Kingdom of Saudi Arabia Ministry of Education University of Jeddah College of Computer Science and Engineering Department of Computer Science and Artificial Intelligence	جامعة جدة University of Jeddah	المملكة العربية السعودية وزارة التعليم جامعة جدّة كلية علوم وهندسة الحاسب قسم علوم الحاسب والذكاء الاصطناعي

Assignement 1

CCAI 321 Artificial Neural Networks

First semester 2023/2024

Exam Date: wednesday 09/20/2023 **Exam Duration: 1 week**

Student Name: Rimas Alshehri **Student ID:** 2110240

Instructor Name	Section
Samia Snoussi	A1

Instructions:

Please Turn Off your mobile phones

Cheating or discussion with colleagues will result in negative marking

Including this cover page, this exam booklet contains 8 pages. Check if you have missing pages

PLO/CLO	SO
PLO K1 (CLO 1): Demonstrate basic knowledge of mathematics and sciences to identify, analyze and solve problems related to relevant disciplines in neural network	**SO 1 :** Analyze a complex computing problem and to apply principles of computing and other relevant disciplines to identify solutions
PLO K2 (CLO 2): Evaluate and **Classify** neural networks algorithms, processes and systems using modeling techniques	
PLO S1 (CLO 2): Design, **implement** and **evaluate** practical solutions to domain-specific problems against set of requirements in the context of neural network discipline	**SO 2 :** Design, implement, and evaluate a computing based solution to meet a given set of computing requirements in the context of the program's discipline

		Max Score	Student Score
PLO K1 / CLO 1 / SO 1	**Question 1**	**0.4**	
PLO K1 / CLO 1 / SO 1	**Question 2**	**0.4**	
PLO K1 / CLO 1 / SO 1	**Question 3**	**0.8**	

PLO K1 / CLO 1 / SO 1	**Question 4**	**0.8**	
PLO K1 / CLO 1 / SO 1	**Question 5**	**0.2**	
PLO K1 / CLO 1 / SO 1	**Question 6**	**0.2**	
PLO K1 / CLO 1 / SO 1	**Question 7**	**0.8**	
PLO K1 / CLO 1 / SO 1	**Question 8**	**0.2**	
PLO K1 / CLO 1 / SO 1	**Question 9**	**0.5**	
PLO K1 / CLO 1 / SO 1	**Question 10**	**0.5**	
PLO K1 / CLO 1 / SO 1	**Question 11**	**0.2**	
Total		**5**	

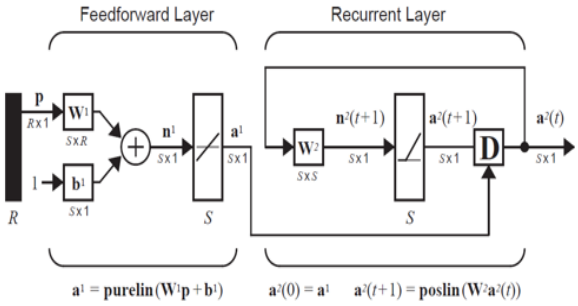
Problem 1: Statement (Understanding inputs)

- The input vector to neural network will have shape $\mathbf{p} = \begin{bmatrix} shape \\ texture \\ weight \end{bmatrix}$
- An orange is round, rough and less than one pound so $\mathbf{po} = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$
- An apple is round, smooth and less than one pound so $\mathbf{pa} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$

The neural network should expect input p_o or p_a as input

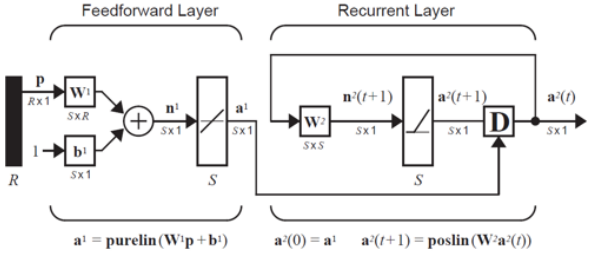
$p_a^T = [1 \;\; 1 \;\; -1]$ $\quad$ $p_o^T = [1 \;\; -1 \;\; -1]$

