Chapter 3

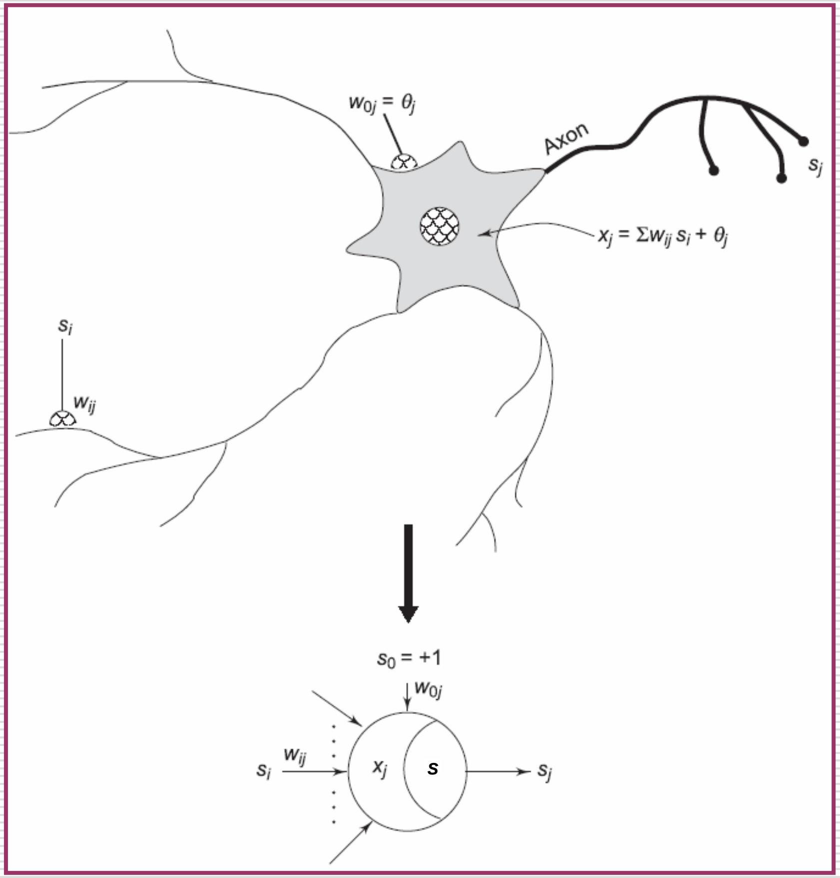
Artificial Neurons, Neural Networks and Architectures

Neural Networks: A Classroom Approach

Satish Kumar
Department of Physics & Computer Science
Dayalbagh Educational Institute (Deemed University)

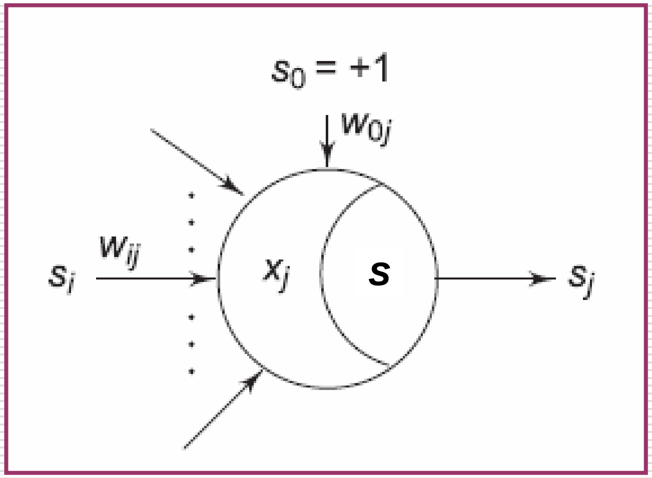
Neuron Abstraction

- Neurons transduce signals—electrical to chemical, and from chemical back again to electrical.
- Each synapseis associated with what we call the synaptic efficacy—the efficiency with which a signal is transmitted from the presynaptic to postsynaptic neuron.



Neuron Abstraction: Activations and Weights

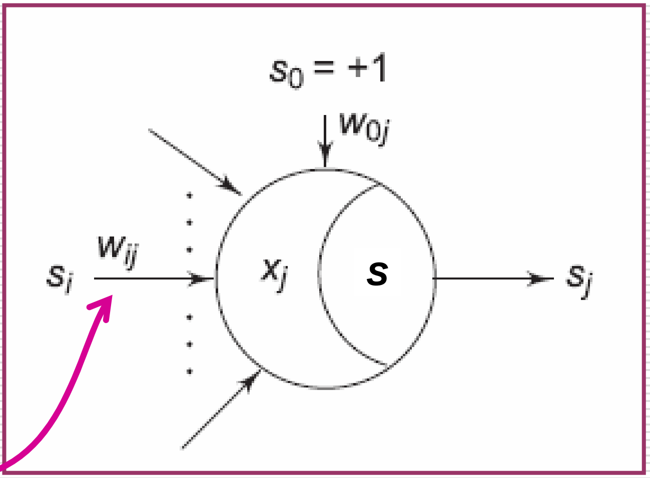
- j th artificial neuron that receives input signals s_i , from possibly n different sources
- an internal activation x_j which is a linear weighted aggregation of the impinging signals, modified by an internal threshold, θ_j
- connection weights w_{ij} model the synaptic efficacies of various interneuron synapses.



