Computational Intelligence
Programming Assignment 1

DUE DATE: NOV 1, 2016

OBJECT OF THE ASSIGNMENT:

To understand the Widrow-Hoff learning rule (also referred as to the Least Mean Square (LMS) algorithm).

- The Perceptron learning rule and the Widrow-Hoff learning rule are both methods of training up a single layer of neurons (or even a single neuron) to learn some task.
- Unlike the perceptron learning rule, the activation function of the Widrow-Hoff learning rule is a linear function and its output values can be any real numbers.

PROBLEM:

Implement both incremental (or call stochastic) and batch versions of the Widrow-Hoff learning rule to classify the digits 0 and 1.

- Use a single neuron with 30 inputs in the recognition system.

INPUT:

REPRESENTATION OF EACH INPUT

In this simple problem, we want to design a recognition system that would classify the digits 0 and 1. Each of the digits is displayed in a 6x5 grid; see the following for an example of digit 0:

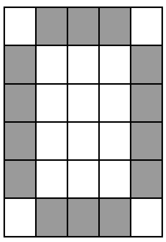


Figure 1 An example of digit 0

We need to convert the digit to a vector, which will become the input for the neuron. Each white square will be represented by a “-1”, and each dark square will be represented by a “1”. To create the input vector, we will scan each 6x5 grid one column at a time. For example, the digit 0 shown in Figure 1 will be

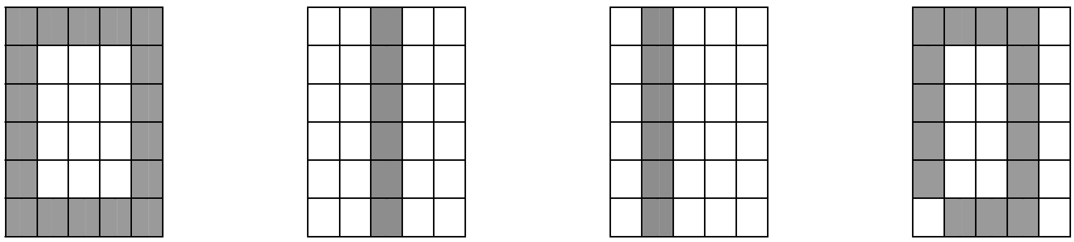
P = [-1 1 1 1 1 -1 1 -1 -1 -1 -1 1 1 -1 -1 -1 -1 1 1 -1 -1 -1 -1 1 -1 1 1 1 1 -1]^T.

For each input pattern, there is a target is associated with it. The targets for the digits 0 and 1 are -1 and 1, respectively.

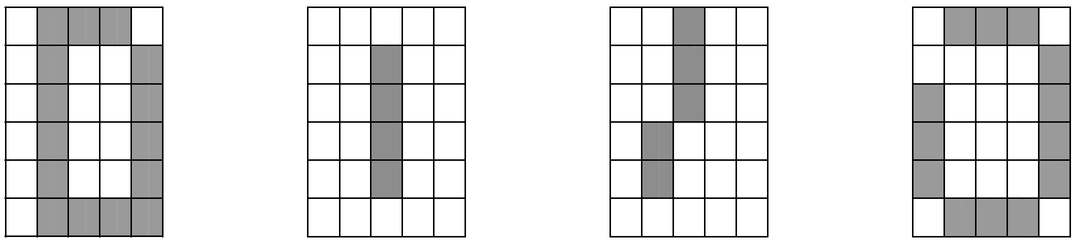
TRAINING SET

The training set that consists of two types of the input patterns is shown below:

Training pattern *i*	Neural inputs	Target t
P1	Vector P1	-1
P2	Vector P2	1
P3	Vector P3	1
P4	Vector P4	-1
P5	Vector P5	-1
P6	Vector P6	1
P7	Vector P7	1
P8	Vector P8	-1



