Artificial Neural Networks : An Introduction

Dr.P.Srinivasan
Professor / CSE
MEC (Autonomous)

Learning Objectives

- Fundamentals of ANN
- Comparison between biological neuron and artificial neuron
- Basic models of ANN
- Different types of connections of NN, Learning and activation function
- Basic fundamental neuron model-McCulloch-Pitts neuron and Hebb network

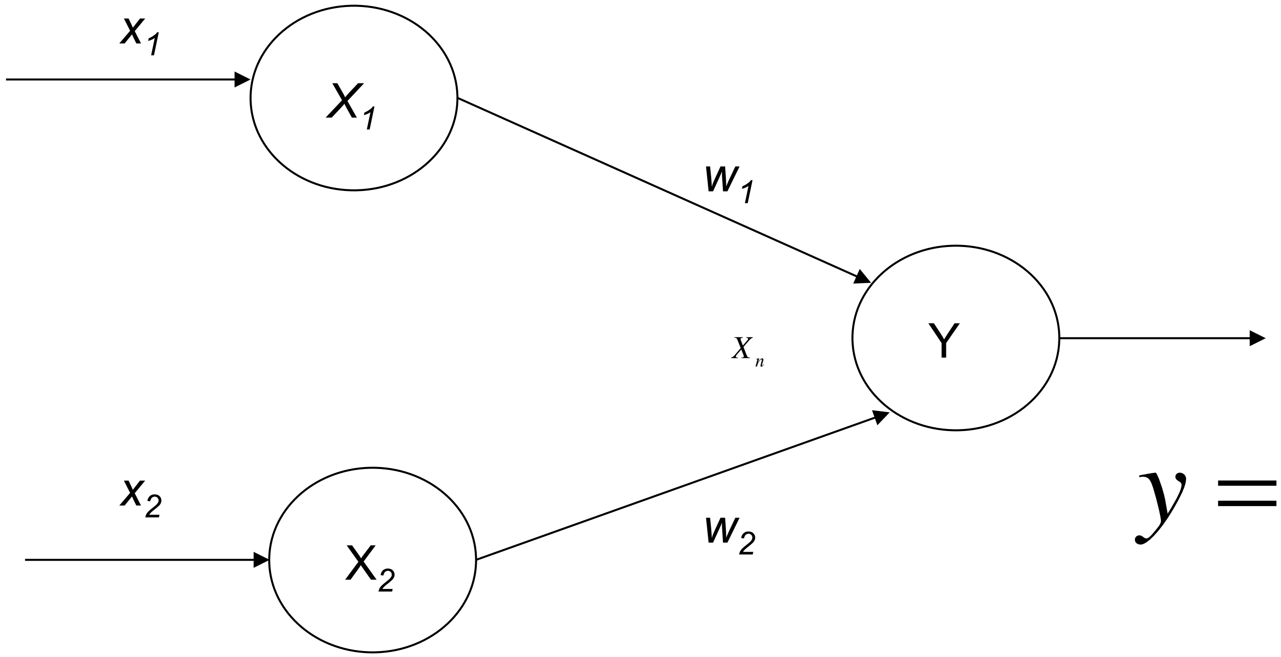
Fundamental concept

- NN are constructed and implemented to model the human brain.
- Performs various tasks such as pattern-matching, classification, optimization function, approximation, vector quantization and data clustering.
- These tasks are difficult for traditional computers

ANN

- ANN posess a large number of processing elements called nodes/neurons which operate in parallel.
- Neurons are connected with others by connection link.
- Each link is associated with weights which contain information about the input signal.
- Each neuron has an internal state of its own which is a function of the inputs that neuron receives- Activation level

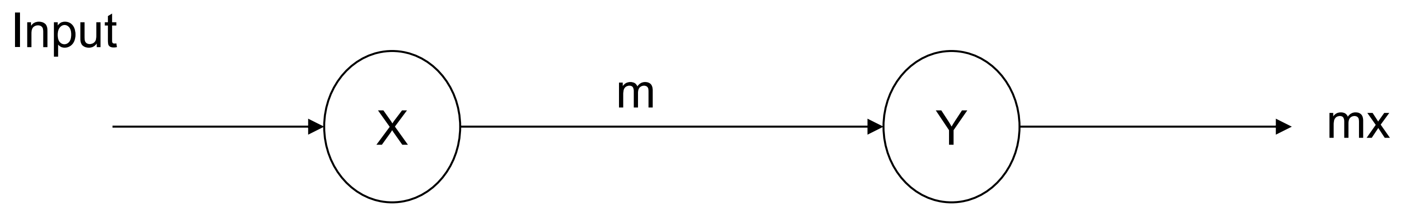
Artificial Neural Networks



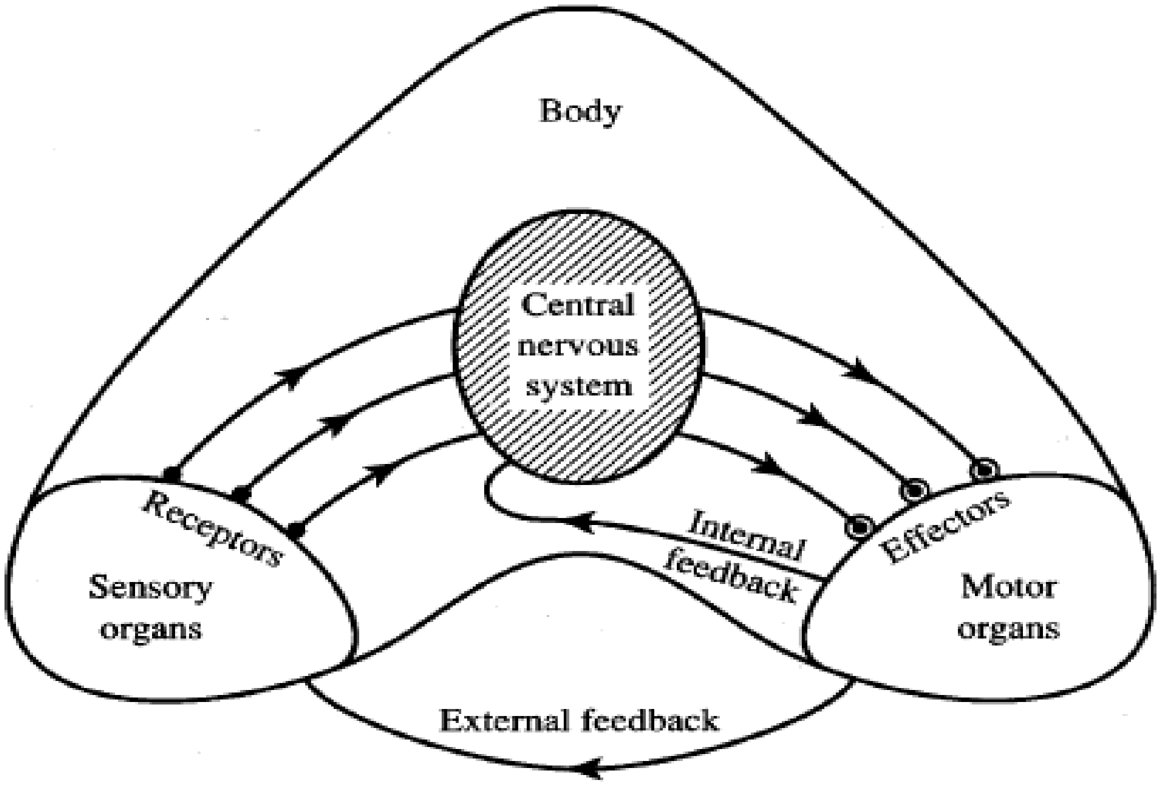
$$y = f(y_{in})$$

$$y_{in} = x_1 w_1 + x_2 w_2$$

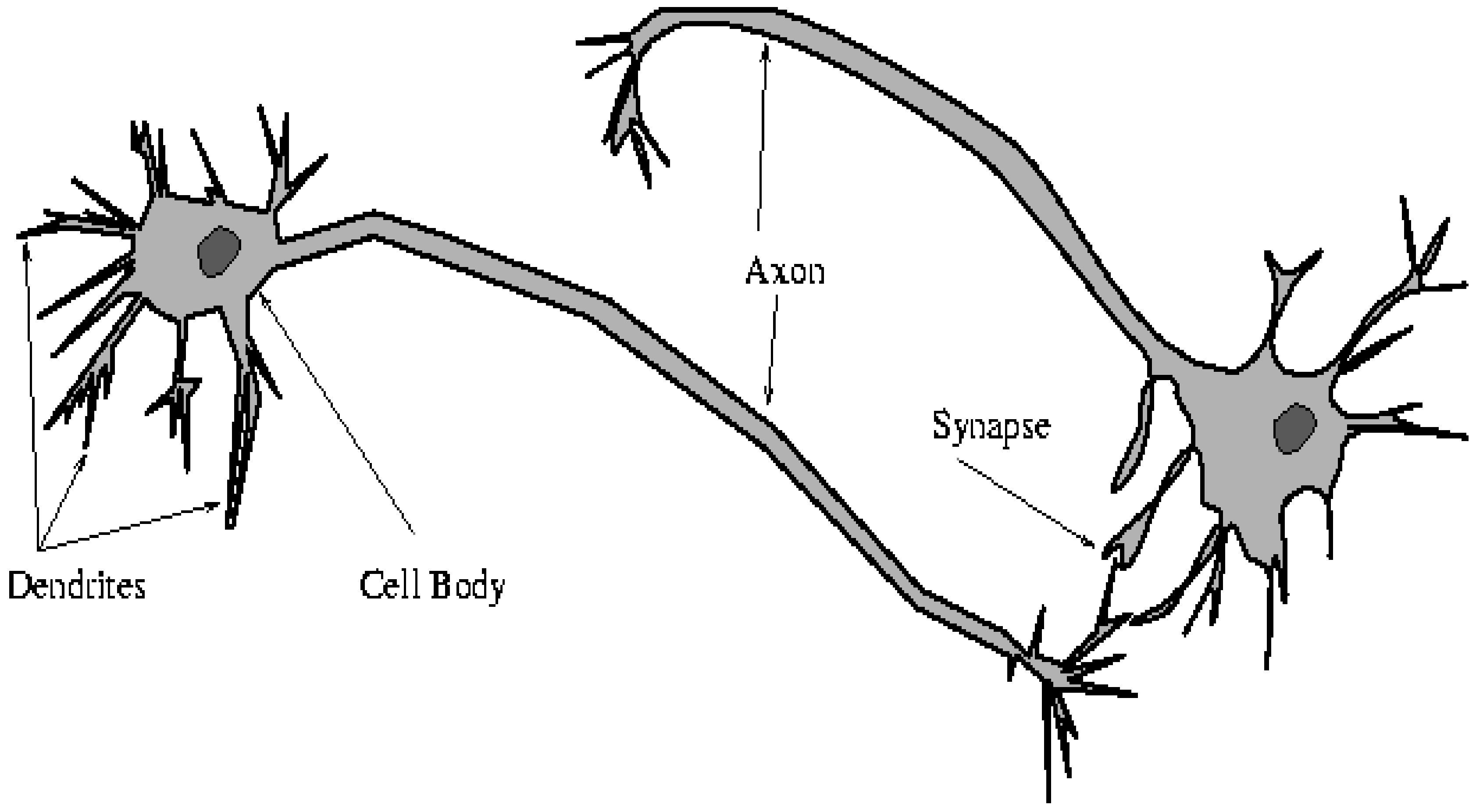
Neural net of pure linear eqn.