$$x_j = \sum_{i=1}^{n} w_{ij} s_i + \theta_j$$

Notation:

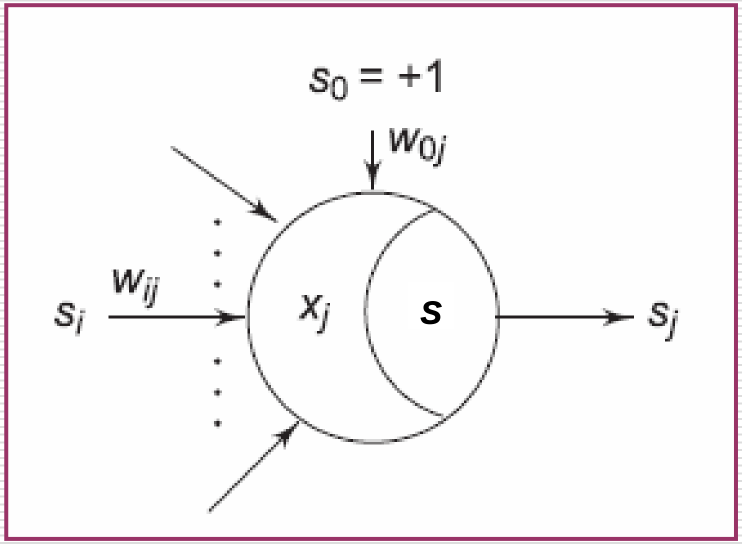
- w_{ij} denotes the weight from neuron *i* to neuron *j*.



$$x_j = \sum_{i=1}^{n} w_{ij} s_i + \theta_j$$

Neuron Abstraction: Signal Function

- The activation of the neuron is subsequently transformed through a *signal function* $S(\cdot)$
- Generates the output signal $s_j = S(x_j)$ of the neuron.
- a signal function may typically be
  - binary threshold
  - linear threshold
  - sigmoidal
  - Gaussian
  - probabilistic.



Activations Measure Similarities

- The activation x_j is simply the inner product of the impinging signal vector $S = (s_0, \ldots, s_n)^T$, with the neuronal weight vector $W_j = (w_{0j}, \ldots, w_{nj})^T$

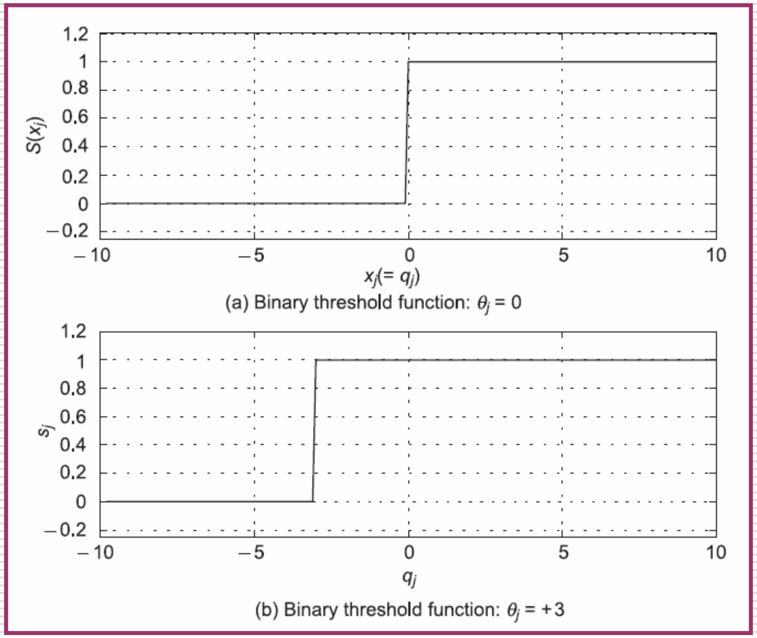
Adaptive Filter

$$x_j = S^T W_j = \sum_{i=0}^{n} w_{ij} s_i$$

Neuron Signal Functions: Binary Threshold Signal Function

- Net positive activations translate to a +1 signal value
- Net negative activations translate to a 0 signal value.
- The threshold logic neuron is a two state machine
  - $s_j = S(x_j) \in \{0, 1\}$

