Introduction to Deep Learning

Arijit Mondal
Dept. of Computer Science & Engineering
Indian Institute of Technology Patna
arijit@iitp.ac.in

Neural Network

Human brain vs von Neumann computer

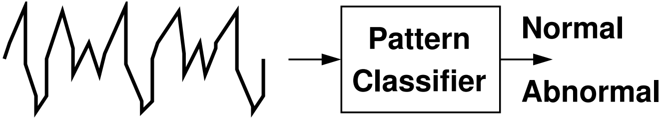
- Massive parallelism
- Distributed representation and computation
- Learning ability
- Generalization ability
- Adaptability
- Inherent contextual information processing
- Fault tolerance
- Low energy consumption

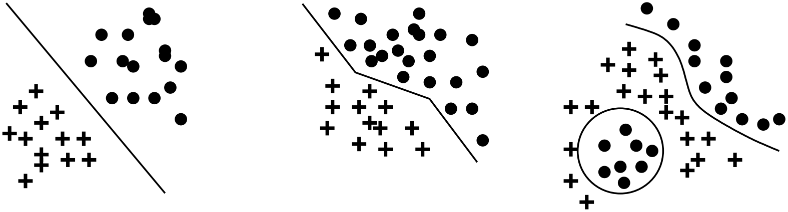
Computer vs Brain

	von Neumann	Neural system
Processor	Complex, high speed, one or a few	Simple, low speed, a large number
Memory	Separate from processor, Localized, Noncontent addressable	Integrated into processor, Distributed, Content addressable
Computing	Centralized, sequential, stored program	Distributed, parallel, self-learning
Reliability	Very vulnerable	Robust
Expertise	Numeric and symbolic manipulations	Perceptual problems
Operating environment	Well defined, well constrained	Poorly defined, unconstrained

