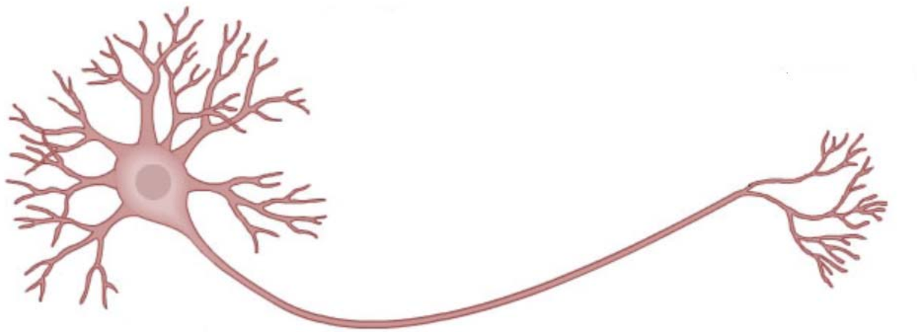
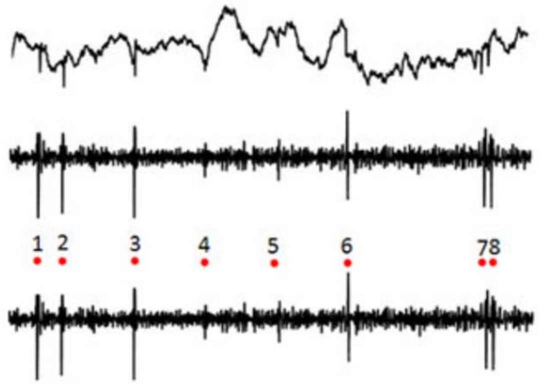
# Neuroengineering (I)
# 1. Artificial Neural Networks

- **Scuola di Ingegneria Industriale e dell'Informazione** – Politecnico di Milano
- Prof. Pietro Cerveri

# From biological to artificial neuron

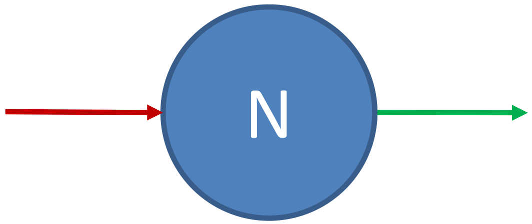
## Information processing point of view






Information encoding: spike frequency, spike sequence pattern, ….

Input




output

Black-box approach

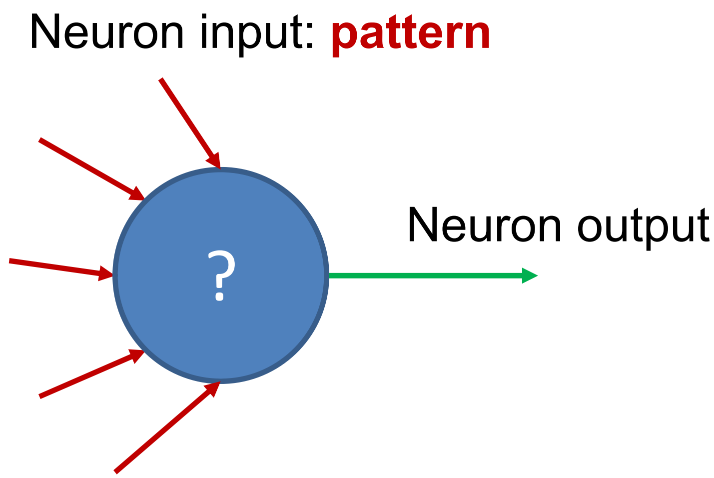
Information encoding: real-valued scalar

# Modeling approach of a neuron (STRUCTURE)

Information processing point of view

Input/output mapping

- Input signals
  - PATTERN: multi-dimensional vector of features
- Input signal processing
  - INTEGRATION
- Action potential
  - EVALUATION
- Activation threshold
  - COMPARISON
- Activation function
  - TRANSFORM TO OUTPUT



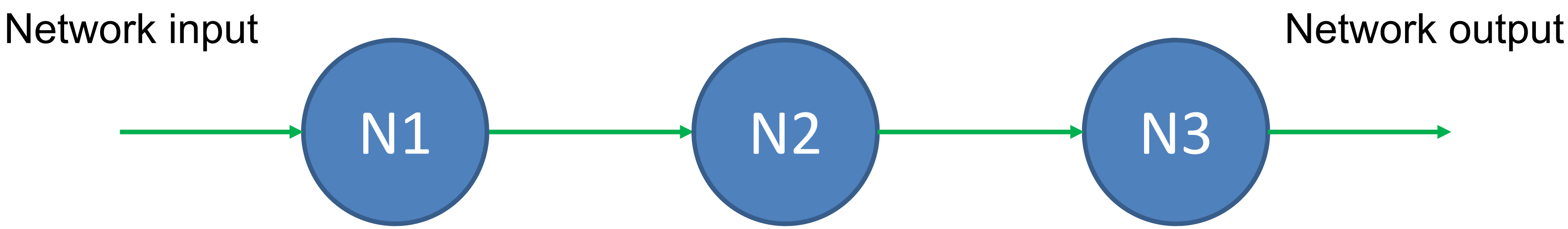

Neural cell abstraction

| Dendrites | Soma | Axon |
| --- | --- | --- |

# Artificial neural network

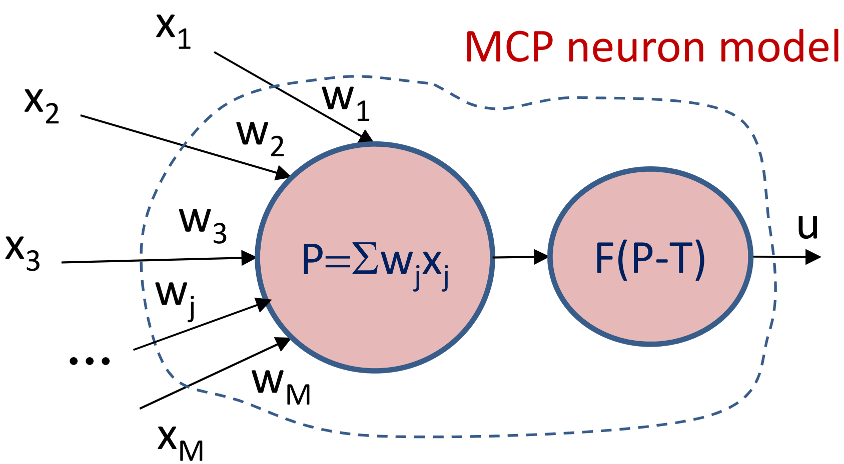
2-component system

1. Set of primary units (neurons, nods, units) each one endowed with input and output lines. Output is triggered by an activation function
2. Set of interconnection lines (channels) among such units: each line is characterized by a number (weight or connection coefficient or synaptic effectiveness) which accounts for the relevance of the signals



