

# ARTIFICIAL NEURAL NETWORKS

Pedro Larrañaga, Concha Bielza, Jose Luis Moreno

Computational Intelligence Group  
Artificial Intelligence Department  
Universidad Politécnica de Madrid



Computational  
Intelligence  
Group



***Machine Learning***  
Master in Data Science + Master in HMDA

# Outline

- 1 Biological neuron
- 2 Multilayer perceptron
- 3 Deep neural network

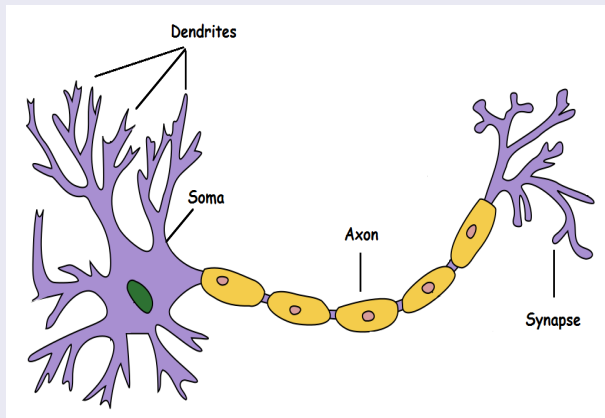
# Outline

- 1 **Biological neuron**
- 2 Multilayer perceptron
- 3 Deep neural network

## Soma, dendrites, axon, synapses

Artificial neural networks are computational models for information processing that attempt to mimic the learning of biological neural networks

### Main elements



# Outline

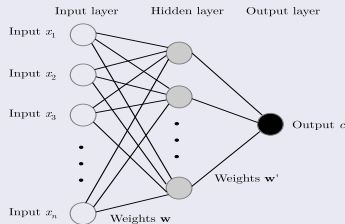
- 1 Biological neuron
- 2 Multilayer perceptron**
- 3 Deep neural network

## MLP

## Input, hidden and output

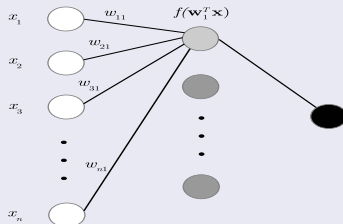
- **MLP**: An ANN arranged as (at least) three layers with **input**, **hidden**, and **output** neurons
- The **edges** connect nodes from one layer to the next one
- These connections consist of a set of **adaptive weights**, that is, numerical parameters that are tuned by the learning algorithm
- The inclusion of **hidden nodes** means that MLPs can approximate **non-linear functions**

## MLP example of structure



## MLP

## Information processing of a single neuron



## A three layer MLP

- $n$  input neurons,  $X_1, \dots, X_n$ ;  $h$  hidden neurons,  $H_1, \dots, H_h$ , and one output neuron,  $C$
- $\mathbf{w}^T = (w_1, \dots, w_h)$  denotes the vector of  $h$  weights connecting input and hidden neurons, where  $\mathbf{w}_j^T = (w_{1j}, w_{2j}, w_{3j}, \dots, w_{nj})$  for  $j = 1, \dots, h$  denotes the vector of weights for the  $j$ -th hidden neuron
- The result of applying the transfer function to this  $j$ -th hidden unit is denoted as  $f(\mathbf{w}_j^T \mathbf{x}) = f(\sum_{i=1}^n w_{ij} x_i)$
- The  $h$  outputs,  $f(\mathbf{w}_1^T \mathbf{x}), \dots, f(\mathbf{w}_h^T \mathbf{x})$ , provided by the  $h$  hidden units should be weighted with the vector  $\mathbf{w}'^T = (w'_1, \dots, w'_h)$ , resulting in the output of the MLP, that is,  

$$\hat{c} = \sum_{j=1}^h w'_j f(\mathbf{w}_j^T \mathbf{x}) = \sum_{j=1}^h w'_j f(\sum_{i=1}^n w_{ij} x_i)$$
- This output  $\hat{c}$  is compared with the real label  $c$  that is known
- MLP weights such that the  $N$  predictions,  $\hat{c}^1, \dots, \hat{c}^N$ , are as close as possible to the true labels,  $c^1, \dots, c^N$

## MLP training process

### A three step process

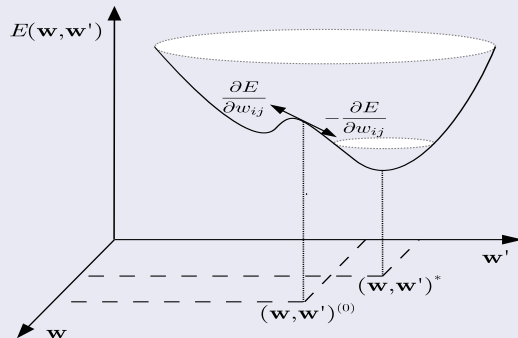
- 1 The MLP is fed with **training instances**
- 2 The **input values for each training instance are weighted and summed** at each hidden layer neuron and the **transfer function** converts the weighted sum into the **input of the output node layer**. The **MLP output values** are calculated and **compared to known labels** to determine how closely the actual MLP matches the desired labels
- 3 The **weights of the connections are changed** so that the MLP can provide a **better approximation** of the desired labels

This process repeats many times until differences between MLP output values and the known labels for all training instances are as small as possible



## MLP

## Backpropagation



Gradient descent method for the minimization of  $E(\mathbf{w}, \mathbf{w}')$

## MLP training process

Minimization of  $E(\mathbf{w}, \mathbf{w}')$ 

- The mean squared error,  $E(\mathbf{w}, \mathbf{w}') = \frac{1}{N} \sum_{k=1}^N (c^k - \hat{c}^k)^2$ , is an example of an error measure often used as the **objective function to be minimized**
- The most important method to solve this unconstrained nonlinear optimization problem is the **backpropagation algorithm**
- This algorithm is a **gradient descent method or method of steepest descent** that finds the direction in which it is best to change the weights in the error space to reduce the error measure most
- This requires **partial derivatives of the error function  $E$  with respect to each weight  $w_{ij}$**  to be calculated
- The weight updating from  $w_{ij}^{old}$  to  $w_{ij}^{new}$ :  $w_{ij}^{new} = w_{ij}^{old} - \eta \frac{\partial E}{\partial w_{ij}}$  is the gradient of the error function  $E$  with respect to  $w_{ij}$  and  $\eta$  is called the learning rate and controls the size of the gradient descent step
- The backpropagation algorithm requires an **iterative process**, and there are two versions of weight updating schemes: **batch mode** and **on-line mode**
- **Batch mode**: weights are updated after all training instances are evaluated
- **On-line mode**, the weights are updated after each instance evaluation
- Each pass of all instances is called an **epoch**

## MLP. Advantages and disadvantages

### Advantages

- They **do not require a priori assumptions** about the underlying data generating process
- The modeling process is **highly adaptive**
- Well-established **mathematical properties** for accurately approximating functions
- **Nonlinear and nonparametric** models
- They are **fault tolerant**, able to model incomplete and noisy information

### Disadvantages

- **High computational burden**
- **Overfitting**
- **Black box** systems

# Outline

- 1 Biological neuron
- 2 Multilayer perceptron
- 3 Deep neural network**

## Deep neural network

## Deep neural network