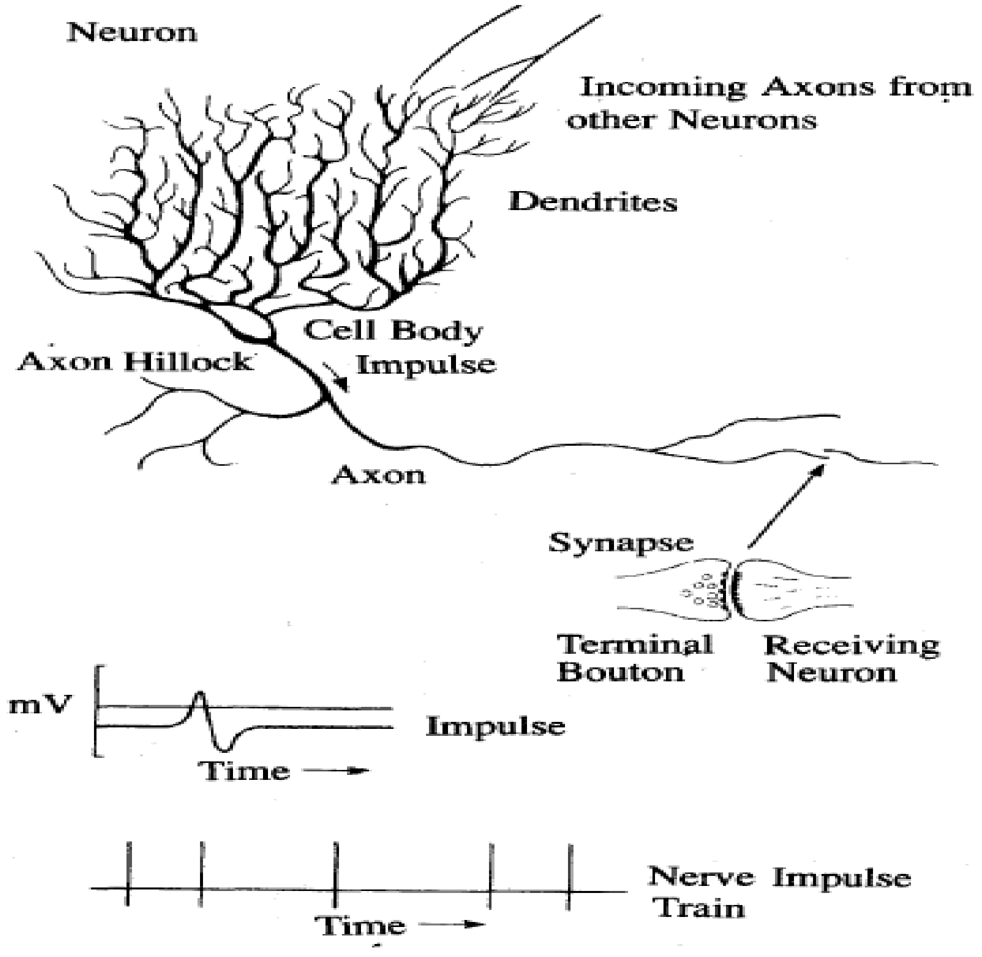
Information flow in nervous system



Biological Neural Network



Neuron and a sample of pulse train

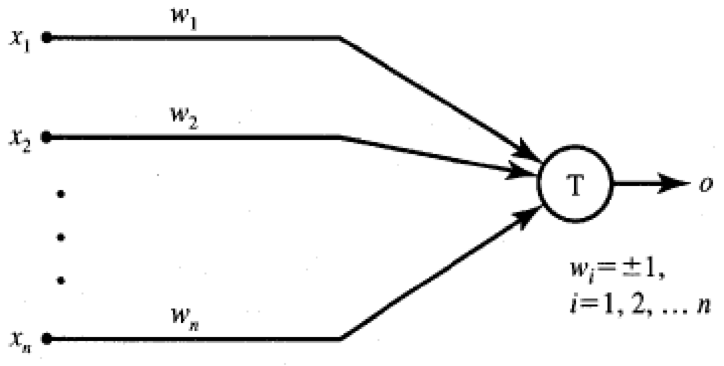


Biological Neuron

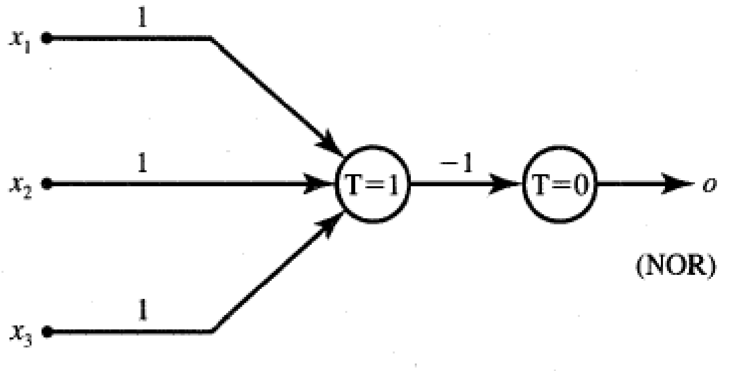
- Has 3 parts
  - Soma or cell body:- cell nucleus is located
  - Dendrites:- nerve connected to cell body
  - Axon: carries impulses of the neuron
- End of axon splits into fine strands
- Each strand terminates into a bulb-like organ called synapse
- Electric impulses are passed between the synapse and dendrites
- Synapses are of two types
  - Inhibitory:- impulses hinder the firing of the receiving cell
  - Excitatory:- impulses cause the firing of the receiving cell
- Neuron fires when the total of the weights to receive impulses exceeds the threshold value during the latent summation period
- After carrying a pulse an axon fiber is in a state of complete nonexcitability for a certain time called the refractory period.

McCulloch-Pitts Neuron Model

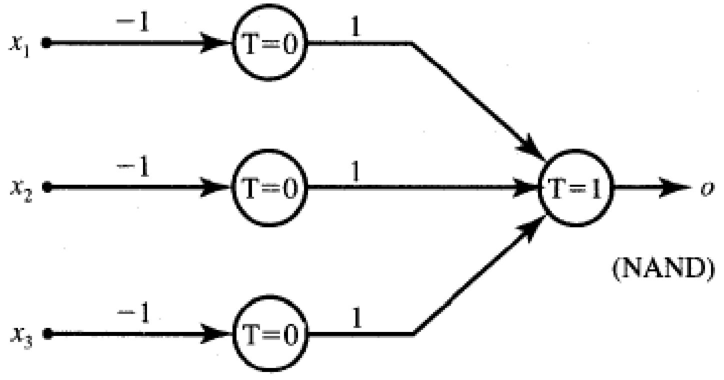
$$o^{k+1} = \begin{cases} 1 & \text{if } \sum_{i=1}^{n} w_i x_i^k \geq T \\ 0 & \text{if } \sum_{i=1}^{n} w_i x_i^k < T \end{cases}$$



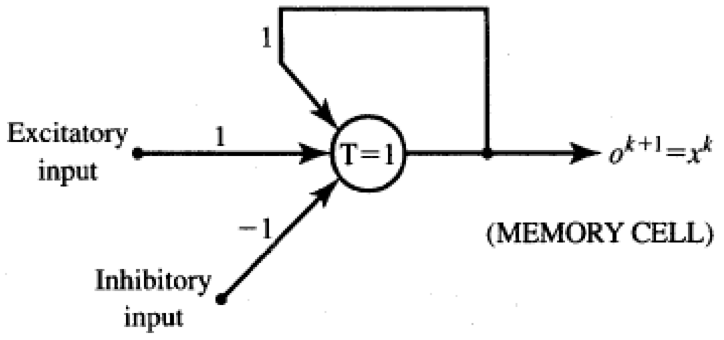
(a)



(b)



(c)

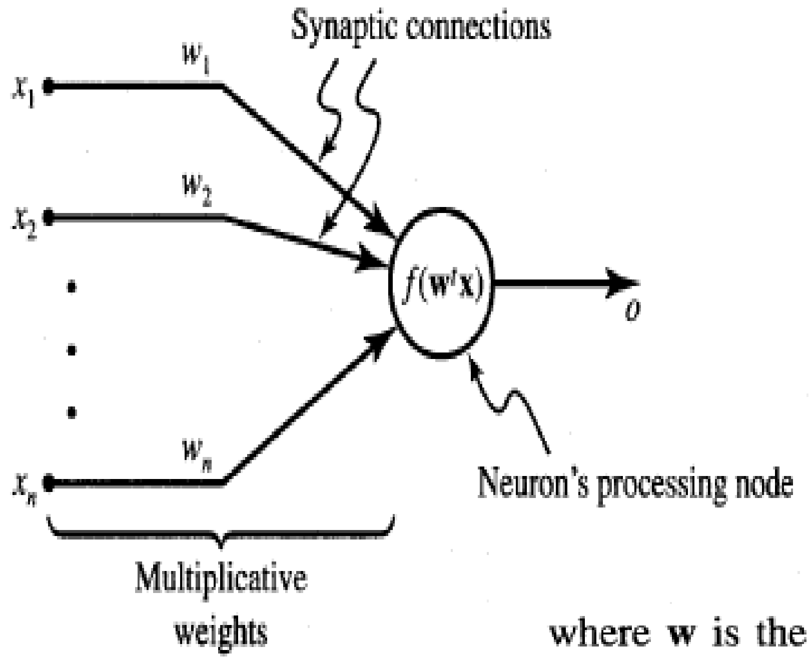


(d)

Features of McCulloch-Pitts model

- Allows binary 0,1 states only
- Operates under a discrete-time assumption
- Weights and the neurons’ thresholds are fixed in the model and no interaction among network neurons
- Just a primitive model

General symbol of neuron consisting of processing node and synaptic connections



$$o = f(\mathbf{w}^t\mathbf{x}), \text{ or}$$

$$o = f\left(\sum_{i=1}^{n} w_i x_i\right)$$

where $\mathbf{w}$ is the *weight vector* defined as

$$\mathbf{w} \overset{\Delta}{=} [w_1 \quad w_2 \quad \cdots \quad w_n]^t$$

and $\mathbf{x}$ is the input vector:

$$\mathbf{x} \overset{\Delta}{=} [x_1 \quad x_2 \quad \cdots \quad x_n]^t$$

Neuron Modeling for ANN

$$o = f(\mathbf{w}^t\mathbf{x}), \text{ or}$$
$$o = f\left(\sum_{i=1}^{n} w_i x_i\right)$$

Is referred to activation function. Domain is set of activation values *net.*

$$net \overset{\Delta}{=} \mathbf{w}^t\mathbf{x}$$

Scalar product of weight and input vector

Neuron as a processing node performs the operation of summation of its weighted input.

Activation function

- Bipolar binary and unipolar binary are called as hard limiting activation functions used in discrete neuron model
- Unipolar continuous and bipolar continuous are called soft limiting activation functions are called sigmoidal characteristics.

Activation functions

Bipolar continuous

$$f(net) \overset{\Delta}{=} \frac{2}{1+\exp(-\lambda net)} - 1$$

$\lambda > 0$

$$f(net) \overset{\Delta}{=} \operatorname{sgn}(net) = \begin{cases} +1, & net > 0 \\ -1, & net < 0 \end{cases}$$