P1 **P2** **P3** **P4**



P5 **P6** **P7** **P8**

Figure 2 Input patterns of the training set

TESTING SET

The testing set that consists of the input patterns only is shown below:

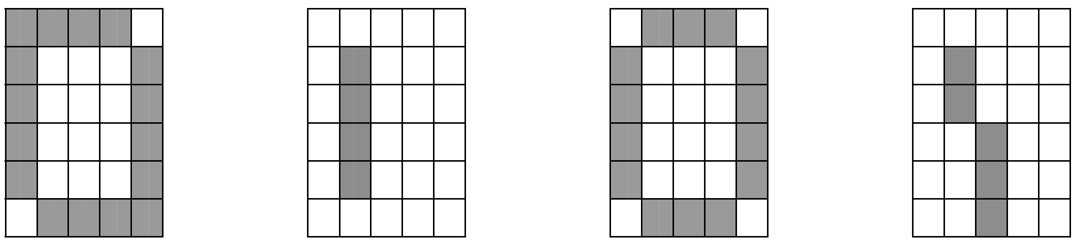


Figure 3 Input patterns of the testing set

OUTPUT:

Display the stop conditon, the adjusted bias and weight values, the training accuracy and testing accuracy.

TESTING CASES:

- Run both incremental and batch versions of the Widrow-Hoff learning rule with same parameter settings, training/testing sets, and compare their performance.
  - Try different learning rates and initial weights when you run the same training/testing sets and see any difference among the results.
- Create your own training and testing sets and repeat the process.

DISCUSSION:

Summarize your results and discuss what your observations. For example:

- Compare the performance of both versions of the Widrow-Hoff learning rule.
- Compare the performance of different learning rates and initial weights.
- Compare the performance of different training/testing sets.
- Everything else you consider it is important.

APPENDIX:

ALGORITHM: Widrow-Hoff learning rule_ Incremental version

- Choose a learning rate η and initialize weights v_i , i = 1, … , $I+1$, to random values
- UNTIL the termination condition is met DO

FOR each training pattern (z_1, z_2, ..., z_I, z_{I+1}) $\in$ dataset

Compute the current output o_p // the value of o_p can be any real number

FOR i = 1 to $I+1$ // $v_{I+1} = \theta$, $z_{I+1} \equiv -1$

$v_i \leftarrow v_i + 2 * \eta * (t_p - o_p) * z_i$

ALGORITHM: Widrow-Hoff learning rule_Batch version

- Choose a learning rate η and initialize weights v_i , i = 1, … , $I+1$, to random values
- UNTIL the termination condition is met DO

//one iteration for whole training examples

Initialize each Δv_i to zero

FOR each training pattern (z_1, z_2, ..., z_I, z_{I+1}) $\in$ dataset DO

//train each example

Compute the current output o_p // the value of o_p can be any real number

FOR i = 1 to $I+1$ // $v_{I+1} = \theta$, $z_{I+1} \equiv -1$

//train each v_i for the given example

$\Delta v_i \leftarrow \Delta v_i + 2 * \eta * (t_p - o_p) * z_i$

//have trained all examples, update v_i

FOR i = 1 to $I+1$ // $v_{I+1} = \theta$

$v_i \leftarrow v_i + \Delta v_i$

Remark:

The algorithm stops when it meets one of the termination conditions. Stopping criteria usually includes:

(a) Stop when a maximum number of epochs has been exceeded.

(b) Stop when the mean squared error (MSE) on the training set, ε, is small enough.

$$\varepsilon = \frac{\sum_{p=1}^{P_T} (t_p - o_p)^2}{P_T},$$

where t_p and o_p are respectively the target and actual output for the p-th pattern, and P_T is the total number of input-target vector pairs (patterns) in the training set.

For example, $\varepsilon < \tau$ (= 10^{-6}), where τ is a given tolerance.

(c) Δv_i < a given weight change tolerance ε_i , i = 1, 2, …,$I+1$.