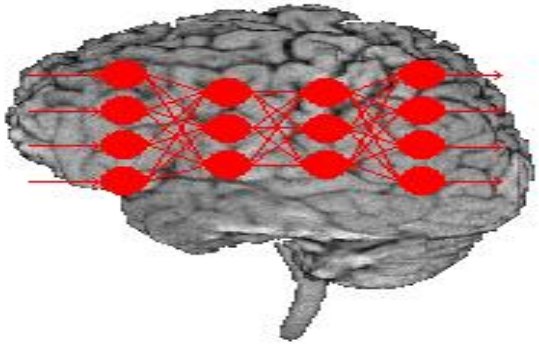
# Deep Learning
# Lecture-4

## Artificial Neural Network



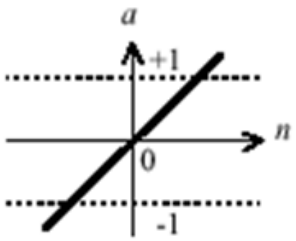
*Asst. Lect. Ali Al-khawaja*

2025-2026

Class Room

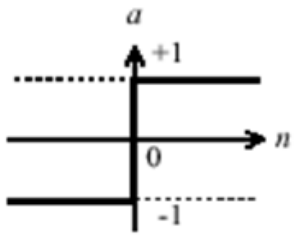
# Activation function

- 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.



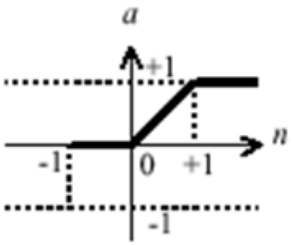

*a = purelin(n)*

Linear Transfer Function



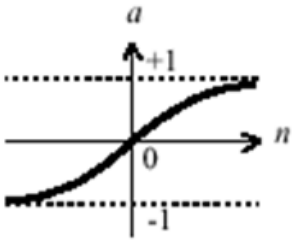

*a = hardlims(n)*

Symmetric Hard Limit Trans. Funct.



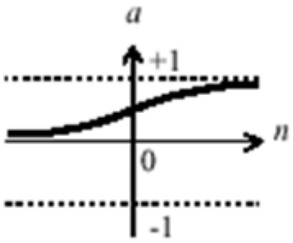

*a = satlin(n)*

Satlin Transfer Function



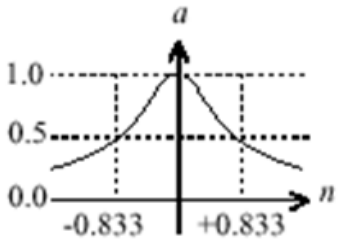

*a = tansig(n)*

Tan-Sigmoid Transfer Function




*a = logsig(n)*

Log-Sigmoid Transfer Function




*a = radbas(n)*

Radial Basis Function

# Activation functions

**Bipolar continuous**

$$f(net) \triangleq \frac{2}{1+\exp(-\lambda net)} - 1$$

$$\lambda > 0$$

**Bipolar binary functions**

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

# Activation functions

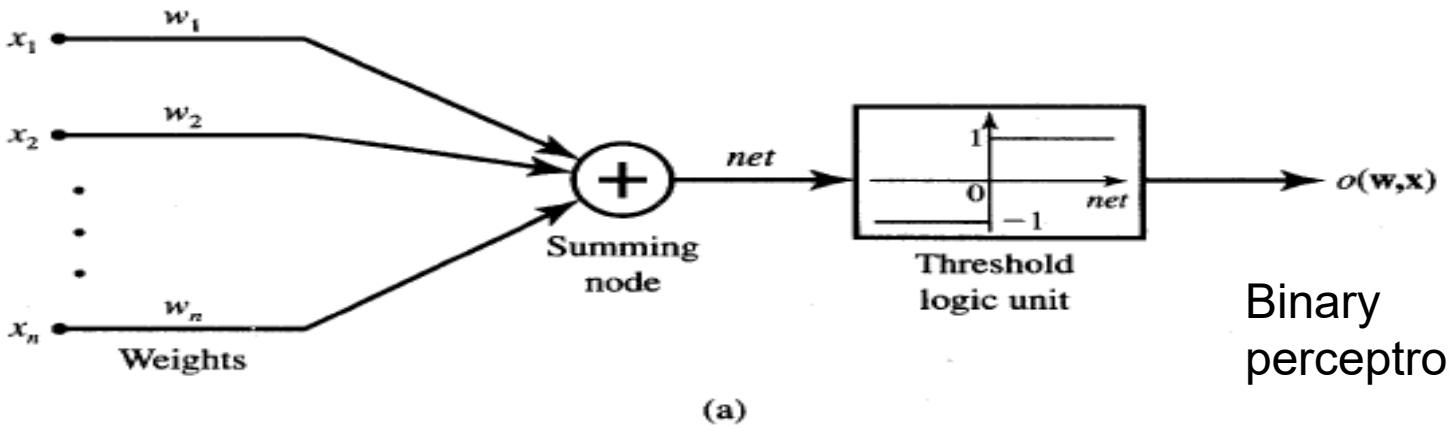
**Unipolar continuous**

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

**Unipolar Binary**

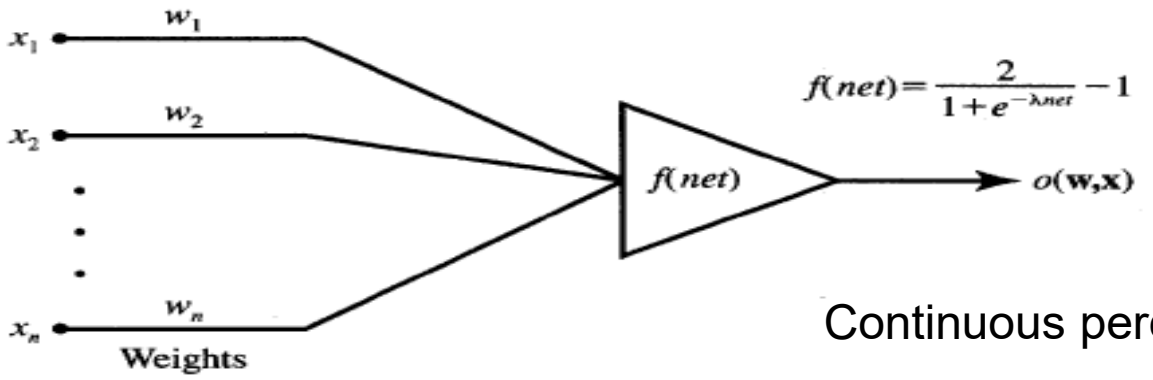
$$f(net) \triangleq \begin{cases} 1, & net > 0 \\ 0, & net < 0 \end{cases}$$