# 1943 - McCulloch and Pitts neuron model (MCP)

- Mathematical model of the cerebral cortex based on binary threshold units (formal neurons)
- A network of formal neurons can realize any Boolean function
- The network is equivalent to an automata (finite state machine)




| Dendrites | Soma | Axon |
|---|---|---|

$w_j$ : Weight

>0 Excitation

<0 Inhibition

$x_j$ : state of the input neurons

u : state of the output neuron

F: activation function

T: activation threshold

P activation potential P=f(wj,xj)

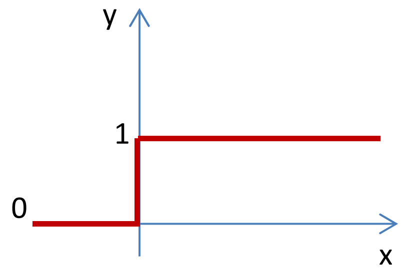
# Threshold binary unit (MCP)

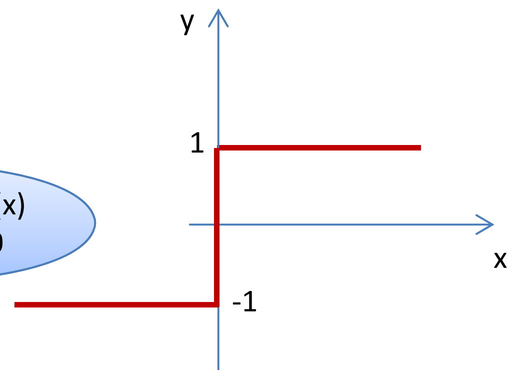
- $T$ is real number named threshold of the unit
- $P$ action potential

$$P = \sum_{j=1}^{M} w_j x_j = w_1 x_1 + w_2 x_2 + \ldots + w_M x_M$$

- $F$ is the activation function
- u is the output with
  - u = F(P-T)
    - F is the Heaviside (step) function
      - F(x) = 1 if x>0 ,
      - F(x) = 0 if x<=0
  - u =sgn(P-T)
    - sgn(x) is the signum function
      - sgn(x) = 1 if x>0
      - sgn(x) = 0 if x=0
      - sgn(x) = -1 if x<0

Actually sgn(x) = -1 if **x<=0**



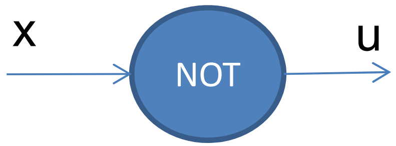



# Boolean units

- Boolean networks
  - Signals are 1 (TRUE) or 0 (FALSE)
  - OR, AND, NOT units
  - The weight setting discriminate the behavior of the unit

NOT

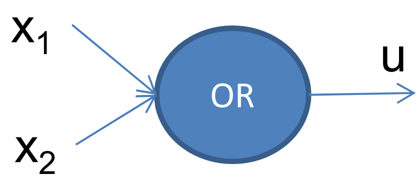
| x | u |
|---|---|
| 0 | 1 |
| 1 | 0 |




$u$ = F($w$*$x$-T)

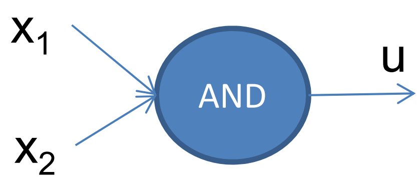
OR

| $x_1$ | $x_2$ | u |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 1 |



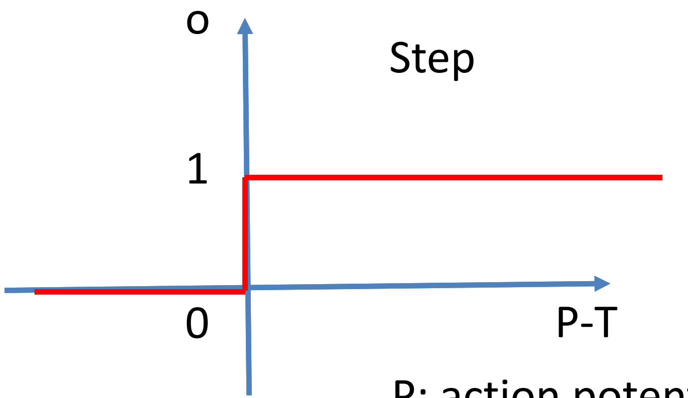

AND

| $x_1$ | $x_2$ | u |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 0 |
| 1 | 0 | 0 |
| 1 | 1 | 1 |




$u = F(w_1 * x_1 + w_2 * x_2 - T)$

# Decision model: binary input, binary output




P: action potential
T: neuron threshold

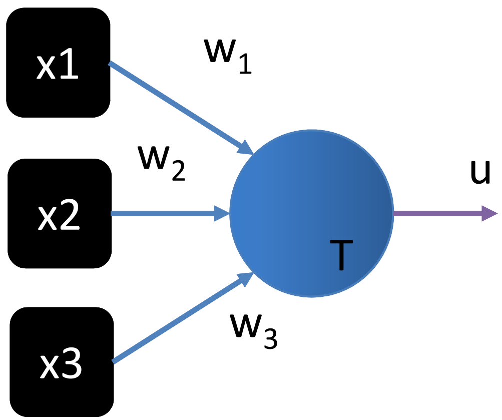
ANN paradigm: combine the three conditions using a neuron with Step activation function

$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

Patient status and exam results are available
How to decide whether administering either drug A or B?