Artificial Neuron: Applications

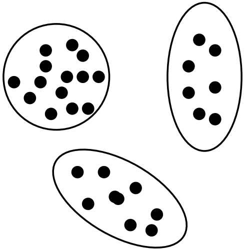
- **Pattern classification**





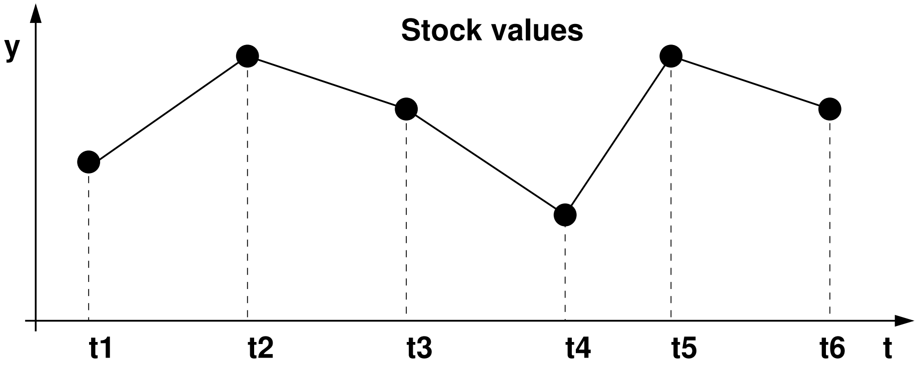
Artificial Neuron: Applications

- **Clustering/categorization**



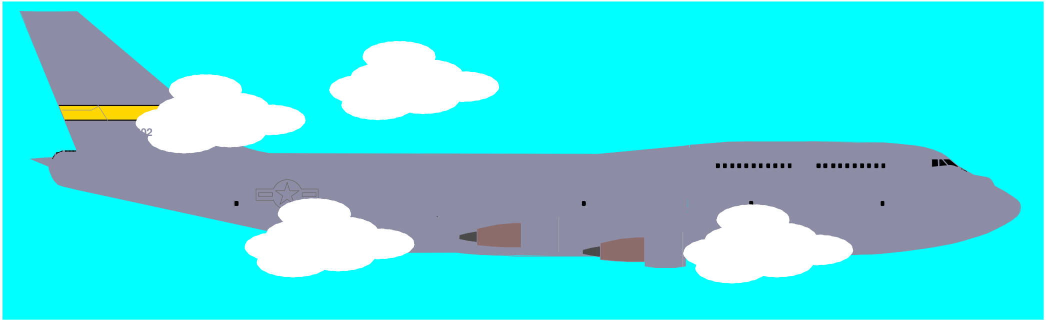
Artificial Neuron: Applications

- **Prediction**



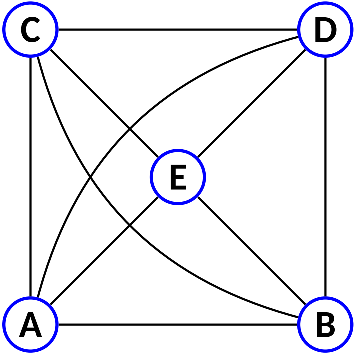
Artificial Neuron: Applications

- **Retrieval**



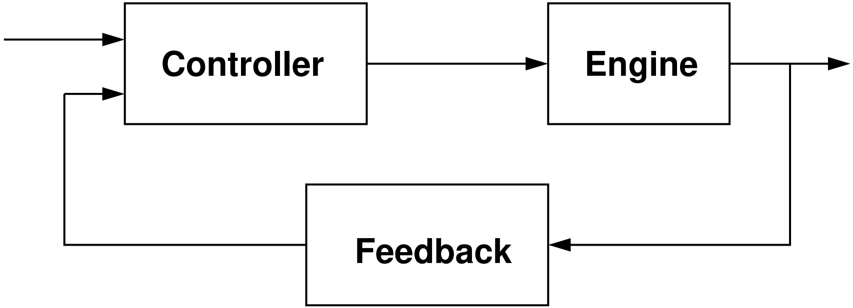
Artificial Neuron: Applications

- **Optimization**



Artificial Neuron

- Control



History

- Started in 1940s by McCulloch and Pitt
- Rosenblatt perceptron convergence theorem (1960)
- In 1980s ANN started gaining popularity
- Again became popular after 2006

Biological Neuron

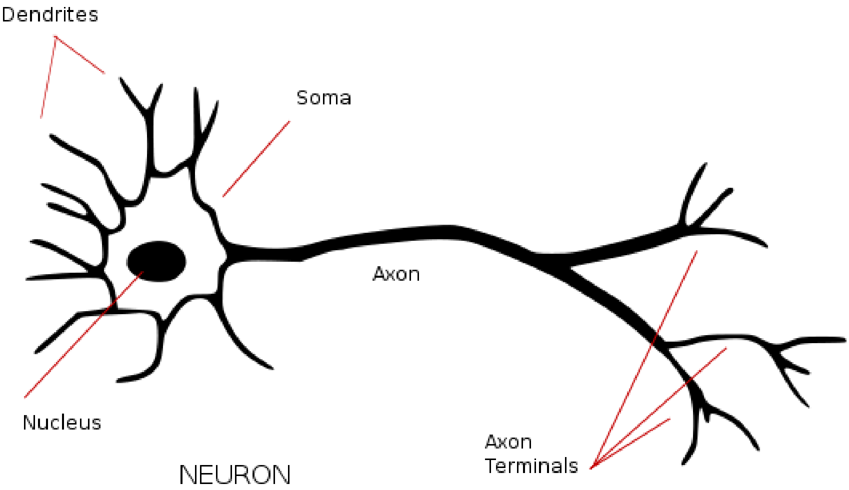


Image source: Internet

Cerebral cortex

- **It is a flat sheet of neurons about 2-3 millimeter thick with surface area is 2200 cm^2**
  - **Twice the area of computer keyboard**
- **It contains around 10^{11} neurons**
  - **Number of stars in the Milky-way**
- **Each neuron is connected to 10^3-10^4 other neurons**
- **Total connections is around 10^{14}-10^{15}**
- **Connectionist model**

