Deep Learning

Vijaya Saradhi

IIT Guwahati

Tue, 08^{th} Sept 2020

Data & Learning Methods

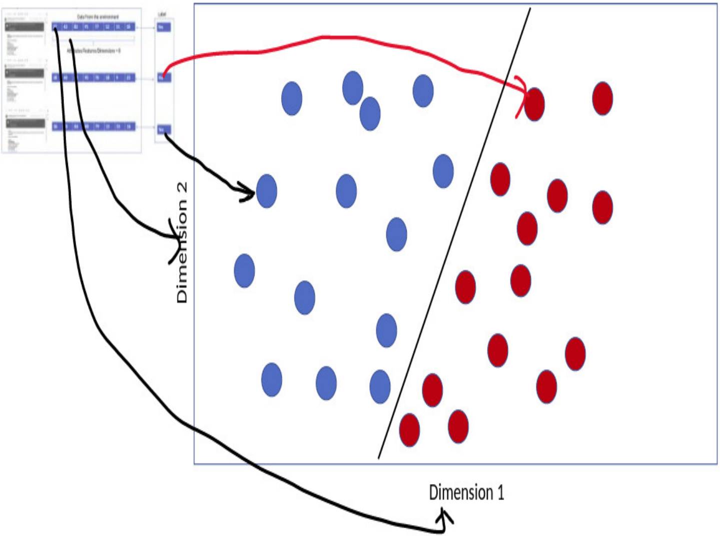
Data

- Linearly separable data
- Linearly non-separable data
- Noise in the data
- Erroneous data

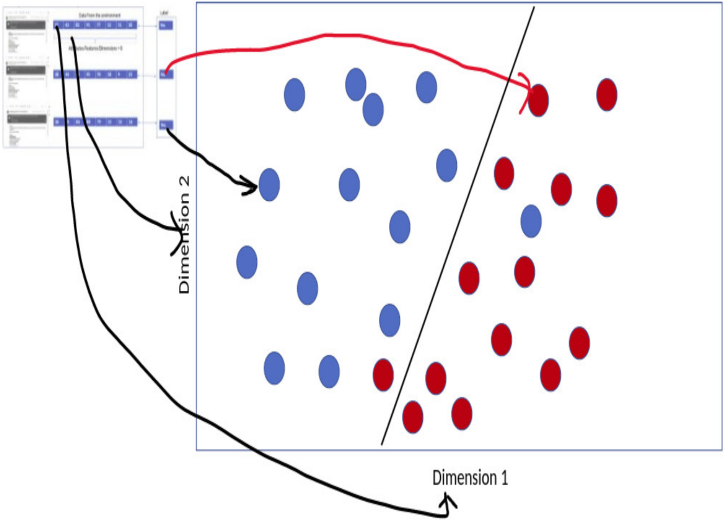
Learning Methods

- Linearly separable data built with guarantees example of perceptron
- Linearly non-separable data multi-layer perceptron