Bipolar binary functions

Activation functions

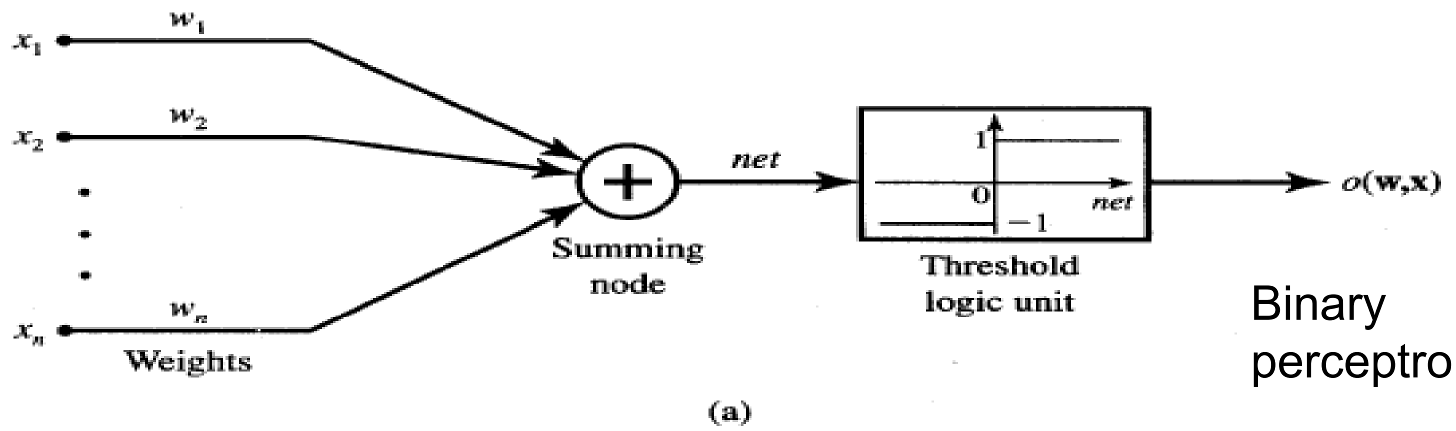
Unipolar continuous

$$f(net) \triangleq \frac{1}{1+\exp(-\lambda net)}$$

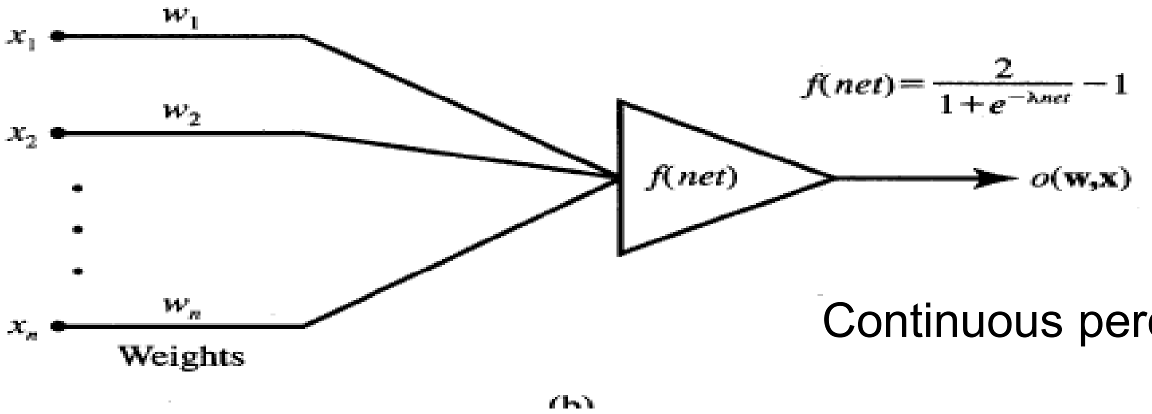
Unipolar Binary

$$f(net) \triangleq \begin{cases} 1, & net > 0 \\ 0, & net < 0 \end{cases}$$

Common models of neurons



Binary perceptrons



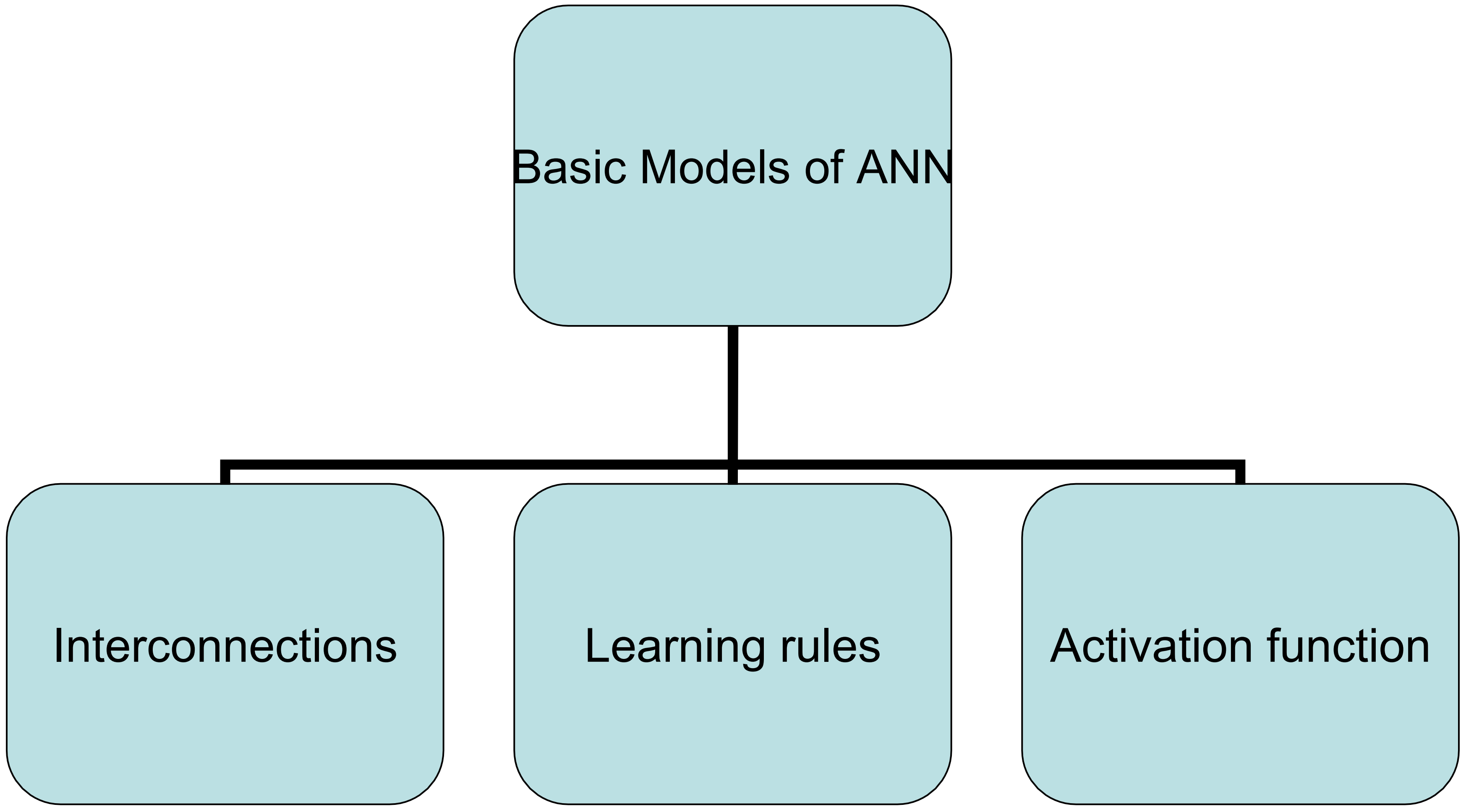
x_1
w_1
x_2
w_2
x_n
w_n
Weights
$f(net)$
$f(net) = \frac{2}{1+e^{-\lambda net}} - 1$
$o(\mathbf{w},\mathbf{x})$

Continuous perceptrons

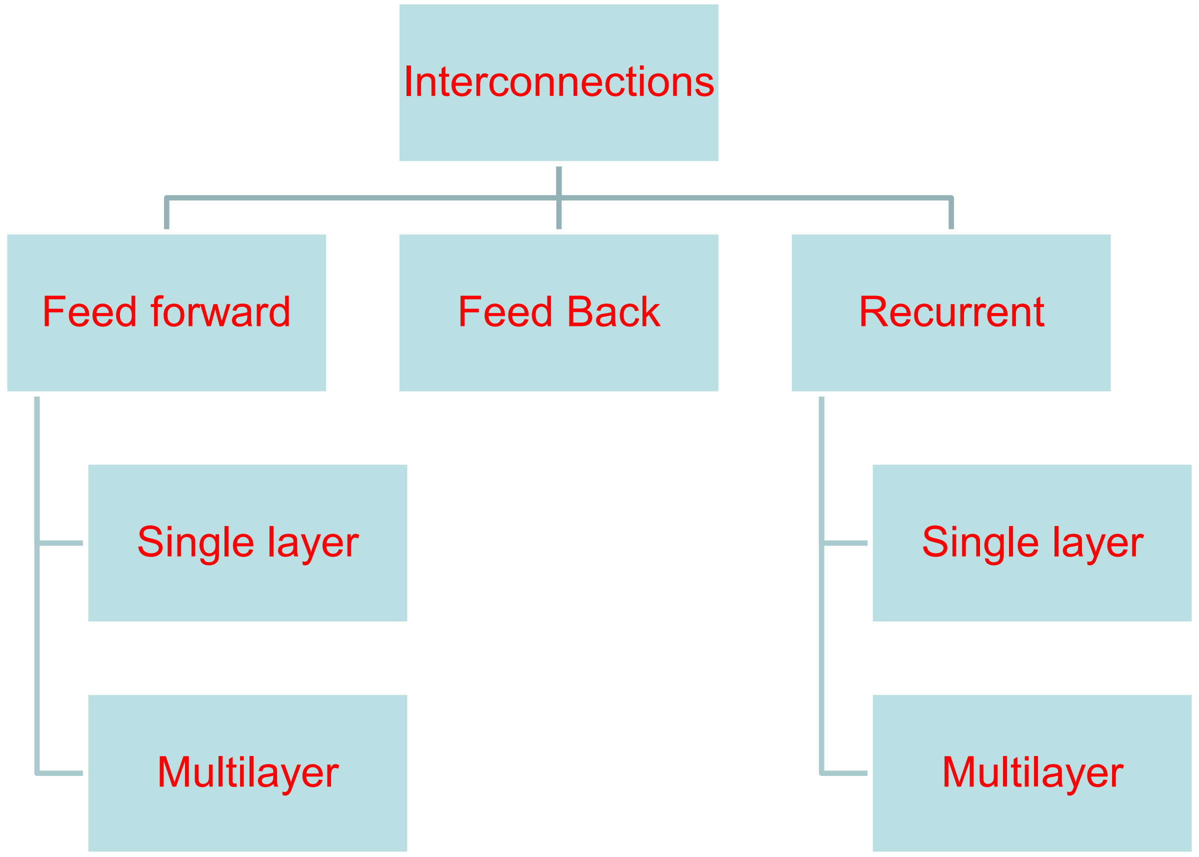
Comparison between brain verses computer

	Brain	ANN
Speed	Few ms.	Few nano sec. massive ||el processing
Size and complexity	10^{11} neurons & 10^{15} interconnections	Depends on designer
Storage capacity	Stores information in its interconnection or in synapse. No Loss of memory	Contiguous memory locations loss of memory may happen sometimes.
Tolerance	Has fault tolerance	No fault tolerance Inf gets disrupted when interconnections are disconnected
Control mechanism	Complicated involves chemicals in biological neuron	Simpler in ANN

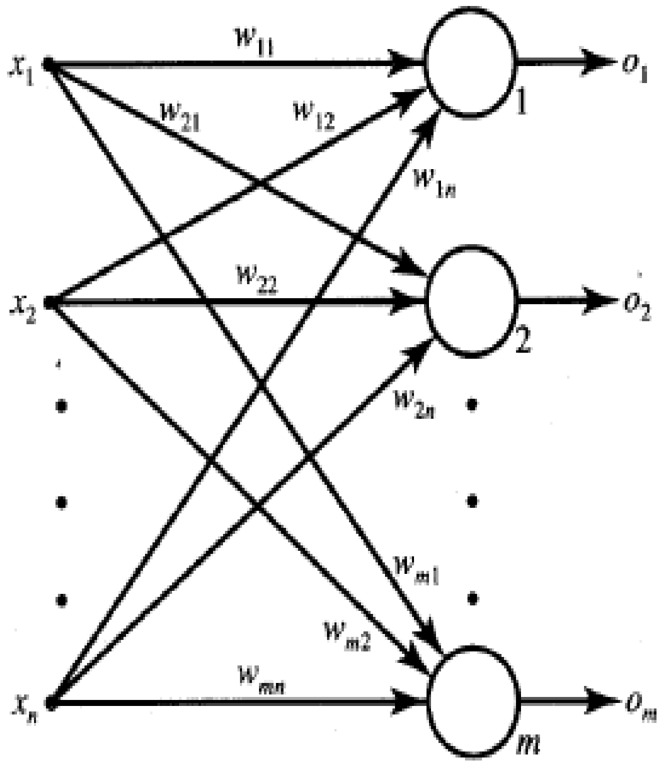
Basic models of ANN

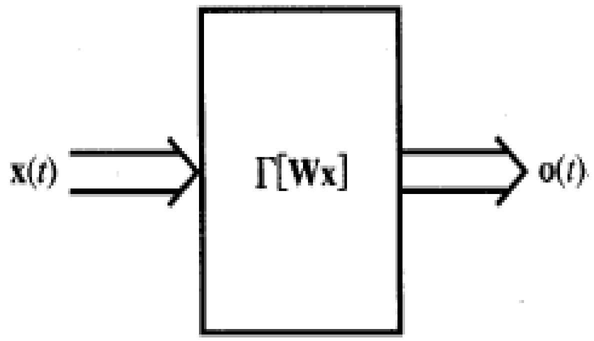


Classification based on interconnections



Single layer Feedforward Network





Feedforward Network

- Its output and input vectors are respectively

$$\mathbf{o} = [o_1 \quad o_2 \quad \cdots \quad o_m]^t$$

$$\mathbf{x} = [x_1 \quad x_2 \quad \cdots \quad x_n]^t$$

- Weight *w*$_{ij}$ connects the *i'th* neuron with *j'th* input. Activation rule of ith neuron is

$$net_i = \sum_{j=1}^{n} w_{ij}x_j, \quad \text{for } i = 1, 2, \ldots, m$$

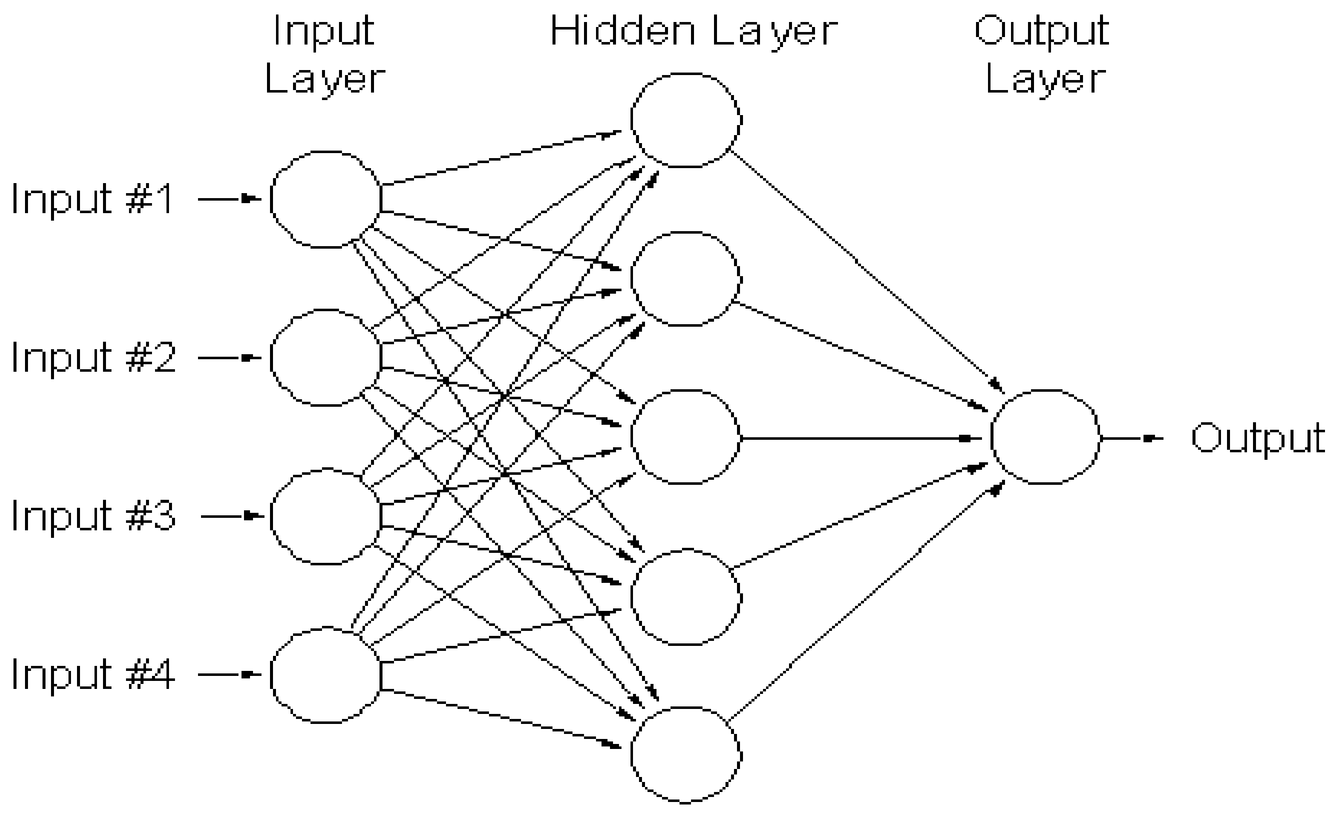
$$o_i = f(\mathbf{w}_i^t\mathbf{x}), \quad \text{for } i = 1, 2, \ldots, m$$

where

$$\mathbf{w}_i \overset{\Delta}{=} [w_{i1} \quad w_{i2} \quad \cdots \quad w_{in}]^t$$

