Artificial Neural Network

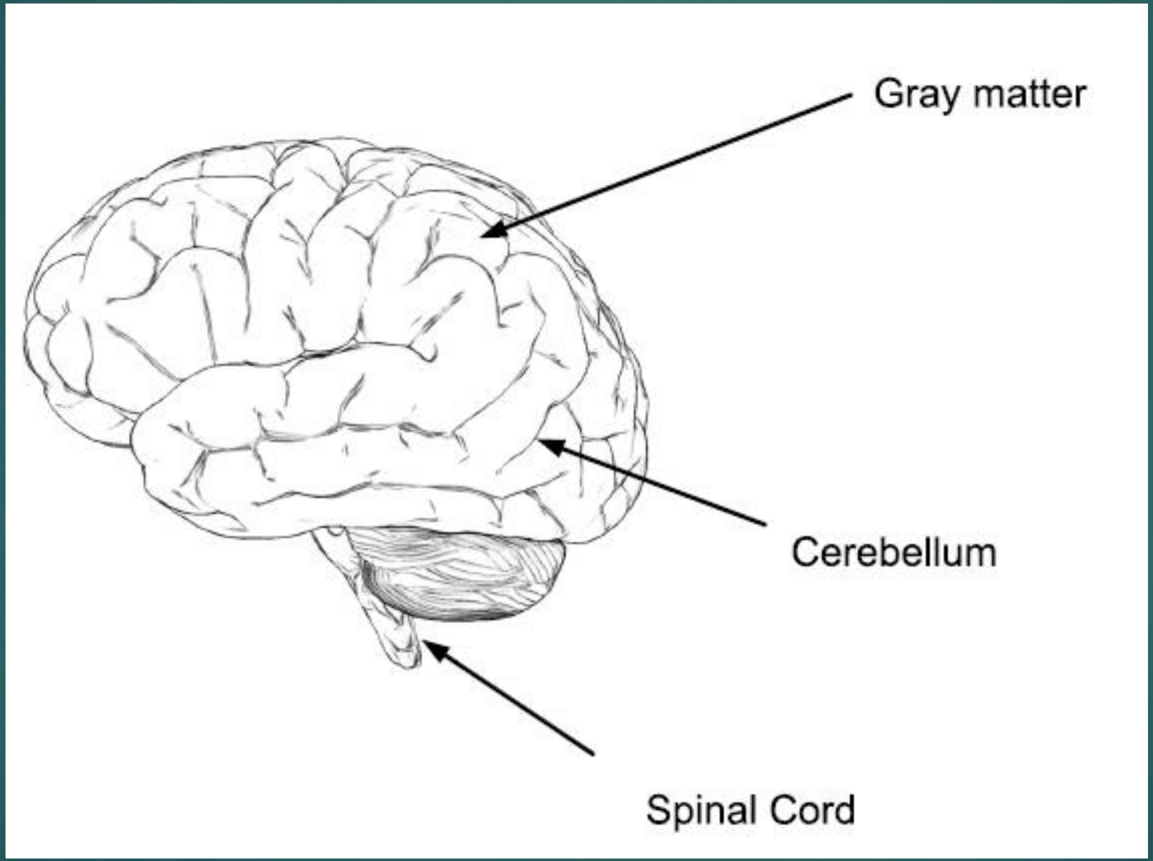
Dr. Navneet Kaur

CHITKARA UNIVERSITY
HIMACHAL PRADESH
NAAC ACCREDITED

Biological nervous system

- Biological nervous system is the most important part of many living things, in particular, human beings.
- There is a part called **brain** at the center of human nervous system.
- In fact, any biological nervous system consists of a large number of interconnected processing units called **neurons**.
- Each neuron is approximately 10μm long and they can operate in parallel.
- Typically, a human brain consists of approximately 10 billion neurons communicating with each other with the help of **electrical impulses.**

Brain: Center of the nervous system

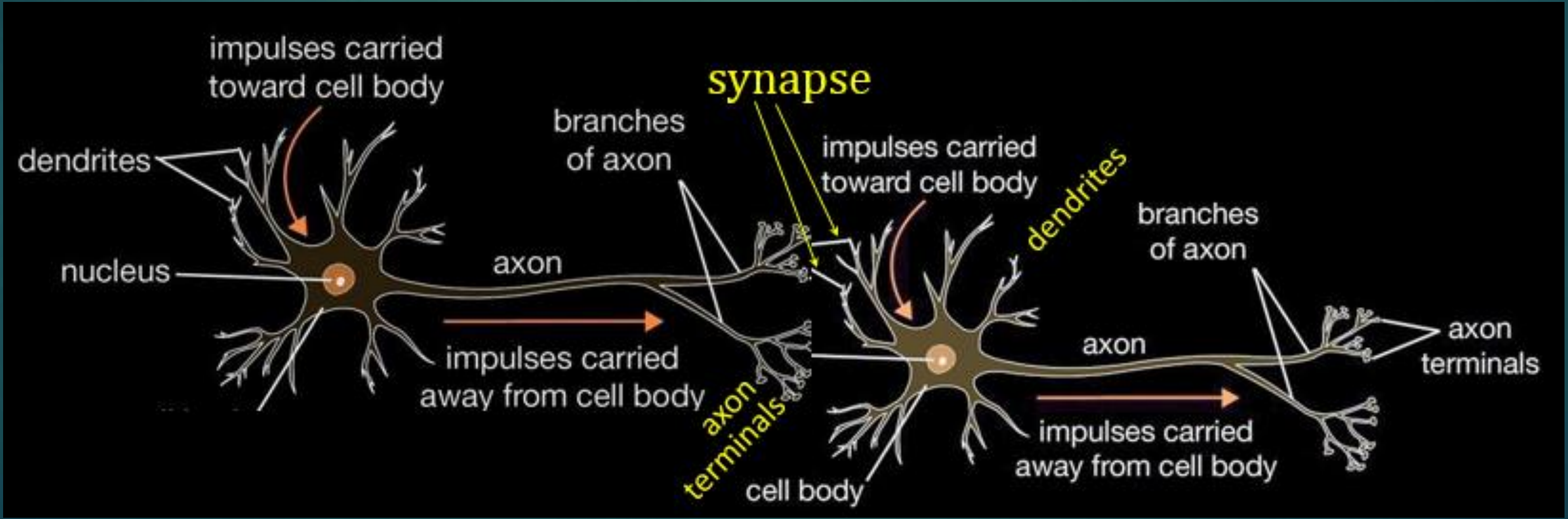


How Does the Brain Work?

- At birth, a brain has great structure and the ability to build up its own rules through what we usually refer to as "experience."
- experience is built up over time, with the most dramatic development
  ~i.e., hard-wiring of human brain take place during first two years from birth
  ~but the development continues well beyond that stage

What is a Biological Neural Network (BNN)?

- A Biological Neural Network is a network composed of a group of chemically connected or functionally associated neurons
- A single neuron may be connected to many other neurons
- Total number of neurons and connections in a network may be extensive


impulses carried toward cell body
dendrites
nucleus
branches of axon
axon
impulses carried away from cell body
axon terminals
synapse
impulses carried toward cell body
dendrites
branches of axon
axon
cell body
impulses carried away from cell body
axon terminals

Neurons are the Fundamental Units of BNNs

- Neurons (also called neurones or nerve cells) are the fundamental units of the brain and nervous system.
- Neurons are responsible for receiving sensory input from the external world
  - ✓ for sending motor commands to our muscles, and
  - ✓ for transforming and relaying the electrical signals at every step in between.
- Neurons pass information from one to another using action potentials
- They connect with one another at synapse, which are junctions between one neuron's axon and another's dendrite
- Information flows from the dendrite to the cell body through the axons to a synapse connecting the axon to the dendrite of the next neuron

Neuron: Basic unit of nervous system

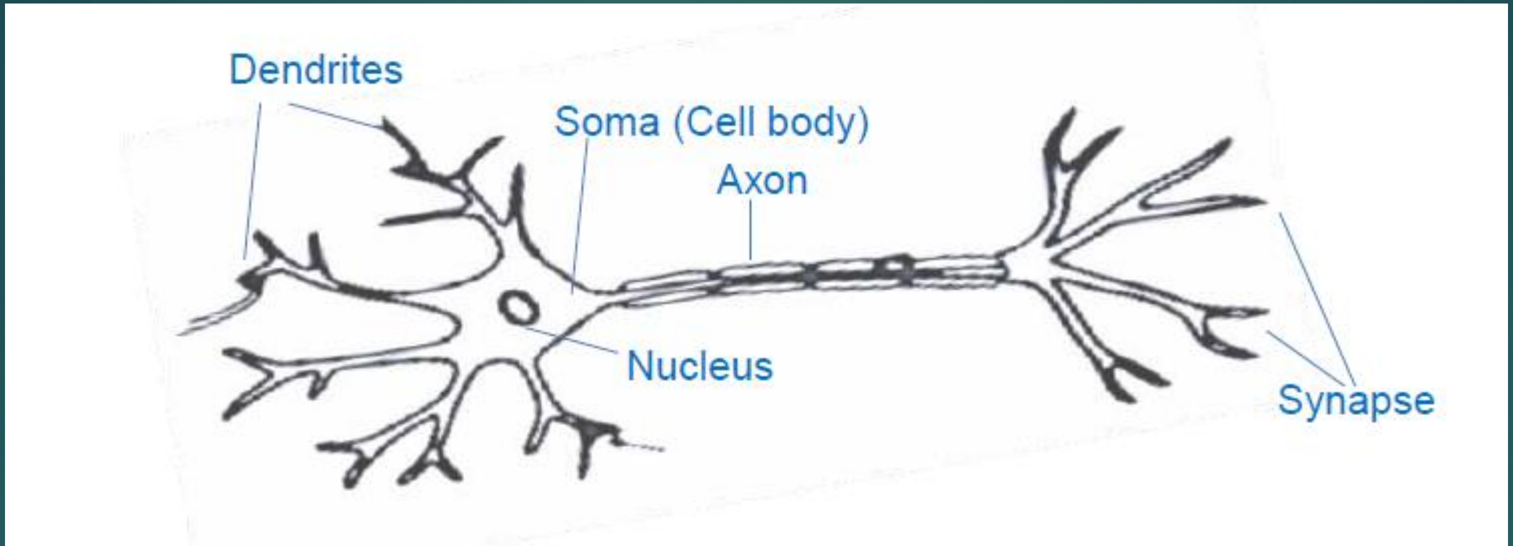


Figure shows a schematic of a biological neuron. There are different parts in it : dendrite, soma, axon and synapse.

Dendrite : A bush of very thin fibre.

Axon : A long cylindrical fibre.

Soma : It is also called a cell body, and just like as a nucleus of cell.

Synapse : It is a junction where axon makes contact with the dendrites of neighbouring dendrites.

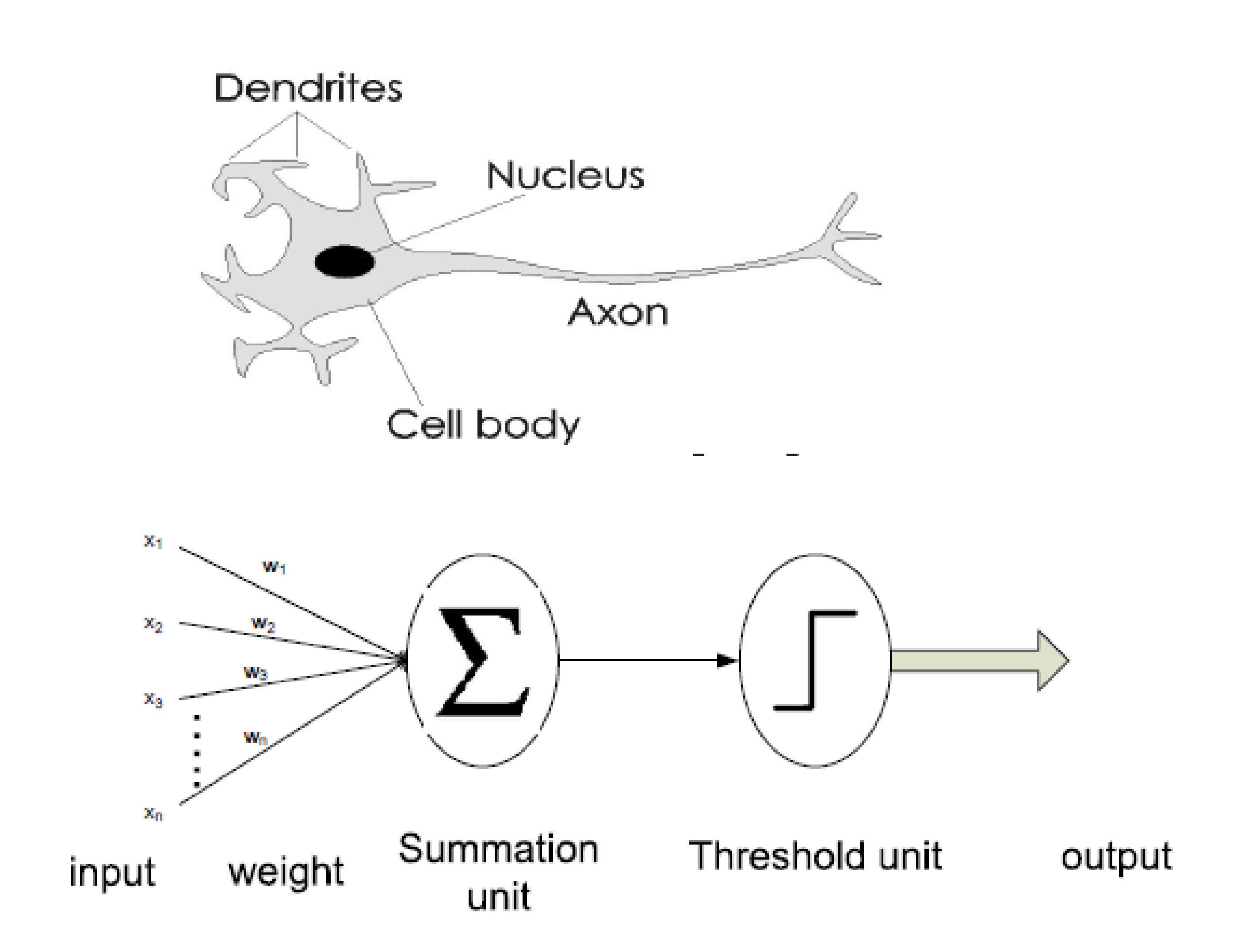
Neuron and its working

- There is a chemical in each neuron called neurotransmitter.
- A signal (also called sense) is transmitted across neurons by this chemical.
- That is, all inputs from other neuron arrive to a neurons through dendrites.
- These signals are accumulated at the synapse of the neuron and then serve as the output to be transmitted through the neuron.
- An action may produce an electrical impulse, which usually lasts for about a millisecond.
- Note that this pulse generated due to an incoming signal and all signal may not produce pulses in axon unless it crosses a **threshold value**.
- Also, note that an action signal in axon of a neuron is commutative signals arrive at dendrites which summed up at soma.

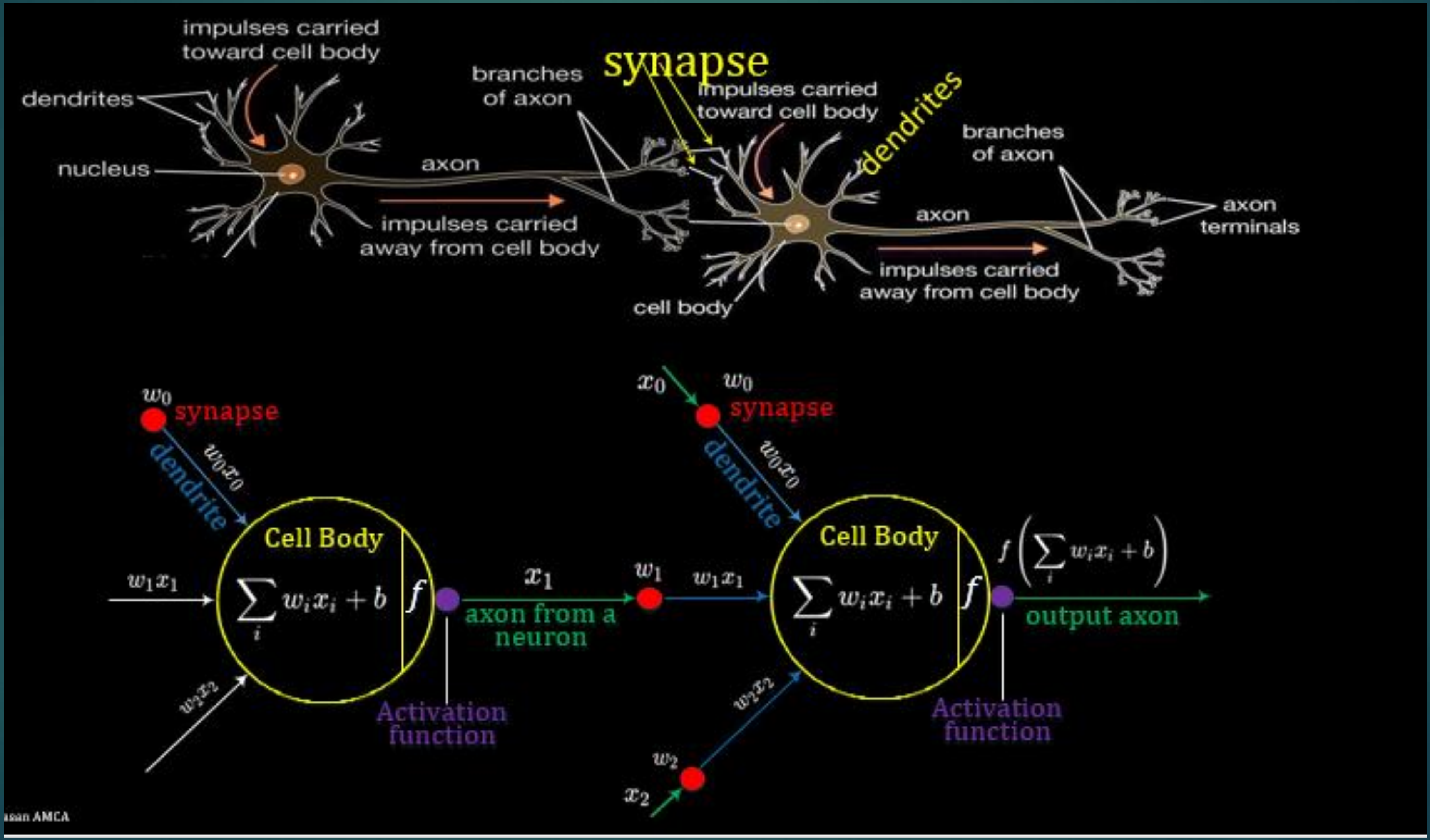
Artificial neural network

- In fact, the human brain is a highly complex structure viewed as a massive, highly interconnected network of simple processing elements called **neurons**.
- Artificial neural networks (ANNs) or simply we refer it as neural network (NNs), which are simplified models (i.e. imitations) of the biological nervous system, and obviously, therefore, have been motivated by the kind of computing performed by the human brain.
- The behavior of a biolgical neural network can be captured by a simple model called artificial neural network.