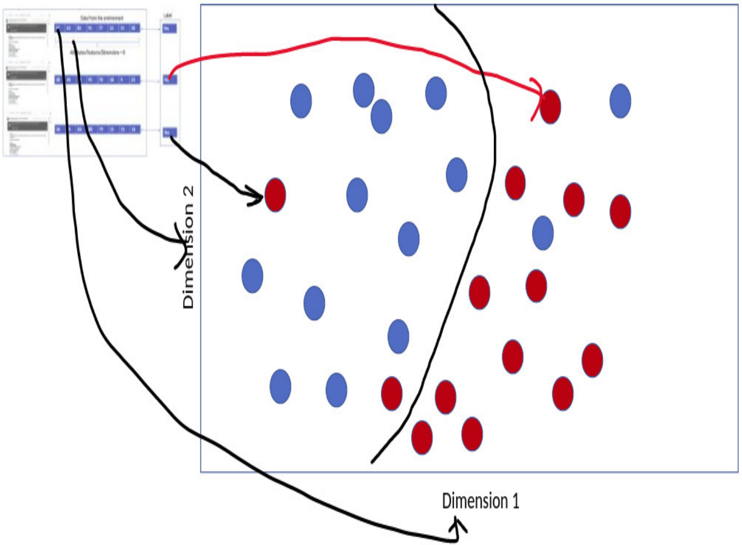
Linearly separable data - example



Linearly non-separable data - example



Noisy data - example



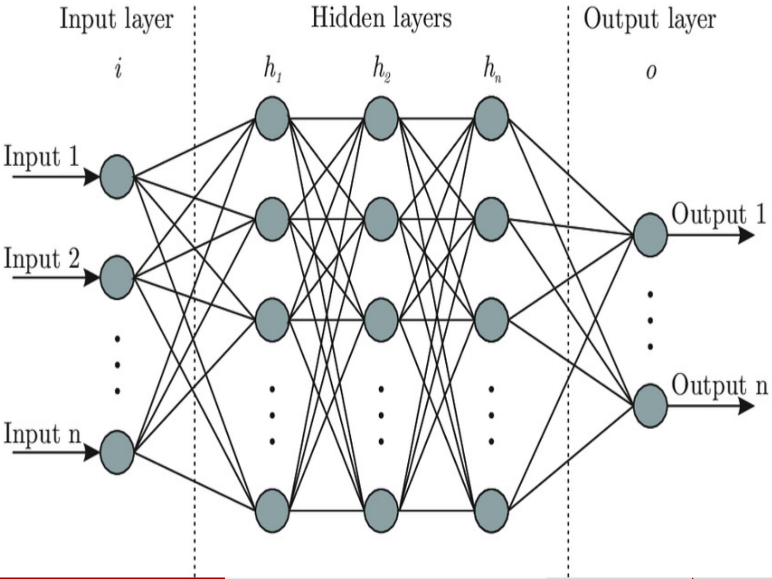
Neural Networks

Definition

A neural network is

- massively parallel
- parallel
- distributed processor
- with simple processing units neurons
- Propensity to store experiential knowledge
- Apply it when required

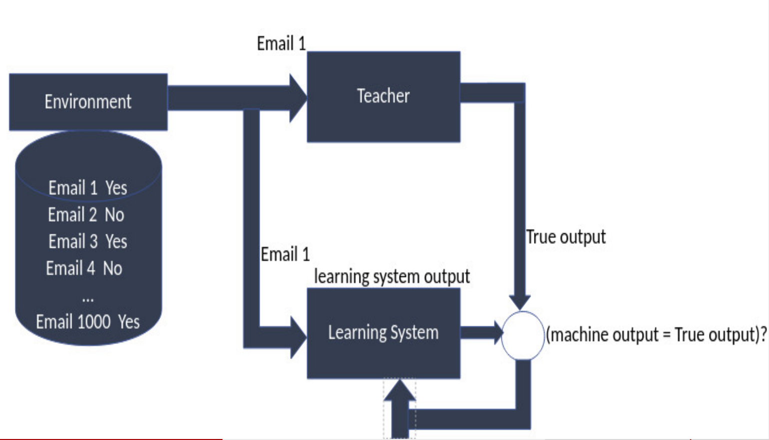
An Example Network



Neural Networks

Resembles the Human Brain

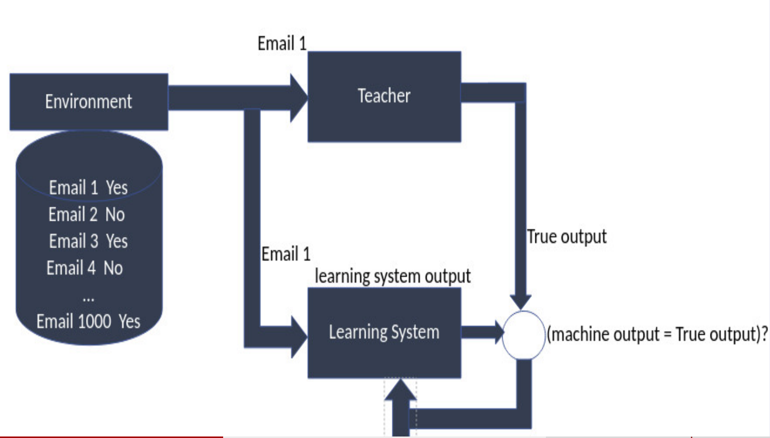
Knowledge is acquired by the network from its environment through a learning process