Hamming Network



- Feedforward Layer
  - The layer calculates
    - Inner product between input and target
    - Only possible if W contains prototype pattern
  - For apple and orange problem prototype or target patterns are
    - $\mathbf{W^1} = \begin{bmatrix} \mathbf{p}_a^T \\ \mathbf{p}_o^T \end{bmatrix}$ =? $= \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$
  - The feedforward network comprises of a linear transfer function
  - Set value of bias vector equal to number of input elements R i.e. $\mathbf{b^1}$ =? $\begin{bmatrix} 3 \\ 3 \end{bmatrix}$
  - The output $\mathbf{a^1} = \mathbf{W^1 p} + \mathbf{b^1}$ =? $\begin{bmatrix} p_a^T \\ p_o^T \end{bmatrix} p + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} p_a^T p + 3 \\ p_o^T p + 3 \end{bmatrix}$
  - Adding R to inner product guarantees that output $\mathbf{a^1}$ is never negative

Hamming Network



- Recurrent Layer
  - Called the competitive layer
    - It decides which correlation is higher
    - Only one neurons output will be nonzero
    - Initialized with outputs of feedfowrd layer ie correlation
  - The weight matrix is defined as
    - $\mathbf{W^2} = \begin{bmatrix} \mathbf{1} & -\varepsilon \\ -\varepsilon & \mathbf{1} \end{bmatrix}$ where $\varepsilon < 1$ / (S-1) where S is number of neurons
  - The output $\mathbf{a^2}(t+1) = poslin(\mathbf{W^2 a^2}(\boldsymbol{t}))$
  - $\mathbf{a^2}(t+1) = poslin\left(\begin{bmatrix} 1 & -\epsilon \\ -\epsilon & 1 \end{bmatrix} a^2(t)\right) = poslin\left(\begin{bmatrix} 1 & -\epsilon \\ -\epsilon & 1 \end{bmatrix} \begin{bmatrix} a_1^2(t) \\ a_2^2(t) \end{bmatrix}\right)$
  - $\mathbf{a^2}(t+1) = poslin\left(\begin{bmatrix} a_1^2(t) - \epsilon\, a_2^2(t) & \square \\ a_2^2(t) - \epsilon\, a_1^2(t) & \square \end{bmatrix}\right)$
  - Each element is reduced by same fraction of other, smaller element will become zero

Let's assume that the inputs are :

$$\mathbf{p1} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \text{ and } \mathbf{p2} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

1) Initialise W^1 with the appropriate values as defined above
2) Intialise W^2 to the appropriate value with Ɛ=0.2
3) Calculate for each input the
   a. Output of Feedforward network
   b. The outputs of the recurrent layer until convergence
   c. Classify P1 and P2 input to their corresponding classes apple or orange
4) Reinitialise W^2 to the appropriate value with Ɛ=0.8 and recalculate:
   a. Output of Feedforward network
   b. The outputs of the recurrent layer until convergence
   c. Classify P1 and P2 input to their corresponding classes apple or orange
5) What do you conclude about the suitable value of Ɛ?
6) If you reinitialise W^1 with $\mathbf{W^1} = \begin{bmatrix} \mathbf{p}_o^T \\ \mathbf{p}_a^T \end{bmatrix}$ what will you have as results ?

① $W_1 = \begin{bmatrix} p_a^T \\ p_o^T \end{bmatrix} = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}$

② $W_2 = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}$

③ a ⇒ for $P_1 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

$a = \text{purelin}(wp+b) = a = \text{purelin}(W p + b)$

$= \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} -1-1+1+3 \\ -1+1+1+3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \text{purelin}\left(\begin{bmatrix} 2 \\ 4 \end{bmatrix}\right) = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$

$2\times3 \quad 3\times1 \quad 2\times1$

b ⇒ for $P_1 \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

$a^2(0) = a^1 = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$

$a^2(1) = \text{poslin}(W^2 a^2(0)) = \text{poslin}\left(\begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \end{bmatrix}\right) = \text{poslin}\left(\begin{bmatrix} 1.2 \\ 3.6 \end{bmatrix}\right) = \begin{bmatrix} 1.2 \\ 3.6 \end{bmatrix}$