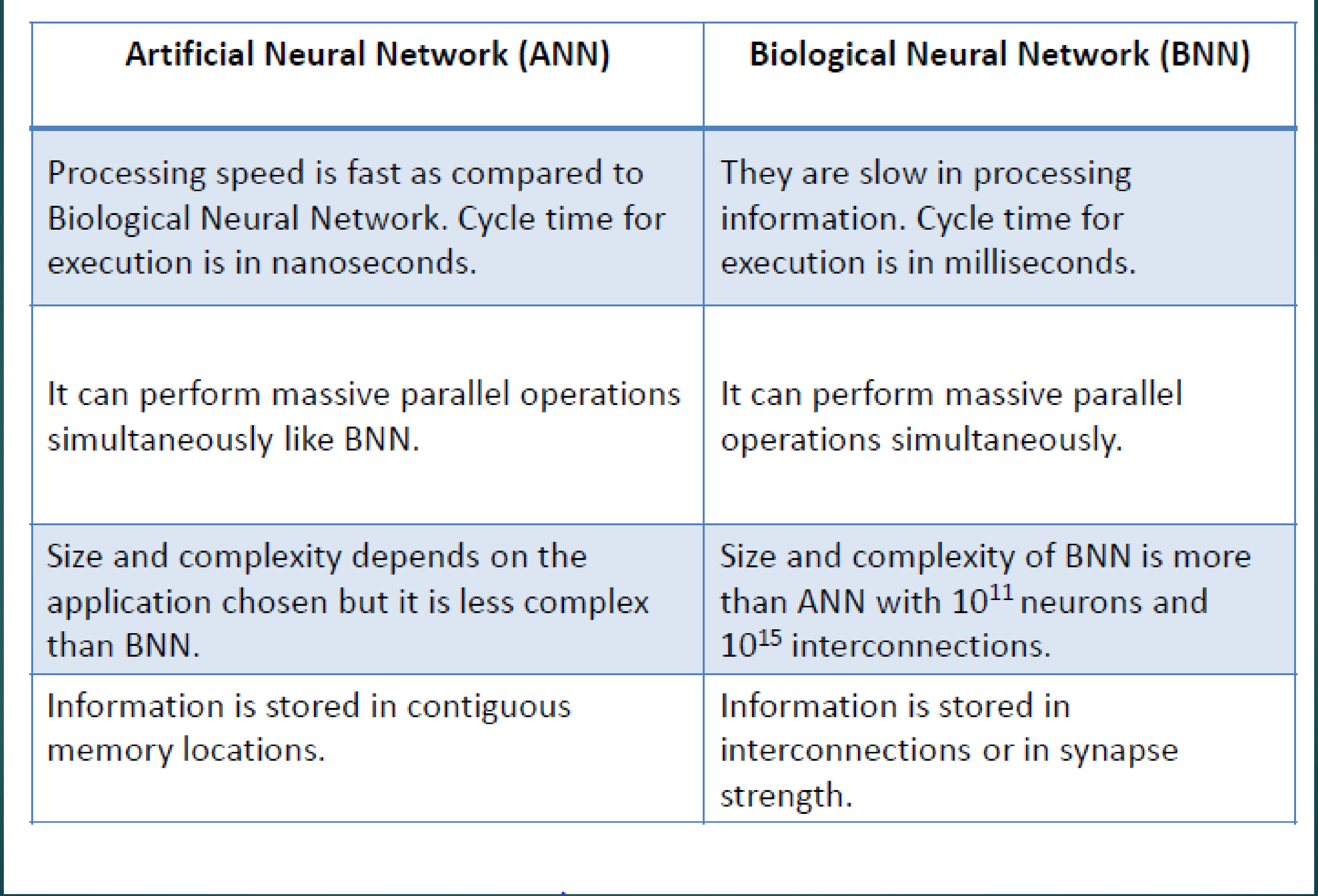
Analogy between BNN and ANN



Artificial Neural Network Connections

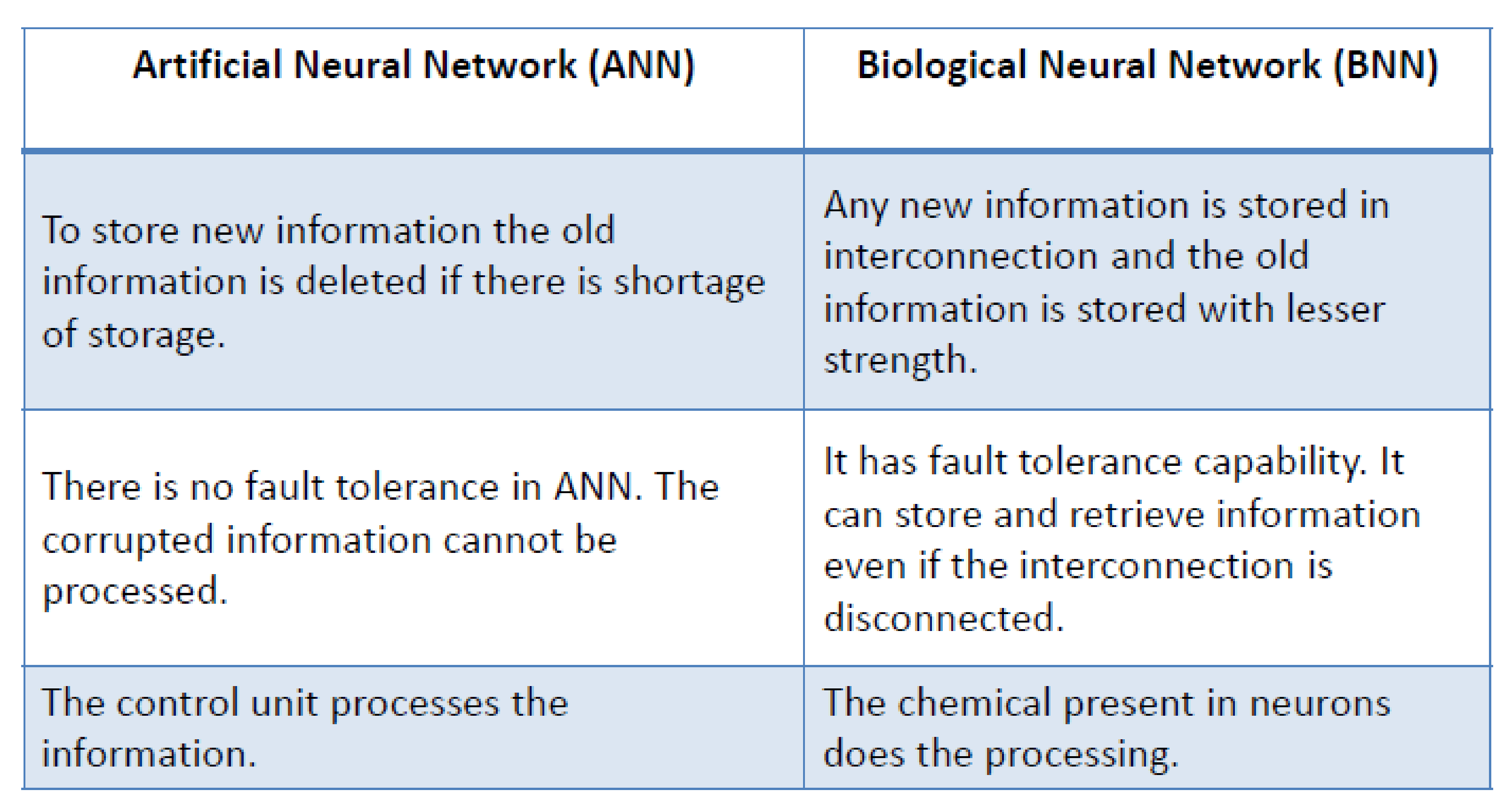


BNN v/s ANN



Artificial Neural Network (ANN)	Biological Neural Network (BNN)
Processing speed is fast as compared to Biological Neural Network. Cycle time for execution is in nanoseconds.	They are slow in processing information. Cycle time for execution is in milliseconds.
It can perform massive parallel operations simultaneously like BNN.	It can perform massive parallel operations simultaneously.
Size and complexity depends on the application chosen but it is less complex than BNN.	Size and complexity of BNN is more than ANN with 10^{11} neurons and 10^{15} interconnections.
Information is stored in contiguous memory locations.	Information is stored in interconnections or in synapse strength.

BNN v/s ANN



Artificial Neural Network (ANN)	Biological Neural Network (BNN)
To store new information the old information is deleted if there is shortage of storage.	Any new information is stored in interconnection and the old information is stored with lesser strength.
There is no fault tolerance in ANN. The corrupted information cannot be processed.	It has fault tolerance capability. It can store and retrieve information even if the interconnection is disconnected.
The control unit processes the information.	The chemical present in neurons does the processing.

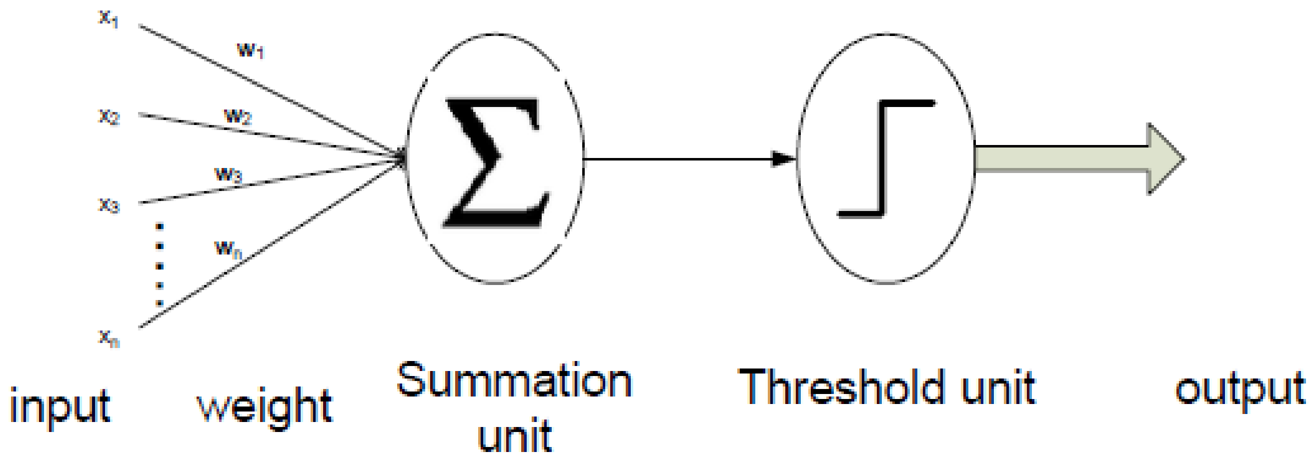
Artificial neural network

We may note that a neutron is a part of an interconnected network of nervous system and serves the following.

- Compute input signals
- Transportation of signals (at a very high speed)
- Storage of information
- Perception, automatic training and learning

We also can see the analogy between the biological neuron and artificial neuron. Truly, every component of the model (i.e. artificial neuron) bears a direct analogy to that of a biological neuron. It is this model which forms the basis of neural network (i.e. artificial neural network).

Artificial neural network



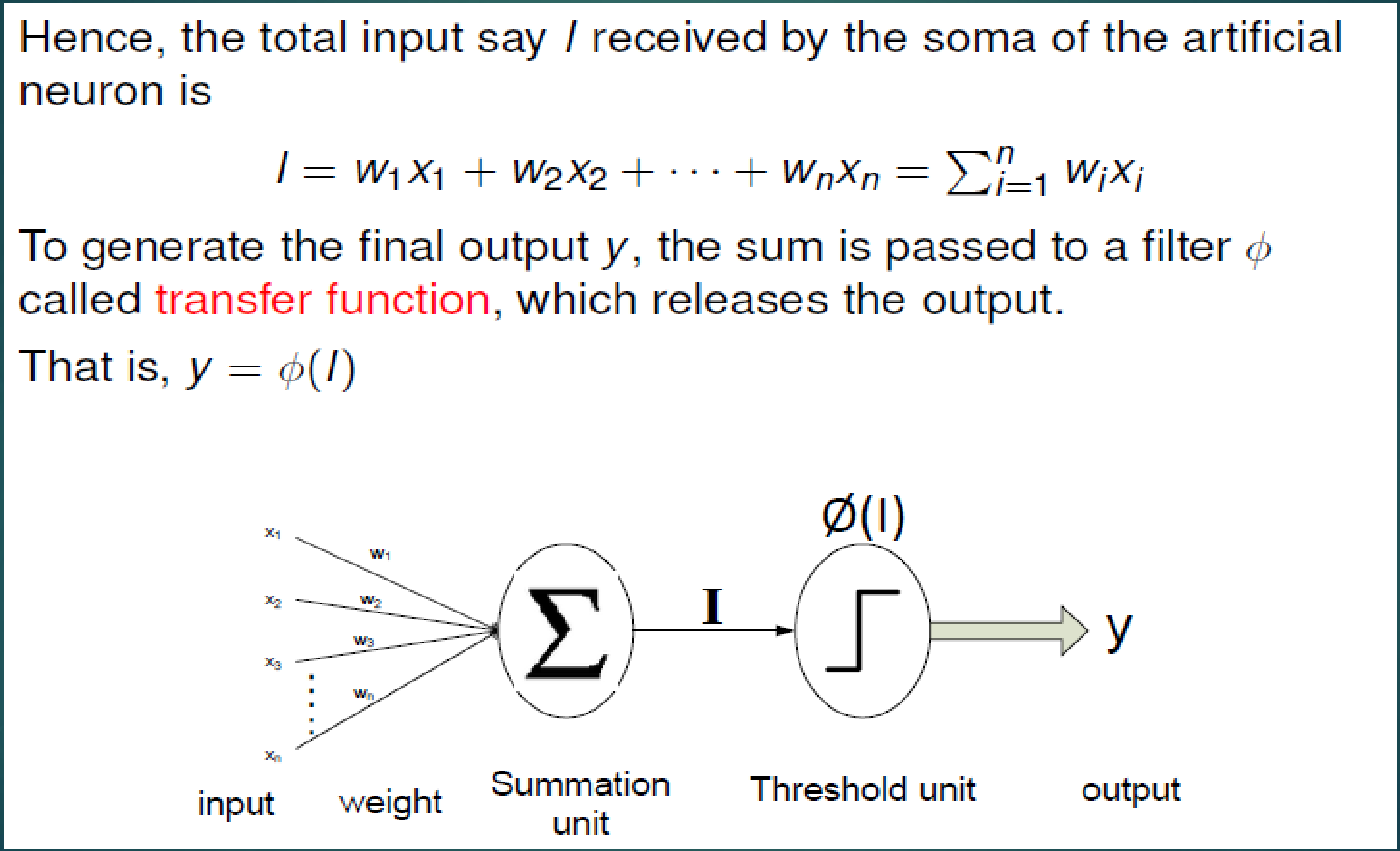
- Here, $x_1, x_2, \cdots, x_n$ are the n inputs to the artificial neuron.
- $w_1, w_2, \cdots, w_n$ are weights attached to the input links.

Soft Computing CS125

Artificial neural network

- Note that, a biological neuron receives all inputs through the dendrites, sums them and produces an output if the sum is greater than a threshold value.
- The input signals are passed on to the cell body through the synapse, which may accelerate or retard an arriving signal.
- It is this acceleration or retardation of the input signals that is modelled by the **weights**.
- An effective synapse, which transmits a stronger signal will have a correspondingly larger weights while a weak synapse will have smaller weights.
- Thus, weights here are multiplicative factors of the inputs to account for the strength of the synapse.

Artificial neural network



Hence, the total input say I received by the soma of the artificial neuron is

$$I = w_1x_1 + w_2x_2 + \cdots + w_nx_n = \sum_{i=1}^{n} w_ix_i$$

To generate the final output y, the sum is passed to a filter ϕ called transfer function, which releases the output.

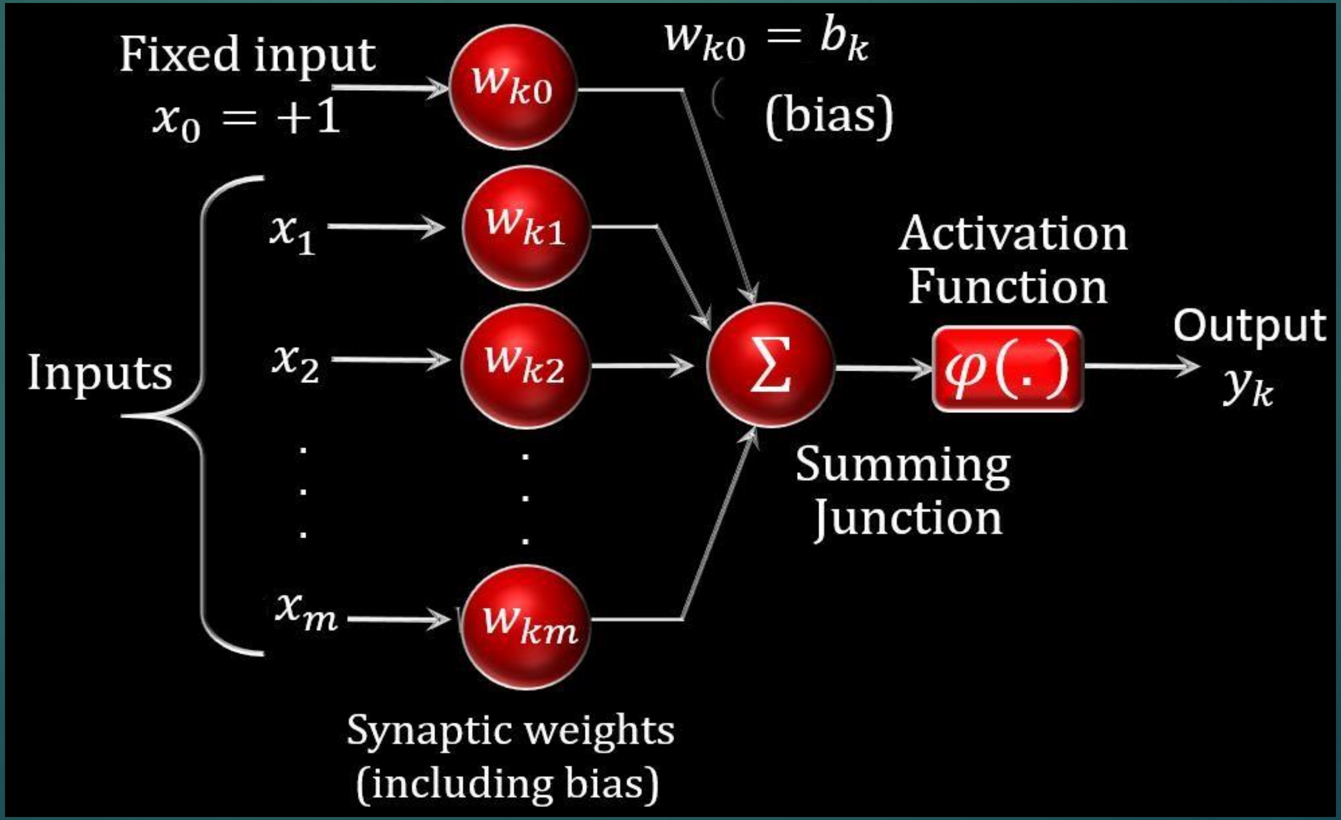
That is, $y = \phi(I)$

Ø(I)
x1
x2
x3
xn
w1
w2
w3
wn
Σ
I
y
input
weight
Summation unit
Threshold unit
output

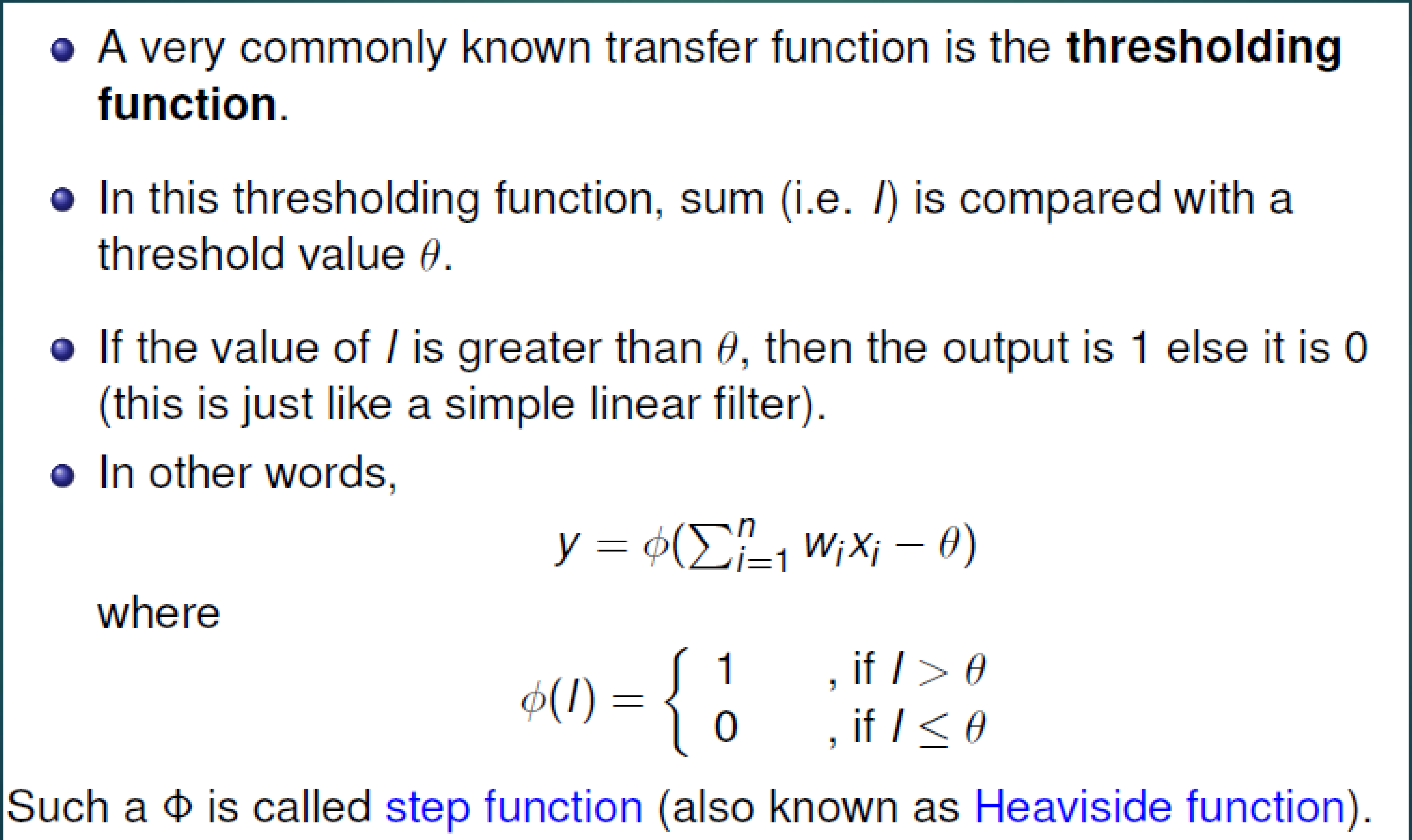
Soft Computing CS125

Artificial Neural Network

- This value is then passed to a non-linear activation function f to produce the output $f(b + \sigma_i\, w_i x_i)$.
- This is the activation of the neuron and is the value passed to other neurons connected to its axon



Artificial neural network



- A very commonly known transfer function is the **thresholding function**.
- In this thresholding function, sum (i.e. I) is compared with a threshold value θ.
- If the value of I is greater than θ, then the output is 1 else it is 0 (this is just like a simple linear filter).
- In other words,

$$y = \phi(\sum_{i=1}^{n} w_i x_i - \theta)$$

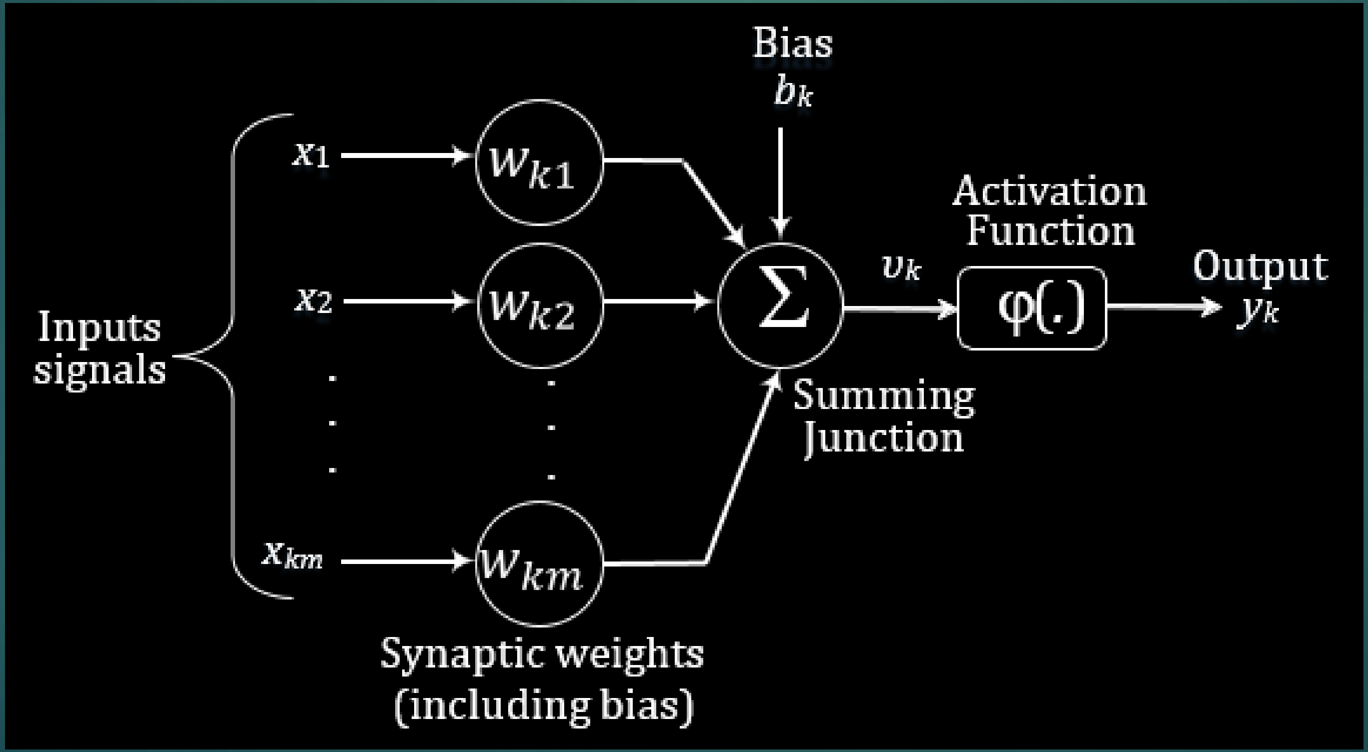
where

$$\phi(I) = \begin{cases} 1 & , \text{if } I > \theta \\ 0 & , \text{if } I \leq \theta \end{cases}$$

Such a Φ is called step function (also known as Heaviside function).