$$\mathcal{S}(x_j) = \begin{cases} 1 & x_j \geq 0 \\ 0 & x_j < 0 \end{cases}$$



(a) Binary threshold function: $\theta_j = 0$

(b) Binary threshold function: $\theta_j = +3$

Threshold Logic Neuron (TLN) in Discrete Time

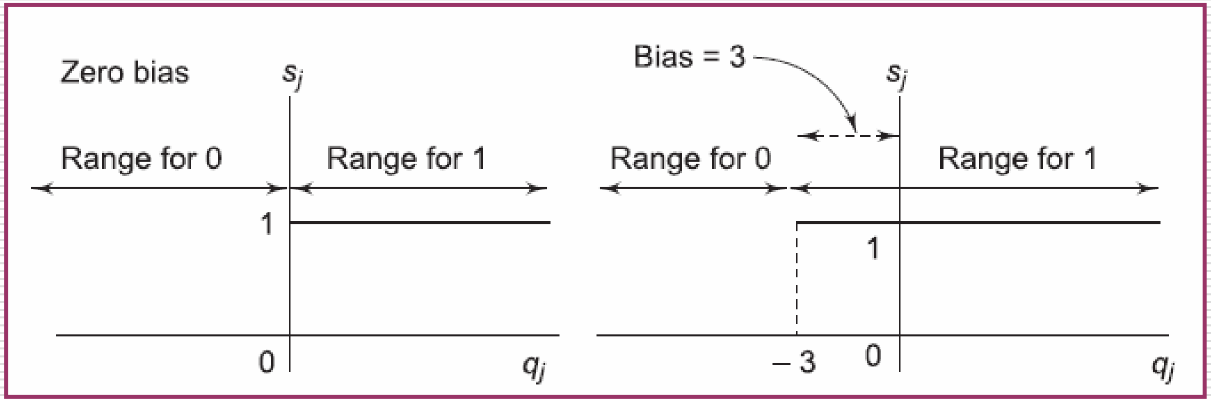
- The updated signal value $S(x_j^{k+1})$ at time instant $k + 1$ is generated from the neuron activation x_i^{k+1} , sampled at time instant $k + 1$.
- The response of the threshold logic neuron as a two-state machine can be extended to the *bipolar* case where the signals are
  - s_j □ $\{-1, 1\}$
- The resulting signal function is then none other than the *signum function*, sign(x) commonly encountered in communication theory.

$$\mathcal{S}(x_j^{k+1}) = \begin{cases} 1 & x_j^{k+1} > 0 \\ \mathcal{S}(x_j^k) & x_j^{k+1} = 0 \\ 0 & x_j^{k+1} < 0 \end{cases}$$

$$\mathcal{S}(x_j) = \begin{cases} +1 & x_j > 0 \\ -1 & x_j < 0 \end{cases}$$

Interpretation of Threshold

$$x_j = \sum_{i=0}^{n} w_{ij} s_i = \sum_{i=1}^{n} w_{ij} s_i + w_{0j} = q_j + \theta_j$$

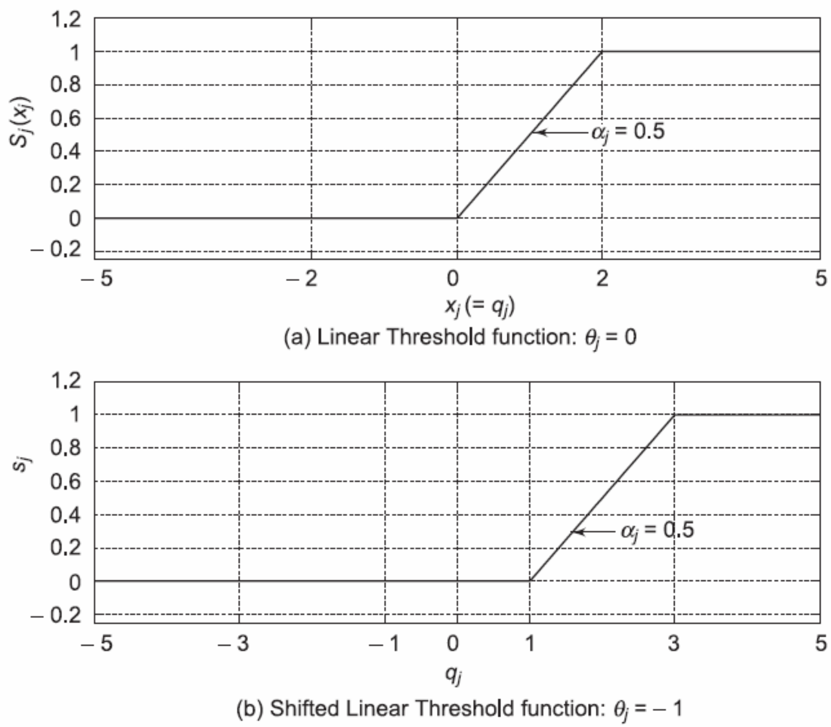


- From the point of view of the net activation x_j
  - the signal is +1 if $x_j = q_j + \theta_j \geq 0$, or $q_j \geq -\theta_j$;
  - and is 0 if $q_j < -\theta_j$.
- The neuron thus "compares" the net external input q_j
  - if q_j is greater than the negative threshold, it fires +1, otherwise it fires 0.

Linear Threshold Signal Function

$$\mathcal{S}_j(x_j) = \begin{cases} 0 & x_j \le 0 \\ \alpha_j\, x_j & 0 < x_j < x_m \\ 1 & x_j \ge x_m \end{cases}$$

- $a_j = 1/x_m$ is the slope parameter of the function
- Figure plotted for $x_m = 2$ and $a_j = 0.5$.
- $S_j(x_j) = \max(0, \min(a_j x_j, 1))$
- Note that in this course we assume that neurons within a network are homogeneous.



