Intelligent Control

ANNs: Concepts, Structures and Developments

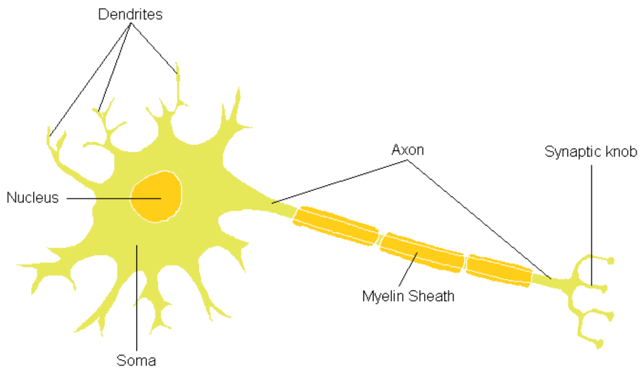
Hassan Bevrani
Professor, University of Kurdistan

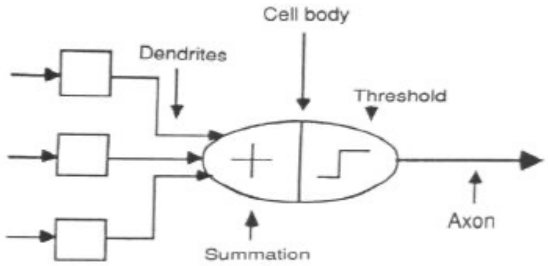
Fall 2023

Contents

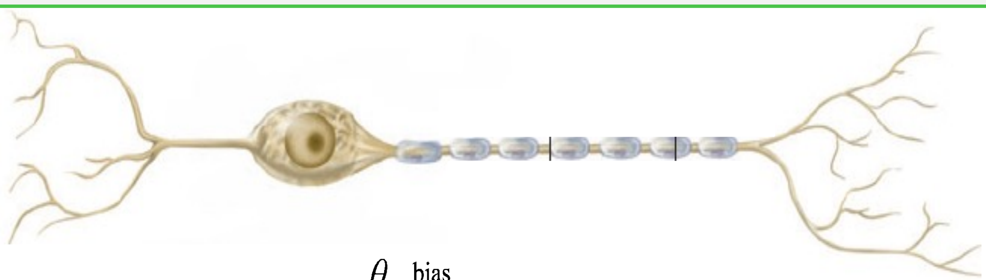
- **ANN Structures**
- **Feedforward and Feedback**
- **Use in Classification & Nonlinear Mapping**
- **Learning Methods**

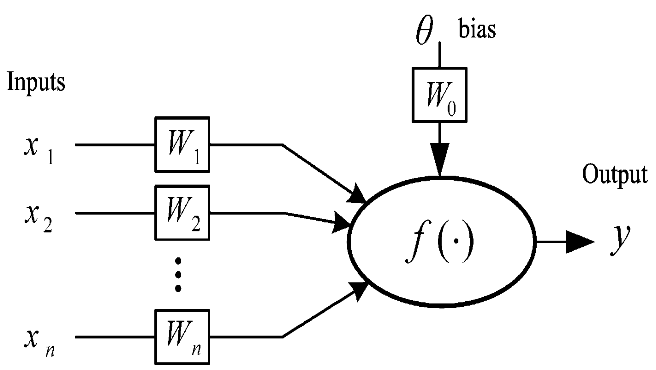
Neuron Model





Neuron Model





$$y(k) = f\left(\sum_{j=1}^{n} W_j x_j(k) + W_0\theta\right)$$

Example 1

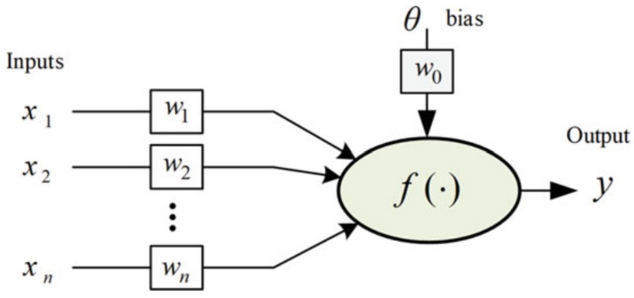
Example 1 Assume a neuron with inputs $\mathbf{x}$, weights $\mathbf{w}$, and a unipolar sigmoid function $f(s) = \frac{1}{(1+e^{-s})}$. Calculate the neuron output for $\mathbf{x} = [0,\ 1,\ 0.5,\ 2]^T$, $\mathbf{w} = [1,\ 1,\ 0.5,\ 1.5]$, and $\theta = 0$.

○ **Solution:**

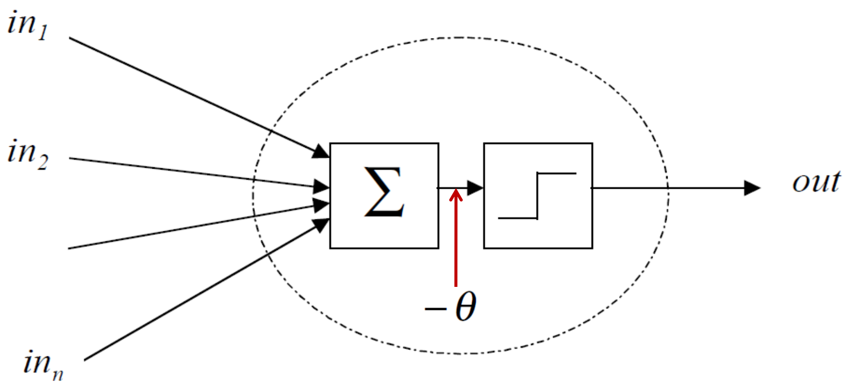
$$y = f\left(\sum_{i=1}^{n} w_i x_i + w_0\theta\right)$$

