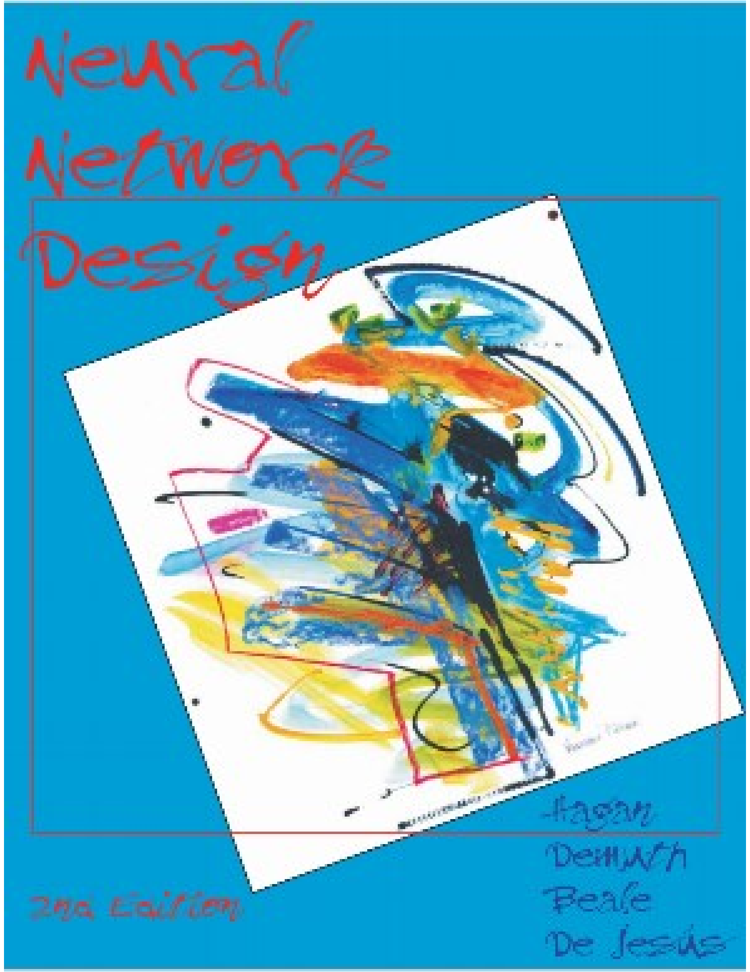
计算思维

课程四

Textbook: Neural Network Design



Neural Network Design
2nd Edition

Martin T. Hagan
Oklahoma State University
Stillwater, Oklahoma

Howard B. Demuth
University of Colorado
Boulder, Colorado

Mark Hudson Beale
MHB Inc.
Hayden, Idaho

Orlando De Jesús
Consultant
Frisco, Texas

Textbook: 机器学习

IEEE Computational Intelligence Society

Definition of Computational Intelligence

Any biologically, naturally, and linguistically motivated computational paradigms include, but not limited to,

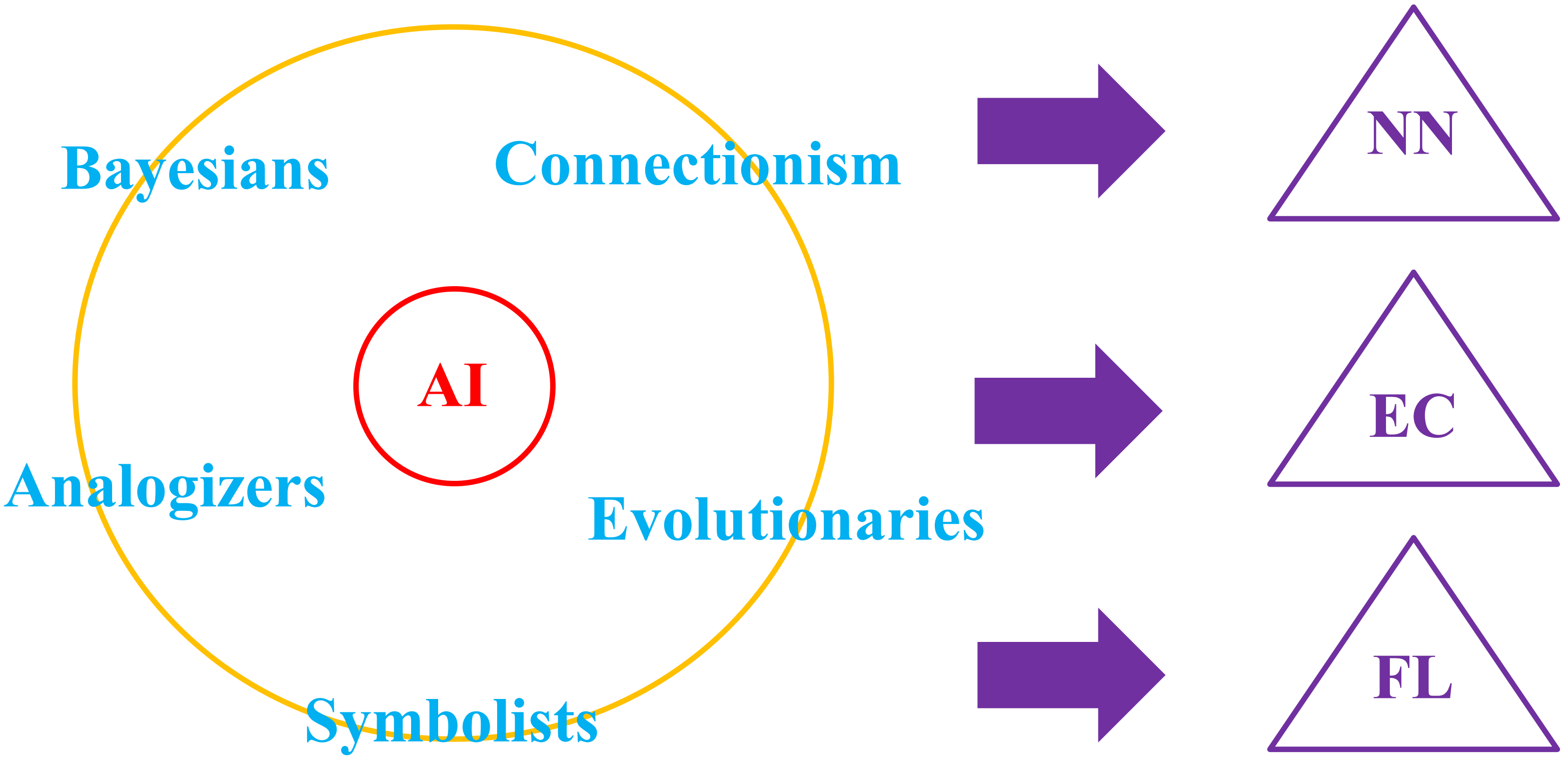
- **Neural Network,**
- **Connectionist machine,**
- **Fuzzy system,**
- **Evolutionary computation,**
- **Autonomous mental development,**

and hybrid intelligent systems in which these paradigms are contained.

Relation between CI and AI

CI is one branch of AI

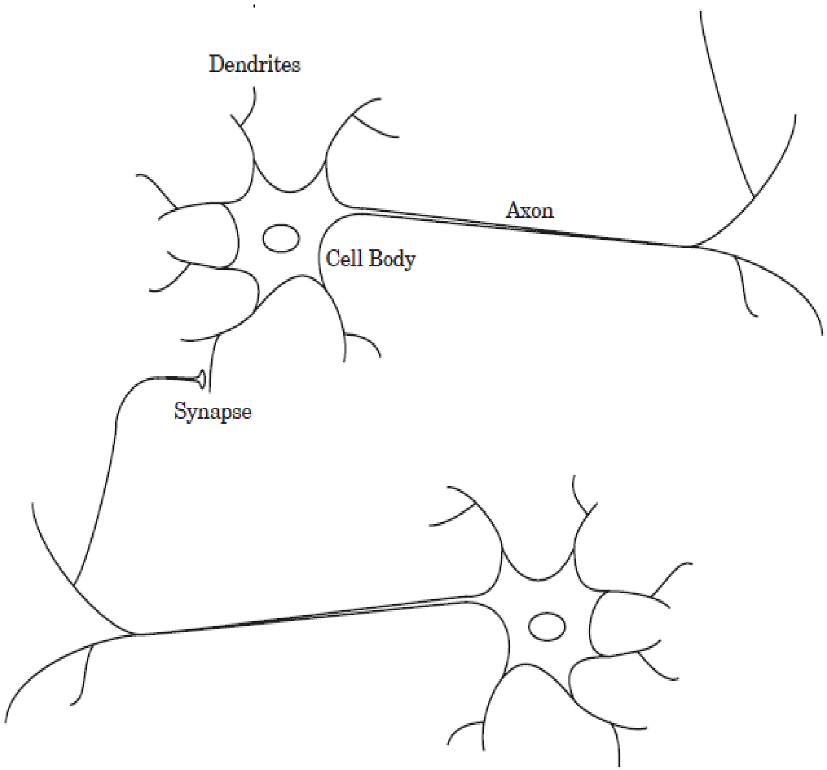
CI is the computational part of AI



Biology

The brain consists of $\approx 10^{11}$ neurons

Each neuron has $\approx 10^{4}$ connections



- **Dendrites**: tree-like receptive networks of nerve fibers that carry electrical signals **into** the cell body
- **Cell body**: sums and thresholds these incoming signals
- **Axon**: a single long fiber that carries the signal from the cell body **out** to other neurons.
- **Synapse**: the point of contact between an axon of one cell and a dendrite of another cell

