# Introduction to Artificial Intelligence



## Chapter 4: Learning (2) Artificial Neural Network - A Brief Overview



Nguyễn Hải Minh, Ph.D
nhminh@fit.hcmus.edu.vn

# Outlines

- ❑Biological Inspiration.
- ❑Artificial Neural Networks.
- ❑ANN Architectures.
- ❑Learning Processes.
- ❑ANN Capabilities & Limitations

# Biological Inspiration

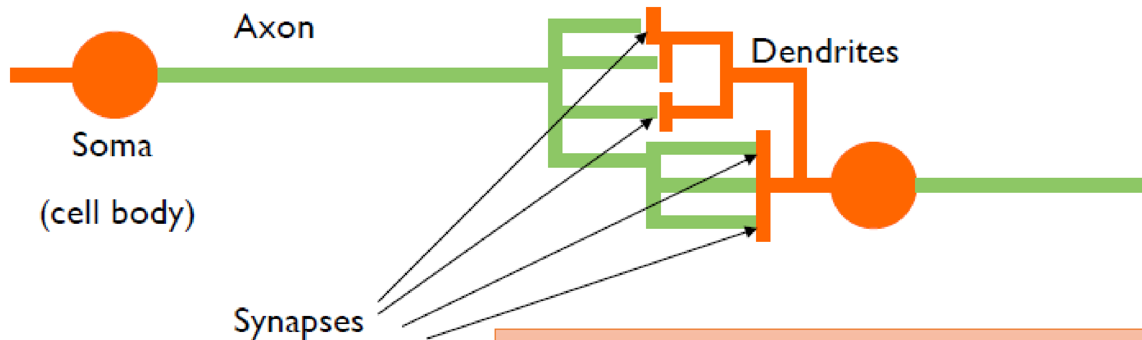
## Some numbers…

- The human brain contains about 10 billion nerve cells (neurons).
- Each neuron is connected to the others through 10000 synapses.

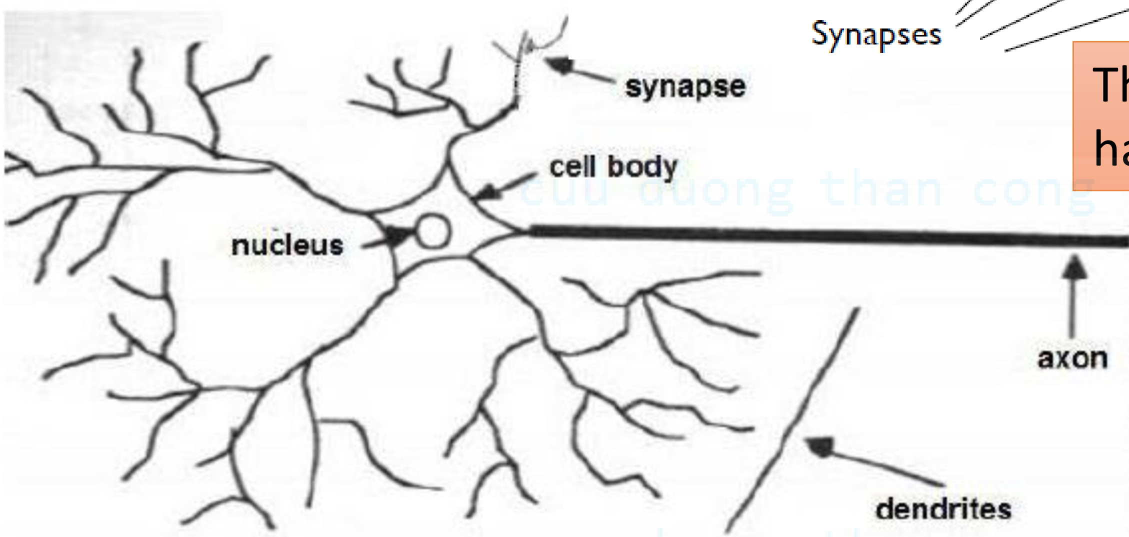
## Properties of the brain:

- It can learn, reorganize itself from experience.
- It adapts to the environment.

# The Neuron in Real Life




The information transmission happens at the synapses.

