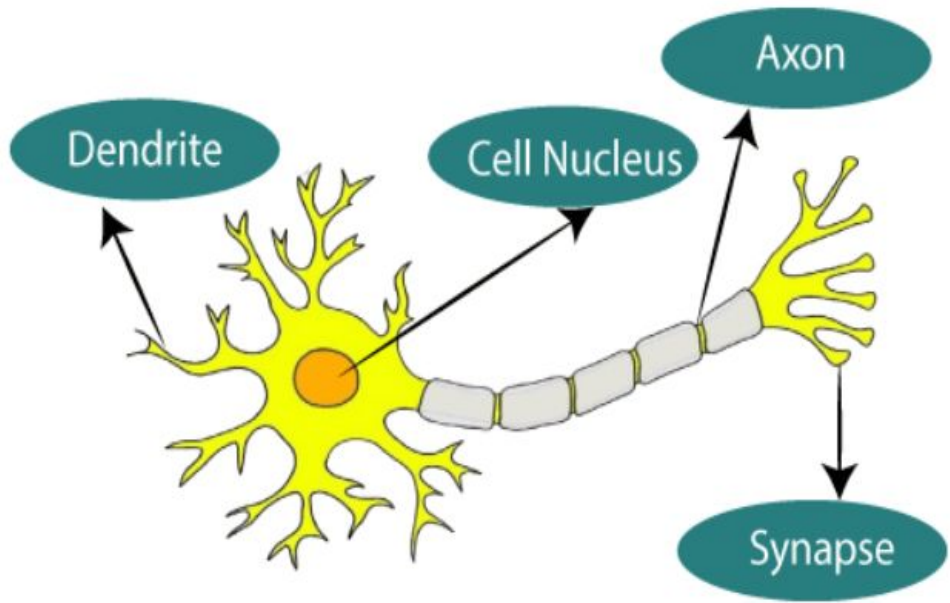


Artificial Neural Network

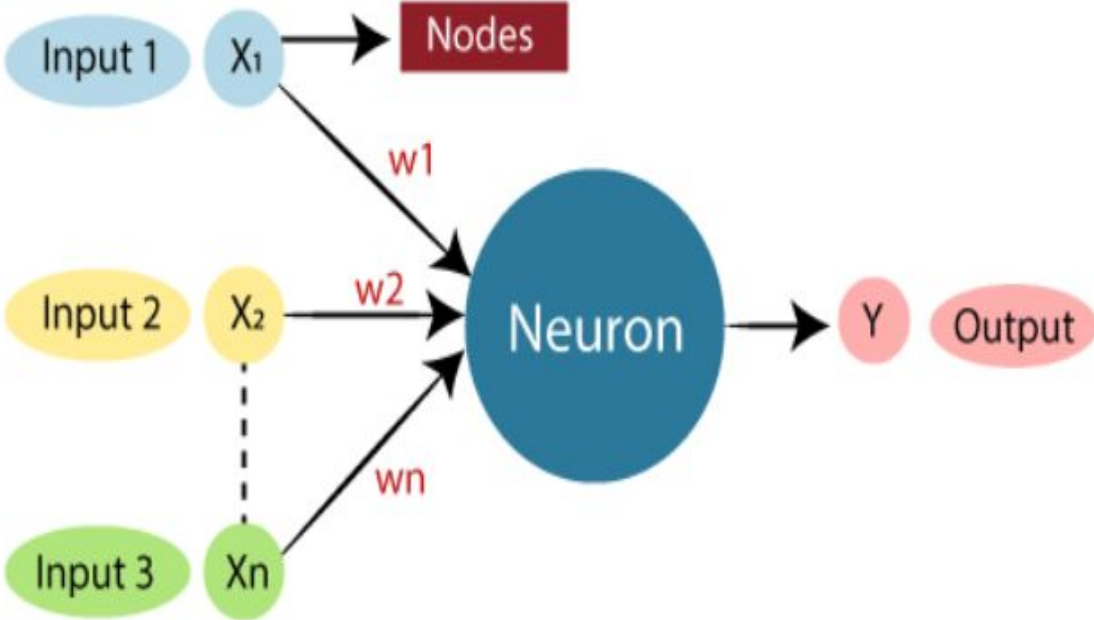
Artificial Neural Network

- The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain.
- Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks.
- These neurons are known as nodes.

Typical diagram of Biological Neural Network.



Typical Artificial Neural Network



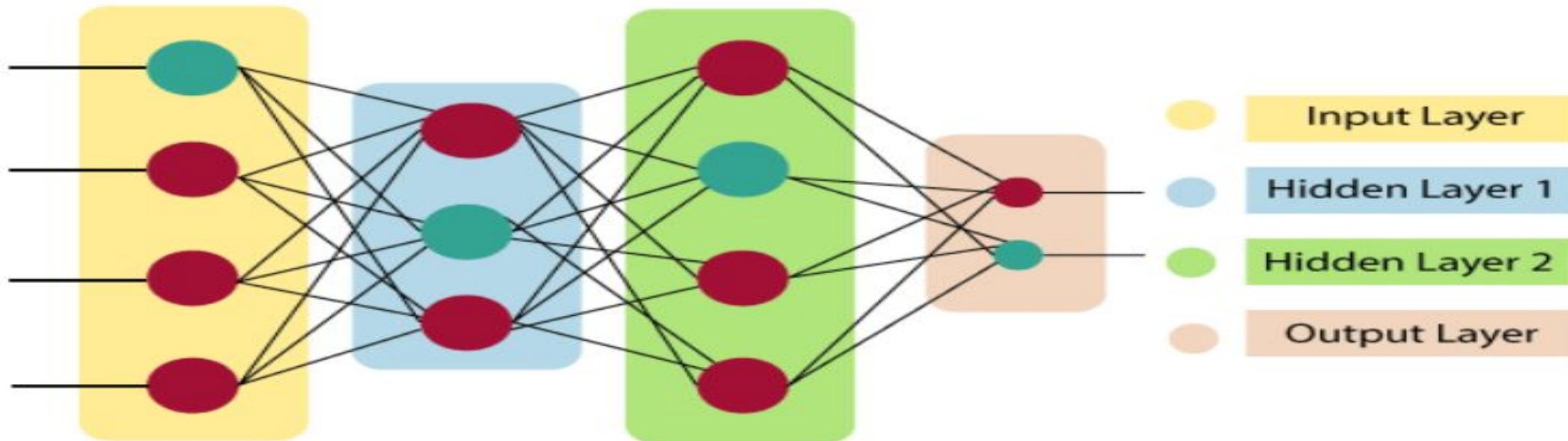
Relationship between Biological neural network and artificial neural network:

Biological Neural Network	Artificial Neural Network
Dendrites	Inputs
Cell nucleus	Nodes
Synapse	Weights
Axon	Output

- An Artificial Neural Network in the field of Artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner.
- The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.
- There are around 1000 billion neurons in the human brain. Each neuron has an association point somewhere in the range of 1,000 and 100,000.
- In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data when necessary from our memory parallelly. We can say that the human brain is made up of incredibly amazing parallel processors.