Evolution

Establish the function of neural network

- By a complex chemical process: arrangement of neurons and the strengths of the individual synapses

Early stages of life

- Some neural structure is defined at birth. Other parts are developed through learning, as new connections are made and others waste away
- If a young cat is denied use of one eye during a critical window of time, it will never develop normal vision in that eye
- Linguists have discovered that infants over six months of age can no longer discriminate certain speech sounds, unless they were exposed to them earlier in life

Continue to change throughout life

- Later changes of neural structures tend to consist mainly of ***strengthening or weakening of synaptic junctions***.
- New memories are formed by ***modification of these synaptic strengths***. Thus, the process of learning a new friend's face consists of altering various synapses
- Neuroscientists have discovered: the hippocampi of London taxi drivers are significantly larger than average, since they must memorize a large amount of navigational information

Biological vs. Artificial

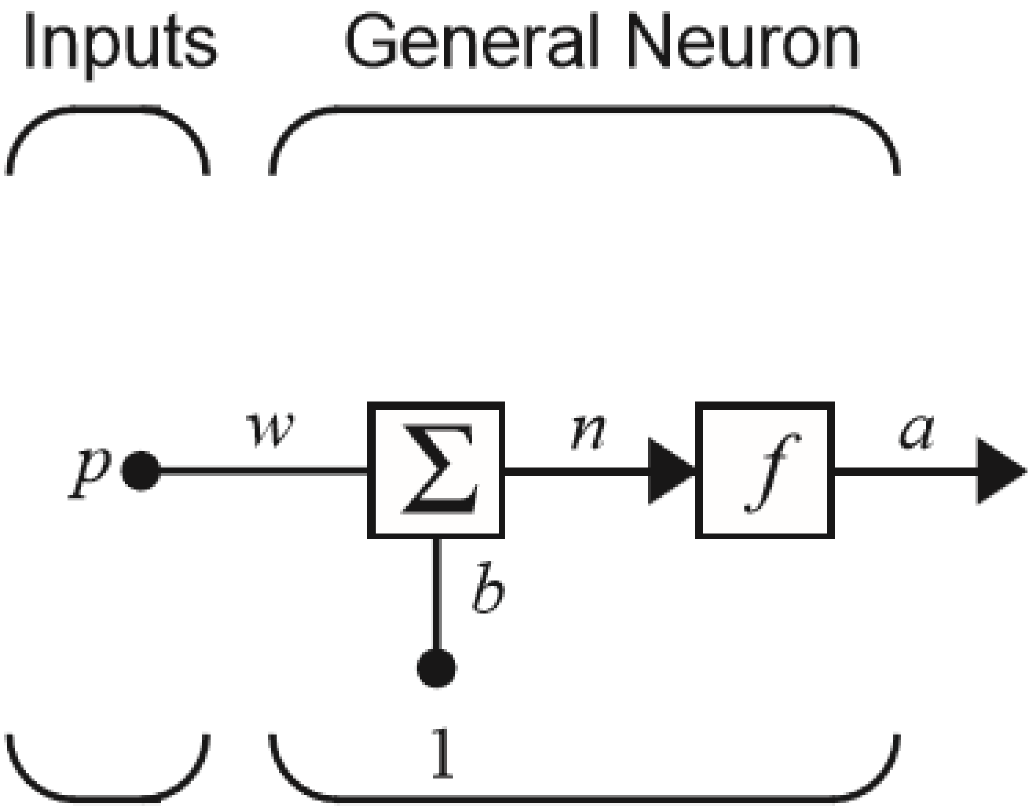
Similarities: parallel structure

- The building blocks of both networks are simple computational devices that are highly interconnected.
- Connections between neurons determine the function of network

Difference: artificial neurons are much simpler

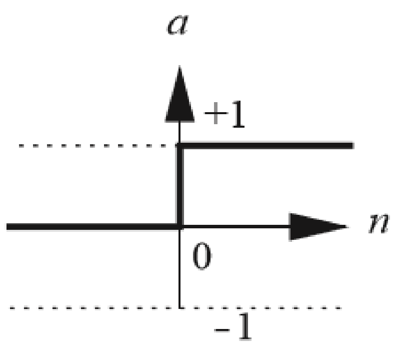
- Biological neurons are very slow compared to electrical circuits: 10^{-3} s compared to 10^{-10} s
- Massively parallel structure of biological neural networks: the brain with all neurons are operating simultaneously performs many tasks much faster
- Implementation: VLSI, optical devices and parallel processors