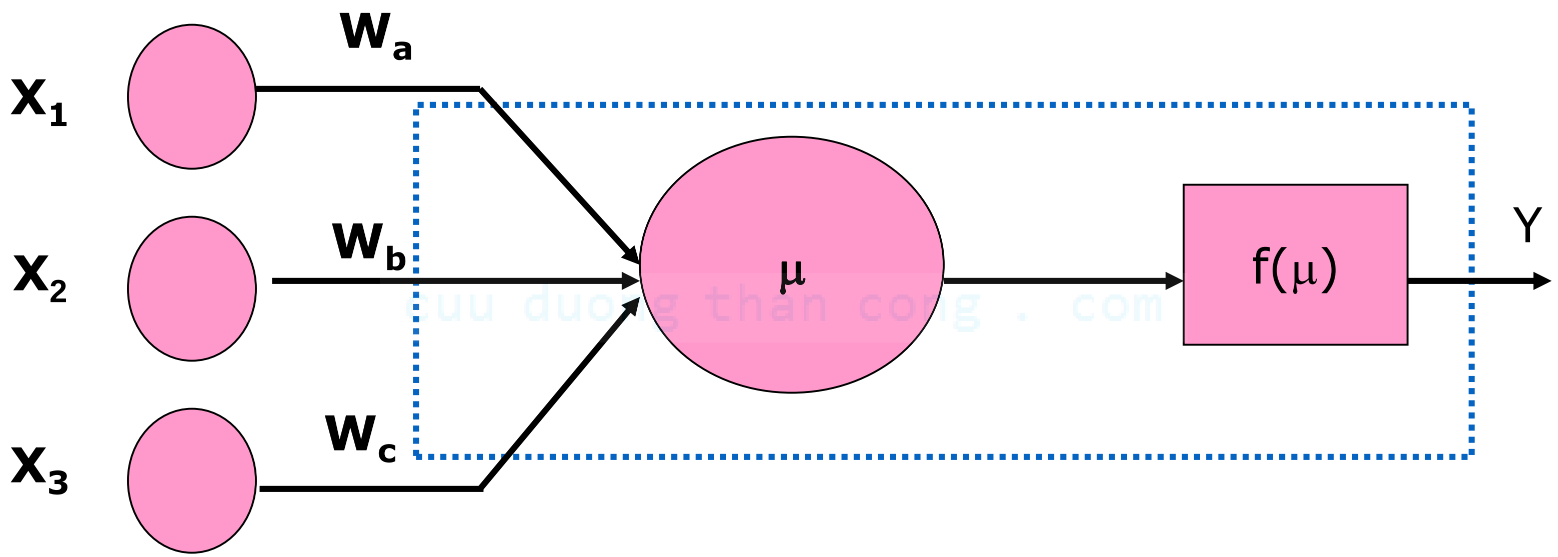





cuu duong than cong . com

- The neuron receives nerve impulses through its **dendrites**.
- It then sends the nerve impulses through its **axon** to the terminals where neurotransmitters are released to stimulate other neurons.

# Model Of A Neuron



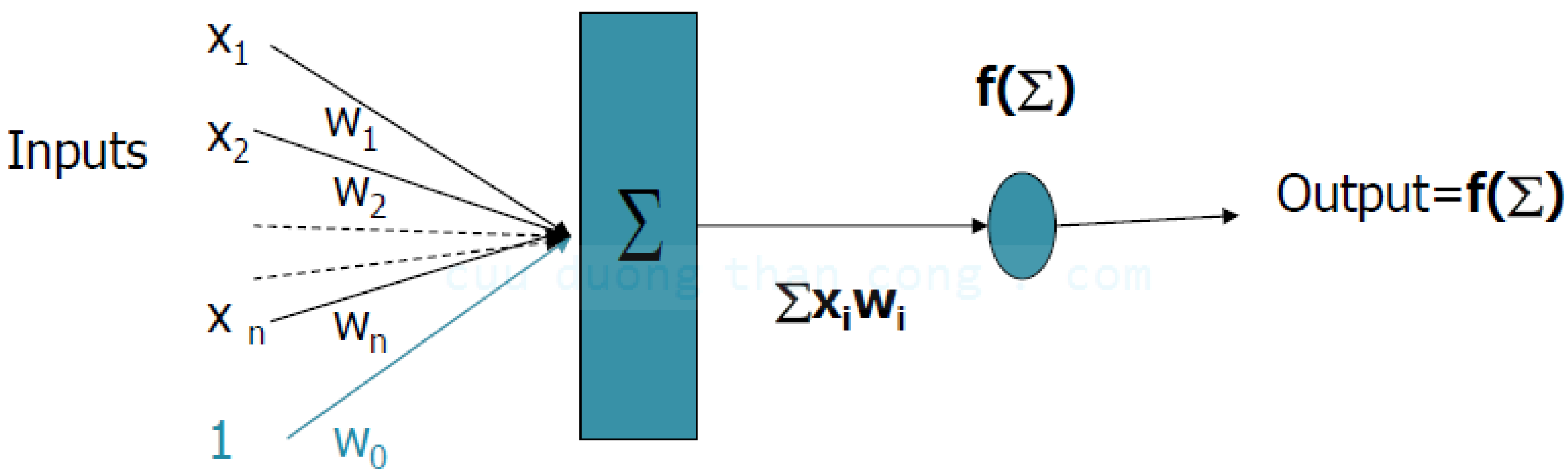

Input units | Connection weights | Summing function | computation

(dendrite) | (synapse) | (soma) | (axon)

# Artificial Neuron

- **Definition**: Neuron is the basic information processing unit of the Neural Networks (NN). It is a non linear, parameterized function with restricted output range.