Models of Neuron

A neuron is an information-processing unit that is fundamental to the operation of a neural network. The block diagram shows the model of a neuron, which forms the basis for designing (artificial) neural networks.



Three Basic Elements of the Neuronal Model

- A set of synapses or connecting links, each of which is characterized by a weight or strength of its own. Specifically, a signal x_j at the input of synapse j connected to neuron k is multiplied by the synaptic weight w_{kj}
- An adder for summing the input signals, weighted by the respective synapses of the neuron
- An activation function for limiting the amplitude of the output of a neuron.

Mathematical Description of a Neuron

- We may describe a neuron *k* by writing the equations:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j$$

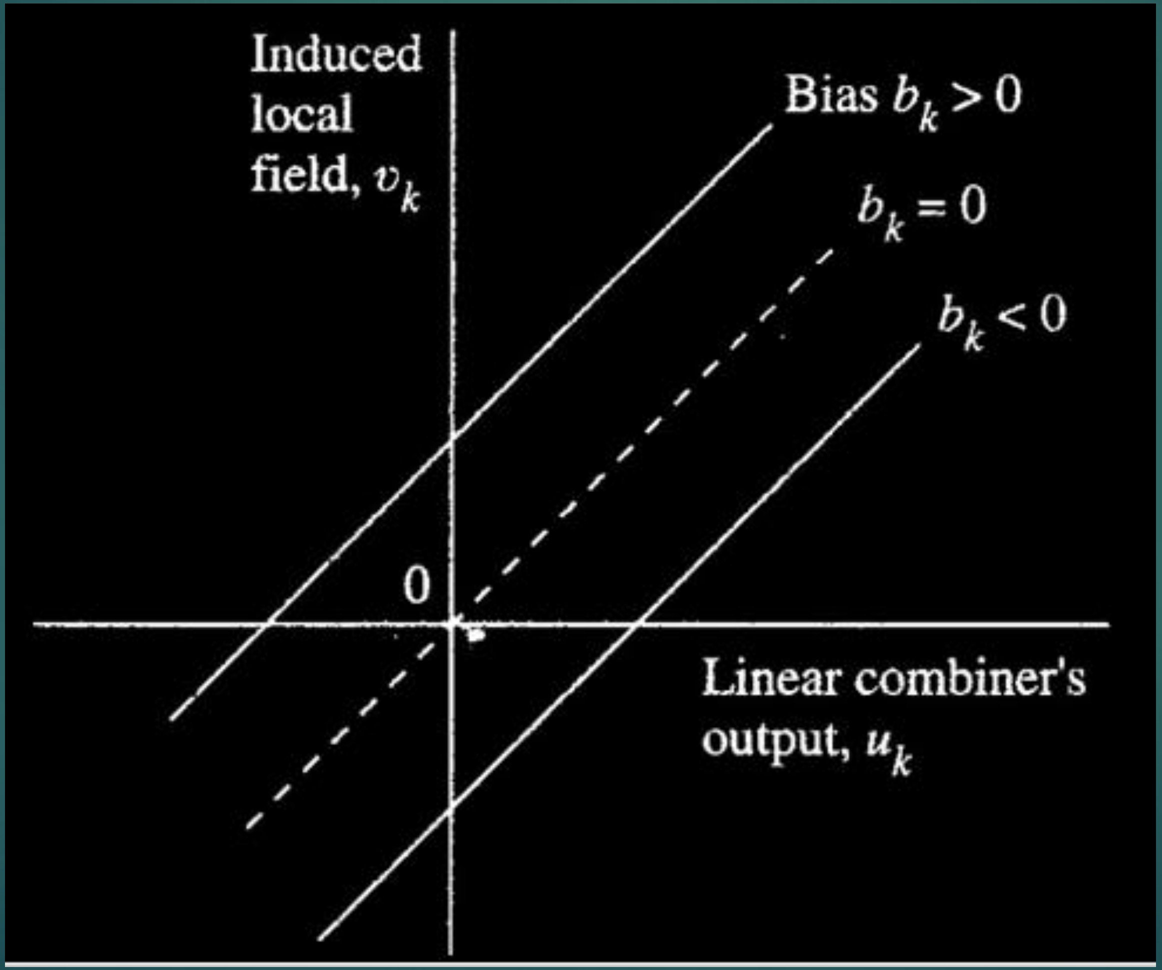
and

$$y_k = \varphi(u_k + b_k)$$

- Where $x_1, x_2, x_3, \ldots . . x_m$ are the input signals and $w_{k1}, w_{k2}, \ldots w_{km}$ are the synaptic weights of neuron *k*; u_k are the linear combiner output due to the input signals; b_k is the bias; φ is the activation function; and y_k is the output signal of the neuron.
- The use of bias b_k applies an offline transformation to the output u_k

$$v_k = u_k + b_k$$

Offline Transformation Produced by the Presence of a bias



Artificial neural network

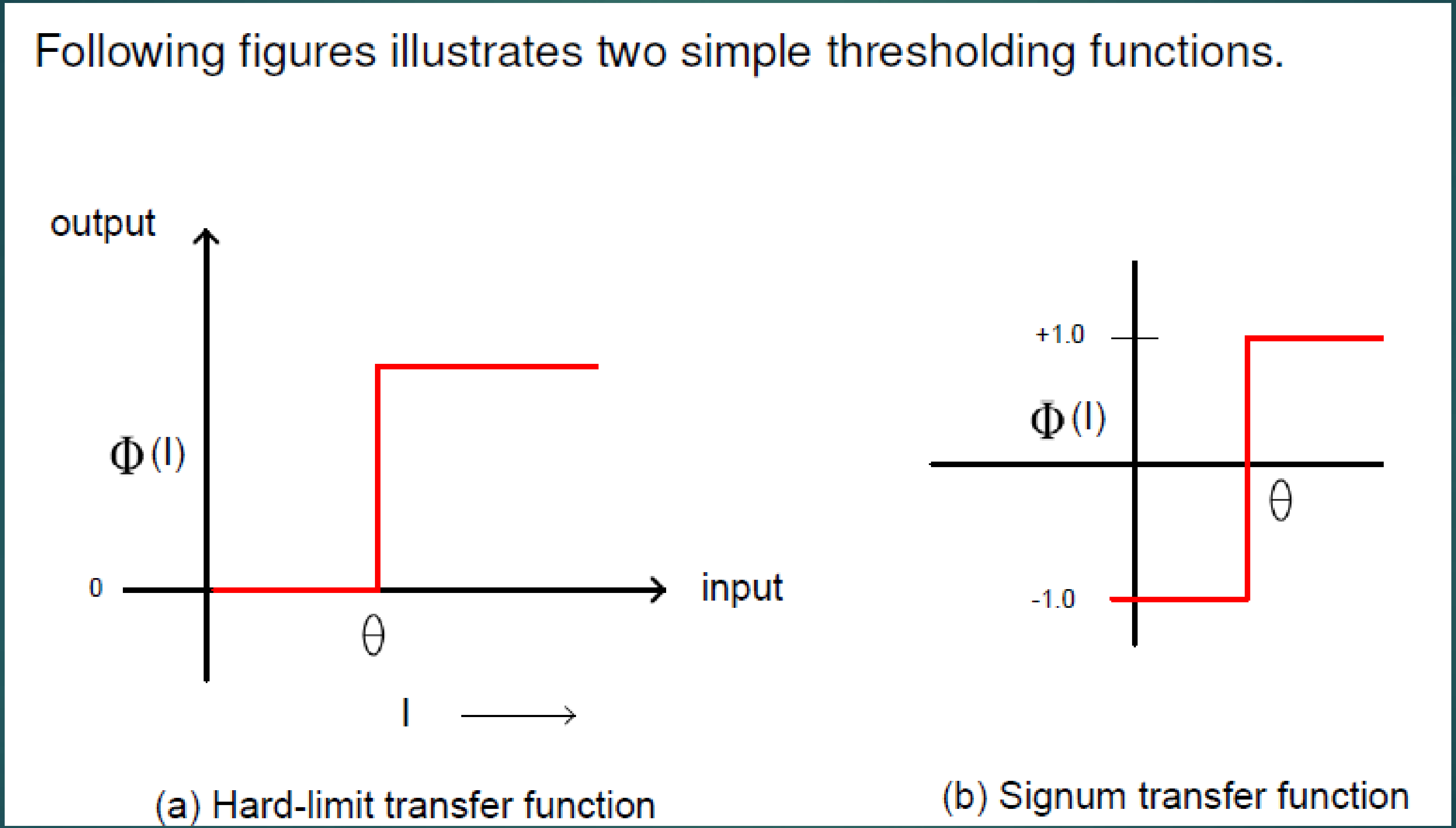
Hard-limit transfer function : The transformation we have just discussed is called hard-limit transfer function. It is generally used in perception neuron. In other words,

$$\phi(I) = \begin{cases} 1 & , \text{if } I > \theta \\ 0 & , \text{if } I \leq \theta \end{cases}$$

Linear transfer function : The output of the transfer function is made equal to its input (normalized) and its lies in the range of -1to +1. It is also known as Signum or Quantizer function and it defined as

$$\phi(I) = \begin{cases} +1 & , \text{if } I > \theta \\ -1 & , \text{if } I \leq \theta \end{cases}$$

Artificial neural network



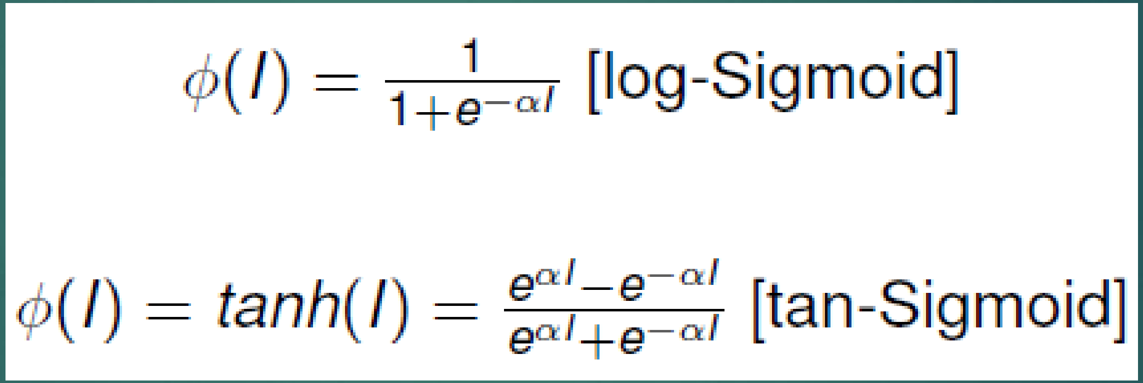
Following figures illustrates two simple thresholding functions.

(a) Hard-limit transfer function

(b) Signum transfer function

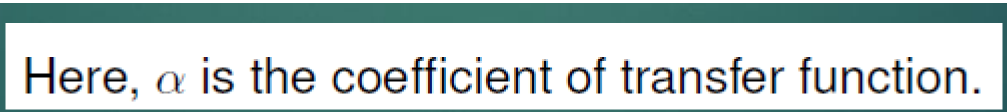
Artificial neural network

Sigmoid transfer function : This function is a continuous function that varies gradually between the asymptotic values 0 and 1 (called log-sigmoid) or -1 and +1 (called Tan-sigmoid) threshold function and is given by



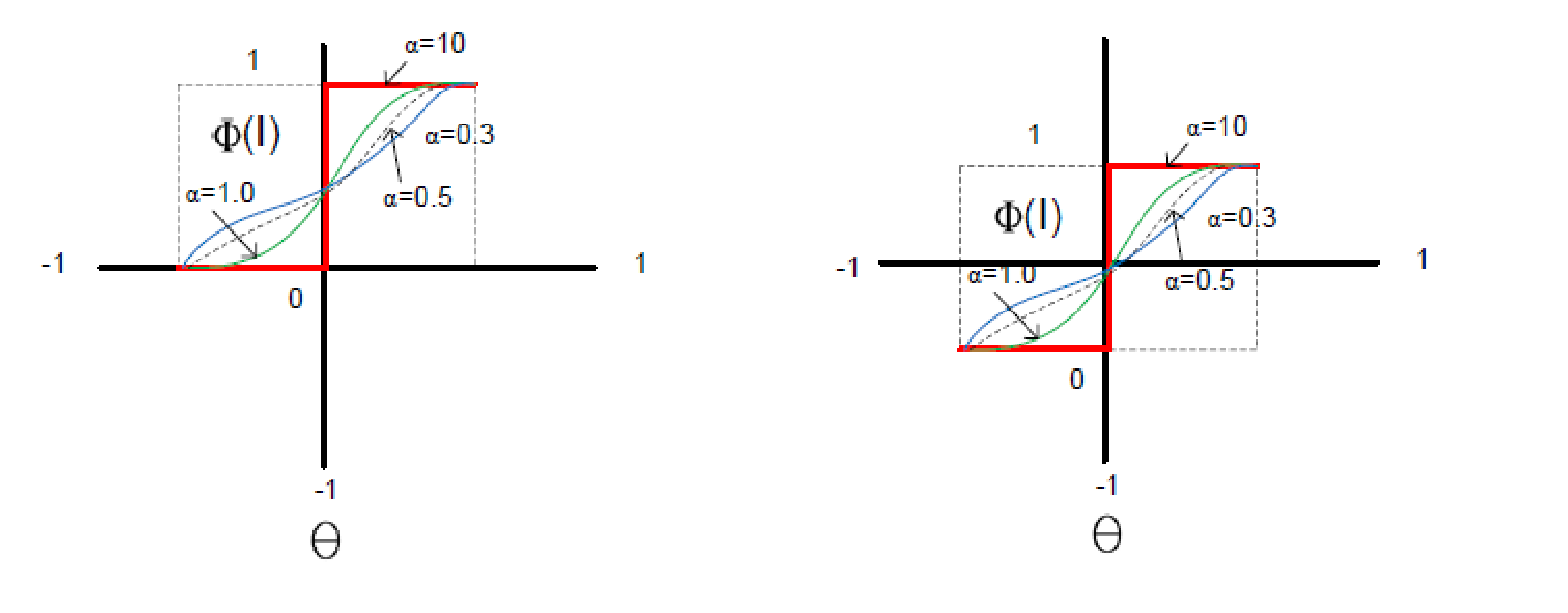
$$\phi(I) = \frac{1}{1+e^{-\alpha I}} \text{ [log-Sigmoid]}$$

$$\phi(I) = tanh(I) = \frac{e^{\alpha I}-e^{-\alpha I}}{e^{\alpha I}+e^{-\alpha I}} \text{ [tan-Sigmoid]}$$



Here, α is the coefficient of transfer function.

Artificial neural network

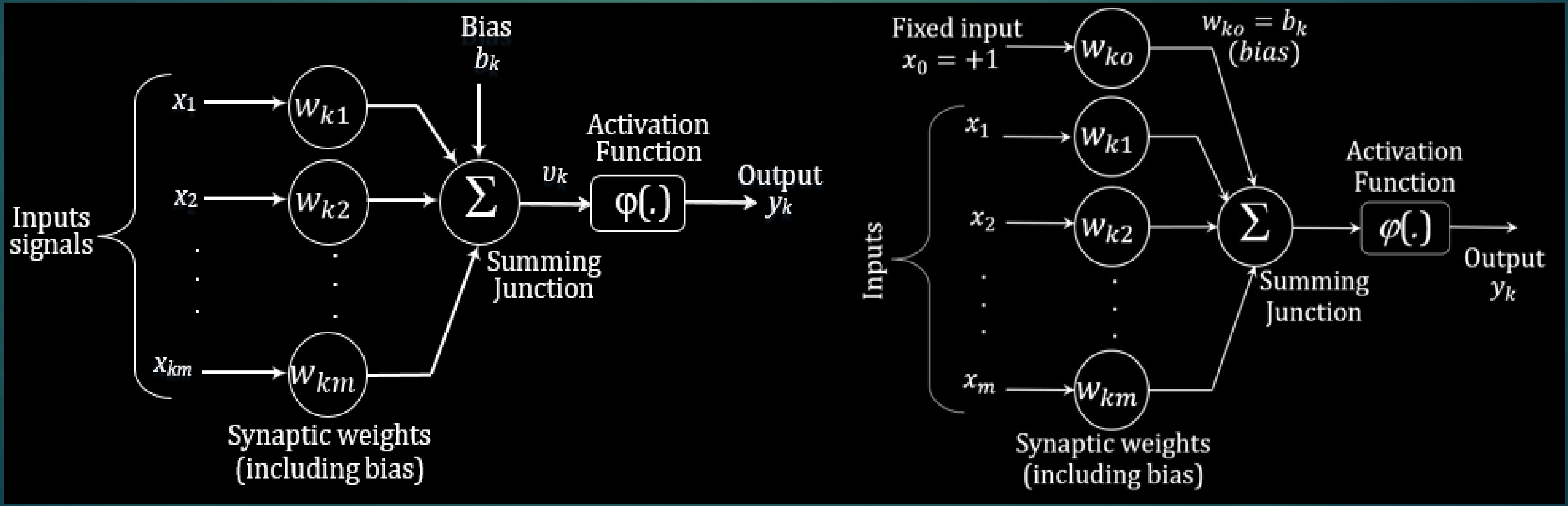


(a) Log-Sigmoid transfer function

(b) Tan-Sigmoid transfer function

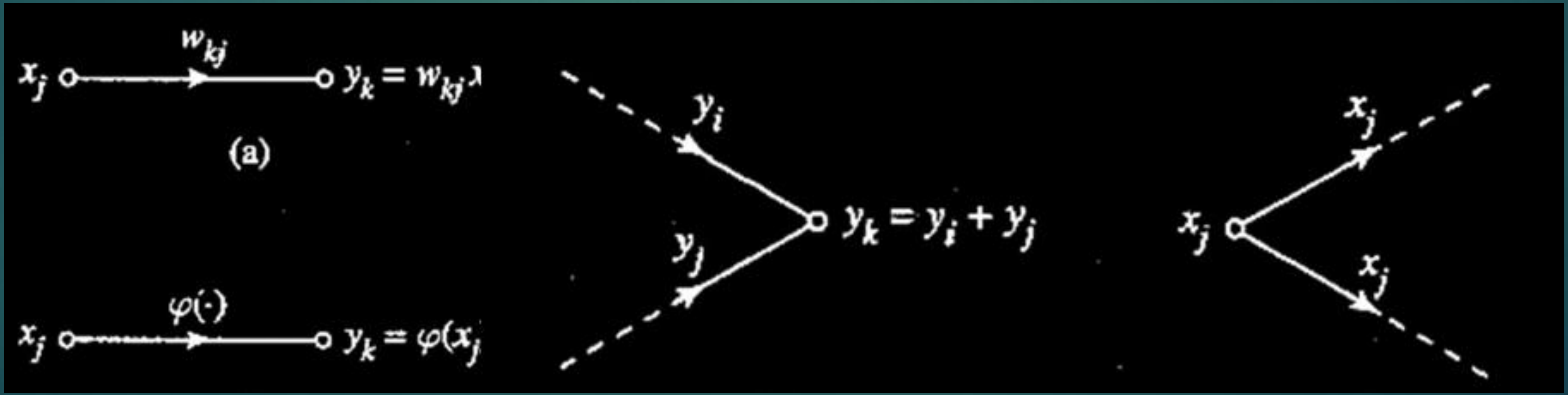
Neural Networks Viewed as Directed Graphs

- The block diagram provides a functional description of the elements that constitute an artificial neuron model.
- We may simplify the appearance of the model by using the signal-flow graphs with a well-defined set of rules.