$2\times2 \quad 2\times1 \quad 2\times1$

$$a^2(2) = \text{poslin}\left(W^2 a^2(1)\right) = \text{poslin}\left(\begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 1.2 \\ 3.6 \end{bmatrix}\right) = \text{poslin}\left(\begin{bmatrix} 0.5 \\ 3.4 \end{bmatrix}\right) = \begin{bmatrix} 0.5 \\ 3.4 \end{bmatrix}$$

$1 \times 1.2 + -0.2 \times 3.6 = 1.2 - 0.72 = 0.48 \approx 0.5$
$-0.2 \times 1.2 + 1 \times 3.6 = -0.24 + 3.6 = 3.4$

$$a^2(3) = \text{poslin}\left(W^2 a^2(2)\right) = \text{poslin}\left(\begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 0.5 \\ 3.4 \end{bmatrix}\right) = \text{poslin}\left(\begin{bmatrix} -0.2 \\ 3.3 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 3.3 \end{bmatrix}$$

$0.5 - 0.68 = -0.18 \approx -0.2$
$-0.1 + 3.4 = 3.3$

$$a^2(4) = \text{poslin}\left(W^2 a^2(3)\right) = \text{poslin}\left(\begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ 3.3 \end{bmatrix}\right) = \text{poslin}\left(\begin{bmatrix} -0.7 \\ 3.3 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 3.3 \end{bmatrix}$$

$-0.66 \approx -0.7$
3.3

Output doesn't change so the network has converged

$$a^2(3) = \begin{bmatrix} 0 \\ 3.3 \end{bmatrix} = a^2(4) = \begin{bmatrix} 0 \\ 3.3 \end{bmatrix} \begin{matrix} \rightarrow \text{apple} \\ \rightarrow \text{orange} \end{matrix}$$

c Since the second value is nonzero the pattern $[1\ -1\ -1]$ is **orange** for input $p_1\ [-1\ -1\ -1]$

3 for input $p_2 \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

a $a = \text{purelin}(Wp + b)$

$$= \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \text{purelin}\left(\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

b $a^2(0) = a^1 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

$$a^2(1) = \text{poslin}\left(W^2 a^2(0)\right) = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 3.6 \\ 1.2 \end{bmatrix} \quad \text{poslin}\left(\begin{bmatrix} 3.6 \\ 1.2 \end{bmatrix}\right) = \begin{bmatrix} 3.6 \\ 1.2 \end{bmatrix}$$

$4 + (-0.2)(2) = 3.6$
$-0.2 \times 4 + 2 = 1.2$

$$a^2(2) = \text{poslin}\left(W^2 a^2(1)\right) = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix}\begin{bmatrix} 3.6 \\ 1.2 \end{bmatrix} = \begin{bmatrix} 3.4 \\ 0.5 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} 3.4 \\ 0.5 \end{bmatrix}\right) = \begin{bmatrix} 3.4 \\ 0.5 \end{bmatrix}$$

$3.6 - 0.24 = 3.36 \approx 3.4$

$$a^2(3) = \left(W^2 a^2(2)\right) = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix} \begin{bmatrix} 3.4 \\ 0.5 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} 3.3 \\ -0.2 \end{bmatrix}\right) = \begin{bmatrix} 3.3 \\ 0 \end{bmatrix}$$

$3.4 + 0.5(-0.2) = 3.3$
$-0.2 \times 3.4 + 0.5 = -0.18 \approx -0.2$

$$a^2(4) = \left(W^2 a^2(3)\right) = \begin{bmatrix} 1 & -0.2 \\ -0.2 & 1 \end{bmatrix} \begin{bmatrix} 3.3 \\ 0 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} 3.3 \\ -0.7 \end{bmatrix}\right) = \begin{bmatrix} 3.3 \\ 0 \end{bmatrix}$$