Human brain

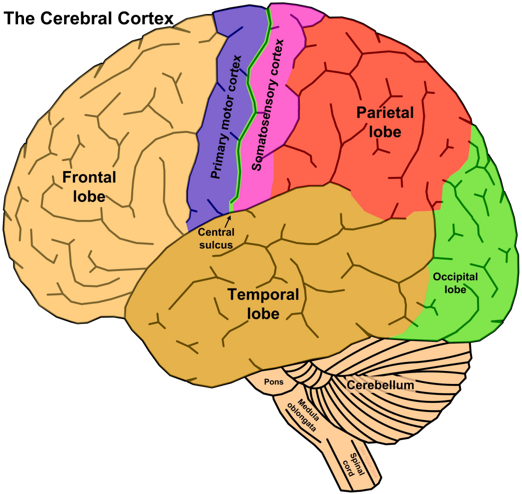
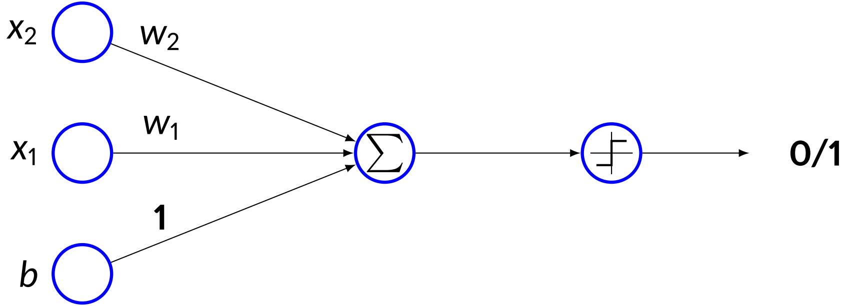


Image source: Internet

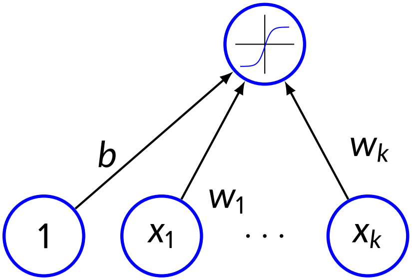
Neuron

- **One of the primitive models**

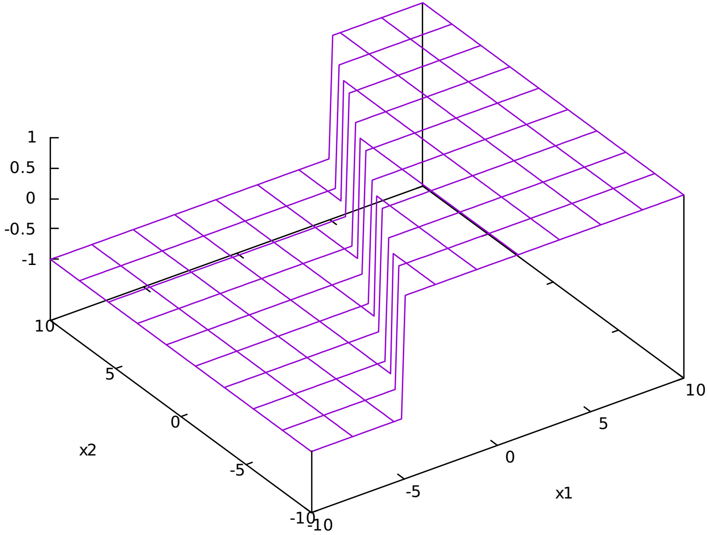


Artificial Neuron

- **Neuron pre-activation function**
  - $a(\mathbf{x}) = \sum_i w_i x_i + b = b + \mathbf{w}^T \mathbf{x}$
- **Neuron output activation function**
  - $h(\mathbf{x}) = g(a(\mathbf{x})) = g\left(\sum_i w_i x_i + b\right)$
- **Notations**
  - $\mathbf{w}$ — **Weight vector**
  - b — **Neuron bias**
  - $g(.)$ — **Activation function**

