NEURAL NETWORK LEARNING RULES

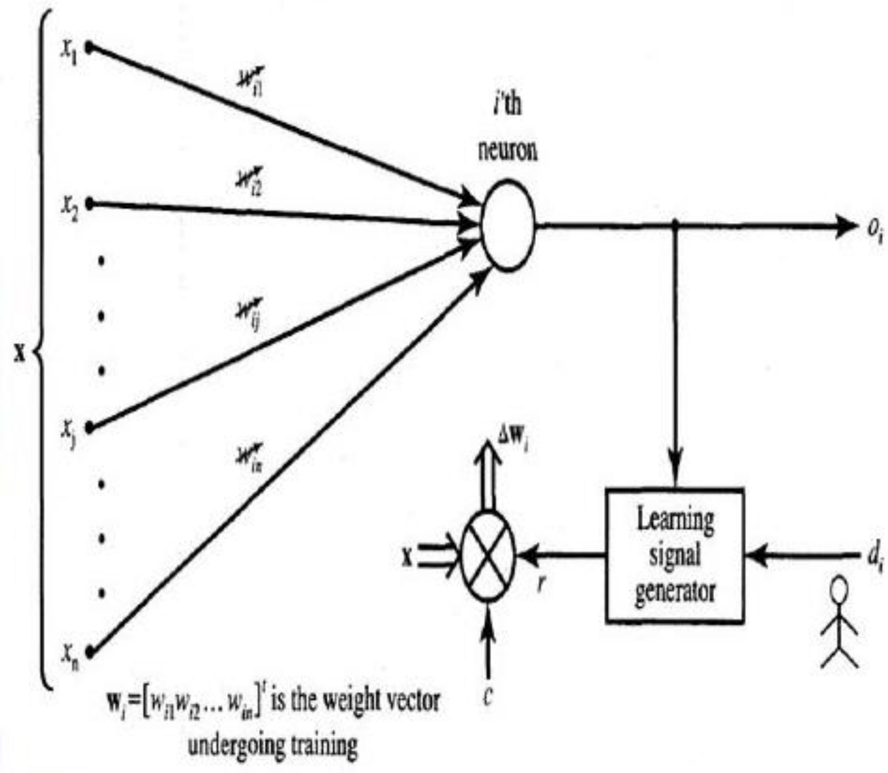
CHAPTER 2


ARTIFICIAL NEURAL NETWORK LEARNING
RULES

Neural Network Learning Rules

We know that, during ANN learning, to change the input/output behavior, we need to adjust the weights. Hence, a method is required with the help of which the weights can be modified. These methods are called Learning rules, which are simply algorithms or equations.

Neural Network Learning Rules



- The learning signal **r** in general a function of **wi**, **x** and sometimes of teacher's signal di.

$$r = r(\mathbf{w}_i, \mathbf{x}, d_i)$$

- Incremental weight vector wi at step t becomes:

$$\Delta \mathbf{w}_i(t) = cr\left[\mathbf{w}_i(t), \mathbf{x}(t), d_i(t)\right]\mathbf{x}(t)$$

Where c is a learning constant having +ve value.

Neural Network Learning Rules

- Perceptron Learning Rule -- Supervised Learning
- Hebbian Learning Rule – Unsupervised Learning
- Delta Learning Rule -- Supervised Learning
- Widrow-Hoffs Learning Rule -- Supervised Learning
- Correlation Learning Rule -- Supervised Learning
- Winner-Take-all Learning Rule -- Unsupervised Learning
- Outstar Learning Rule -- Supervised Learning

Hebbian Learning Rule

For the Hebbian learning rule the learning signal is equal simply to the neuron's output.

$$\boxed{r = f(w_i^t x)} \quad \text{—— (1)}$$

The increment vector Δw_i becomes

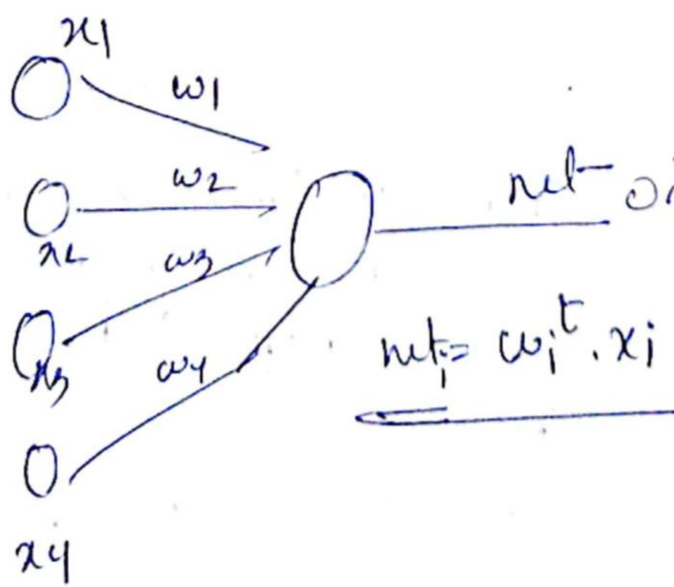
$$\boxed{\Delta w_i = c \cdot f(w_i^t \cdot x) \cdot x} \quad \text{—— (2)}$$

Single weight w_{ij} is adapted using,

$$\boxed{\Delta w_{ij} = c f(w_i^t \cdot x) x_j} \quad j = 1, 2, \ldots n \quad \text{—— (3)}$$

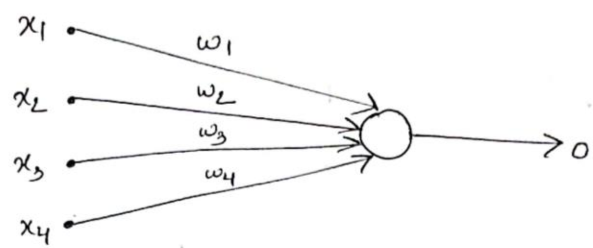
⇒ This learning rule requires weight initialization at small random values around $w_i = 0$ prior to learning.