# Artificial Neural Networks

- **Artificial Neural Network (ANN):** is a machine learning approach that models human brain and consists of a number of artificial neurons that are linked together according to a specific network architecture.
- **Neuron** in ANNs tend to have fewer connections than biological neurons. each neuron in ANN receives a number of inputs.
- **An activation function** is applied to these inputs which results in activation level of neuron (output value of the neuron).
- Knowledge about the learning task is given in the form of examples called training examples.

# Applications of ANN

Some tasks to be solved by Artificial Neural Networks:

- ❖ **Classification**: Linear, non-linear.
- ❖ **Recognition**: Spoken words, Handwriting. Also recognizing a visual object: Face recognition.
- ❖ **Controlling**: Movements of a robot based on self perception and other information.
- ❖ **Predicting**: Where a moving object goes, when a robot wants to catch it.
- ❖ **Optimization**: Find the shortest path for the TSP.

# Artificial Neural Networks

- Before using ANN, we have to define:

1. Artificial Neuron Model.
2. ANN Architecture.
3. Learning Mode.

# Computing with Neural Units

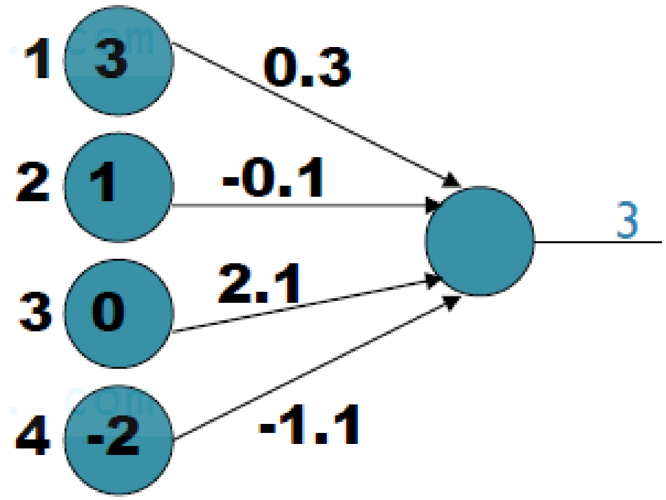
- ❑Incoming signals to a unit are presented as inputs.
- ❑How do we generate outputs?

- One idea: *Summed Weighted Inputs.*
- **Input**: (3, 1, 0, -2)
- **Processing**

3(0.3) + 1(-0.1) + 0(2.1) + -2(-1.1)

= 0.9 + (-0.1) + 0 + 2.2

- **Output**: 3



# Activation Functions

- Usually, do not just use weighted sum directly.
- Apply some function to the weighted sum before it is used (e.g., as output).

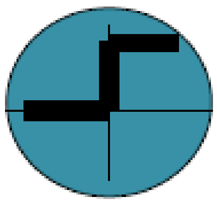
→***Activation function***.

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

$\theta$ Is called the threshold

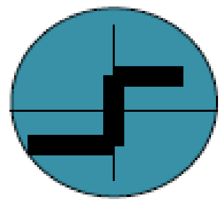
**Step function**

# Activation Functions



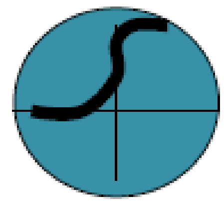
Step Function

**step(x) =**
**1, if x >= threshold**
**0, if x < threshold**
**(in picture above, threshold = 0)**



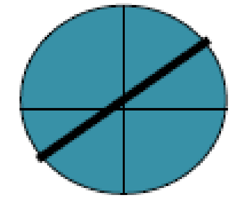
Sign Function

**sign(x) =**
**+1, if x >= 0**
**-1, if x < 0**



Sigmoid (Logistic) Function

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



Linear Function

**pl(x) = x**



- The choice of activation function determines the ***Neuron Model.***

# Bias of a Neuron

- Bias is like another weight. It's included by adding a component $x_0=1$ to the input vector X.
- $X=(1,X_1,X_2...X_i,...X_n)$
- Bias is of two types
  - Positive bias: increase the net input
  - Negative bias: decrease the net input

# Bias of a Neuron

- The bias ***b*** has the effect of applying a transformation to the weighted sum ***u***

$$v = u + b$$

- The bias is an external parameter of the neuron. It can be modeled by adding an extra input.
- ***v*** is called **induced field** of the neuron:

$$v = \sum_{j=0}^{m} w_j x_j$$

$$w_0 = b$$

# Learning rate

- Denoted by α.
- Used to control the amount of weight adjustment at each step of training
- E.g. Perceptron rule:

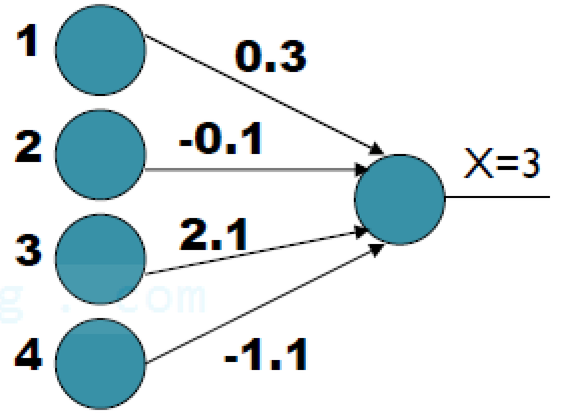
$$w_i \leftarrow w_i + \alpha(y - h_w(X))*x_\mathrm{i}$$