$$s = \mathbf{wx} = 4.25$$

$$o = f(s) = f(4.25) = 0.9859$$

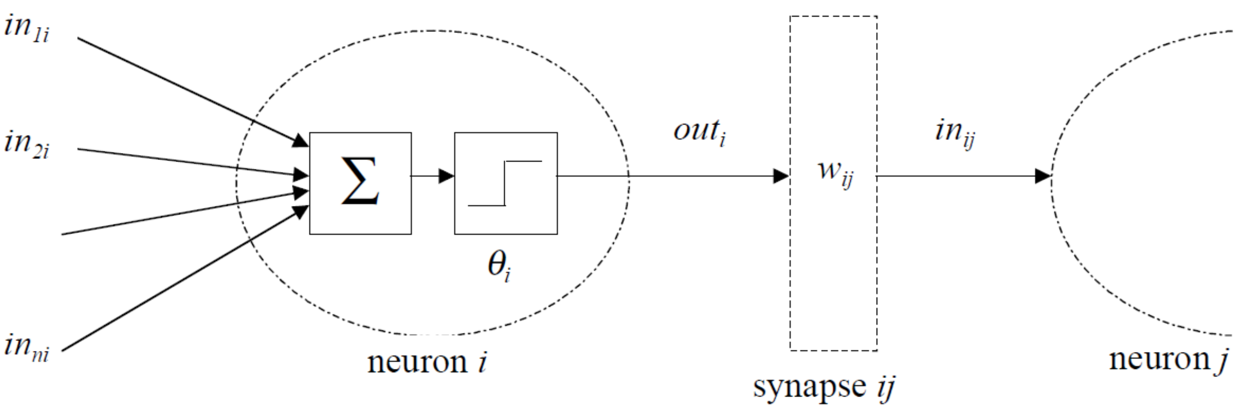


The McCulloch-Pitts Neuron Model



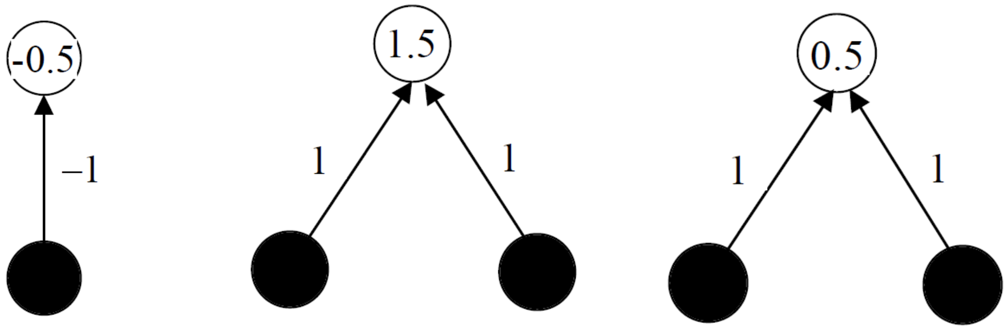
$$out = \mathrm{sgn}\left(\sum_{i=1}^{n} in_i - \theta\right) \Rightarrow \begin{cases} out = 1 & \text{if } \sum_{k=1}^{n} in_k \geq \theta \\ out = 0 & \text{if } \sum_{k=1}^{n} in_k < \theta \end{cases}$$

Network of the McCulloch-Pitts Neurons



$$in_{ki} = out_k w_{ki} \qquad out_i = \mathrm{sgn}\left(\sum_{k=1}^{n} in_{ki} - \theta_i\right) \qquad in_{ij} = out_i w_{ij}$$

Building Logic Gates



NOT

in	out
0	1
1	0

AND

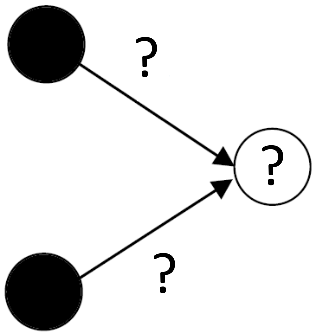
in_1	in_2	out
0	0	0
0	1	0
1	0	0
1	1	1

OR

in_1	in_2	out
0	0	0
0	1	1
1	0	1
1	1	1

Analytical Solution

Ex.: AND



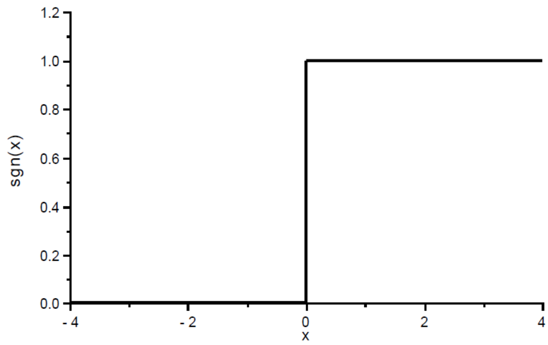
$$out = \operatorname{sgn}\left(w_1 in_1 + w_2 in_2 - \theta\right)$$

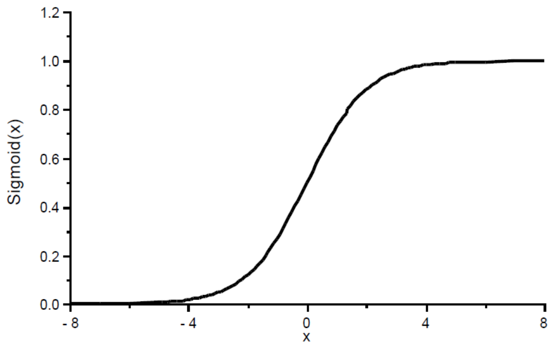
in_1	in_2	out
0	0	0
0	1	0
1	0	0
1	1	1

$$w_1 0 + w_2 0 - \theta < 0$$
$$w_1 0 + w_2 1 - \theta < 0$$
$$w_1 1 + w_2 0 - \theta < 0$$
$$w_1 1 + w_2 1 - \theta \geq 0$$