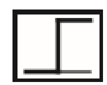
Single-Input Neuron

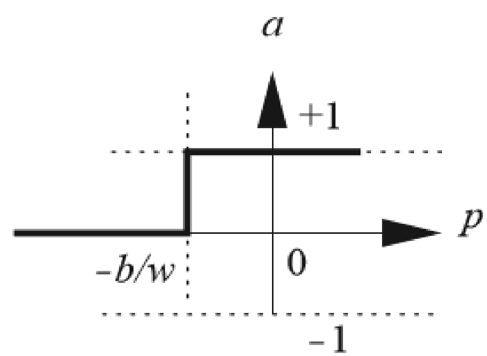


$$a = f(wp + b)$$

Transfer Function





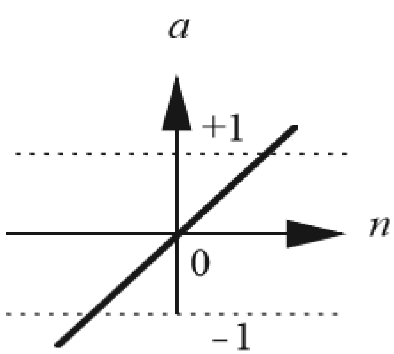


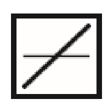
$a = hardlim(n)$

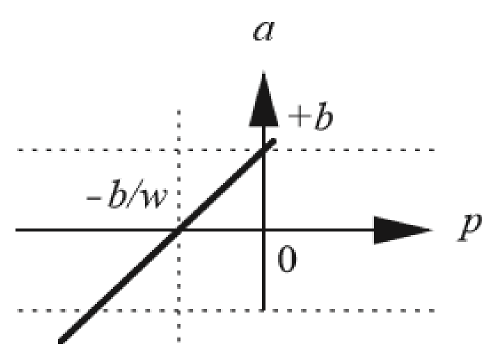
Hard Limit Transfer Function

$a = hardlim(wp + b)$

Single-Input *hardlim* Neuron





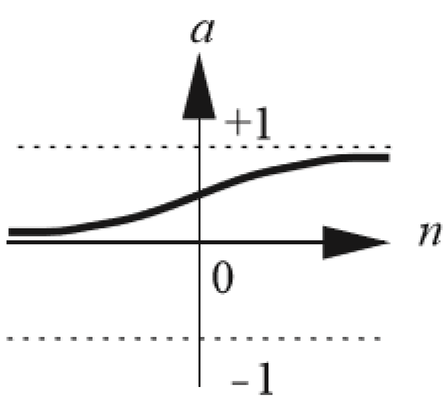


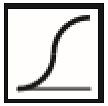
$a = purelin(n)$

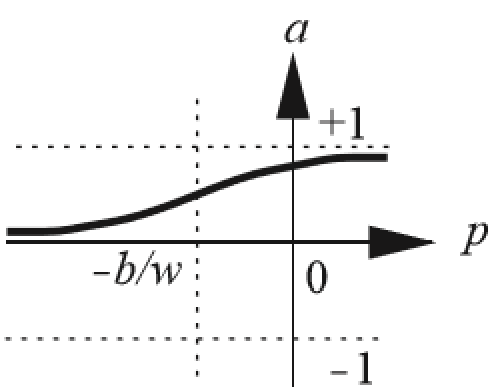
Linear Transfer Function

$a = purelin(wp + b)$

Single-Input *purelin* Neuron







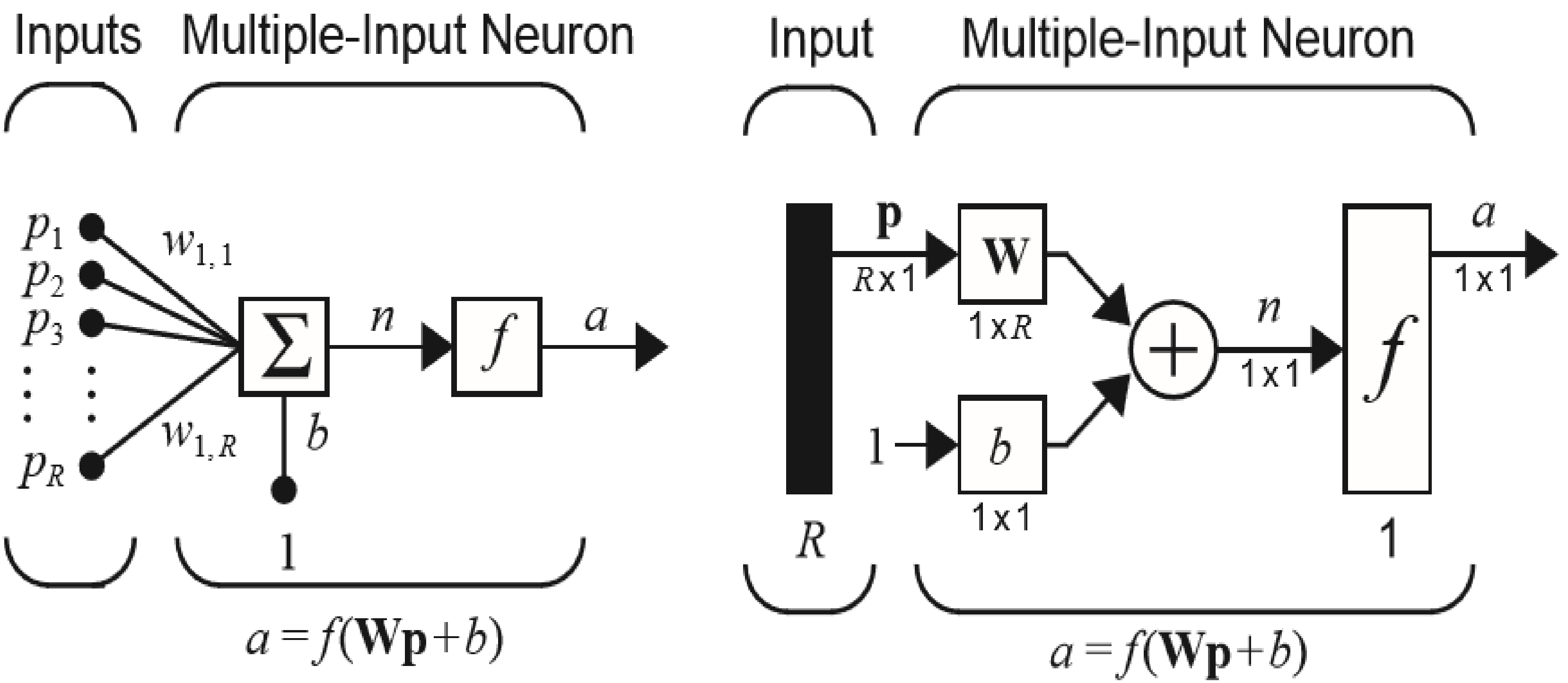
$a = logsig(n)$

Log-Sigmoid Transfer Function

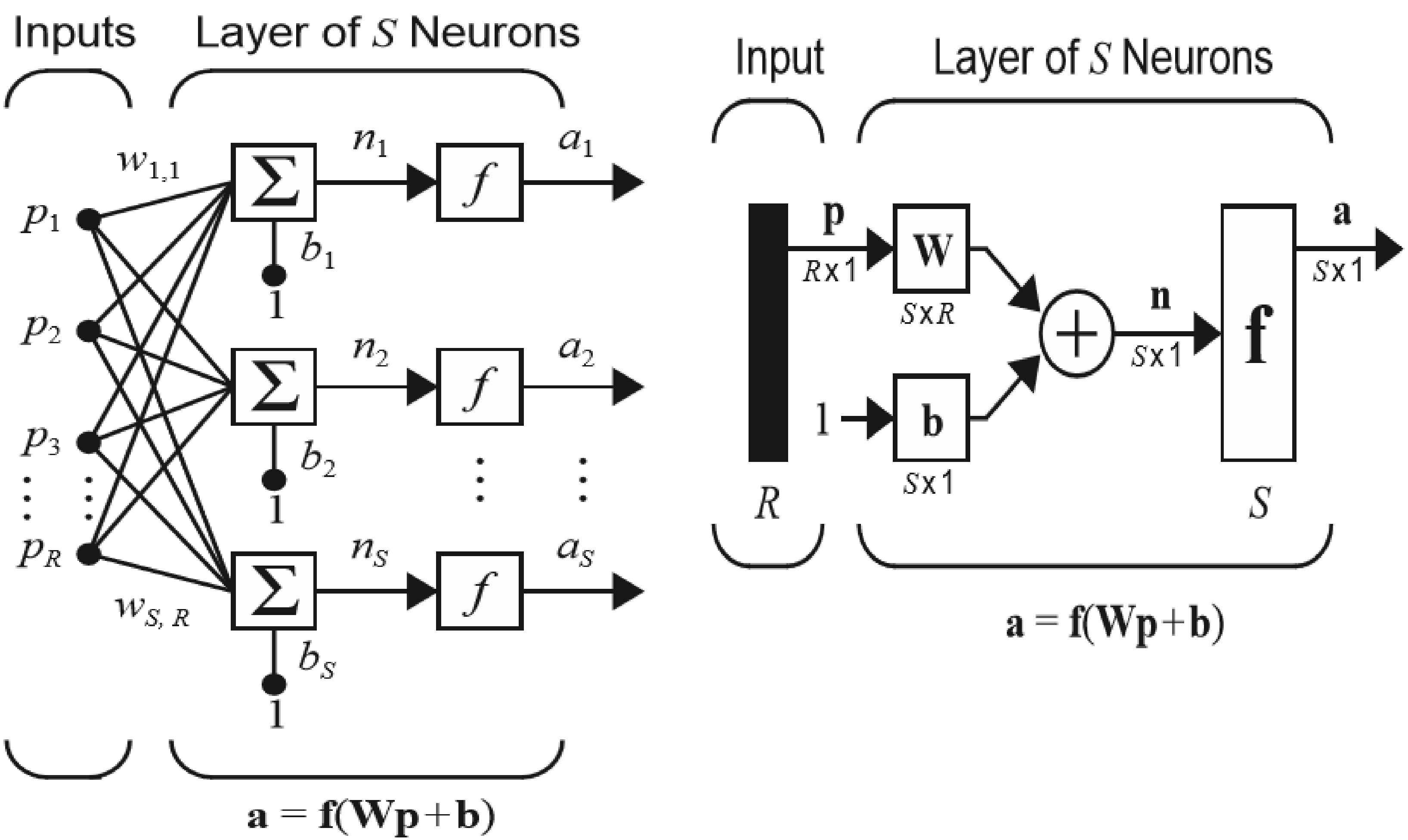
$a = logsig(wp+b)$

Single-Input *logsig* Neuron

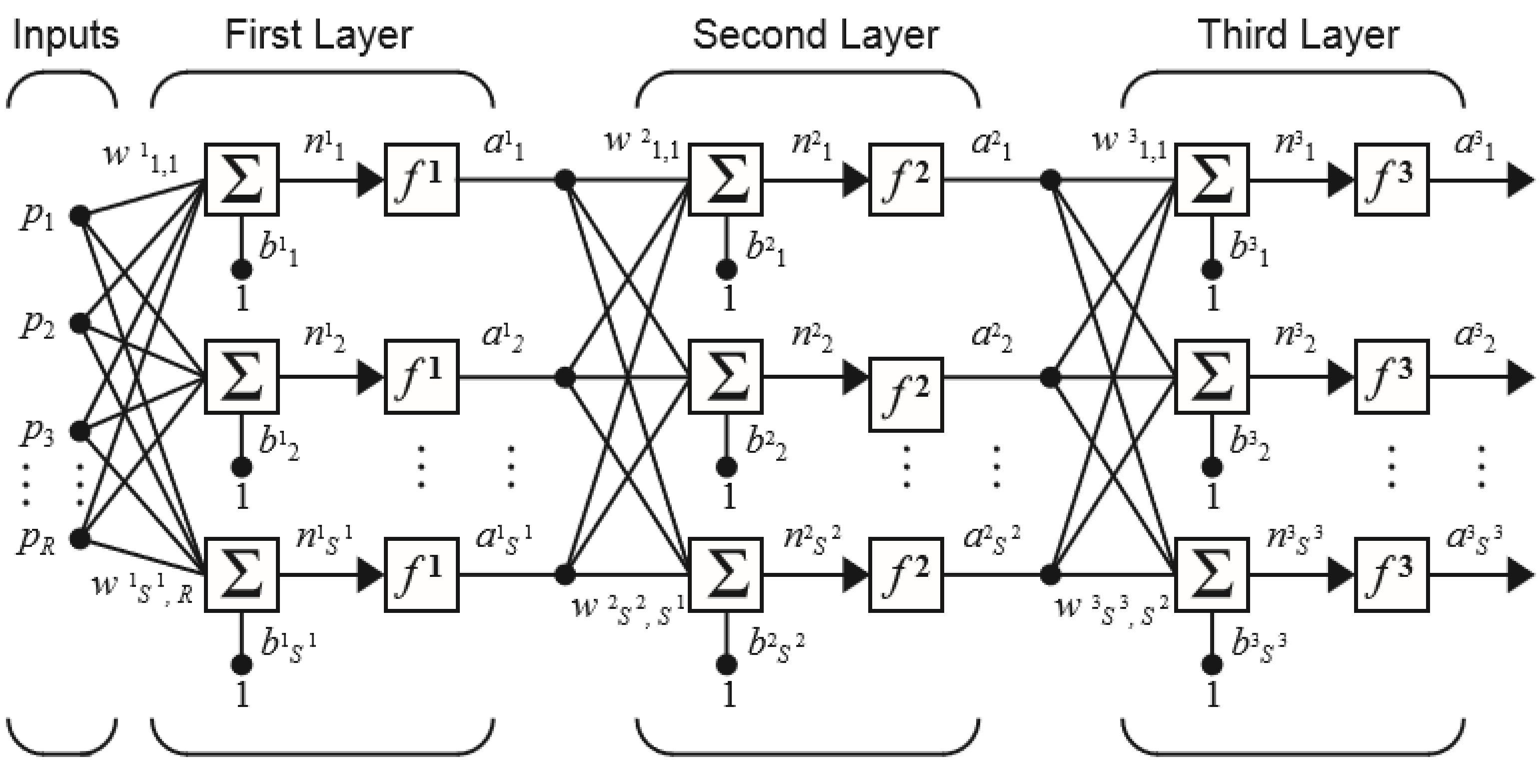
Multiple-Input Neuron