1. Is blood red cell level normal?
2. Is patient temperature higher than normal?
3. Is breath flow lower than normal?

You must assign a score/weight to each possibility

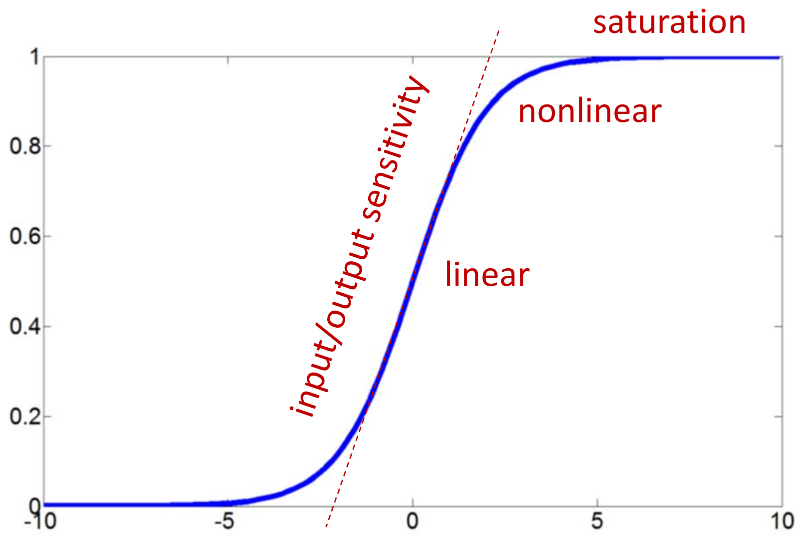


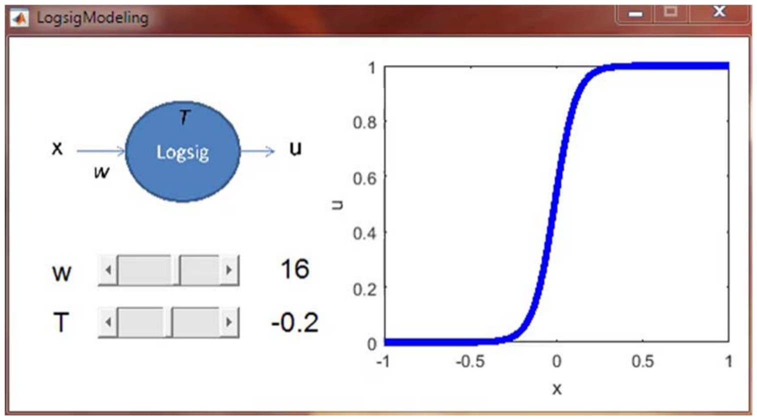
# Sigmoidal activation

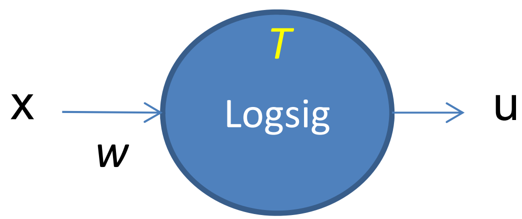
Logsig: the output can assume any real value between 0 and 1

$$u = \frac{1}{1 + e^{-(P-T)}}$$

$$P = wx$$




Changes of the neuron parameters produce dramatic differences in the output

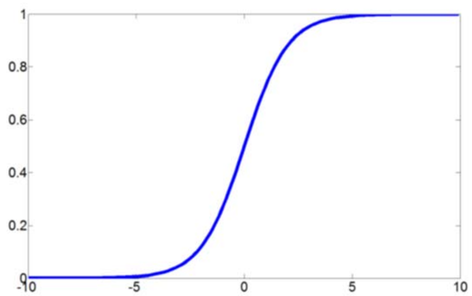
- Input range
- Strength of the response (sensitivity)
- Increase/decrease output

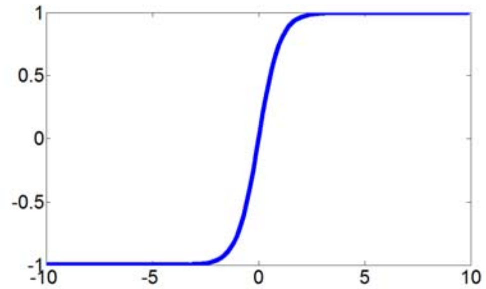
# Sigmoidal activation

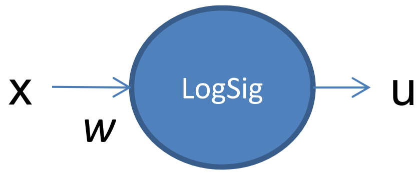
Hyperbolic tangent: the output can assume any real value between -1 and 1

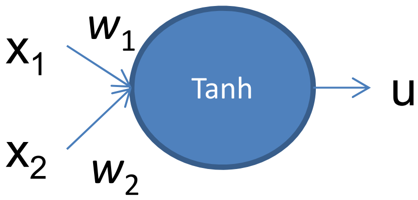
$$u = \frac{1}{1 + e^{-(P-T)}}$$

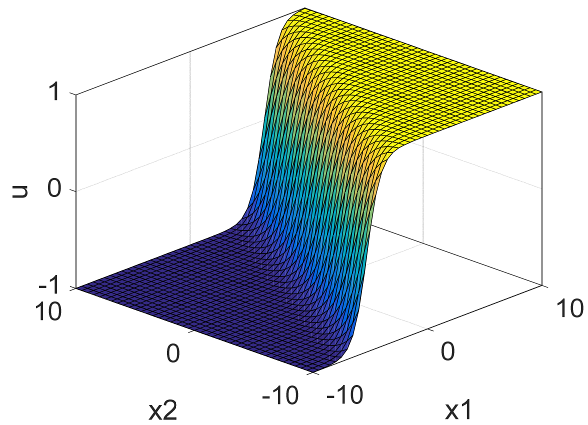
$$u = \tanh(P-T) = \frac{e^{(P-T)} - e^{(P-T)}}{e^{(P-T)} + e^{(P-T)}}$$



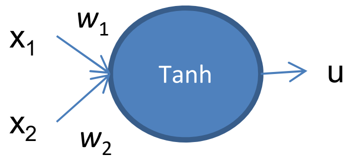



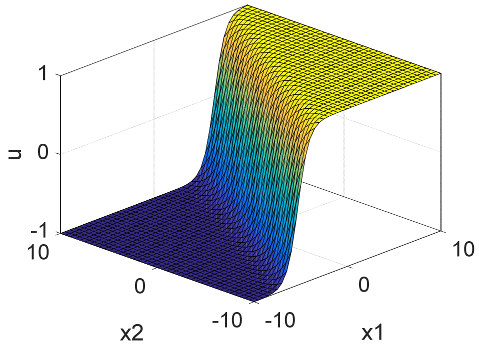



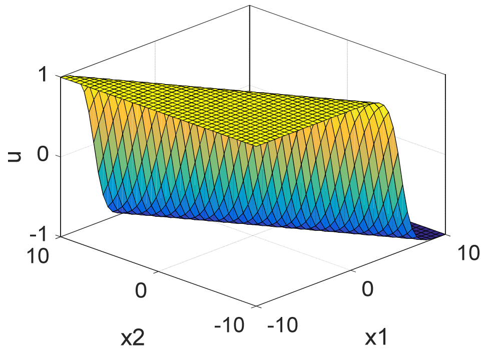
# Example with TANH



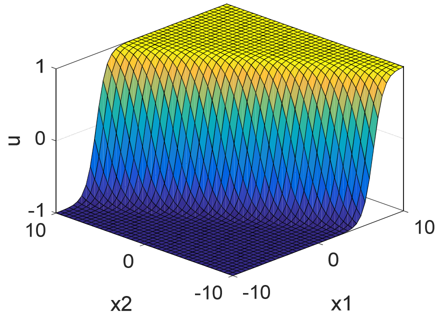

w1 = 0.85; w2 = -0.5; T = 0.5;