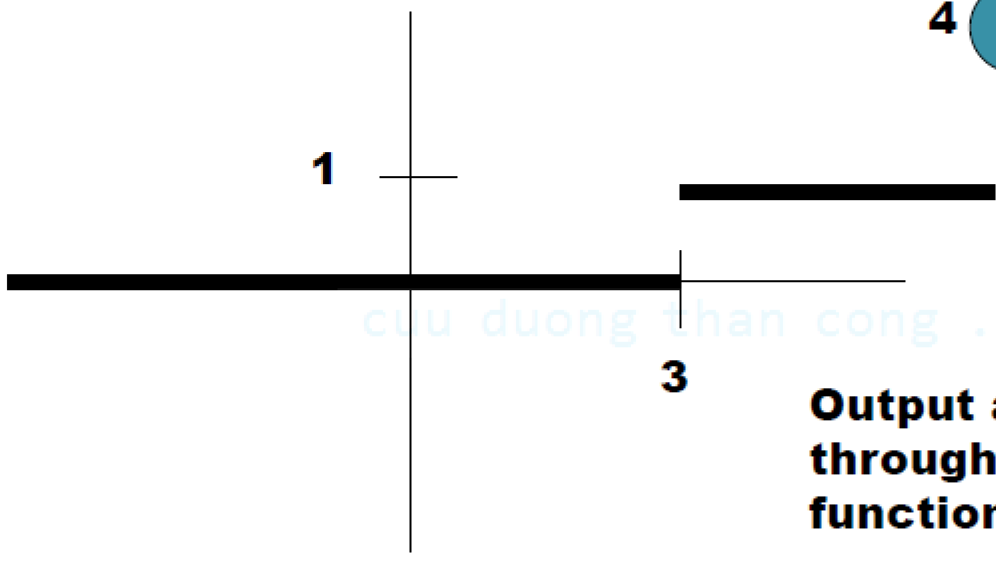
- Learning rate ranging from 0 to 1 determines the rate of learning in each time step