Neural Networks

Resembles the Human Brain

Inter-neuron connections strengths (weights) are used to store the acquired information



Models Of A Neuron

Constituent Elements

- An information processing unit
- Fundamental to the operation of a complex neural network

Constituent Elements

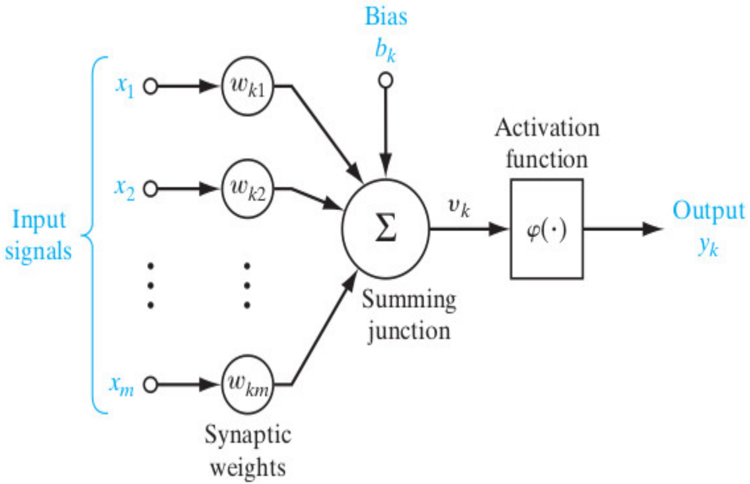
Connecting links Or Synapses or connections

Adder function for summing input signals weighted by the synaptic strengths

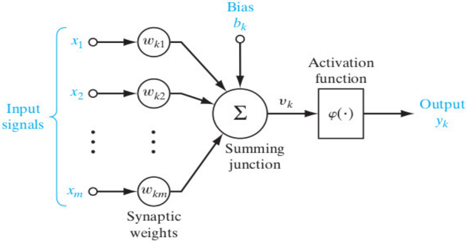
Activation function for limiting the output of a neuron

Range Output range [0, 1] or [-1, 1]

Neuron Model

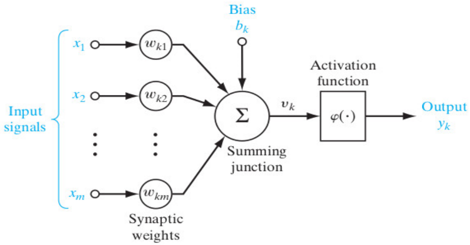


Neuron Model



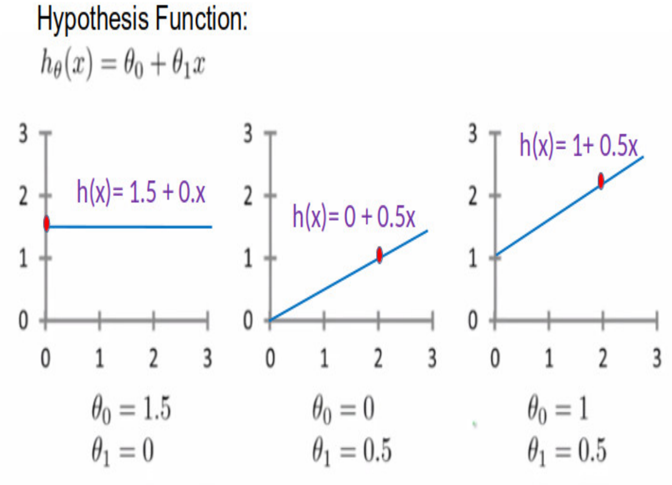
- Connecting links carrying weights
- Adder function Σ
- Activation function $\phi(.)$