$$\theta > 0$$
$$w_2 < \theta$$
$$w_1 < \theta$$
$$w_1 + w_2 \geq \theta$$

Useful Functions

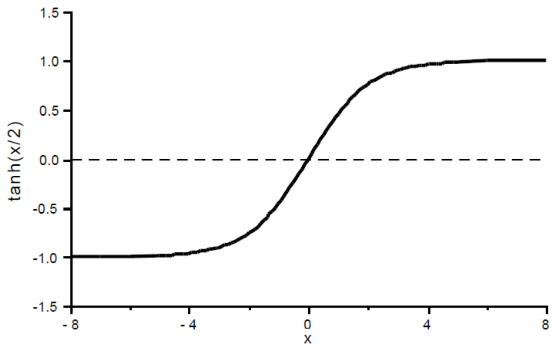




$$\operatorname{sgn}(x) = \begin{cases} 1 & \text{if } x \geq 0 \\ 0 & \text{if } x < 0 \end{cases}$$

$$\operatorname{Sigmoid}(x) = \frac{1}{1 + e^{-x}}$$

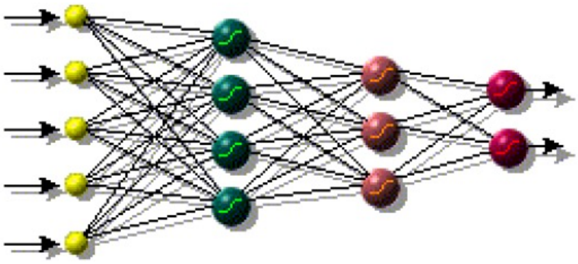
Useful Functions



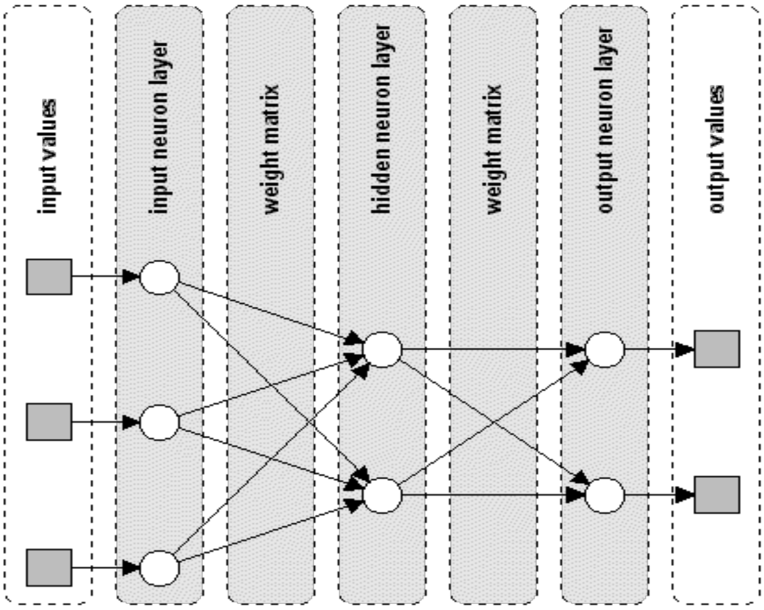
$$\tanh\left(\tfrac{x}{2}\right) = \frac{1-e^{-x}}{1+e^{-x}}$$

$$f(x) = \begin{cases} 1 & \text{if } x \geq 0.5 \\ x+0.5 & \text{if } -0.5 \leq x \leq 0.5 \\ 0 & \text{if } x \leq 0.5 \end{cases}$$

ANN

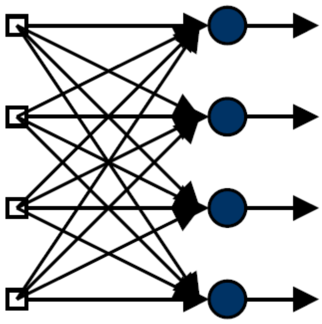


3-Layer ANN

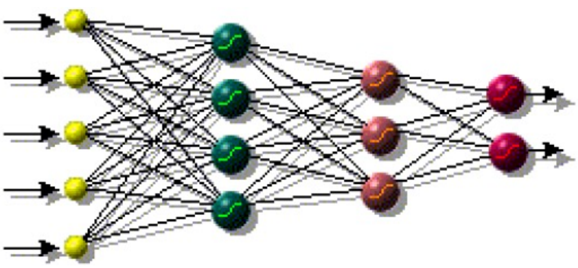


ANN Architectures

Single Layer



Multi-Layer



ANNs Characteristics

- Parallel processing,
- Simple learning process,
- High speed in learning process,
- Flexibility in structure,
- Nonlinearity mapping ability,
- Linear and nonlinearity function(s).

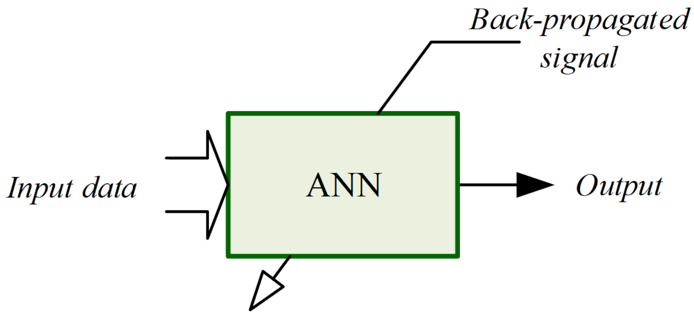
ANNs: Differences

- Network topology (architecture),
- Neuron types,
- Learning rule(s),
- Applied algorithm.

