COM 312 - Neural Networks and Deep Learning

Assignment 3 – Training a Multi-Layer Network on the MNIST Database

---

In this Assignment, you will build on the second assignment to implement the Generalized Delta for a multi-layer Network. Using the same database as in Assignments 1 and 2 (the MNIST database), the connection architecture should follow Figure 1. Again, input to the network will come from 784 input nodes corresponding to the pixels of pictures of ten hand-drawn digit categories. The two hidden layers may each have a variable number of nodes. The output layer of the network will have two nodes. One output node should be trained to predict whether the input digit is odd or even (as in the last assignment) and the other output node should be trained to predict if the input digit is a prime number {0, 1, 2, 3, 5, 7}, or not. You may use any number of nodes you like in each hidden layer. Input to the network should be continuous values, rather than binary (we used binary input values in the first two assignments).

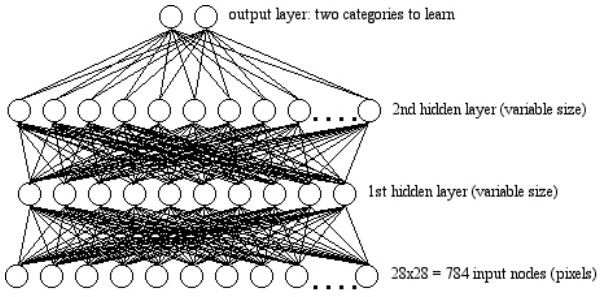

Figure 1: Multi-layer neural network for the MNIST database

Start by initializing all weights of the network to small random values. Then train the network through training epochs composed of training cycles. A training cycle is where the network's weights are updated in response to an input stimulus. A training epoch is complete once each of the 60,000 training patterns have been used to update the weights of the network. Because a single training epoch will usually not result in optimal training results, regardless of learning rate, multiple training epochs need to be run to reduce the error to about as low as it will go. After each training epoch, test how well the network predicts the correct categories using the 10,000 testing patterns. Use the same learning rules from the last assignment.

Deliverables:

1) Turn in your code (a single .c or .cpp file). Your code should compile and run on the classroom machines running linux. DO NOT include rand.h or nmist.h or the MNIST database. (50 points). If your program needs special flags to compile, please give example usage in the program header.

2) Provide a written analysis of your results including graphs - as a .pdf file. In your analysis, discuss how many hidden units you used in each hidden layer, and why (based on testing) you chose that number of units per hidden layer. This analysis should include results from testing with various sizes of hidden layers. You should also say what learning rate(s) you used, and if you used momentum or 'pre-training' of weights to improve your results (50 points).

Compress all of your files into a single .zip file that includes your name, the name of the class, and reference to Assignment 2. For example: COM312_Almaz_Ass2.zip. Also include you name in the header of your code and your name on the top of your written report.

Submit your .zip file (with .c or .cpp file, and .pdf file) on e-course.