

Lab 3

CS 5233 – Fall 2007
Tom Bylander, Instructor

assigned November 1, 2007
due midnight, November 28, 2007

In Lab 3, you will implement a program to recognize handwritten digits. Your grade on the lab will depend on the behavior and accuracy of the program that is implemented.

NIST environment

The NIST training dataset at <http://yann.lecun.com/exdb/mnist/> contains 60,000 images of handwritten digits and their labels. Each image is 28×28 in grayscale. For reasons of time and space efficiency in this lab, an 8×8 bitmap was derived from each image.

The `nist` program repeatedly does the following steps. It outputs a randomly selected (without repetition) bitmap as 64 ones and zeroes on a single line. It then inputs a prediction, a single digit on a line by itself. After the prediction, it outputs whether the prediction is correct or incorrect, the correct label, and a summary of the error rate so far.

The `-v` option presents a crude ASCII picture of the bitmap so that you can see how well you can do.

Agent

Your task is to write a program that will perform online learning, i.e., its accuracy will improve as additional examples are presented to it. The suggested algorithm is called a *linear machine* (N. J. Nilsson, *Learning Machines*, McGraw-Hill, New York, 1965).

In a linear machine each class y is represented by a linear function h_y with bias weight w_{y0} and n attribute weights w_{y1}, \dots, w_{yn} :

$$h_y(\mathbf{x}) = w_{y0} + \sum_j^n w_{yj}x_j$$

The prediction of the linear machine is the class with the highest value of h_y .

$$h(\mathbf{x}) = \arg \max_y h_y(\mathbf{x})$$

To learn with the linear machine, suppose that the correct label is t . Then for each class $y \neq t$, if $h_t(\mathbf{x}) - h_y(\mathbf{x}) < 1$, perform the following update:

$$\begin{aligned} w_{t0} &\leftarrow w_{t0} + \alpha \\ w_{y0} &\leftarrow w_{y0} - \alpha \\ w_{ti} &\leftarrow w_{ti} + \alpha x_i \quad \text{for } 1 \leq i \leq n \\ w_{yi} &\leftarrow w_{yi} - \alpha x_i \quad \text{for } 1 \leq i \leq n \end{aligned}$$

where α is the learning rate. This will tend to minimize $\max(0, 1 - h_t(\mathbf{x}) + h_y(\mathbf{x}))$.

The class notes suggest a learning rate of $1/640$ for 64 inputs. A higher learning rate might result in faster learning or might result in unstable weights. Initially, all weights should be set to 0. The expected result is a decreasing error rate with additional examples.

Turning in Your Lab and Report

Somewhere in your directory, you should create a `lab3` subdirectory. This directory *must* have a `Makefile` (or *must* have an executable file `make_lab3`) that compiles and links your agent programs, which *must* be named `lab3`. Any source code that is linked to these programs *must* be in this directory. That is, you should be able to copy the files in your `lab3` directory to another directory and be able to run the following command sequence.

```
setenv A /home/bylander/agents/linux/bin or A=/home/bylander/agents/linux/bin
/bin/rm lab3
source /etc/.login revert to default values for shell variables
make or ./make_lab3
$A/interact $A/nist ./lab3
```

Zip or tar the entire directory and submit it using WebCT. If you submit multiple times, only the last one is kept. Be sure that WebCT has registered your submission.

In addition, provide a brief report describing the performance of your program.