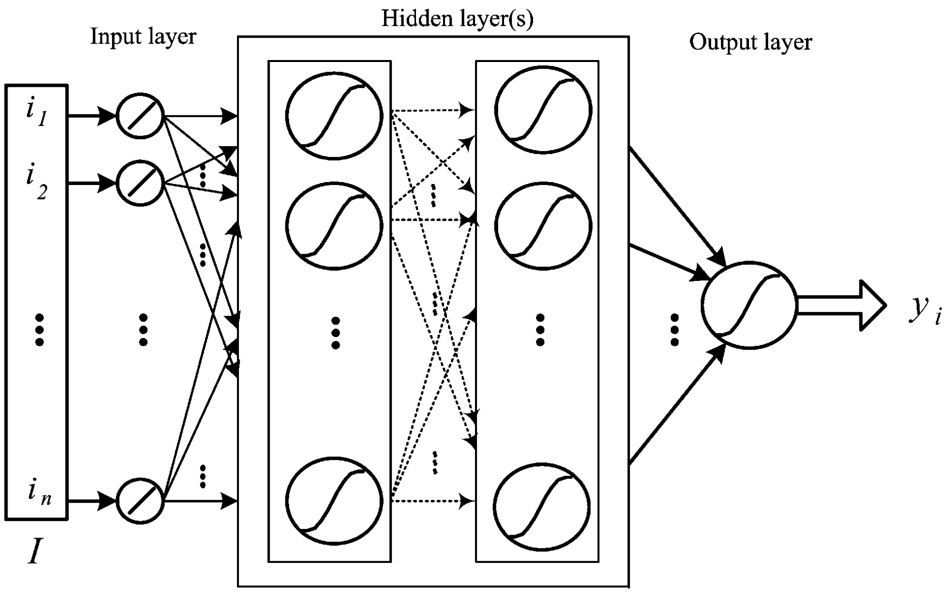
Common ANN Configurations

Feedforward

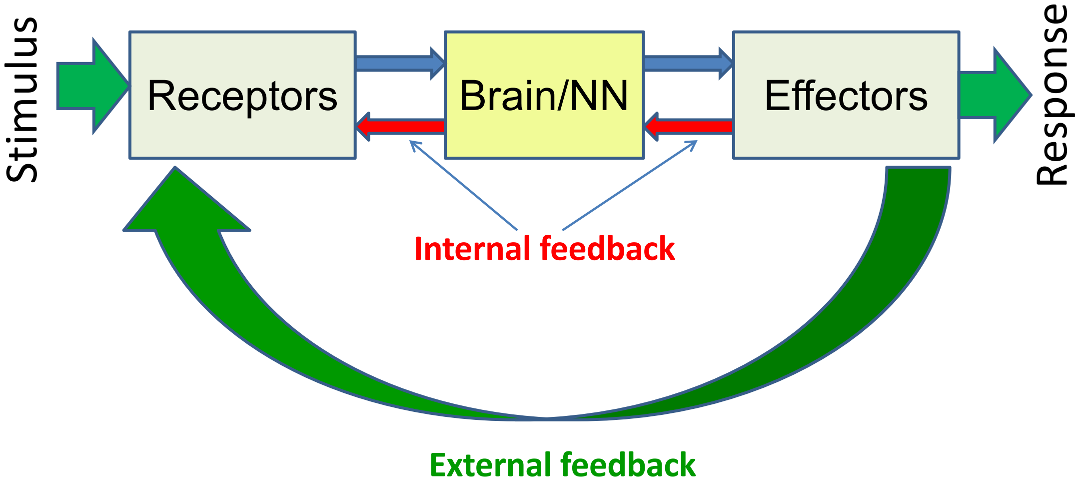
Feedback



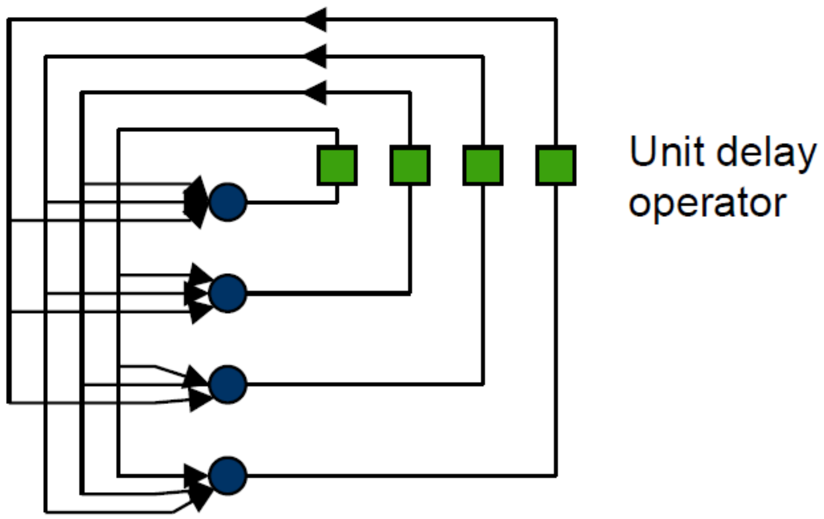
Feedforward ANN



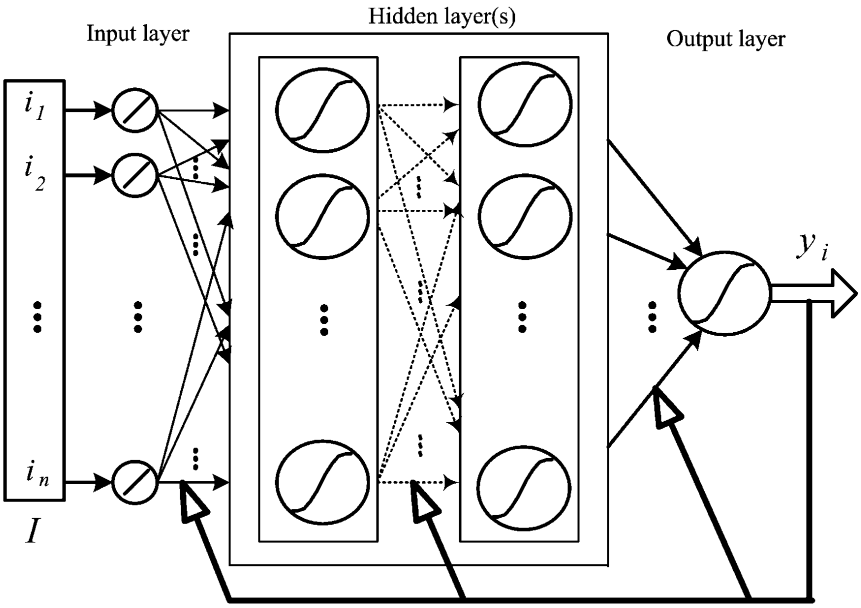
Feedbacks



Internal Feedback

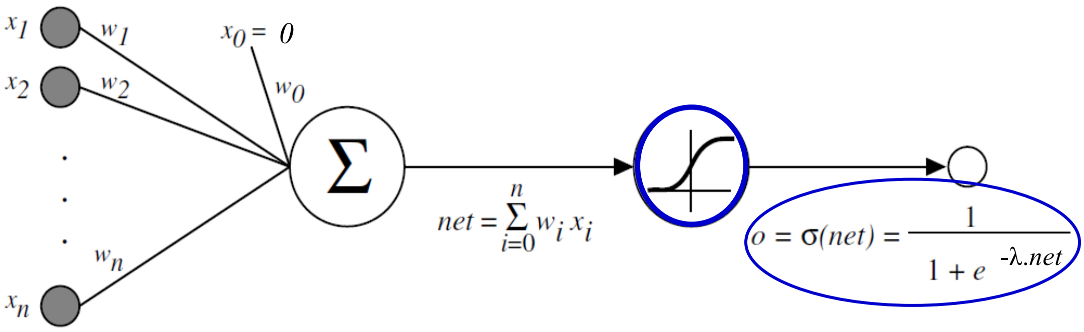


External Feedback



Feedforward

Unipolar Sigmoid Function (USF)

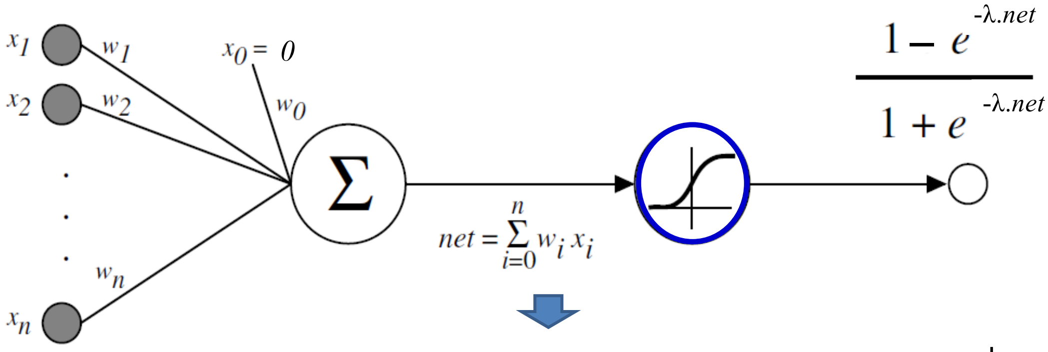


$X^T=[0, 1, 0.5, 2]; W^T=[1, 1, 0.5, 1.5]; \lambda=1$

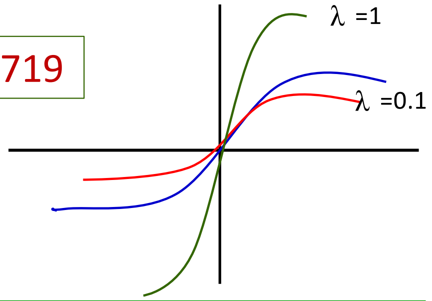