# Common models of neurons




(a)

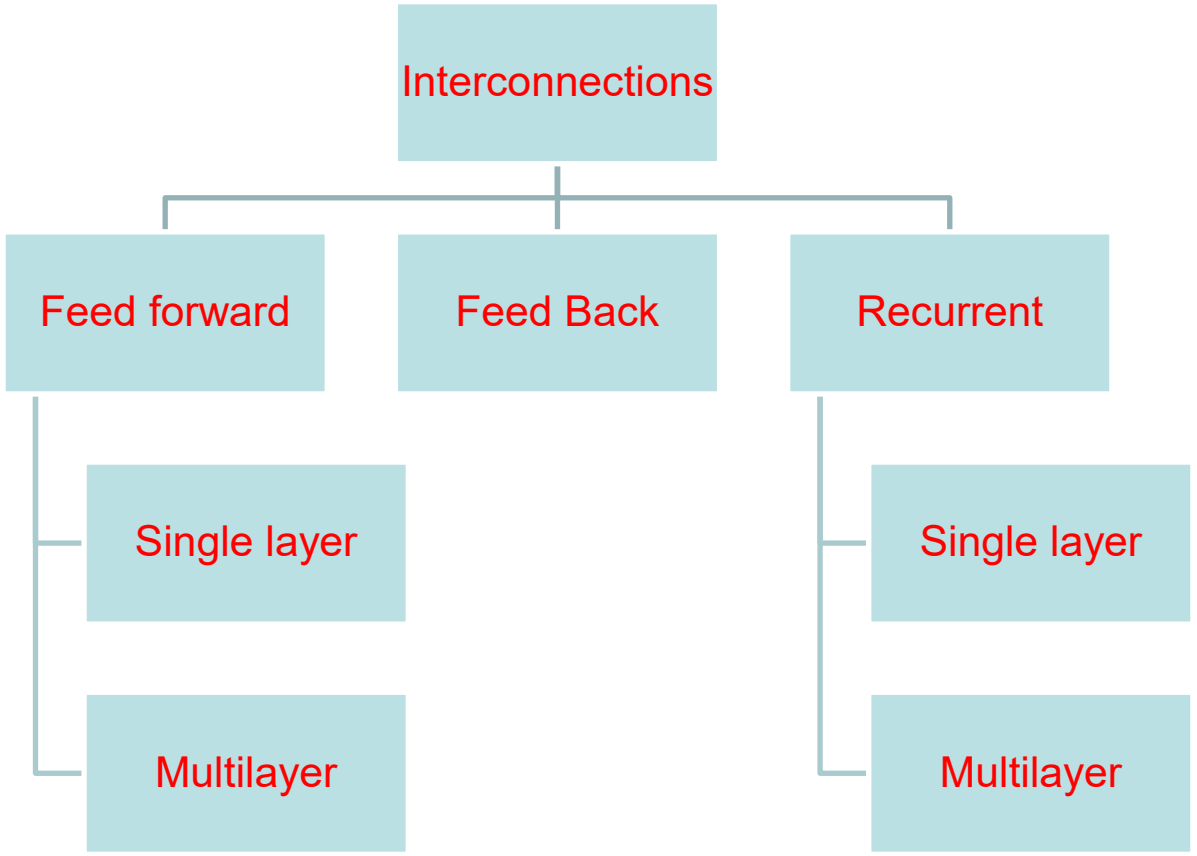
Binary perceptrons



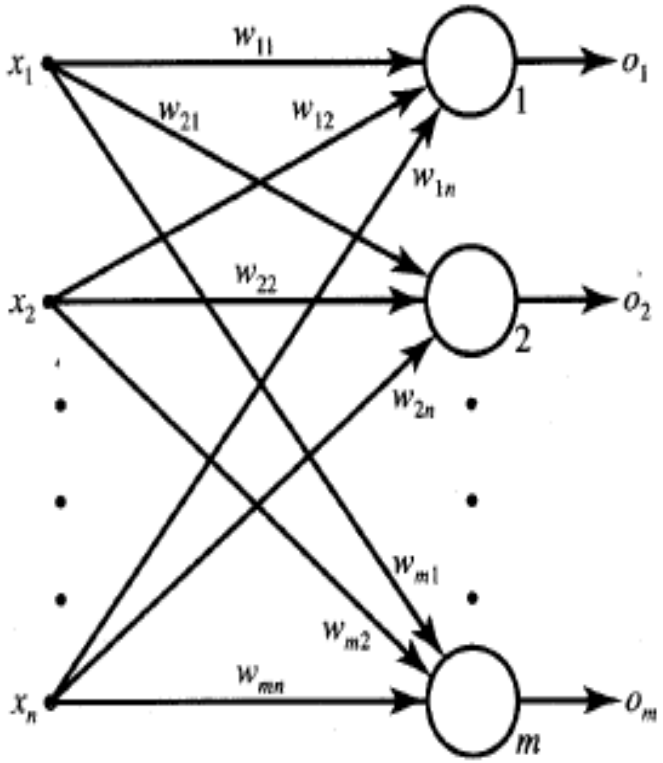

(b)