Neural Networks Viewed as Directed Graphs

- A signal-flow graph is a network of directed links (branches) that are interconnected at certain points called nodes
- A typical node *j* has an associated node signal x_j where a directed link originates at node *j* and terminates on node *k*
- it has an associated transfer function or transmittance that specifies the manner in which the signal y_k at node *k* depends on the signal x_j at node *j*.



(a) (b) (c)

Soft Computing CS125

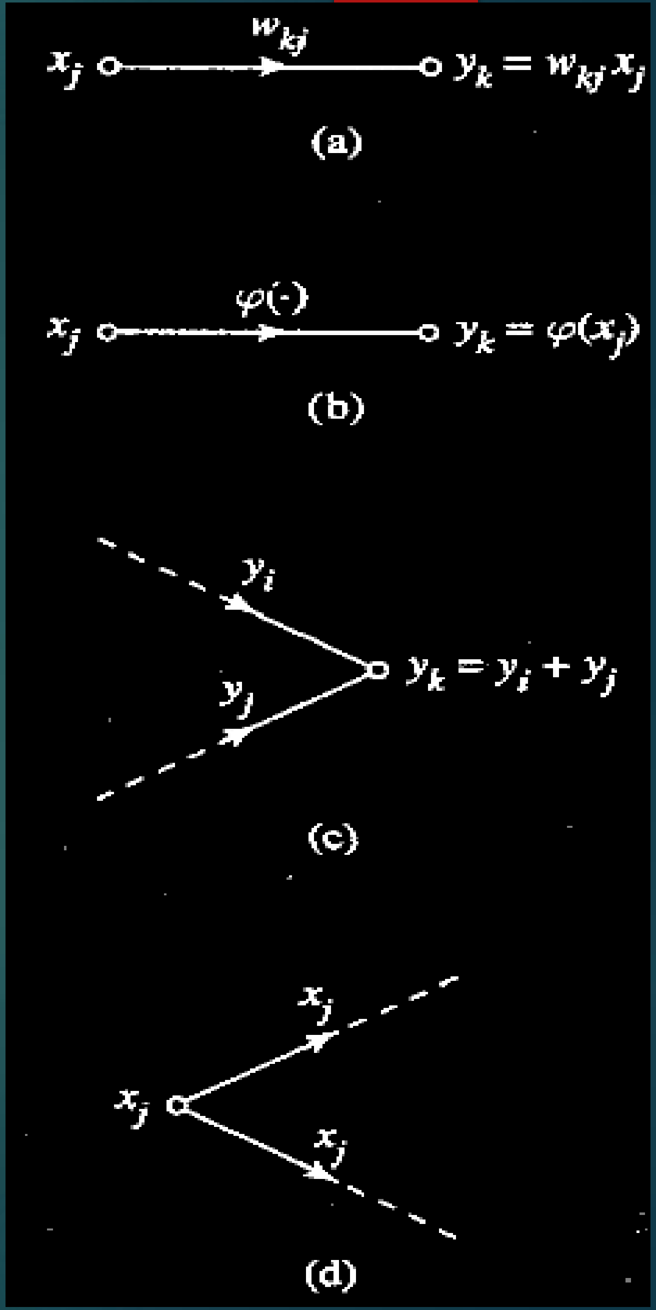
Signal-Flow Graph Rules

- **Rule 1:** A signal flows along a link only in the direction defined by the arrow on the link. There are two different types of links

  a) Synaptic links, whose behavior is governed by a linear input-output relation. Specifically, the node signal x_j is multiplied by the synaptic weight w_{kj} to produce the node signal y_k as illustrated in Fig. a.

  b) Activation links, whose behavior is governed in general by nonlinear input-output relation.

- **Rule 2:**A node signal equals the algebraic sum of all signals entering the pertinent node via the incoming links.
- **Rule 3:** The signal at a node is transmitted to each outgoing link originating from that node, with the transmission being entirely independent of transfer functions of outgoing links

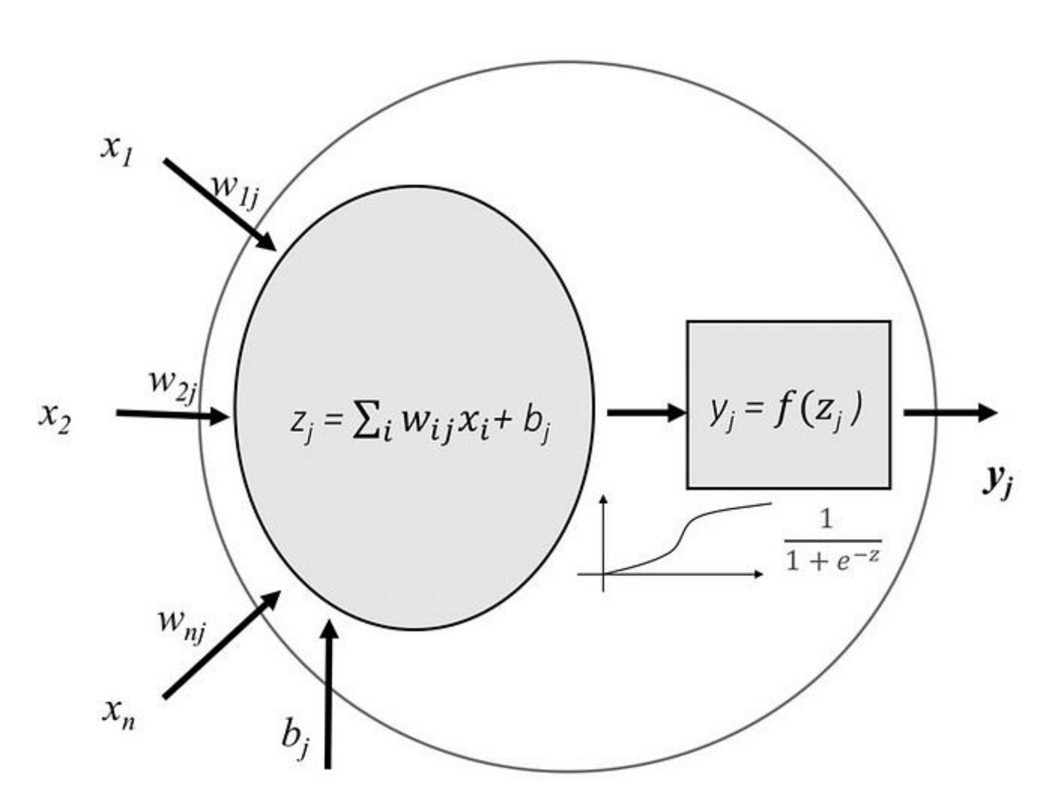


Advantages of ANN

- ANNs exhibits **mapping capabilities**, that is, they can map input patterns to their associated output pattern.
- The ANNs **learn by examples**. Thus, an ANN architecture can be trained with known example of a problem before they are tested for their inference capabilities on unknown instance of the problem. In other words, they can identify new objects previous untrained.
- The ANNs posses the capability to generalize. This is the power to apply in application where exact mathematical model to problem are not possible.
- The ANNs are **robust system and fault tolerant.** They can therefore, recall full patterns from incomplete, partial or noisy patterns.
- The ANNS can process information in parallel, at **high speed** and in a distributed manner. Thus a massively parallel distributed processing system made up of highly interconnected (artificial) neural computing elements having ability to learn and acquire knowledge is possible.

Learning process of a neural network

A neural network is made up of **neurons** connected to each other; at the same time, each connection of our neural network is associated with a weight that dictates the importance of this relationship in the neuron when multiplied by the input value.

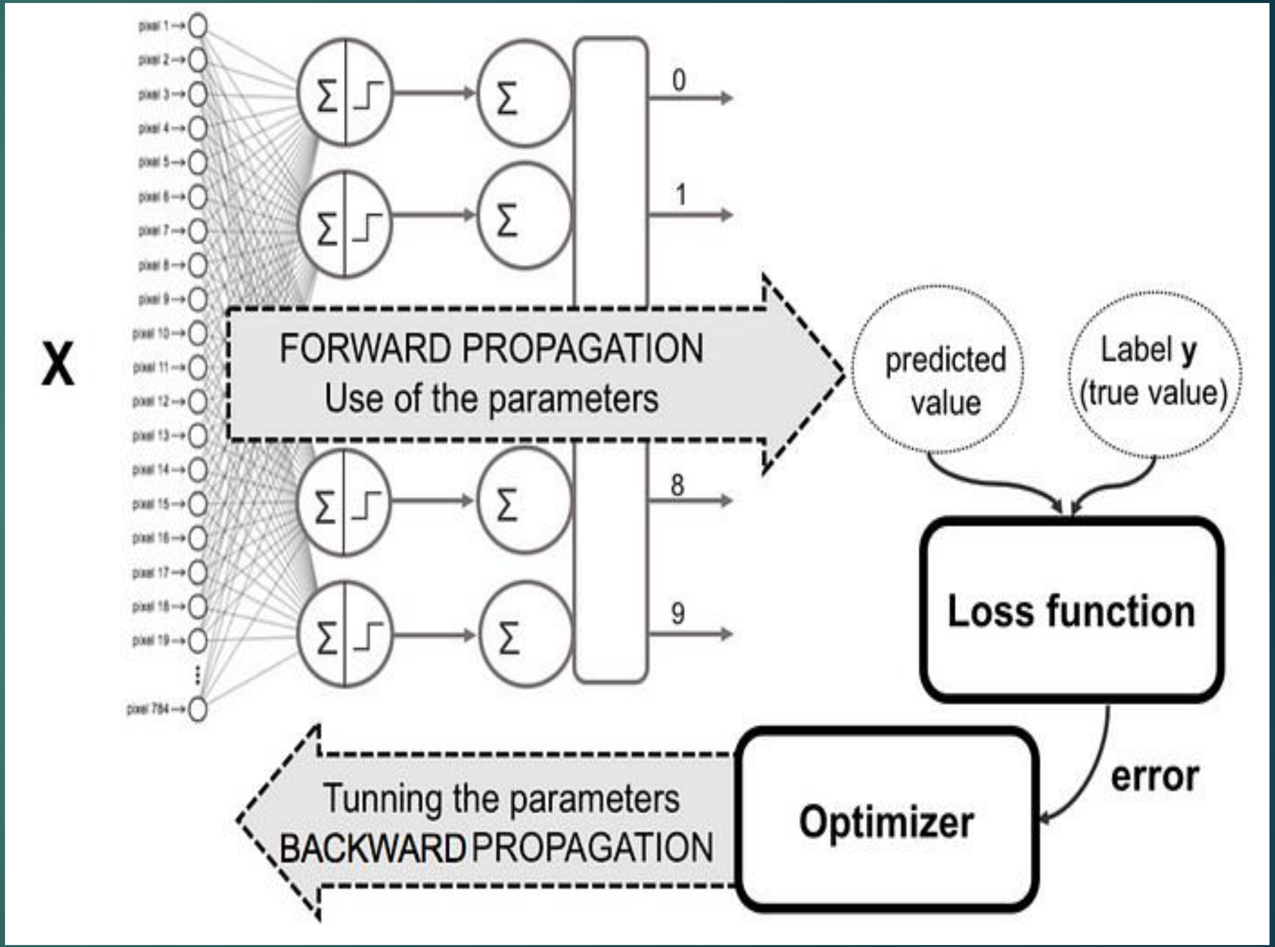


Learning process of a neural network

- Training the neural network, that is, learning the values of our parameters (weights *wij* and *bj* biases) is the most genuine part of Deep Learning.
- Learning process in a neural network as an iterative process of “going and return” by the layers of neurons.
- The “going” is a forward propagation of the information and the “return” is a backpropagation of the information.

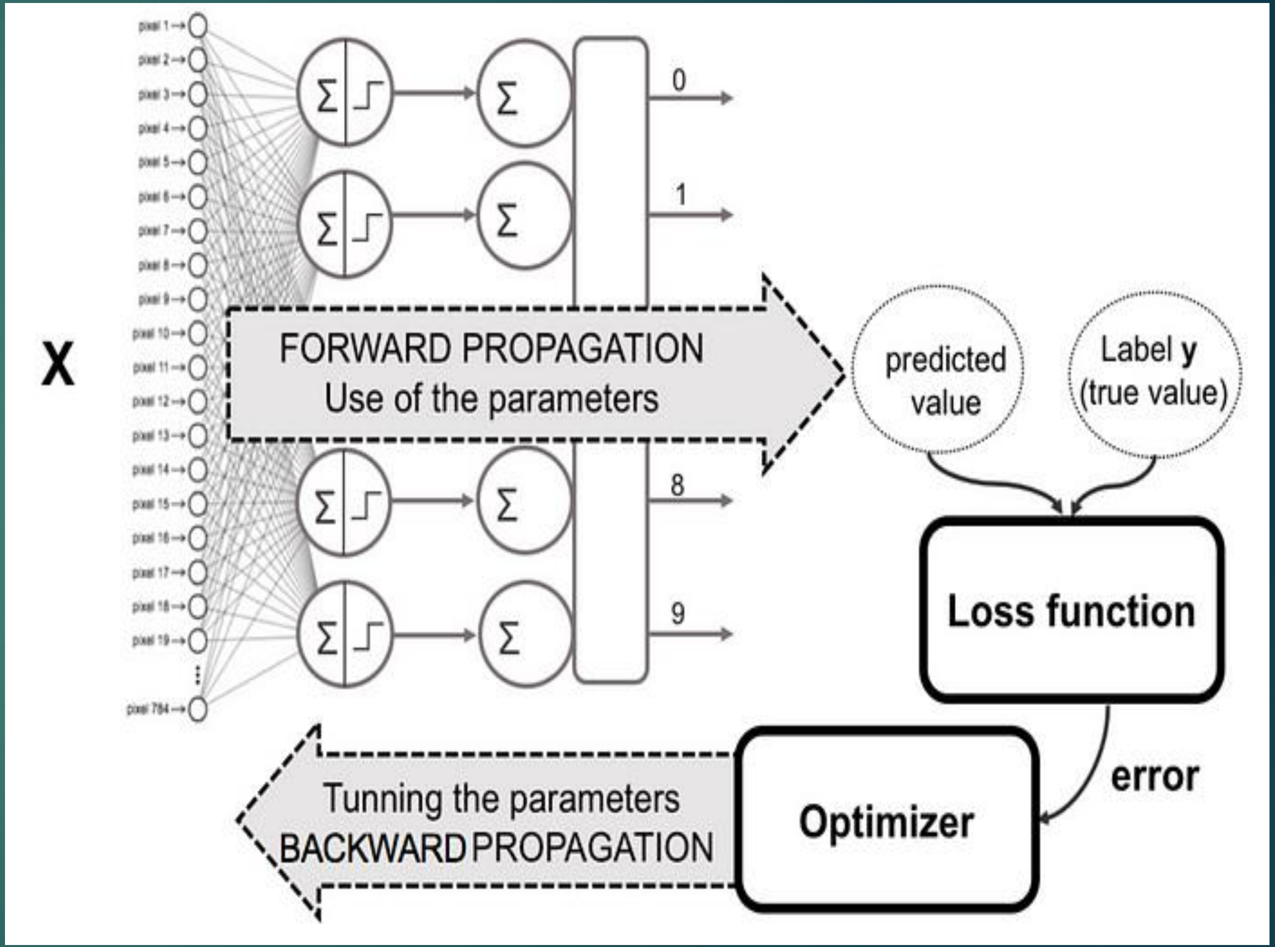
Learning process of a neural network

- The first phase **forward propagation** occurs when the network is exposed to the training data and these cross the entire neural network for their predictions (labels) to be calculated.
- That is, passing the input data through the network in such a way that all the neurons apply their transformation to the information they receive from the neurons of the previous layer and sending it to the neurons of the next layer.
- When the data has crossed all the layers, and all its neurons have made their calculations, the final layer will be reached with a result of label prediction for those input examples.



Learning process of a neural network

- Next, a **loss function** is used to estimate the loss (or error) and to compare and measure how good/bad the prediction result was in relation to the correct result (remember that we are in a supervised learning environment and we have the label that tells us the expected value).
- Therefore, as the model is being trained, the weights of the interconnections of the neurons will gradually be adjusted until good predictions are obtained.



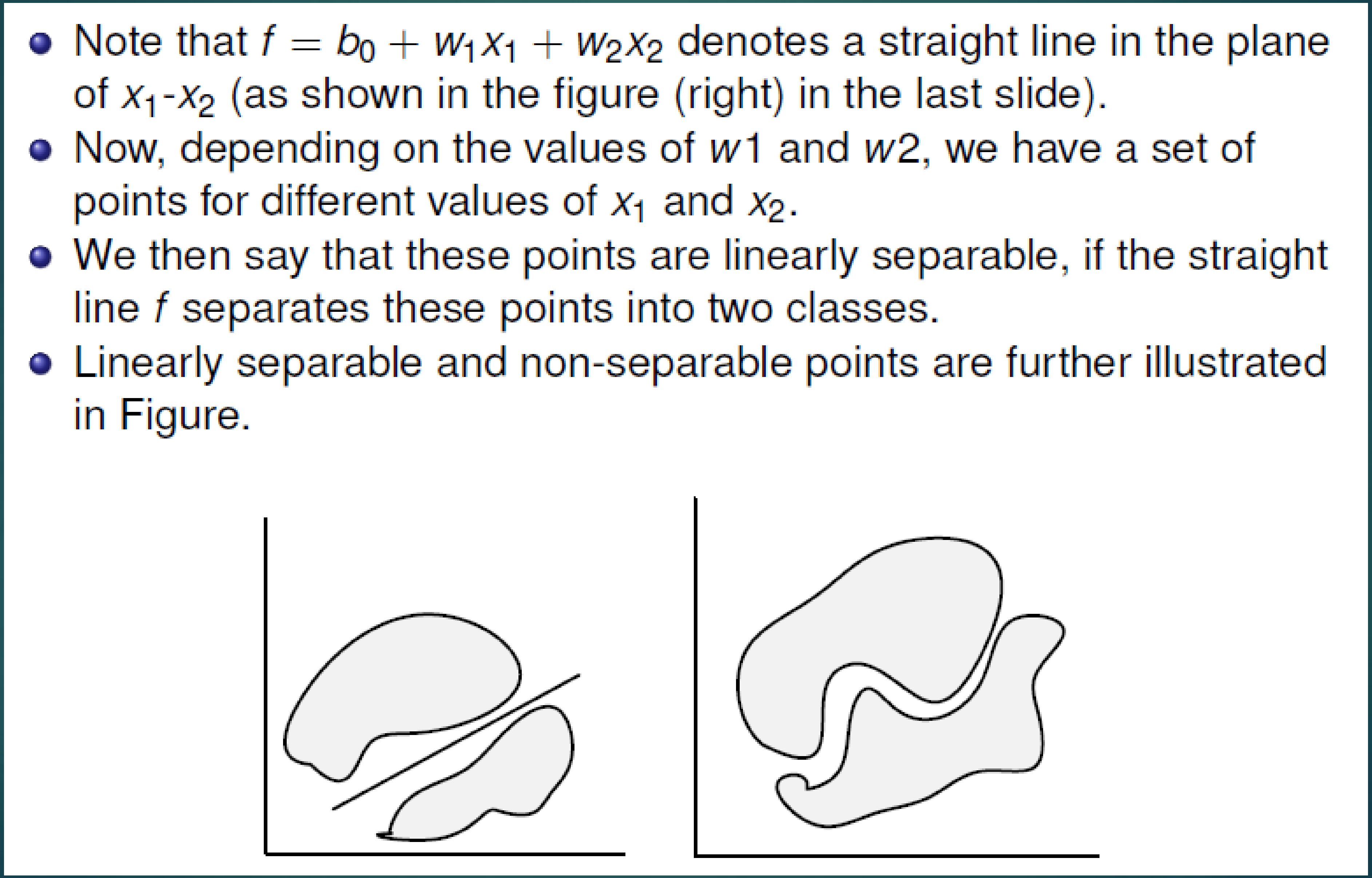
Learning process of a neural network

- Once the loss has been calculated, this information is propagated backwards. Hence, its name: **backpropagation**.
- Starting from the output layer, that loss information propagates to all the neurons in the hidden layer that contribute directly to the output.
- However, the neurons of the hidden layer only receive a fraction of the total signal of the loss, based on the relative contribution that each neuron has contributed to the original output.
- This process is repeated, layer by layer, until all the neurons in the network have received a loss signal that describes their relative contribution to the total loss.

Learning process of a neural network

- Now that we have spread this information back, we can adjust the weights of connections between neurons. What we are doing is making the loss as close as possible to zero the next time we go back to using the network for a prediction.
- For this, we will use a technique called **gradient descent**.
- This technique changes the weights in small increments with the help of the calculation of the derivative (or gradient) of the loss function, which allows us to see in which direction “to descend” towards the global minimum; this is done in general in batches of data in the successive iterations (epochs) of all the dataset that we pass to the network in each iteration.