$net=W^T X=4.25 \Rightarrow O=0.9859$

Feedforward

Bipolar Sigmoid Function (BSF)



$net = W^T X = 4.25 \Rightarrow O = 0.9719$

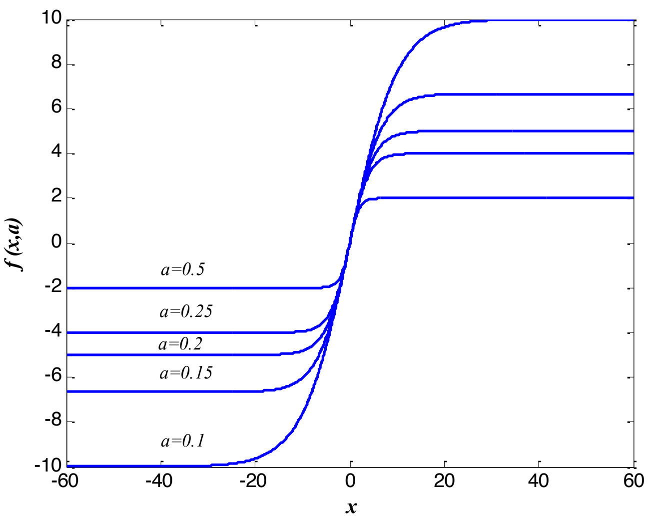


Q: Repeat Example for λ=0.5 ⇨ O=0.7866

Flexible Neurons

$$f(x,a) = \frac{1 - e^{-2xa}}{a(1 + e^{-2xa})}$$

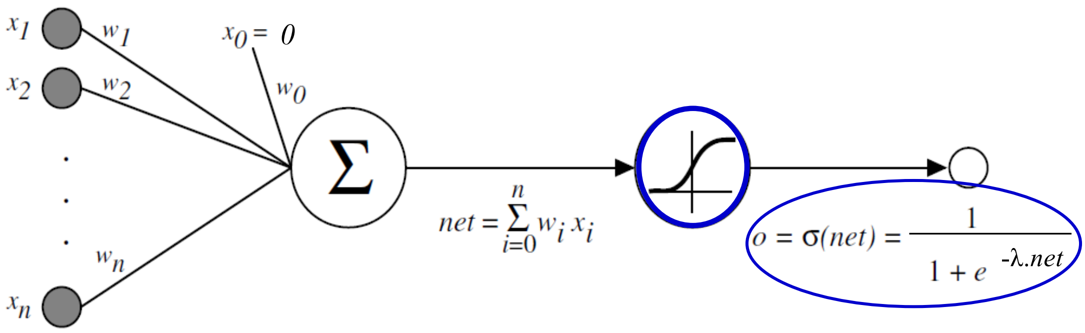
$$\lim_{a \to 0} f(x,a) = x$$



The flexibility of ANNs can be increased using flexible sigmoid functions: Flexible Neural Network (FNN)

Example 2 (Inhibitory Weight)

Unipolar Sigmoid Function (USF)