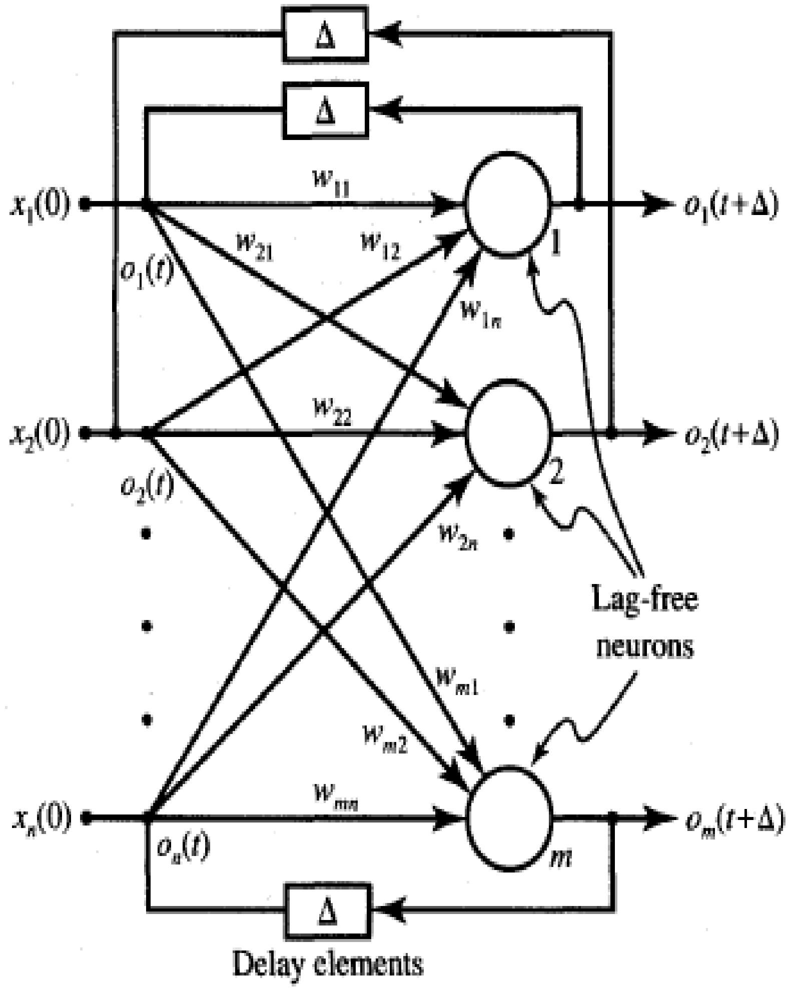
EXAMPLE

Multilayer feed forward network

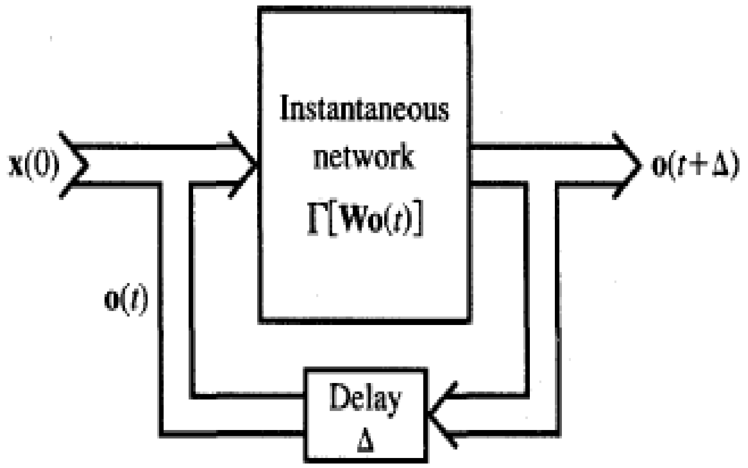


Can be used to solve complicated problems

Feedback network

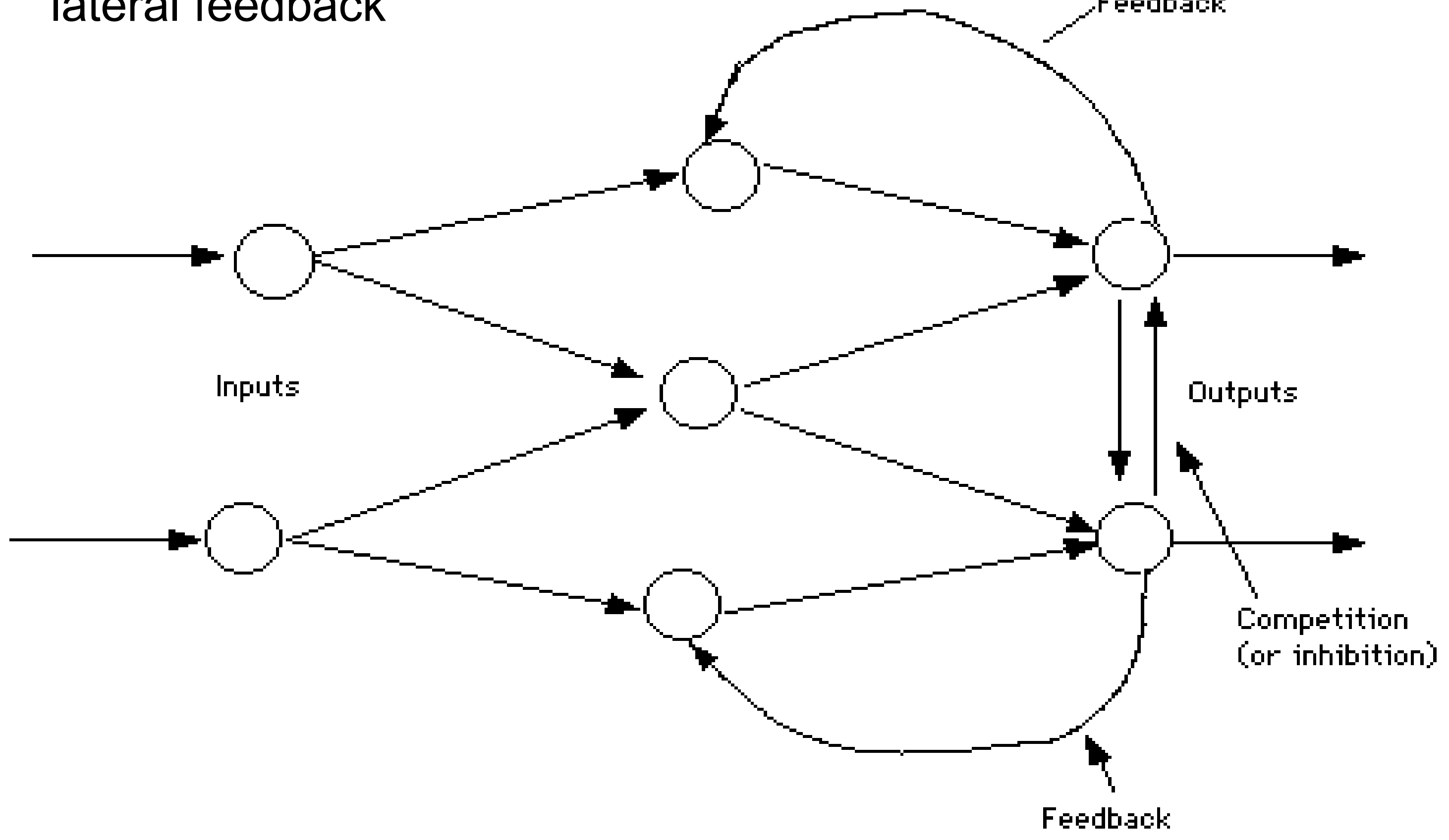


When outputs are directed back as inputs to same or preceding layer nodes it results in the formation of feedback networks



Lateral feedback

If the feedback of the output of the processing elements is directed back as input to the processing elements in the same layer then it is called lateral feedback

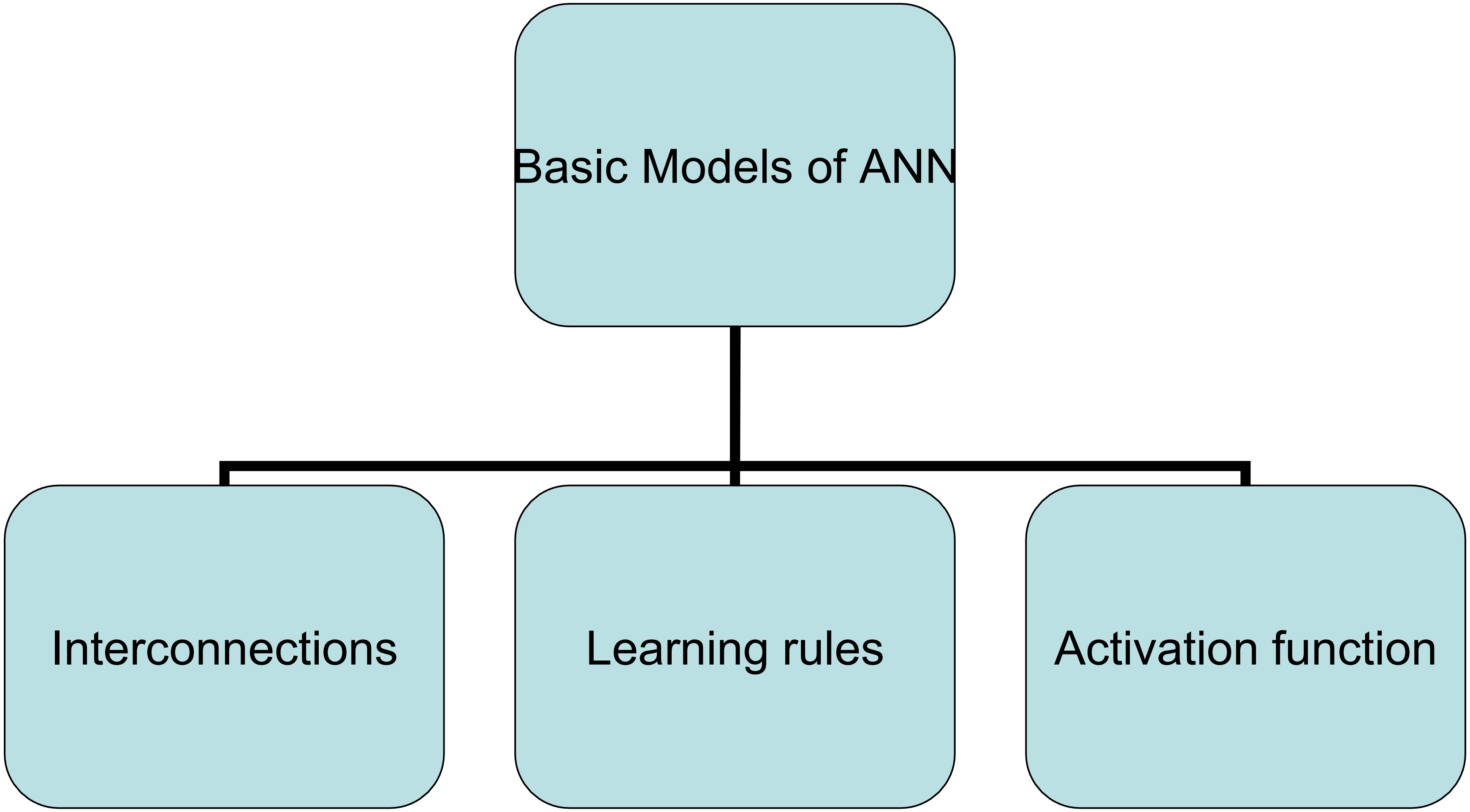


Recurrent n/ws

Feedback networks with closed loop are called **Recurrent Networks**. The response at the k+1'th instant depends on the entire history of the network starting at k=0.
Automaton: A system with discrete time inputs and a discrete data representation is called an automaton

- Single node with own feedback
- Competitive nets
- Single-layer recurrent nts
- Multilayer recurrent networks

Basic models of ANN



Learning

- It’s a process by which a NN adapts itself to a stimulus by making proper parameter adjustments, resulting in the production of desired response
- Two kinds of learning
  - Parameter learning:- connection weights are updated
  - Structure Learning:- change in network structure

Training

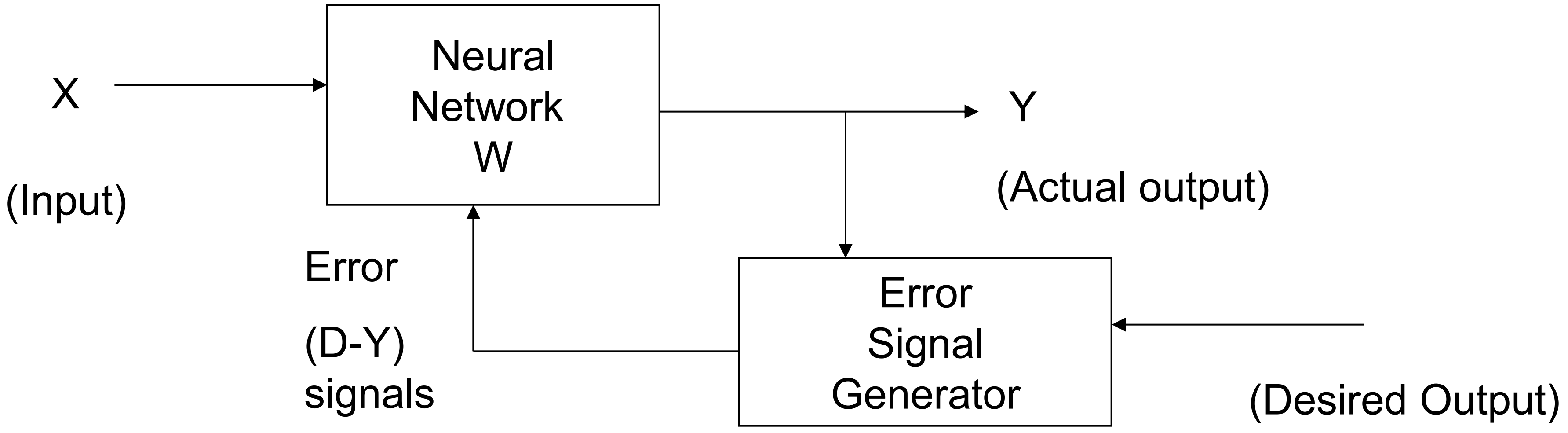
- The process of modifying the weights in the connections between network layers with the objective of achieving the expected output is called training a network.
- This is achieved through
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning

Classification of learning

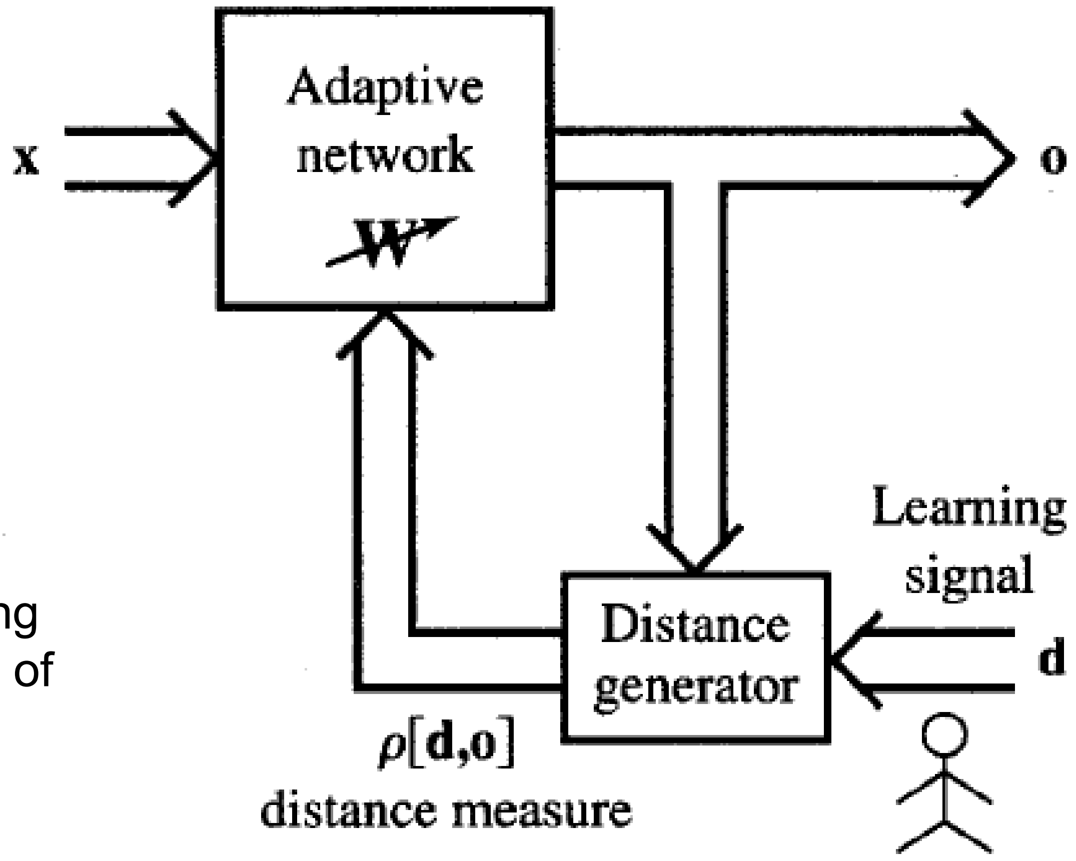
- Supervised learning
- Unsupervised learning
- Reinforcement learning

Supervised Learning

- Child learns from a teacher
- Each input vector requires a corresponding target vector.
- Training pair=[input vector, target vector]



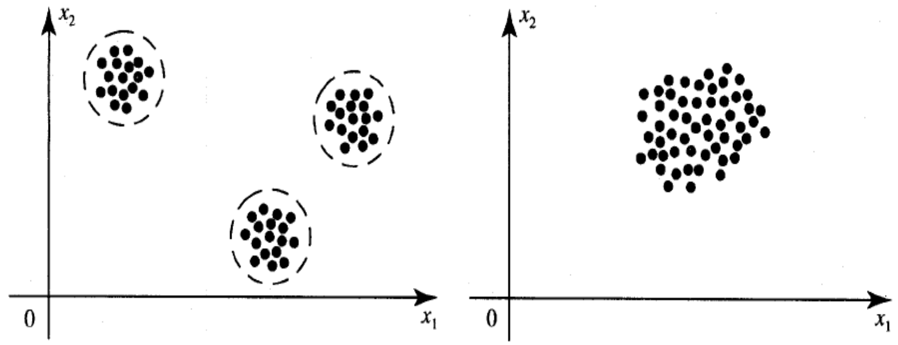
Supervised learning contd.



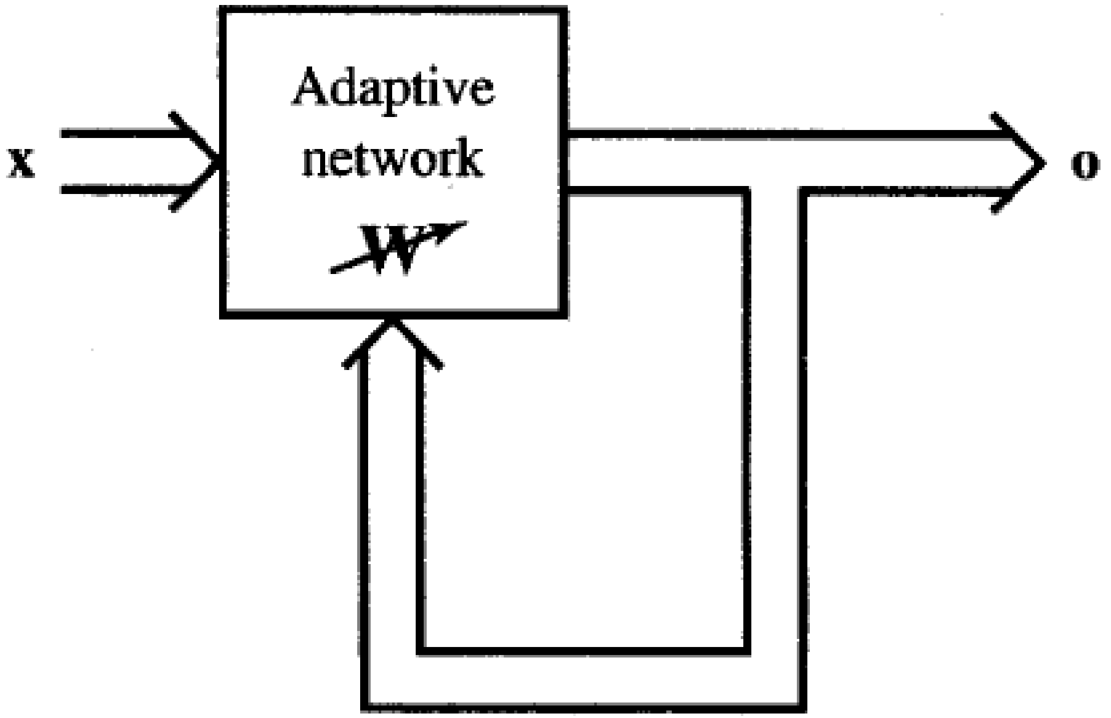
Supervised learning does minimization of error

Unsupervised Learning

- How a fish or tadpole learns
- All similar input patterns are grouped together as clusters.
- If a matching input pattern is not found a new cluster is formed



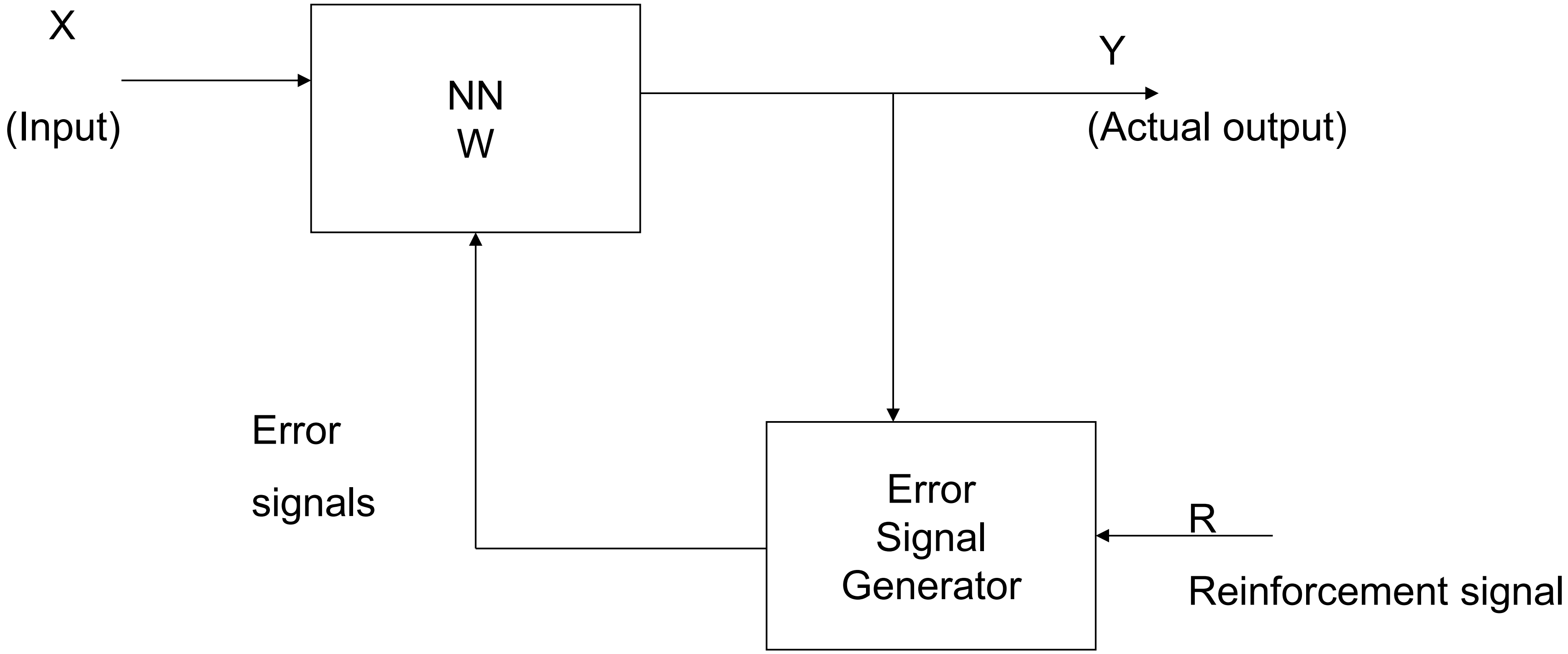
Unsupervised learning



Self-organizing

- In unsupervised learning there is no feedback
- Network must discover patterns, regularities, features for the input data over the output
- While doing so the network might change in parameters
- This process is called self-organizing

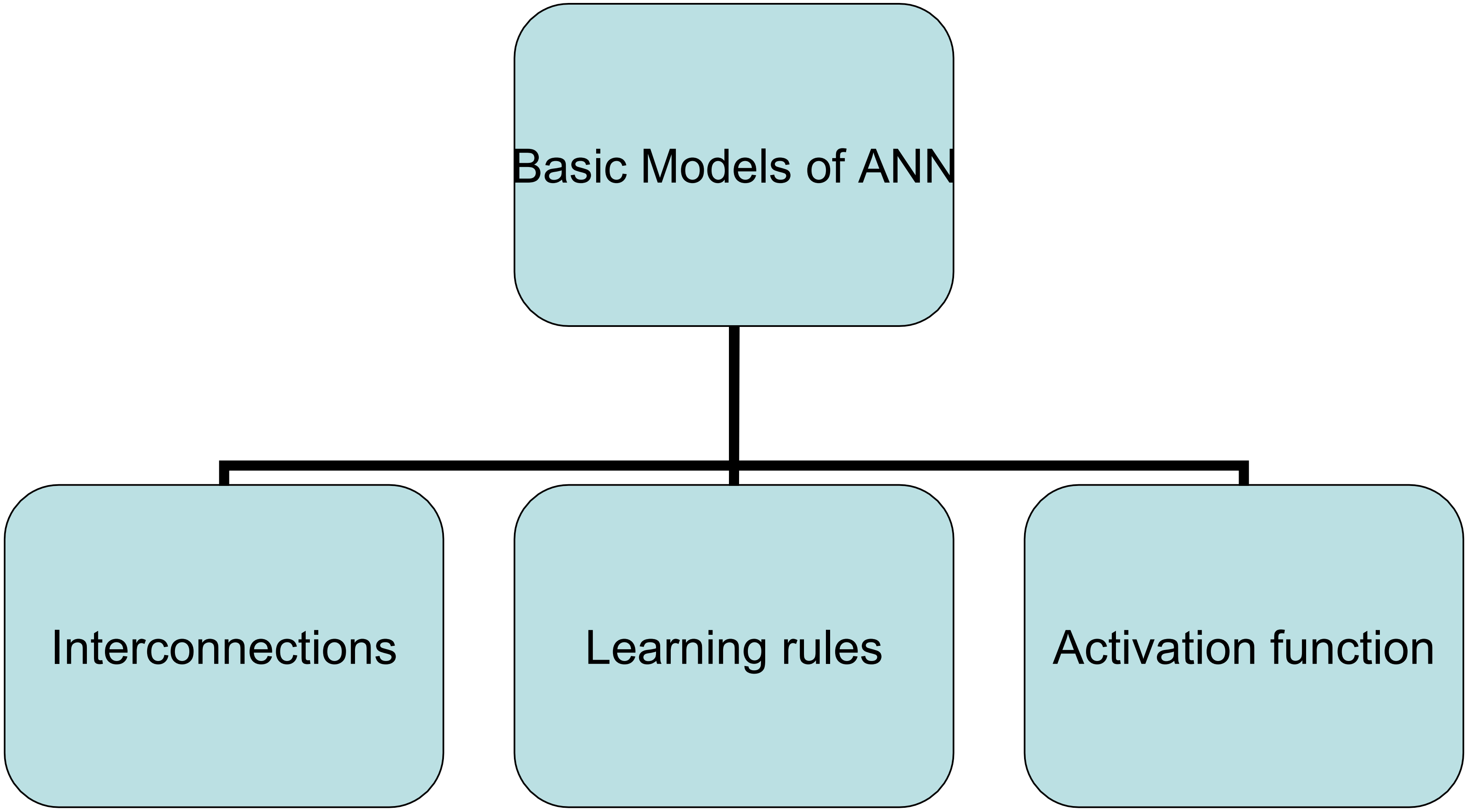
Reinforcement Learning



When Reinforcement learning is used?