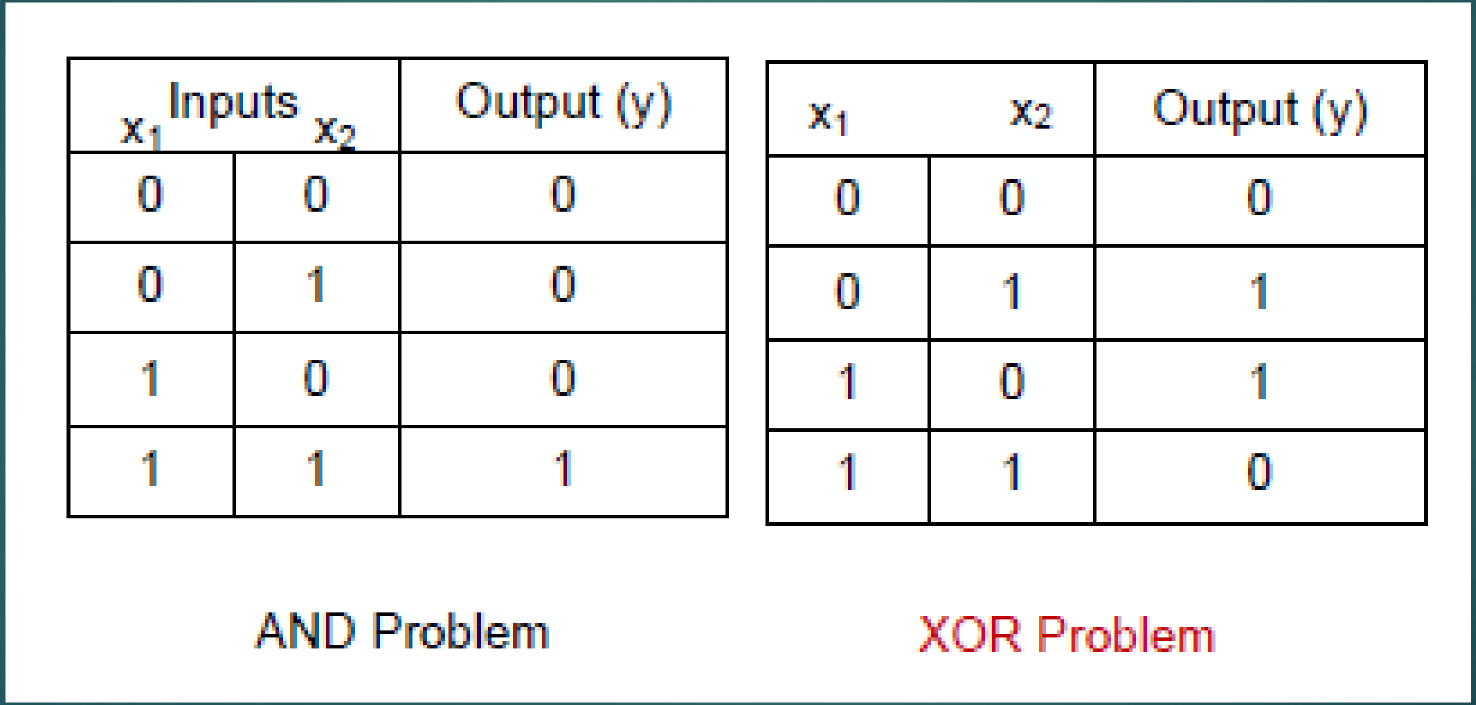
Revisit of a single neural network



- Note that $f = b_0 + w_1x_1 + w_2x_2$ denotes a straight line in the plane of x_1-x_2 (as shown in the figure (right) in the last slide).
- Now, depending on the values of $w1$ and $w2$, we have a set of points for different values of x_1 and x_2.
- We then say that these points are linearly separable, if the straight line f separates these points into two classes.
- Linearly separable and non-separable points are further illustrated in Figure.

AND and XOR problems

To illustrate the concept of linearly separable and non separable tasks to be accomplished by a neural network, let us consider the case of AND problem and XOR problem.



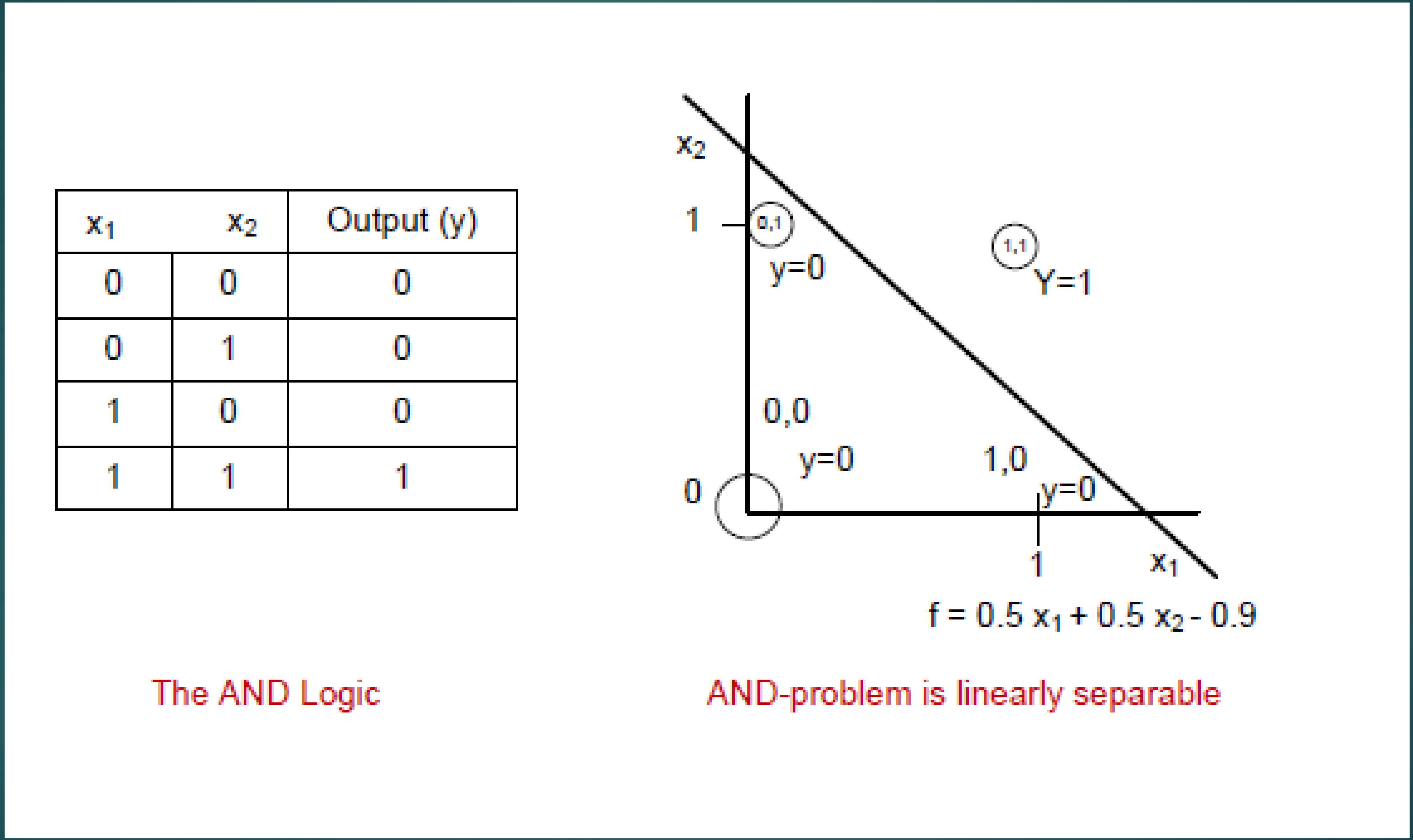
Inputs		Output (y)
x_1	x_2	
0	0	0
0	1	0
1	0	0
1	1	1

AND Problem

x_1	x_2	Output (y)
0	0	0
0	1	1
1	0	1
1	1	0

XOR Problem

AND problem is linearly separable



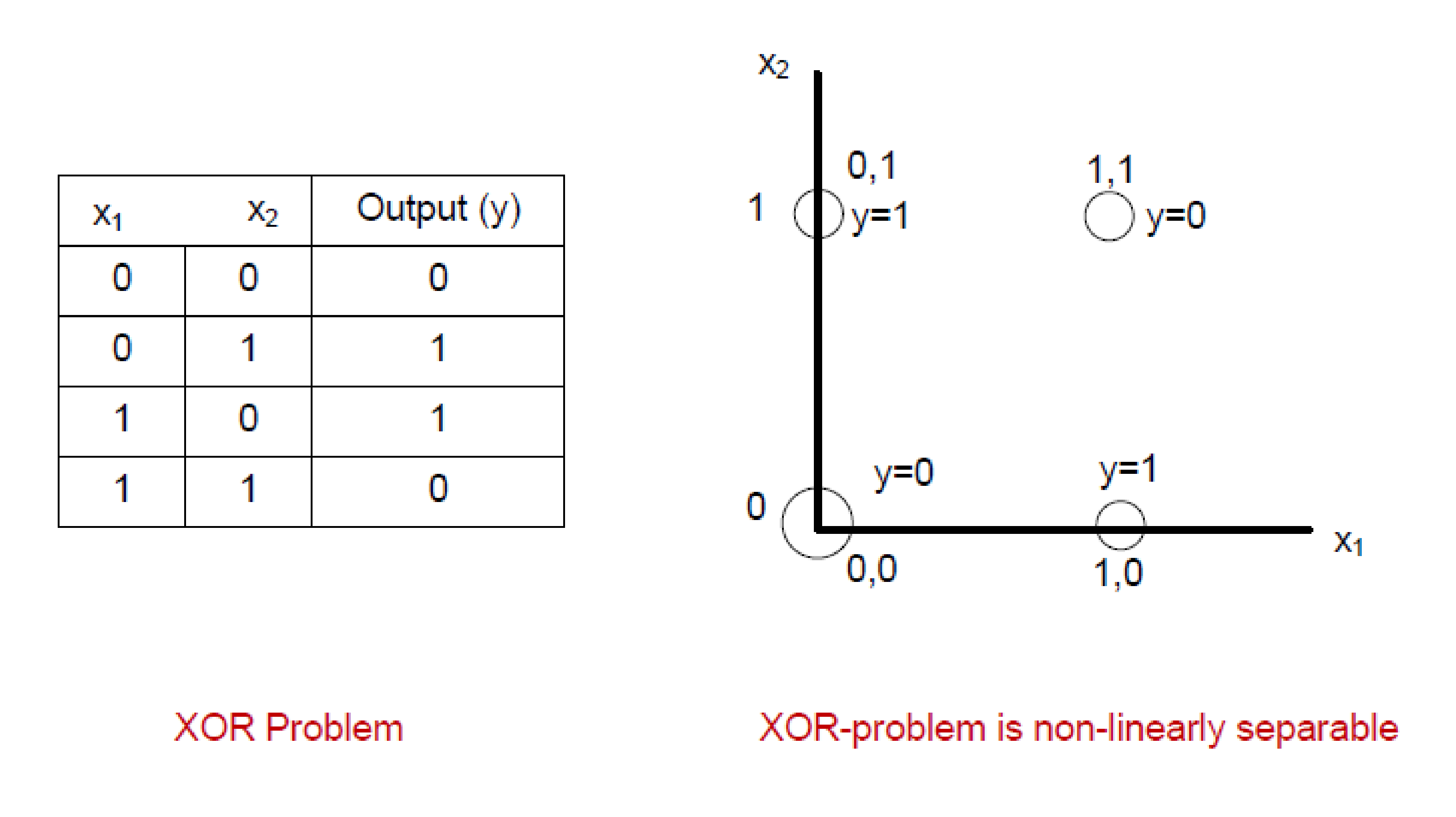
x_1	x_2	Output (y)
0	0	0
0	1	0
1	0	0
1	1	1

The AND Logic

$f = 0.5\,x_1 + 0.5\,x_2 - 0.9$

AND-problem is linearly separable

XOR problem is non-linearly separable



x_1	x_2	Output (y)
0	0	0
0	1	1
1	0	1
1	1	0

XOR Problem

XOR-problem is non-linearly separable

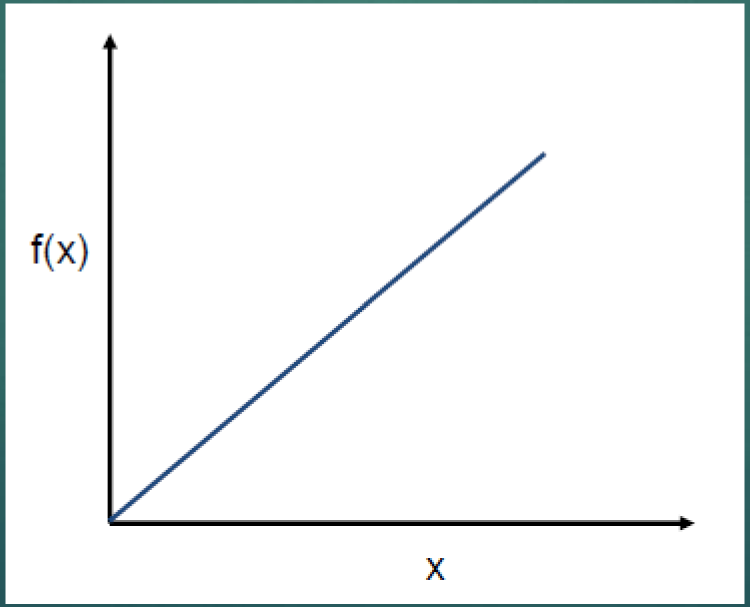
Activation functions in ANN

- The activation function decides whether a neuron should be activated or not by calculating the weighted sum and further adding bias to it.
- The purpose of the activation function is to introduce non-linearity into the output of a neuron.
- A neural network without an activation function is essentially just a linear regression model.
- The activation function does the non-linear transformation to the input making it capable to learn and perform more complex tasks.
- Types of activation functions like Linear, Sigmoid, Tanh, ReLU, etc.

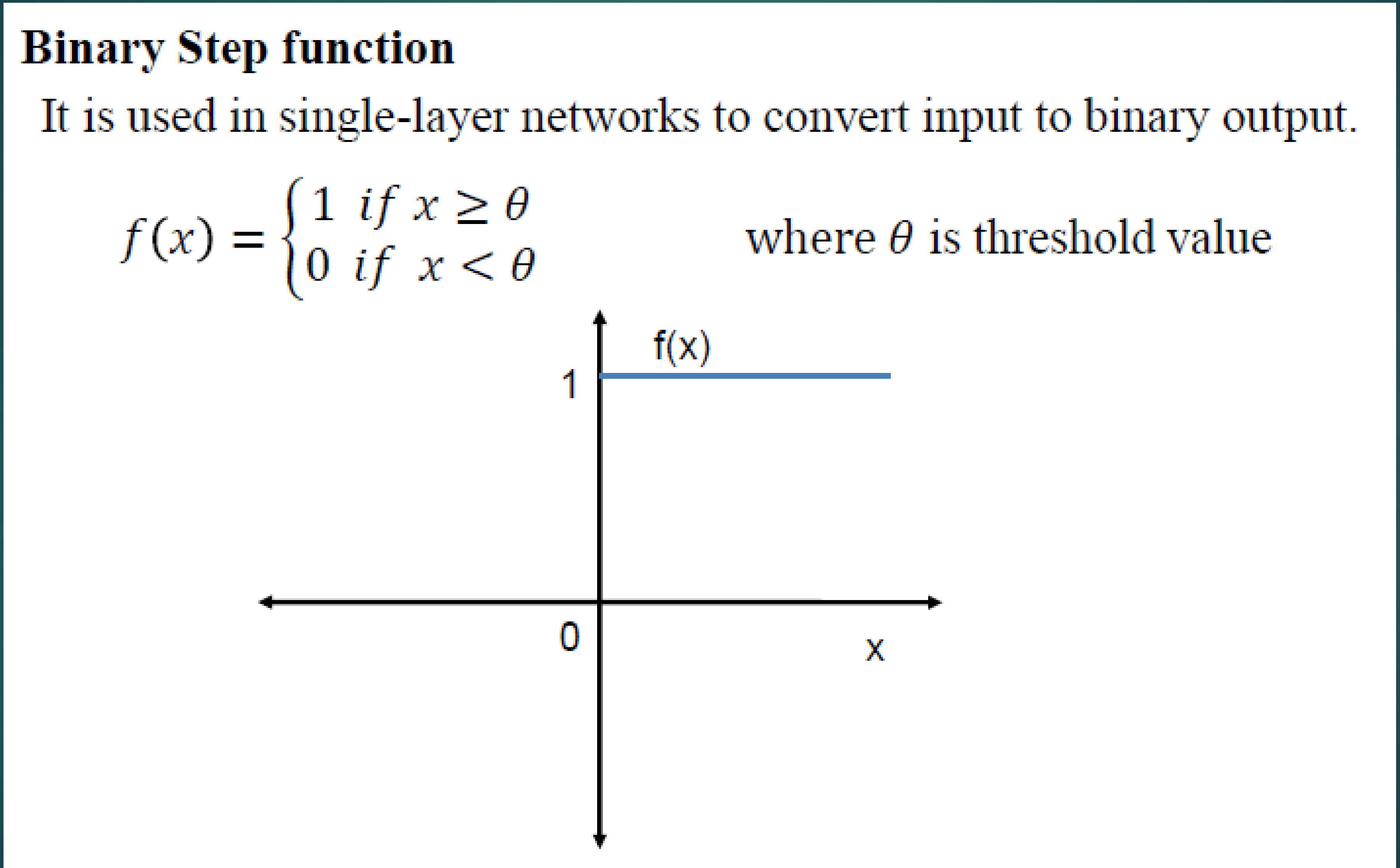
Linear/Identity Activation Function

- The linear activation function is basically the identity function in which, in practical terms, it means that the signal does not change. It’s a linear function. The output is same as input.

$$f(x) = x \quad \text{for all } x$$



Binary Step Function



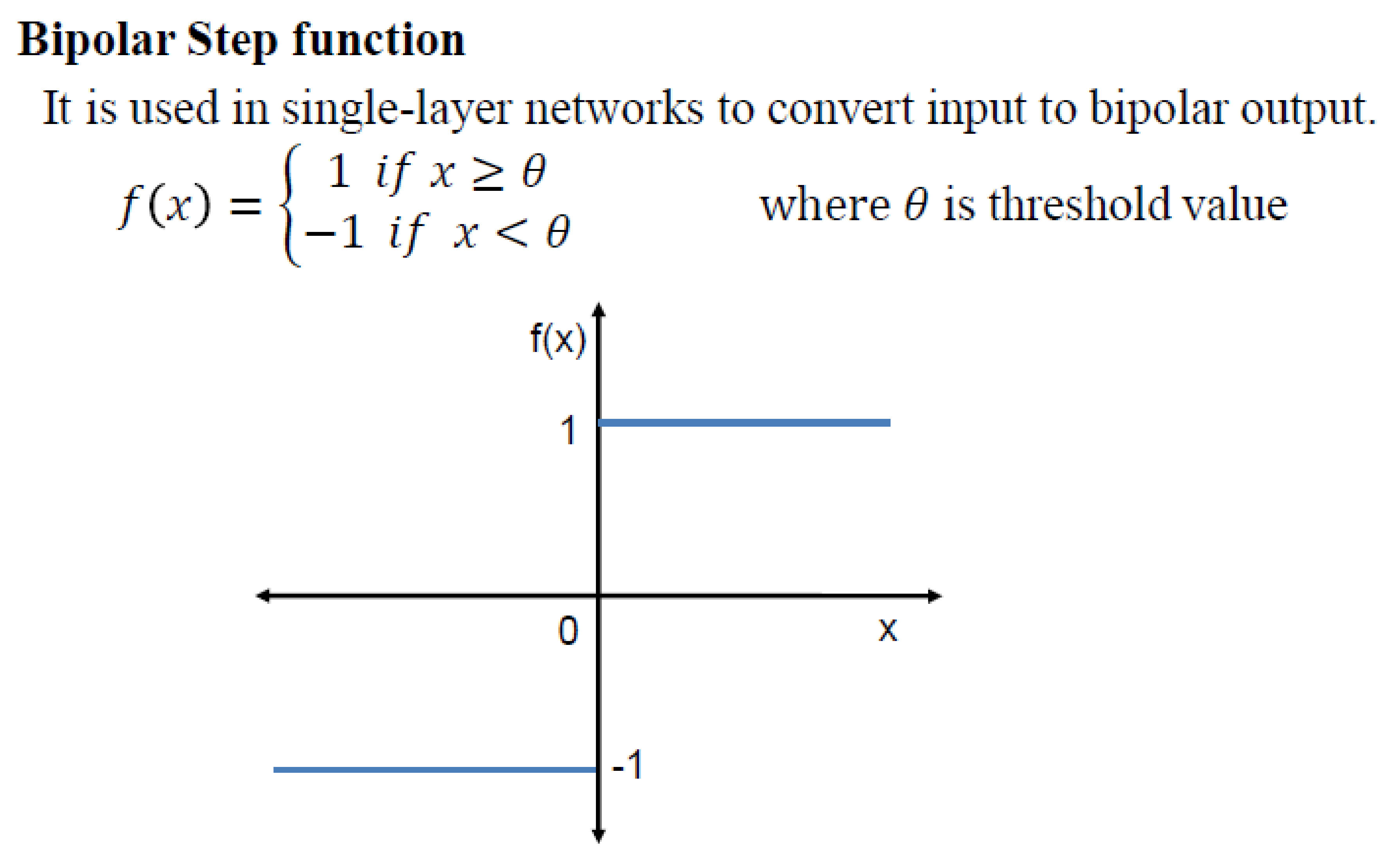
Binary Step function

It is used in single-layer networks to convert input to binary output.

$$f(x) = \begin{cases} 1 \quad if \; x \geq \theta \\ 0 \quad if \;\; x < \theta \end{cases}$$

where θ is threshold value

Bipolar Step Function

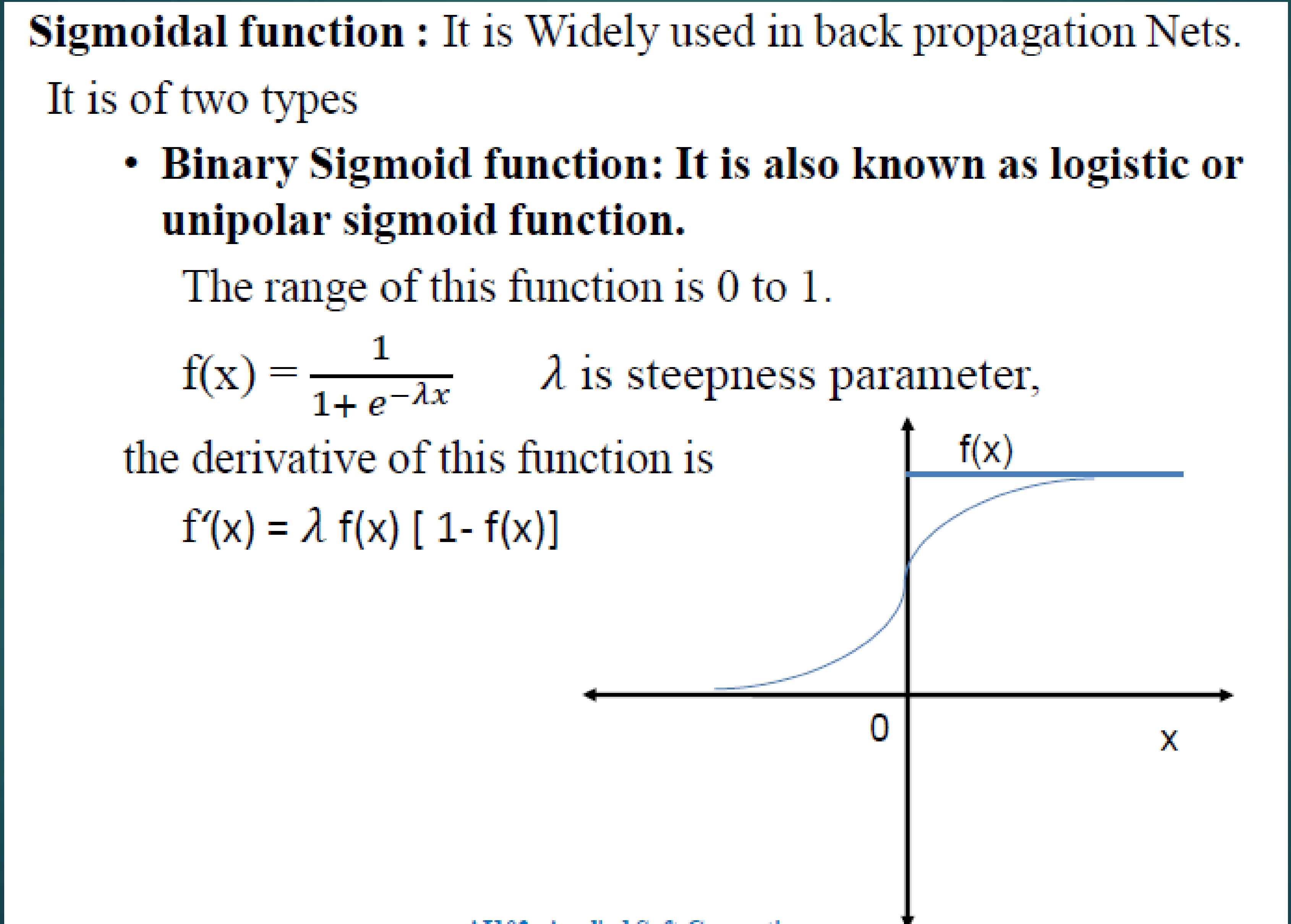


Bipolar Step function

It is used in single-layer networks to convert input to bipolar output.

$$f(x) = \begin{cases} 1 \ \ if \ x \geq \theta \\ -1 \ \ if \ \ x < \theta \end{cases}$$ where θ is threshold value

Sigmoid Function



Sigmoidal function : It is Widely used in back propagation Nets.