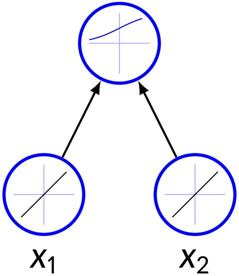


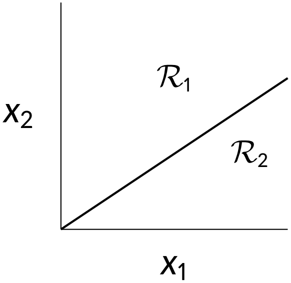
Physical interpretation



Classification using single neuron

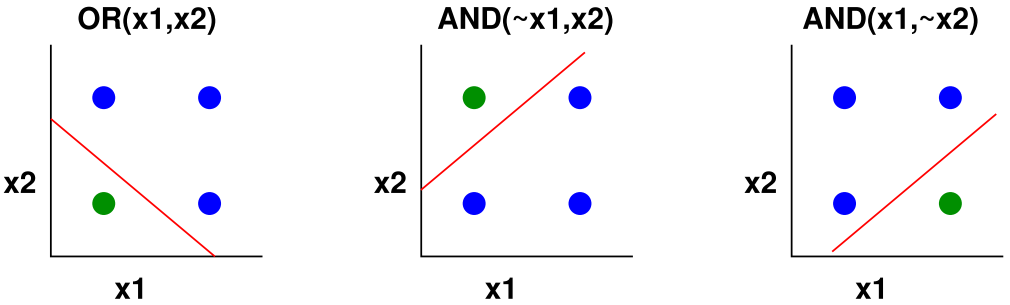
- **Single neuron can do binary classification**
  - **Also known as logistic regression classifier**





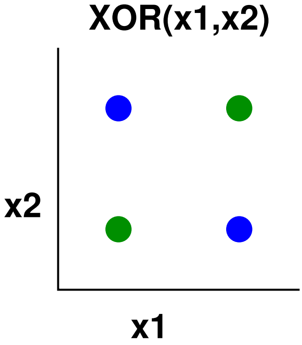
Artificial neuron

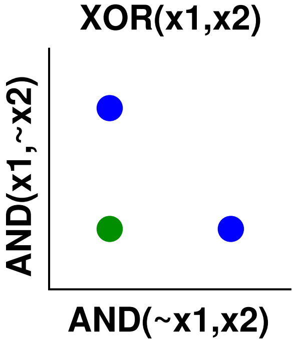
- Can solve linearly separable problems



Artificial neuron: XOR problem

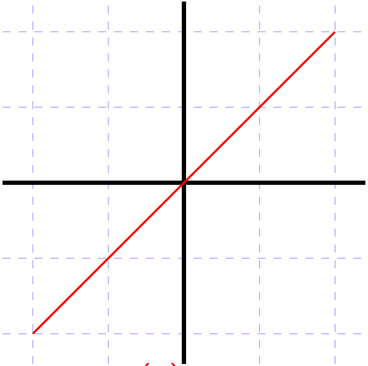
- There are issues for linear separation





Activation function

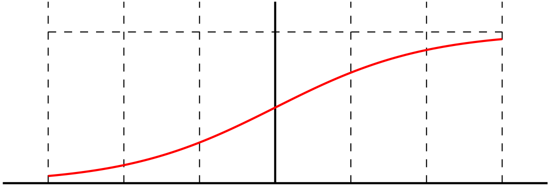
- **Linear activation function**
  - **Not very interesting**
  - **No change in values**
  - **Huge range**



