

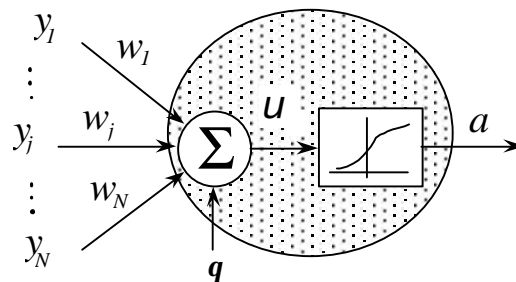
**ECE/CS/ME 539 Introduction to Artificial Neural Networks and Fuzzy Systems**

**Homework #1**

Total 100 points. Problem labeled with CC will not be graded on correctness, but will be graded on completion of work only. Homework must be submitted in typed written format on letter size papers. A cover page is needed and the set of pages should be stapled.

Topics covered: Introduction to ANN, learning theory, competitive learning, Hebbian learning, PCA network, error correcting learning, LMS algorithm, and perceptron

1. (20 points) *McCulloch-Pitts Neuron model*



Refer to the simple neuron model shown above to answer the following questions:

- (a) (8 points) Let  $N = 1$ , and  $y = y_1$ . If  $w_1 = 2$ , and  $q = -1$ , plot the following functions  $a(y)$ :
  - a.  $a(u) = 0.5u$
  - b.  $a(u) = \exp(-|u-1|^2)$
  - c.  $a(u) = 1/(1+\exp(-u))$
  - d.  $a(u) = \tan^{-1}(u)$

- (b) (4 points) Let  $N = 2$ ,  $a(u) = +1$  if  $u \geq 1$ , and  $a(u) = 0$  for  $u < 1$ . For  $w_1 = 2$ ,  $w_2 = -1$ , and  $q = 2$ , plot the function  $a(y_1, y_2)$  on the  $y_1$ – $y_2$  plane such that the region  $a(y_1, y_2)=1$  is shaded, and the region  $a(y_1, y_2)=0$  is clear.

- (c) (8 points, CC) Derive the derivatives  $df(x)/dx$  of activation functions  $f(x)$ :

Activation Function	$f(x)$	$df(x)/dx$
Radial Basis	$\exp[-\mathbf{a}/ x-x_o ^2]$	$-2\mathbf{a}(x-x_o)f(x)$
Sigmoid	$1/(1+\exp(-x/T))$	$(1/T)f(x)(1-f(x))$
Hyperbolic tangent	$\tanh(x/T)$	$(1/T)(1-(f(x))^2)$
Inverse tangent	$(2/\mathbf{p}) \tan^{-1}(x/T)$	$(2/(T\mathbf{p}))/ (1+(x/T)^2)$

2. (20 points) *Error Correcting Learning (LMS algorithm)*

Let  $y(n) = w_0 + w_1x_1(n) + w_2x_2(n)$ ,  $e(n) = d(n) - y(n)$ , and  $E(n) = [e(n)]^2$ . Denote a vector  $\mathbf{w} = [w_0 \ w_1 \ w_2]$ , and  $\mathbf{x}(n) = [1 \ x_1(n) \ x_2(n)]$ .

- (a) (5 points, CC) Show that  $\nabla_{\mathbf{w}} E(n) = \begin{bmatrix} \frac{\partial E(n)}{\partial w_0} & \frac{\partial E(n)}{\partial w_1} & \frac{\partial E(n)}{\partial w_2} \end{bmatrix} = -2e(n)\mathbf{x}(n)$

(b) (10 points) The on-line error-correcting learning formula for  $\mathbf{w}$  is:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mathbf{h}' \tilde{\mathbf{N}}_w E(n) = \mathbf{w}(n) + \mathbf{h} e(n) \mathbf{x}(n) \quad (*)$$

where  $\mathbf{h} = 2\mathbf{h}'$  is a small positive *step size*. We will let  $\mathbf{h} = 0.2$  in this part of the question. Given a data set under the file name **hw1p2.txt** that can be downloaded from the homework web page. Each row of the hw1p2.txt file consists of  $[x_1(n) \ x_2(n) \ d(n)]$ . Let  $\mathbf{w}(1) = [0 \ 0 \ 0]^T$ . Use the LMS algorithm to find the values of  $w_0, w_1, \text{ and } w_2$  as they converge. Plot the convergence curve of  $w_0(n), w_1(n), w_2(n)$  and  $e(n)$  versus  $n$ .