It is of two types

- **Binary Sigmoid function: It is also known as logistic or unipolar sigmoid function.**

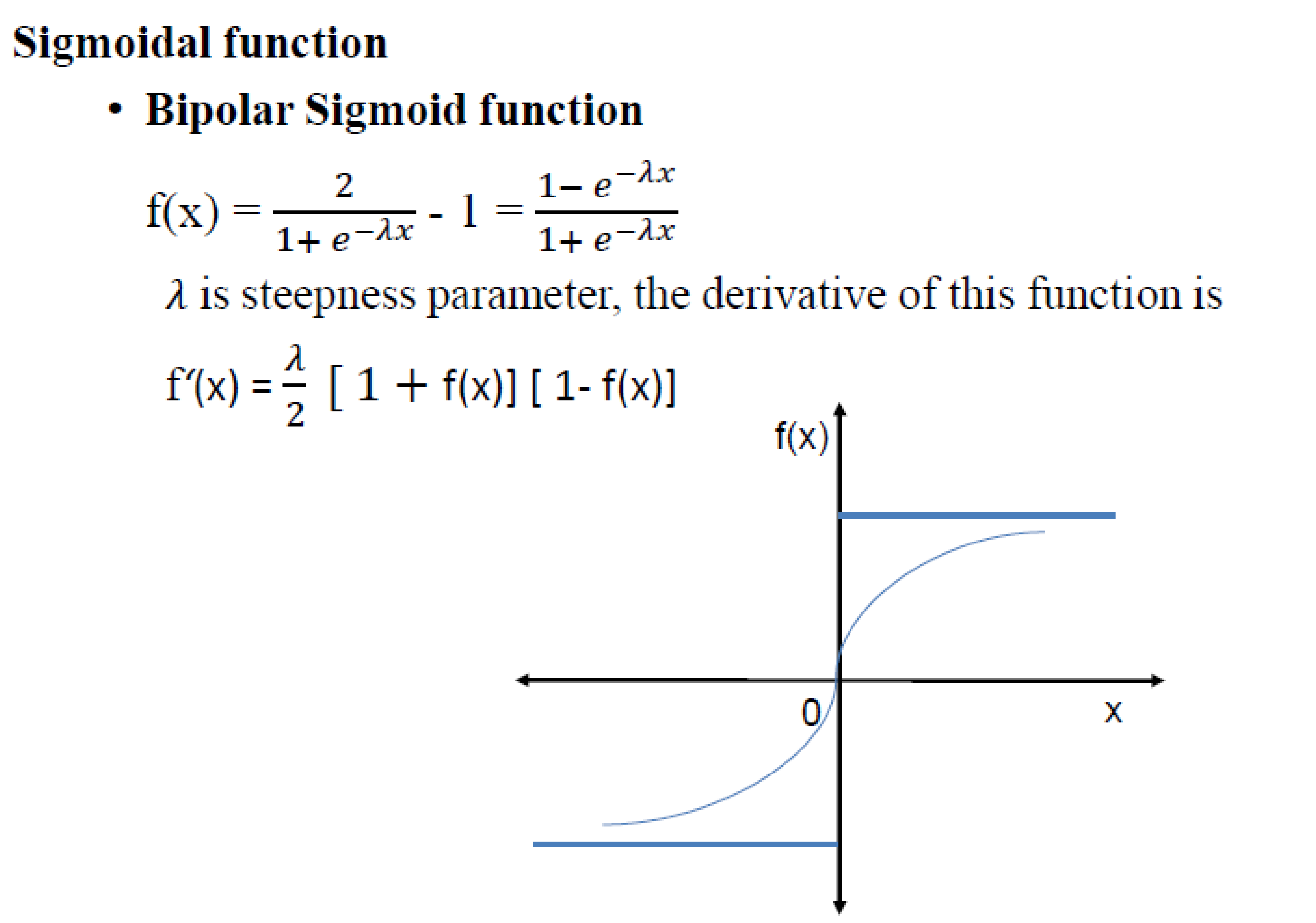
The range of this function is 0 to 1.

$f(x) = \dfrac{1}{1+ e^{-\lambda x}}$ λ is steepness parameter,

the derivative of this function is

$f'(x) = \lambda\, f(x)\, [\, 1- f(x)]$

Sigmoid Function



Sigmoidal function

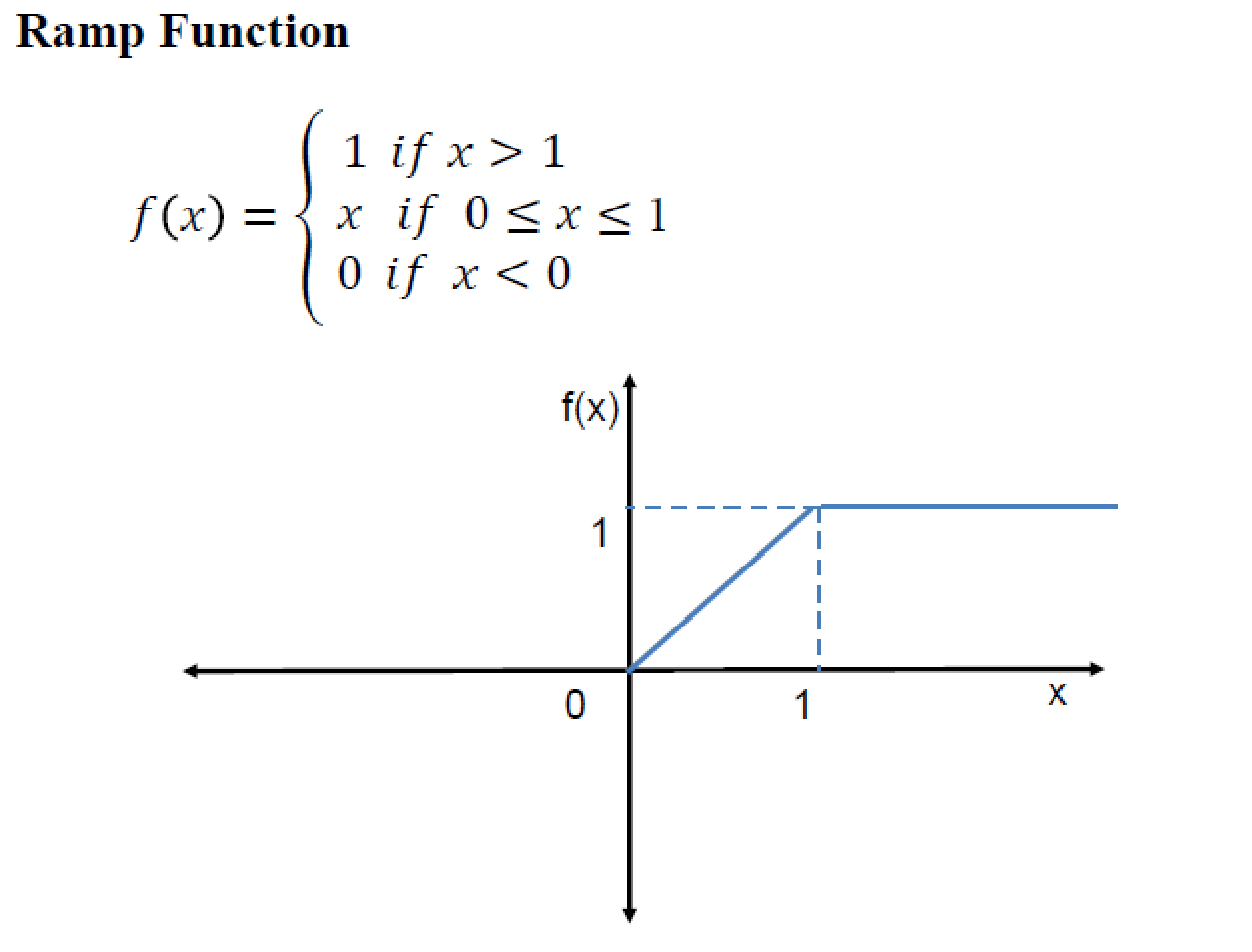
- **Bipolar Sigmoid function**

$$f(x) = \frac{2}{1+e^{-\lambda x}} - 1 = \frac{1-e^{-\lambda x}}{1+e^{-\lambda x}}$$

λ is steepness parameter, the derivative of this function is

$$f'(x) = \frac{\lambda}{2}\,[\,1 + f(x)]\,[\,1 - f(x)]$$

Ramp Function

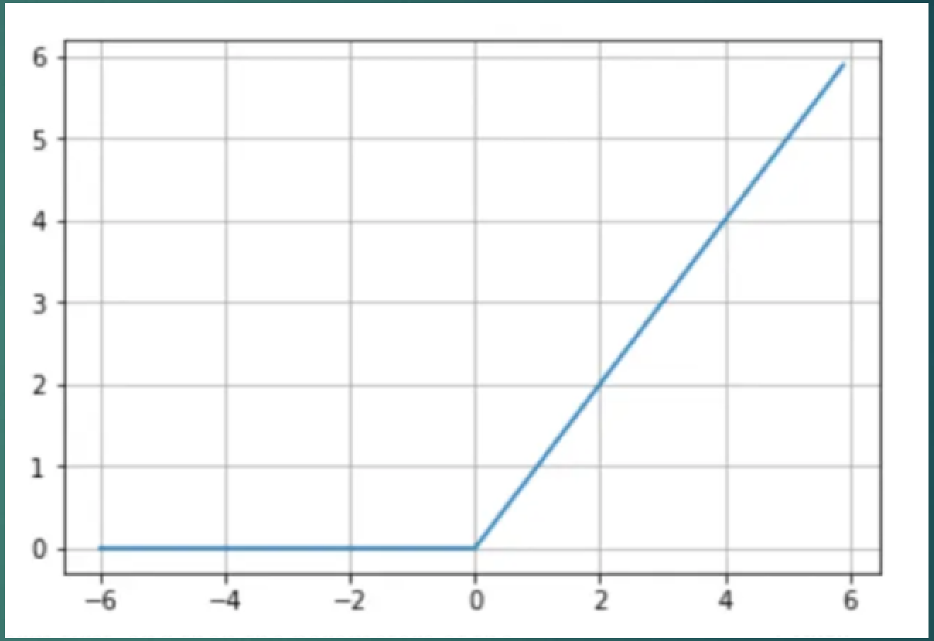


Ramp Function

$$f(x) = \begin{cases} 1 \ \ if \ x > 1 \\ x \ \ if \ \ 0 \le x \le 1 \\ 0 \ \ if \ \ x < 0 \end{cases}$$

ReLU activation function

- The activation function rectified linear unit (ReLU) is a very interesting transformation that activates a single node if the input is above a certain threshold.
- The default and more usual behaviour is that, as long as the input has a value below zero, the output will be zero but, when the input rises above, the output is a linear relationship with the input variable of the form *f(x)*=x.
- The ReLU activation function has proven to work in many different situations and is currently widely used.



Neural network architectures

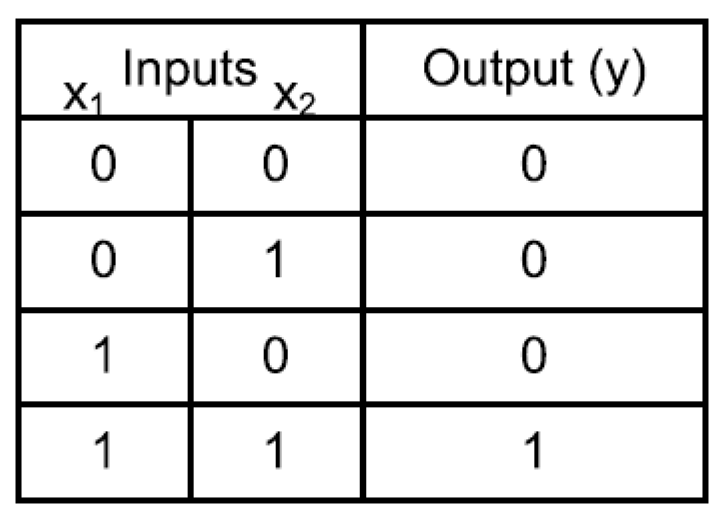
There are three fundamental classes of ANN architectures:

- Single layer feed forward architecture
- Multilayer feed forward architecture
- Recurrent networks architecture

Before going to discuss all these architectures, we first discuss the mathematical details of a neuron at a single level. To do this, let us first consider the AND problem and its possible solution with neural network.

The AND problem and its Neural network

- The simple Boolean AND operation with two input variables x1 and x2 is shown in the truth table.
- Here, we have four input patterns: 00, 01, 10 and 11.
- For the first three patterns output is 0 and for the last pattern output is 1.

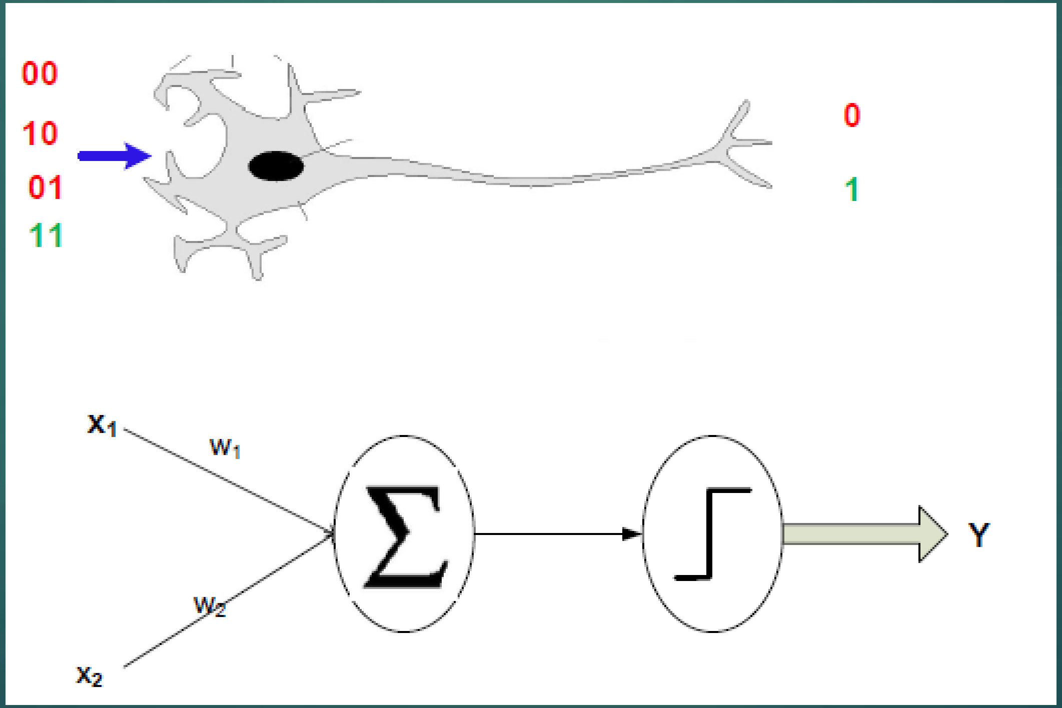


Inputs		Output (y)
x_1	x_2	
0	0	0
0	1	0
1	0	0
1	1	1

Soft Computing CS125

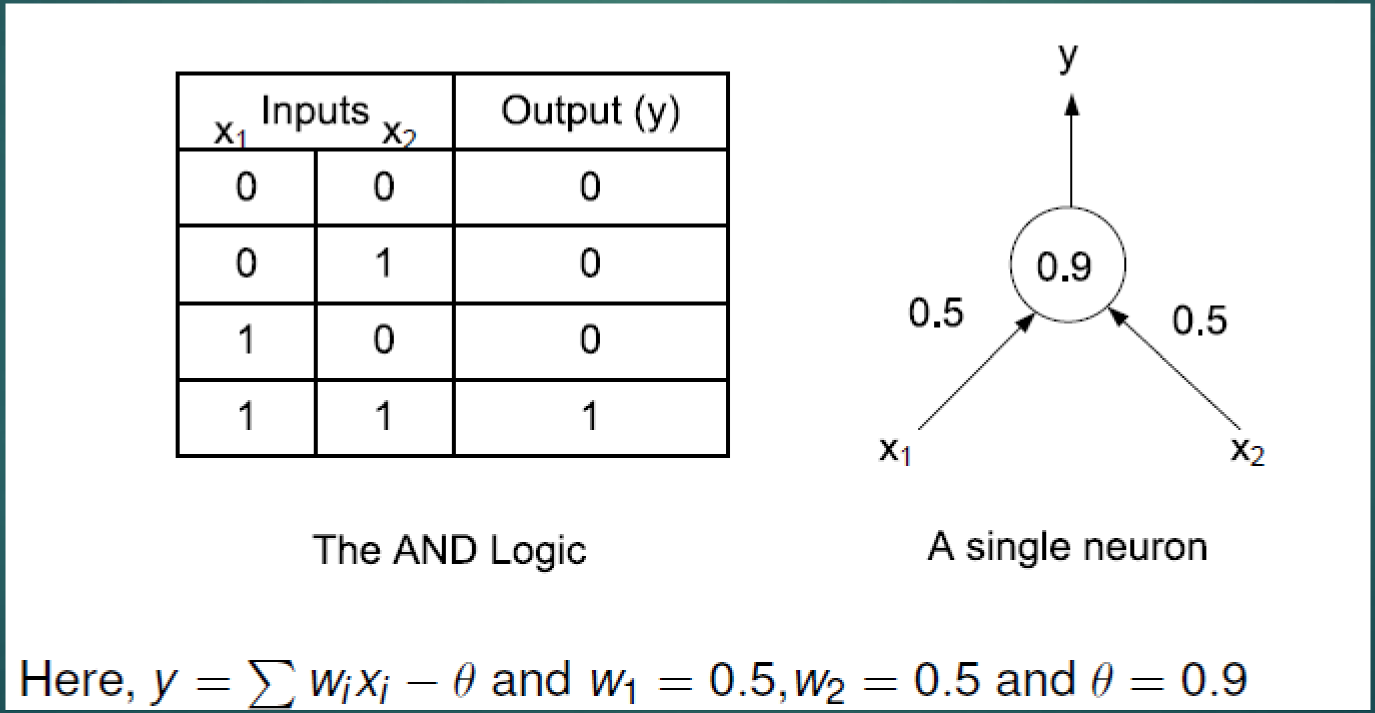
The AND problem and its Neural network

- Alternatively, the AND problem can be thought as a perception problem where we have to receive four different patterns as input and perceive the results as 0 or 1.



The AND problem and its Neural network

- A possible neuron specification to solve the AND problem is given in the following. In this solution, when the input is 11, the weight sum exceeds the threshold (θ = 0.9) leading to the output 1 else it gives the output 0.