- If less information is available about the target output values (critic information)
- Learning based on this critic information is called reinforcement learning and the feedback sent is called reinforcement signal
- Feedback in this case is only evaluative and not instructive

Basic models of ANN



Activation Function

1. Identity Function
   f(x)=x for all x
2. Binary Step function

$$f(x) = \{ \begin{matrix} 1 if x \geq \theta \\ 0 if x < \theta \end{matrix}$$

3. Bipolar Step function

$$f(x) = \{ \begin{matrix} 1 if x \geq \theta \\ -1 if x < \theta \end{matrix}$$

4. Sigmoidal Functions:- Continuous functions
5. Ramp functions:-

$$\begin{matrix} & 1 if x > 1 \\ f(x) = & x if\, 0 \leq x \leq 1 \\ & 0 if x < 0 \end{matrix}$$

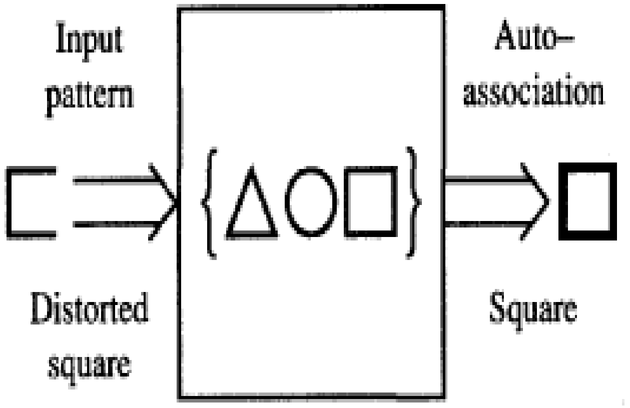
Some learning algorithms we will learn are

- Supervised:
  - Adaline, Madaline
  - Perceptron
  - Back Propagation
  - multilayer perceptrons
  - Radial Basis Function Networks
- Unsupervised
  - Competitive Learning
  - Kohenen self organizing map
  - Learning vector quantization
  - Hebbian learning

Neural processing

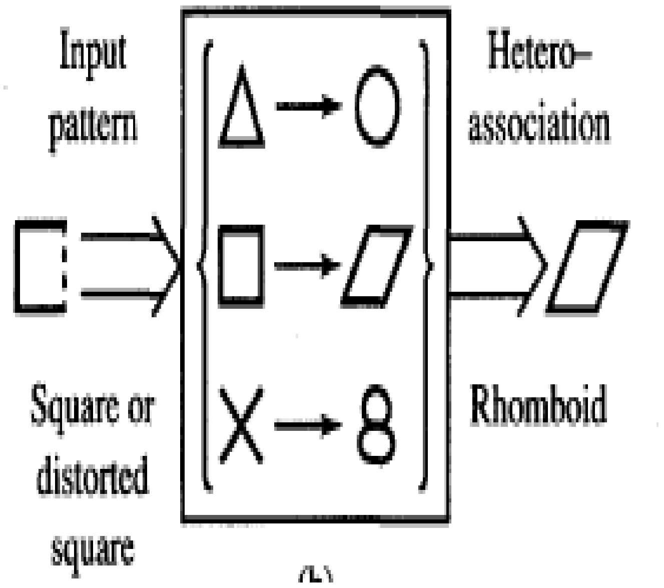
- **Recall:-** processing phase for a NN and its objective is to retrieve the information. The process of computing **o** for a given **x**
- **Basic forms of neural information processing**
  - Auto association
  - Hetero association
  - Classification

Neural processing-Autoassociation



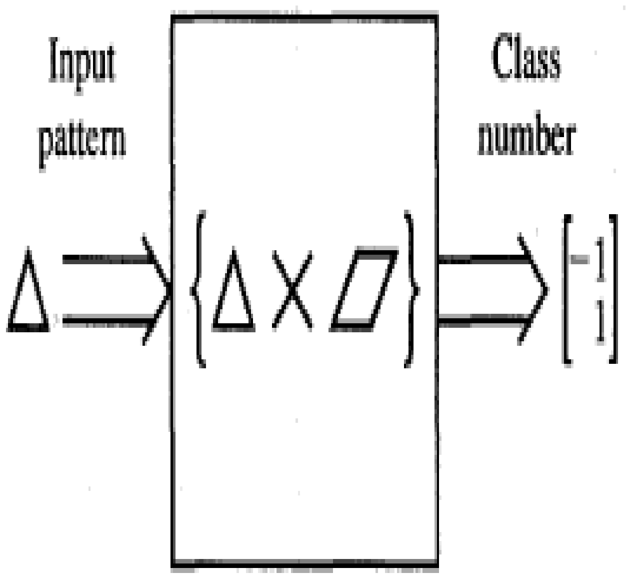
- Set of patterns can be stored in the network
- If a pattern similar to a member of the stored set is presented, an association with the input of closest stored pattern is made

Neural Processing-Heteroassociation



- Associations between pairs of patterns are stored
- Distorted input pattern may cause correct heteroassociation at the output

Neural processing-Classification



- Set of input patterns is divided into a number of classes or categories
- In response to an input pattern from the set, the classifier is supposed to recall the information regarding class membership of the input pattern.

Important terminologies of ANNs

- Weights
- Bias
- Threshold
- Learning rate
- Momentum factor
- Vigilance parameter
- Notations used in ANN

Weights

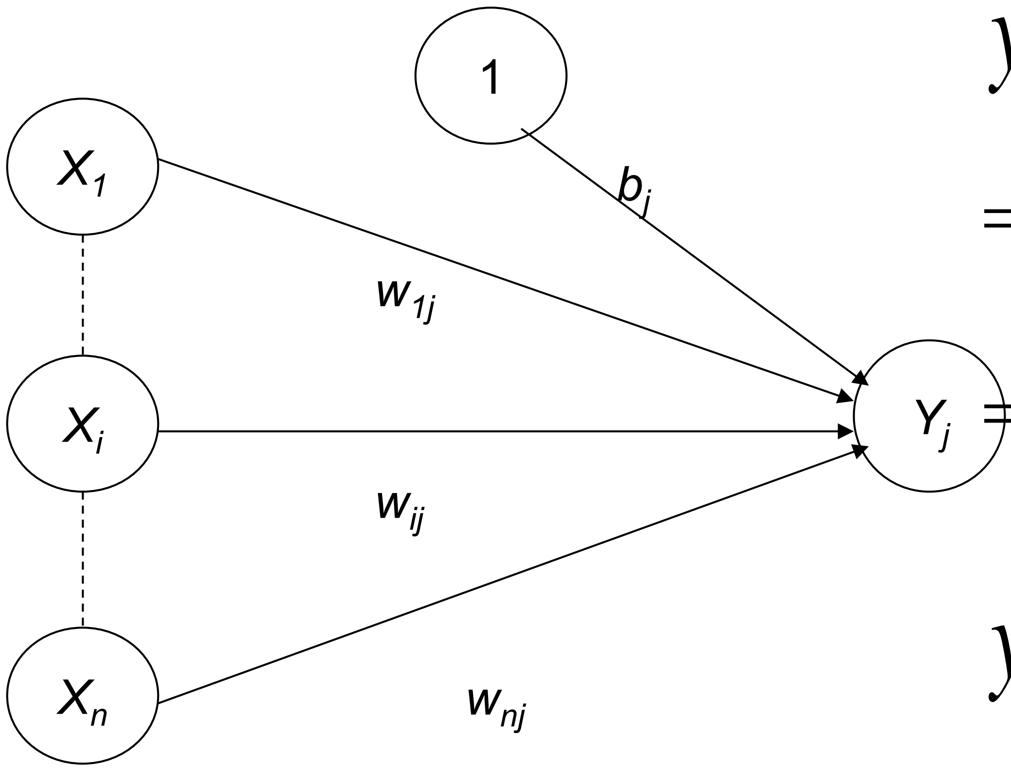
- Each neuron is connected to every other neuron by means of directed links
- Links are associated with weights
- Weights contain information about the input signal and is represented as a matrix
- Weight matrix also called <u>connection matrix</u>

Weight matrix

$$W = \begin{bmatrix} w_1^T \\ w_2^T \\ w_3^T \\ . \\ . \\ . \\ . \\ . \\ w_n^T \end{bmatrix} = \begin{bmatrix} w_{11} w_{12} w_{13} \cdots w_{1m} \\ w_{21} w_{22} w_{23} \cdots w_{2m} \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots\cdots\cdots \\ w_{n1} w_{n2} w_{n3} \cdots w_{nm} \end{bmatrix}$$

Weights contd…

- w_{ij} _is the weight from processing element *"i"* (source node) to processing element *"j"* (destination node)



$$y_{inj} = \sum_{i=0}^{n} x_i w_{ij}$$

$$= x_0 w_{0j} + x_1 w_{1j} + x_2 w_{2j} + ... + x_n w_{nj}$$

$$= w_{0j} + \sum_{i=1}^{n} x_i w_{ij}$$

$$y_{inj} = b_j + \sum_{i=1}^{n} x_i w_{ij}$$

Activation Functions

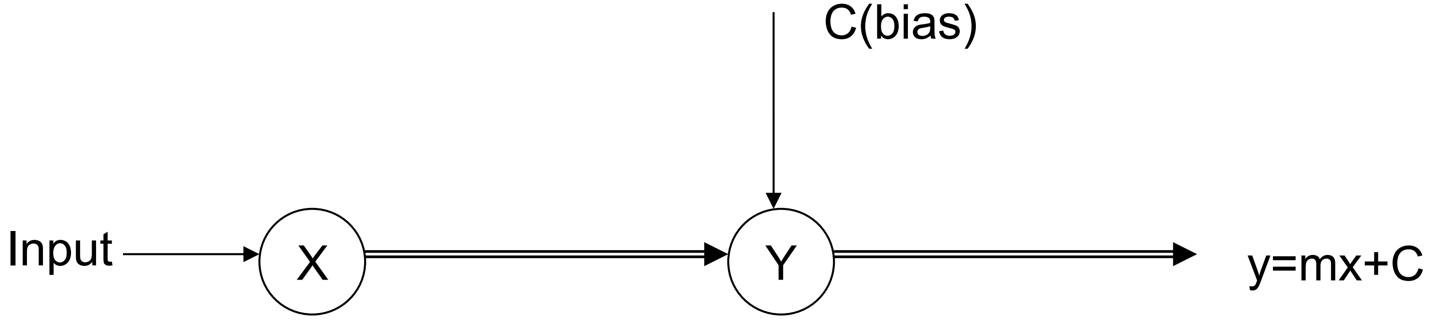
- Used to calculate the output response of a neuron.
- Sum of the weighted input signal is applied with an activation to obtain the response.
- Activation functions can be linear or non linear
- Already dealt
  - Identity function
  - Single/binary step function
  - Discrete/continuous sigmoidal function.

Bias

- Bias is like another weight. Its included by adding a component $x_0=1$ to the input vector X.
- $X=(1,X_1,X_2\ldots X_i,\ldots X_n)$
- Bias is of two types
  - Positive bias: increase the net input
  - Negative bias: decrease the net input

Why Bias is required?

- The relationship between input and output given by the equation of straight line y=mx+c



Threshold

- Set value based upon which the final output of the network may be calculated
- Used in activation function
- The activation function using threshold can be defined as

$$f(net) = \left\{ \begin{matrix} 1 if net \geq \theta \\ -1 if net < \theta \end{matrix} \right\}$$

