EEE-5813 Neural Networks SUMMER 2012

INDIVIDUAL PROJECT I **Single Layer Networks**

I. Objective:

In this project single-layer networks will discriminate 2 classes of 5x4 characters (one class at a time). One prototype of each class will be given to build a training set. Then test sets with increasing levels of "noise" will be used in testing the trained networks. Hebbian, Perceptron and Adaline learning will be used for training weights in the single-layer network.

II. Prototypes for the 5 classes, targets and conventions

The patterns used with the networks will be (5 columns x 4 rows) matrices of bipolar data (+1 o -1). The indexing in the matrices will be top to bottom for row number (starting with 1), and left to right for column number (starting with 1. Graphically, a value of +1 is an ON (dark pixel), while a -1 value is an OFF (blank) pixel. The five prototypes for the classes are shown in the last page of this assignment.

NOTE: For this project, transform the matrices to vectors by row-scanning left to right, starting at the top. In your networks, make the bias term (if you use one) the last element of the input and weight vectors.

III. Generating the TRAINING SET:

There will be 5 patterns for each class (A, E, I, O, U) in the training set: The given prototype for the class and 4 modifications that you will generate by turning ON the indicated pixel (last page of assignment). So, the total number of training patterns will be 25.

IV. Generating three different TEST SETS:

There will be 3 different test sets: I, II and III, of increasing "difficulty". Each of them will have 5 patterns for each class: the prototype and 4 modifications, generated according to the following instructions for each level of difficulty:

For I, toggle the original state of the pixel indicated in column "I" of the table (last page)
For II, toggle the states of the pixels indicated in columns "I" **and** "II"
For III, toggle the states of the pixels indicated in columns "I", "II" **and** "III"

V. SINGLE LAYER NETWORKS:

V.1 Design, train and test a single layer, single output network to recognize the **"A" class**, separating its members for the rest of the patterns.

a) Train the network with **Hebbian Learning**

Show the last set of weights

Apply the test patterns to the trained network and show the classification for all of them

b) Train the network with **Perceptron Learning**

Plot:

- The maximum weight change (absolute value) in each epoch
- The evolution of *any* 5 weights and bias, as they change *after each pattern presentation*

Repeat the training process for at least three different combinations of α value and stopping condition. In each case, clearly indicate the parameter values and stopping criterion used.
Also in each case, clearly list and/or plot the final values of the trained weights.

Use the **weights obtained from your "best" training attempt** and apply the test sets to the network. Indicate in a table which patterns are classified correctly ("hits") and which patterns are misclassified ("misses"). Then calculate the "hit ratio" (hits/ total patterns).

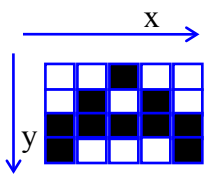
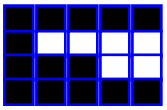
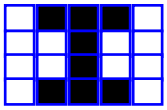
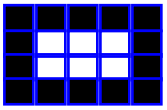
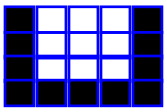
c) Train the network with **Adaline (Delta Rule) Learning**

* Report as for perceptron learning. – Only difference: "error" for Adaline during training is the difference between the net input (y_in) and the target (t)

Comment on the results obtained from each of the learning methods indicating advantages and disadvantages

V.2 Repeat V.1 for the classification of the **"E" class**.

Again comment on observed advantages / disadvantages between training methods.
Also comment on observed differences in training networks for classifying the "A" class and the "E" class.

	Turn ON for training set: (x,y)	Toggle for Test Set: I	II	III
	(2,1) (3,2) (3,4) (4,4)	(4,1) (5,3) (2,4) (1,2)	(3,2) (1,3) (3,3) (2,1)	(5,2) (3,3) (1,3) (5,1)
	(2,2) (4,2) (5,3) (4,3)	(2,4) (5,2) (4,3) (1,3)	(3,2) (3,3) (5,4) (1,2)	(3,4) (1,4) (5,2) (1,3)
	(1,1) (4,2) (1,4) (4,3)	(4,4) (2,4) (2,2) (3,4)	(1,3) (2,2) (4,3) (1,4)	(1,1) (1,2) (2,4) (2,1)
	(2,2) (4,2) (2,3) (4,3)	(1,3) (3,2) (5,4) (3,3)	(4,3) (4,3) (5,1) (2,4)	(1,2) (4,2) (3,3) (1,4)
	(2,1) (4,1) (2,3) (4,3)	(5,3) (3,4) (1,4) (5,4)	(3,3) (2,2) (2,1) (1,3)	(3,2) (5,2) (5,4) (4,2)