$g(x) = x$

Activation function

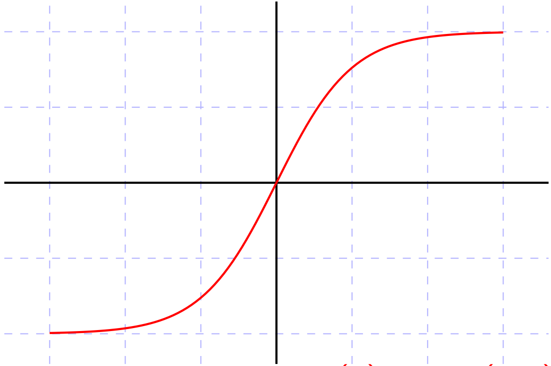
- **Sigmoid function**
  - **Values lie between 0 and 1**
  - **Strictly increasing function**
  - **Bounded**



$$g(x) = \mathbf{sigm}(x) = \frac{1}{1 + \exp(-x)}$$

Activation function

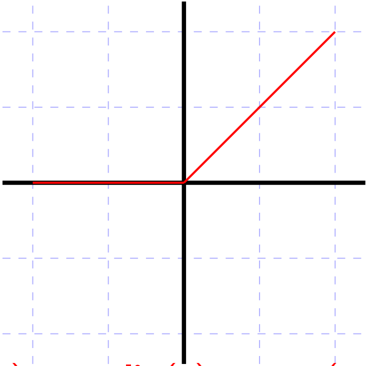
- **Hyperbolic Tangent (Tanh) function**
  - **Can be positive or negative**
  - **Values lie between -1 and 1**
  - **Strictly increasing function**
  - **Bounded**



$$g(x) = \mathbf{tanh}(x) = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

Activation function

- **Rectified linear activation function**
  - **Bounded below by 0**
  - **Strictly increasing function**
  - **Not upper bounded**



$$g(x) = \textbf{reclin}(x) = \max(0, x)$$

Single hidden layer neural network

- **Hidden layer pre-activation**

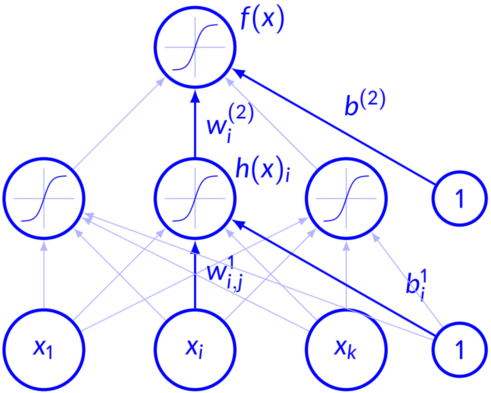
$$a(\mathbf{x}) = \mathbf{b}^1 + \mathbf{w}^1\mathbf{x}$$

- **Hidden layer activation**

$$h(\mathbf{x}) = g(a(\mathbf{x}))$$

- **Output layer activation**

$$f(\mathbf{x}) = o(b^{(2)} + \mathbf{w}^{(2)T}h^1(\mathbf{x}))$$



Multi layer neural network

- **Pre-activation in layer**

$$k > 0 \ (\mathbf{h}^{(0)}(\mathbf{x}) = x)$$

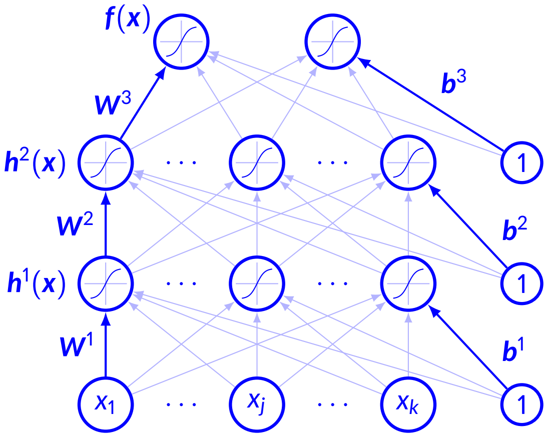
$$\mathbf{a}^{(k)}(\mathbf{x}) = \mathbf{b}^{(k)} + \mathbf{W}^{(k)}\mathbf{h}^{(k-1)}\mathbf{x}$$