$3.3 \times 1 + 0 = 3.3$
$-0.2 \times 3.3 + 1 \times 0 = -0.66 \approx -0.7$

Output doesn't change network has converged $a^2(3) = a^2(4) = \begin{bmatrix} 3.3 \\ 0 \end{bmatrix}$

(c) The first value is nonzero so pattern $[1\ 1\ -1]$ is Apple for input $\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$

---

for $p_1 = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

4

(a) $a = \text{purelin}(Wp + b)$

$$a = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \text{purelin}\left(\begin{bmatrix} 2 \\ 4 \end{bmatrix}\right) = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$$

$-1 + 1 + 1 + 3 = 2$
$1 + 1 + 1 + 3 = 4$

(b) $a^2(0) = a^1 = \begin{bmatrix} 2 \\ 4 \end{bmatrix}$

$$a^2(1) = \text{poslin}\left(W, a^2(0)\right) = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 4 \end{bmatrix} = \begin{bmatrix} -1.2 \\ 2.4 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} -1.2 \\ 2.4 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 2.4 \end{bmatrix}$$

$1 \times 2 + 4(-0.8) = -1.2$
$2 \times -0.8 + 4 = 2.4$

$$a^2(2) = \text{poslin}\left(W, a^2(1)\right) = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 2.4 \end{bmatrix} = \begin{bmatrix} -1.9 \\ 2.4 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} -1.9 \\ 2.4 \end{bmatrix}\right) = \begin{bmatrix} 0 \\ 2.4 \end{bmatrix}$$

$-0.8 \times 2.4 = -1.92$
2.4

Output converged because $a^2(1) = a^2(2) = \begin{bmatrix} 0 \\ 2.4 \end{bmatrix}$

(c) Output of pattern $[1\ -1\ -1]$ is Orange for input $p_1 \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$

4 for input $p_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

a

$$a = \text{purelin}(Wp + b)$$

$$a = \begin{bmatrix} 1 & 1 & -1 \\ 1 & -1 & -1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

b $a^2(0) = a^1 = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$

$$a^2(1) = \text{poslin}(W^2 a^2(0)) = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 2 \end{bmatrix} = \begin{bmatrix} 2.4 \\ -1.2 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} 2.4 \\ -1.2 \end{bmatrix}\right) = \begin{bmatrix} 2.4 \\ 0 \end{bmatrix}$$

$1 \times 4 + 2(-0.8) = 2.4$

$-0.8(4) + 1 \times 2 = -1.2$

$$a^2(2) = \text{poslin}(W^2 a^2(1)) = \begin{bmatrix} 1 & -0.8 \\ -0.8 & 1 \end{bmatrix} \begin{bmatrix} 2.4 \\ 0 \end{bmatrix} = \begin{bmatrix} 2.4 \\ -1.9 \end{bmatrix} = \text{poslin}\left(\begin{bmatrix} 2.4 \\ -1.9 \end{bmatrix}\right) = \begin{bmatrix} 2.4 \\ 0 \end{bmatrix}$$

$1 \times 2.4 + 0 = 2.4$

$-0.8 \times 2.4 + 0 = -1.92$

Output has converged $a^2(1) = a^2(2) = \begin{bmatrix} 2.4 \\ 0 \end{bmatrix}$

c first value is nonzero so pattern $[1\ 1\ -1]$ is class Apple for input $p_2 \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

---

5 both values of $\varepsilon = 0.2$ and $\varepsilon = 0.8$ gave me the same result for the classes respectively, but $\varepsilon = 0.8$ converged faster than $\varepsilon = 0.2$

(6)

reinitializing $W_1 = \begin{bmatrix} p_o^T \\ p_a^T \end{bmatrix} = \begin{bmatrix} 1 & -1 & -1 \\ 1 & 1 & -1 \end{bmatrix}$