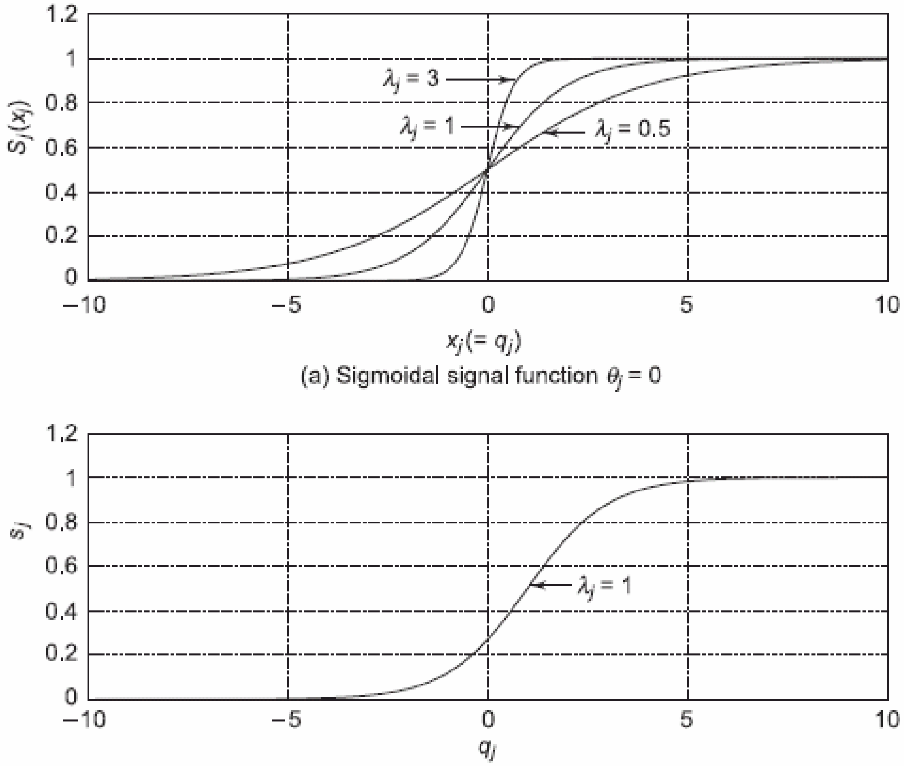
(a) Linear Threshold function: $\theta_j = 0$

(b) Shifted Linear Threshold function: $\theta_j = – 1$

Sigmoidal Signal Function

$$S_j(x_j) = \frac{1}{1 + e^{-\lambda_j x_j}}$$

- λ_j is a gain scale factor
- In the limit, as $\lambda_j \rightarrow \infty$ the smooth logistic function approaches the non-smooth binary threshold function.
- The sigmoidal signal function has some very useful mathematical properties. It is
  - monotonic
  - continuous
  - bounded



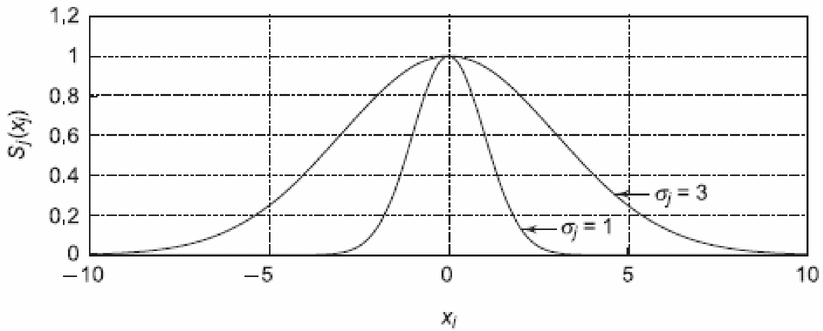
(a) Sigmoidal signal function $\theta_j = 0$

(b) Shifted sigmoidal signal function $\theta_j = -1$

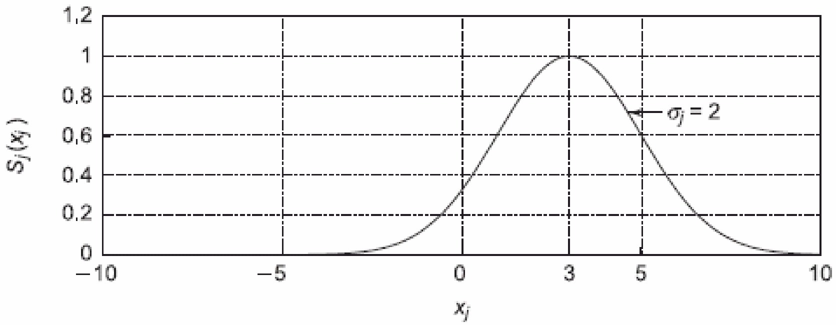
Gaussian Signal Function

$$\mathcal{S}_j(x_j) = \exp\left(-\frac{(x_j - c_j)^2}{2\sigma_j^2}\right)$$

- σ_j is the Gaussian spread factor and c_j is the center.
- Varying the spread makes the function sharper or more diffuse.
- Changing the center shifts the function to the right or left along the activation axis
- This function is an example of a *non-monotonic* signal function