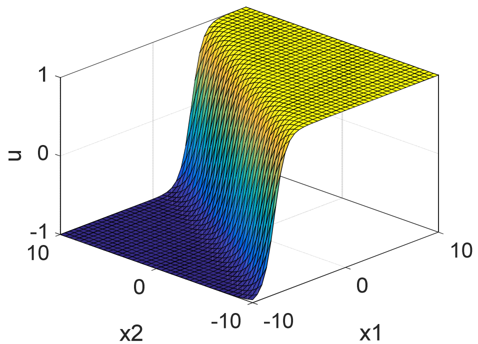

w1 = -0.85; w2 = -0.5; T = 0.5;



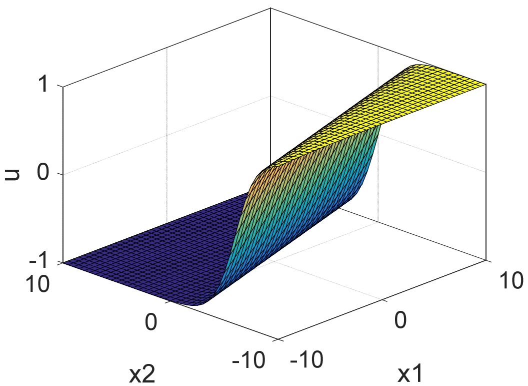

w1 = 0.85; w2 = 0.5; T = 0.5;



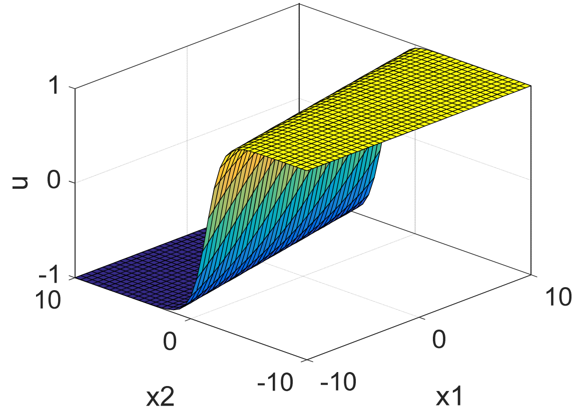

w1 = 0.85; w2 = -0.5; T = -1.5;




w1 = 0.15; w2 = -0.5; T = 1.5;

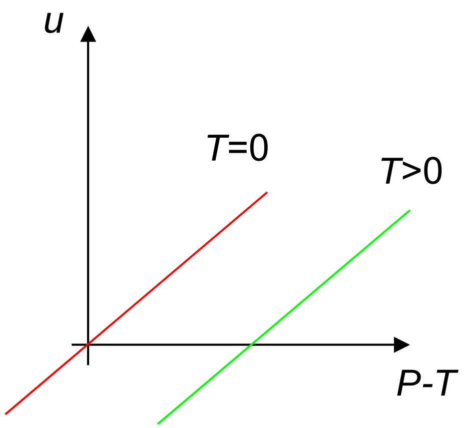


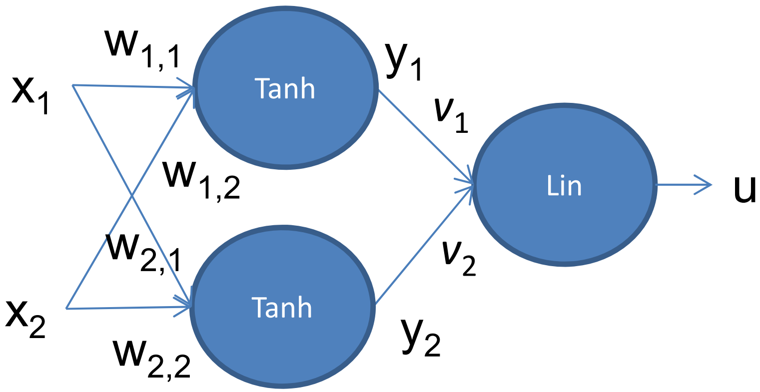

w1 = 0.15; w2 = -0.75; T = -0.5;



# Linear unit

- The output can assume any real value (unbounded)
  - *u=P-T*
  - With or without threshold



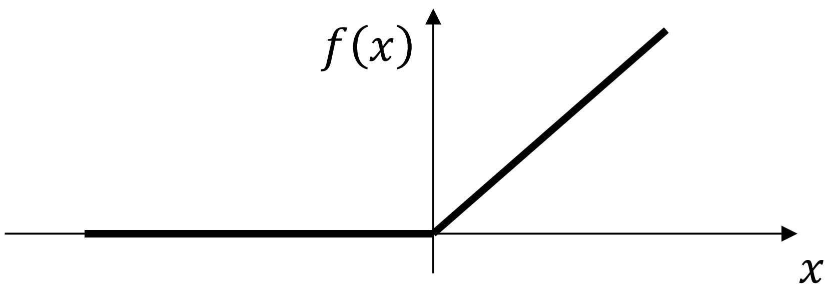




$$u = v_1 * y_1 + v_2 * y_2 - T_3$$
$$y_1 = \tanh(w_{1,1} * x_1 + w_{1,2} * x_2 - T_1)$$
$$y_2 = \tanh(w_{2,1} * x_1 + w_{2,2} * x_2 - T_2)$$

# ReLU
# Rectified linear unit

It performs a non-linear threshold operation, where any input value less than zero is set to zero, i.e.,

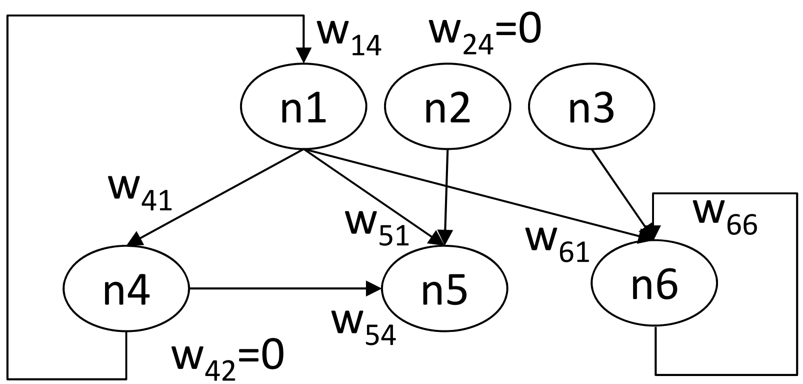
$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$