The architecture of an artificial neural network:

- To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of.
- In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers.
- Artificial Neural Network primarily consists of three layers:



Input Layer:

As the name suggests, it accepts inputs in several different formats provided by the programmer.

Hidden Layer:

The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns.

Output Layer:

The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

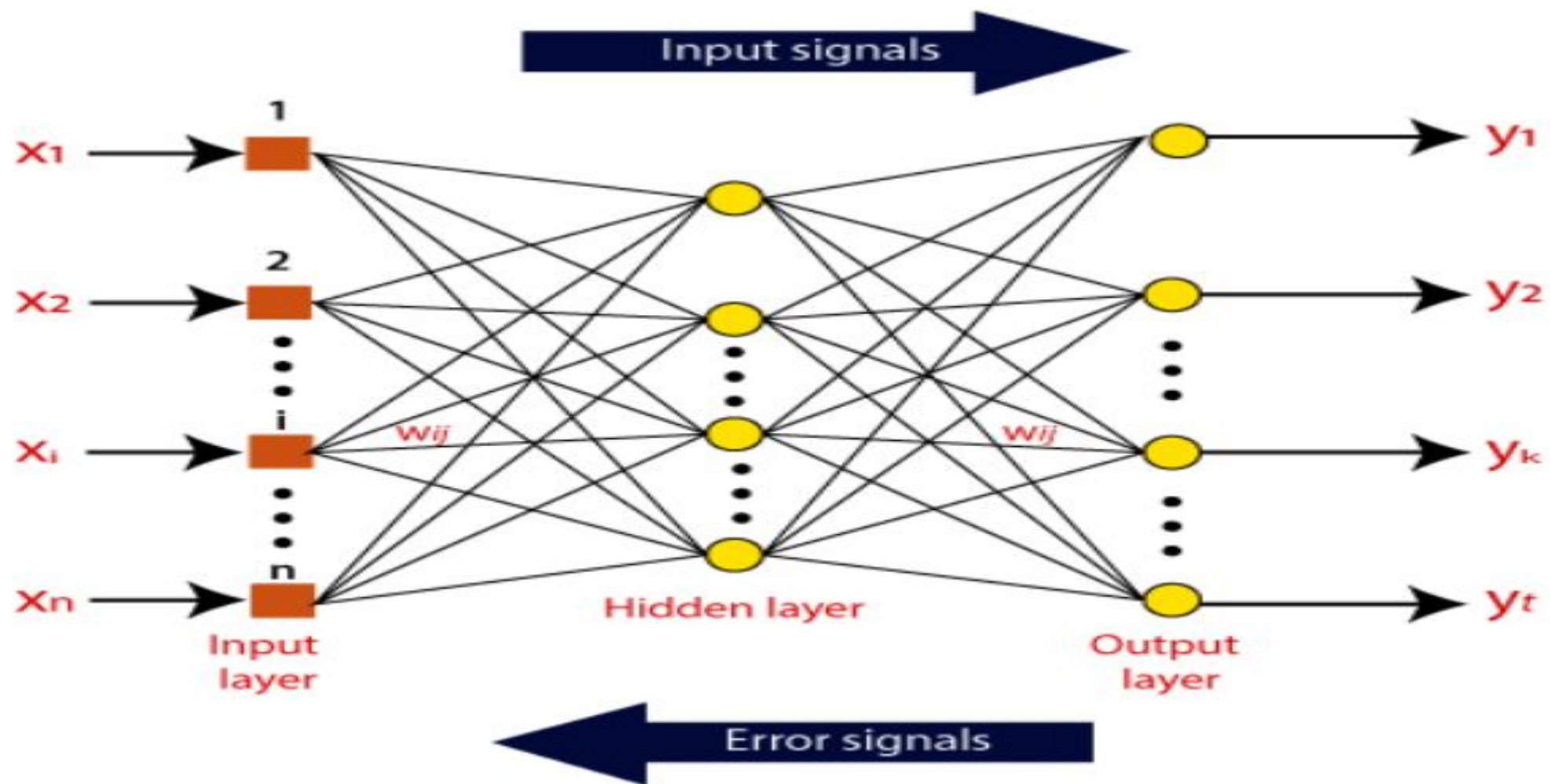
The artificial neural network takes input and computes the weighted sum of the inputs and includes a bias. This computation is represented in the form of a transfer function.

$$\sum_{i=1}^n W_i * X_i + b$$

It determines weighted total is passed as an input to an activation function to produce the output. Activation functions choose whether a node should fire or not. Only those who are fired make it to the output layer. There are distinctive activation functions available that can be applied upon the sort of task we are performing.

How do artificial neural networks work?

- Artificial Neural Network can be best represented as a weighted directed graph, where the artificial neurons form the nodes.
- The association between the neurons outputs and neuron inputs can be viewed as the directed edges with weights.
- The Artificial Neural Network receives the input signal from the external source in the form of a pattern and image in the form of a vector.
- These inputs are then mathematically assigned by the notations $x(n)$ for every n number of inputs.



- Afterward, each of the input is multiplied by its corresponding weights (these weights are the details utilized by the artificial neural networks to solve a specific problem).
- In general terms, these weights normally represent the strength of the interconnection between neurons inside the artificial neural network. All the weighted inputs are summarized inside the computing unit.
- If the weighted sum is equal to zero, then bias is added to make the output non-zero or something else to scale up to the system's response.
- Bias has the same input, and weight equals to 1. Here the total of weighted inputs can be in the range of 0 to positive infinity.
- Here, to keep the response in the limits of the desired value, a certain maximum value is benchmarked, and the total of weighted inputs is passed through the activation function.
- The activation function refers to the set of transfer functions used to achieve the desired output.
- There is a different kind of the activation function, but primarily either linear or non-linear sets of functions. Some of the commonly used sets of activation functions are the Binary, linear, Tangent hyperbolic, sigmoidal, etc. activation functions.

Activation Function

- Activation functions refer to the functions used in neural networks to compute the weighted sum of input and biases, which is used to choose the neuron that can be fire or not.
- It controls the presented information through some gradient processing, normally gradient descent. It produces an output for the neural network that includes the parameters in the data.
- Activation function can either be linear or non-linear, relying on the function it shows. It is used to control the output of outer neural networks across various areas, such as speech recognition, segmentation, fingerprint detection, cancer detection system, etc.
- In the artificial neural network, we can use activation functions over the input to get the precise output. These are some activation functions that are used in ANN.

Linear Activation Function:

- The equation of the linear activation function is the same as the equation of a straight line i.e.

$$Y = mx + c$$

- If we have many layers and all the layers are linear in nature, then the final activation function of the last layer is the same as the linear function of the first layer. The range of a linear function is $-\infty$ to $+\infty$.
- Linear activation function can be used at only one place that is the output layer.