# Example (1): Step Function

- Let $\Theta = 3$

**Input: (3, 1, 0, -2)**



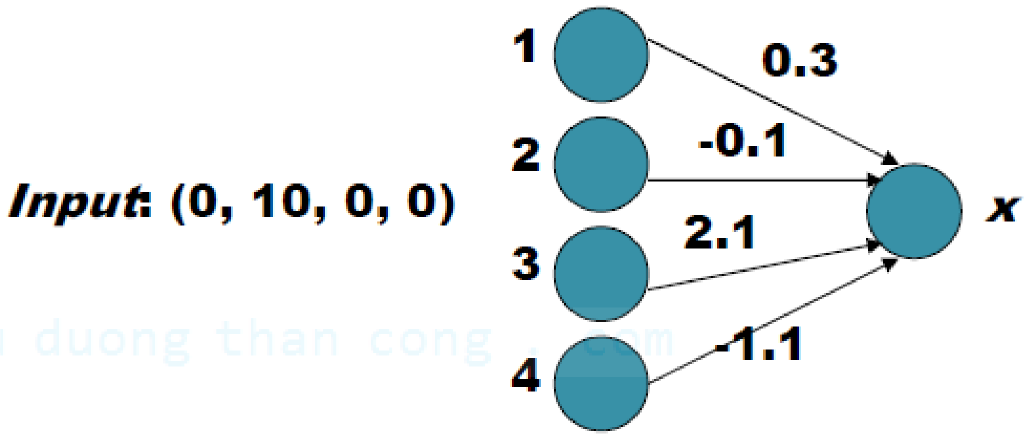



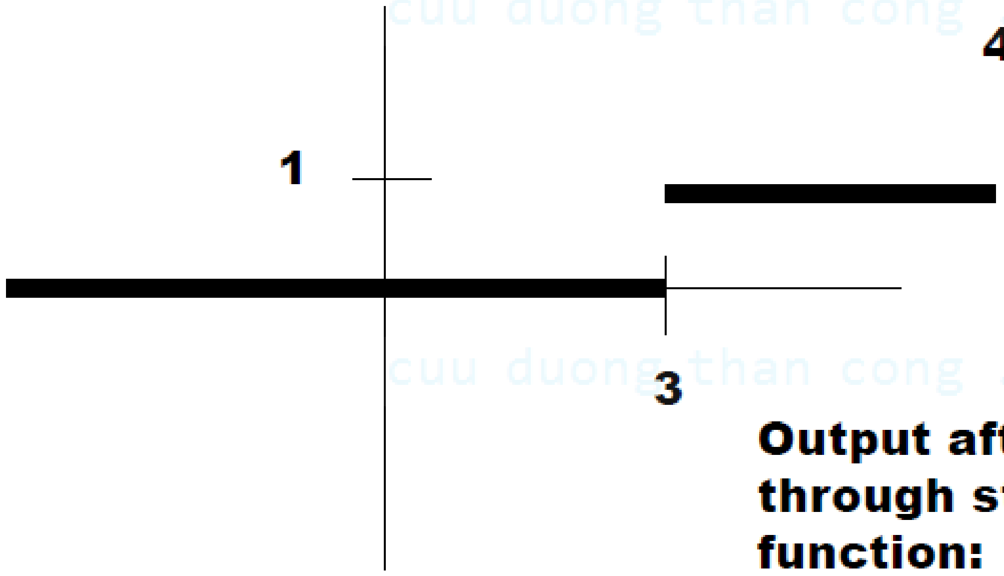

**Output after passing through step activation function:**

$$f(3) = 1$$

# Example (2): Another Step Function

- Let $\Theta = 3$






**Output after passing through step activation function:**

$$f(x) = ?$$

# Example (3): Sigmoid Function

- The math of some neural nets requires that the activation function be continuously differentiable.

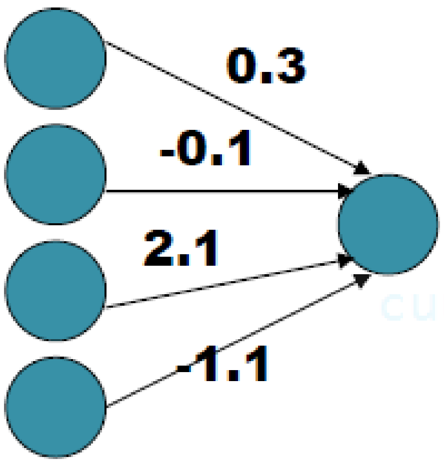


→ A sigmoidal function often used to approximate the step function.

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

$\sigma$ **Is the steepness parameter**

# Example (3): Sigmoid Function




$$\sigma = 2$$

***Input*: (3, 1, 0, -2)**

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



$$f(3) = \frac{1}{1+e^{-2x}} = .998$$

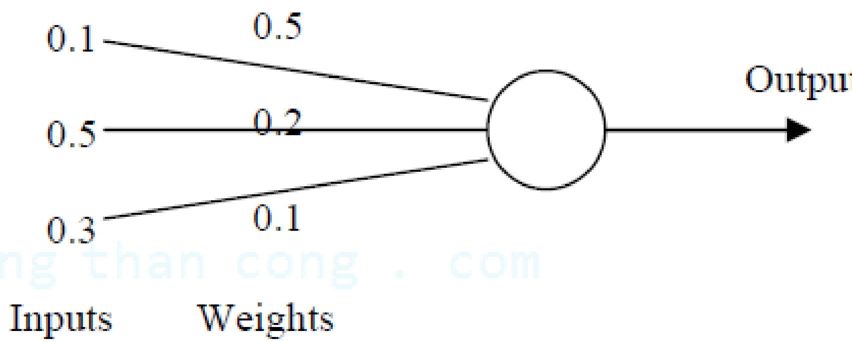