The ReLU layer does not modify the amplitude of positive input

# Network knowledge

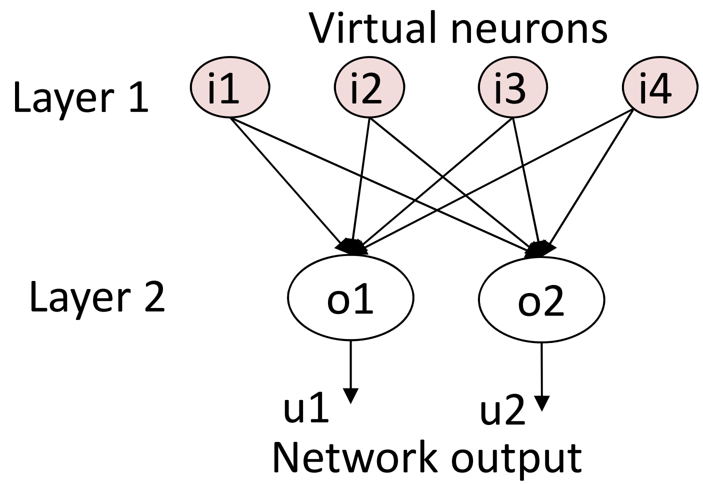
- $w_{ij}$ connection weight between the $j^{th}$ (input) and $i^{th}$ (output) neuron
- Given that $w_{ij}$=0, it does not exist a connection between units j and i
- $w_{ij}$>0 implies that unit j excites unit i
- $w_{ij}$<0 implies that unit j inhibits unit i
- $w_{ii}$: self-connection

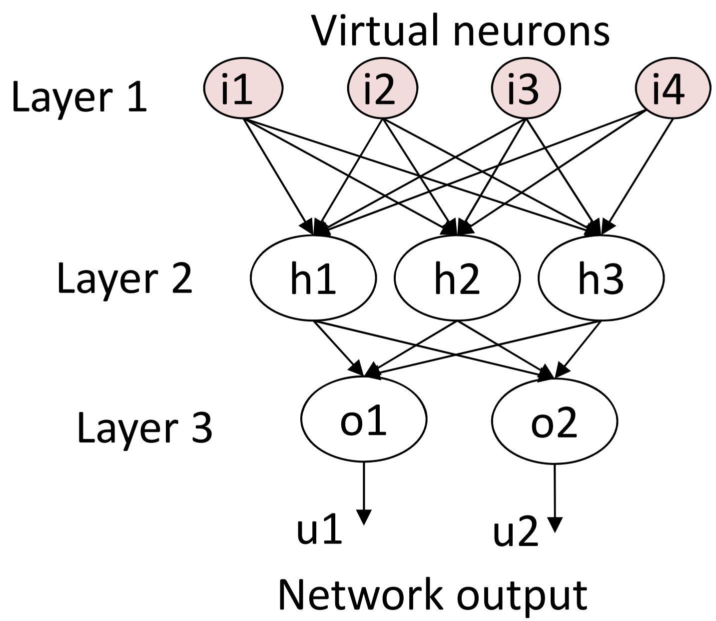



- Interconnection matrix $\{w_{ij}\}$ for a network
- $\{w_{ij}\}= \{w_{ji}\}$: symmetric network
- $\{w_{ij}\}= \{-w_{ji}\}$: anti-symmetric network

# ANN architectures

- Network layers

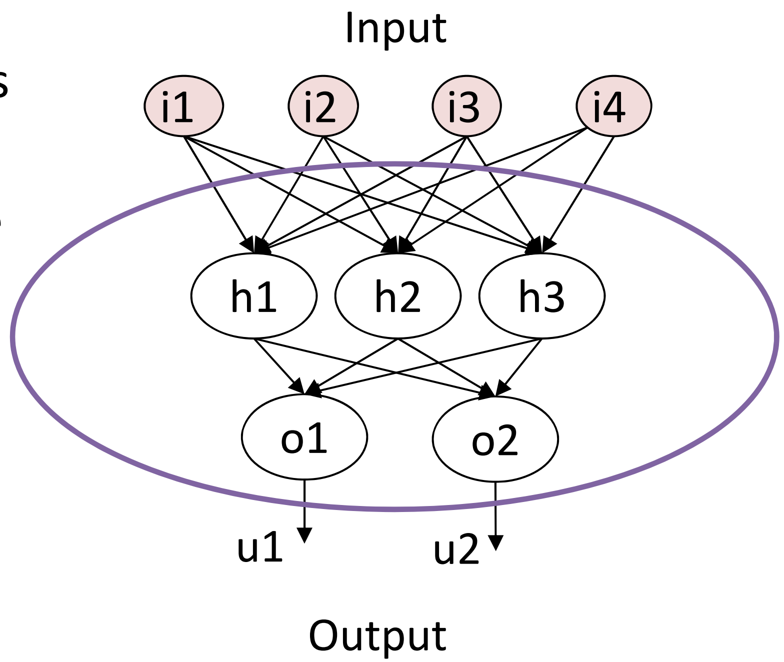


# ANN architectures

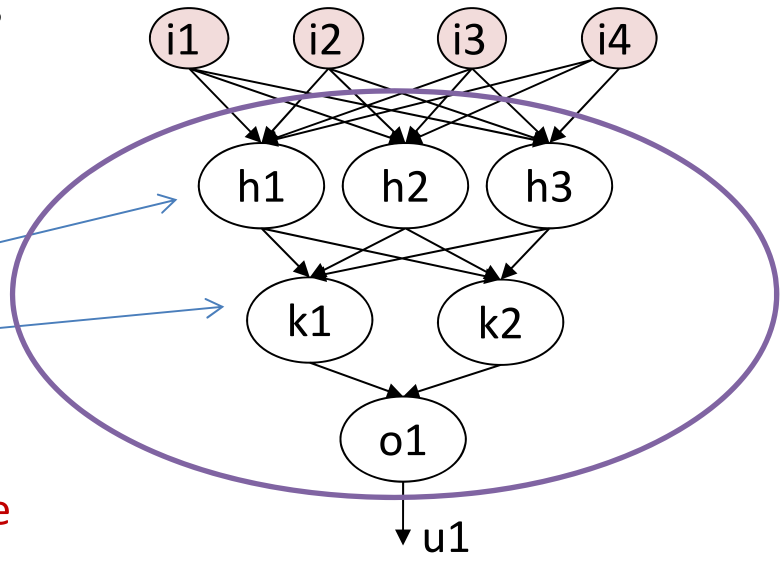
- Network layers
- **Input layer**: neurons in this layer get signals from the outside of the network
- **Output layer**: neurons in this layer produce signals towards outside of the network