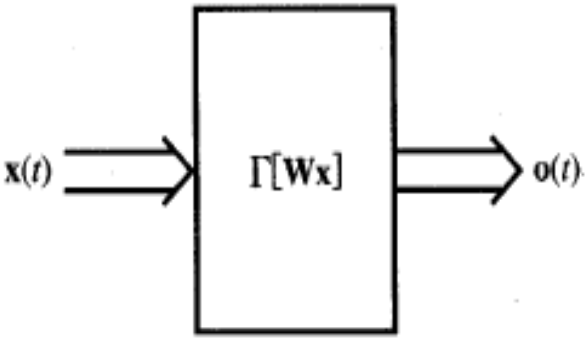
Continuous perceptrons

# 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

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

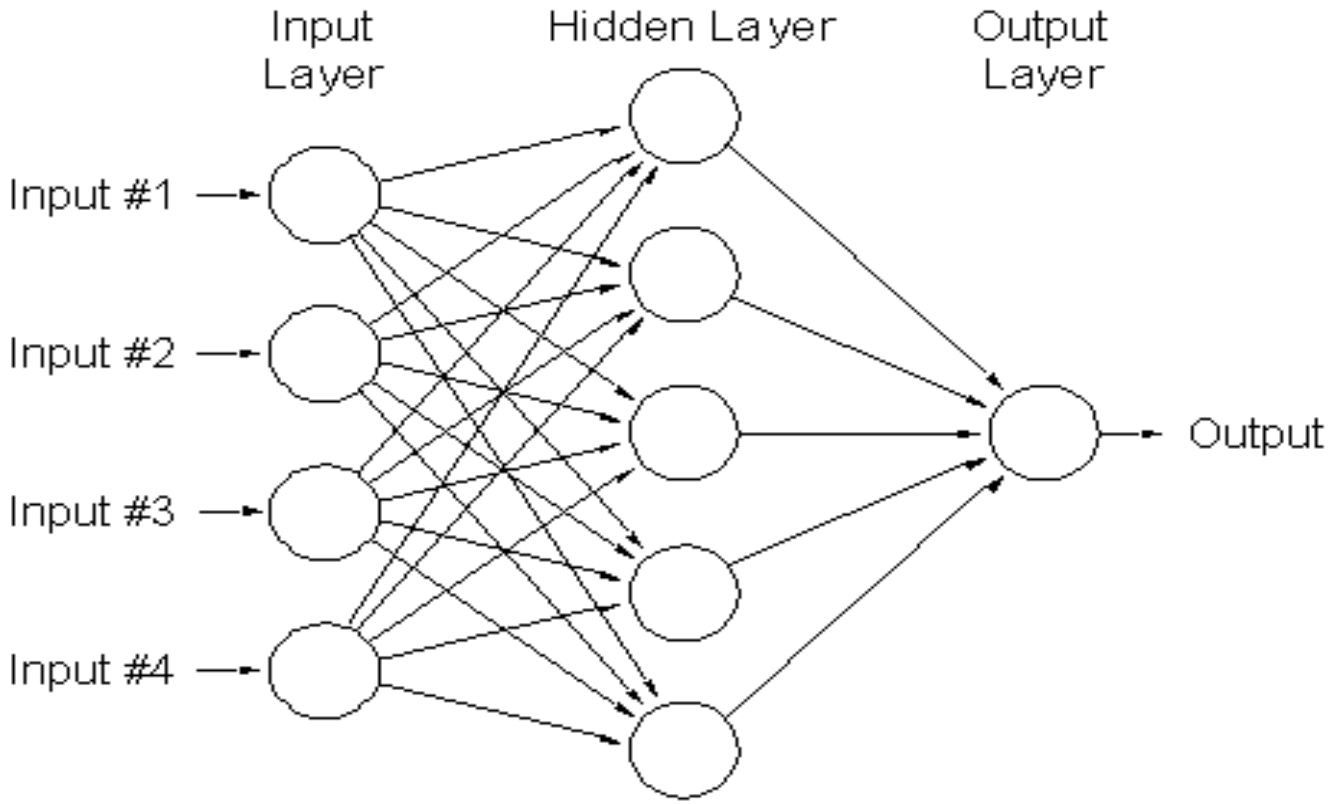
$$o_i = f(\mathbf{w}_i^t \mathbf{x}), \quad \text{for } i = 1, 2, \ldots, m$$