(c) (5 points) Use Batch mode learning, repeat part (b) to find the least square solution

$$\mathbf{w}_{LS} = \mathbf{R}^{-1} \mathbf{r}$$

Give the values of the covariance matrix  $\mathbf{R}$  and cross-correlation vector  $\mathbf{r}$ .

3. (20 points) *Associative Learning, PCA*

(a) (10 points) Refer to the modified Hebbian learning equation (by Kohonen) with subscript omitted:

$$\Delta w(n) = w(n+1) - w(n) = \mathbf{h} d(n)x(n) - \mathbf{a} d(n)w(n)$$

Suppose that  $|1 - \alpha d(n)| \leq \lambda < 1$ , and  $|d(n)x(n)| < K$  for all  $n$ . Show that  $\lim_{n \rightarrow \infty} |w(n)| = B < \infty$ . Find the expression of  $B$  in terms of  $\mathbf{h}, \mathbf{I}$ , and  $K$ .

(b) (10 points, CC) Given a linear associative memory network with  $M$  input and  $N$  output. Prove that this network can store at most  $M$  different patterns such that perfect recall is possible.

4. (20 points) *Learning theory: model complexity*

Let  $f(x)$  a polynomial in  $x$  with an unknown order, and  $y(x) = f(x) + \mathbf{e}$  be noisy observations of  $f(x)$  where the additive noise  $\mathbf{e} \sim \text{normal}(0, \mathbf{s}^2)$  has a normal distribution with zero mean and variance  $\mathbf{s}^2$ . Ten pairs of training samples  $\{(x_i, y(x_i)); 1 \leq i \leq 10\}$  are drawn randomly from the interval  $[-1 \ 1]$ , and their values shown below can be download from the file **hw4p4r.txt** from the web. Each line of the file consists of one pair of training sample where  $x_i$  is listed in the first column and  $y(x_i)$  is the second column.

$x_i$	$y(x_i)$
-0.5945	0.2472
-0.058	-0.3308
-0.0318	-0.3191
0.1291	-0.5411
0.1592	-0.4312
0.3330	-0.5266
0.3535	-0.4414
0.5344	-0.2876
0.5597	-0.0432
0.6044	-0.1082

41 testing samples are taken at uniform interval between  $-1$  and  $1$  in increment of  $0.05$ . The noisy observations are taken from these sample points. Their values can be downloaded from the file **hw1p4t.txt** from the web that has the same format as **hw1p4r.txt**.

(a) (10 points) Develop a program to fit these training data to a polynomial model with orders  $p = 1, 2, 3, 4, \text{ and } 5$ . For each value of  $p$ , give the estimated polynomial  $f_p(x)$ . For example, you may write your answer as  $f_1(x) = 1 + 0.5x$ . (the parameters here are just for illustration purpose).

(b) (10 points) For each estimated polynomial model, compute the corresponding training error and testing error (sum of square error). Plot these errors as a function of model

- order  $p$ . Decide, based on these errors, what is the best estimate of the model order. Give a brief justification of your decision.
5. (20 points) *perceptron*
- Four training data samples  $\{(x_{1i}, x_{2i}, d(x_{1i}, x_{2i})); 1 \leq i \leq 4\} = \{(-1, 1, 1), (0, 0, 0), (-1, -1, 1), (1, 0, 0)\}$  are applied to a perceptron neural network that has two inputs  $x_1$ , and  $x_2$ , and three weights  $w_0$  (*bias*),  $w_1$  (*for input  $x_1$* ), and  $w_2$  (*for input  $x_2$* ).
- (c) (10 points) List the four linear inequality equations corresponding to each of the four training samples. Then, with  $w_0 = -1$ , plot the area in the  $w_1$ — $w_2$  plane corresponding to the feasible solutions to  $w_1, w_2$ .
- (d) (10 points) Develop a program (or use the one provided in the web page) to simulate this perceptron using these four data points. Submit a plot showing all data samples and the initial location of the dividing hyperplane, and another plot showing final location of the hyperplane. Also, indicate how many iterations for the perceptron algorithm to converge to the final solution. Note that even there are only four data samples, you may repeatedly applying them (recycle data) until the algorithm converges.