❑ Input neurons are significant only for their output signals which are delivered possibly to the network neurons



# ANN architectures

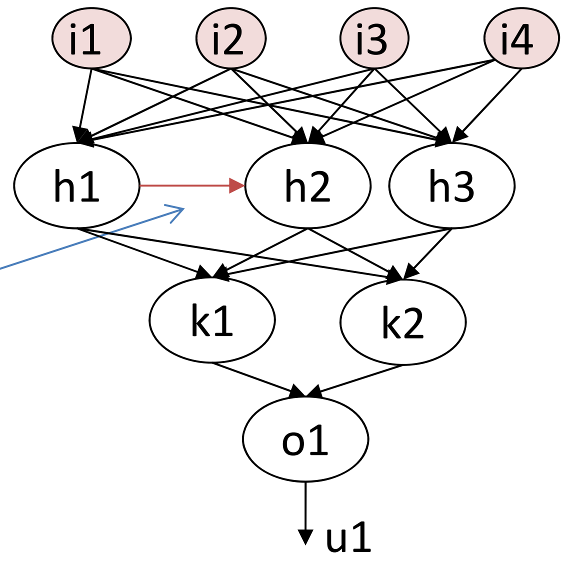
- Network layers
- Input layer: neurons in this layer get signals from the outside of the network
- Output layer: neurons in this layer produce signals towards outside of the network
- Internal (hidden) layers




- ❑ We do not have any corresponding measure of the output of such neurons
- ❑ We can only get information (measure) about network output (u1)
- ❑ Hidden neurons – further processing (usually non-linear)

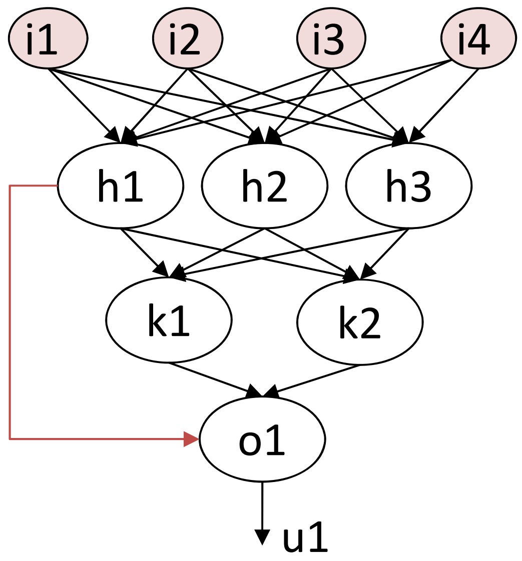
# ANN architectures

- Network layers
- Input layer: neurons in this layer get signals from the outside of the network
- Output layer: neurons in this layer produce signals towards the outside of the network
- Internal (hidden) layers
- Connections:
  - Intra-layer



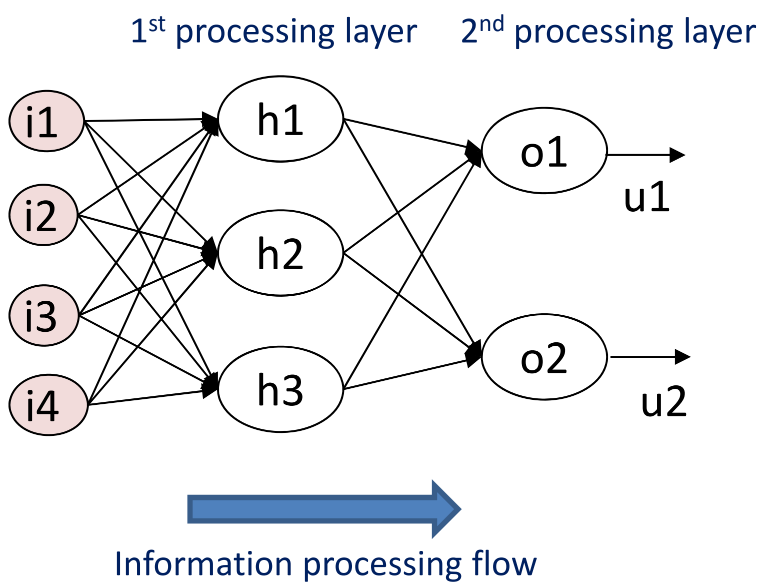
# ANN architectures

- Network layers
- Input layer: neurons in this layer get signals from the outside of the network
- Output layer: neurons in this layer produce signals towards outside of the network
- Internal (hidden) layers
- Connections:
  - Intra-layer
  - Inter-layer



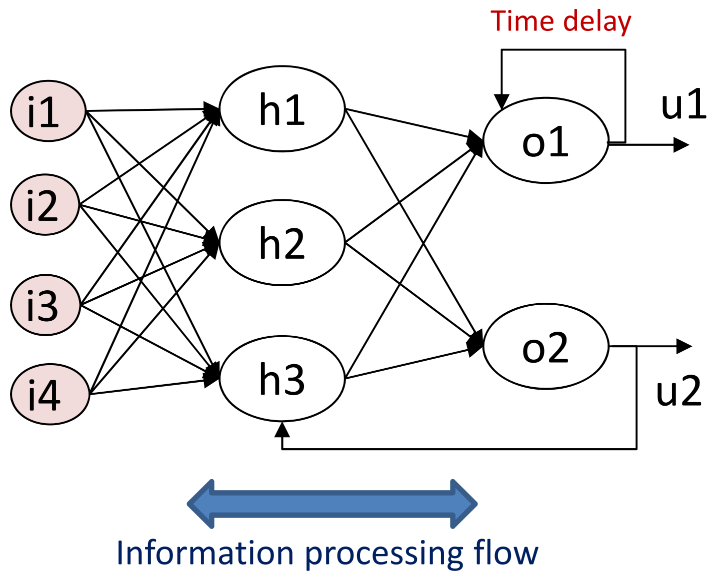
# ANN architectures

- Network layers
- Input layer: neurons in this layer get signals from outside of the network
- Output layer: neurons in this layer produce signals towards outside of the network
- Internal (hidden) layers
- Connections:
  - Intra-layer
  - Inter-layer
  - Feedforward
    - No intra- and inter-layer connections
    - No self-connections
    - No feedback connections