⇒ Purely feedforward, unsupervised learning.



$$net_i = w_i^t \cdot x_i$$

E.g.: For the given simple network apply Hebbian learning with Bipolar binary and continuous activation functions.



Input

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \qquad w^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

Needs to be trained using the set of three input vectors as below:-

$$x_1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} \qquad x_2 = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} \qquad x_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

for an arbitrary constants $c=1$.

Solⁿ: Since, the initial weights are nonzero value, the network has apparently been trained beforehand. Assume first that bipolar binary neurons are used, and thus

$$f(net) = sgn(net).$$

Step 1: Input x_1 applied to the network results in activation net^1 as below:-

$$net^1 = w^{1t} x_1 = [1 \;\; -1 \;\; 0 \;\; 0.5] \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}$$

$$= 1 + 2 + 0 + 0 = 3$$

The updated weights are, $\underline{\Delta w_1}$

$$w^2 = w^1 + c.\,\text{sgn}(net^1)\, x_1 \qquad \boxed{\begin{array}{l} \text{sgn}(net^1) = 1 \\ \text{because} \\ net^1 > 0 \end{array}}$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + 1 \cdot 1 \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix} = \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix}$$

- **Step 2:** This learning step is with x_2 as input,

$$net^2 = w^{2t} x_2 = [2 \;\; -3 \;\; 1.5 \;\; 0.5] \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}$$

$$= 2 + 1.5 - 3.0 - 0.75 = -1.0 + 0.75$$

$$= -0.25$$

The updated weights are,

$$w^3 = w^2 + \text{sgn}(net^2)\, x_2 \qquad \boxed{\begin{array}{l} \text{sgn}(net^2) = -1 \\ \text{because} \\ net^2 < 0 \end{array}}$$

$$= w^2 + (-1)\, x_2$$

$$= \begin{bmatrix} 2 \\ -3 \\ 1.5 \\ 0.5 \end{bmatrix} - \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix} = \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix}$$

Step 3: This learning step is with x_3 as input,

$$net^3 = w^{3t} x_3 = [1 \;\; -2.5 \;\; 3.5 \;\; 2] \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$$= 0 - 2.5 - 3.5 + 3.0 = -3.0$$

The updated weights are,

$$w^4 = w_3 + \mathrm{sgn}(net^3)\, x_3$$

$$= \begin{bmatrix} 1 \\ -2.5 \\ 3.5 \\ 2 \end{bmatrix} + (-1)\begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix} = \begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$$

Revisiting, the same problem with continuous bipolar activation function $f(net)$, using input x_1 and initial weight w_1, we obtain neuron output values and updated weights for $\lambda = 1$. Here $f(net)$ is computed as,

$$\boxed{f(net) = \frac{2}{1 + e^{-\lambda net}} - 1}$$

Step 1:-

$$net = 3$$

$$f(net^1) = 0.905$$

$$w^2 = w^1 + c\, f(net^1)\, x_1$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + 0.905\begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix} + \begin{bmatrix} 0.905 \\ -1.81 \\ 1.36 \\ 0 \end{bmatrix} = \begin{bmatrix} 1.905 \\ -2.81 \\ 1.36 \\ 0.5 \end{bmatrix}$$

$1+e$

$$net^2 = w^{2t} x_2 = [1.905 \;\; -2.81 \;\; 1.36 \;\; 0.5] \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}$$

Step 2:

$$= -0.16$$

$$f(net^2) = \frac{2}{1+e^{-\lambda(-0.16)}} - 1$$

$$= \frac{2}{1+1.17} - 1 = -0.077$$

$$w^3 = w_2 + f(net^2)\, x_2$$

$$= \begin{bmatrix} 1.905 \\ -2.81 \\ 1.36 \\ 0.5 \end{bmatrix} + -0.077 \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}$$

$$= \begin{bmatrix} 1.905 \\ -2.81 \\ 1.36 \\ 0.5 \end{bmatrix} + \begin{bmatrix} -0.077 \\ 0.038 \\ 0.154 \\ 0.115 \end{bmatrix}$$

$$w_3 = \cancel{\begin{bmatrix} 1.828 \\ -0.944 \\ 1.514 \\ 0.616 \end{bmatrix}} \quad \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$$

~~ep~~ 3: $f(net^3) = w_3^{3t} x_3$

$$= [1.828 \quad -2.772 \quad 1.512 \quad 0.616] \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$$= 0 - 2.772 - 1.512 + 0.924$$

$$= -3.36$$

$$f(net^3) = \frac{2}{1 + e^{-\lambda net_3}} - 1$$

$$= \frac{2}{1 + e^{-(1)(-3.36)}} - 1$$

$$= \frac{2}{1 + 28.78} - 1 = -0.932$$

$$w_4 = w_3 + f(net^3) x_3$$

$$= \begin{bmatrix} 1.905 \\ -2.81 \\ 1.36 \\ 0.5 \end{bmatrix} + (-0.932) \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$$= \cancel{\begin{bmatrix} 1.905 \\ -2.81 \\ 1.36 \\ 0.5 \end{bmatrix}} + \begin{bmatrix} 0 \\ -0.932 \\ 0.932 \\ -1.398 \end{bmatrix} + \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$$

$$= \cancel{\begin{bmatrix} 1.905 \\ -3.742 \end{bmatrix}} = \begin{bmatrix} 1.828 \\ \cancel{-1.412} \; -3.704 \\ \cancel{2.444} \; 2.44 \\ -0.782 \end{bmatrix}$$