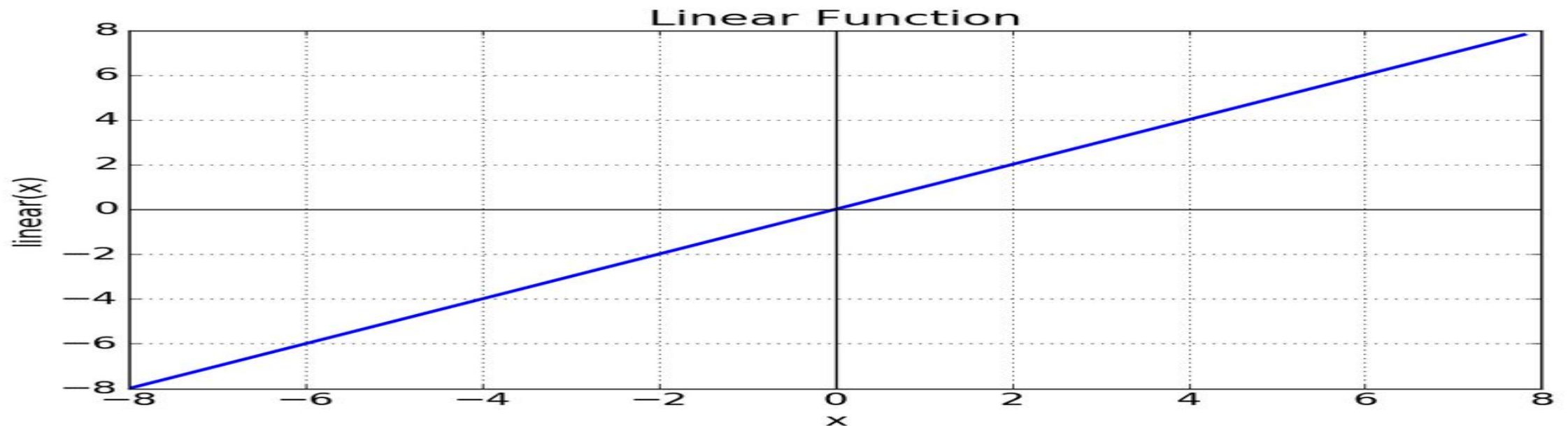
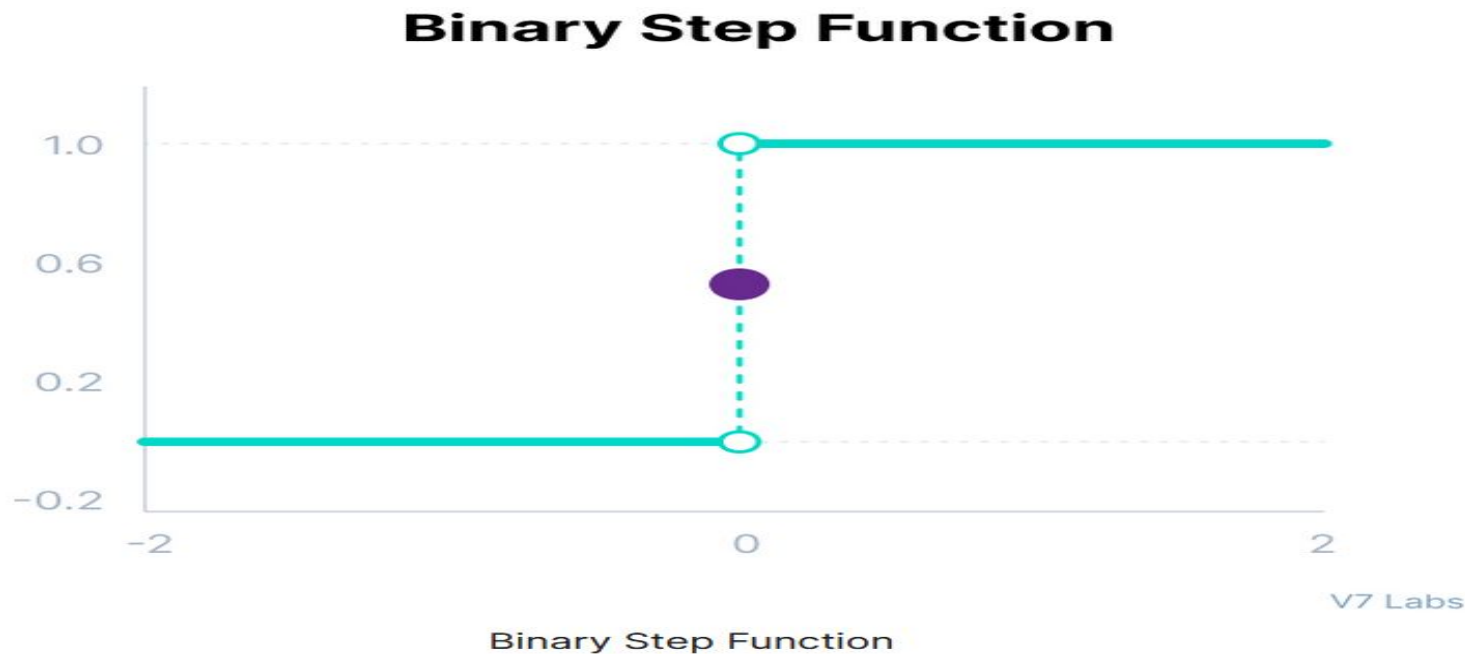


Fig: Linear Activation Function

Binary Step Function

Binary step function depends on a threshold value that decides whether a neuron should be activated or not.

The input fed to the activation function is compared to a certain threshold; if the input is greater than it, then the neuron is activated, else it is deactivated, meaning that its output is not passed on to the next hidden layer.



Mathematically it can be represented as:

Binary step

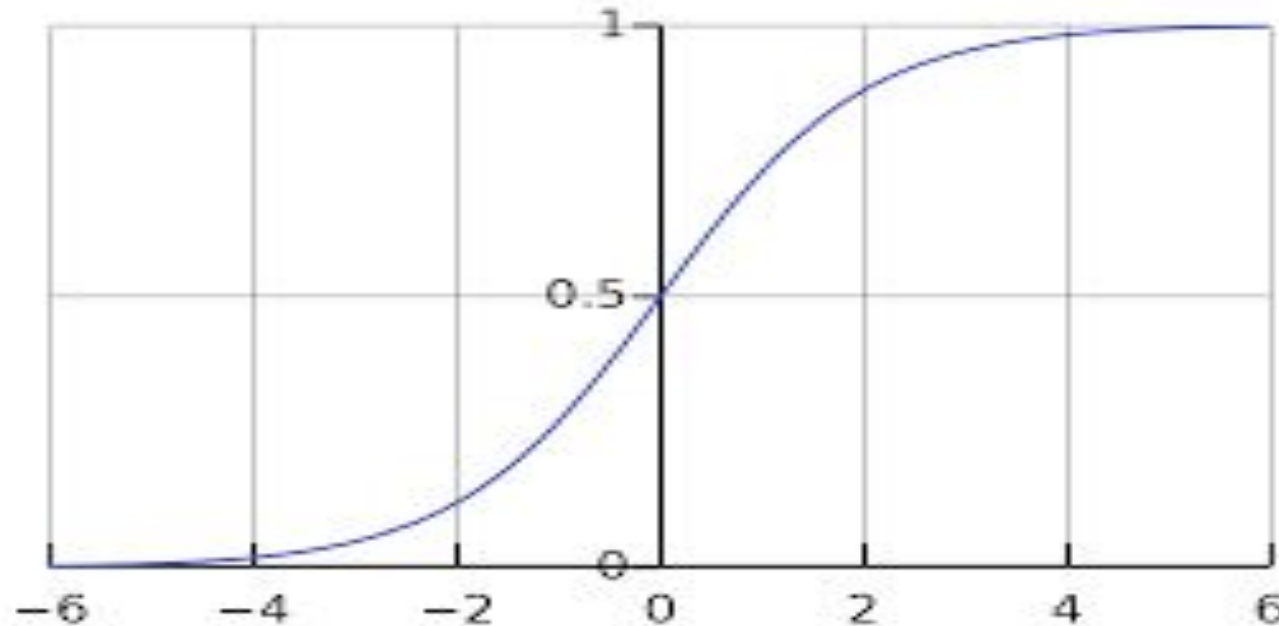
$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$$