Layer of Neurons



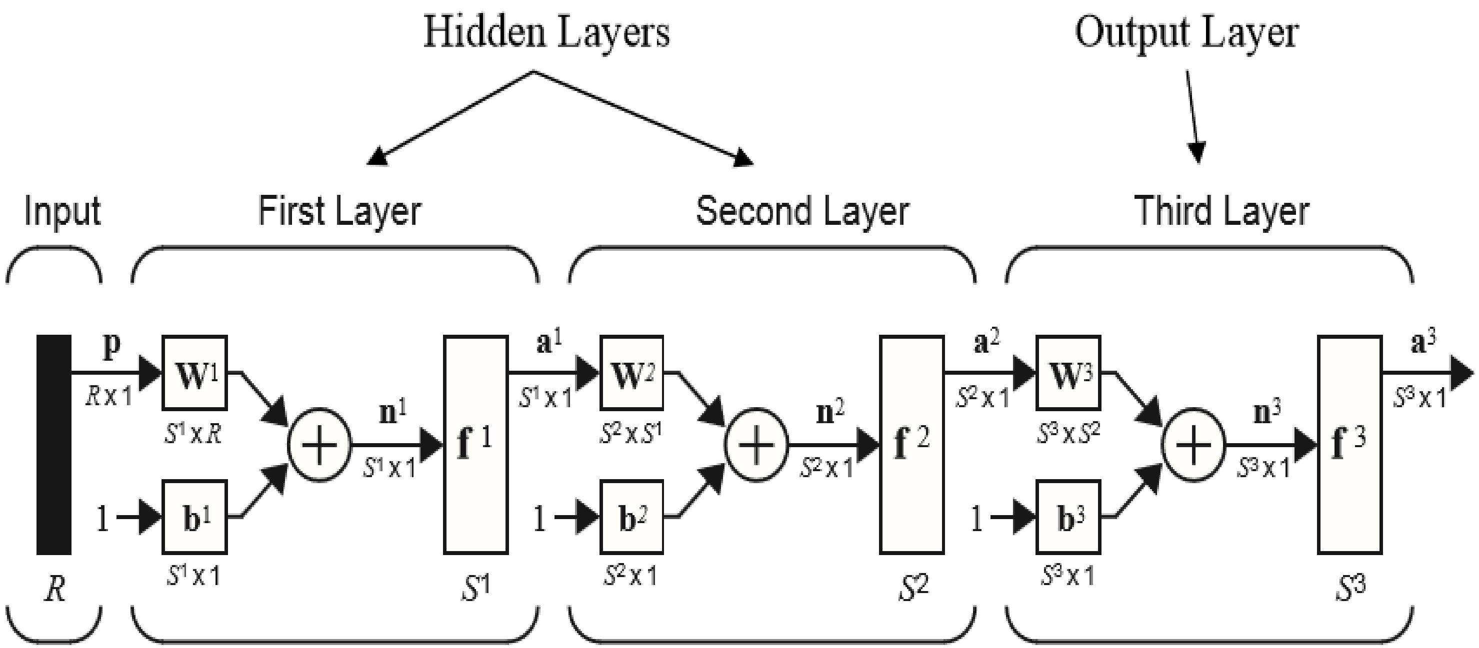
Multilayer Networks



$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)$ $\mathbf{a}^2 = \mathbf{f}^2(\mathbf{W}^2\mathbf{a}^1+\mathbf{b}^2)$ $\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{a}^2+\mathbf{b}^3)$

$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

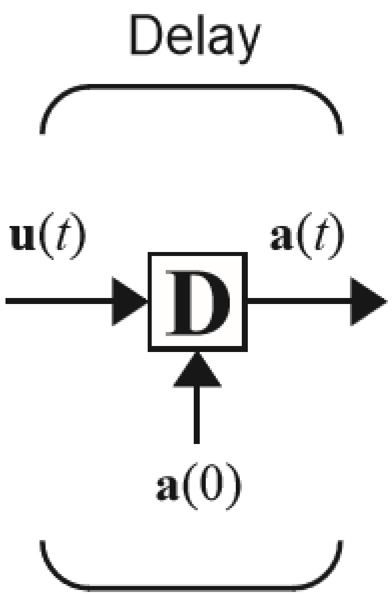
Multilayer Networks



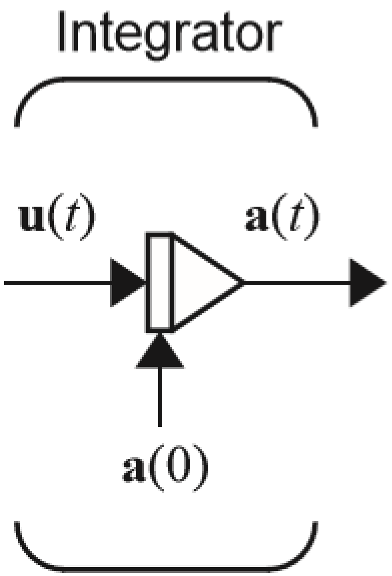
$$\mathbf{a}^1 = \mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1) \qquad \mathbf{a}^2 = \mathbf{f}^2(\mathbf{W}^2\mathbf{a}^1+\mathbf{b}^2) \qquad \mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{a}^2+\mathbf{b}^3)$$

$$\mathbf{a}^3 = \mathbf{f}^3(\mathbf{W}^3\mathbf{f}^2(\mathbf{W}^2\mathbf{f}^1(\mathbf{W}^1\mathbf{p}+\mathbf{b}^1)+\mathbf{b}^2)+\mathbf{b}^3)$$