Inputs x_1	 x_2	Output (y)
0	0	0
0	1	0
1	0	0
1	1	1

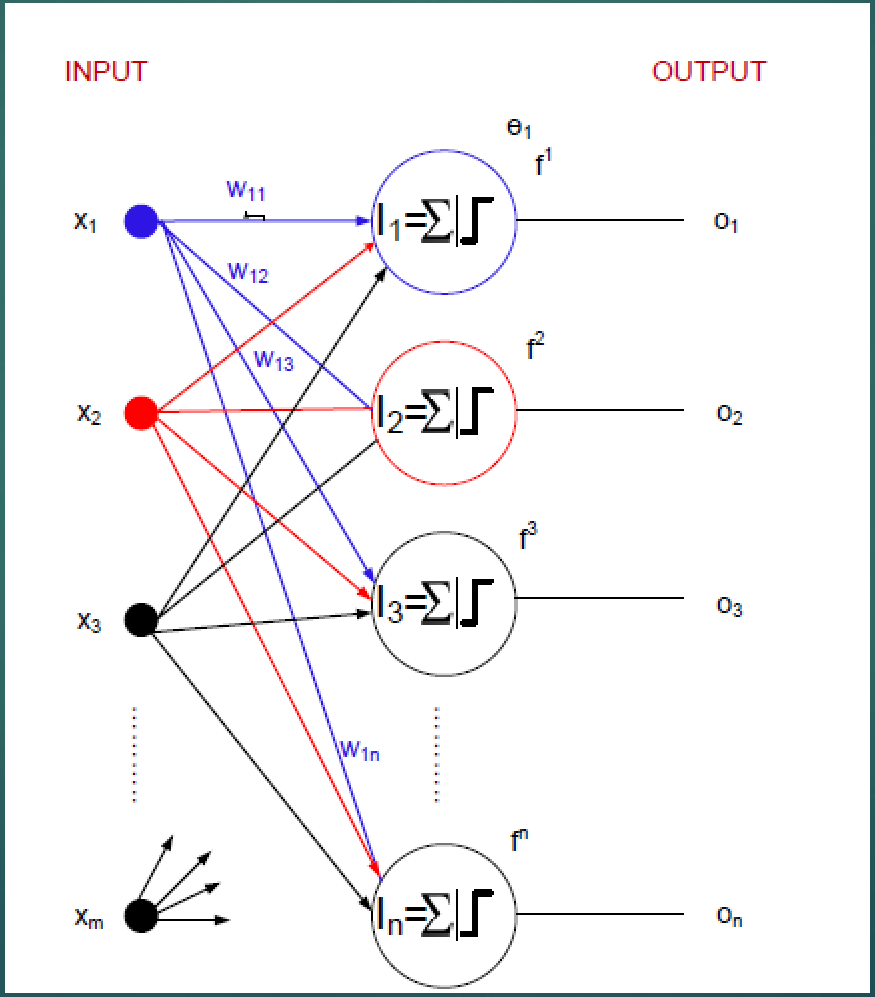
The AND Logic

A single neuron

Here, $y = \sum w_i x_i - \theta$ and $w_1 = 0.5, w_2 = 0.5$ and $\theta = 0.9$

Single layer feed forward neural network

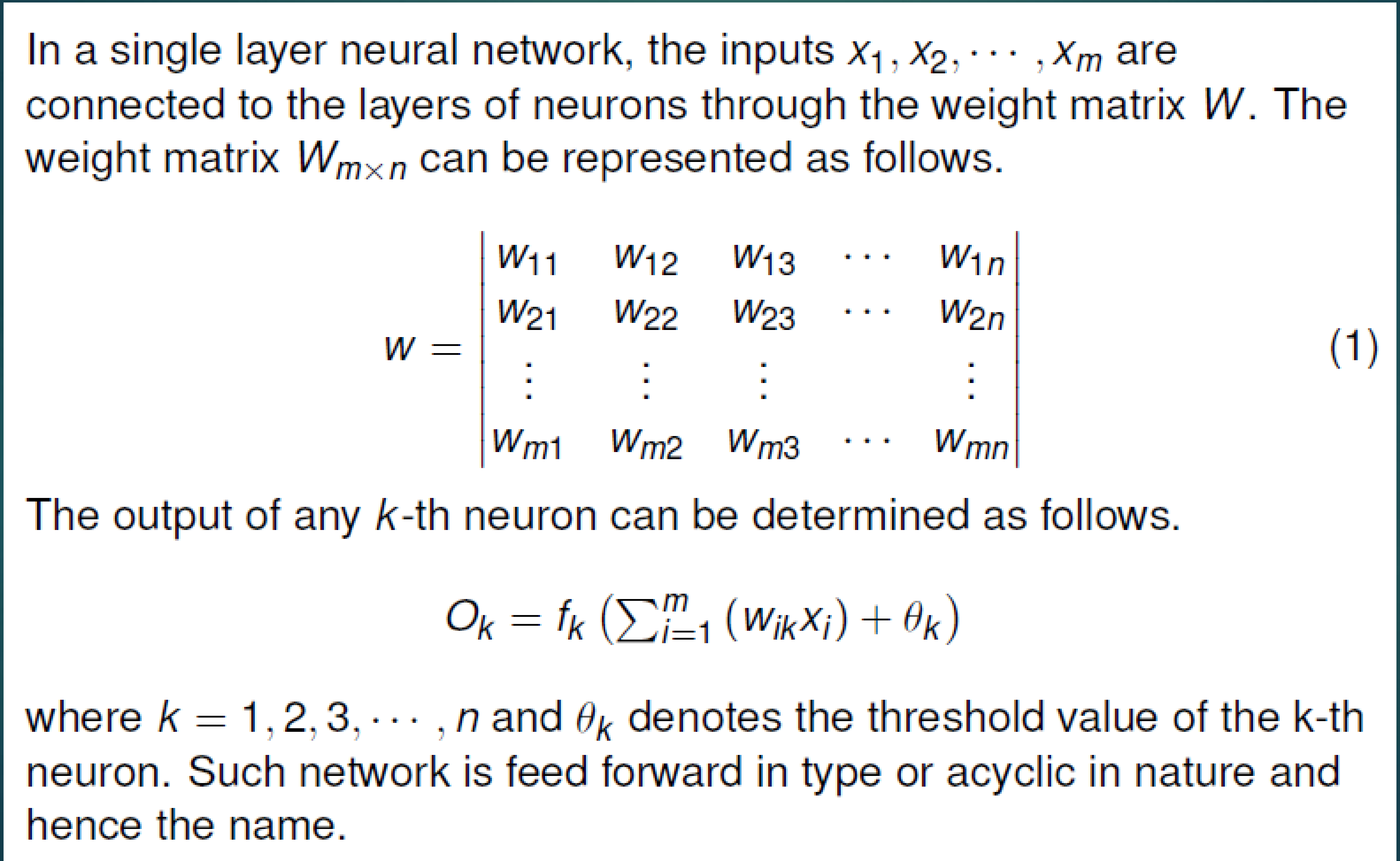
- The concept of the AND problem and its solution with a single neuron can be extended to multiple neurons.



Single layer feed forward neural network

- We see, a layer of n neurons constitutes a single layer feed forward neural network.
- This is so called because, it contains a single layer of artificial neurons.
- Note that the input layer and output layer, which receive input signals and transmit output signals are although called layers, they are actually boundary of the architecture and hence truly not layers.
- The only layer in the architecture is the synaptic links carrying the weights connect every input to the output neurons.

Modelling SLFFNN



In a single layer neural network, the inputs $x_1, x_2, \cdots, x_m$ are connected to the layers of neurons through the weight matrix W. The weight matrix $W_{m \times n}$ can be represented as follows.

$$w = \begin{vmatrix} w_{11} & w_{12} & w_{13} & \cdots & w_{1n} \\ w_{21} & w_{22} & w_{23} & \cdots & w_{2n} \\ \vdots & \vdots & \vdots & & \vdots \\ w_{m1} & w_{m2} & w_{m3} & \cdots & w_{mn} \end{vmatrix} \tag{1}$$

The output of any k-th neuron can be determined as follows.

$$O_k = f_k \left(\sum_{i=1}^{m} (w_{ik} x_i) + \theta_k \right)$$

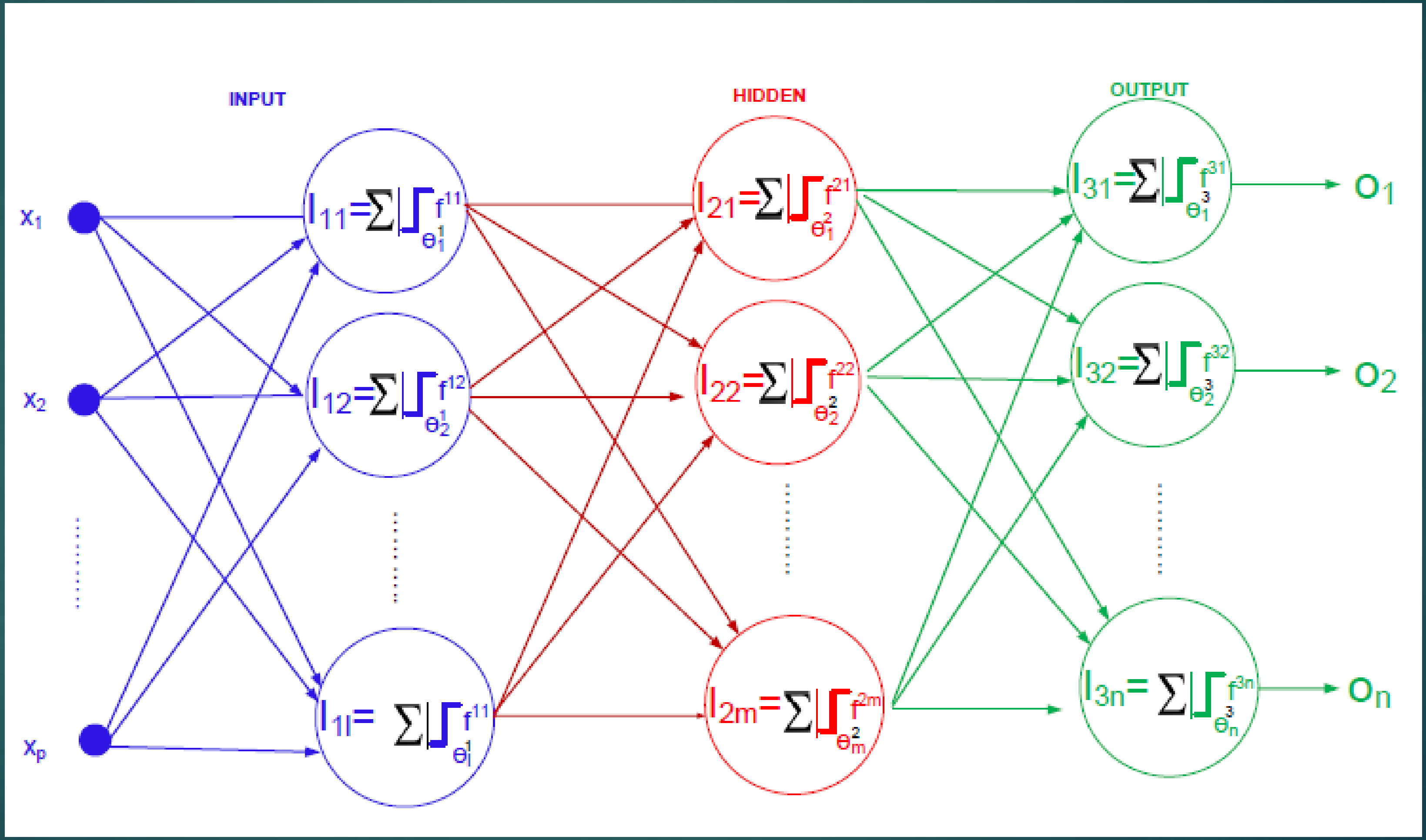
where $k = 1, 2, 3, \cdots, n$ and θ_k denotes the threshold value of the k-th neuron. Such network is feed forward in type or acyclic in nature and hence the name.

Multilayer feed forward neural networks

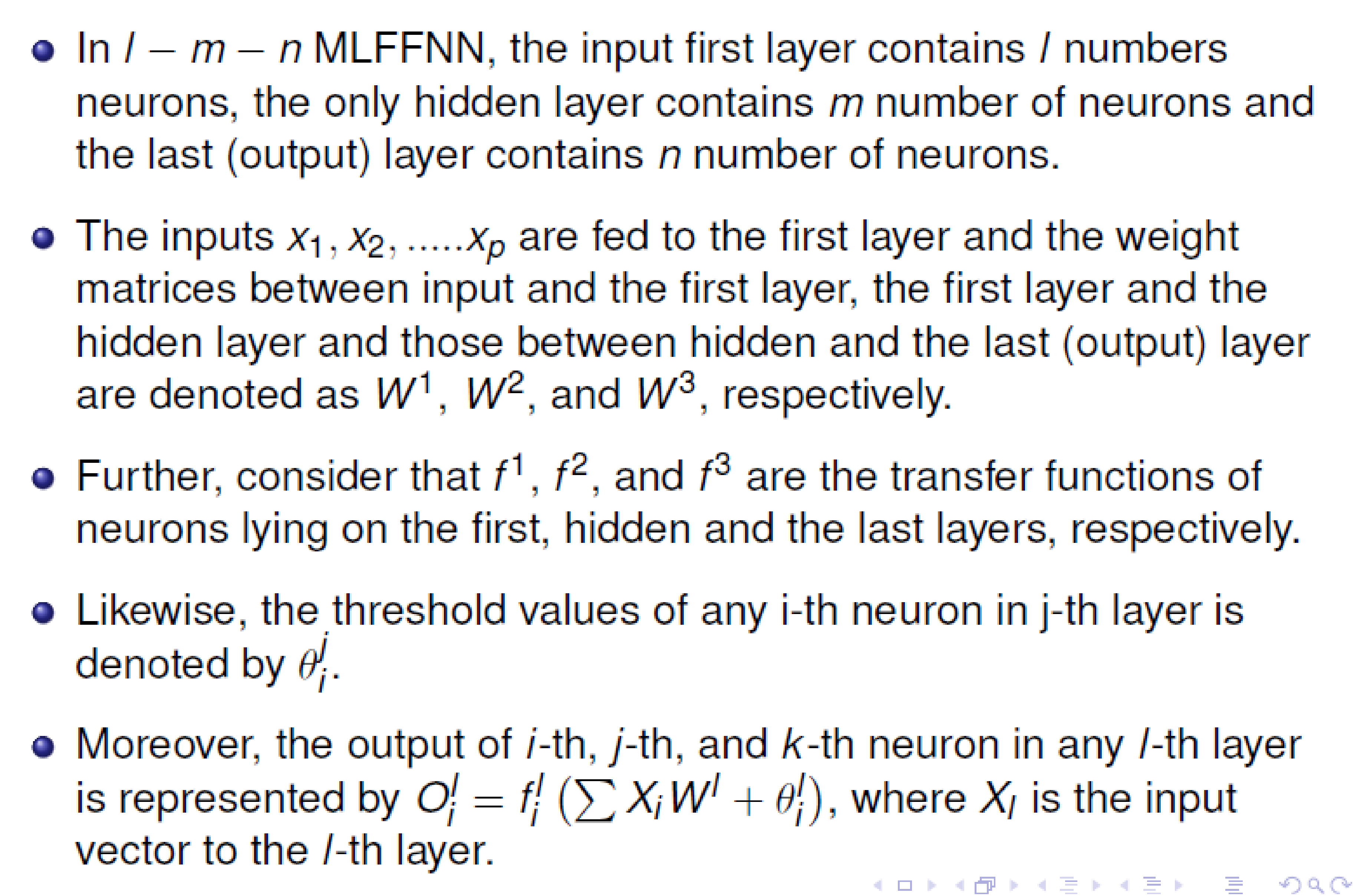
- This network, as its name indicates is made up of multiple layers.
- Thus architectures of this class besides processing an input and an output layer also have one or more intermediary layers called hidden layers.
- The hidden layer(s) aid in performing useful intermediary computation before directing the input to the output layer.
- A multilayer feed forward network with l input neurons (number of neuron at the first layer), m1;m2;…;mp number of neurons at i-th hidden layer (i = 1; 2;…; p) and n neurons at the last layer (it is the output neurons) is written as l –m1-m2-…-mp-n MLFFNN.

Multilayer feed forward neural networks

- Figure shows a schematic diagram of multilayer feed forward neural network with a configuration of l-m-n.


INPUT
HIDDEN
OUTPUT
x_1
x_2
x_p
$I_{11}=\Sigma$ f^{11} θ_1^1
$I_{12}=\Sigma$ f^{12} θ_2^1
$I_{1l}=\Sigma$ f^{11} θ_l^1
$I_{21}=\Sigma$ f^{21} θ_1^2
$I_{22}=\Sigma$ f^{22} θ_2^2
$I_{2m}=\Sigma$ f^{2m} θ_m^2
$I_{31}=\Sigma$ f^{31} θ_1^3
$I_{32}=\Sigma$ f^{32} θ_2^3
$I_{3n}=\Sigma$ f^{3n} θ_n^3
O_1
O_2
O_n

Multilayer feed forward neural networks

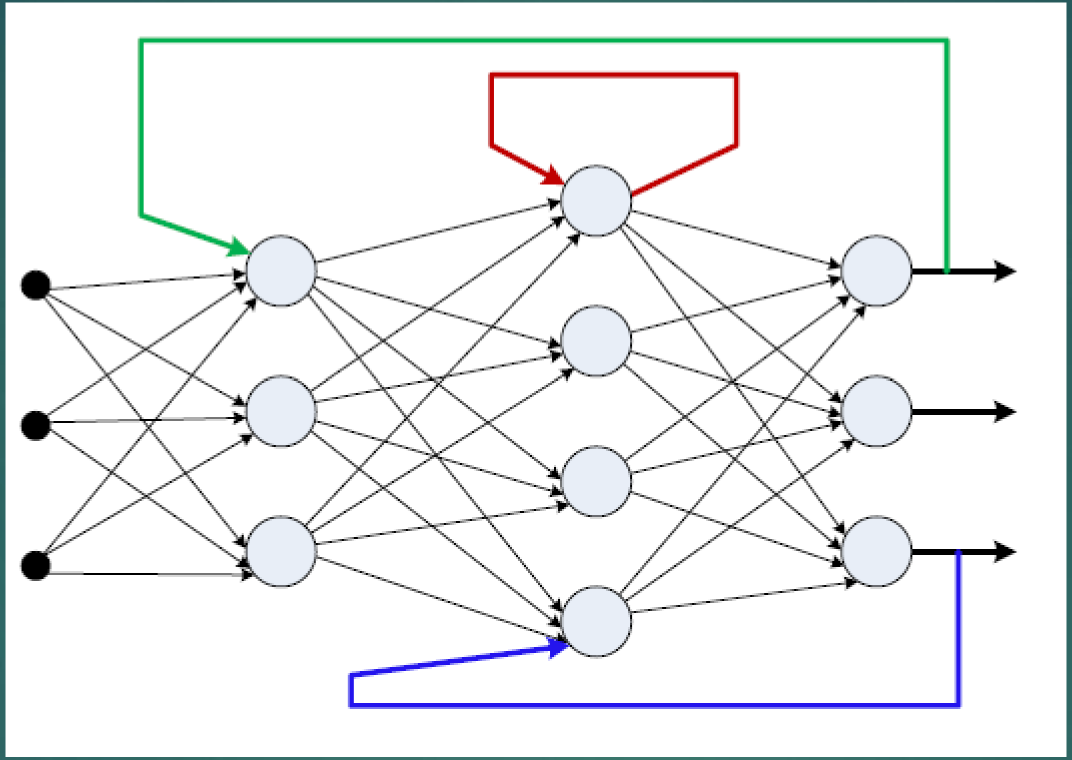


- In $l - m - n$ MLFFNN, the input first layer contains l numbers neurons, the only hidden layer contains m number of neurons and the last (output) layer contains n number of neurons.
- The inputs $x_1, x_2, ...x_p$ are fed to the first layer and the weight matrices between input and the first layer, the first layer and the hidden layer and those between hidden and the last (output) layer are denoted as W^1, W^2, and W^3, respectively.
- Further, consider that f^1, f^2, and f^3 are the transfer functions of neurons lying on the first, hidden and the last layers, respectively.
- Likewise, the threshold values of any i-th neuron in j-th layer is denoted by θ_i^j.
- Moreover, the output of i-th, j-th, and k-th neuron in any l-th layer is represented by $O_i^l = f_i^l\left(\sum X_i W^l + \theta_i^l\right)$, where X_l is the input vector to the l-th layer.

Recurrent neural network architecture

- The networks differ from feedback network architectures in the sense that there is at least one ”feedback loop”.
- Thus, in these networks, there could exist one layer with feedback connection.
- There could also be neurons with self-feedback links, that is, the output of a neuron is fed back into itself as input.