Neuron Model

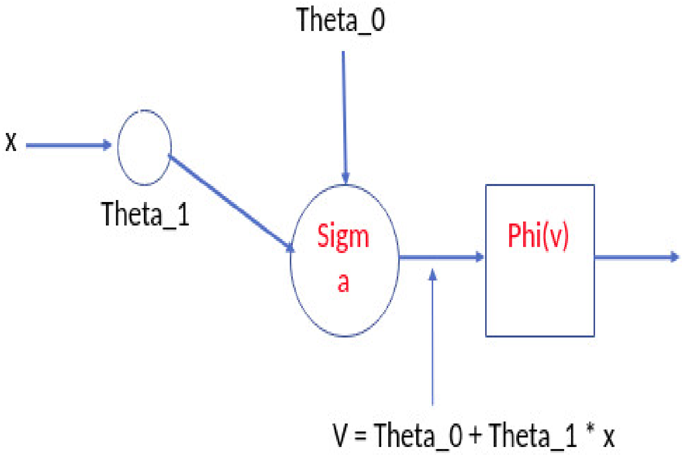


- Neuron having m inputs $x_1, x_2, \cdots, x_m$
- Adder function
$$u_k = \left(\sum_{j=1}^{m} w_{kj} x_j \right)$$
- Add Bias $v_k = (u_k + b_k)$
- Add Bias
$$v_k = \left(\sum_{j=1}^{m} w_{kj} x_j \right) + b_k$$
- Activation function
$$\phi \left(\left(\sum_{j=1}^{m} w_{kj} x_j \right) + b_k \right)$$

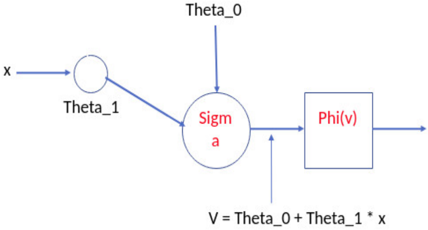
Neuron Model - One input example



Neuron Model - One input example

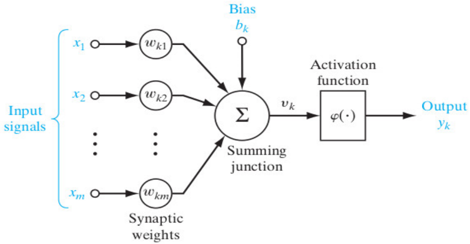


Neuron Model



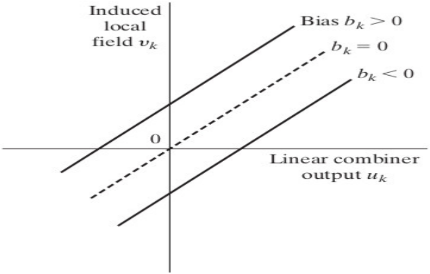
- Neuron having $m = 1$ input x
- Adder function
  $$u_k = \left(\sum_{j=1}^{m} w_{k1} x \right) = w_{k1} \times x$$
- Add Bias $v_k = (u_k + b_k)$
- Add Bias $v_k = (w_{k1} \times x) + b_k$
- Activation function
  $\phi\left(w_{k1} \times x + b_k\right)$
- Equivalent of saying:
  $\phi\left(\theta_1 x + \theta_0\right)$

Neuron Model with two inputs



- Neuron having $m = 2$ inputs x_1, x_2
- Adder function $u_k = (w_{k1}x_1 + w_{k2}x_2)$
- Add Bias $v_k = (u_k + b_k)$
- Add Bias $v_k = (w_{k1}x_1 + w_{k2}x_2) + b_k$
- Activation function $\phi((w_{k1}x_1 + w_{k2}x_2) + b_k)$

Influence of bias value

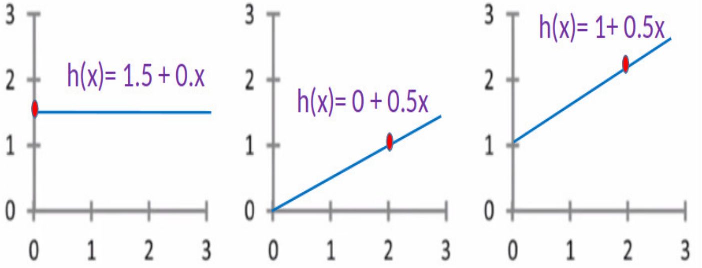


- $b_k = 0$
- Add Bias
  $v_k = (w_{k1}x_1 + w_{k2}x_2) + 0$
- $b_k = +ve$
- Add Bias
  $v_k = (w_{k1}x_1 + w_{k2}x_2) + b_k$
- $b_k = -ve$
- Add Bias
  $v_k = (w_{k1}x_1 + w_{k2}x_2) - b_k$
- $(w_{k1}x_1 + w_{k2}x_2)$ vary along y-axis explaining the intercept and bias