***Input*: (0, 10, 0, 0)**

**network output?**

# Example

- Calculate the output from the neuron below assuming a threshold of 0.5:




  - Sum = (0.1 x 0.5) + (0.5 x 0.2) + (0.3 x 0.1) = 0.05 + 0.1 + 0.03 = 0.18
  - Since 0.18 is less than the threshold, the Output = 0
  - Repeat the above calculation assuming that the neuron has a sigmoid output function:

$$\text{Sum is still 0.18, but now Output} = \frac{1}{1+e^{-0.18}} = 0.545$$

# Network Architecture

- The Architecture of a neural network is linked with the learning algorithm used to train.

- **There are different classes of network architecture**:
  - Single-Layer Neural Networks.
  - Multi-Layer Neural Networks.





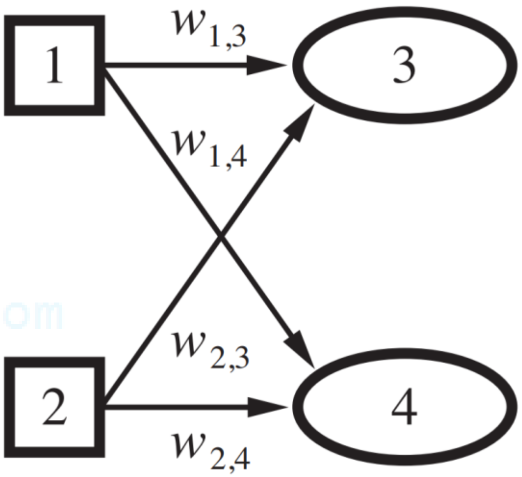
- → *The number of layers and neurons depend on the specific task.*

# Single Layer Neural Network

- Another name: Perceptron
  - A network with all inputs connected directly to the output.
  - *m* outputs = *m* separate training processes
  - Learning rule: **Perceptron learning rule** or **gradient descent rule**