where

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

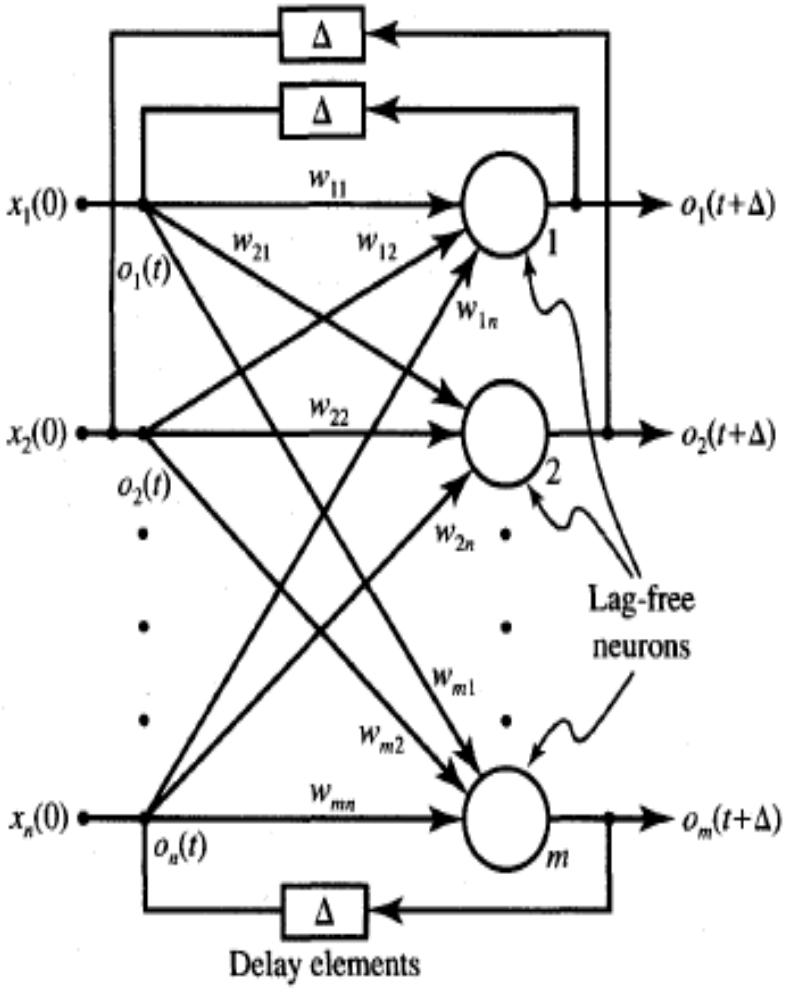
EXAMPLE

# Multilayer feed forward network

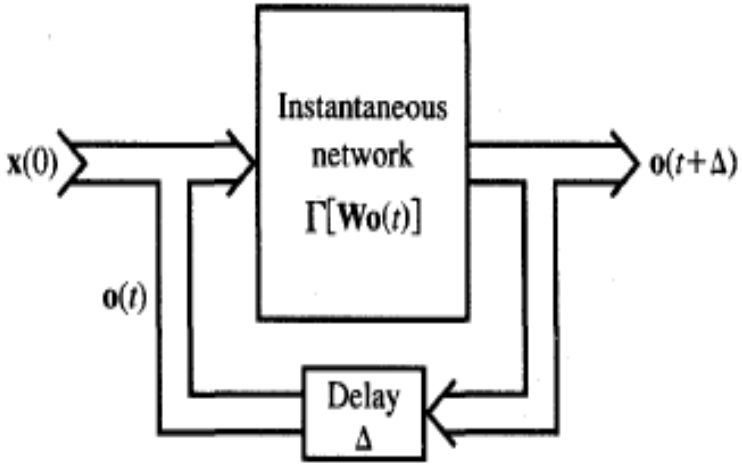



Can be used to solve complicated problems

# Feedback network

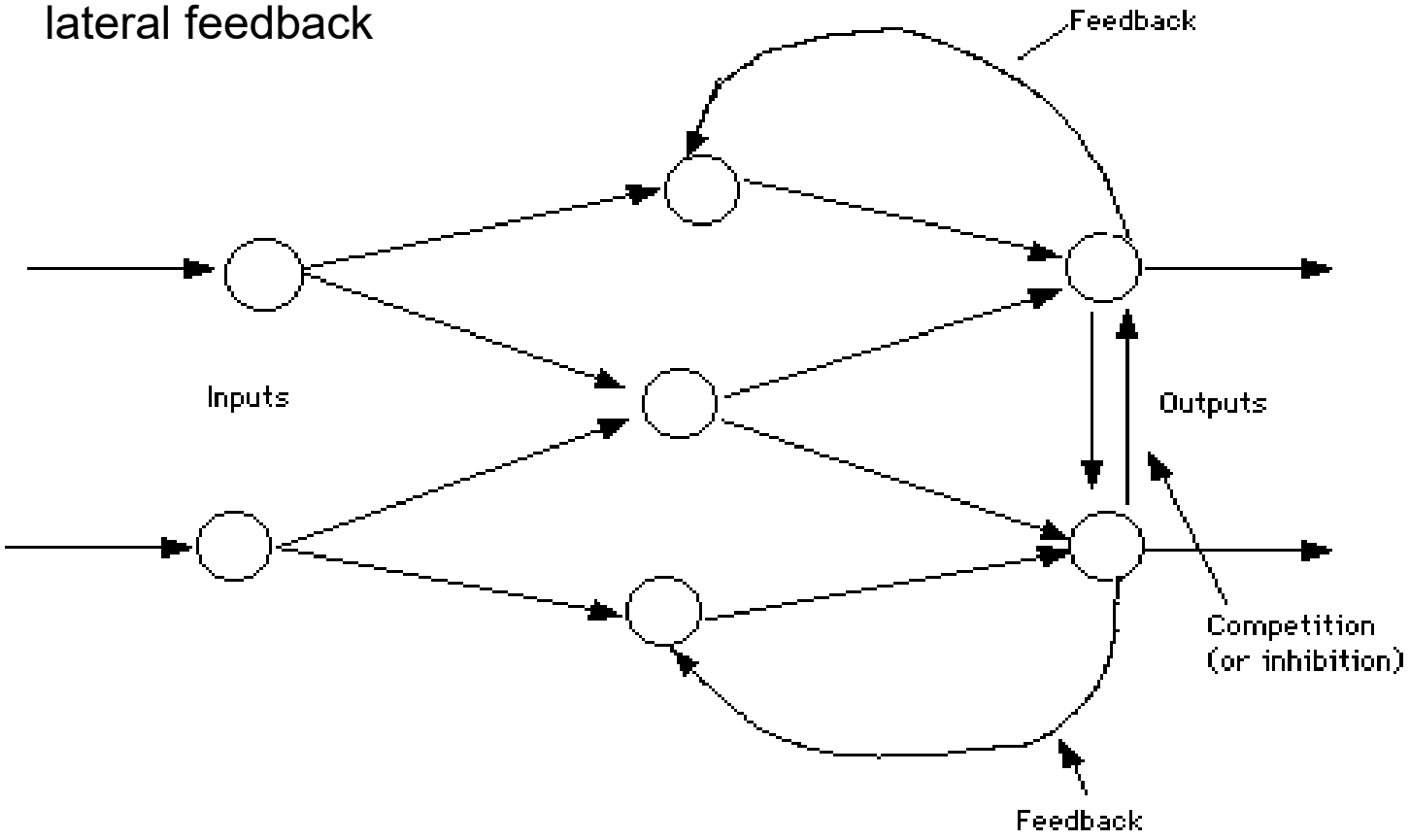



When outputs are directed back as inputs to same or preceding layer nodes it results in the formation of feedback networks