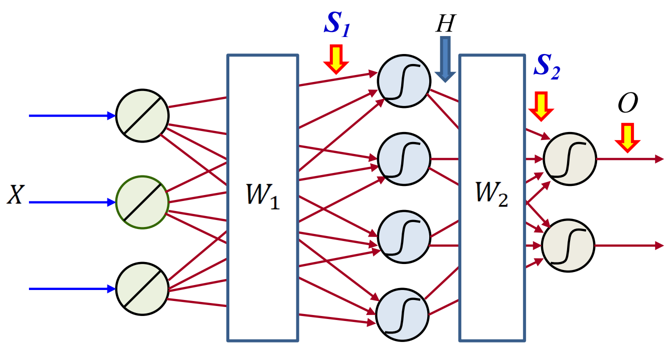
n=3, X^T=[0, 1, 0.5, 2]; W^T=[1, **-1**, 0.5, 1.5]; λ=1

$net = W^T X = 3.25$;
USF: O=0.963;
BSF: O=0.925;

3-Layer ANN (Feedforward Mechanism): Example 3

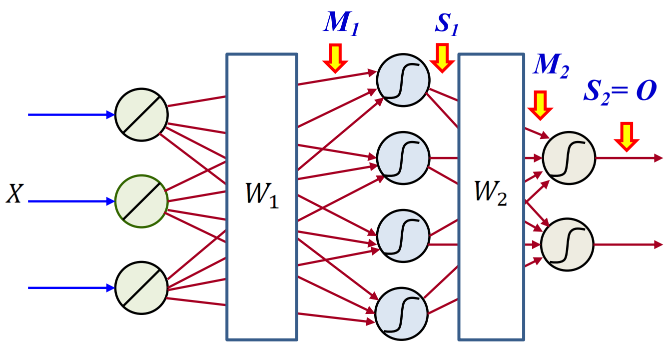
;

$$X = \begin{bmatrix} 0.5 \\ -2 \\ 1 \end{bmatrix}; W_1^T = \begin{bmatrix} 1 & 0 & 0.5 \\ 0.2 & 1 & -1 \\ 2 & 0 & -3 \\ 1 & 2 & 0 \end{bmatrix}; W_2^T = \begin{bmatrix} 0.5 & 1 \\ 1 & 2 \\ 0 & 3 \\ 1 & 0.5 \end{bmatrix}; \text{ USF: } \lambda=1$$



MATLAB Program (for Example 3)

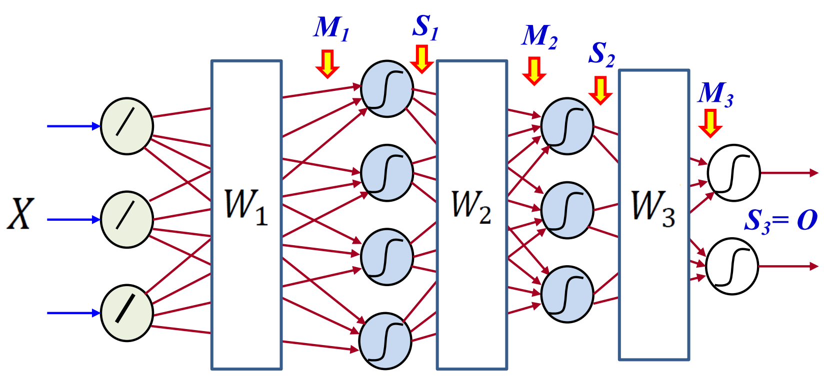
```
%% Example 3 Solution
% Initial Data
W1=[1 0.2 2 1; 0 1 0 2; 0.5 -1 -3 0.5];
X=[0.5; -2; 1];
W2=[0.5 1; 1 2; 0 3; 1 0.5];
Landa=1;
% 2-Layer input
M1=W1'*X
% 2-Layer Output
S1=1./(1.+(exp(1)).^(-Landa*M1)) %USF
% 3-Layer input
M2=W2'*S1
% 3-Layer Output
O=1./(1.+(exp(1)).^(-Landa*M2)) %USF
```




```
M1 =            S1 =          M2 =          O =
   1.0000          0.7311        0.4651        0.6142
  -2.9000          0.0522        1.2167        0.7715
  -2.0000          0.1192
  -3.0000          0.0474
```

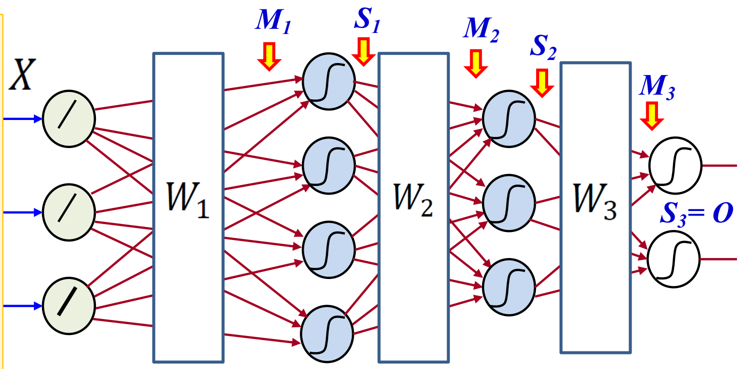

4-Layer ANN (Feedforward Mechanism): Example 4

$$X = \begin{bmatrix} 0.5 \\ -2 \\ 1 \end{bmatrix}; W_1{}^T = \begin{bmatrix} 1 & 0 & 0.5 \\ 0.2 & 1 & -1 \\ 2 & 0 & -3 \\ 1 & 2 & 0 \end{bmatrix}; W_3{}^T = \begin{bmatrix} 0.5 & 1 & 0 \\ 1 & 2 & 3 \end{bmatrix}; \; W_2{}^T = \begin{bmatrix} -1 & 0.5 & 2 & 1 \\ 3 & -2 & -1 & -0.5 \\ -1 & 1 & 2 & 2 \end{bmatrix}$$