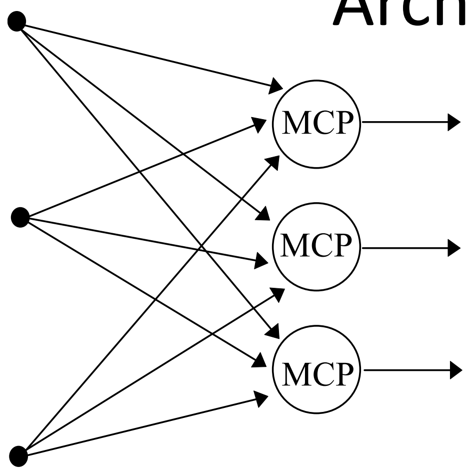


# ANN architectures

- Network layers
- Input layer: neurons in this layer get signals from the outside of the network
- Output layer: neurons in this layer produce signals towards the outside of the network
- Internal (hidden) layers
- Connections:
  - Intra-layer
  - Inter-layer
  - Feedforward – feedback (recurrent)
    - Need a time scale to delay the feedback signal

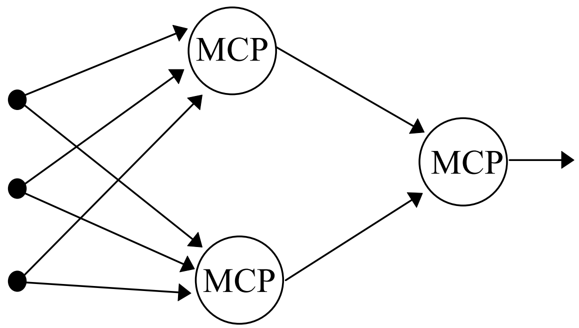


# Architecture synthesis



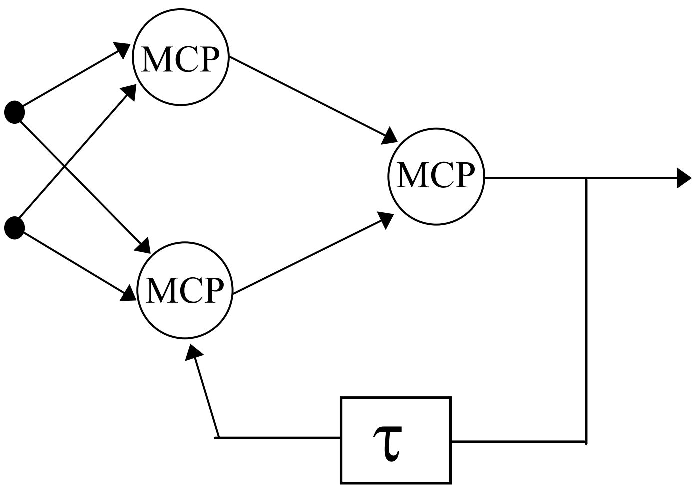

(a)

**Perceptron**

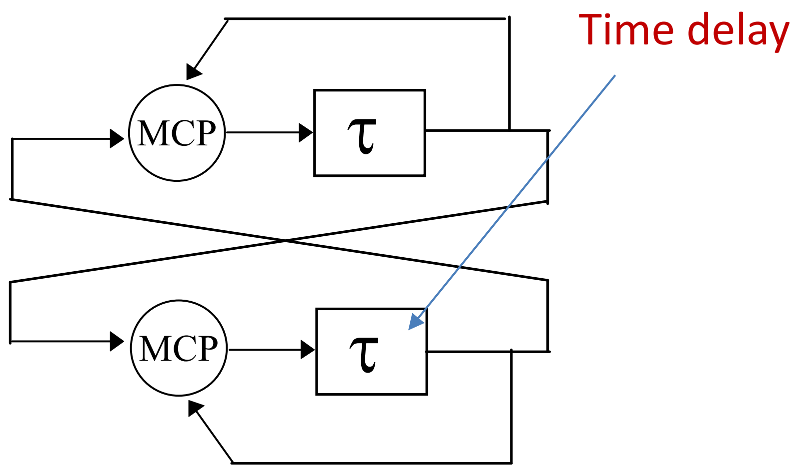



(b)

**Multi-layer perceptron**




**Recurrent**




**Fully-connected recurrent**
***Auto-associative network***

# Typical applications of ANN

- *Pattern* classification and recognition

$l = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in X \subset R^h \quad l \in C \subset N$

- *Function approximation*

$\boldsymbol{y} = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in X \subset R^h \quad \boldsymbol{y} \in Y \subset R^k$

- *Data synthesis and compression*

$\boldsymbol{h} = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in X \subset R^h \quad \boldsymbol{h} \in H \subset R^k$ *k<<h*

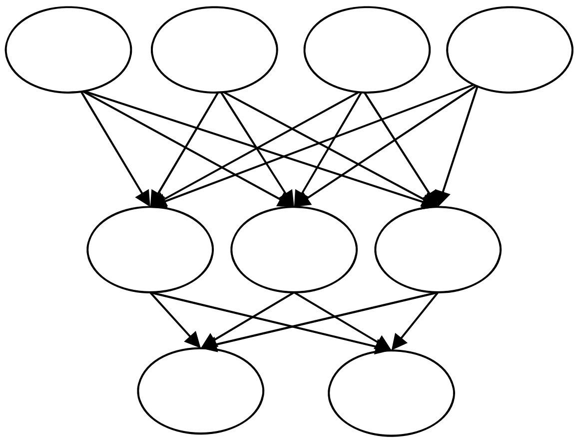
- *Time-series forecasting*

$\boldsymbol{x}(t) = f(\boldsymbol{x}_{t-1}, \boldsymbol{x}_{t-2}, \boldsymbol{x}_{t-3}, \dots)$