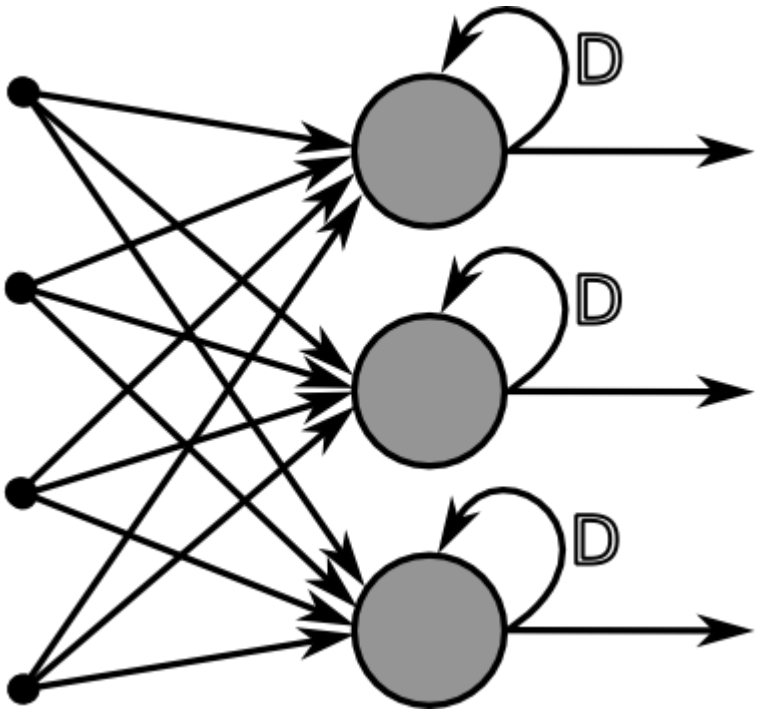


# Lateral feedback

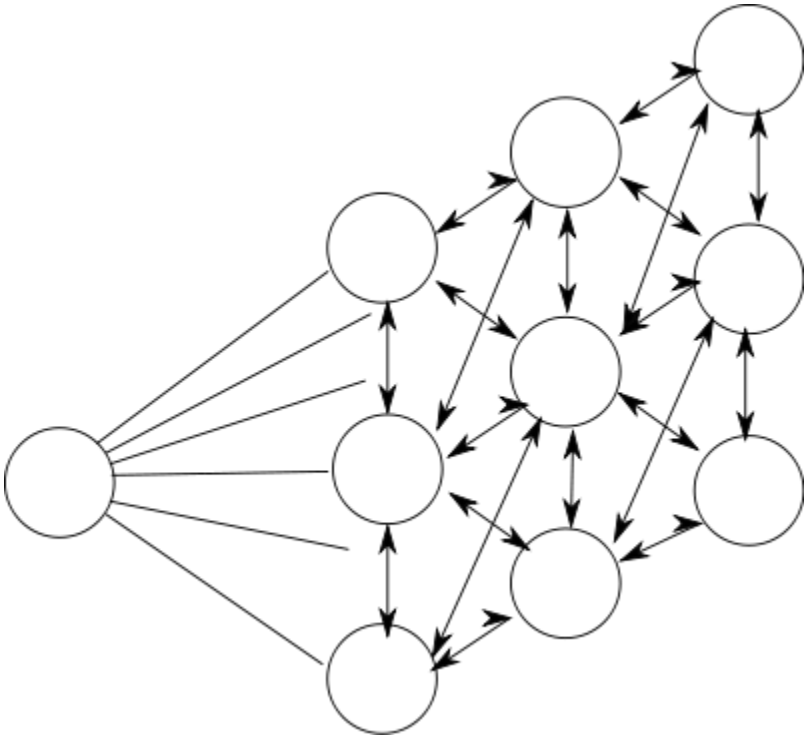
If the feedback of the output of the processing elements is directed back as input to the processing elements in the same layer then it is called lateral feedback



# Single layer Recurrent Networks



# Competitive networks



# 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

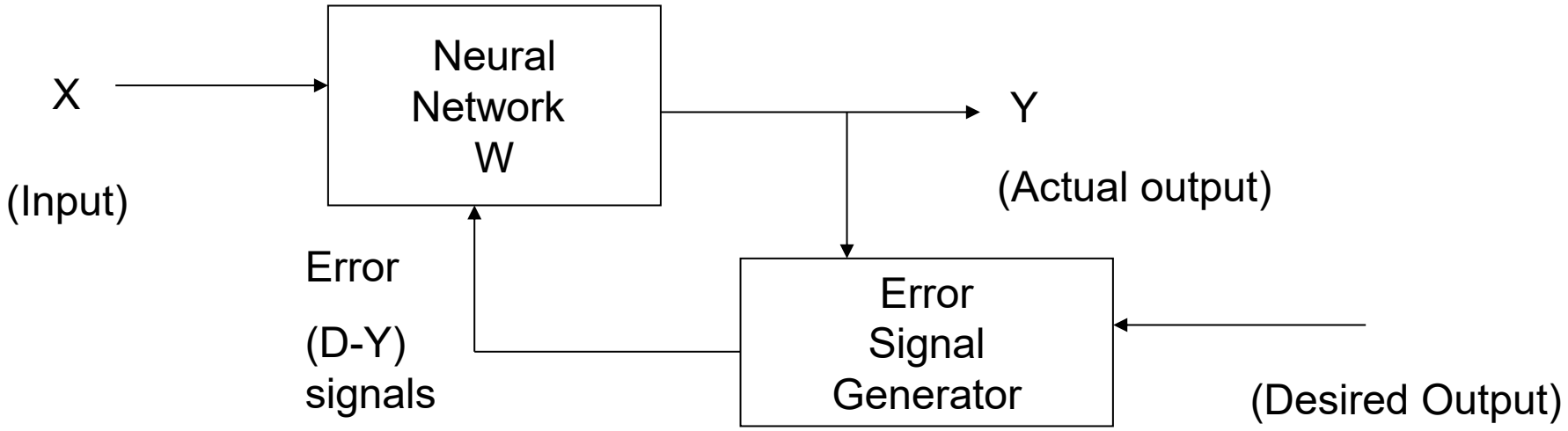
- The process of modifying the weights in the connections between network layers with the objective of achieving the expected output is called training a network.
- This is achieved through
  - Supervised learning
  - Unsupervised learning
  - Reinforcement learning

# Classification of learning

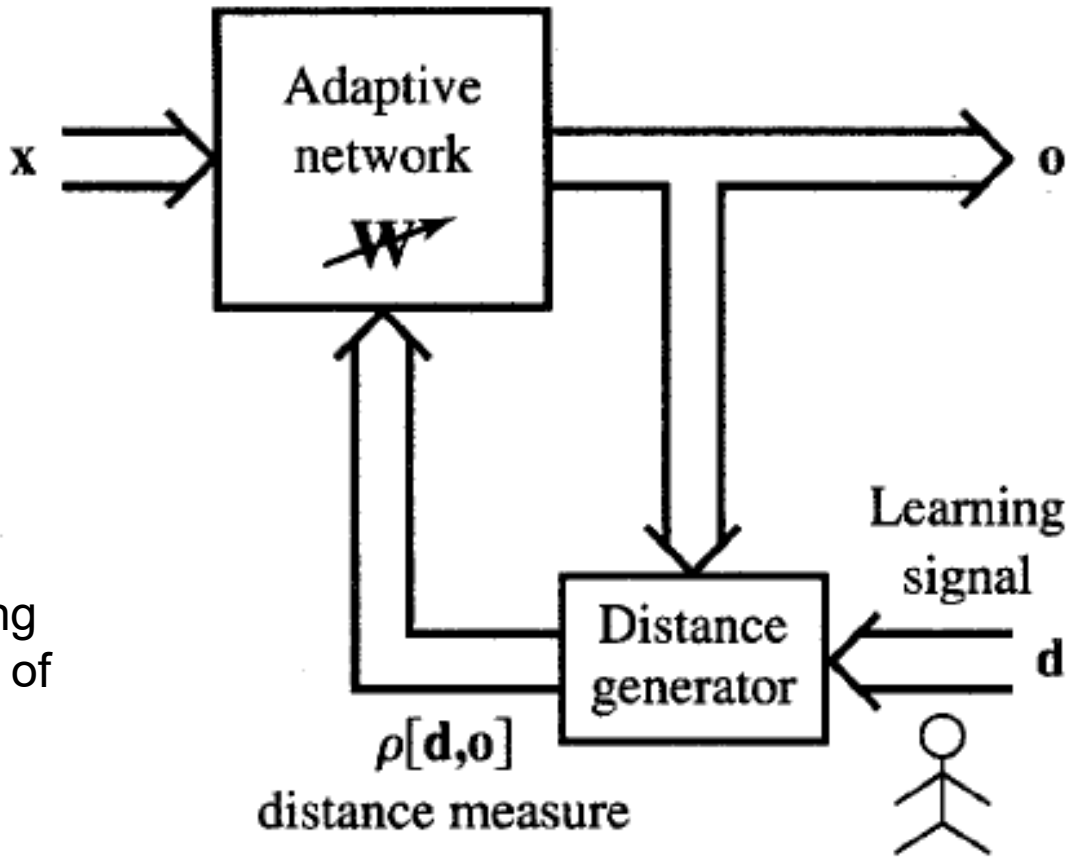
- Supervised learning
- Unsupervised learning
- Reinforcement learning