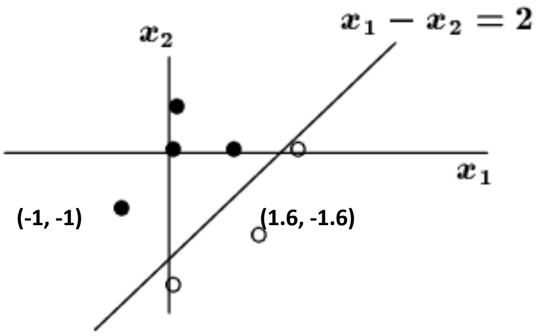
MATLAB Program (for Example 4)

```
%% Example 4 Solution
% Initial Data
X=[0.5; -2; 1];
W1=[1 0.2 2 1; 0 1 0 2; 0.5 -1 -3 0.5];
W2=[-1 3 -1; 0.5 -2 1; 2 -1 2; 1 -0.5 2];
W3=[0.5 1; 1 2; 0 3];
Landa=1;
% 2-Layer input
M1=W1'*X;
% 2-Layer Output
S1=1./(1.+(exp(1)).^(-Landa*M1)); %USF
% 3-Layer input
M2=W2'*S1
% 3-Layer Output
S2=1./(1.+(exp(1)).^(-Landa*M2)) %USF
% 4-Layer input
M3=W3'*S2
% 4-Layer Output
S3=1./(1.+(exp(1)).^(-Landa*M3)) %USF
```



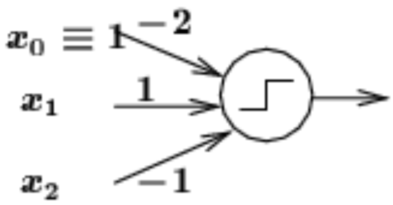

```
M2 =            S2 =            M3 =            S3 =
 -0.4192         0.3967          1.0734          0.7452
  1.9460         0.8750          3.3900          0.9674
 -0.3456         0.4144
```


Application in Classification



A set of (2D) patterns (x_1, x_2) of two classes is linearly separable if there exists a line on the (x_1, x_2) plane

$w_0 + w_1 x_1 + w_2 x_2 = 0$



Application in Classification

– For n dimensional patterns $(x_1,\ldots, x_n)$
  - Hyperplane $w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_n x_n = 0$ dividing the space into two regions

– The weights can be obtained from a set of sample patterns If the problem is linearly separable

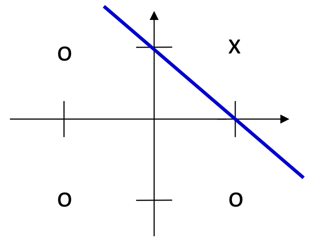
Application in Classification

Linearly Separable Classes

- Logical **AND** function

patterns (bipolar) decision boundary

x1	x2	output	
-1	-1	-1	w1 = 1
-1	1	-1	w2 = 1
1	-1	-1	w0 = -1
1	1	1	-1 + x1 + x2 = 0

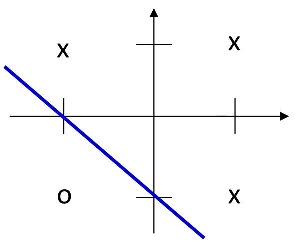


x: class I (output = 1)
o: class II (output = -1)

- Logical **OR** function

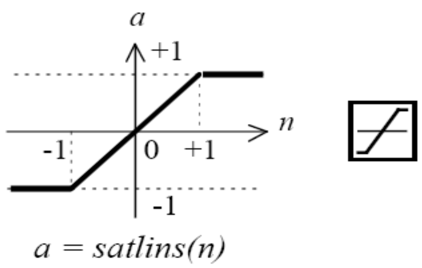
patterns (bipolar) decision boundary

x1	x2	output	
-1	-1	-1	w1 = 1
-1	1	1	w2 = 1
1	-1	1	w0 = 1
1	1	1	1 + x1 + x2 = 0

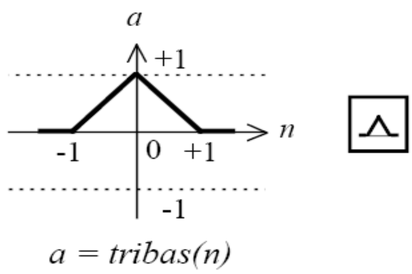


x: class I (output = 1)
o: class II (output = -1)