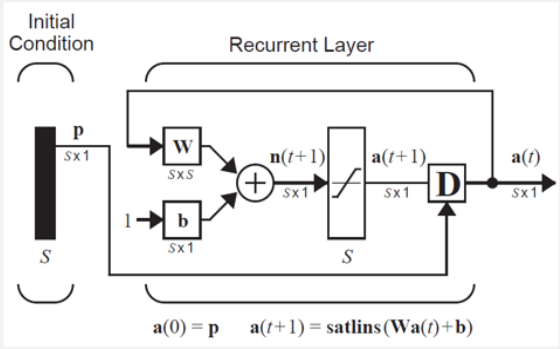
switching places in the matrix will switch the classes so that it will be

for $P_1 \Rightarrow a^2 = \begin{bmatrix} 2.4 \\ 0 \end{bmatrix}$ → Orange $\quad P_2 \Rightarrow a^2 = \begin{bmatrix} 0 \\ 2.4 \end{bmatrix}$ → Apple

Problem 2:

Hopfield Network

- It is a recurrent Neural Network
- It is initialized with input vector
- Network iterates till convergence
- Resulting output is one of the target / prototype vectors
- Output at time t = 0
  - a(0) = p
- Output at time t+1
  - a(t+1) = satlins(Wa(t)+b)
- Transfer function satlins(n) is linear in range [-1 1] and then saturates i.e. becomes 1 or -1 constantly
- Calculating W and b for Hopfield network are not discussed here because of complexity



- Assume that

- $\mathbf{W} = \begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 1.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} 0.9 \\ 0 \\ -0.9 \end{bmatrix}$

Let's assume that $\mathbf{p} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$ $\Longrightarrow$ then $a(0) = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

7) Calculate a(t) until convergence
8) Deduce the class of **p**

(7) $a(1) = satlin\left(\begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 1.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}\right)\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} + \begin{bmatrix} 0.9 \\ 0 \\ -0.9 \end{bmatrix}$

$= satlin\left(\begin{bmatrix} 0.2+0.9 \\ 1.2 \\ 0.2-0.9 \end{bmatrix}\right) = satlin\left(\begin{bmatrix} 1.1 \\ 1.2 \\ -0.7 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$ → Convergence

$a(2) = satlin\left(\begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 1.2 & 0 \\ 0 & 0 & 0.2 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + \begin{bmatrix} 0.9 \\ 0 \\ -0.9 \end{bmatrix}\right) = \begin{bmatrix} 0.2+0.9 \\ 1.2 \\ -0.2-0.9 \end{bmatrix}$

$= satlin\left(\begin{bmatrix} 1.1 \\ 1.2 \\ -1.1 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$ → Convergence

No change between a(1) and a(2) and a(2) indicates their the fruit is apple.

(8) The class of p is Apple

Problem 3

- The input vector to neural network will have shape $\mathbf{p} = \begin{bmatrix} shape \\ texture \\ weight \end{bmatrix}$
- An orange is round, rough and less than one pound so $\mathbf{po} = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$ → target = 0
- An apple is round, smooth and less than one pound so $\mathbf{pa} = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$ → target = 1
- The neural network should expect input p_o or p_a as input

Example

Randomly initialize W and b

- W=[0.5 -1 -0.5], b =0.5

- Calculate output for first input p_1
  - a = hardlim(Wp_1+b)=

hardlim([0.5 -1 -0.5] $\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$ +0.5) = hardlim(0.5+1+0.5+0.5)=hardlim(2.5)=1

  - Calculate error e = t – a = 0 – 1 =-1
- Update weights and bias
  - $W^{new} = W^{old} + ep^T$ = [0.5 -1 -0.5] (-1)[1 -1 -1] =[-0.5 0 0.5]
  - $b^{new} = b^{old} + e$ = 0.5 -1 =-0.5

  → [0.5 -1 -0.5] + [-1 1 1]
  ⇒ [-0.5 0 0.5]
- This completes first iteration
- Calculate output for second input p_2
  - a = hardlim(Wp_2+b)=

hardlim([-0.5 0 0.5] $\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$ -0.5) = hardlim(-0.5+0-0.5-0.5)=hardlim(-1.5)=0