# Supervised Learning

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



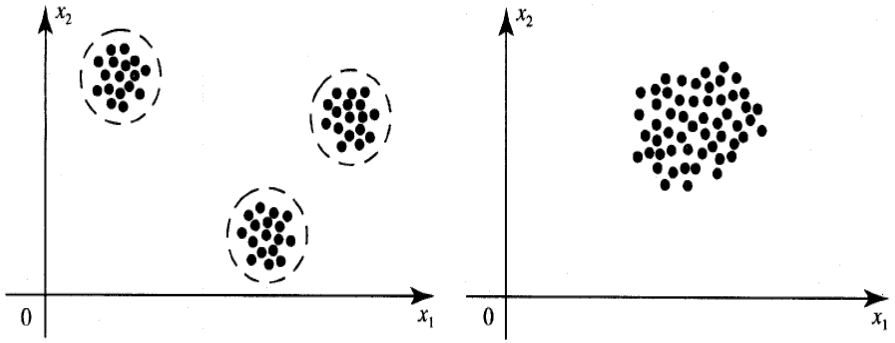
# Supervised learning contd.



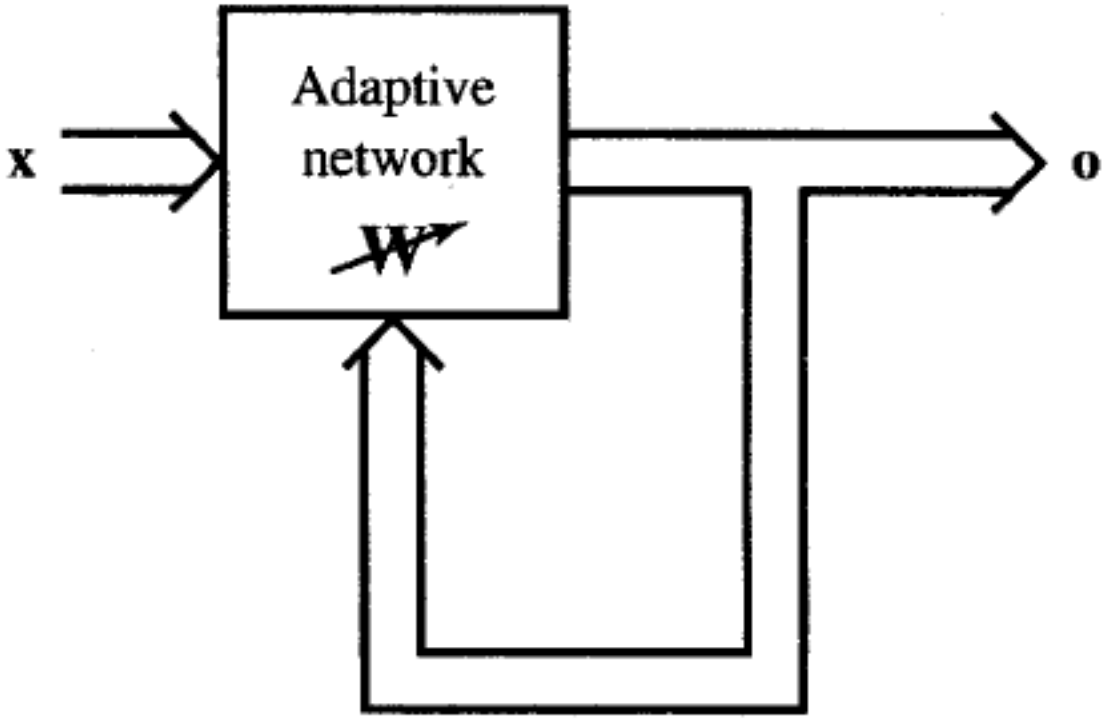

Supervised learning does minimization of error

# 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

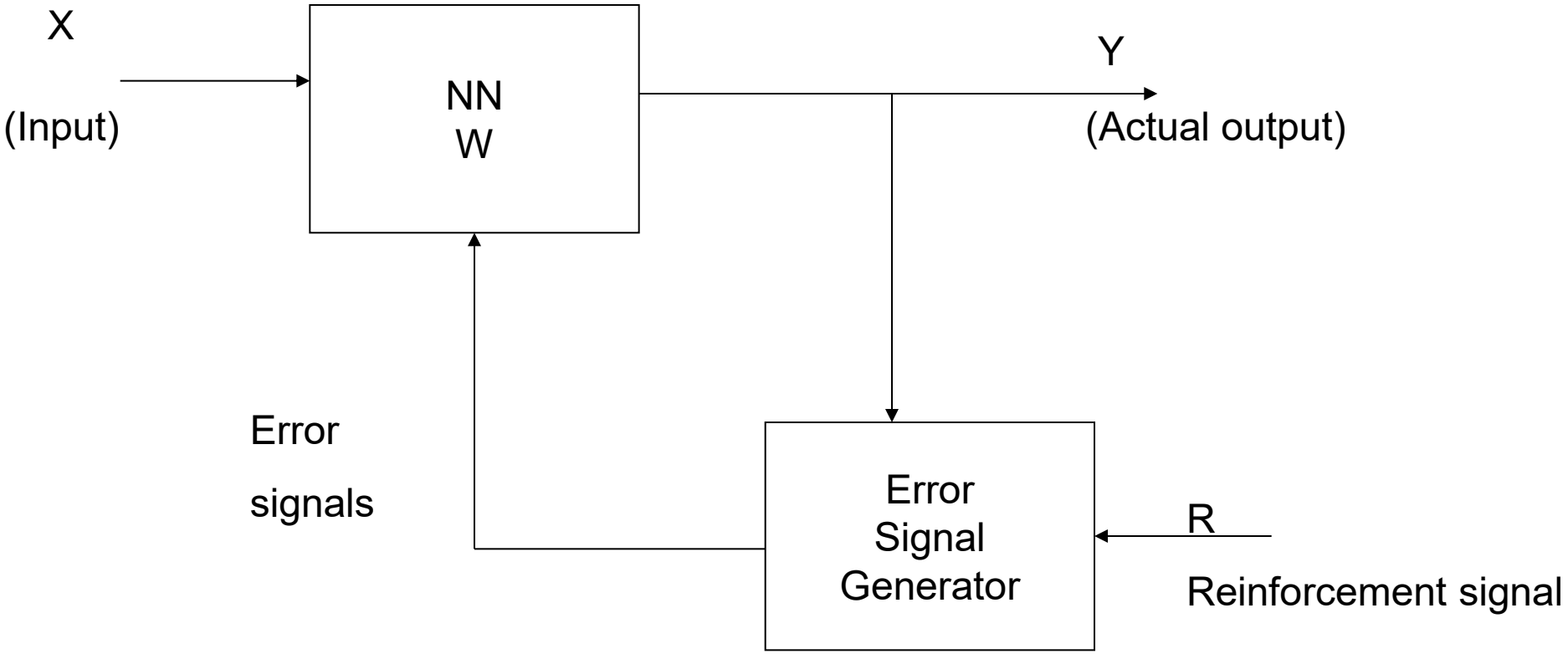
- In unsupervised learning there is no feedback
- Network must discover patterns, regularities, features for the input data over the output
- While doing so the network might change in parameters
- This process is called self-organizing

# Reinforcement Learning

- Reinforcement learning is the training of machine learning models to make a sequence of decisions.
- The agent learns to achieve a goal in an uncertain, potentially complex environment.
- In reinforcement learning, an artificial intelligence faces a game-like situation. The computer employs trial and error to come up with a solution to the problem. To get the machine to do what the programmer wants, the artificial intelligence gets either rewards or penalties for the actions it performs. Its goal is to maximize the total reward.
- Although the designer sets the reward policy–that is, the rules of the game–he gives the model no hints or suggestions for how to solve the game.