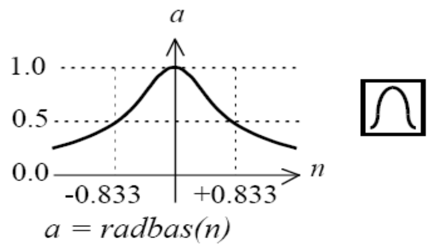
More Neuron Transfer Functions



Satlins Transfer Function



Triangular Basis Function



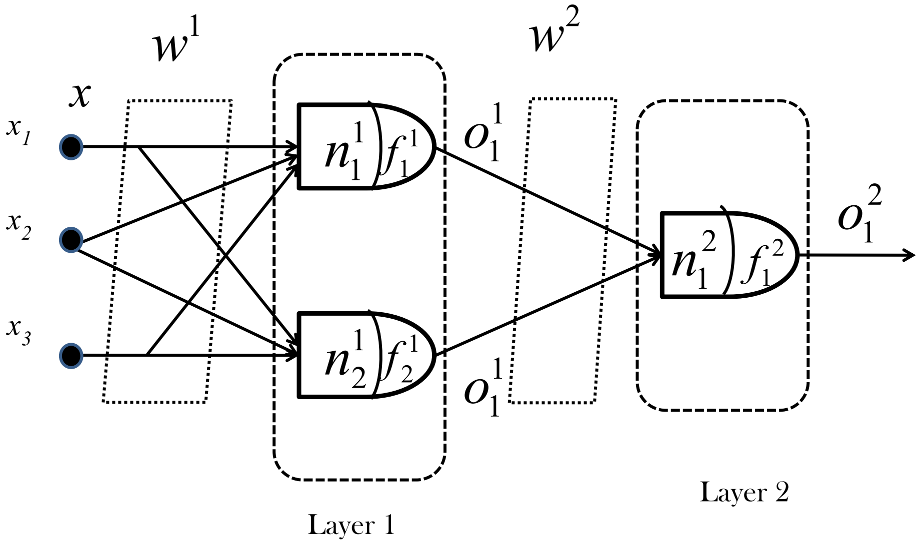
Radial Basis Function

Satlins(*n*) = -1, if n <= -1
= n, if -1 <= n <= 1
= 1, if 1 <= n

Tribas(*n*) = 1 - abs(*n*), if -1 <= n <= 1
= 0, otherwise

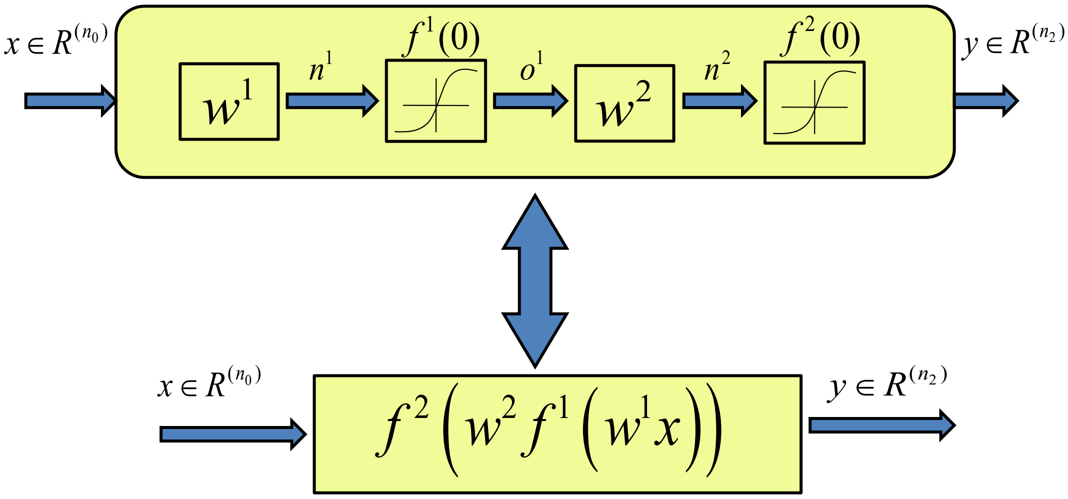
Radbas(*n*) = *exp*(-*n*^2)

Nonlinear Mapping



ANN with two activation layers

Nonlinear Mapping



Two-layered NN as nonlinear mapping for input ($x \in R^{(n_0)}$) to output ($y \in R^{(n_2)}$)

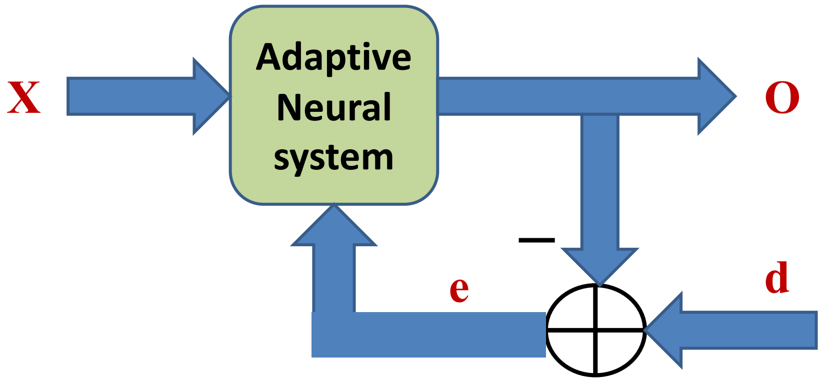
Learning Methods

i) Supervised learning

ii) Unsupervised learning

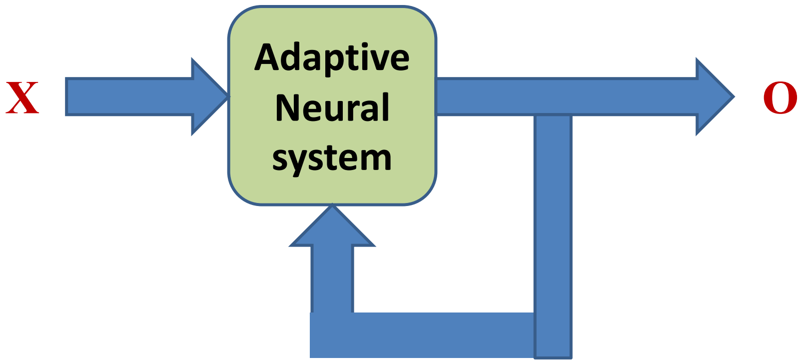
iii) Reinforcement learning

Supervised Learning



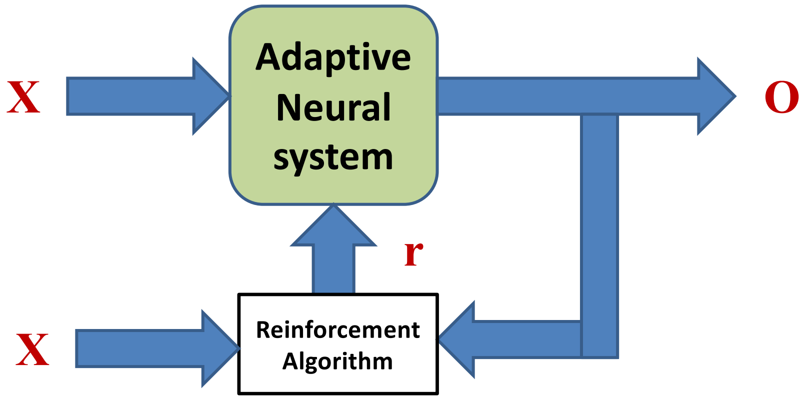
Supervised learning algorithm uses a supervisor that knows the desired outcomes and tunes the weights accordingly.

Unsupervised Learning



Unsupervised learning algorithm uses local data, instead of supervisor, according to emergent collective properties.

Reinforcement Learning



Reinforcement learning algorithm uses some reinforcement signal, instead of output error of that neuron, to tune the weights.

Homework 1

○ **Like Slide 8 of this lecture:**

1. Find analytical solutions for NOT, OR, and XOR logical gates.
2. Design a simple ANN for realizing a Majority logical function with 3 input.

Deadline: The day before next Meeting

Please only use this email address: bevranih18@gmail.com

Thank You!