Delays and Integrators

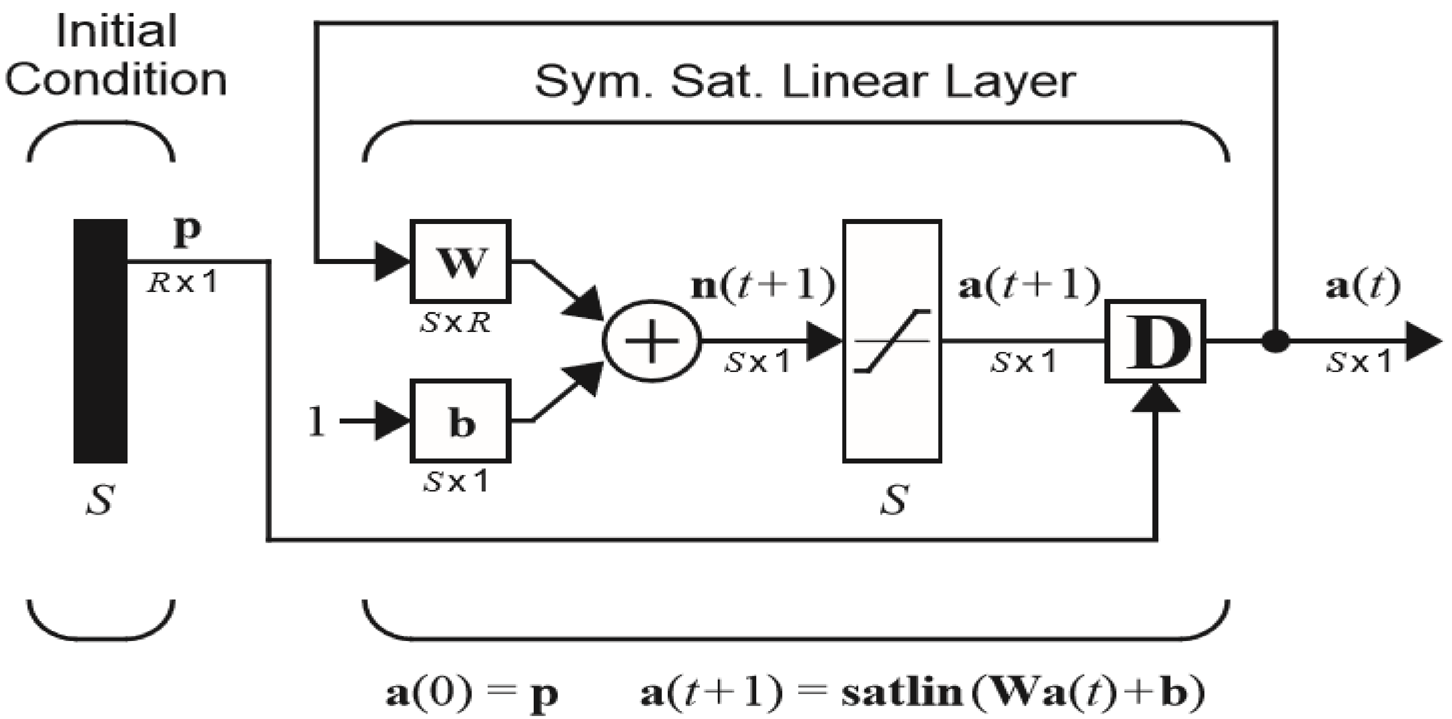


$$\mathbf{a}(t) = \mathbf{u}(t - 1)$$



$$\mathbf{a}(t) = \int_0^t \mathbf{u}(\tau)\, d\tau + \mathbf{a}(0)$$

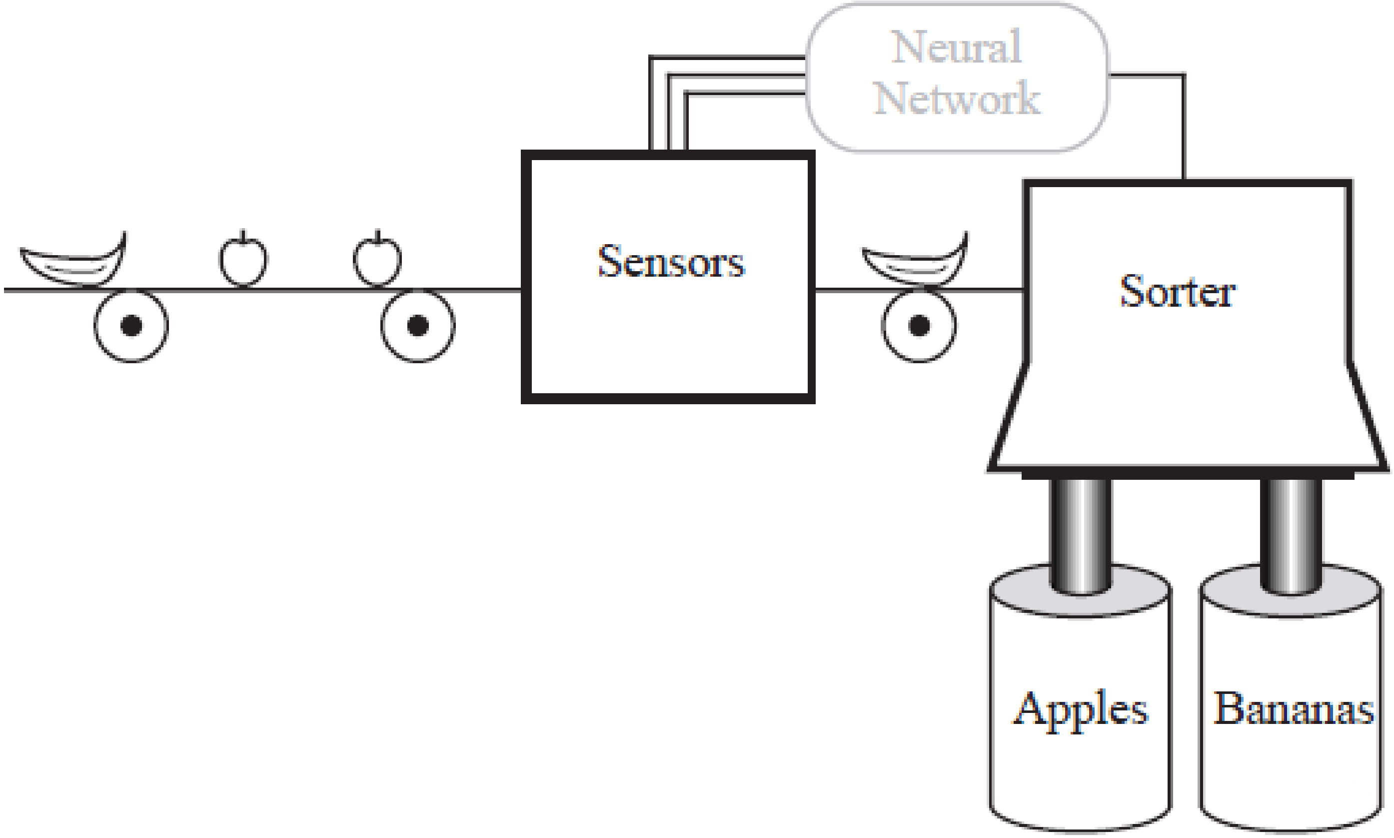
Recurrent Network



$$\mathbf{a}(1) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(0) + \mathbf{b}) = \mathbf{satlins}(\mathbf{W}\mathbf{p} + \mathbf{b})$$

$$\mathbf{a}(2) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(1) + \mathbf{b})$$

Apply/Banana Sorter Problem



Prototype Vectors

Measurement Vector

$$\mathbf{p} = \begin{bmatrix} \text{shape} \\ \text{texture} \\ \text{weight} \end{bmatrix}$$