(a) Gaussian signal function: center = 0



(b) Gaussian signal function: center = 3

Stochastic Neurons

- The signal is assumed to be two state
  - $s_j \in \{0, 1\}$ or $\{-1, 1\}$
- Neuron switches into these states depending upon a *probabilistic function of its activation*, $P(x_j)$.

$$P(x_j) = \frac{1}{1 + e^{-x_j/T}}$$

Summary of Signal Functions

Name	*Function*	*Characteristics*
Binary threshold	$\mathcal{S}(x_j) = \begin{cases} 1 & x_j \geq 0 \\ 0 & x_j < 0 \end{cases}$	Non-differentiable, step-like, $s_j \in \{0, 1\}$
Bipolar threshold	$\mathcal{S}(x_j) = \begin{cases} 1 & x_j \geq 0 \\ -1 & x_j < 0 \end{cases}$	Non-differentiable, step-like, $s_j \in \{-1, 1\}$
Linear	$\mathcal{S}_j(x_j) = \alpha_j x_j$	Differentiable, unbounded, $s_j \in (-\infty, \infty)$
Linear threshold	$\mathcal{S}_j(x_j) = \begin{cases} 0 & x_j \leq 0 \\ \alpha_j x_j & 0 < x_j < x_m \\ 1 & x_j \geq x_m \end{cases}$	Differentiable, piece-wise linear, $s_j \in [0, 1]$
Sigmoid	$\mathcal{S}_j(x_j) = \dfrac{1}{1 + e^{-\lambda_j x_j}}$	Differentiable, monotonic, smooth, $s_j \in (0, 1)$
Hyperbolic tangent	$\mathcal{S}_j(x_j) = \tanh(\lambda_j x_j)$	Differentiable, monotonic, smooth, $s_j \in (-1, 1)$
Gaussian	$e^{-(x_j - c_j)^2 / 2\sigma_j^2}$	Differentiable, non-monotonic, smooth, $s_j \in (0, 1)$
Stochastic	$\mathcal{S}_j(x_j) = \begin{cases} +1 & \text{with probability } P(x_j) \\ -1 & \text{with probability } 1 - P(x_j) \end{cases}$	Non-deterministic step-like, $s_j \in \{0, 1\}$ or $\{-1, 1\}$

Neural Networks Defined

- Artificial neural networks are massively parallel adaptive networks of simple nonlinear computing elements called neurons which are intended to abstract and model some of the functionality of the human nervous system in an attempt to partially capture some of its computational strengths.

Eight Components of Neural Networks

- *Neurons*. These can be of three types:
  - Input: receive external stimuli
  - Hidden: compute intermediate functions
  - Output: generate outputs from the network
- *Activation state vector*. This is a vector of the activation level x_i of individual neurons in the neural network,
  - $X = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$.
- *Signal function*. A function that generates the output signal of the neuron based on its activation.

Eight Components of Neural Networks

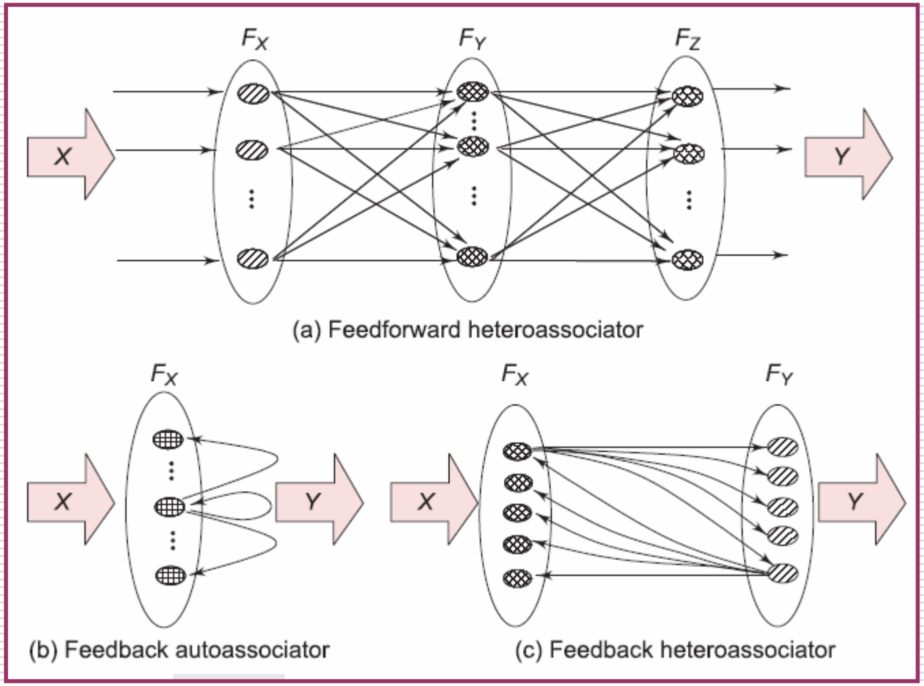
- *Pattern of connectivity.* This essentially determines the inter-neuron connection architecture or the graph of the network. Connections which model the inter-neuron synaptic efficacies, can be
  - excitatory (+)
  - inhibitory (−)
  - absent (0).
- *Activity aggregation rule.* A way of aggregating activity at a neuron, and is usually computed as an inner product of the input vector and the neuron fan-in weight vector.