- **Hidden layer activation**

$$\mathbf{h}^{(k)}(\mathbf{x}) = \mathbf{g}(\mathbf{a}^{(k)}(\mathbf{x}))$$

- **Output layer activation**

$$\mathbf{h}^{(L+1)}(\mathbf{x}) = o(\mathbf{a}^{(L+1)}(\mathbf{x})) = \mathbf{f}(\mathbf{x})$$



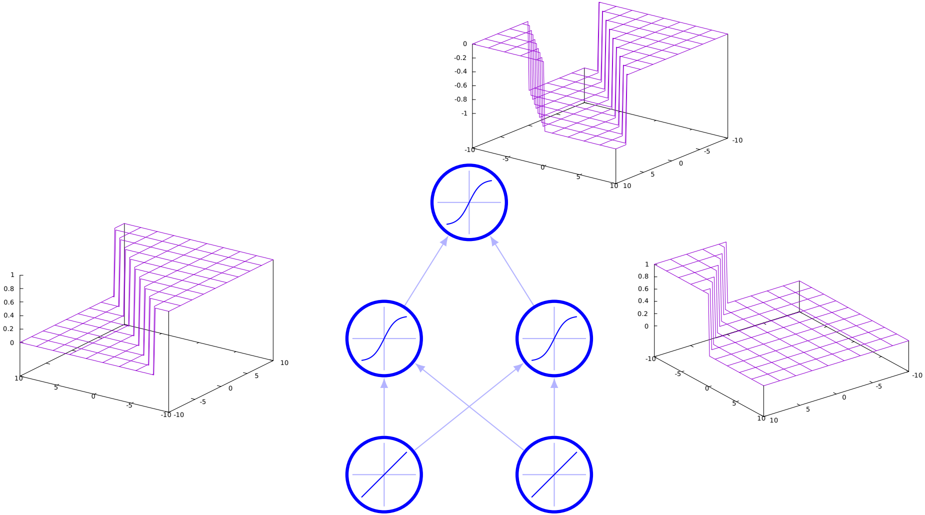
Multiclass classification

- **Need multiple outputs that is one neuron for each class**
- **Need to determine probability of** $p(y = c|\mathbf{x})$
- **Softmax activation function is used at the output**

$$\boldsymbol{o}(\boldsymbol{a}) = \mathbf{softmax}(a) = \left[\frac{\exp(a_1)}{\sum_c \exp(a_c)} \quad \frac{\exp(a_2)}{\sum_c \exp(a_c)} \quad \dots \quad \frac{\exp(a_c)}{\sum_c \exp(a_c)} \right]^T$$

  - **Strictly positive**
  - **Sum to 1**
  - **Class having the highest probability will be the predicted output**

Capacity of neural network



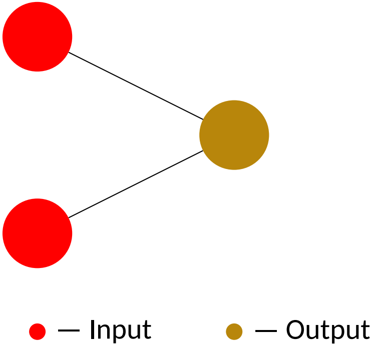
Capacity of neural network

- **Universal approximation theorem (Hornik,1991)**
  - **A single hidden layer neural network with a linear output unit can approximate any continuous function arbitrarily well, given enough hidden units.**
- **The result is applicable for other hidden layer activation functions such as sigmoid, tanh, etc.**
- **This is a promising result, but it does not say that there is a learning algorithm to find the necessary parameter values!**