- Calculate error e = t – a = 1 – 0 =1

- Update weights and bias
  - $W^{new} = W^{old} + ep^T$ = [-0.5 0 0.5] +(1)[1 1 -1] =[0.5 1 -0.5]
  - $b^{new} = b^{old} + e$ = -0.5 +1 =0.5
- This completes second iteration
- Calculate output for first input p_1
  - a = hardlim(Wp_1+b)=

hardlim([0.5 1 -0.5] $\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$ +0.5) = hardlim(0.5-1+0.5+0.5)=hardlim(0.5)=1

  - Calculate error e = t – a = 0 – 1 =-1
- Update weights and bias
  - $W^{new} = W^{old} + ep^T$ = [0.5 1 -0.5] -(1)[1 -1 -1] =[-0.5 2 0.5]
  - $b^{new} = b^{old} + e$ = 0.5 +(-1) =-0.5
- This completes third iteration
- Calculate output for second input p_2
  - a = hardlim(Wp_2+b)=

hardlim([-0.5 2 0.5] $\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$ -0.5) = hardlim(-0.5+2-0.5-0.5)=hardlim(0.5)=1

  - Calculate error e = t – a = 1 – 1 =0
- Calculate output for first input p_1
  - a = hardlim(Wp_1+b)=

hardlim([-0.5 2 0.5] $\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$ -0.5) = hardlim(-0.5-2-0.5-0.5)=hardlim(-3.5)=0

  - Calculate error e = t – a = 0 – 0 =0

- The algorithm has converged to a solution

9) Repeat the same calculation as in the example above by changing

Randomly initialize W and b to W=[0.2 -1 -0.2], b =0.2

10) Repeat the same calculation as in the example above by changing

Randomly initialize W and b to W=[0.8 -1 -0.8], b =0.8

11) What do you conclude?

begin with p_1

9) $a = hardlim\left(\begin{bmatrix}0.2 & -1 & -0.2\end{bmatrix}\begin{bmatrix}1\\-1\\-1\end{bmatrix} + 0.2\right)$

$(1\times 3)\quad (3\times 1)$

$= hardlim\left(\begin{bmatrix}0.2+1+0.2\end{bmatrix} + 0.2\right) = hardlim(1.6) = 1$

$e = t - a = 0 - 1 = -1$ → Update W and b

$W_{new} = W_{old} + ep^T = \begin{bmatrix}0.2 & -1 & -0.2\end{bmatrix} + (-1)\begin{bmatrix}1 & -1 & -1\end{bmatrix}$

$= \begin{bmatrix}0.2 & -1 & -0.2\end{bmatrix} + \begin{bmatrix}-1 & 1 & 1\end{bmatrix} = \begin{bmatrix}-0.8 & 0 & 0.8\end{bmatrix}$

$b_{new} = b_{old} + e = 0.2 + (-1) = -0.8$

1st iteration complete

---

p_2

$a = hardlim\left(\begin{bmatrix}-0.8 & 0 & 0.8\end{bmatrix}\begin{bmatrix}1\\1\\-1\end{bmatrix} - 0.8\right)$

$a = hardlim\left(\begin{bmatrix}-0.8 & 0 & -0.8\end{bmatrix} - 0.8\right) = hardlim(-2.4) = 0$

$e = t - a = 1 - 0 = 1$ → update W and b

$$W_{new} = W_{old} + eP^T = [-0.8 \quad 0 \quad 0.8] + (1)[1 \quad 1 \quad -1]$$

$$= [-0.8 \quad 0 \quad 0.8] + [1 \quad 1 \quad -1] = [0.2 \quad 1 \quad -0.2]$$

$$b_{new} = b_{old} + e = -0.8 + 1 = 0.2$$

2nd iteration complete

$P_1 = \begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}$

$$a = hardlim\left([0.2 \quad 1 \quad -0.2]\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix} + 0.2\right)$$

$$= hardlim([0.2 \quad -1 \quad 0.2] + 0.2) = hardlim(-0.4) = 0$$

$$e = t - a = 0 - 0 = 0$$

$P_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$

$$a = hardlim\left([0.2 \quad 1 \quad -0.2]\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + 0.2\right)$$

$$= hardlim([0.2 \quad 1 \quad 0.2] + 0.2) = hardlim(1.6) = 1$$

$e = t - a = 1 - 1 = 0$ → now that the error is zero then no need to update W and b anymore, because the algorithm has converged.