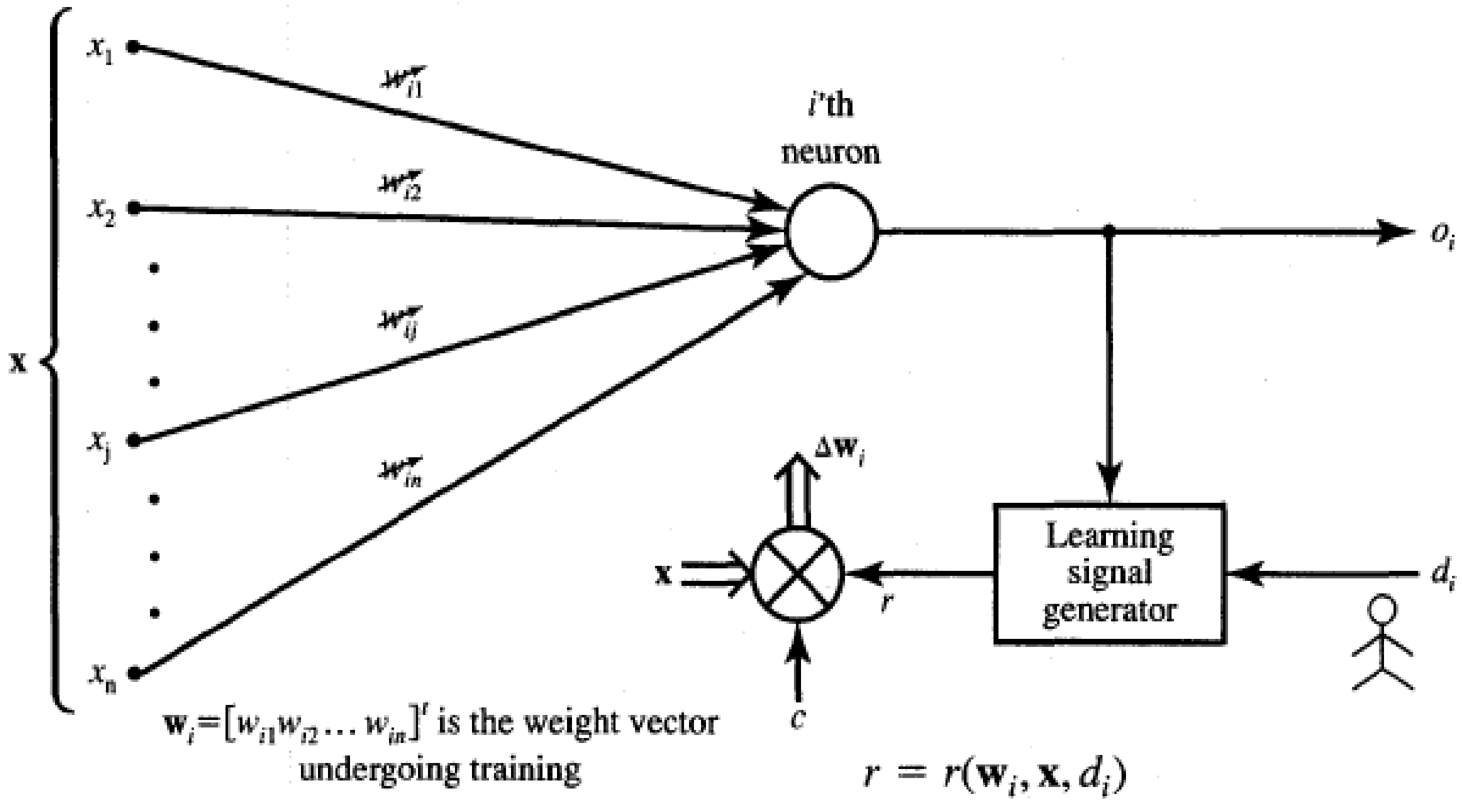
Learning rate

- Denoted by α.
- Used to control the amount of weight adjustment at each step of training
- Learning rate ranging from 0 to 1 determines the rate of learning in each time step

Other terminologies

- Momentum factor:
  - used for convergence when momentum factor is added to weight updation process.
- Vigilance parameter:
  - Denoted by ρ
  - Used to control the degree of similarity required for patterns to be assigned to the same cluster

Neural Network Learning rules



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

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

c – learning constant

$$\mathbf{w}_i(t+1) = \mathbf{w}_i(t) + cr\left[\mathbf{w}_i(t), \mathbf{x}(t), d_i(t)\right] \mathbf{x}(t)$$

Hebbian Learning Rule

FEED FORWARD UNSUPERVISED LEARNING

- The learning signal is equal to the <u>neuron's output</u>

$$r \triangleq f(\mathbf{w}_i^t \mathbf{x})$$

$$\Delta \mathbf{w}_i = c f(\mathbf{w}_i^t \mathbf{x})\mathbf{x}$$

The single weight w_{ij} is adapted using the following increment:

$$\Delta w_{ij} = c f(\mathbf{w}_i^t \mathbf{x}) x_j$$

This can be written briefly as

$$\Delta w_{ij} = c o_i x_j, \quad \text{for } j = 1, 2, \ldots, n$$

Features of Hebbian Learning

- Feedforward unsupervised learning
- “When an axon of a cell A is near enough to exicite a cell B and repeatedly and persistently takes place in firing it, some growth process or change takes place in one or both cells increasing the efficiency”
- If $o_i x_j$ is positive the results is increase in weight else vice versa

$$\mathbf{w}^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -2 \\ 1.5 \\ 0 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ -0.5 \\ -2 \\ -1.5 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} 0 \\ 1 \\ -1 \\ 1.5 \end{bmatrix}$$

$$c = 1$$

$$f(net) = \operatorname{sgn}(net).$$

Final answer: $\begin{bmatrix} 1 \\ -3.5 \\ 4.5 \\ 0.5 \end{bmatrix}$

- For the same inputs for bipolar continuous activation function the final updated weight is given by

$$f(net^1) = 0.905 \qquad f(net^2) = -0.077$$

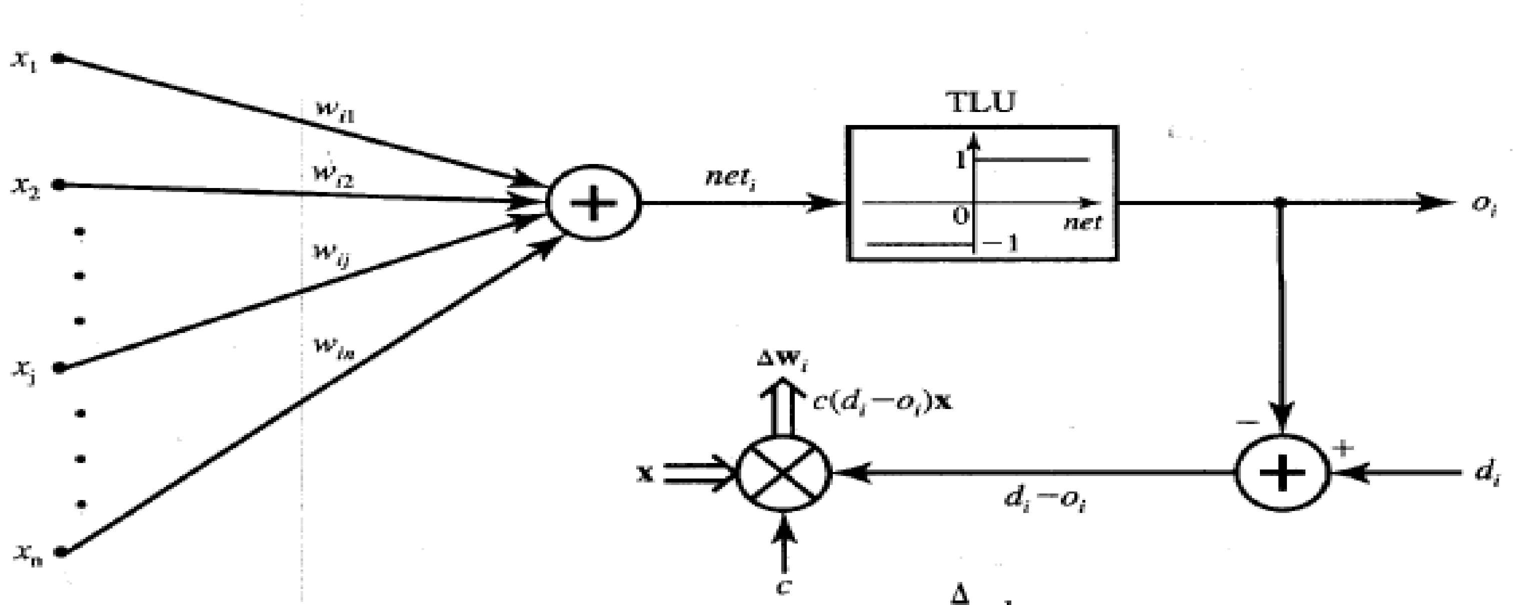
$$\mathbf{w}^2 = \begin{bmatrix} 1.905 \\ -2.81 \\ 1.357 \\ 0.5 \end{bmatrix} \qquad \mathbf{w}^3 = \begin{bmatrix} 1.828 \\ -2.772 \\ 1.512 \\ 0.616 \end{bmatrix}$$

$$f(net^3) = -0.932$$

$$\mathbf{w}^4 = \begin{bmatrix} 1.828 \\ -3.70 \\ 2.44 \\ -0.783 \end{bmatrix}$$

Perceptron Learning rule

- Learning signal is the difference between the desired and actual neuron's response
- Learning is supervised



$$r \overset{\Delta}{=} d_i - o_i$$

$$o_i = \operatorname{sgn}(\mathbf{w}_i^t \mathbf{x})$$

$$\Delta \mathbf{w}_i = c\,[d_i - \operatorname{sgn}(\mathbf{w}_i^t \mathbf{x})]\,\mathbf{x}$$

$$\mathbf{x}_1 = \begin{bmatrix} 1 \\ -2 \\ 0 \\ -1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1.5 \\ -0.5 \\ -1 \end{bmatrix}, \quad \mathbf{x}_3 = \begin{bmatrix} -1 \\ 1 \\ 0.5 \\ -1 \end{bmatrix}$$

$$\mathbf{w}^1 = \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0.5 \end{bmatrix}$$

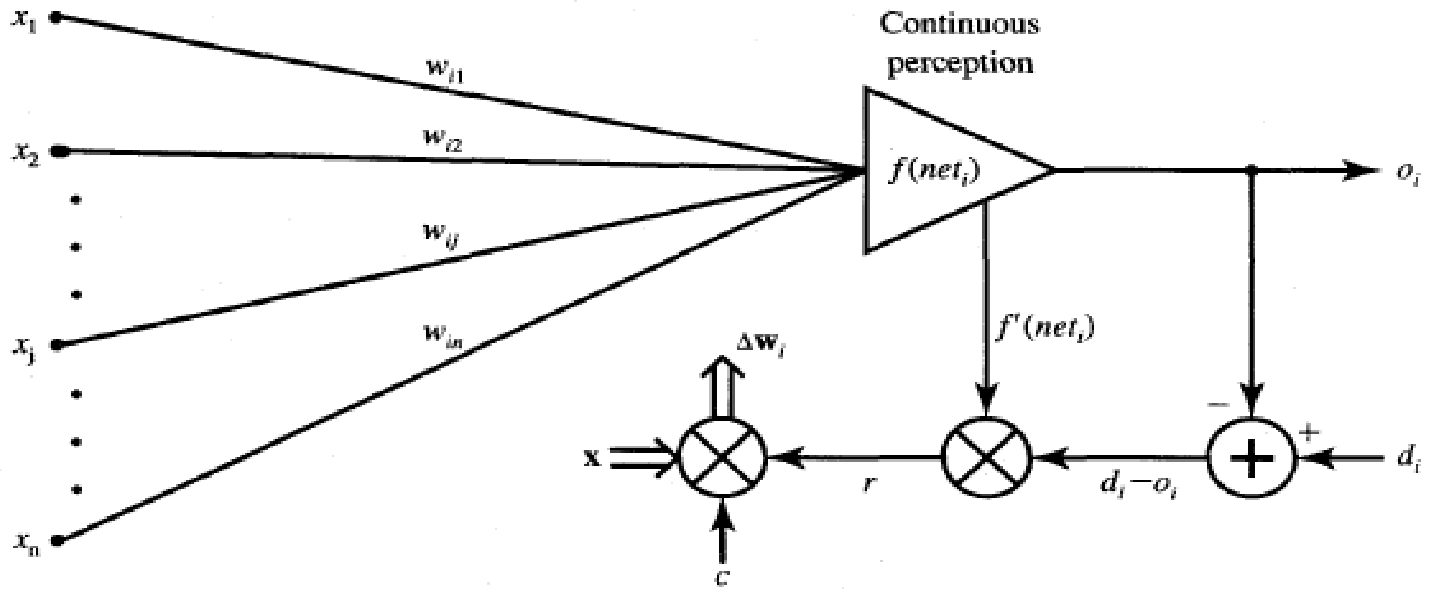
$c = 0.1.$

$d_1 = -1, d_2 = -1, \text{ and } d_3 = 1;$

$$\mathbf{w}^4 = \begin{bmatrix} 0.6 \\ -0.4 \\ 0.1 \\ 0.5 \end{bmatrix}$$

Delta Learning Rule

- Only valid for continuous activation function
- Used in supervised training mode
- Learning signal for this rule is called delta
- The aim of the delta rule is to minimize the error over all training patterns



Delta Learning Rule Contd.

$$r \overset{\Delta}{=} [d_i - f(\mathbf{w}_i^t\mathbf{x})]f'(\mathbf{w}_i^t\mathbf{x})$$

Learning rule is derived from the condition of least squared error.