Shape: {1 : round ; -1 : elliptical}

Texture: {1 : smooth ; -1 : rough}

Weight: {1 : > 1 lb. ; -1 : < 1 lb.}

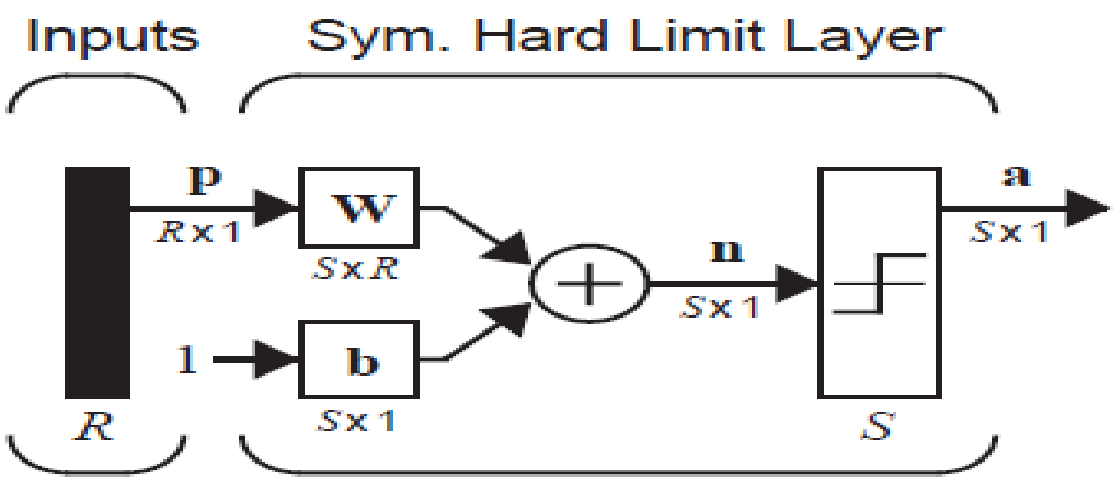
Prototype Banana

$$\mathbf{p}_1 = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$

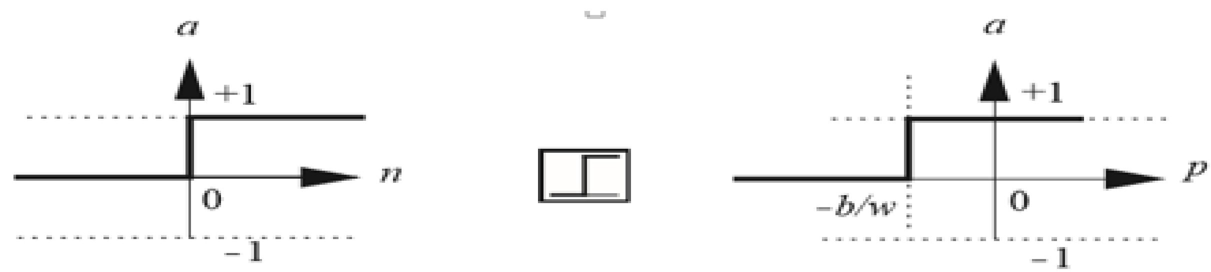
Prototype Apple

$$\mathbf{p}_2 = \begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix}$$

Perceptron



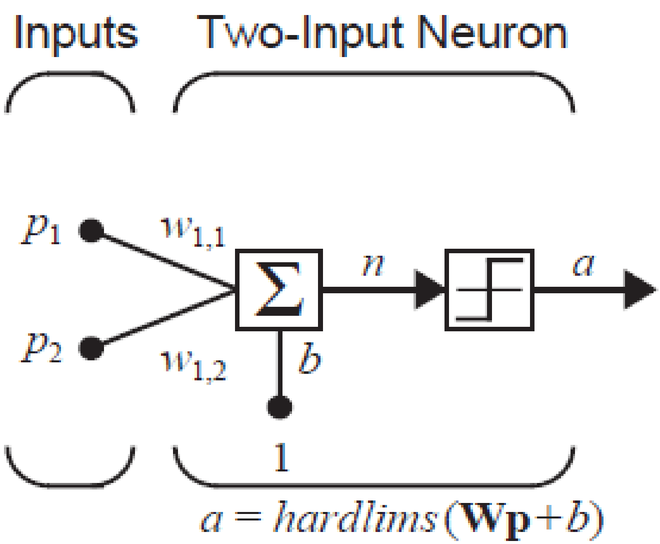
$$\mathbf{a} = \mathbf{hardlims}(\mathbf{Wp} + \mathbf{b})$$

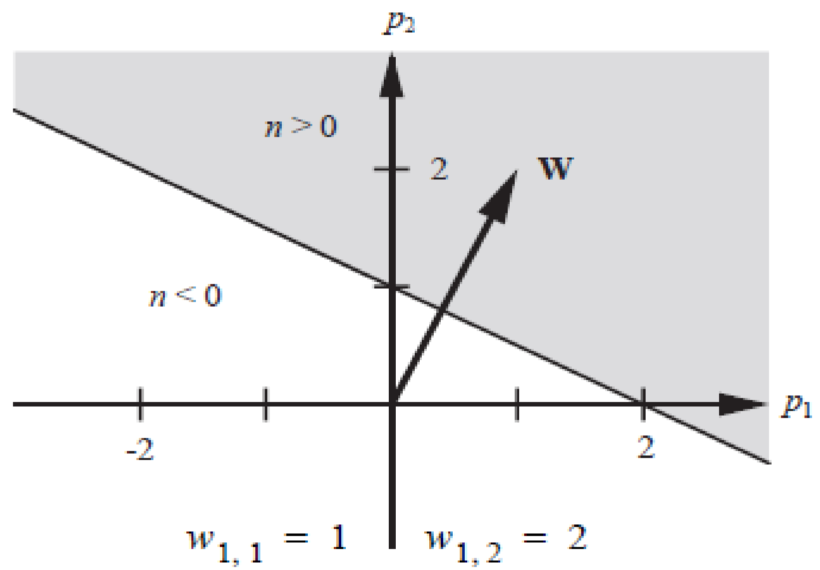


$a = hardlim(n)$
Hard Limit Transfer Function

$a = hardlim(wp + b)$
Single-Input *hardlim* Neuron

Two-Input Case





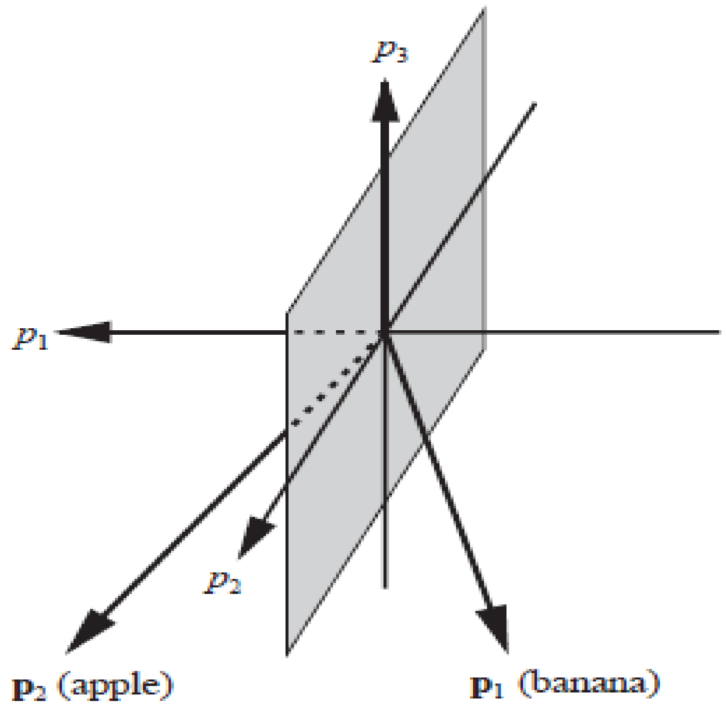
$$a = hardlims(n) = hardlims(\begin{bmatrix} 1 & 2 \end{bmatrix}\mathbf{p} + (-2))$$

Decision Boundary

$$\mathbf{Wp} + b = 0 \qquad \begin{bmatrix} 1 & 2 \end{bmatrix}\mathbf{p} + (-2) = 0$$

Apple/Banana Example

$$a = hardlims\left(\begin{bmatrix} w_{1,1} & w_{1,2} & w_{1,3} \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + b\right)$$



The decision boundary should separate the prototype vectors.

$$p_1 = 0$$

The weight vector

- Be orthogonal to the decision boundary
- Point to the direction of the vector with an output of 1

The bias

- Determines the position of the boundary

$$\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} + 0 = 0$$

Testing the Network

Banana

$$a = hardlims\left(\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} + 0\right) = 1 (\text{banana})$$

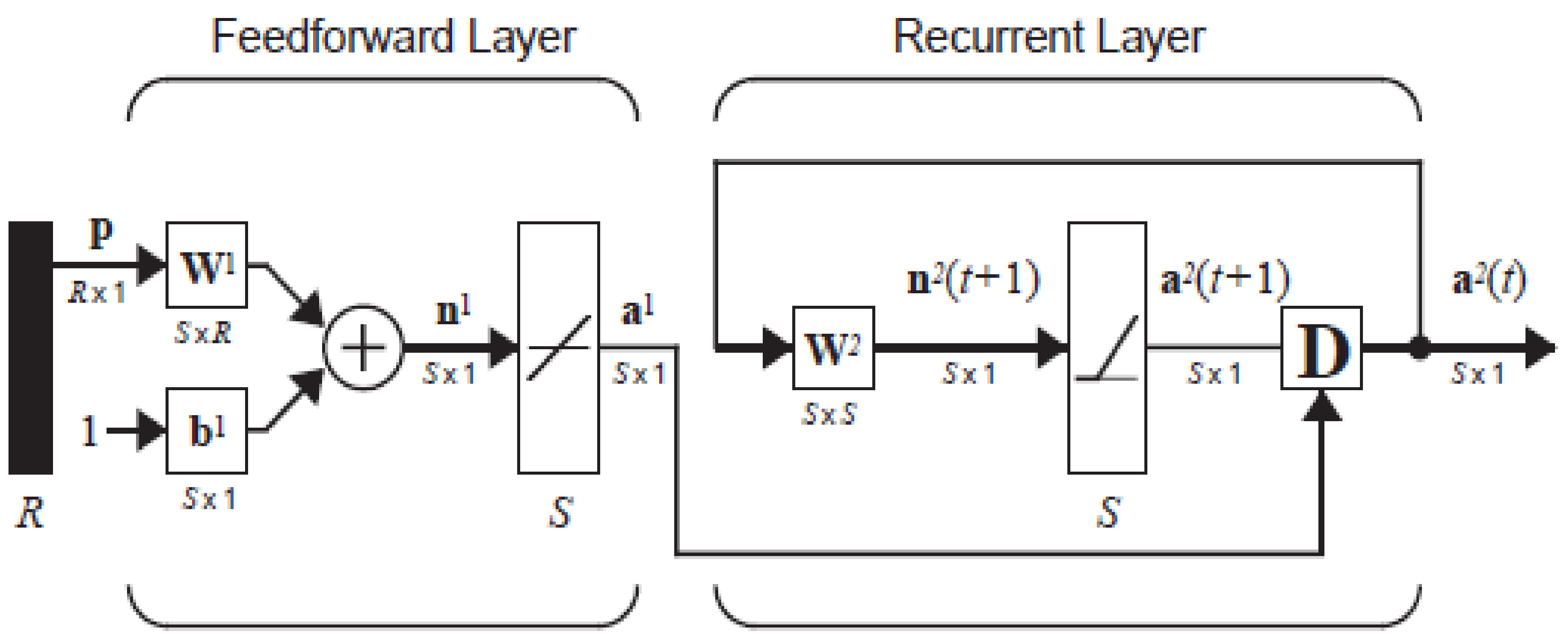
Apple

$$a = hardlims\left(\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ -1 \end{bmatrix} + 0\right) = -1\ (\text{apple})$$

