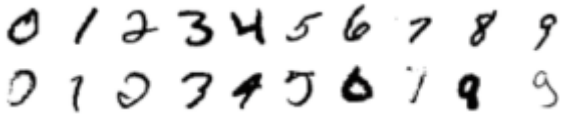
ITCS 6156 Fall 2016
Programming Homework 3
Points: 100

This assignment is to be completed individually. No group work is allowed.

Character Recognition Using Neural Networks



Description

The objective of this exercise is for you to gain practical experience in designing, implementing, training and optimizing a neural network to carry out a specific real world task.
The task is the Optical Recognition of Handwritten Digits, and the data sets you must use are provided by the UCI Machine Learning Repository.

Requirements

(1) Download the data sets and descriptions available on the assignment description page in Canvas. In the ITCS6156_HW3.zip archive, you should find the Optical Recognition of Handwritten Digits data set (also available from the original source http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits). This data set consists of preprocessed normalized bitmaps of handwritten digits from a preprinted form. From a total of 43 people, 30 contributed to the training set and different 13 to the test set. 32x32 bitmaps are divided into nonoverlapping blocks of 4x4 and the number of on pixels are counted in each block. This generates an input matrix of 8x8 where each element is an integer in the range 0..16. This reduces dimensionality and gives invariance to small distortions.
The three files in this archive are:
optdigits_test.csv: a comma delimited file of the training set
optdigits_raining.csv: a comma delimited file of the training set
optdigits_names.txt: contains a description of the data

(2) Design a Multi-Layer Perceptron neural network that can learn from the training data set to generalize well to the testing data set using the Back-Propagation learning algorithm.

(3) Implement your neural network, using any programming language and tools you like. You may use any code you find on the internet, as long as you properly reference the sources in your report, or you can program your own neural network from scratch. Hint: Writing your own code often proves easier than figuring out how to get someone else's code to do what you need, and you will probably learn more, but also keep in mind that there is limited credit/marks available for the implementation aspect of this exercise.

(4) Experiment systematically with the neural network details that you think are most likely to improve the generalization performance you achieve. Hints: Consider how you are going to avoid under-fitting and over-fitting of the training data, and what parameters are going to have the biggest effect on that. It is usually better to aim to optimize two or three things well, rather than a large number of things not very well.

(5) Write a report explaining what you did and what you found. You should include the following sections:

a. Introduction - Say what the data sets involved and what you aimed to achieve. [5%]
b. Design - Describe and justify the neural network you designed for the task, and the factors you decided to experiment with. [15%]
c. Implementation - Describe how you implemented your neural network and the associated performance analysis mechanisms. Explain why you chose to do it that way. Remember to cite any sources you used. [20%]
d. Experiments –
    i. Describe the experiments you carried out to optimize your network's generalization performance, and present the results you obtained.
    ii. Explain in detail how you used the training and testing data sets. Present the results in a statistically rigorous manner, e.g. by computing simple statistics of performance measures (such as means and standard deviations of percentages correct) across multiple runs of network training/testing, and plotting graphs with error-bars. [50%]
e. Conclusions - Summarize your key findings, including which factors proved most crucial, and what was the best generalization performance you achieved. [10%]

Bonus

Backpropagation is a notoriously difficult algorithm to debug and get right. Carrying out the derivative checking procedure will significantly increase your confidence in the correctness of your code.
As a bonus step for this homework you may choose to use gradient checking to numerically check the derivatives computed by your code to make sure that your implementation is correct. Document your procedure and calculations clearly and include it in your report. [25 bonus points - there will be no partial credit for this step]

Assignment Submissions

What to submit using Canvas (Email submissions will NOT be accepted):

1. **HW3_report.pdf** – PDF document with your write up for step # 5.
2. **HW3.zip** - An archive of the entire programming project stored in a standard ZIP file. Make sure to include all packages and libraries used to run your programs if any.
3. **README.txt** –This file should detail all the files in your project archive, libraries and packages used and any special setup you have used in your programming environment.
4. **INFO.pdf** – PDF document with the following assignment information:
    a) Explanation of status and stopping point, if incomplete.
    b) Explanation of additional functions and analysis, if any.
    c) Discuss the easy and challenging parts of the assignment. How did you overcome all or some of the challenges?