$f$ *is unknown*

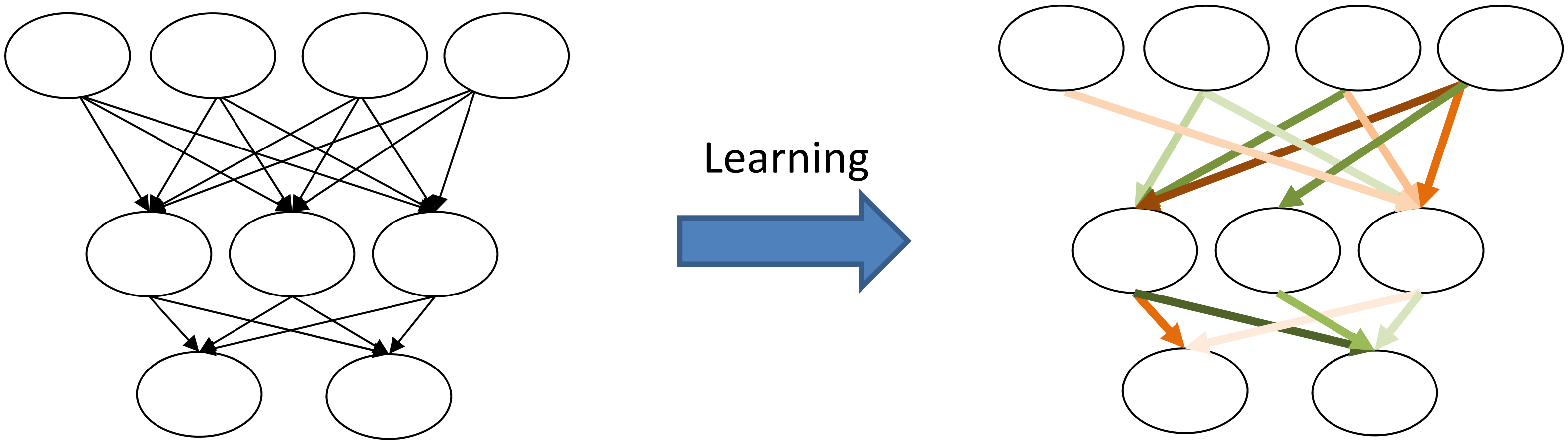
# Neural structure

| Parameters | Hyper-parameters |
|---|---|
| Weights and thresholds | Network architecture (FFNN, RNN, CNN, …) |
| | Number and typology of inputs |
| | Number and typology of outputs |
| | Number of layers |
| | Activation functions |
| | |



# Learning in ANN

it means computing neural weights and thresholds
(network parameters)




*Positive connections (excitatory) vs Negative connections (inhibitory)*

*Learning does not change the topology of the neural structure*
*Learning modifies only the extent of the neural connections*

*Learning requires a set of data called training dataset*

# Learning principles

## Supervised learning

Data: (x, y)
x is input, y is reference label (category, numerical value,…)

Goal: learn a function to map x -> y by minimizing differences between predicted $\underline{y}$ and real output y

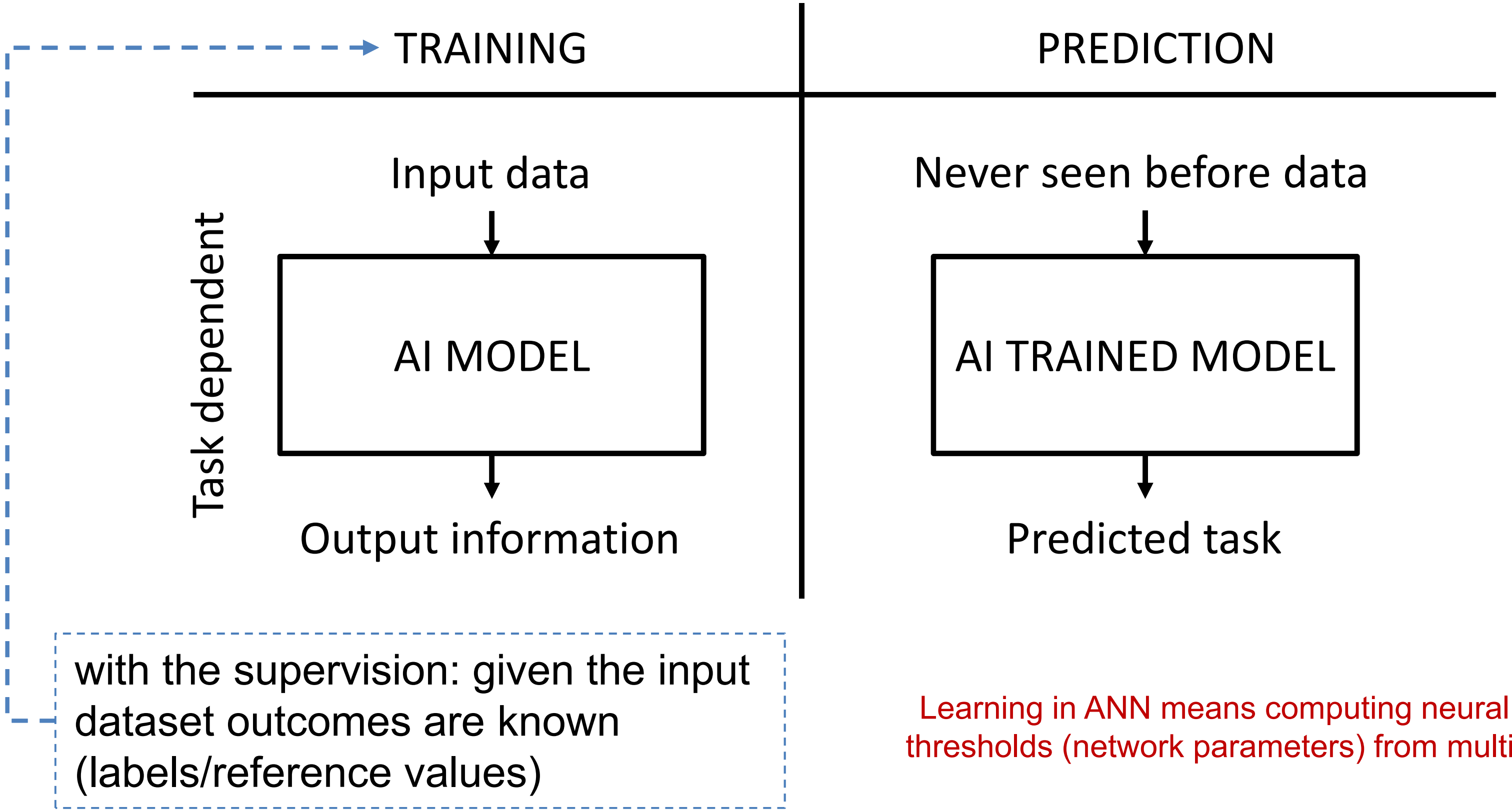
## Unsupervised learning

Data: x
No labels

Goal: learn some underlying hidden structure of the data exploiting coherence rules, relations among inputs, ….

# ANN Supervised Learning (data-driven)

- a computer technique that builds or train a predictive neural model from input data
- applied to heterogeneous data types: text, numerical, images, audios, video, …




Learning in ANN means computing neural weights and thresholds (network parameters) from multiple examples

# Features to explore next

- <u>Learning and architectures</u>
  - Perceptron learning rule
  - Delta learning rule
  - Multi-layer networks and Backpropagation