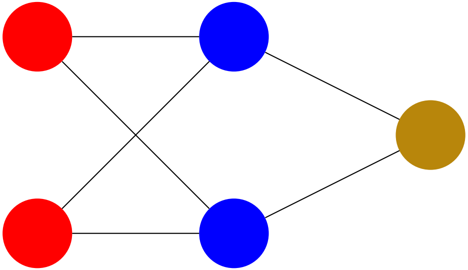
Types of Neural Network

- **Feed forward neural network**
- **Radial basis function network**
- **Recurrent neural network**
- **Boltzmann machine**
- **Long short term memory network**
- **and many more**

Perceptron



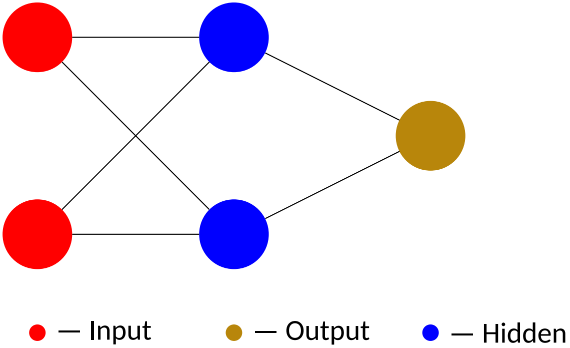
Feed Forward



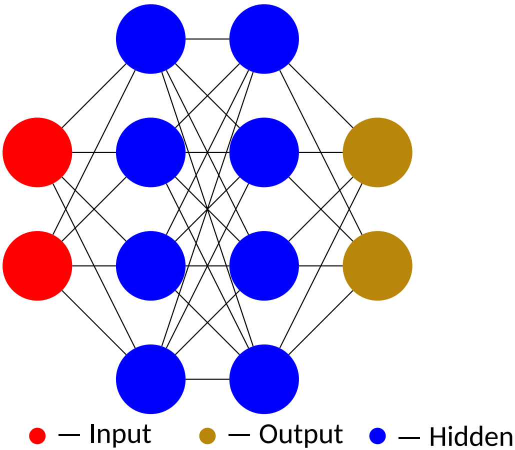
● — Input ● — Output ● — Hidden

Radial Basis Function

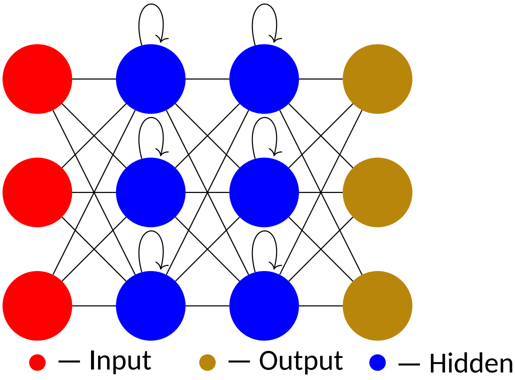
- **Typically it will have 3 layers**
- **Distance from a center vector is computed**