- It’s up to the model to figure out how to perform the task to maximize the reward, starting from totally random trials and finishing with sophisticated tactics and superhuman skills.
- By leveraging the power of search and many trials, reinforcement learning is currently the most effective way to hint machine’s creativity.
- In contrast to human beings, artificial intelligence can gather experience from thousands of parallel gameplays if a reinforcement learning algorithm is run on a sufficiently powerful computer infrastructure.

# Reinforcement Learning



# When Reinforcement learning is used?

- A potential application of reinforcement learning in autonomous vehicles is the following interesting case.
- A developer is unable to predict all future road situations, so letting the model train itself with a system of penalties and rewards in a varied environment is possibly the most effective way for the AI to broaden the experience it both has and collects.

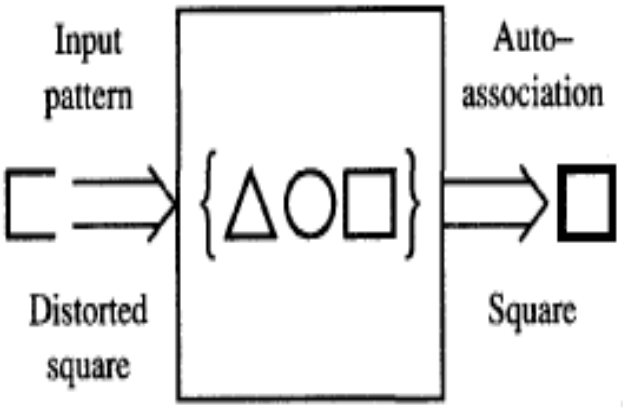
# Some learning algorithms

- 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

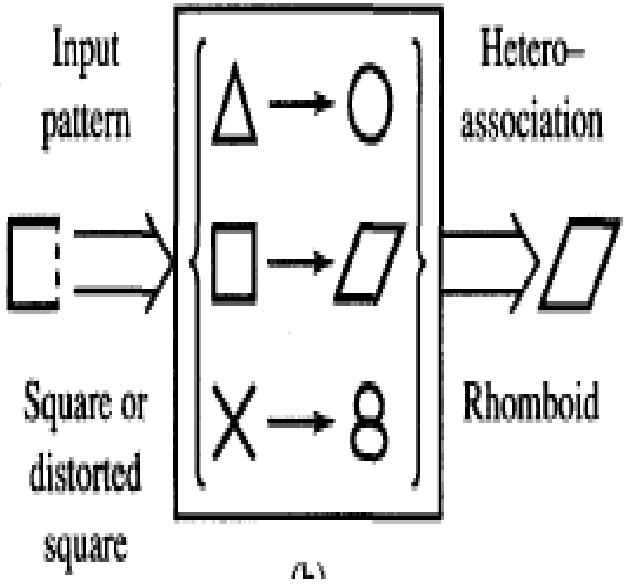
- **Recall:-** processing phase for a NN and its objective is to retrieve the information. The process of computing **o** for a given **x**
- **Basic forms of neural information processing**
  - Auto association
  - Hetero association
  - Classification