Sigmoid function:

- Sigmoid function refers to a function that is projected as S - shaped graph.

$$A = 1/(1+e^{-x})$$

- This function is non-linear. The value of X is directly proportional to the values of Y. It means a slight change in the value of x would also bring about changes in the value of y.



Tanh Function:

- The activation function, which is more efficient than the sigmoid function is Tanh function.
- Tanh function is also known as Tangent Hyperbolic Function.
- It is a mathematical updated version of the sigmoid function.
- Sigmoid and Tanh function are similar to each other and can be derived from each other.

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

OR

$$\text{Tanh}(x) = 2 * \text{sigmoid}(2x) - 1$$

- This function is non-linear, and the value range lies between -1 to +1

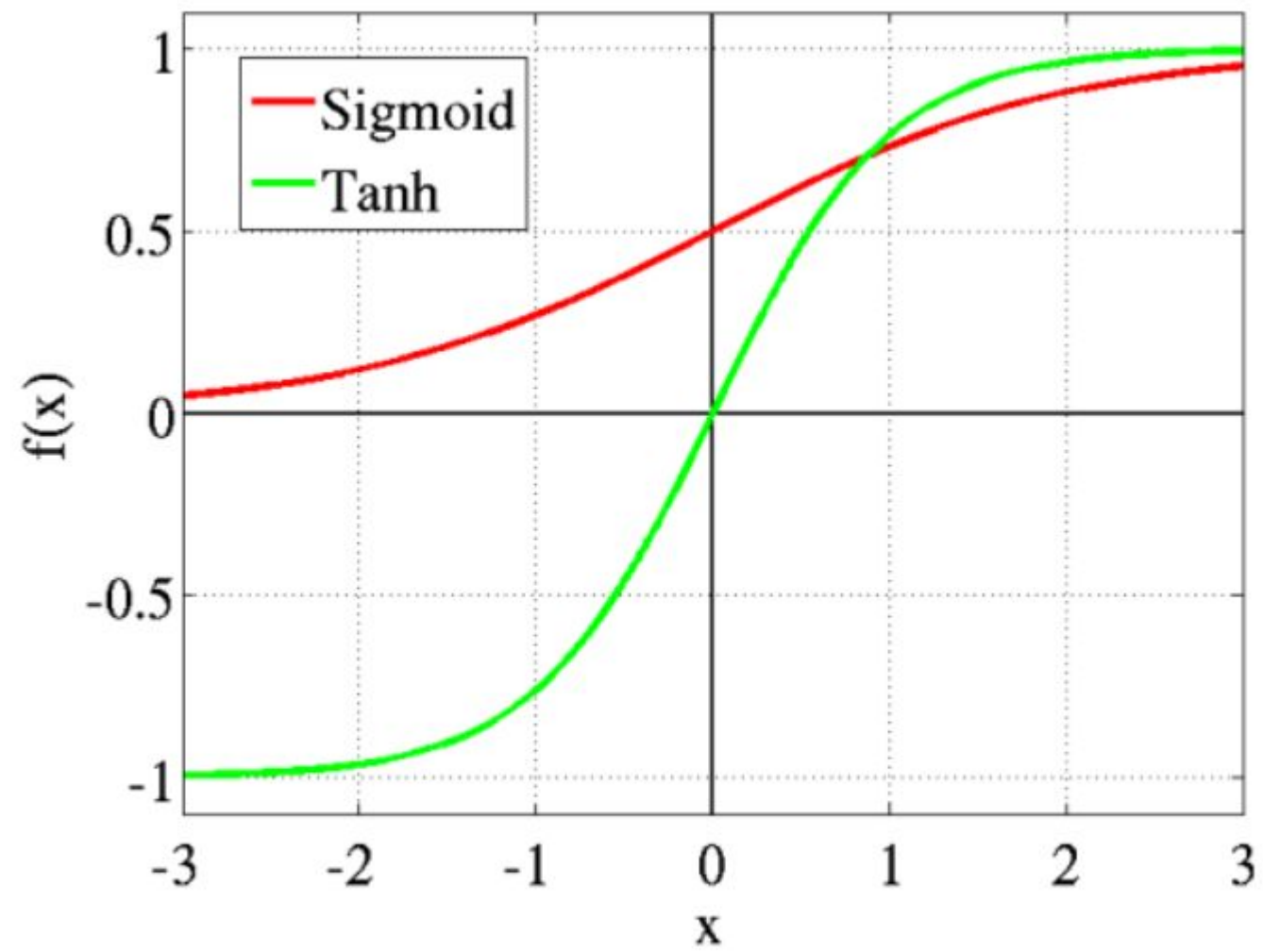
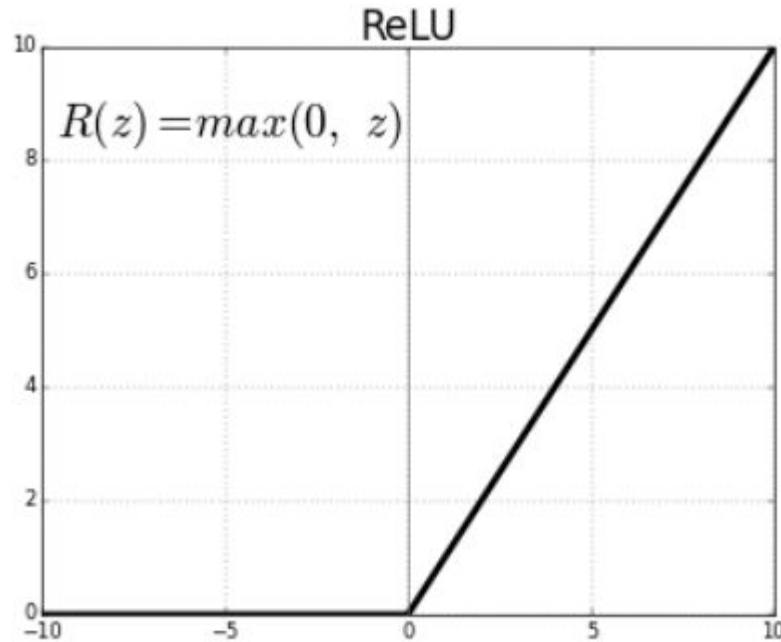


Fig: tanh v/s Logistic Sigmoid


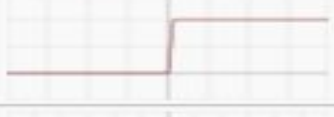







ReLU (Rectified Linear Unit) Activation Function

- The ReLU is the most used activation function in the world right now. Since, it is used in almost all the convolutional neural networks or deep learning.