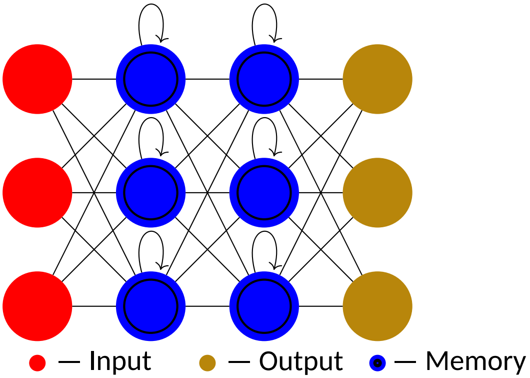
Deep Feed Forward



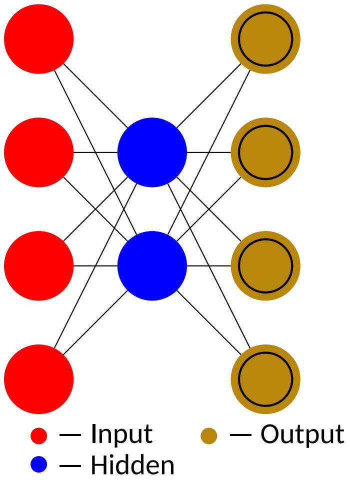
Recurrent Neural Network



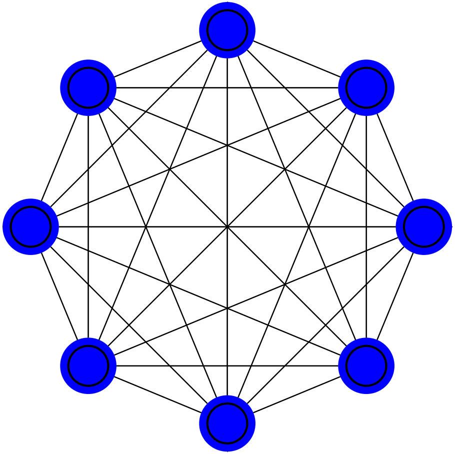
Long Short Term Memory



Auto Encoder

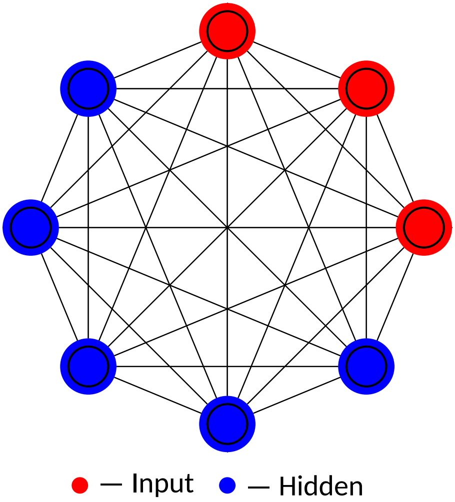


Markov chain



— Memory

Boltzmann Machine



Learning the parameters

- The network must learn the connection weights from available training examples
- Learning can be
  - Supervised
  - Unsupervised
  - Hybrid
- Four basic types of learning rule
  - Error correction rule
  - Boltzmann learning
  - Hebbian
  - Competitive learning

Error correction rule

- **Output is generated based on the weight values but this may vary from desired value**
- **The error information is used to update the weight value**
- **Perceptron learning algorithm**
  - **Initialize the weights and threshold to small random numbers**
  - **Present a pattern vector and evaluate the output of neuron**
  - **Update the weight according to** $w_j(t+1) = w_j(t) + \eta(d - y)x_j$
- **Back propagation algorithm**

Boltzmann learning

- Usually symmetric recurrent network consisting of binary units
- A subset of neurons interact with environment
- Generally it has two modes
  - Clamped — Visible neurons are clamped to specific states
  - Free-running - Visible and hidden unit operate freely
- Stochastic learning rule derived from information theoretic and thermodynamic principles
- Learning rule is given by $\Delta w_{ij} = \eta(\bar{\rho}_{ij} - \rho_{ij})$

Hebbian rule

- **One of the oldest learning rules**
- **If neuron on both sides of a synapse are activated synchronously and repeatedly, the synapse's strength is selectively increased**
- **Mathematically, it can be described as** $w_{ij}(t+1) = w_{ij}(t) + \eta y_j(t) x_i(t)$

Competitive learning rule

- **Output units compete among themselves for activation**
- **Only one output is active at time**
- **Also known as winner-take-all**
- **Mathematically, it can be represented as** $\mathbf{w}_{i^*}\mathbf{x} \geq \mathbf{w}_i\mathbf{x}$
- **Competitive learning rule can be stated as**

$$\Delta w_{ij} = \begin{cases} \eta(x_j^u - w_{i^*j}) & i = i^* \\ 0 & i \neq i^* \end{cases}$$

Summary

- Error correction rule — Single or multilayer perceptron
  - Pattern classification, function approximation, prediction, control
- Boltzmann — Recurrent
  - Pattern classification
- Hebbian — Multilayer feed forward
  - Pattern classification, data analysis
- Competitive
  - Within class categorization, data compression