# Neural processing (Autoassociation)



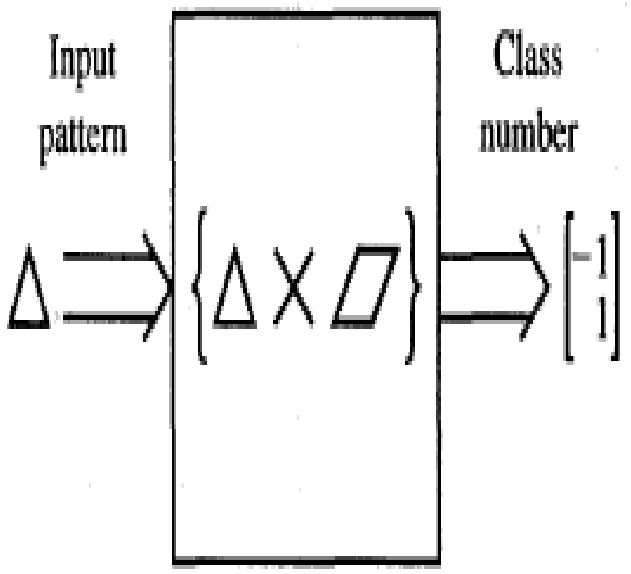

- Set of patterns can be stored in the network
- If a pattern similar to a member of the stored set is presented, an association with the input of closest stored pattern is made

# 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

# Weights

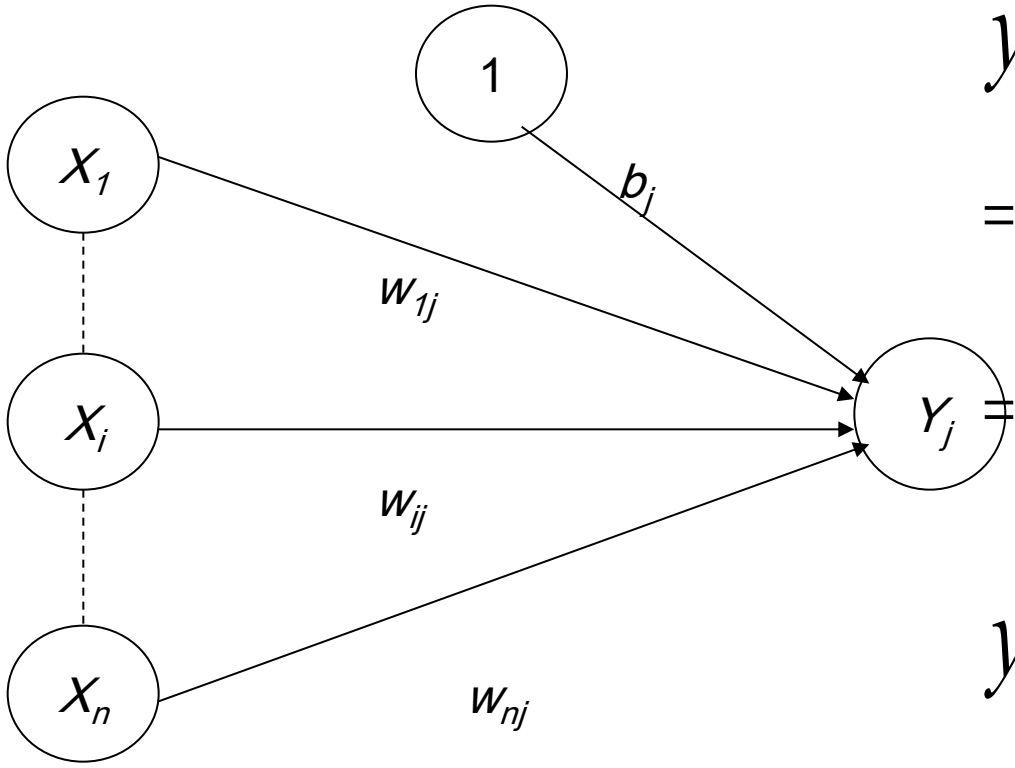
- Each neuron is connected to every other neuron by means of directed links
- Links are associated with weights
- Weights contain information about the input signal and is represented as a matrix
- Weight matrix also called connection matrix

# Weight matrix

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

# Weights contd…

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




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

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

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

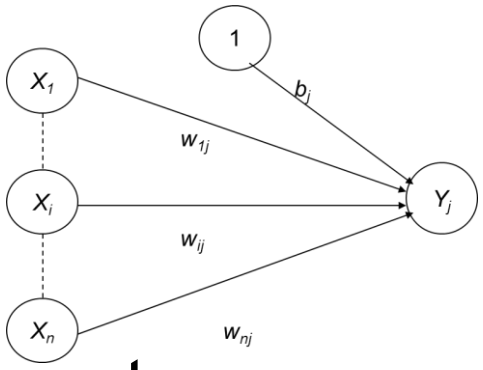
$$y_{inj} = b_j + \sum_{i=1}^{n} x_i w_{ij}$$

# 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.

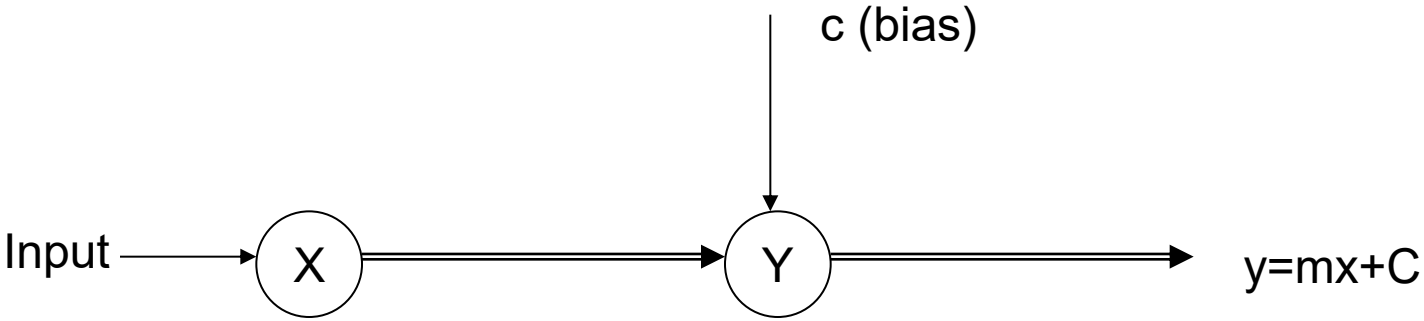
# Bias

- 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

$$f(net) = \begin{Bmatrix} 1.....ifnet \geq \theta \\ -1...ifnet < \theta \end{Bmatrix}$$

## 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 which determines the rate of learning in each time step.

## Momentum factor

– used for convergence when momentum factor is added to weight updating process.

# *Thank you…*

***Any questions??***

يرجى مسح رمز الاستجابة السريعة QR Code لتعبئة نموذج التغذية الراجعة حول المحاضرة. ملاحظاتكم مهمة لتحسين المحاضرات القادمة.

**My google site**