Eight Components of Neural Networks

- *Activation rule.* A function that determines the new activation level of a neuron on the basis of its current activation and its external inputs.
- *Learning rule.* Provides a means of modifying connection strengths based both on external stimuli and network performance with an aim to improve the latter.
- *Environment.* The environments within which neural networks can operate could be
  - deterministic (noiseless) or
  - stochastic (noisy).

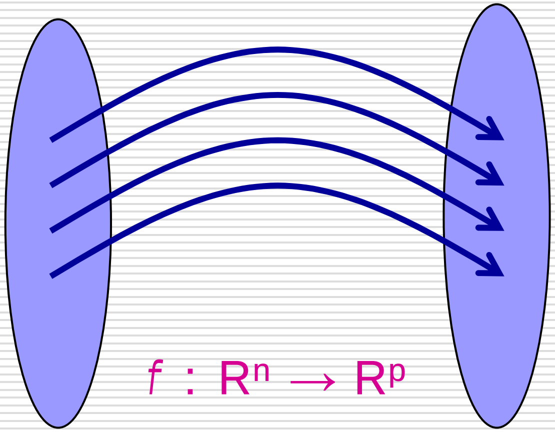
Architectures: Feedforward and Feedback

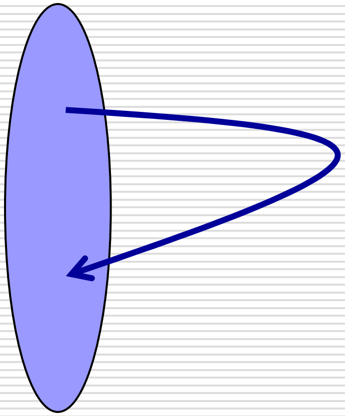
- Local groups of neurons can be connected in either,
  - a *feedforward* architecture, in which the network has no loops, or
  - a *feedback* (recurrent) architecture, in which loops occur in the network because of feedback connections.



Neural Networks Generate Mappings

- Multilayered networks that associate vectors from one space to vectors of another space are called *heteroassociators*.
  - Map or associate two different patterns with one another—one as input and the other as output. Mathematically we write, $f : R^n \to R^p$.
- When neurons in a single field connect back onto themselves the resulting network is called an *autoassociator* since it associates a single pattern in R^n with itself.



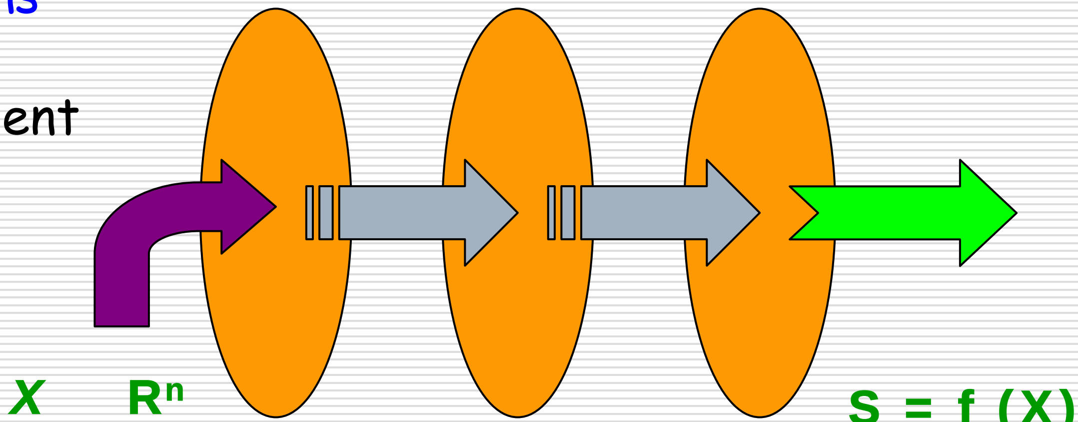


Activation and Signal State Spaces

- For a p-dimensional field of neurons, the activation state space is R^p.
- The signal state space is the Cartesian cross space,
  - $I^p = [0, 1] \times \cdots \times [0, 1]$ p *times* $= [0, 1]^p \subset R^p$ if the neurons have continuous signal functions in the interval $[0, 1]$
  - $[-1, 1]^p$ if the neurons have continuous signal functions in the interval $[-1, 1]$.
- For the case when the neuron signal functions are binary threshold, the signal state space is
  - $B^p = \{0, 1\} \times \cdots \times \{0, 1\}$ p *times* $= \{0, 1\}^p \subset I^p \subset R^p$
  - $\{-1, 1\}^p$ when the neuron signal functions are bipolar threshold.