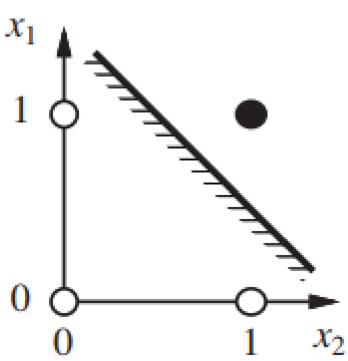
| $x_1$ | $x_2$ | $y_3$ (carry) | $y_4$ (sum) |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 0 |

✓ Unit 3: the carry function
✓ Unit 4: the sum function

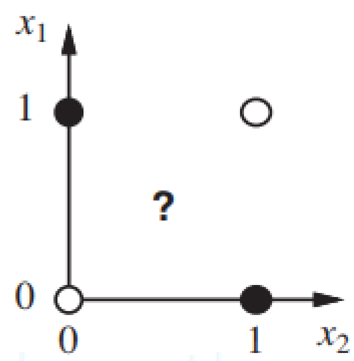


A perceptron network with 2 inputs and 2 outputs

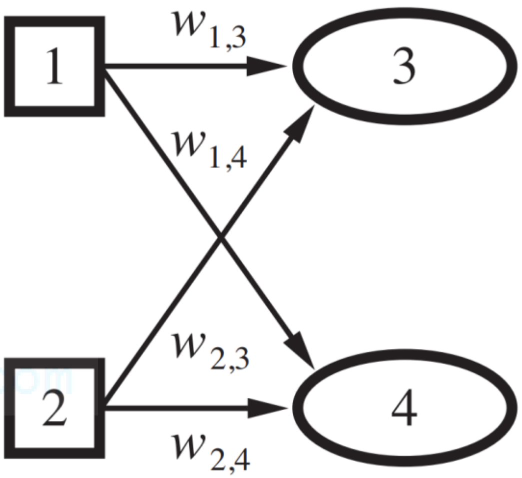
# Single Layer Neural Network




$x_1$ **AND** $x_2$




$x_1$ **XOR** $x_2$




Unit 3 learns the carry function easily

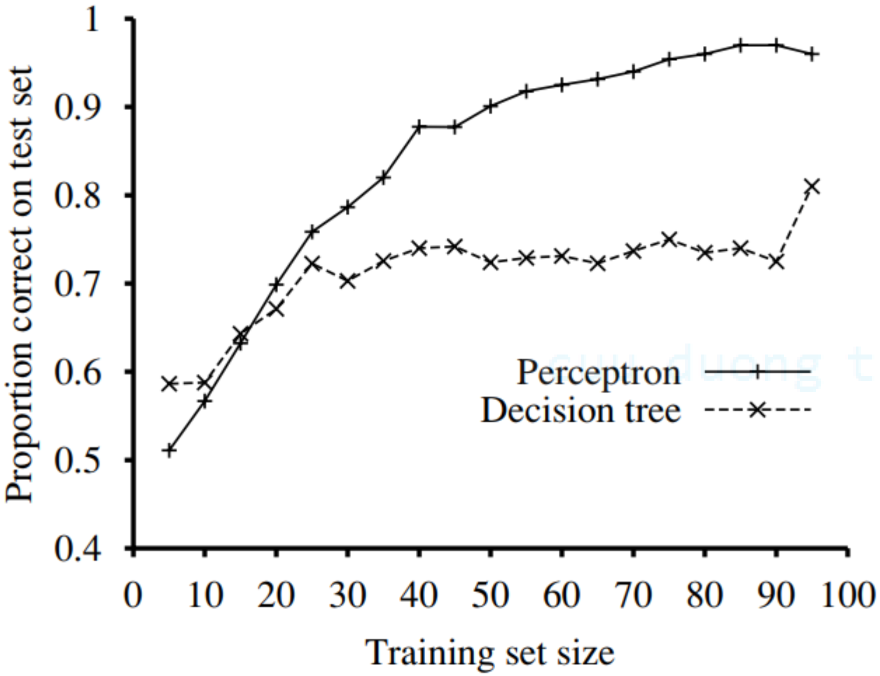
Unit 4 fails to learn the sum function





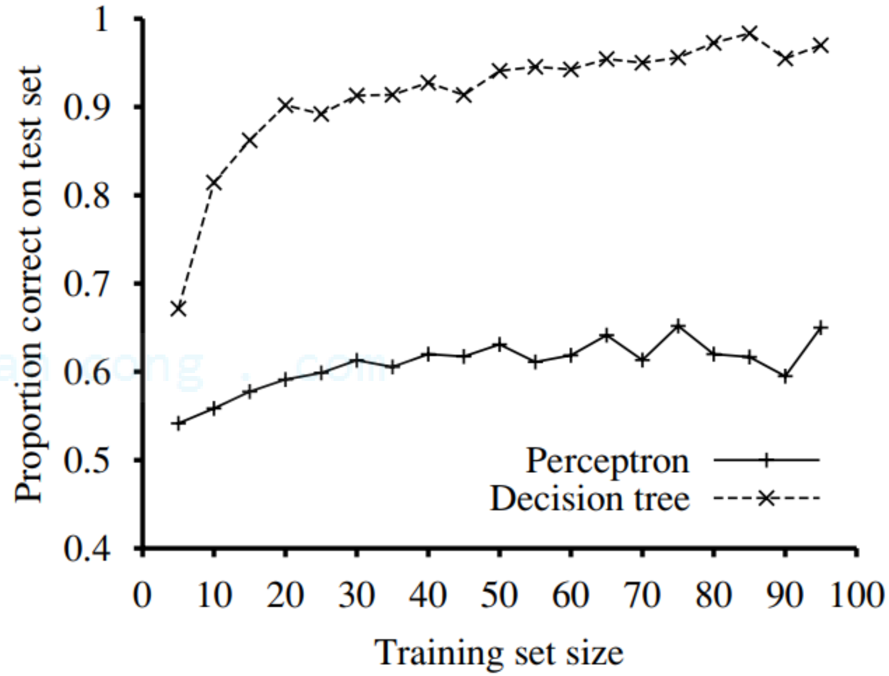
→ *Perceptron cannot learn a non-linearly separable function*

# Perceptron vs Decision Trees




(a)

Majority function

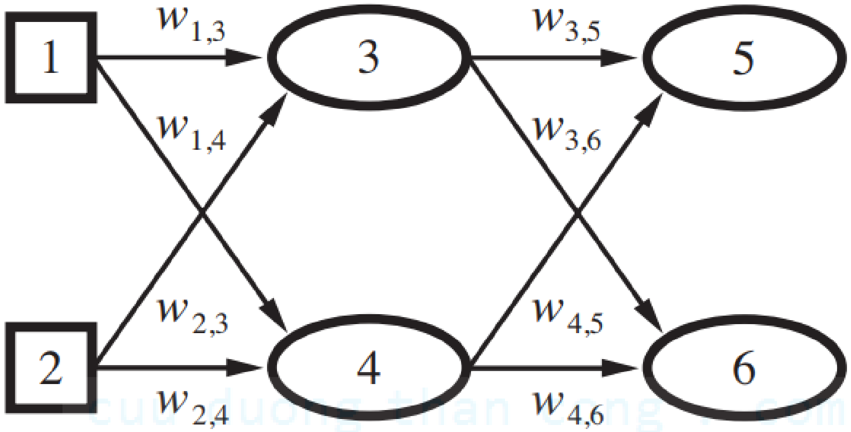



(b)

*WillWait* function

→ *Perceptron can represent some quite "complex" Boolean functions very compactly.*