Bias Impact

Hypothesis Function:

$$h_\theta(x) = \theta_0 + \theta_1 x$$



$\theta_0 = 1.5$, $\theta_1 = 0$ | $\theta_0 = 0$, $\theta_1 = 0.5$ | $\theta_0 = 1$, $\theta_1 = 0.5$

Neuron Including Bias

Modified Equations

$$v_k = \sum_{j=0}^{m} w_{kj} x_j$$

$$w_{k0} = b_k$$

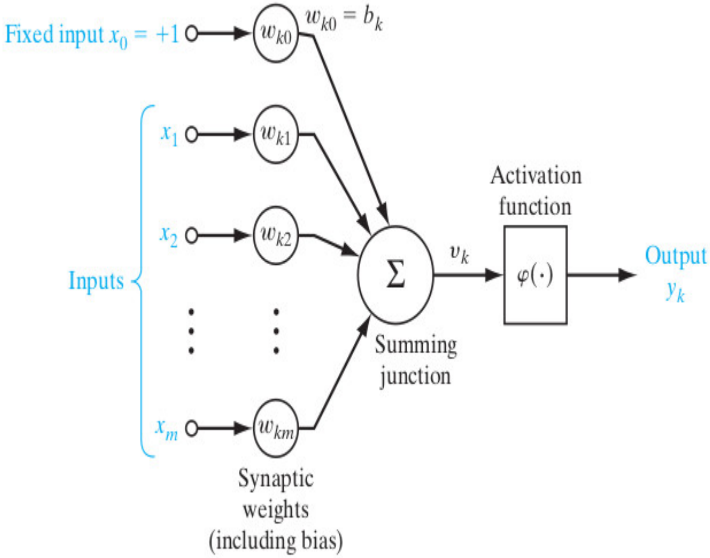
$$x_0 = +1$$

Modified Equations

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + w_{k0} x_0$$

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k$$

Modified Neuron Model



Types of activation functions

Types

- Threshold function
- Sigmoid function
- Signum function

Threshold activation function

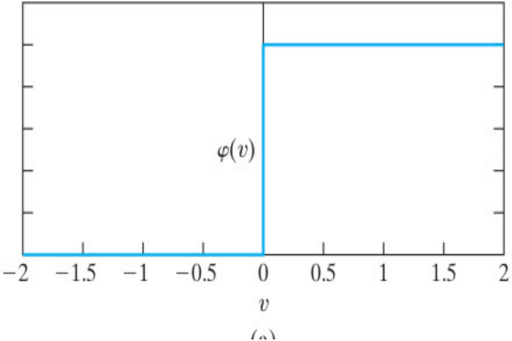
Threshold function

$$\phi(v) = \begin{cases} 1 & \text{if v} \geq 0 \\ 0 & \text{if } v < 0 \end{cases}$$

k^{th} Neuron Output

$$y_k = \begin{cases} 1 & \text{if } v_k \geq 0 \\ 0 & \text{if } v_k < 0 \end{cases}$$