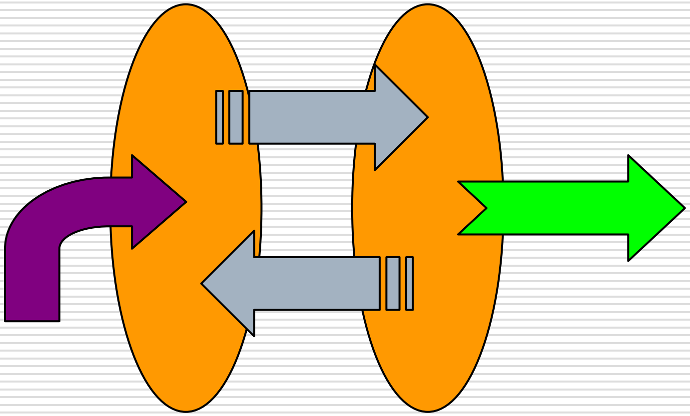
Feedforward vs Feedback: *Multilayer Perceptrons*

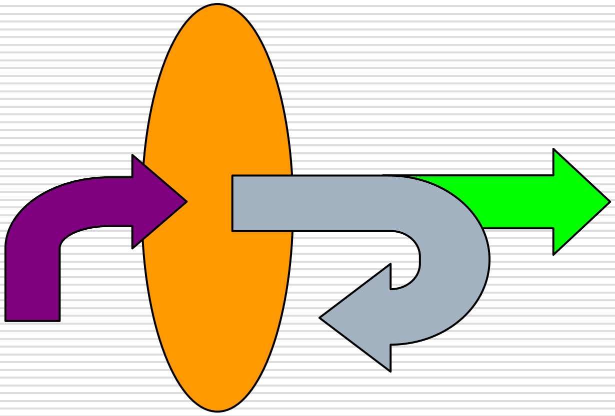
- Organized into different layers
- Unidirectional connections
- memory-less: output depends only on the present input
- Possess no dynamics
- Demonstrate powerful properties
  - Universal function approximation
- Find widespread applications in pattern classification.



Feedforward vs Feedback: Recurrent Neural Networks

- *Non-linear dynamical systems*
- New state of the network is a function of the current input and the present state of the network
- Possess a rich repertoire of dynamics
- Capable of performing powerful tasks such as
  - pattern completion
  - topological feature mapping
  - pattern recognition





More on Feedback Networks

- Network activations and signals are in a flux of change until they settle down to a steady value

$$\boxed{\dot{x}_i = f_i(X)}$$

- Issue of Stability: Given a feedback network architecture we must ensure that the network dynamics leads to behavior that can be interpreted in a sensible way.

- Dynamical systems have variants of behavior like
  - fixed point equilibria where the system eventually converges to a fixed point
  - Chaotic dynamics where the system wanders aimless in state space

Summary of Major Neural Networks Models

Model	*Architecture* *Neuron Characteristic*	*Learning Algorithm*	*Application*
Perceptron	Single-node, feedforward Binary-threshold	Supervised, error-correction	Pattern classification
Adaline	Single-node, feedforward Linear	Supervised, gradient descent	Regression
Multilayer perceptron	Multilayered, feedforward nonlinear sigmoid	Supervised, gradient descent	Function approximation
Reinforcement learning	Multilayered Binary-threshold	Supervised reward-punishment	Control
Support vector machines	Multilayered kernel based, binary-threshold	Supervised quadratic optimization	Classification, regression
Radial basis function net	Multilayered distance based, linear	Supervised gradient descent	Interpolation, regression, classification
Hopfield network	Single layer, feedback Binary threshold/sigmoid	Outer product correlation	CAM, optimization

