter-recognition data), one hidden layer with *n* hidden units (where *n* is a parameter of your program), and 26 output units. 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. Thus the total number of weights in the system is $43n$+26.

Task: Each output unit corresponds to one of the 26 classes (A−Z). The target value t_k for output unit *k* is 0.9 if the input class is the *k*th 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.

Training and test sets: Split the data into approximately equal-size training and test sets. Make sure the training set is shuffled into random order of examples.

Preprocessing: Scale the training data to have zero mean and unit variance along each column (i.e., along each feature). This is called *standardization* of the data, and helps avoid imbalance between features with different scales. In particular, let μ_i denote the mean value of feature *i* in the training data, and σ_i denote the corresponding standard deviation. For each training example **x**, replace each x_i as follows:

$$x_i' = \frac{x_i - \mu_i}{\sigma_i}$$

Scale the test data in the same way, using the μ_i and σ_i values computed from the training data, **not** the test data.

Initial weights: Your network should start off with small ($-.25 < w < .25$) random positive and negative weights.

Experiment 1: Set the learning rate η to 0.3, the momentum α to 0.3, and the number of hidden units n to 4.

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. Continue training until the network is 100% accurate on the training set, or for some maximum number of epochs (set by you), whichever comes first. 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). Is there evidence that your network has overfit to the training data? If so, what is that evidence?

Experiment 2: Repeat Experiment 1, but with a low learning rate ($\eta = 0.05$) and a high learning rate ($\eta = 0.6$). Provide plot of training and test accuracy as a function of epoch number for each of these (graphed in same plot). How does changing the learning rate change your results?

Experiment 3: Repeat Experiment 1, but with a low momentum rate ($\alpha = 0.05$) and a high momentum rate ($\alpha = 0.6$). (Set η back to 0.3.) Provide plot of training and test accuracy as a function of epoch number for each of these (graphed in same plot). How does changing the momentum change your results?

Experiment 4: Repeat Experiment 1, but with a smaller number of hidden units ($n = 2$) and a larger number ($n = 8$). (Set α and η back to 0.3.) Provide plot of training and test accuracy as a function of epoch number for each of these (graphed in same plot). How does changing the number of hidden units change your results?

Report: Your report should include a short description of each experiment, along with the plots requested above, and answers to the questions given 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 mm@pdx.edu by 2pm on 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 HW 2" in the subject line.

If there are any questions, don't hesitate to ask me or e-mail the class mailing list.

Policy on late homework: If you are having trouble completing the assignment on time for any reason, please see me before the due date to find out if you can get an extension. Any homework turned in late without an extension from me will have 5% of the grade subtracted for each day the assignment is late.