# Multi Layer Neural Network

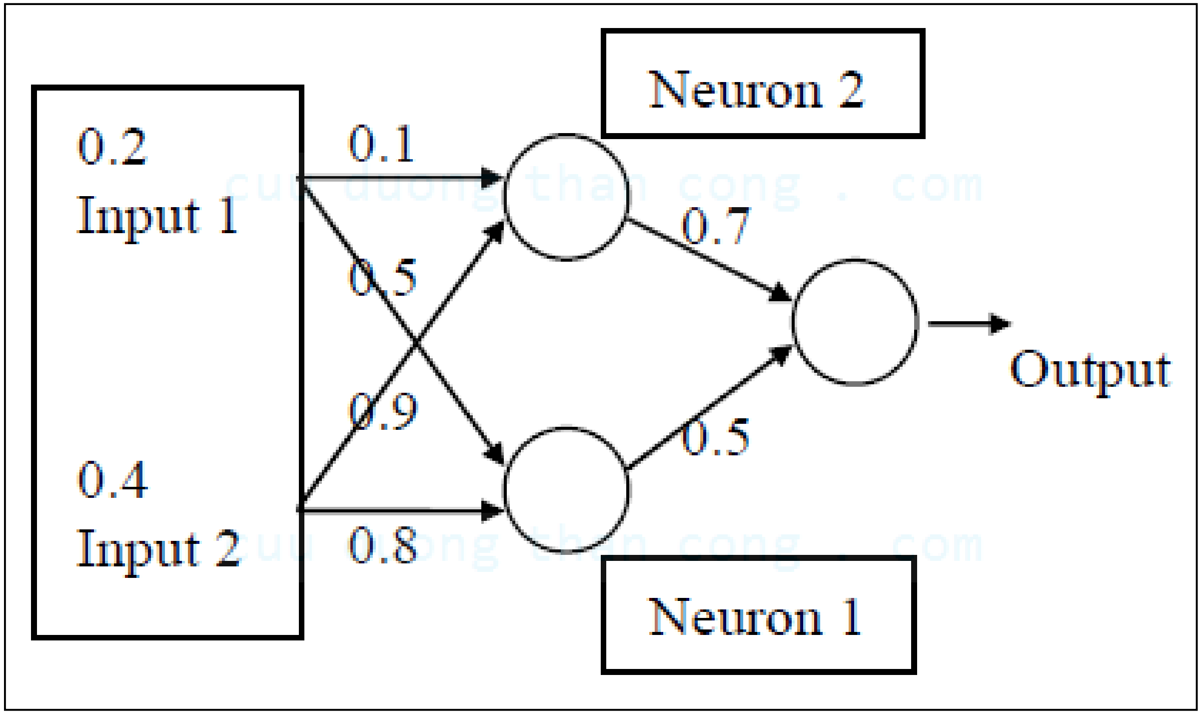



A multi layer network with 2 inputs, 2 hidden units, and 2 outputs

- More general network architecture, where there are hidden layers between input and output layers.
- Hidden nodes do not directly receive inputs nor send outputs to the external environment.
- Multi Layer NN overcome the limitation of Single-Layer NN, they can handle non-linearly separable learning tasks.
- Learning algorithm: **Back-Propagation**

# Example of multilayer ANN

- ❑Calculate the output from this network assuming a Sigmoid Squashing Function.



# Example of multilayer ANN

- Calculate the output from this network assuming a Sigmoid Squashing Function.

Input to neuron 1 = (0.2 x 0.5) + (0.4 x 0.8) = 0.42. Output = $\frac{1}{1+e^{-0.42}} = 0.603$



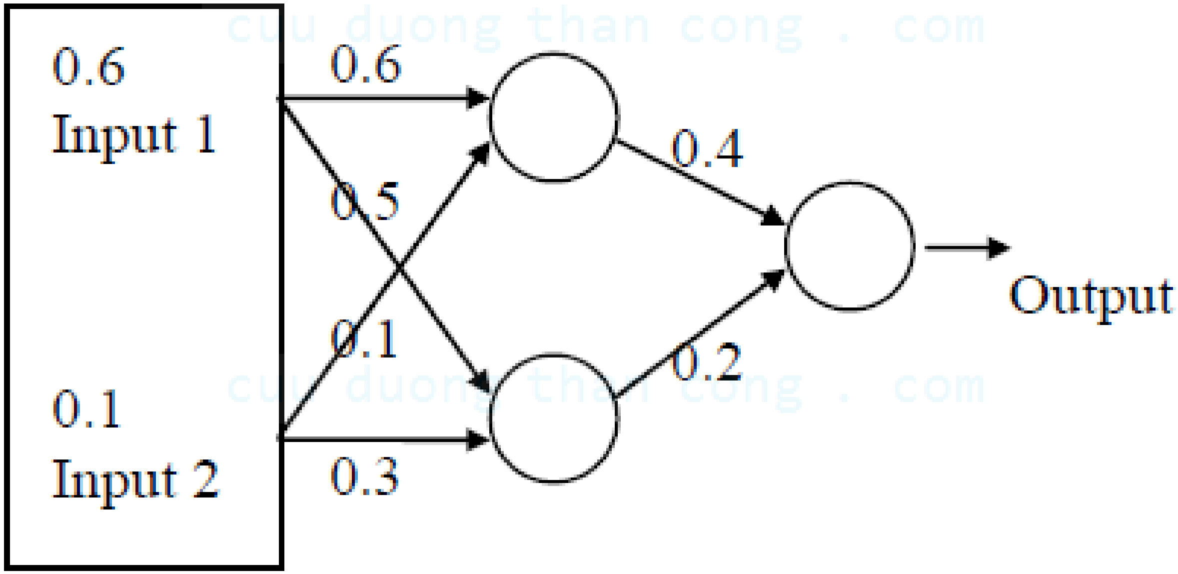
Input to neuron 2 = (0.2 x 0.1) + (0.4 x 0.9) = 0.38. Output = $\frac{1}{1+e^{-0.38}} = 0.594$

Input to final neuron = (0.594 x 0.7) + (0.603 x 0.5) = 0.717.

Final Output = $\frac{1}{1+e^{-0.717}} = 0.672$

# Exercise

- Try calculating the output of this network yourself.



# Comparison between brain verses computer

| | Brain | ANN |
|---|---|---|
| Speed | Few ms. | Few nano sec. massive \|\|el processing |
| Size and complexity | $10^{11}$ neurons & $10^{15}$ interconnections | Depends on designer |
| Storage capacity | Stores information in its interconnection or in synapse.<br>No Loss of memory | Contiguous memory locations<br>loss of memory may happen sometimes. |
| Tolerance | Has fault tolerance | No fault tolerance Inf gets disrupted when interconnections are disconnected |
| Control mechanism | Complicated involves chemicals in biological neuron | Simpler in ANN |

# ANN Capabilities & Limitations

**Main capabilities of ANN includes:**

✓ Learn well in complex system (which cannot be solved by mathematical models)

→ Deep Neural Networks



✓ Generalization capability: it can handle large amount of data

✓ Easily implemented in parallel architectures

# ANN Capabilities & Limitations

**Main problems includes**:

❖ ANN is a blackbox, you don’t know how and why an ANN came up with a certain output.

❖ Need a lot of training data

❖ Computationally expensive

❖ Learning sometimes difficult/slow.