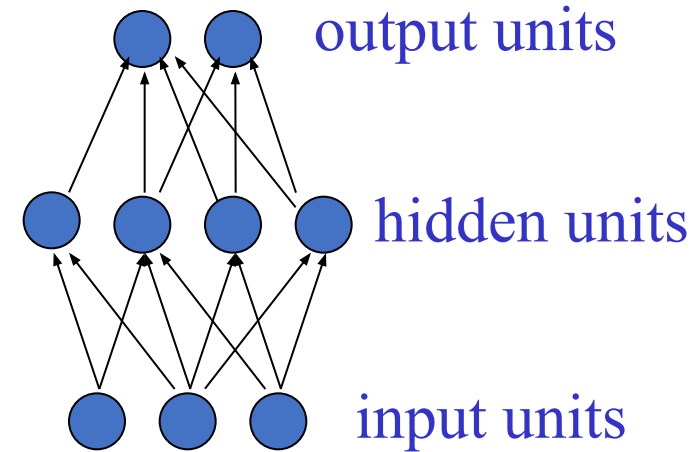
- As you can see, the ReLU is half rectified (from bottom). $f(z)$ is zero when z is less than zero and $f(z)$ is equal to z when z is above or equal to zero.
- **Range:** [0 to infinity)

Summarization (including other functions)

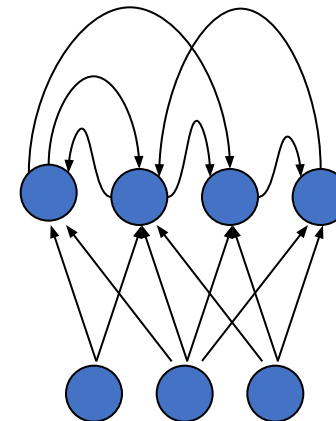
Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2 + 1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU) [2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU) [3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \frac{1}{1 + e^{-x}}$

Types of Artificial Neural Network

- Feedforward networks
 - These compute a series of transformations
 - Typically, the first layer is the input and the last layer is the output.



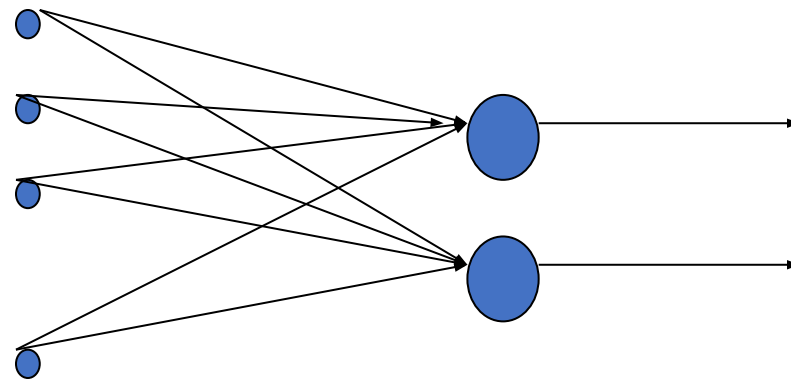
- Recurrent networks
 - These have directed cycles in their connection graph. They can have complicated dynamics.
 - More biologically realistic.



Different Network Topologies

- Single layer feed-forward networks

- Input layer projecting into the output layer



Input
layer

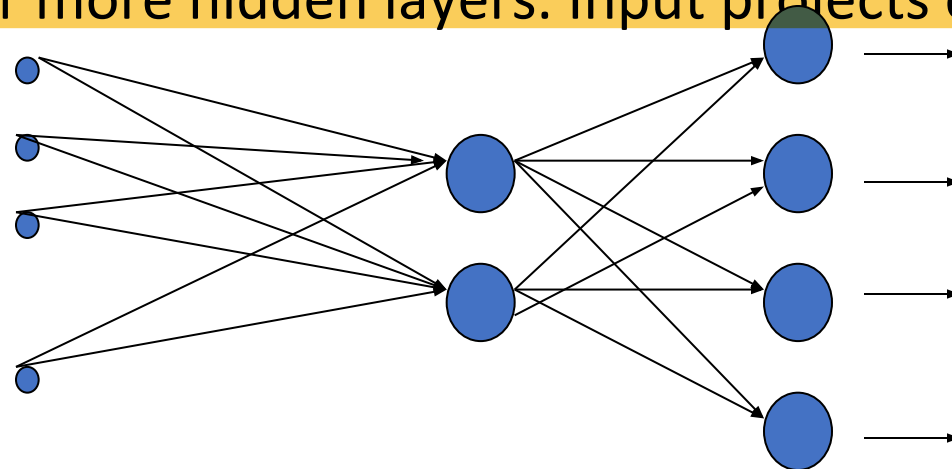
Output
layer

Single layer
network

Different Network Topologies

- Multi-layer feed-forward networks

- One or more hidden layers. Input projects only from previous layers onto a layer.