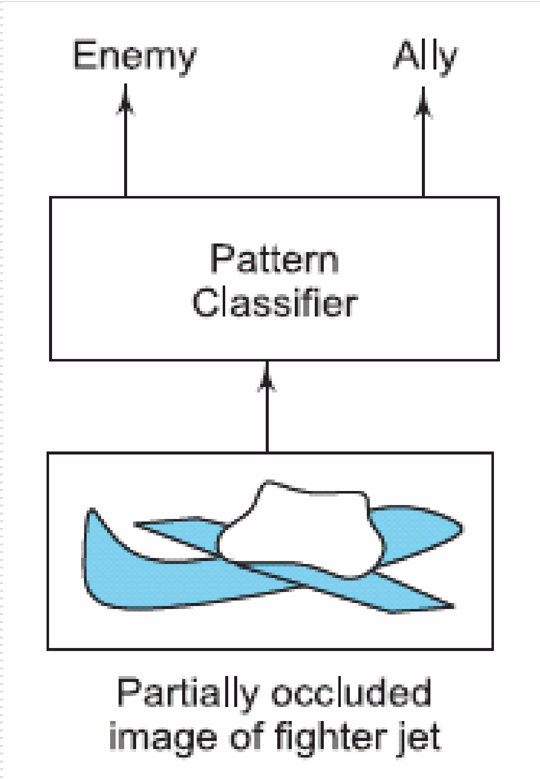
Summary of Major Neural Networks Models

Model	*Architecture* *Neuron Characteristic*	*Learning Algorithm*	*Application*
Boltzmann machine	Two layered, feedback Binary threshold	Stochastic gradient descent	Optimization
BSB	Single layered, feedback Linear threshold	Outer product correlation	Clustering
Bidirectional associative memory	Two layered, feedback Binary threshold	Outer product correlation	Associative memory
Adaptive resonance theory	Two layered Binary, faster-than-linear	Unsupervised competitive	Clustering, classification
Vector quantization	Single layer, feedback Faster than linear	Supervised, unsupervised competitive	Quantization, clustering
Mexican hat net	Single layer, feedback Linear threshold	None fixed weights	Activity clustering
Kohonen SOFM	Single layer Linear threshold	Unsupervised soft-competitive	Topological mapping
Pulsed neuron models	Single/multilayer Pulsed/IF neuron	None	Coincidence detection Temporal processing

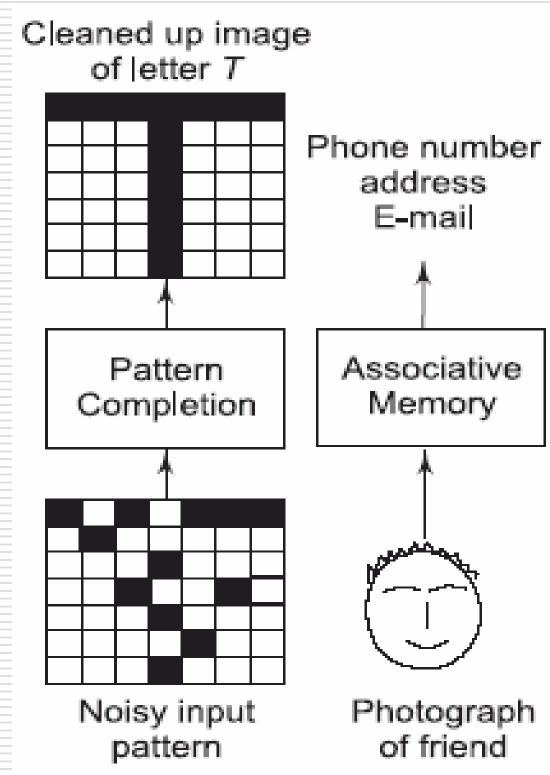
Salient Properties of Neural Networks

- Robustness Ability to operate, albeit with some performance loss, in the event of damage to internal structure.
- Associative Recall Ability to invoke related memories from one concept.
  - For e.g. a friend's name elicits vivid mental pictures and related emotions
- Function Approximation and Generalization Ability to approximate functions using learning algorithms by creating internal representations and hence not requiring the mathematical model of how outputs depend on inputs. So neural networks are often referred to as adaptive function estimators.

Application Domains of Neural Networks



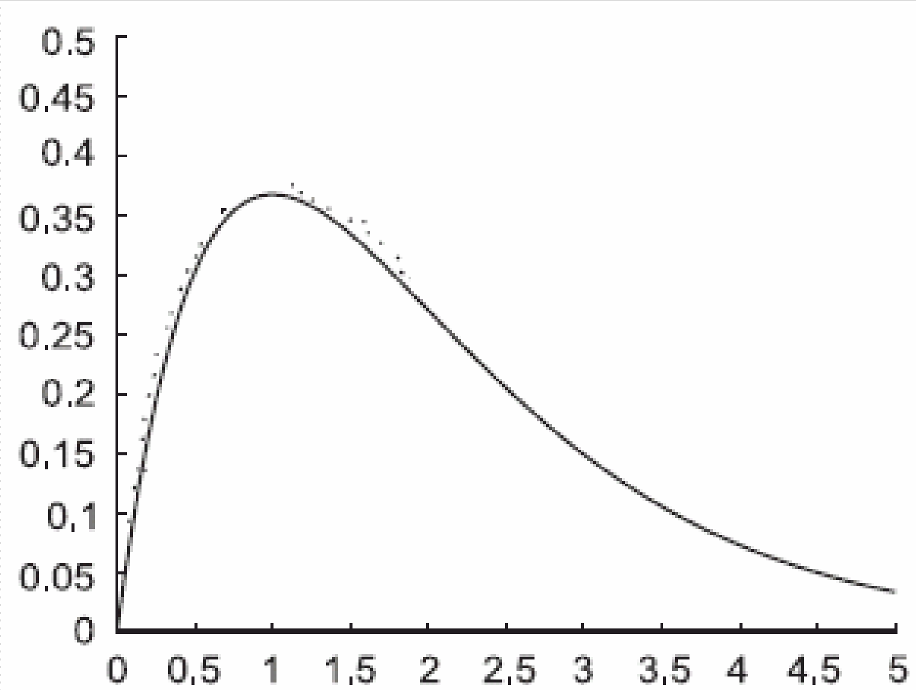
Fault Tolerance



Associative Recall

Application Domains of Neural Networks

Function Approximation



Prediction