Algorithm converged

(10)

$$a = hardlim\left(\begin{bmatrix} 0.8 & -1 & -0.8 \end{bmatrix}\underbrace{\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}}_{P_1} + \begin{bmatrix} 0.8 \end{bmatrix}\right) = hardlim\left(\begin{bmatrix} 0.8 & 1 & 0.8 \end{bmatrix} + 0.8\right)$$

$$= hardlim(3.4) = 1$$

$e = t - a = 0 - 1 = -1 \longrightarrow$ update W and b

$W_{new} = W_{old} + ep^T = [0.8 \quad -1 \quad -0.8] + (-1)[1 \quad -1 \quad -1] =$

$= [0.8 \quad -1 \quad -0.8] + [-1 \quad 1 \quad 1] = [-0.2 \quad 0 \quad 0.2]$

$b_{new} = b_{old} + e = 0.8 + (-1) = -0.2$

1st iteration complete

$$a = hardlim\left(\begin{bmatrix} -0.2 & 0 & 0.2 \end{bmatrix}\underbrace{\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}}_{P_2} + \begin{bmatrix} -0.2 \end{bmatrix}\right)$$

$= hardlim([-0.2 \quad 0 \quad -0.2] + [-0.2])$

$= hardlim(-0.6) = 0$

$e = t - a = 1 - 0 = 1 \longrightarrow$ update W, b

$W_{new} = W_{old} + ep^T = [-0.2 \quad 0 \quad 0.2] + (1)[1 \quad 1 \quad -1]$

$= [-0.2 \quad 0 \quad 0.2] + [1 \quad 1 \quad -1] = [0.8 \quad 1 \quad -0.8]$

$b_{new} = b_{old} + e = [-0.2] + 1 = 0.8$

2nd iteration complete

$$a = hardlim\left(\begin{bmatrix} 0.8 & 1 & -0.8 \end{bmatrix}\underbrace{\begin{bmatrix} 1 \\ -1 \\ -1 \end{bmatrix}}_{P_1} + 0.8\right)$$

$= hardlim([0.8 \quad -1 \quad 0.8] + 0.8) = hardlim(1.4) = 1$

$e = t - a = 0 - 1 = -1 \longrightarrow$ update W, b

$W_{new} = W_{old} + ep^T = [0.8 \quad 1 \quad -0.8] + (-1)[1 \quad -1 \quad -1]$

$= [0.8 \quad 1 \quad -0.8] + [-1 \quad 1 \quad 1] = [-0.2 \quad 2 \quad 0.2]$

$b_{new} = b_{old} + e = 0.8 + (-1) = -0.2$

3rd iteration complete

$a = hardlim\left(\begin{bmatrix}-0.2 & 2 & 0.2\end{bmatrix}\begin{bmatrix}1\\1\\-1\end{bmatrix} + (-0.2)\right)$ (P_2)

$= hardlim\left(\begin{bmatrix}-0.2 & 2 & -0.2\end{bmatrix} + (-0.2)\right)$

$= hardlim(1.4) = 1$

$e = t - a = 1 - 1 = 0$

$a = hardlim\left(\begin{bmatrix}-0.2 & 2 & 0.2\end{bmatrix}\begin{bmatrix}1\\-1\\1\end{bmatrix} + (0.2)\right)$ (P_1)

$= hardlim\left(\begin{bmatrix}-0.2 & -2 & -0.2\end{bmatrix} + (-0.2)\right] = hardlim(-2.6) = 0$

$e = t - a = 0 - 0 = 0$ → now that the error is zero then no need to update W and b anymore, because the algorithm has converged.

Algorithm Converged

11) for W=[0.2 -1 -0.2], b =0.2 it required 2 iterations to reach the convergence.

While for W=[0.8 -1 -0.8], b =0.8 it required 3 iterations to reach convergence.

so in conclusion W=[0.2 -1 -0.2], b =0.2 was faster.

The values of weight and bias change with different initializations