Calculating the gradient vector with respect to **wi**

$$E \overset{\Delta}{=} \frac{1}{2}(d_i - o_i)^2$$

$$\nabla E = -(d_i - o_i)f'(\mathbf{w}_i^t\mathbf{x})\mathbf{x}$$

The components of the gradient vector are

$$\frac{\partial E}{\partial w_{ij}} = -(d_i - o_i)f'(\mathbf{w}_i^t\mathbf{x})x_j, \quad \text{for } j = 1, 2, \ldots, n$$

Minimization of error requires the weight changes to be in the negative gradient direction

$$\Delta \mathbf{w}_i = -\eta \nabla E$$

$$\Delta \mathbf{w}_i = \eta(d_i - o_i)f'(net_i)\mathbf{x}$$

Widrow-Hoff learning Rule

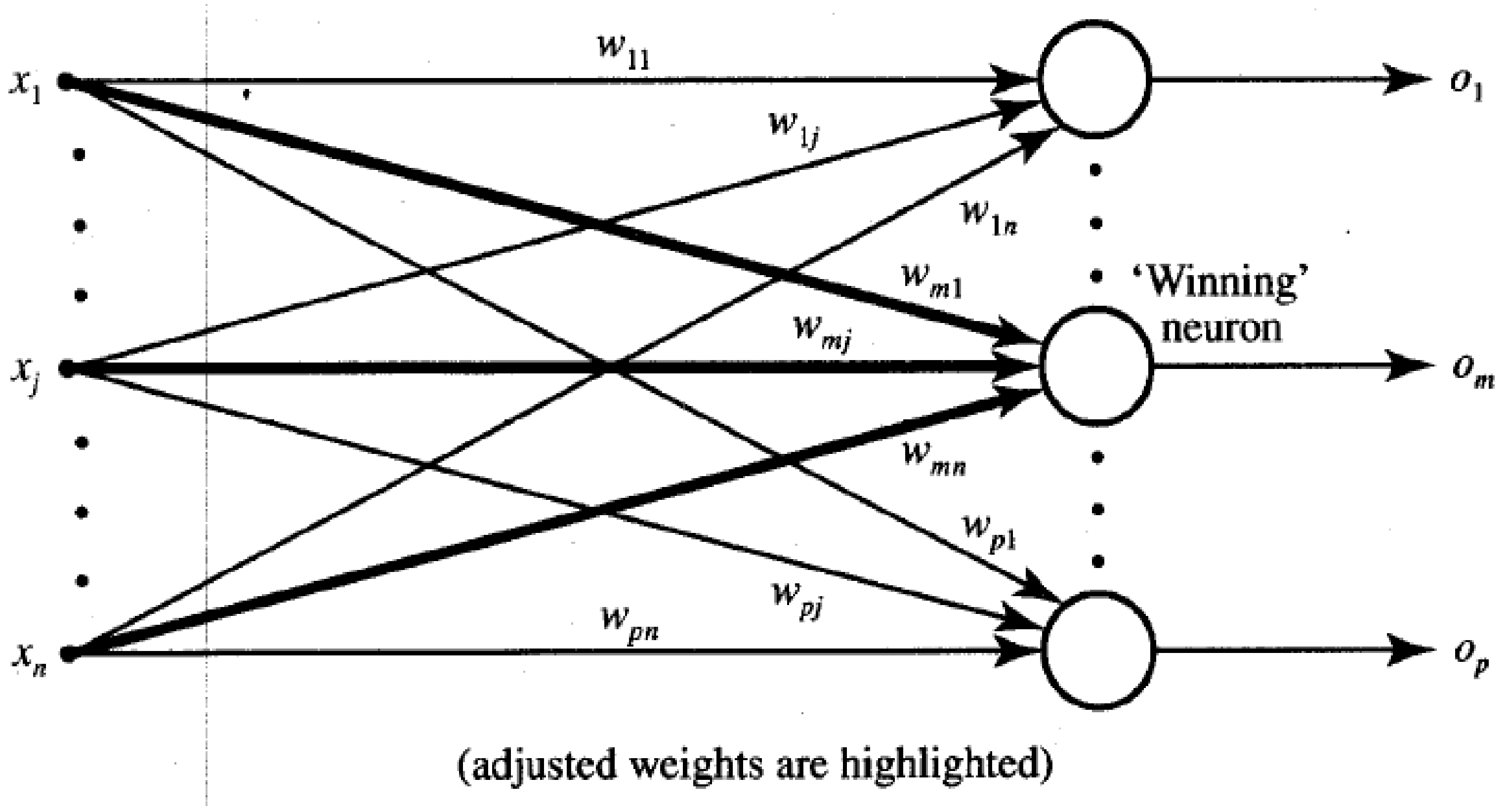
- Also called as least mean square learning rule
- Introduced by Widrow(1962), used in supervised learning
- Independent of the activation function
- Special case of delta learning rule wherein activation function is an identity function ie *f(net)=net*
- Minimizes the squared error between the desired output value d_i and net_i

$$r \overset{\Delta}{=} d_i - \mathbf{w}_i^t \mathbf{x}$$

The weight vector increment under this learning rule is

$$\Delta \mathbf{w}_i = c(d_i - \mathbf{w}_i^t \mathbf{x})\mathbf{x}$$

Winner-Take-All learning rules



Winner-Take-All Learning rule Contd…

- Can be explained for a layer of neurons
- Example of competitive learning and used for unsupervised network training
- Learning is based on the premise that one of the neurons in the layer has a maximum response due to the input x
- This neuron is declared the winner with a weight

$$\mathbf{w}_m = \begin{bmatrix} w_{m1} & w_{m2} & \cdots & w_{mn} \end{bmatrix}^t$$

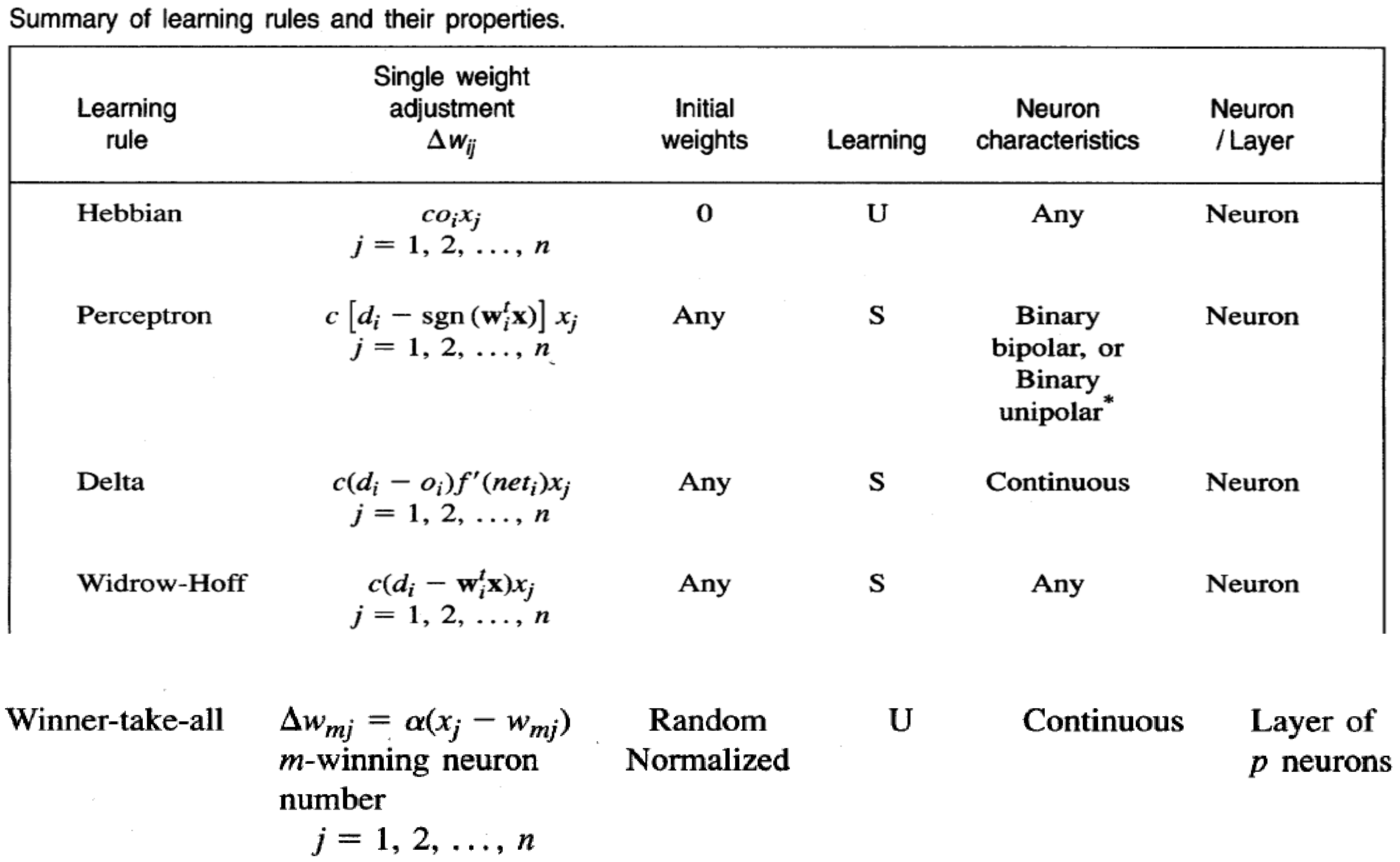
Its increment is computed as follows

$$\Delta \mathbf{w}_m = \alpha(\mathbf{x} - \mathbf{w}_m)$$

The winner selection is based on the following criterion of maximum activation among all p neurons participating in a competition:

$$\mathbf{w}_m^t\mathbf{x} = \max_{i=1,2,\ldots,p} (\mathbf{w}_i^t\mathbf{x})$$

Summary of learning rules



Summary of learning rules and their properties.

Learning rule	Single weight adjustment Δw_{ij}	Initial weights	Learning	Neuron characteristics	Neuron / Layer
Hebbian	$c o_i x_j$ $j = 1, 2, \ldots, n$	0	U	Any	Neuron
Perceptron	$c\left[d_i - \operatorname{sgn}(\mathbf{w}_i^t \mathbf{x})\right] x_j$ $j = 1, 2, \ldots, n$	Any	S	Binary bipolar, or Binary unipolar*	Neuron
Delta	$c(d_i - o_i) f'(net_i) x_j$ $j = 1, 2, \ldots, n$	Any	S	Continuous	Neuron
Widrow-Hoff	$c(d_i - \mathbf{w}_i^t \mathbf{x}) x_j$ $j = 1, 2, \ldots, n$	Any	S	Any	Neuron
Winner-take-all	$\Delta w_{mj} = \alpha(x_j - w_{mj})$ m-winning neuron number $j = 1, 2, \ldots, n$	Random Normalized	U	Continuous	Layer of p neurons

Linear Separability

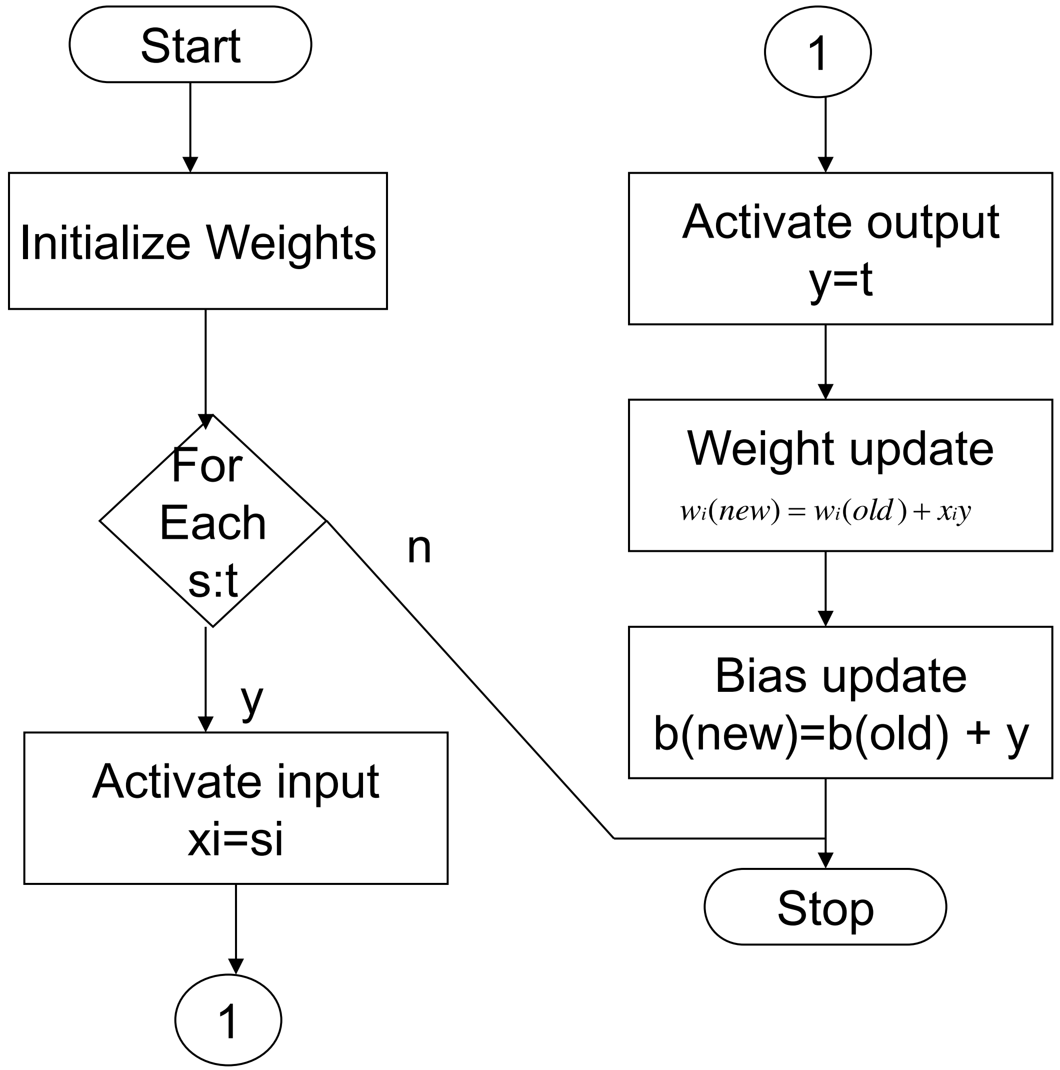
- Separation of the input space into regions is based on whether the network response is positive or negative
- Line of separation is called linear-separable line.
- Example:-
  - AND function & OR function are linear separable Example
  - EXOR function Linearly inseparable. Example

Hebb Network

- Hebb learning rule is the simpliest one
- The learning in the brain is performed by the change in the synaptic gap
- When an axon of cell A is near enough to excite cell B and repeatedly keep firing it, some growth process takes place in one or both cells
- According to Hebb rule, weight vector is found to increase proportionately to the product of the input and learning signal.

$$w_i(new) = w_i(old) + x_i y$$

Flow chart of Hebb training algorithm



- Hebb rule can be used for pattern association, pattern categorization, pattern classification and over a range of other areas
- Problem to be solved:

Design a Hebb net to implement OR function

How to solve

X1	X2	B	y
1	1	1	1
1	-1	1	1
-1	1	1	1
-1	-1	1	-1

Use bipolar data in the place of binary data

Initially the weights and bias are set to zero

w1=w2=b=0

Inputs			y	Weight changes			weights		
X1	X2	b	Y	W1	W2	B	W1(0)	W2(0)	(0)b
1	1	1	1	1	1	1	1	1	1
1	-1	1	1	1	-1	1	2	0	2
-1	1	1	1	-1	1	1	1	1	3
-1	-1	1	-1	1	1	-1	2	2	2

Home work

- Using the hebb rule, find the weights required to perform the following classification that given input patterns shown in figure