"Rough" Banana

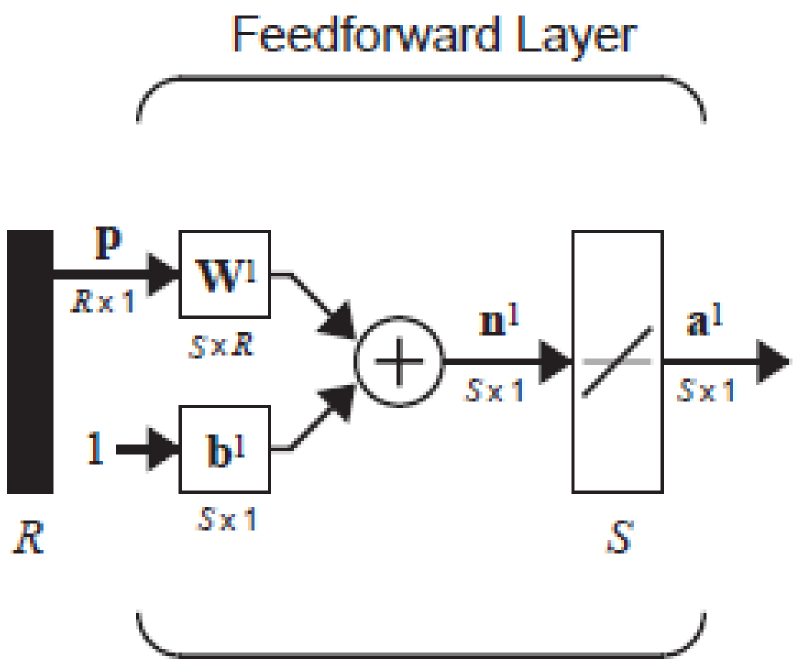
$$a = hardlims\left(\begin{bmatrix} -1 & 0 & 0 \end{bmatrix}\begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + 0\right) = 1 (\text{banana})$$

Hamming Network



$\mathbf{a}^1 = \mathrm{purelin}(\mathbf{W}^1\mathbf{p} + \mathbf{b}^1)$ $\quad$ $\mathbf{a}^2(0) = \mathbf{a}^1$ $\quad$ $\mathbf{a}^2(t+1) = \mathrm{poslin}(\mathbf{W}^2\mathbf{a}^2(t))$

Feedforward Layer



$$\mathbf{a}^1 = \mathrm{purelin}(\mathbf{W}^1\mathbf{p} + \mathbf{b}^1)$$

For Banana/Apple Recognition

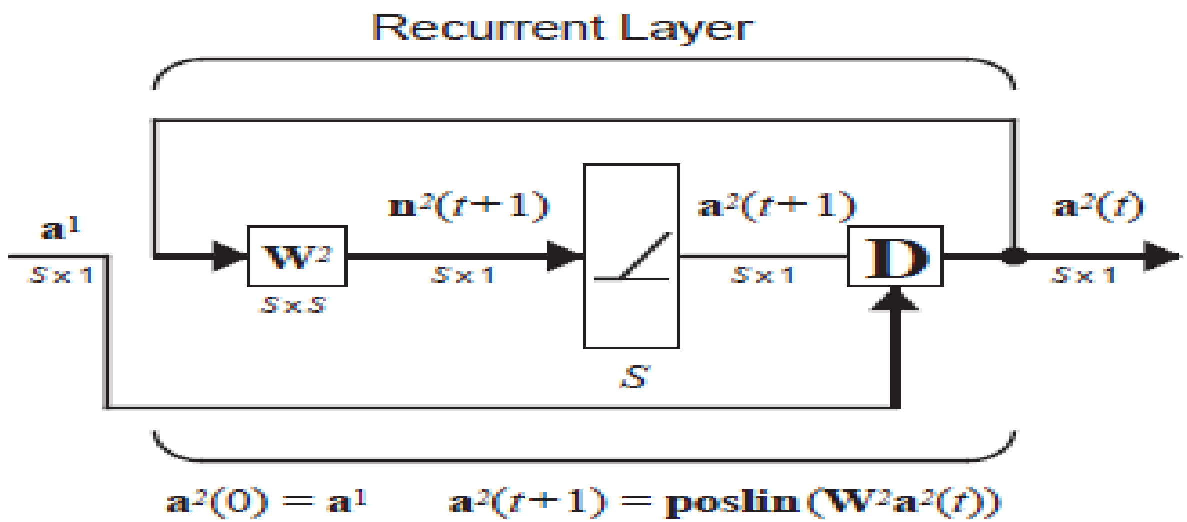
$$S = 2$$

$$\mathbf{W}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix} = \begin{bmatrix} -1 & 1 & -1 \\ 1 & 1 & -1 \end{bmatrix}$$

$$\mathbf{b}^1 = \begin{bmatrix} R \\ R \end{bmatrix} = \begin{bmatrix} 3 \\ 3 \end{bmatrix}$$

$$\mathbf{a}^1 = \mathbf{W}^1\mathbf{p} + \mathbf{b}^1 = \begin{bmatrix} \mathbf{p}_1^T \\ \mathbf{p}_2^T \end{bmatrix}\mathbf{p} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} \mathbf{p}_1^T\mathbf{p} + 3 \\ \mathbf{p}_2^T\mathbf{p} + 3 \end{bmatrix}$$

Recurrent Layer



$$\mathbf{W}^2 = \begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix} \qquad \varepsilon < \frac{1}{S-1}$$

$$\mathbf{a}^2(t+1) = \mathbf{poslin}\left(\begin{bmatrix} 1 & -\varepsilon \\ -\varepsilon & 1 \end{bmatrix}\mathbf{a}^2(t)\right) = \mathbf{poslin}\left(\begin{bmatrix} a_1^2(t) - \varepsilon a_2^2(t) \\ a_2^2(t) - \varepsilon a_1^2(t) \end{bmatrix}\right)$$

Hamming Operation

First Layer

Input (Rough Banana)