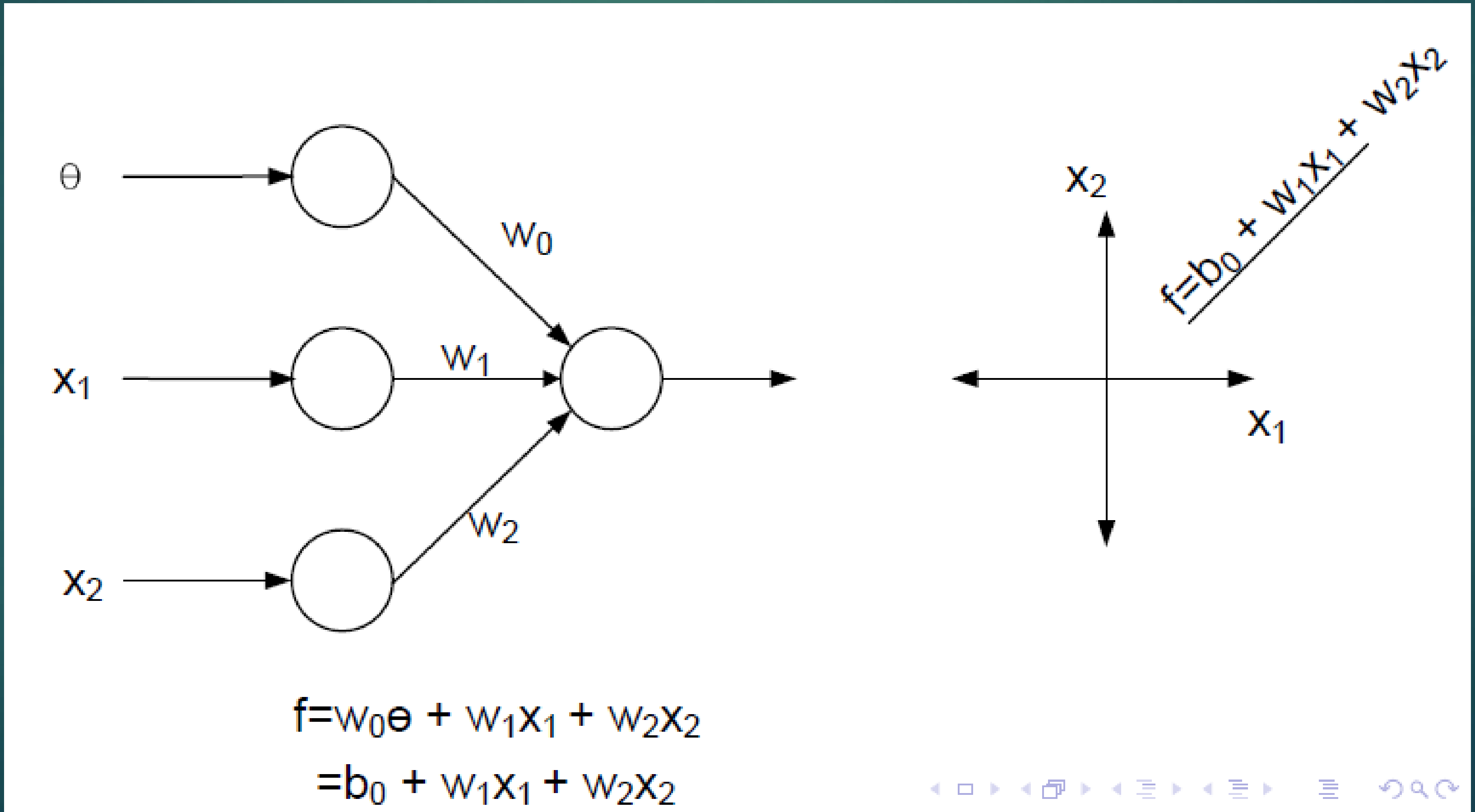
Recurrent neural network architecture



Depending on different type of feedback loops, several recurrent neural networks are known such as Hopfield network, Boltzmann machine network etc.

Why are different type of neural network architecture?

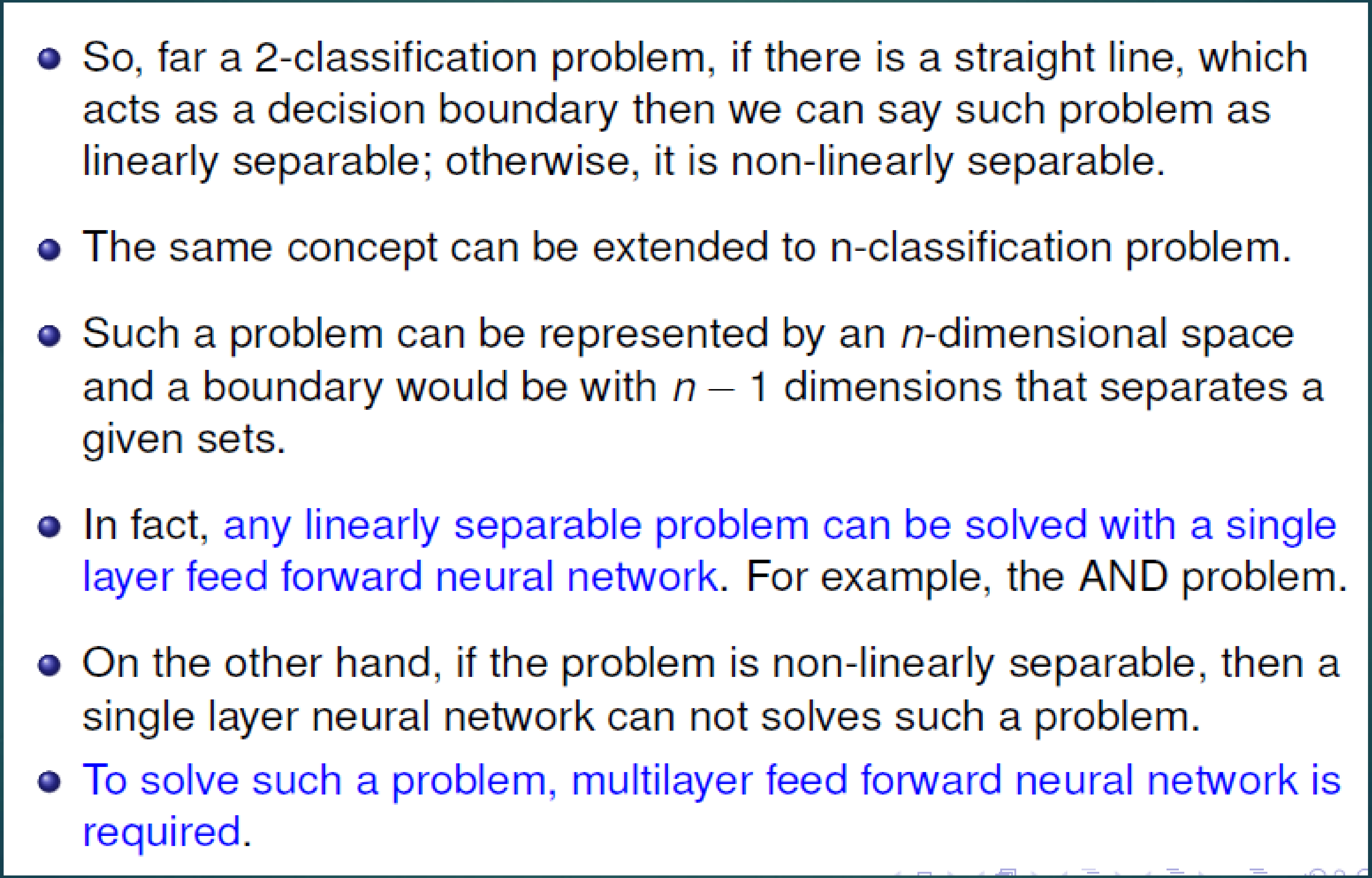
To give the answer to this question, let us first consider the case of a single neural network with two inputs as shown below.


θ
x_1
x_2
w_0
w_1
w_2
$f = w_0 \theta + w_1 x_1 + w_2 x_2$
$= b_0 + w_1 x_1 + w_2 x_2$
x_2
x_1
$f = b_0 + w_1 x_1 + w_2 x_2$

Our observations

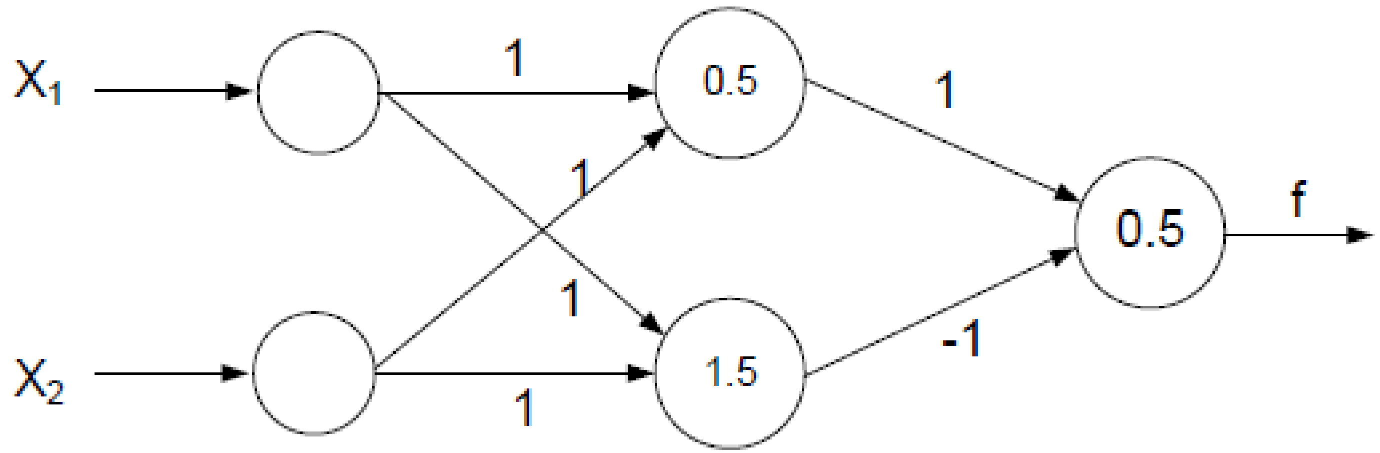
- From the example discussed, we understand that a straight line is possible in AND-problem to separate two tasks namely the output as 0 or 1 for any input.
- However, in case of XOR problem, such a line is not possible.
- Note: horizontal or a vertical line in case of XOR problem is not admissible because in that case it completely ignores one input.

Example



- So, far a 2-classification problem, if there is a straight line, which acts as a decision boundary then we can say such problem as linearly separable; otherwise, it is non-linearly separable.
- The same concept can be extended to n-classification problem.
- Such a problem can be represented by an *n*-dimensional space and a boundary would be with $n - 1$ dimensions that separates a given sets.
- In fact, any linearly separable problem can be solved with a single layer feed forward neural network. For example, the AND problem.
- On the other hand, if the problem is non-linearly separable, then a single layer neural network can not solves such a problem.
- To solve such a problem, multilayer feed forward neural network is required.

Example: Solving XOR problem

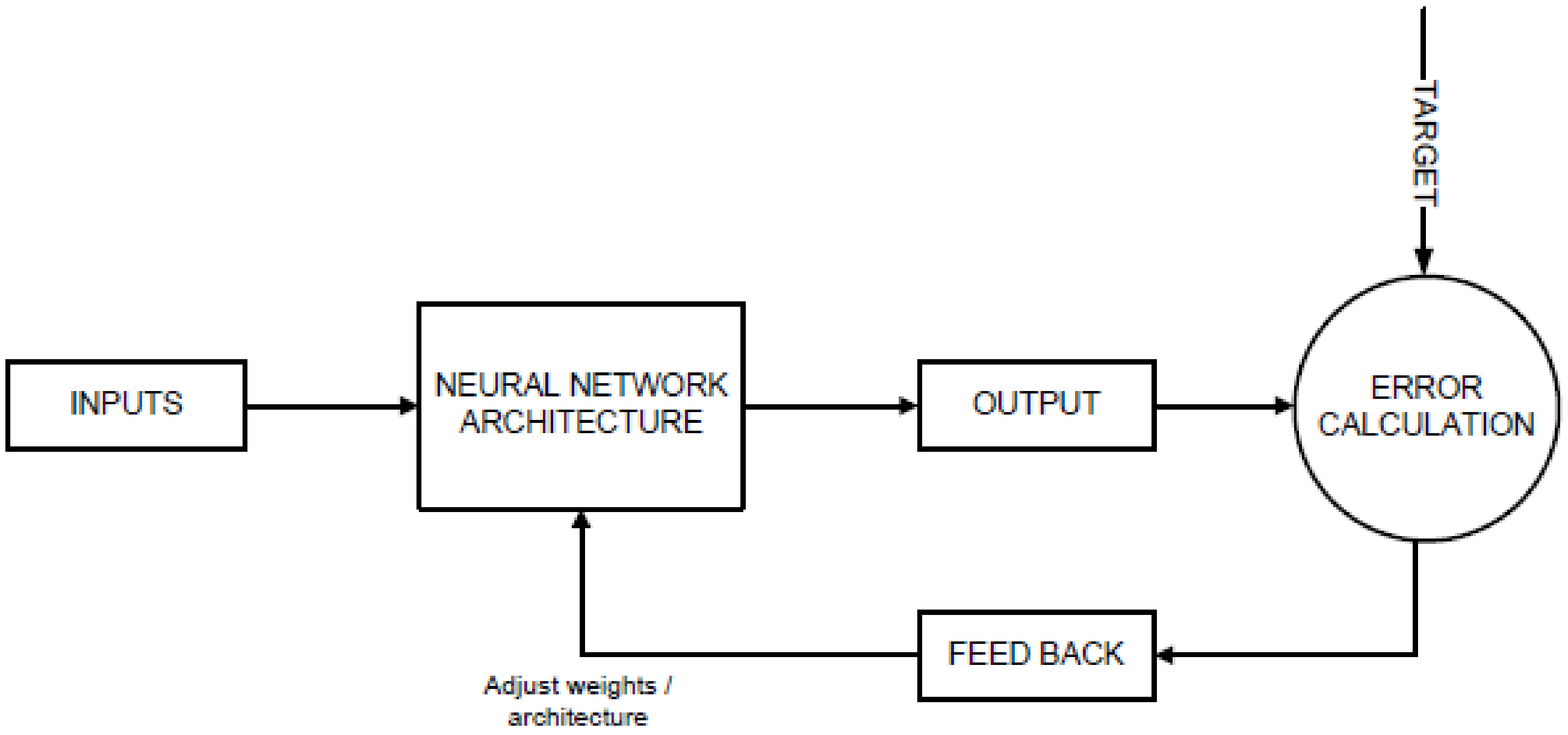


Neural network for XOR-problem

Dynamic neural network

- In some cases, the output needs to be compared with its target values to determine an error, if any.
- Based on this category of applications, a neural network can be static neural network or dynamic neural network.
- In a static neural network, error in prediction is neither calculated nor feedback for updating the neural network.
- On the other hand, in a dynamic neural network, the error is determined and then feed back to the network to modify its weights (or architecture or both).

Dynamic neural network



Framework of dynamic neural network

Dynamic neural network

From the above discussions, we conclude that

- For linearly separable problems, we solve using single layer feed forward neural network.
- For non-linearly separable problem, we solve using multilayer feed forward neural networks.
- For problems, with error calculation, we solve using recurrent neural networks as well as dynamic neural networks.

Learning of neural networks: Topics

- Concept of learning
- Learning in
    - Single layer feed forward neural network
    - multilayer feed forward neural network
    - recurrent neural network
- Types of learning in neural networks

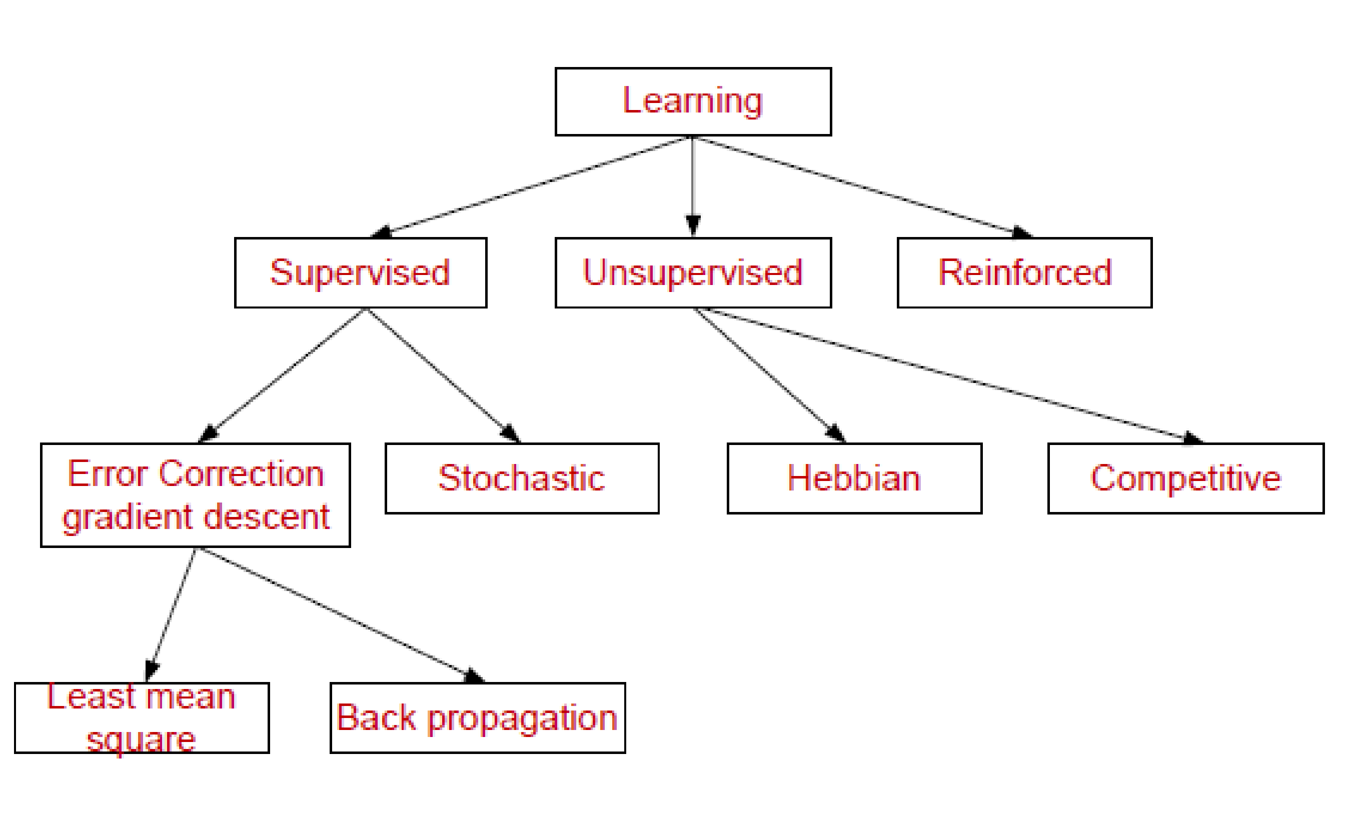
Concept of learning

Concept of learning

- The learning is an important feature of human computational ability.
- Learning may be viewed as the change in behaviour acquired due to practice or experience, and it lasts for relatively long time.
- As it occurs, the effective coupling between the neuron is modified.
- In case of artificial neural networks, it is a process of modifying neural network by updating its weights, biases and other parameters, if any.
- During the learning, the parameters of the networks are optimized and as a result process of curve fitting.
- It is then said that the network has passed through a learning phase.

Types of learning

- There are several learning techniques.
- A taxonomy of well known learning techniques are shown in the following.



Different learning techniques: Supervised learning

- **Supervised learning**

  In this learning, every input pattern that is used to train the network is associated with an output pattern.
- This is called ”training set of data”. Thus, in this form of learning, the input-output relationship of the training scenarios are available.
- Here, the output of a network is compared with the corresponding target value and the error is determined.
- It is then feed back to the network for updating the same. This results in an improvement.
- This type of training is called learning with the help of teacher.

Different learning techniques: Unsupervised learning

- **Unsupervised learning**

  If the target output is not available, then the error in prediction can not be determined and in such a situation, the system learns of its own by discovering and adapting to structural features in the input patterns.

- This type of training is called learning without a teacher.

Different learning techniques: Reinforcement learning

- **Reinforced learning**

  In this techniques, although a teacher is available, it does not tell the expected answer, but only tells if the computed output is correct or incorrect. A reward is given for a correct answer computed and a penalty for a wrong answer. This information helps the network in its learning process.

- **Note :** Supervised and unsupervised learnings are the most popular forms of learning. Unsupervised learning is very common in biological systems.
- It is also important for artificial neural networks : training data are not always available for the intended application of the neural network.

Different learning techniques : Gradient descent learning

- **Gradient Descent learning :**

  This learning technique is based on the minimization of error E defined in terms of weights and the activation function of the network.

- Also, it is required that the activation function employed by the network is differentiable, as the weight update is dependent on the gradient of the error E.

- Thus, if ΔWij denoted the weight update of the link connecting the i-th and j-th neuron of the two neighbouring layers then

$$\Delta W_{ij} = \eta \frac{\partial E}{\partial W_{ij}}$$

- where *n* is the **learning rate parameter** and $\frac{\partial E}{\partial W_{ij}}$ is the **error gradient** with reference to the weight Wij.

- The least mean square and back propagation are two variations of this learning technique.

Different learning techniques : Stochastic learning

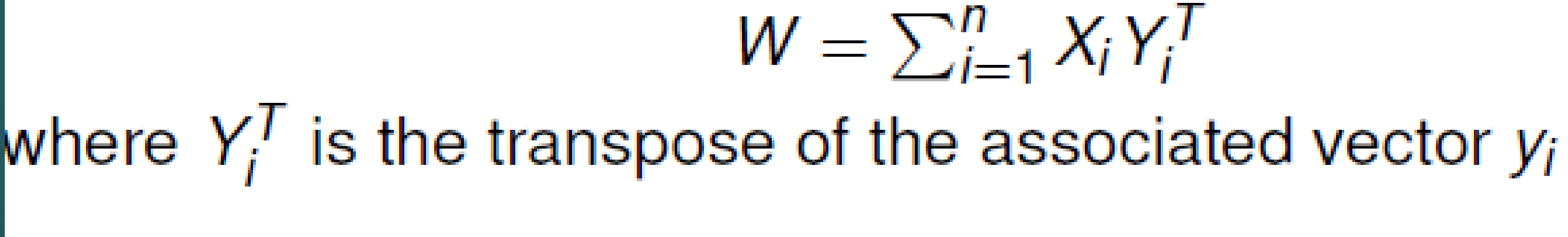
- **Stochastic learning**

  In this method, weights are adjusted in a probabilistic fashion. Simulated annealing is an example of such learning (proposed by Boltzmann and Cauch)

Different learning techniques: Hebbian learning

Hebbian learning

- This learning is based on correlative weight adjustment. This is, in fact, the learning technique inspired by biology.
- Here, the input-output pattern pairs $(x_i, y_i)$are associated with the weight matrix W. W is also known as the correlation matrix.
- This matrix is computed as follows.



$$W = \sum_{i=1}^{n} X_i Y_i^T$$

where Y_i^T is the transpose of the associated vector y_i

Different learning techniques : Competitive learning

- **Competitive learning**

In this learning method, those neurons which responds strongly to input stimuli have their weights updated.

- When an input pattern is presented, all neurons in the layer complete and the winning neuron undergoes weight adjustment.
- This is why it is called a Winner-takes-all strategy.