Input
layer

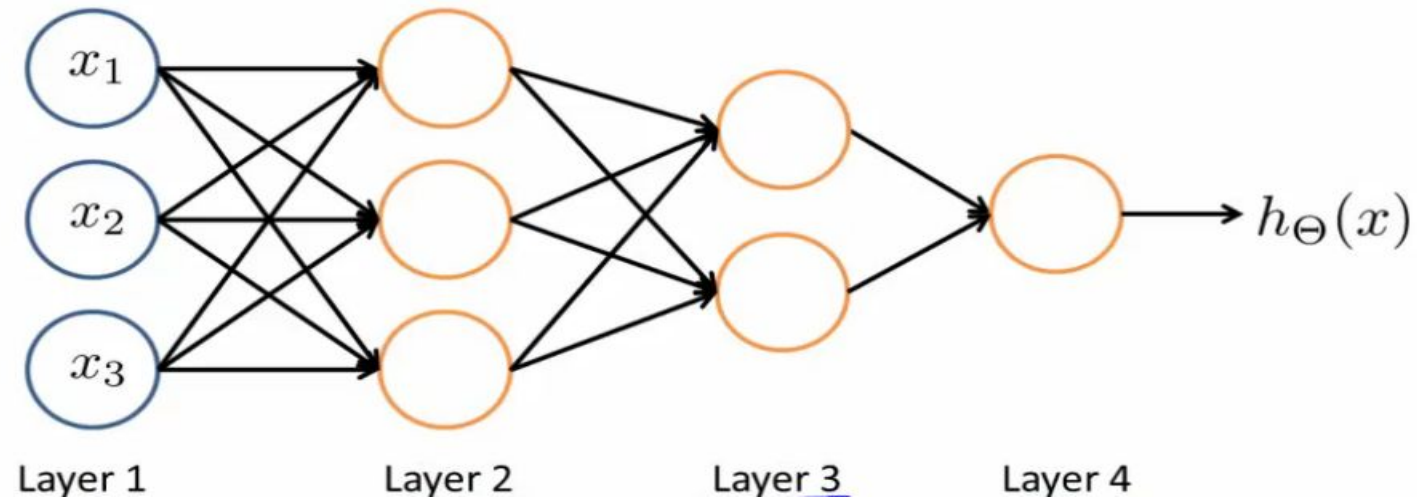
Hidden
layer

Output
layer

2-layer or
1-hidden layer
fully connected
network

Different Network Topologies

- Multi-layer feed-forward networks



Input
layer

Hidden
layers

Output
layer

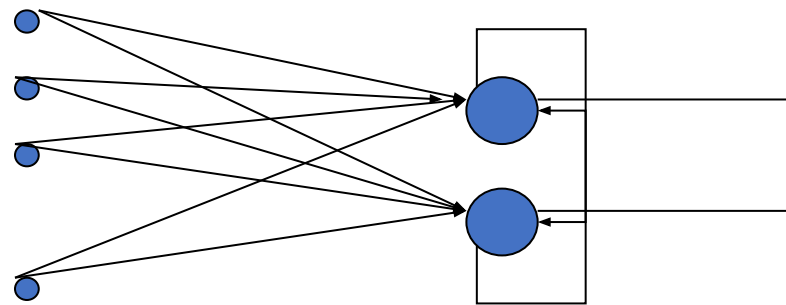
Recurrent Neural Network

- A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data.
- Like feedforward and convolutional neural networks (CNNs), recurrent neural networks utilize training data to learn.
- They are distinguished by their “memory” as they take information from prior inputs to influence the current input and output.
- While traditional deep neural networks assume that inputs and outputs are independent of each other, the output of recurrent neural networks depend on the prior elements within the sequence.

Different Network Topologies

- Recurrent networks

- A network with feedback, where some of its inputs are connected to some of its outputs (discrete time).

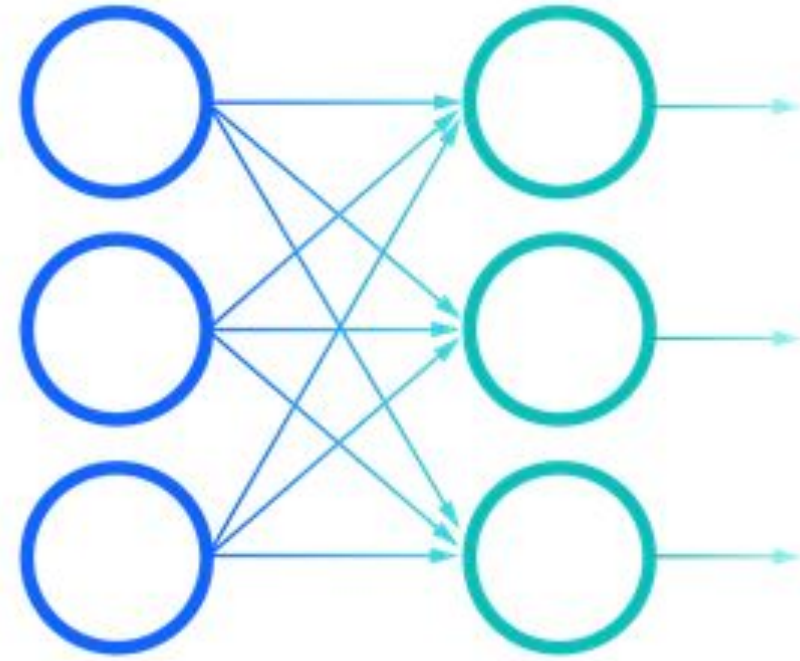
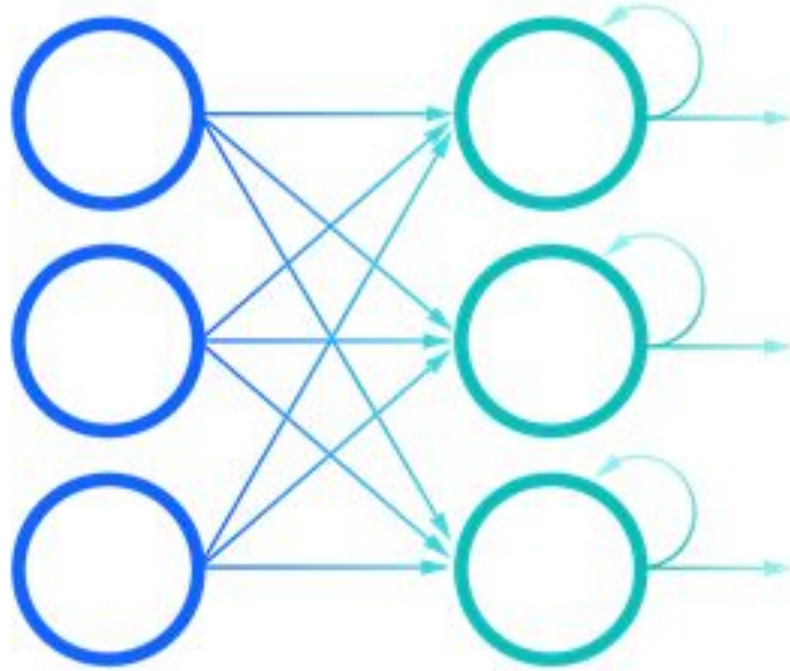