$$\mathbf{p} = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix}$$

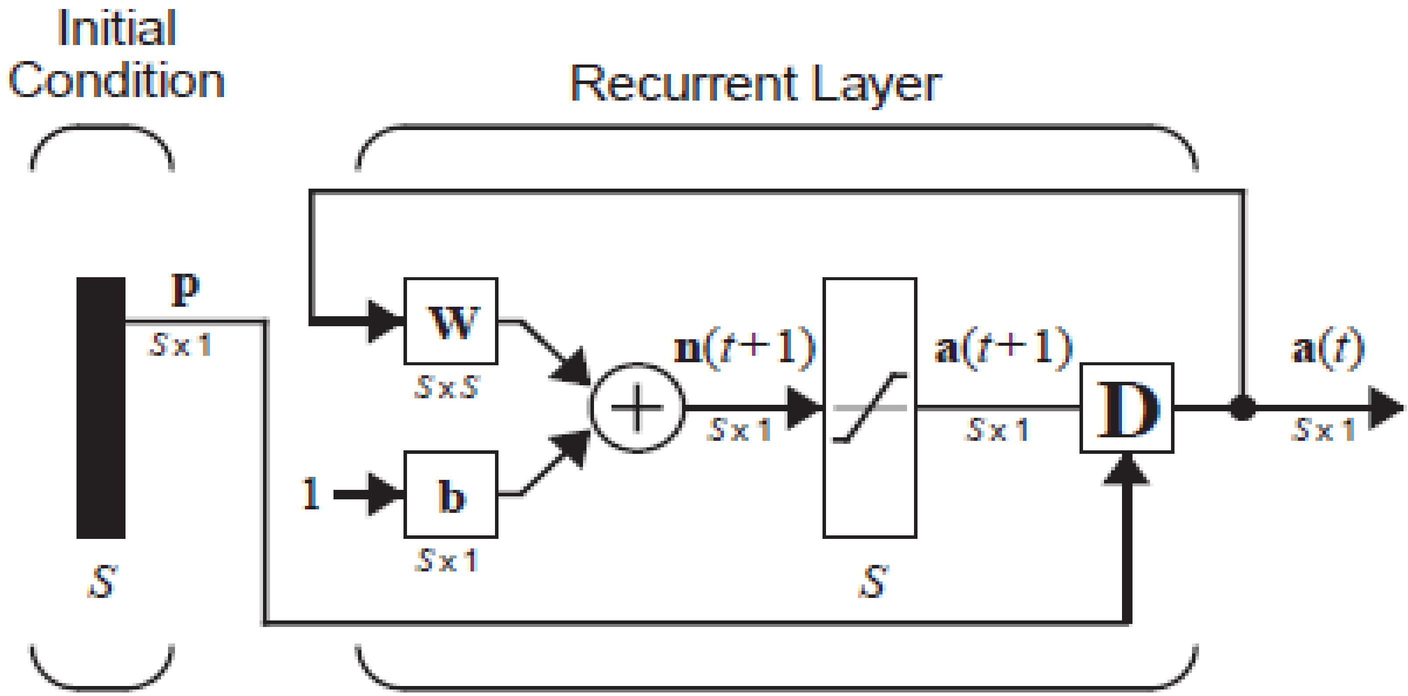
$$\mathbf{a}^1 = \begin{bmatrix} -1 & 1 & -1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} (1+3) \\ (-1+3) \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

Second Layer

$$\mathbf{a}^2(1) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(0)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 4 \\ 2 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

$$\mathbf{a}^2(2) = \mathbf{poslin}(\mathbf{W}^2\mathbf{a}^2(1)) = \begin{cases} \mathbf{poslin}\left(\begin{bmatrix} 1 & -0.5 \\ -0.5 & 1 \end{bmatrix}\begin{bmatrix} 3 \\ 0 \end{bmatrix}\right) \\ \mathbf{poslin}\left(\begin{bmatrix} 3 \\ -1.5 \end{bmatrix}\right) = \begin{bmatrix} 3 \\ 0 \end{bmatrix} \end{cases}$$

Hopfield Network



$\mathbf{a}(0) = \mathbf{p} \qquad \mathbf{a}(t+1) = \mathbf{satlins}(\mathbf{W}\mathbf{a}(t)+\mathbf{b})$

Apple/Banana Problem

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

$$a_1(t+1) = satlins(1.2a_1(t))$$

$$a_2(t+1) = satlins(0.2a_2(t) + 0.9)$$

$$a_3(t+1) = \mathrm{satlins}(0.2a_3(t) - 0.9)$$

Test: "Rough" Banana

$$\mathbf{a}(0) = \begin{bmatrix} -1 \\ -1 \\ -1 \end{bmatrix} \quad \mathbf{a}(1) = \begin{bmatrix} -1 \\ 0.7 \\ -1 \end{bmatrix} \quad \mathbf{a}(2) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix} \quad \mathbf{a}(3) = \begin{bmatrix} -1 \\ 1 \\ -1 \end{bmatrix}$$ (Banana)

Summary

Perceptron

- Feedforward Network
- Linear Decision Boundary
- One Neuron for Each Decision

Hamming Network

- Competitive Network
- First Layer – Pattern Matching (Inner Product)
- Second Layer – Competition (Winner-Take-All)
- # Neurons = # Prototype Patterns

Hopfield Network

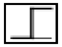
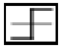
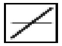
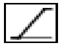
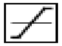
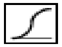
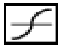
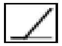
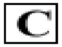
- Dynamic Associative Memory Network
- Network Output Converges to a Prototype Pattern
- # Neurons = # Elements in each Prototype Pattern

How to Pick an Architecture Problem

Specifications help define the network in the following ways

- Number of network inputs = number of problem inputs
- Number of neurons in output layer = number of problem outputs
- Output layer transfer function choice at least partly determined by problem specification of the outputs

The Summary of Transfer Function

Name	Input/Output Relation	Icon	MATLAB Function
Hard Limit	$a = 0 \quad n < 0$ $a = 1 \quad n \geq 0$		hardlim
Symmetrical Hard Limit	$a = -1 \quad n < 0$ $a = +1 \quad n \geq 0$		hardlims
Linear	$a = n$		purelin
Saturating Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n \leq 1$ $a = 1 \quad n > 1$		satlin
Symmetric Saturating Linear	$a = -1 \quad n < -1$ $a = n \quad -1 \leq n \leq 1$ $a = 1 \quad n > 1$		satlins
Log-Sigmoid	$a = \frac{1}{1 + e^{-n}}$		logsig
Hyperbolic Tangent Sigmoid	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
Positive Linear	$a = 0 \quad n < 0$ $a = n \quad 0 \leq n$		poslin
Competitive	$a = 1$ neuron with max n $a = 0$ all other neurons		compet

Exercise

1. The input to a single-input neuron is 2.0, its weight is 2.3 and its bias is -3.

   i. What is the net input to the transfer function?

   ii. If it has the following transfer functions: Hard limit, Linear, and Log-sigmoid, what is the output of the neuron

Answer:

i. The net input is given by:

$n=wp+b=2.3*2+(-3)=1.6$

ii. a=*hardlim* (1.6), a=*purelin* (1.6), a=*logsig* (1.6),

Exercise

2. A single-layer neural network is to have six inputs and two outputs. The outputs are to be limited to and continuous over the range 0 to 1. What can you tell about the network architecture?

Specifically:

i. How many neurons are required?

ii. What are the dimensions of the weight matrix?

iii. What kind of transfer functions could be used?

iv. Is a bias required?

The problem specifications allow you to say the following about the network.

i. Two neurons, one for each output, are required.

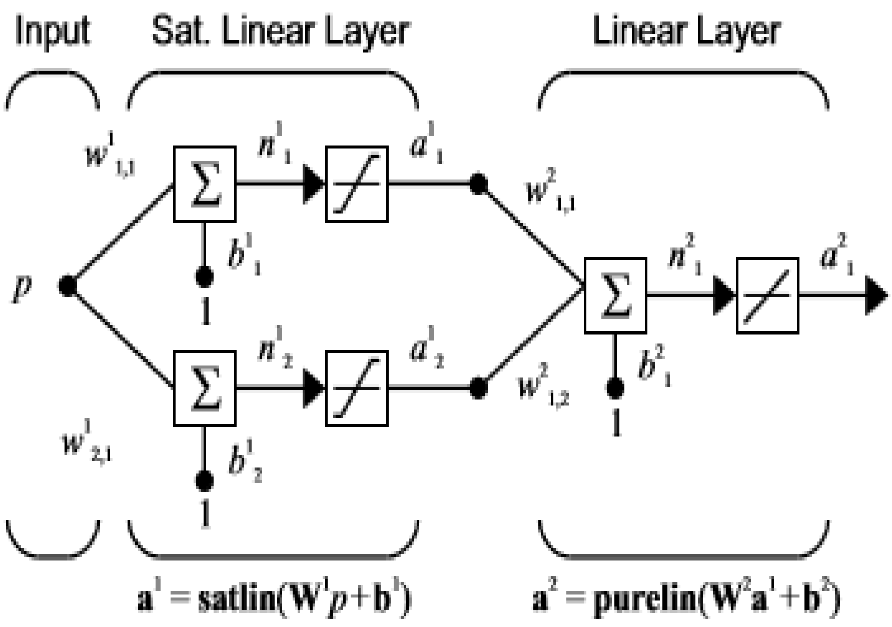
ii. The weight matrix has two rows corresponding to the two neurons and six columns corresponding to the six inputs. (The product is a two-element vector.)

iii. Of the transfer functions we have discussed, the transfer function would be most appropriate.

iv. Not enough information is given to determine if a bias is required

Homework

Problem 1: Consider the following neural network



$w^1_{1,1} = 2$, $w^1_{2,1} = 1$, $b^1_1 = 2$, $b^1_2 = -1$, $w^2_{1,1} = 1$, $w^2_{1,2} = -1$, $b^2_1 = 0$

Sketch the following responses (plot the indicated variable versus p for -3<p<3):

i. n^1_1.

ii. a^1_1.

iii. n^1_2

iv. a^1_2.

v. n^2_1.

vi. a^2_1.

Homework

Problem 2: Consider the following prototype patterns

$$\mathbf{p}_1 = \begin{bmatrix} 1 \\ 0.5 \end{bmatrix}, \mathbf{p}_2 = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Find weights and bias which will produce the decision boundary for a perceptron network that will recognize these two vectors.