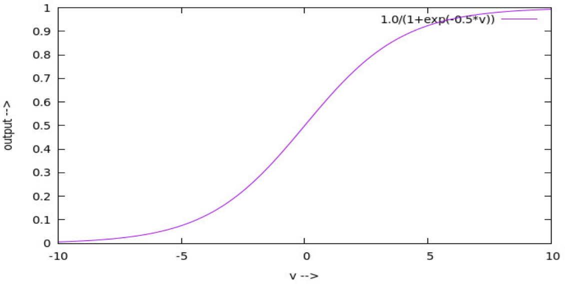
Threshold Activation Function



Sigmoid function

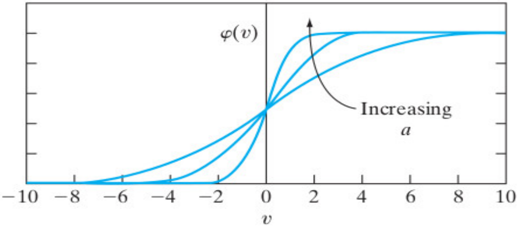
Sigmoid function

$$\phi(v) = \frac{1}{1 + \exp^{(-a \times v)}}$$



Sigmoid function

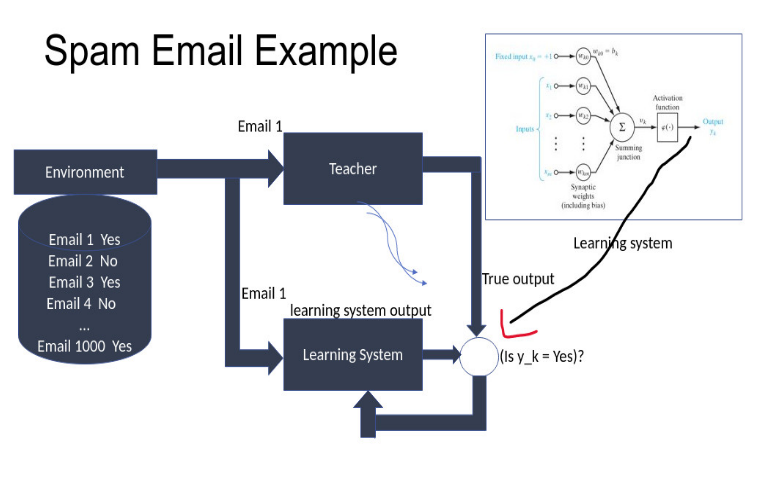
Slope parameter effect



(b)

Supervised learning

General procedure



Signum functions

Slope parameter effect

- Threshold and sigmoid function output ranges between [0, 1]
- To modified the limit as [-1, $+1$] modify the activation function as signum function

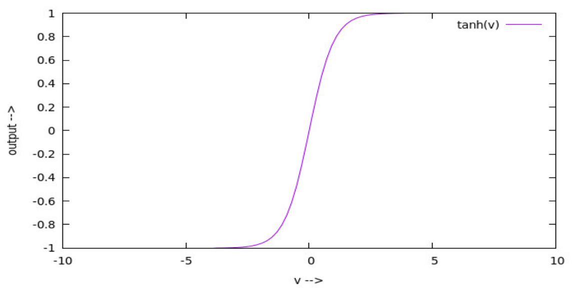
Signum function

$$\phi(v) = \begin{cases} +1 & \text{if } v > 0 \\ 0 & \text{if } v = 0 \\ -1 & \text{if } v < 0 \end{cases}$$

Sigmoid function

tanh function

output is in the range [-1, 1]



Neural Networks as Directed Graphs

Directed Graphs

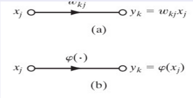
- Consists of links and nodes
- A node has associated signal x_j
- A directed link orignates at node j and terminates at node k
- links are of two types
  - Synaptic links
  - Activation links

Neural Networks as Directed Graphs

Rules

Rule 1 A signal flows along a link only in one direction (arrow decides the flow)

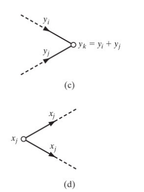
- **Synaptic links** Node signal x_j is multiplied by weight w_{kj} to produce node signal y_k
- **Activation links** This links behavior is governed by activation function $\phi(.)$



Neural Networks as Directed Graphs

Rules

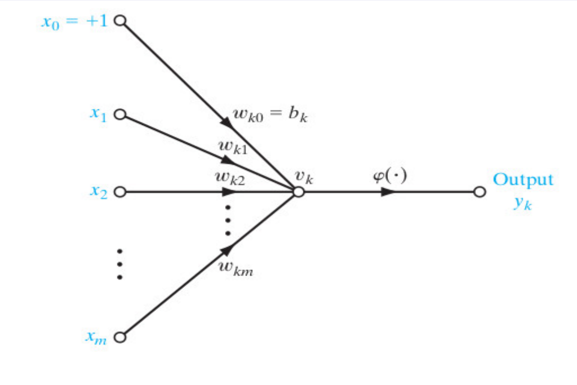
Rule 2 A node signal equal to the sum of all signals entering the node



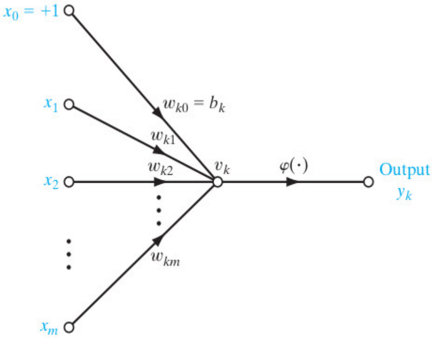
Rule 3 Signal at node is transmitted to each out going link with the same signal

Neuron Example as Directed Graphs

Neuron Model



Neuron Model - Directed Graph



- Rule 1 synaptic link: $x_0 \times w_{k0}$
- Rule 1 synaptic link: Second link: $x_1 \times w_{k1}$
- Rule 1 synaptic link: m^{th} link: $x_m \times w_{km}$
- Rule 2: Node v_k: $x_0 \times w_{k0} + x_1 \times w_{k1} + \cdots + x_m \times w_{km}$
- Rule 1: activation link between node v_k and y_k
- Rule 1: activation link:
$$y_k = \phi\left(\sum_{j=1}^{m} w_{kj} x_j\right)$$

Neural Network Architectures

Types

- Single-layer feedforward networks
- Multi-layer feedforward networks
- Recurrent networks

Single layer feedforward networks

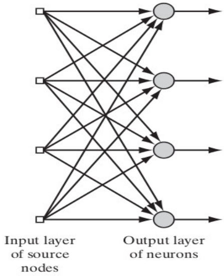


FIGURE 15 Feedforward network with a single layer of neurons.

- Input layer
- Output layer
- Each node is a neuron model
- The arrow emerging out of single node is the output of the neuron model (y_k)

Single layer feedforward networks

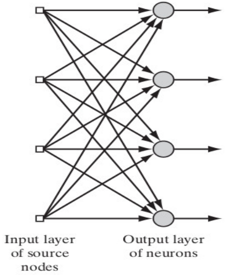
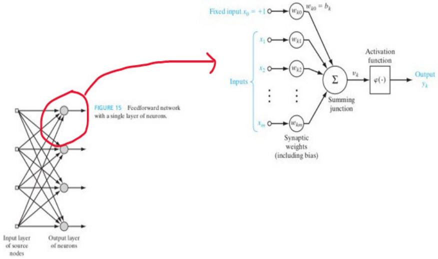


FIGURE 15 Feedforward network with a single layer of neurons.



Single layer feedforward networks

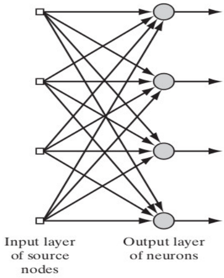


FIGURE 15 Feedforward network with a single layer of neurons.

- Let the inputs be: $x_1, x_2, \cdots, x_m$
- Let the weights on the first neuron be:

  $w_{11}, w_{12}, w_{13}, \cdots, w_{1m}$
- Let the weights on the second neuron be:

  $w_{21}, w_{22}, w_{23}, \cdots, w_{2m}$
- Output of the first neuron will be: $y_1 = \phi\left(\sum_{j=0}^{m} w_{1j} x_j\right)$
- Output of the second neuron will be: $y_2 = \phi\left(\sum_{j=0}^{m} w_{2j} x_j\right)$

Single layer feedforward networks

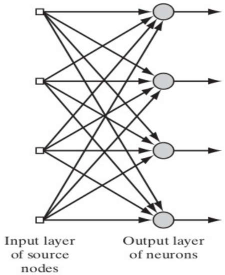


FIGURE 15 Feedforward network with a single layer of neurons.

- Network is feed forward as the inputs and weigths are passing along the direction of the arrows of the network in one direction
- One example of the environment is presented to this network
- Known quantities:
  - One input example (one spam email and its assocaited features) that is $x_{i1}, x_{i2}, \cdots, x_{im}$
  - Input examples class label: d_i

Single layer feedforward networks

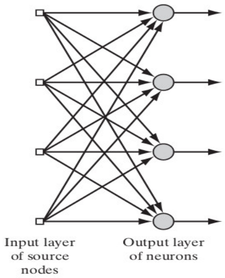
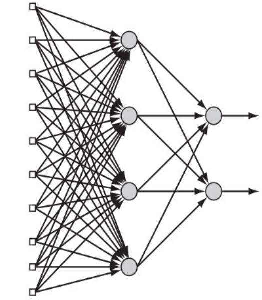


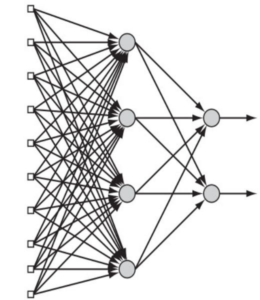
FIGURE 15 Feedforward network with a single layer of neurons.

- What is to be learned?
  - Weights for first neuron: $w_{11}, w_{12}, w_{13}, \cdots, w_{1m}$
  - Weights for second neuron: $w_{21}, w_{22}, w_{23}, \cdots, w_{2m}$
  - Weights for the last neuron: $w_{l1}, w_{l2}, w_{l3}, \cdots, w_{lm}$

Multi layer feedforward networks

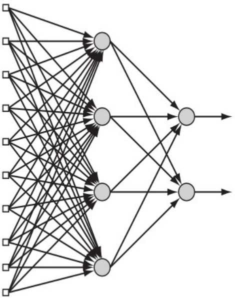


Multi layer feedforward networks



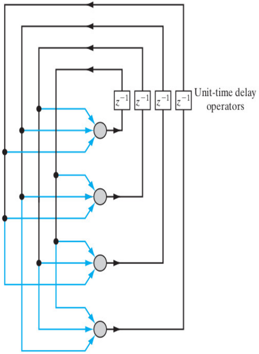
- Input layer, number of hidden layers and output layer
- Architecture is referred as: $m - h_1 - h_2 - q$
- m input features; h_1 hidden units in the first layer
- h_2 hidden units in the second layers and q-output nodes
- First layers is the input layer; last layer is the output layer

Multi layer feedforward networks



- Computation at the first node of the output layer:
- $y_{21} = \phi\left(\sum_{j=0}^{4} y_{1j} w_{2j}\right)$
- Output depends on the chosen activation function
- Input to the output layers is the 1st hidden layer
- Let its outputs are denoted as $y_{11}, y_{12}, y_{13}, y_{14}$
- The inputs in the 1st hidden layer are multiplied with the weights on the synaptic links going out of the first hidden

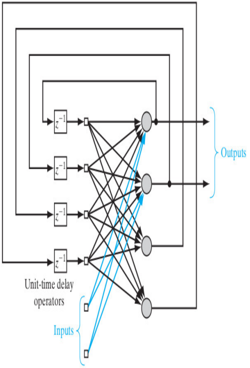
Recurrent networks



- Recurrent with no hidden layer
- Contains at least one feedback loop
- First neuron output is fed to rest of the three neurons
- Second neuron output is fed to rest of the other three neurons

Recurrent networks

with one hidden layer



Modern Architectures

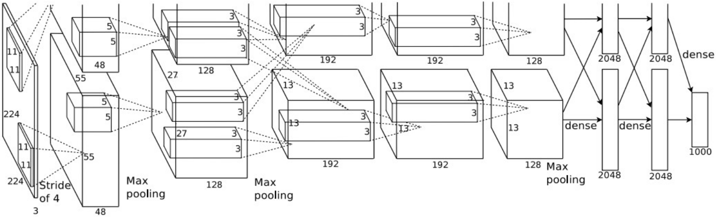


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264–4096–4096–1000.

Knowledge Representation

Four main points

- Rule 1 Similar inputs from similar classes should produce similar representations inside the network
- Rule 2 Inputs to be categorized as separate classes should be given widely different representation in the network
- Rule 3 Importance to specific features is given throguh involving large number of neurons
- Rule 4 Prior information is achieved through design of neural network.