Input
layer

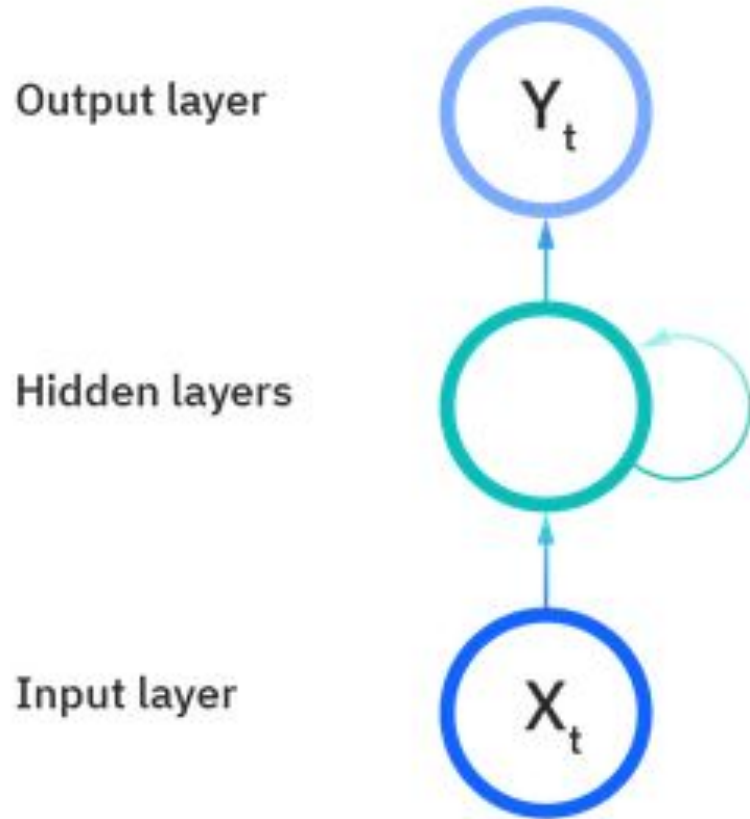
Output
layer

Recurrent
network

Recurrent Neural Network vs. Feedforward Neural Network

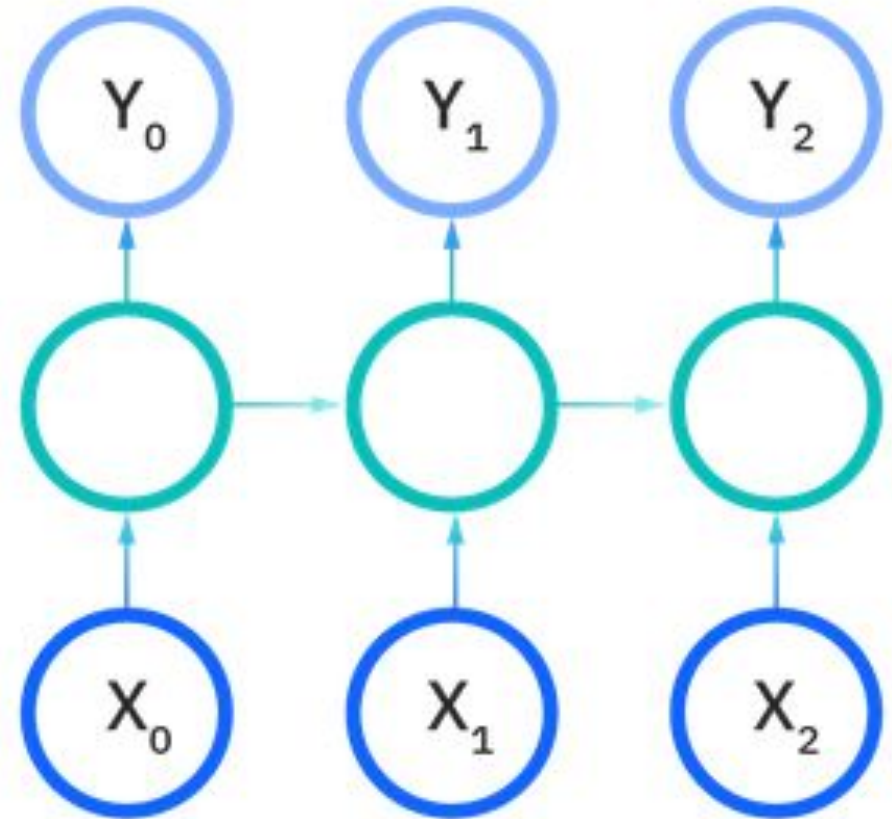


Rolled RNN



=

Unrolled RNN



Time

Formula for calculating current state:

$$h_t = f(h_{t-1}, x_t)$$

where:

h_t -> current state

h_{t-1} -> previous state

x_t -> input state

Formula for applying Activation function(tanh):

$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

where:

w_{hh} -> weight at recurrent neuron

w_{xh} -> weight at input neuron

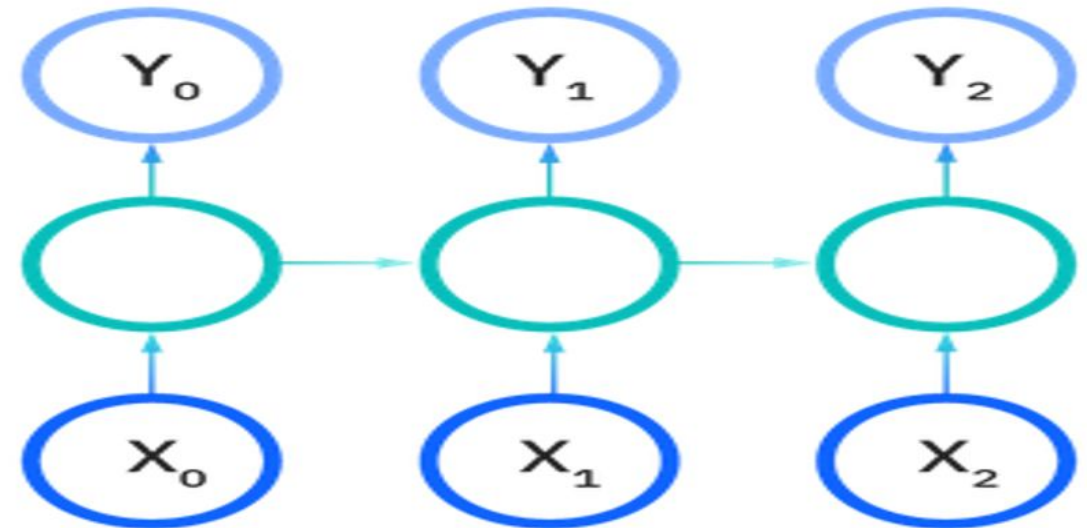
Formula for calculating output:

$$y_t = W_{hy}h_t$$

Where:

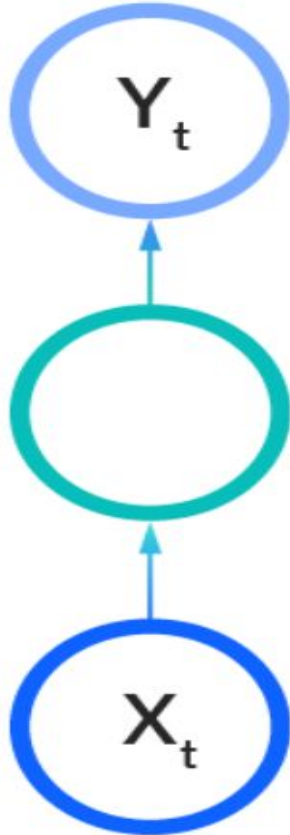
Y_t -> output

W_{hy} -> weight at output layer

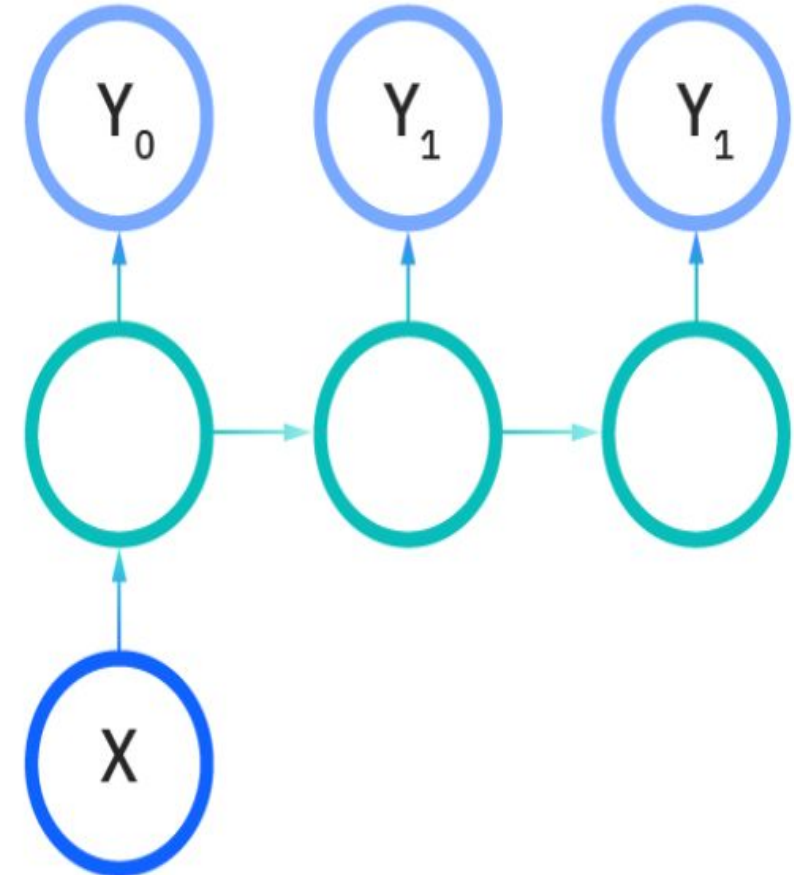
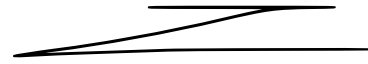


Types of recurrent neural networks

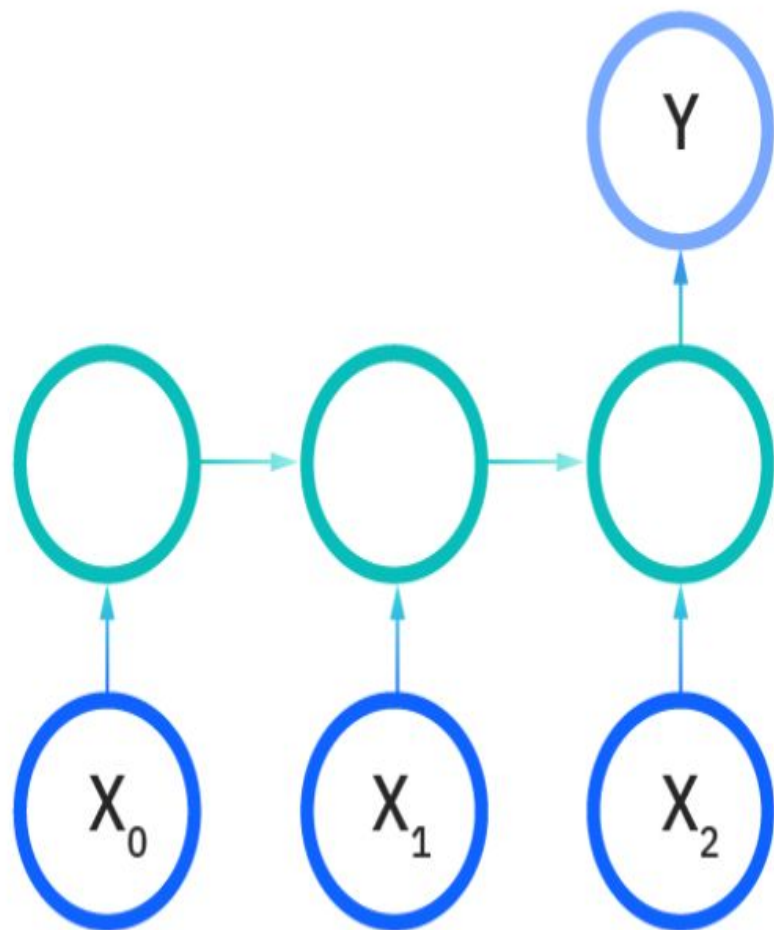
One-to-one:



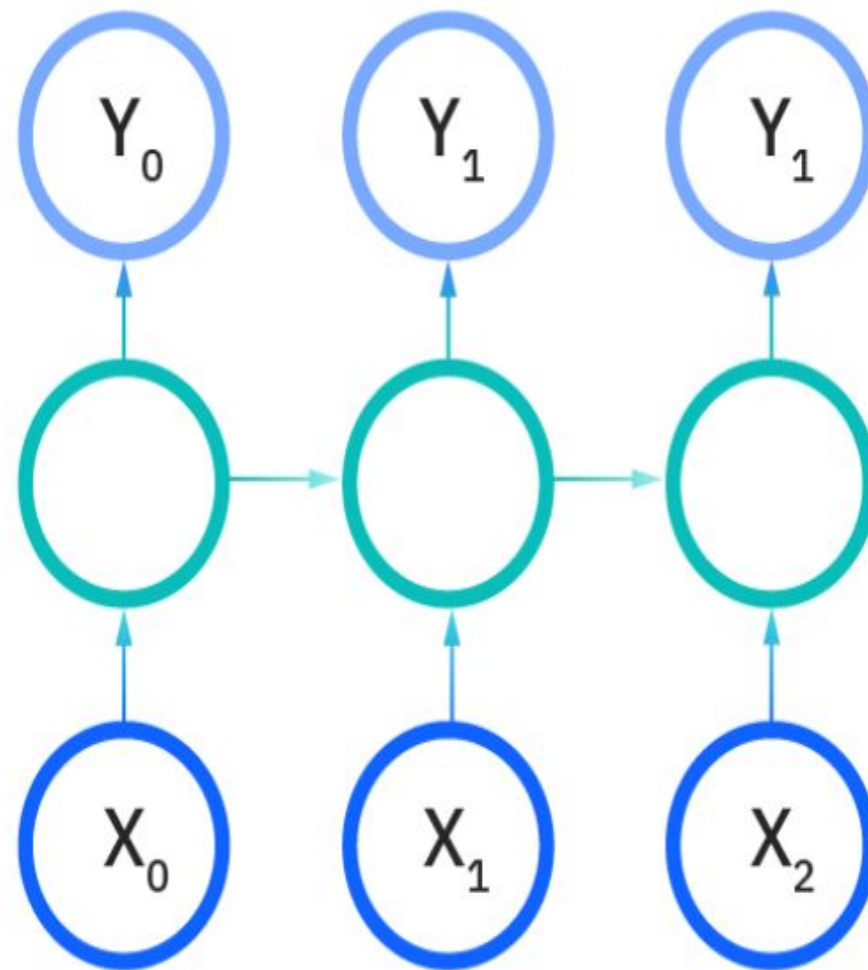
One-to-many:



Many-to-one:



~~Many-to-many:~~



Many-to-many:

