Deep Learning Fundamentals

Mihai DOGARIU

www.mdogariu.aimultimedialab.ro

Summary

M1. Introduction

M2. The basic process of learning

M3. Terminology

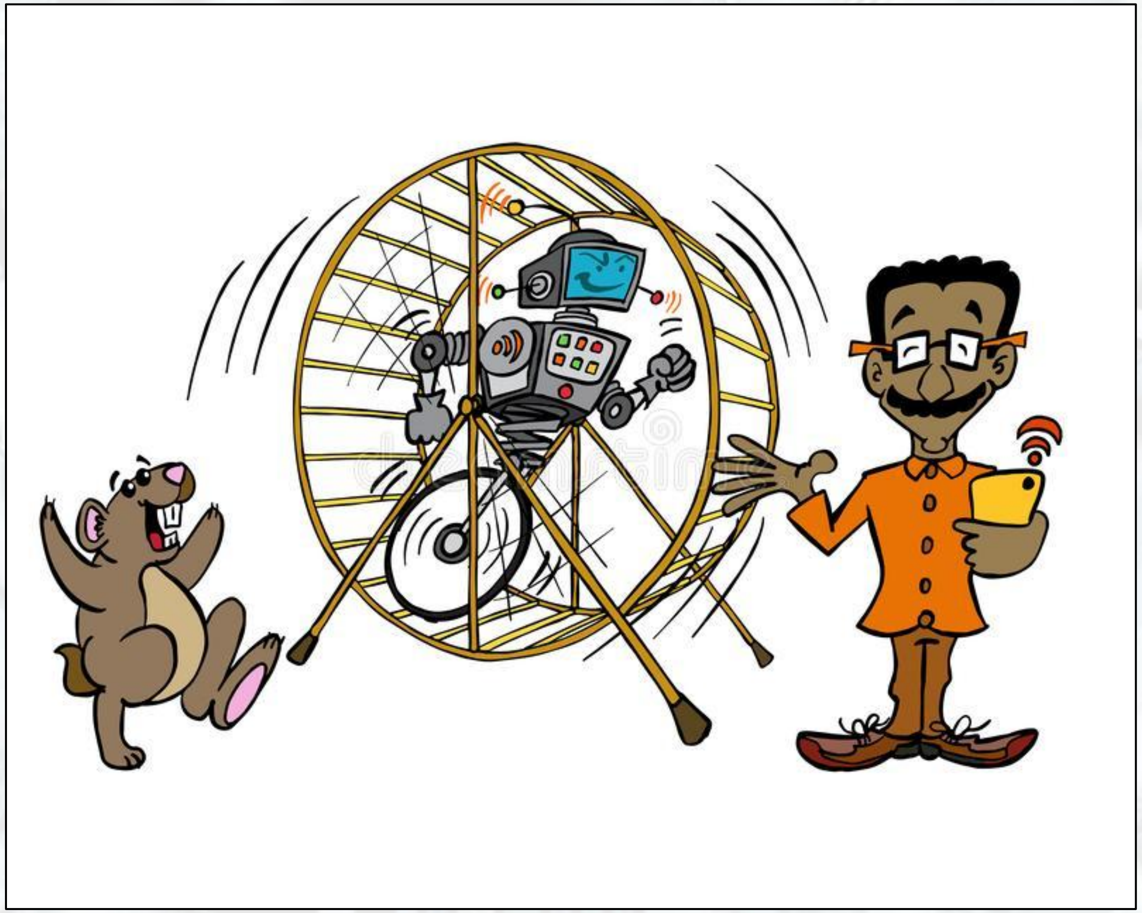
M4. Practical considerations

M1. Introduction

Image: www.dreamstime.com

Identifying the problems

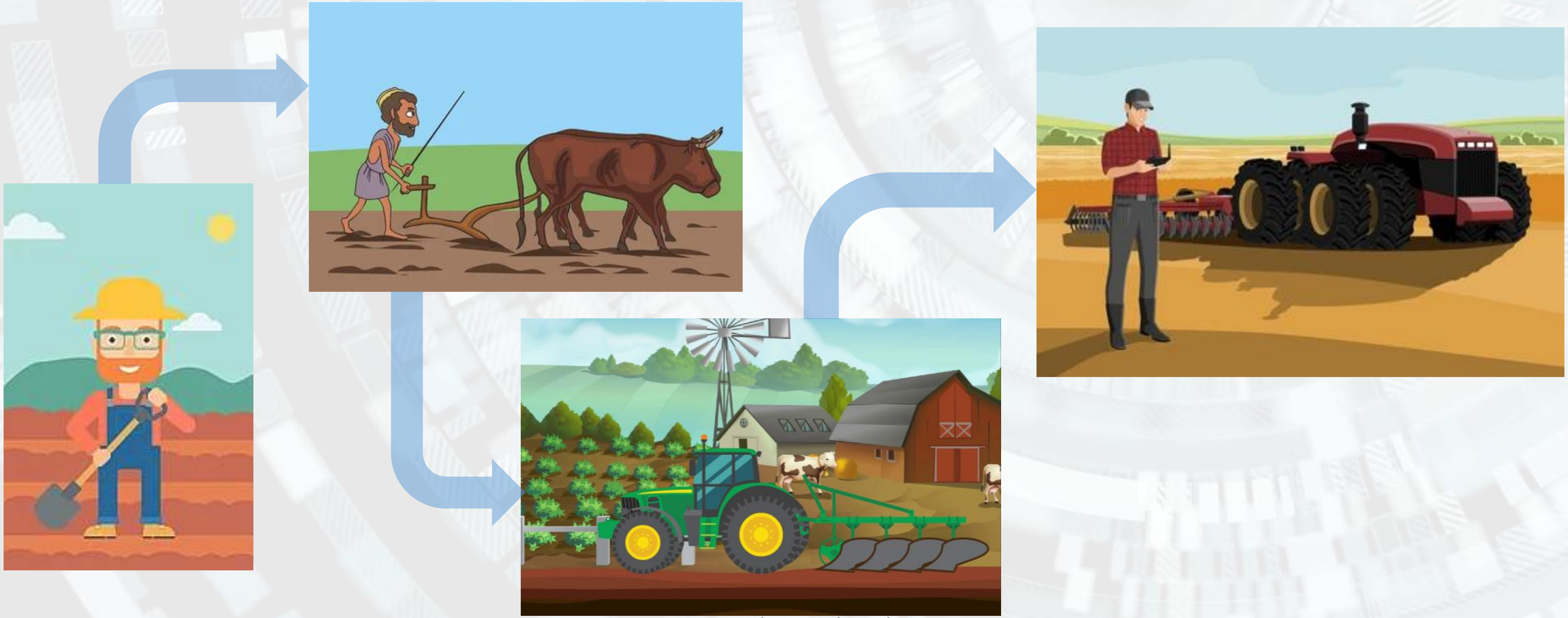
- ➢Need for process automation (redundant ones, most of the time).
- ➢Human capacity is limited and prone to erros, especially after longer time periods (approx. 6h).
- ➢Increasing cost/efficiency ratio.



Images: www.canstockphoto.com, www.123rf.com, www.shutterstock.com

Solving the problems

- Externalizing/replacing human effort.



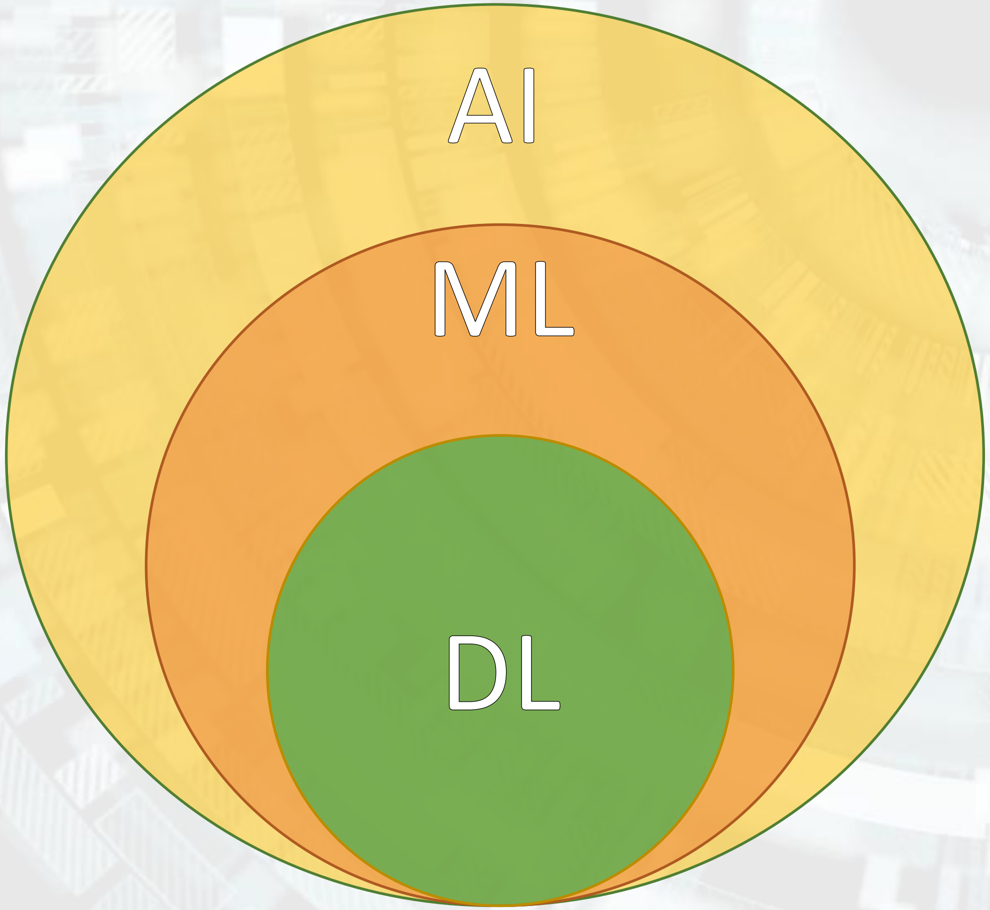
AI, ML, DL?

Artificial Intelligence (AI) = the ability of computer systems to perform tasks normally requiring human intelligence.

Machine Learning (ML) = AI subdomain in which systems are designed with the ability to learn based on previous examples.

Deep Learning (DL) = ML subdomain in which systems are designed based on the human neural network model.

AI, ML, DL?



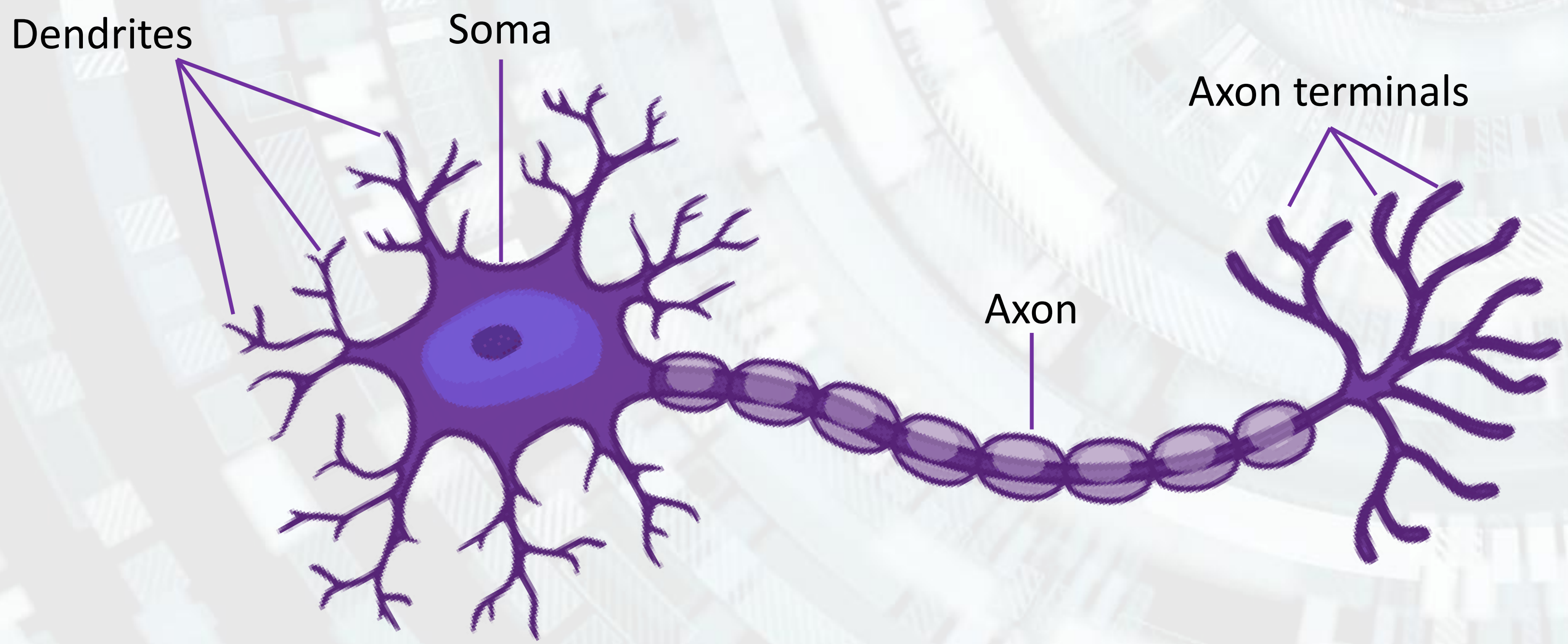
The Deep Learning Boom

What lead to the current exponential advancement of deep learning?

1. Higher computing power (hardware) – GPU Let's test it out – unit #1
2. A lot more data => better results
3. Optimized frameworks (software): Tensorflow, PyTorch, Caffe, MXNet, DeepLearning4J etc.
4. Attention and financial influx from the industry: Facebook AI Research (FAIR), Google Deepmind, NVIDIA, OpenAI, Microsoft Research, AWS Deep Learning etc.

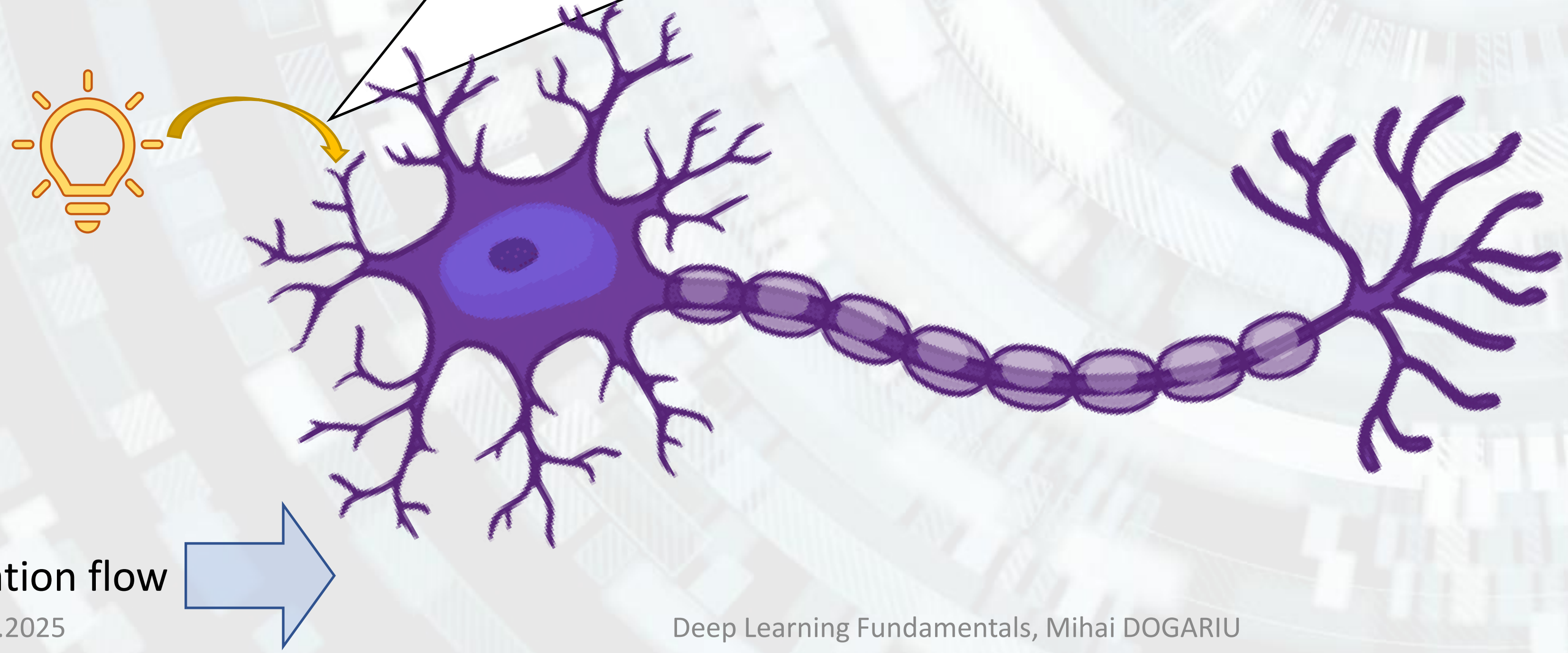
The biologic neuron

- The fundamental cell of the nervous system



The biologic neuron

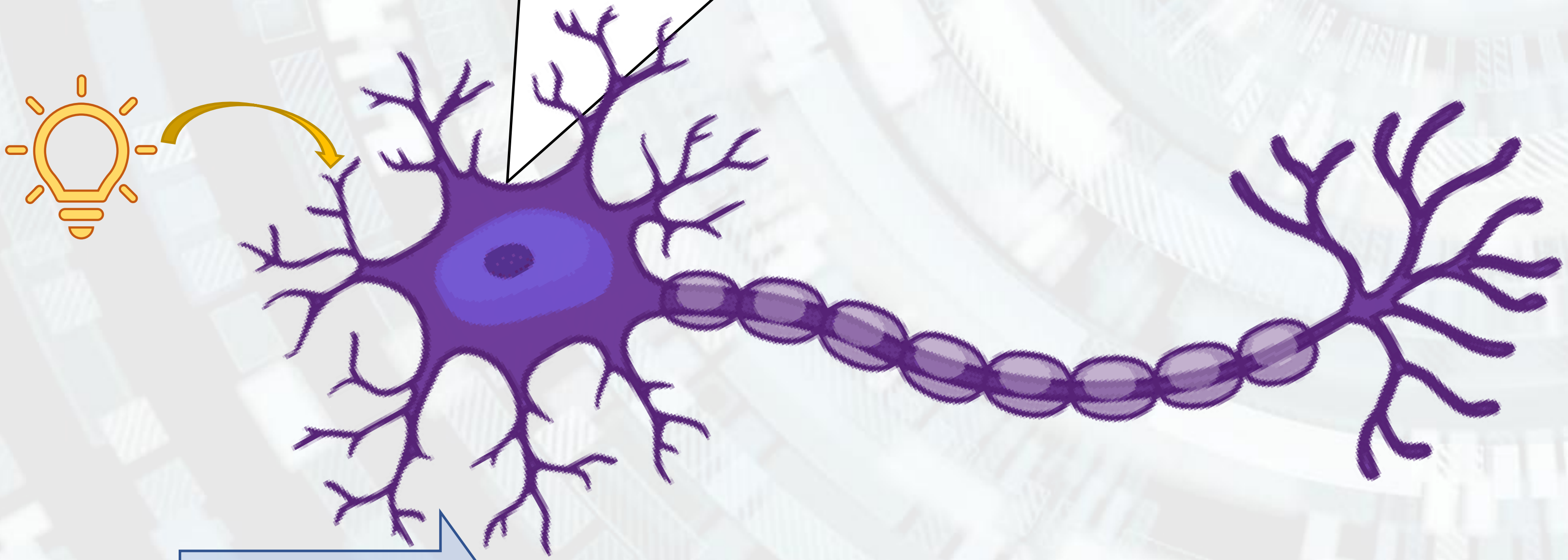
Dendrites are the input gate for neurons. They are connected to and receive signals from other neurons (>1000). Each dendrite weights the signal that it receives in a different manner (inhibit or excite).



Information flow

The biologic neuron

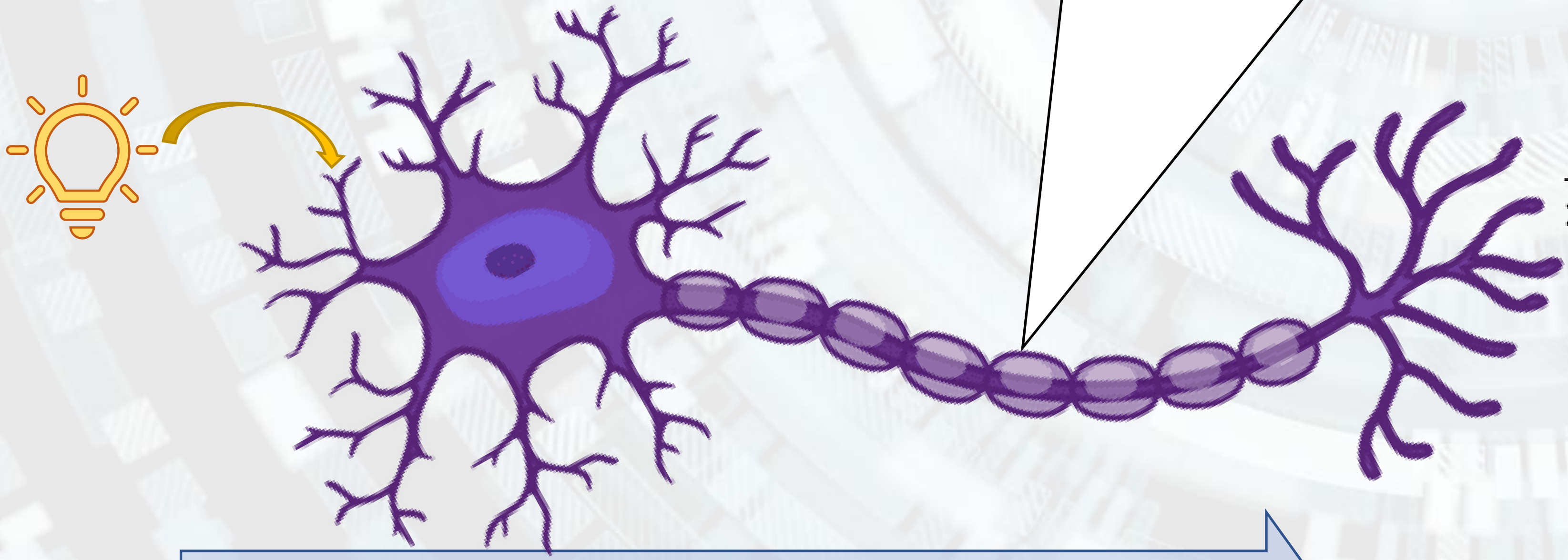
The soma gathers all synaptic signals and sums their electrical potentials.

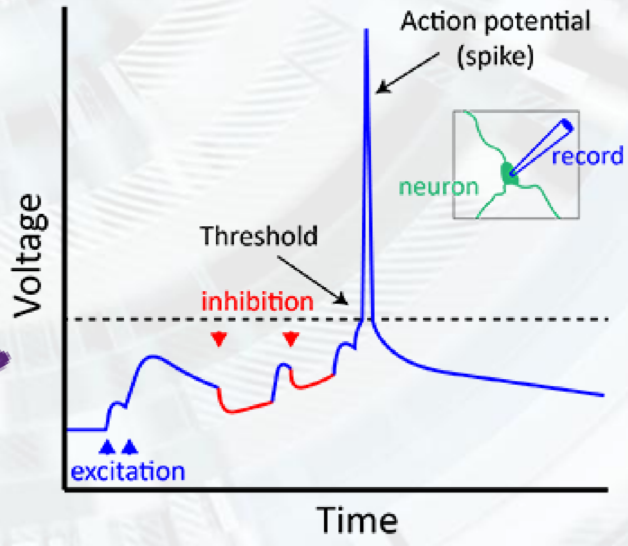


Information flow

The biologic neuron

If the electrical charge from the soma exceeds a given threshold (bias), then the neuron fires and the signal is transmitted onwards.





Information flow

Image: Pinterest

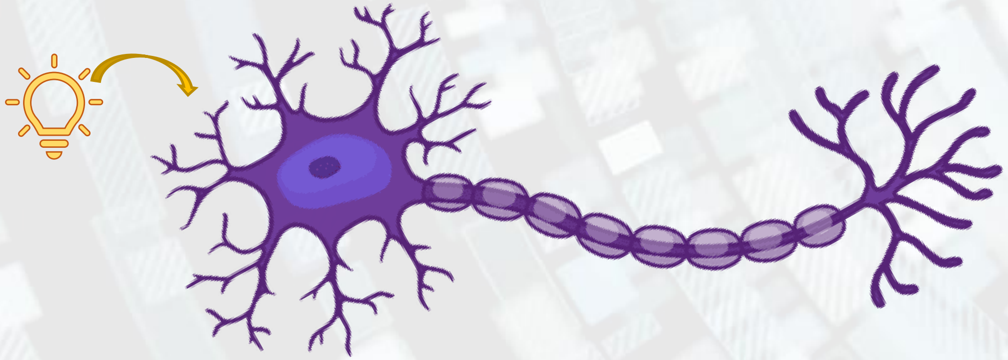
The biologic neuron

If the neuron was activated, its signal is sent to all other neurons connected to its axon terminals.

Information flow

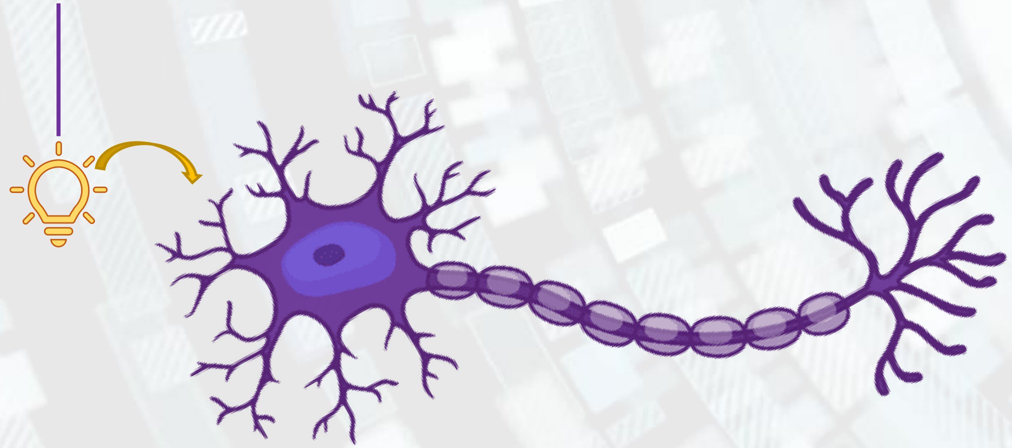
The artificial neuron

Represents a mathematical model of the biologic neuron.



The artificial neuron

The biologic neuron



The artificial neuron

x_1

x_2

x_3

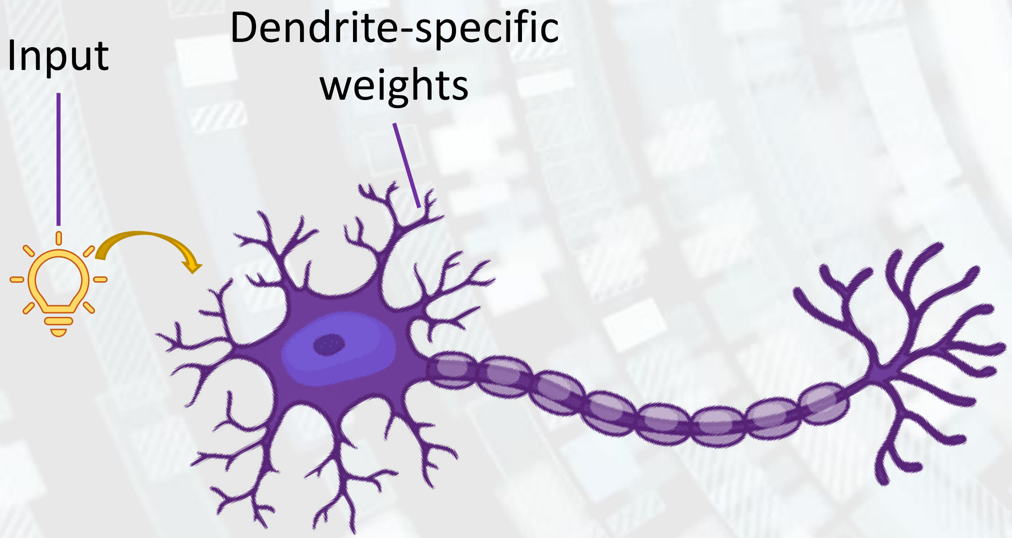
x_4

$\vdots$

x_n

The artificial neuron

The biologic neuron

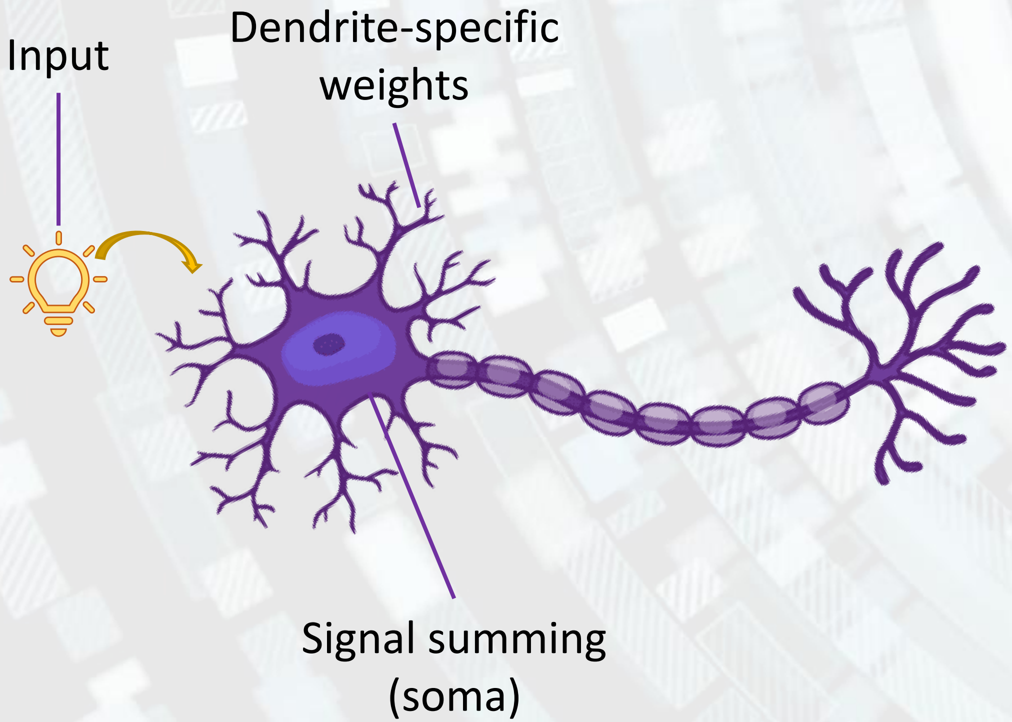


The artificial neuron

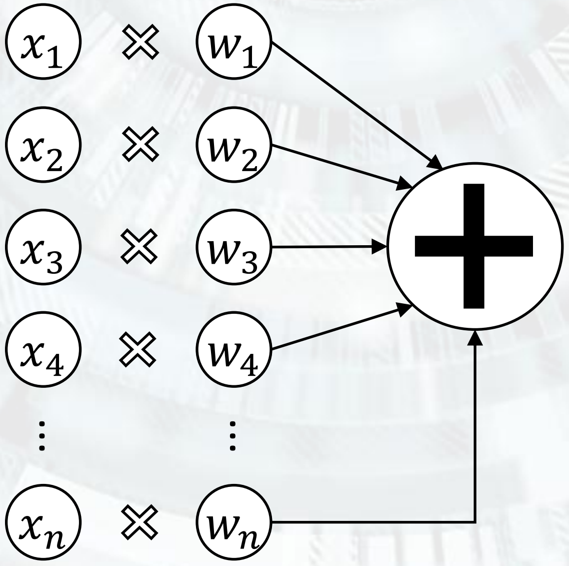
$$
\begin{array}{ccc}
x_1 & \otimes & w_1 \\
x_2 & \otimes & w_2 \\
x_3 & \otimes & w_3 \\
x_4 & \otimes & w_4 \\
\vdots & & \vdots \\
x_N & \otimes & n
\end{array}
$$

The artificial neuron

The biologic neuron

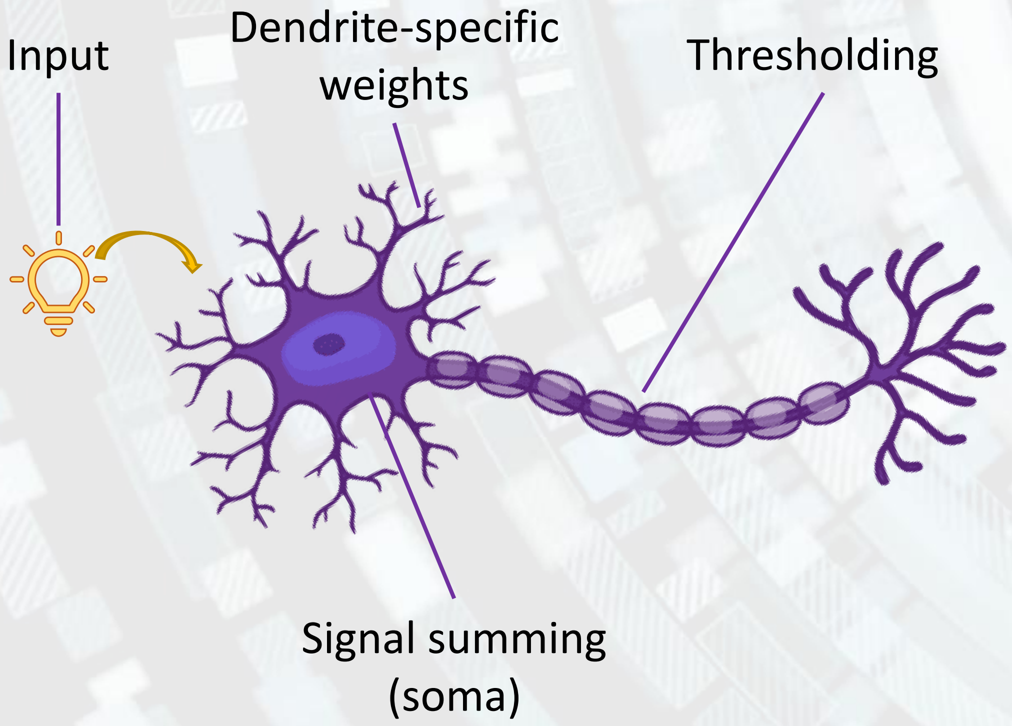


The artificial neuron

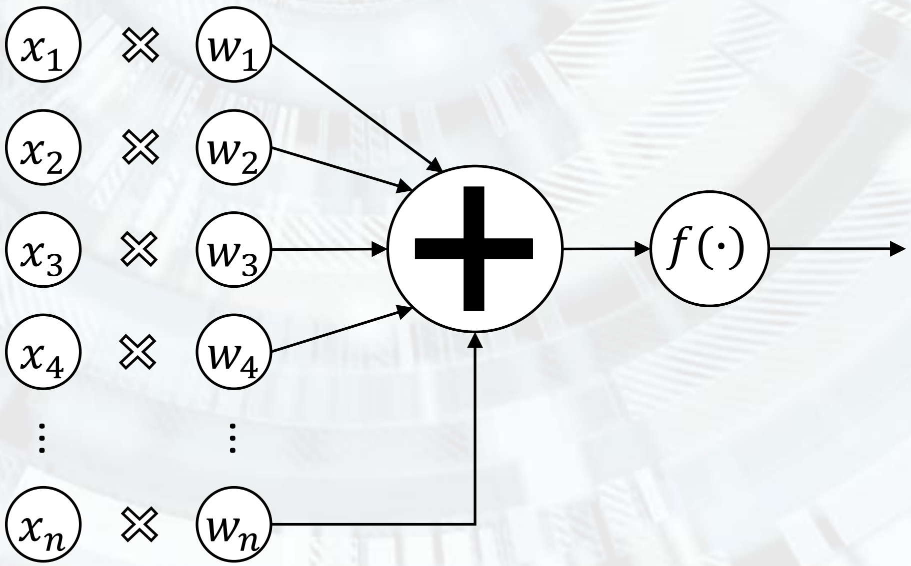


The artificial neuron

The biologic neuron

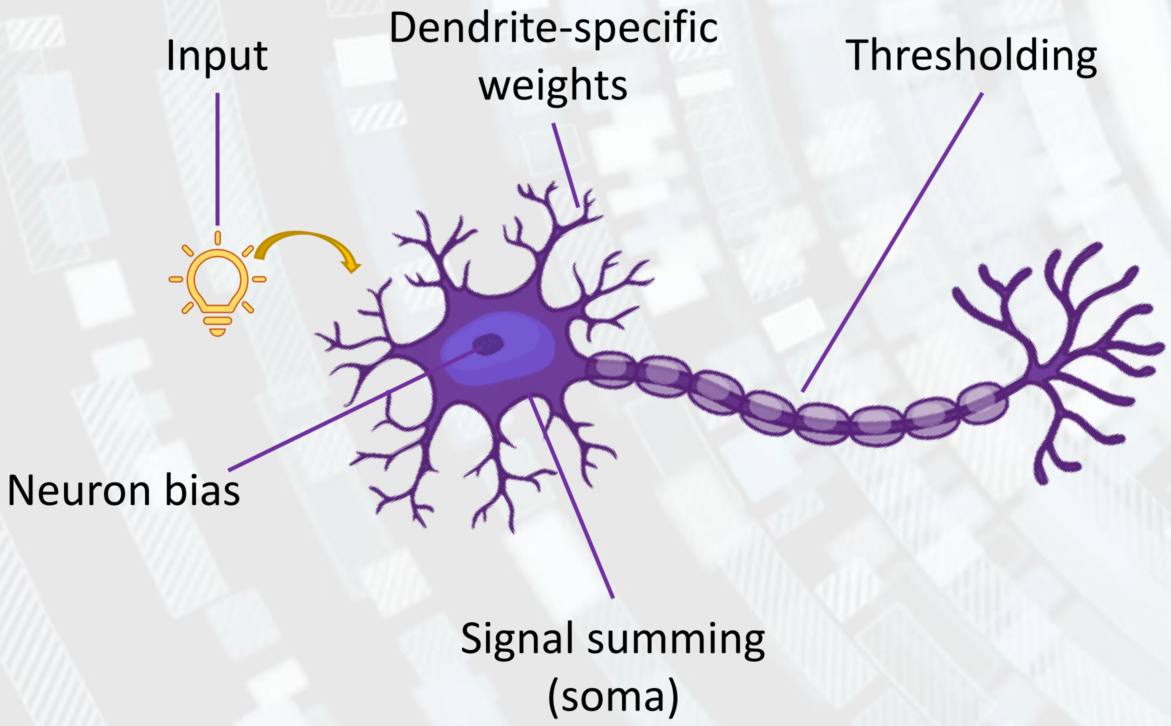


The artificial neuron

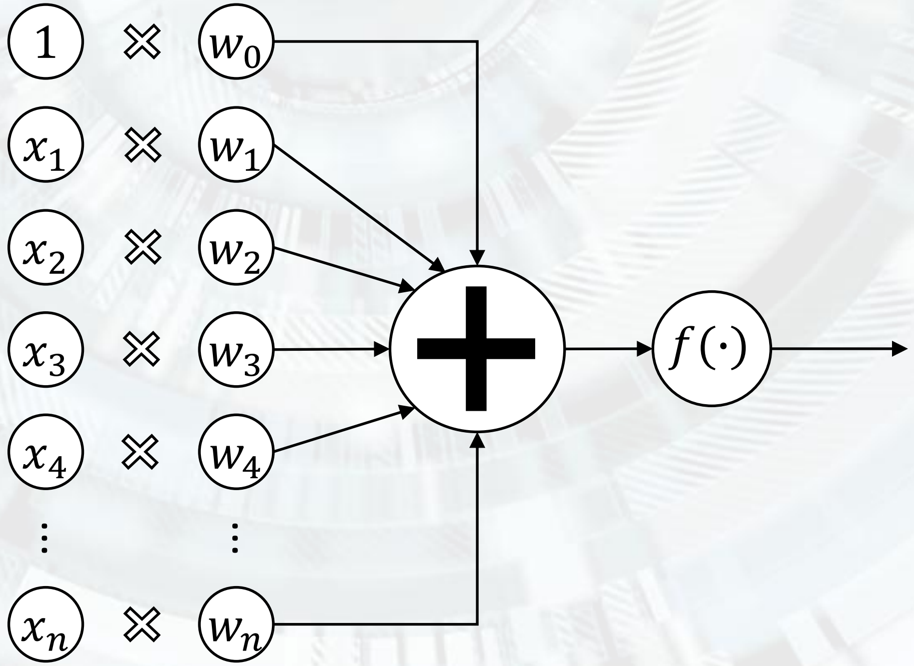


The artificial neuron

The biologic neuron



The artificial neuron



The artificial neuron

The mathematical model:

$$y = f\left(\sum_{i=0}^{n} w_i x_i\right) = f\left(\sum_{i=1}^{n} w_i x_i + b\right) = f(\boldsymbol{w} \cdot \boldsymbol{x} + b)$$

w_i – weights; $w_0 \rightarrow b$ - bias
x_i – input vector; $x_0 = 1$
n – input vector dimension
$f(\cdot)$ – activation function
y – neuron output.

The artificial neuron – graphical representation

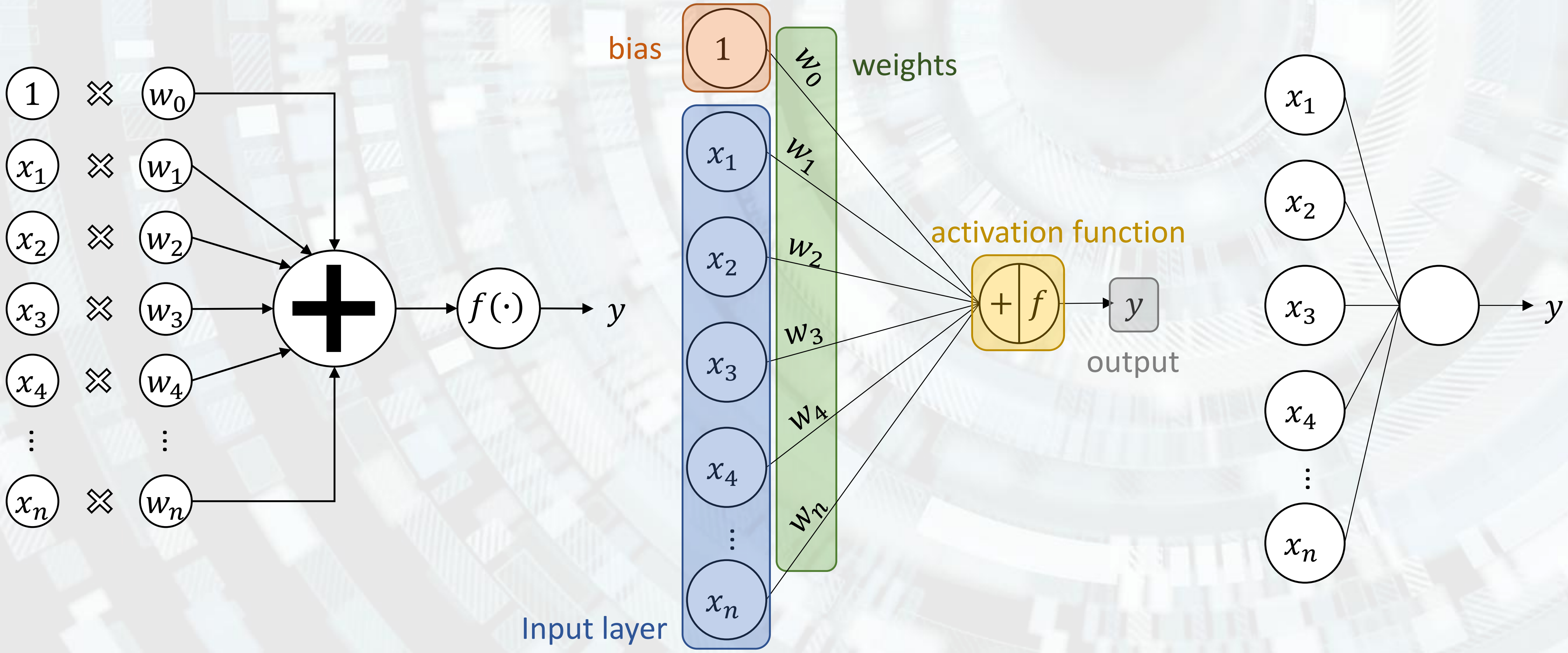
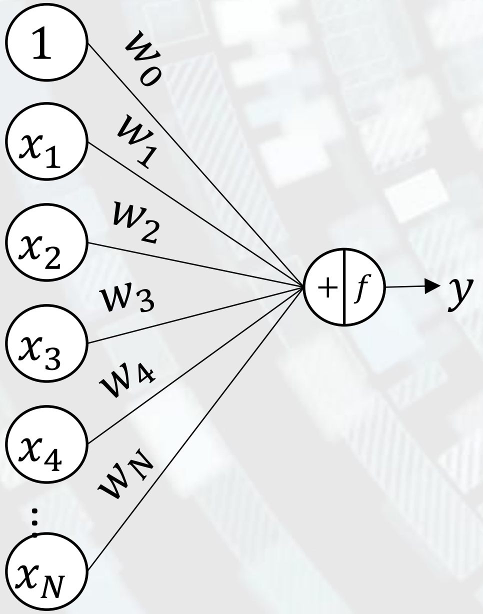


Image: veryicon

Learning of an artificial neuron

Learning = iterative process through which all parameters belonging to the neuron are learned such that the model correctly classifies the input data.



What are the fixed parameters?
- input
- output
- transfer function

What are the adjustable parameters?
- weights
- bias

Image: veryicon

Learning of an artificial neuron

<u>Algorithm:</u>

1. initialize weights with a low enough value.
2. for each input-output pair $(\boldsymbol{x}_j, \hat{y}_j)$ from the training set:

   2.1. compute the output of the system:

$$y_j(\mathrm{t}) = f\left(\sum_{i=0}^{n} w_i x_{j,i}\right)$$

   2.2. update the weights, $\forall i, 0 \leq i \leq n$:

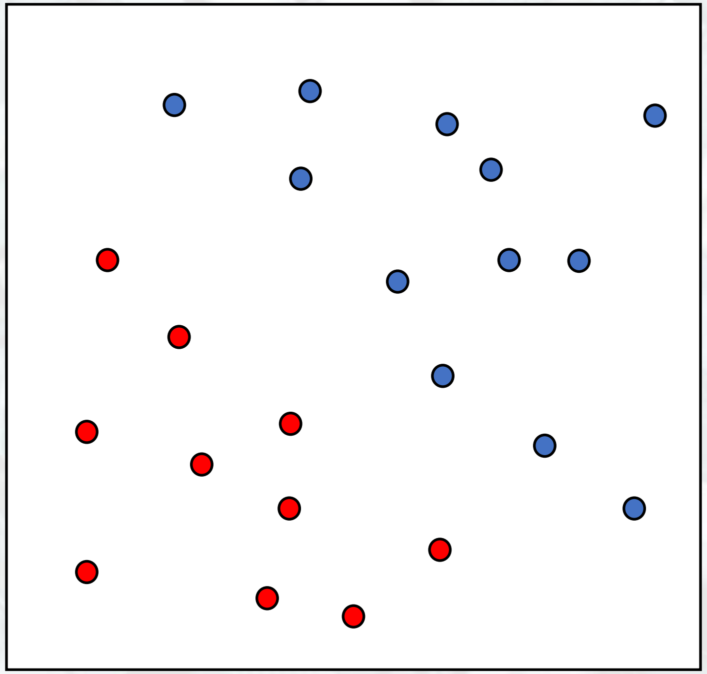
$$w_i(\mathrm{t}+1) = w_i(t) + \alpha * \left(\hat{y}_j - y_j(\mathrm{t})\right) * x_{j,i}$$

3. repeat step 2 until the iteration error is lower than a given threshold or until sufficient iterations have been run.

Let’s test it out – unit #2

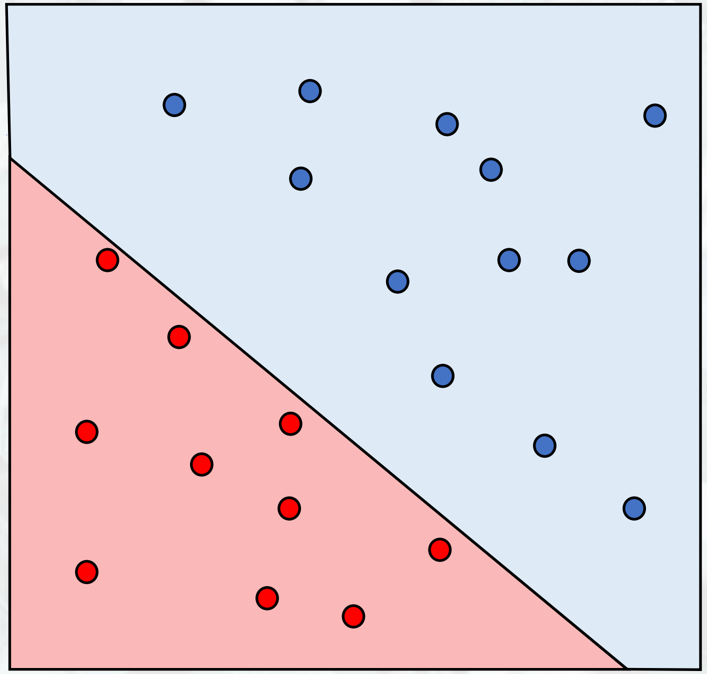
Linear separable space

A bidimensional space consisting of 2 classes of points is linearly separable if there exists at least one straight line separating the space into its 2 distinct classes.



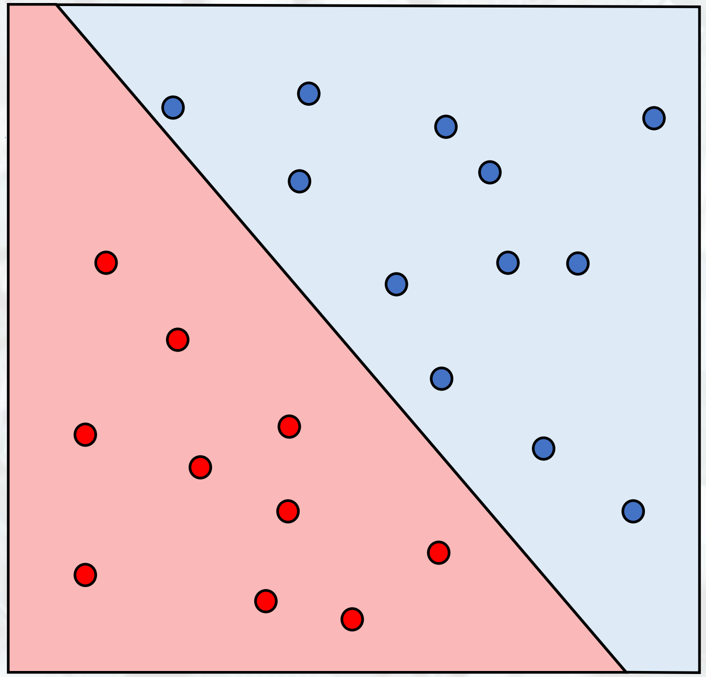
Linear separable space

A bidimensional space consisting of 2 classes of points is linearly separable if there exists at least one straight line separating the space into its 2 distinct classes.



Linear separable space

A bidimensional space consisting of 2 classes of points is linearly separable if there exists at least one straight line separating the space into its 2 distinct classes.



Linear separable space

A bidimensional space consisting of 2 classes of points is linearly separable if there exists at least one straight line separating the space into its 2 distinct classes.

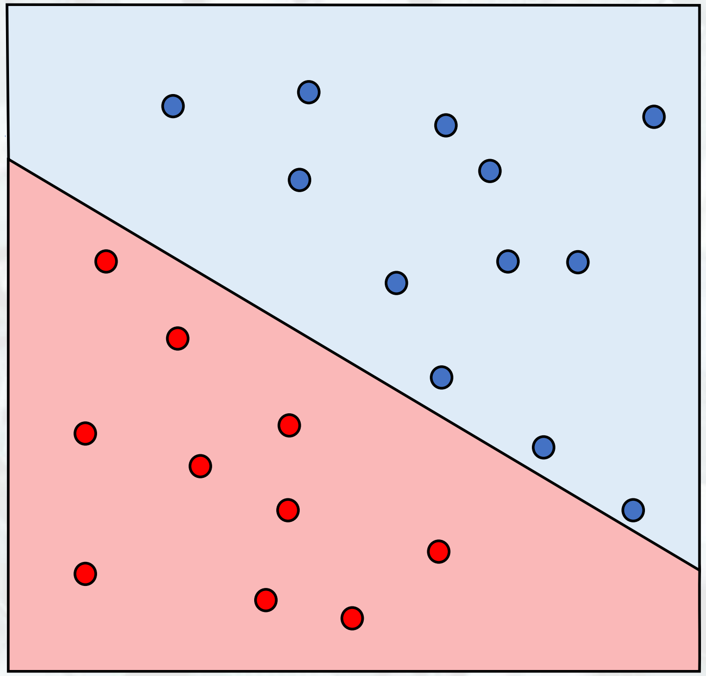
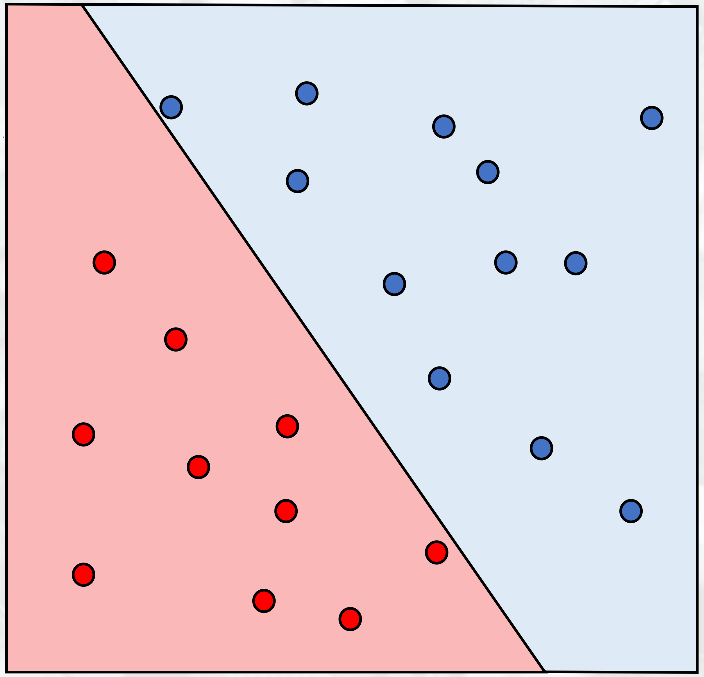


Image: Pinterest

Linear separable space

A bidimensional space consisting of 2 classes of points is linearly separable if there exists at least one straight line separating the space into its 2 distinct classes.



Non-linear separable space

What if the space is not linearly separable (e.g. XOR function)?

1. Combine several linear functions in cascade?
2. Apply non-linearities?
3. Combine several non-linear functions in cascade?

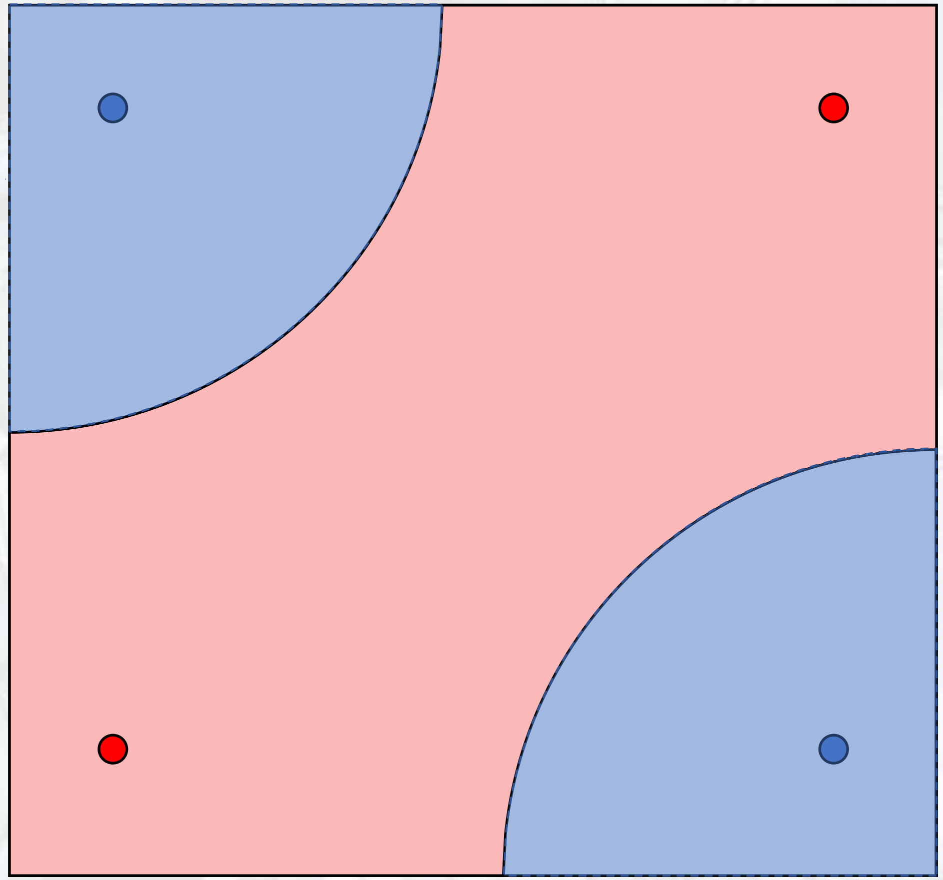
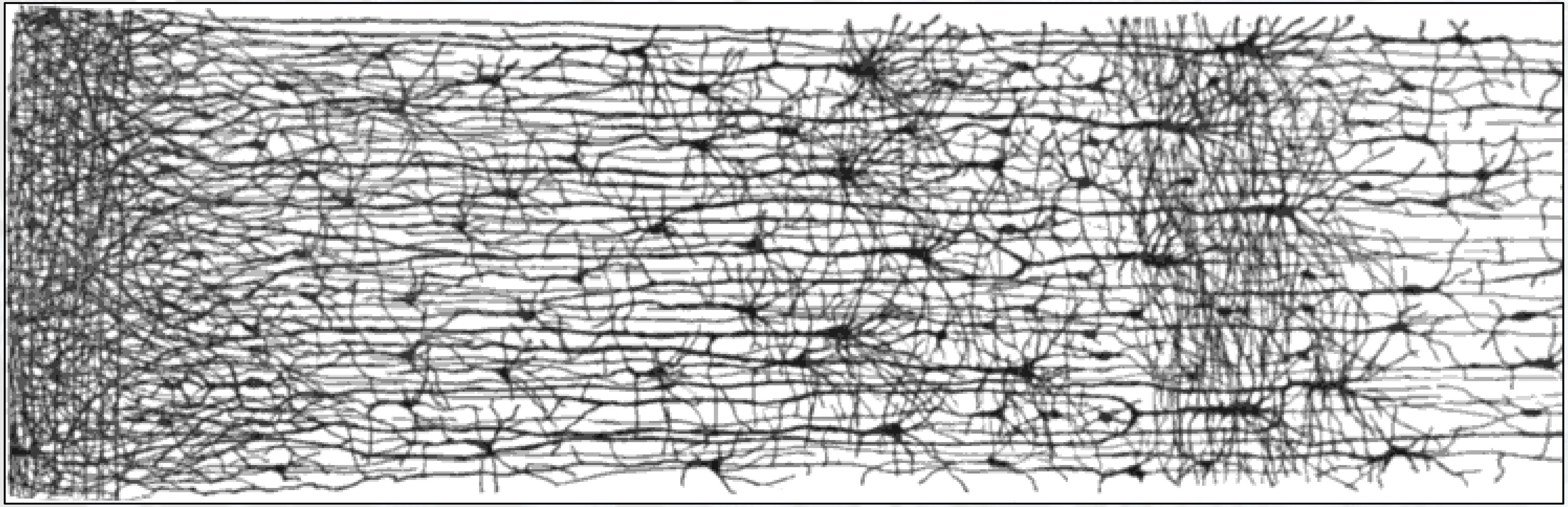


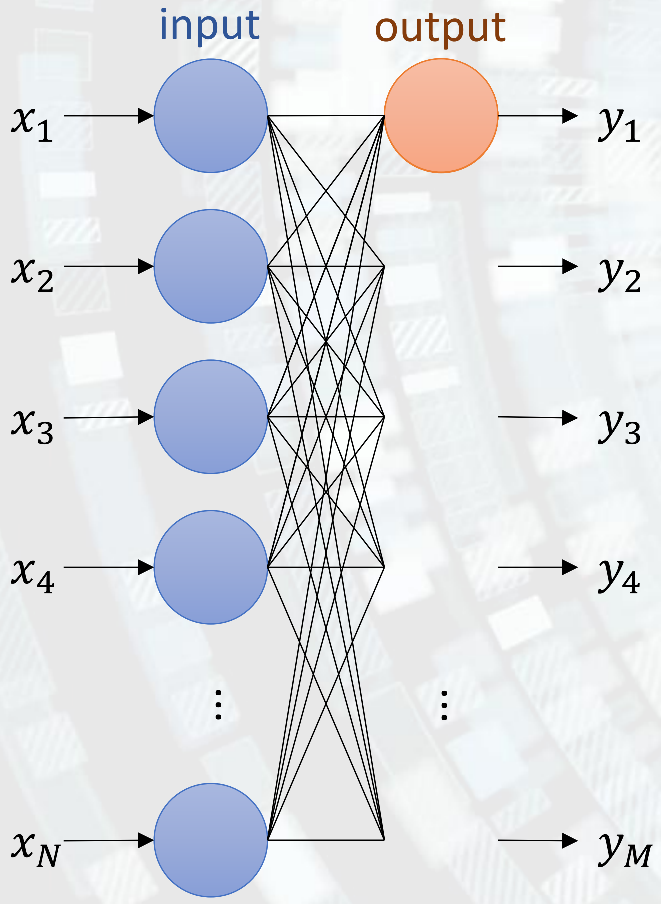
Image: Santiago Ramón y Cajal

The neuron network

The nervous system is formed of an entire network of neurons. Total number is estimated to be 86.1 ± 8.1 bn.



Single-layer neuron network

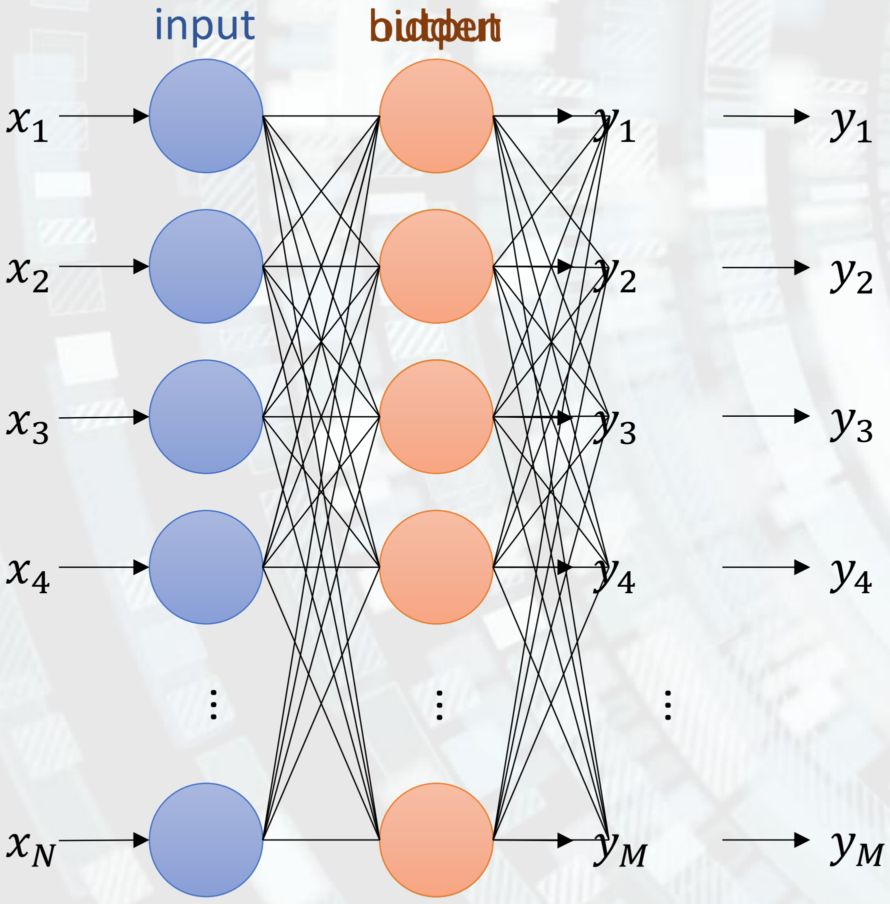


N can be (and usually is) different from M

Given a thresholding function, a single neuron can classify information in only 2 classes. Moreover, this is a "hard" decision.

A partial solution: a network formed of several neurons, placed in parallel, at the same level.
+ can be adapted to multi-class problems;
- still can't model non-linear separable spaces;
- every output is binary

Multi-layer neuron network



N can be (and usually is) different from M

+ a multi-layer neuron network can classify more complex data, e.g. XOR function.
- every output is binary

M2. The basic process of learning

The learning process

Machine learning = we say that a system „learns" from experience E regarding a set of tasks T and a performance measure P, if the performance in solving the tasks T, measured by P, grows with the experience E.

Goal: finding a function that associates an input dataset with their correct output, in the best way possible (smallest error; ideally – 0).

Consists of 2 steps:

1. Forward propagation;
2. Backpropagation.

Forward propagation

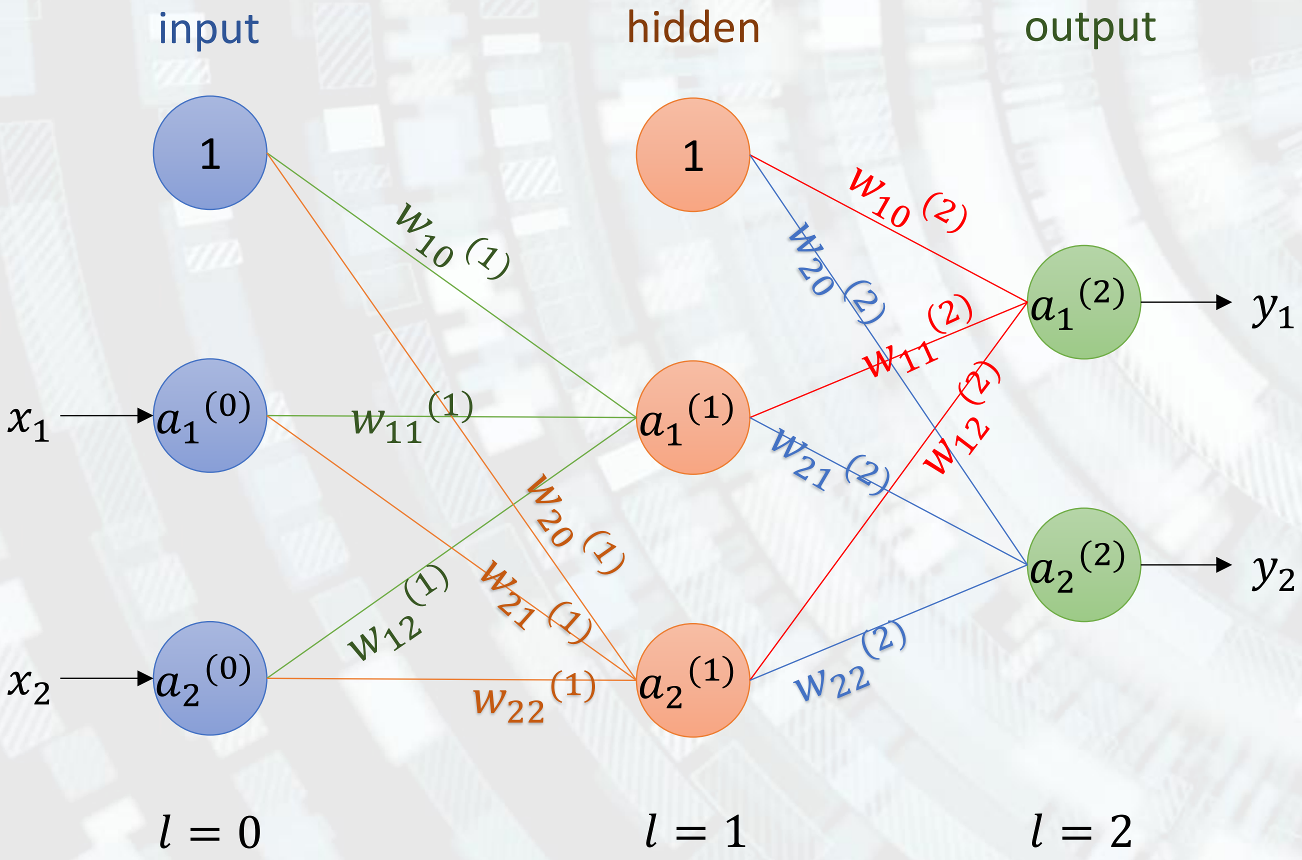
- **Goal**: obtaining the output $y = M(x)$;

- M = transfer function of the neural network, obtained by composition off all transfer functions associated to the network's layers.

- It involves passing the neural network from the input to the output.

- It breaks down to computing the activations for each neuron inside the network, from the lower layers towards the higher ones.

Forward propagation

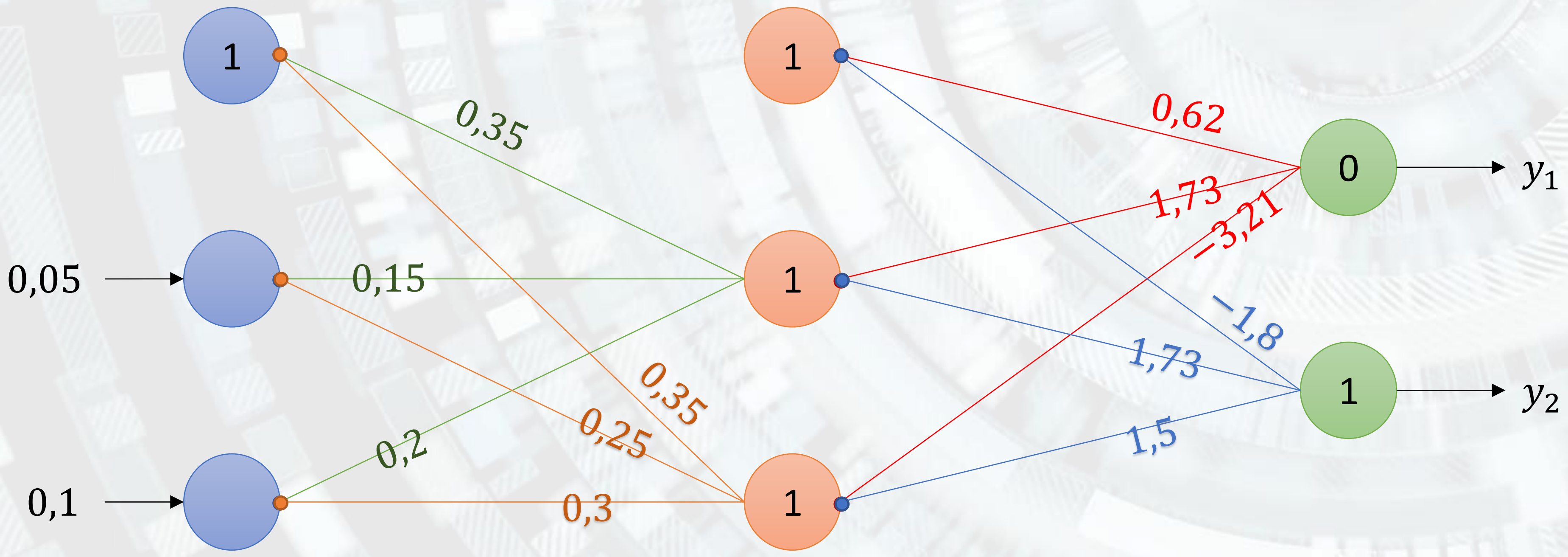


$$a_i^{(l)} = f\left(\sum_{j=0}^{n} w_{i,j}^{(l)} a_i^{(l-1)}\right)$$

$$a_0^{(l)} = 1$$

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

Forward propagation – numerical example



Backpropagation

- **Goal**: learning the internal representation of a neural network. It involves computing all weights (and biases) which bring the model's output tu a form as close as possible, ideally identical, to the desired/real output (groundtruth).

- How do we measure what it means to be „as close as possible"?
  - A: by using a cost function, e.g. mean square error.

- What rule do we use to adjust the weights?
  - A: gradient descent.

Image: AdobeStock

Gradient descent – intuition

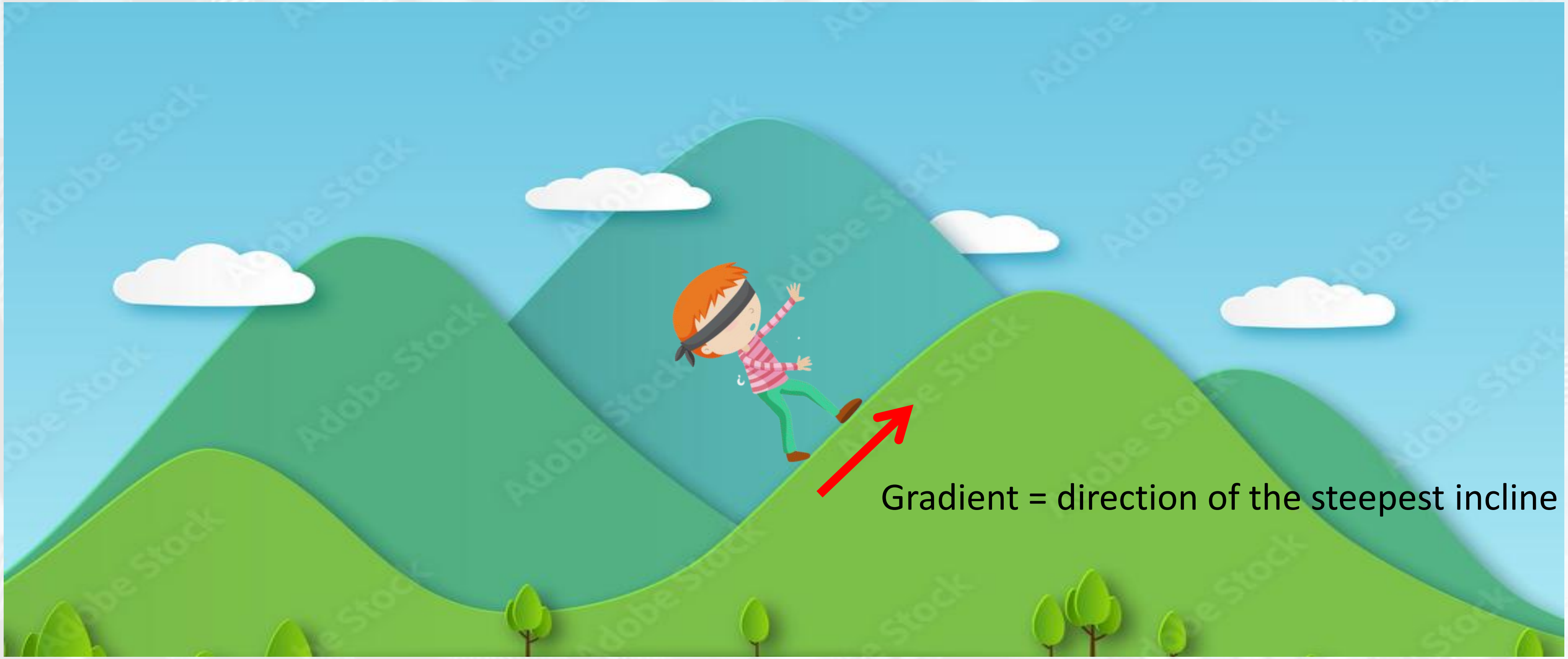


Image : AdobeStock

Gradient descent – intuition

Image : AdobeStock

Gradient descent – intuition

Image : AdobeStock

Gradient descent – intuition

Image : AdobeStock

Gradient descent – intuition

Gradient descent – intuition

Gradient descent

- Method to compute the gradient for a neural network's parameters.

$$y = f(\boldsymbol{w} \cdot \boldsymbol{x} + b) = f(\boldsymbol{w}^T \boldsymbol{x} + b)$$

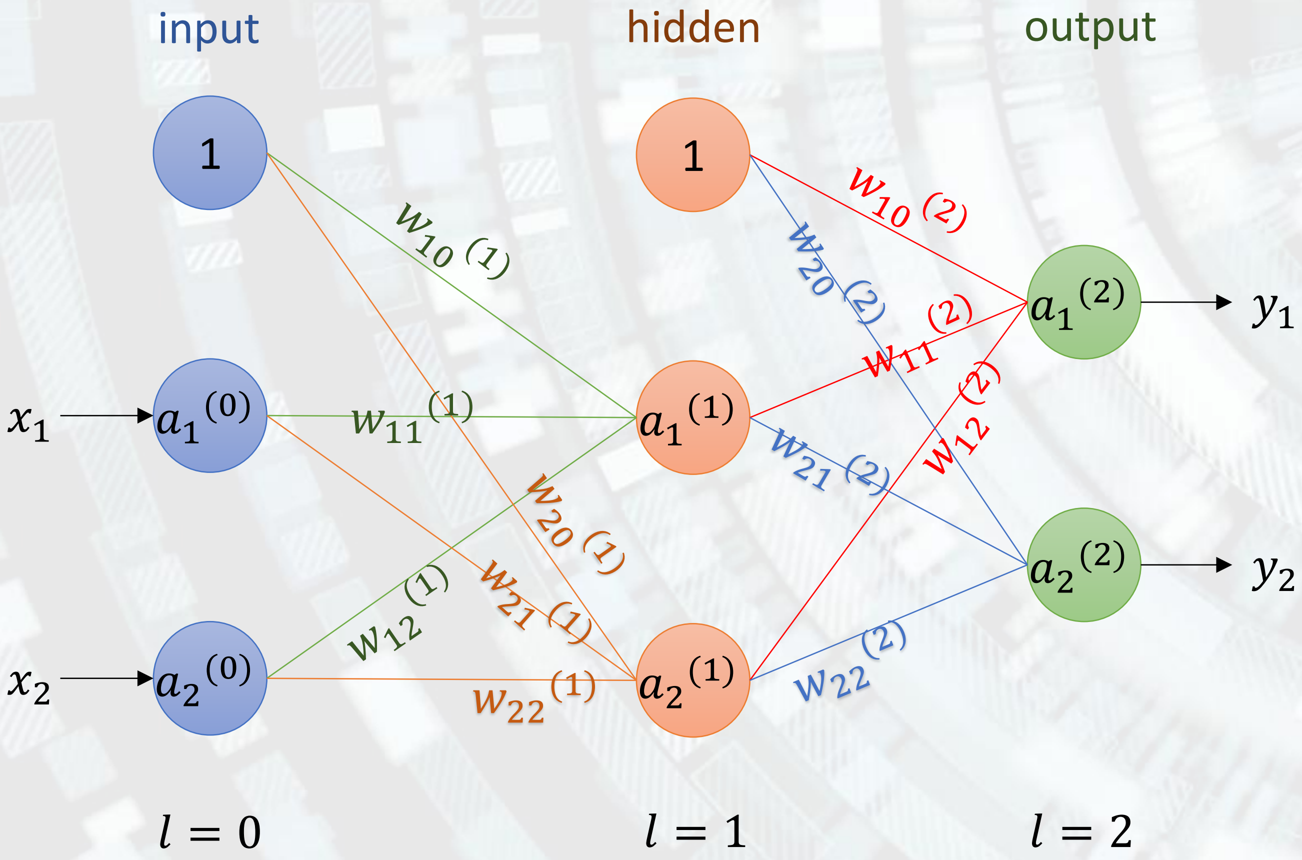
Funcție de pierdere (se calculează pe o singură pereche de eșantioane)

$$\mathcal{L}(y, \hat{y}) \overset{e.g.}{\Longrightarrow} \mathcal{L}(y, \hat{y}) = (y - \hat{y})^2$$

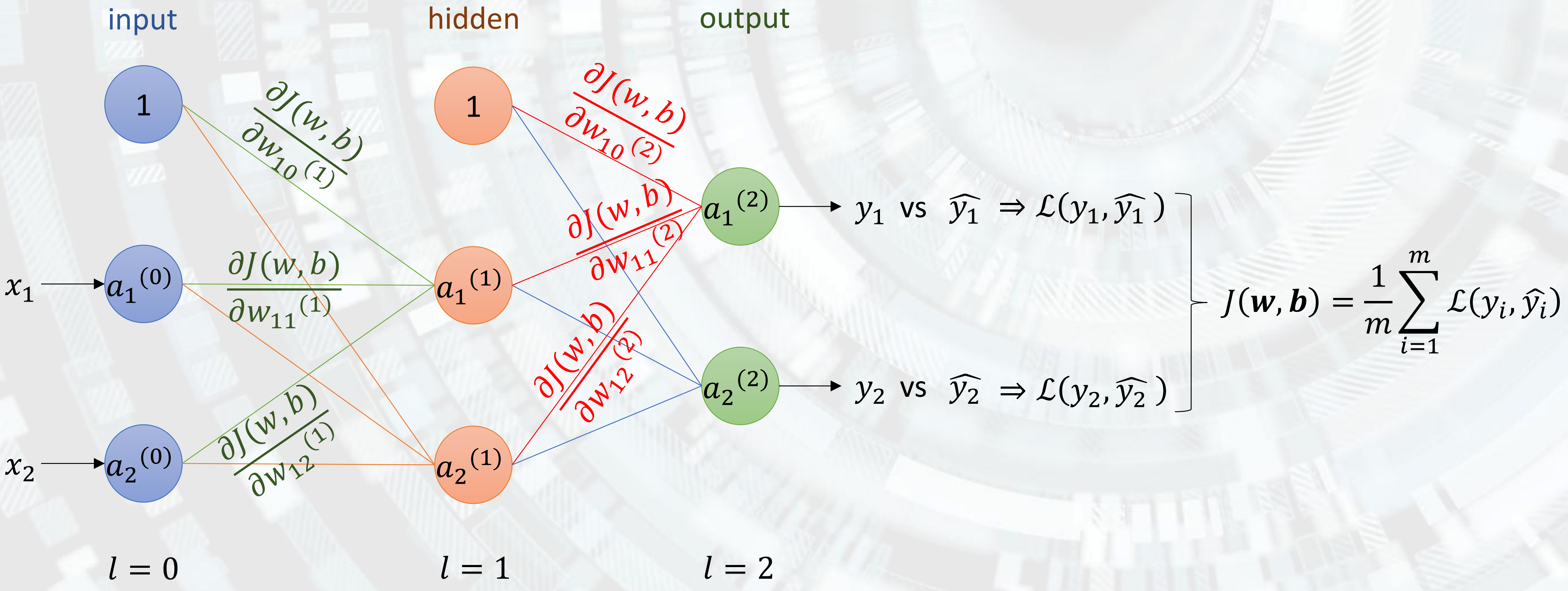
Funcție de cost (se calculează pe un set de perechi de eșantioane)

$$J(\boldsymbol{w}, \boldsymbol{b}) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(y_i, \widehat{y_i})$$

Gradient descent



Gradient descent



Gradient descent

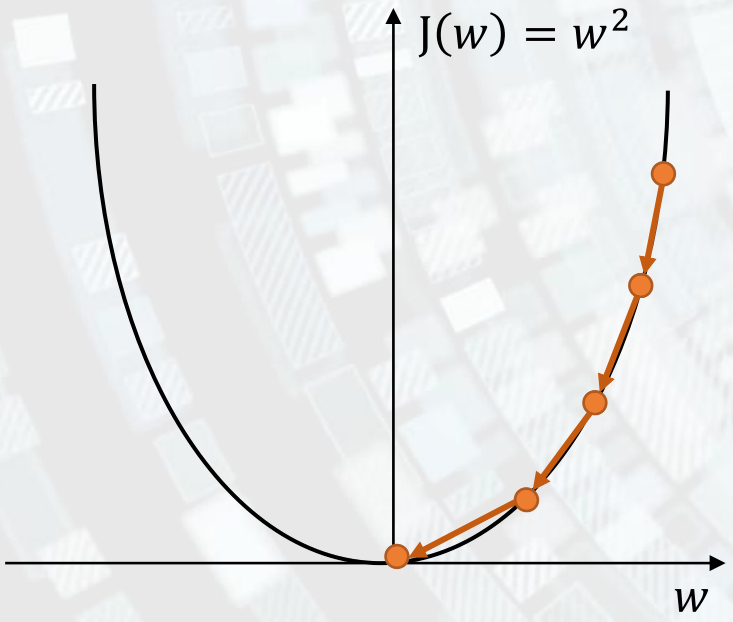
- Weights update follows this rule:

$$w^{(t+1)} = w^{(t)} - \alpha \frac{\partial J(w)}{\partial w^{(t)}}$$

Simplified writing: $$w \coloneqq w - \alpha \frac{\partial J(w)}{\partial w}$$

Gradient descent

- Graphical example: consider the simplified case in which the cost function depends on a single variable and we choose MSE.

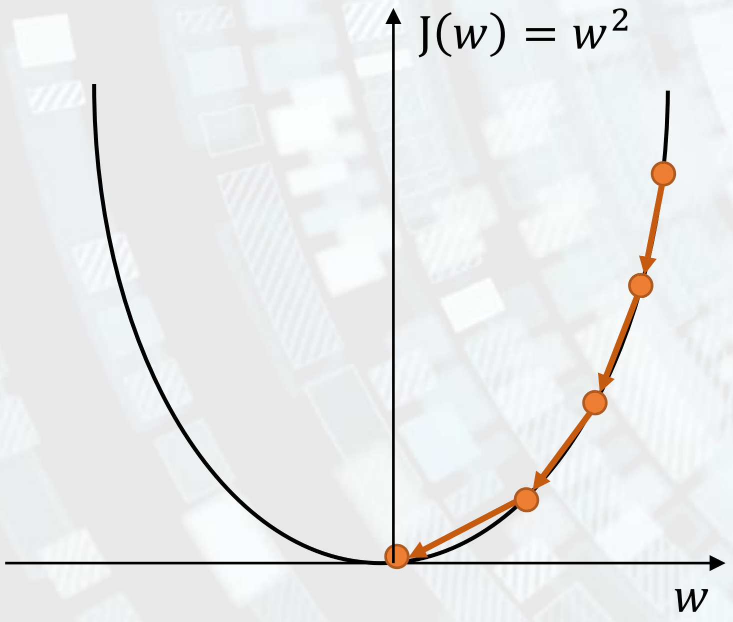


$$w \coloneqq w - \alpha \frac{\partial J(w)}{\partial w}$$

Ideally, we find w s.t. $J(w) = 0$, but, in practice, this almost never happens.

Gradient descent

➢ Graphical example: consider the simplified case in which the cost function depends on a single variable and we choose MSE.

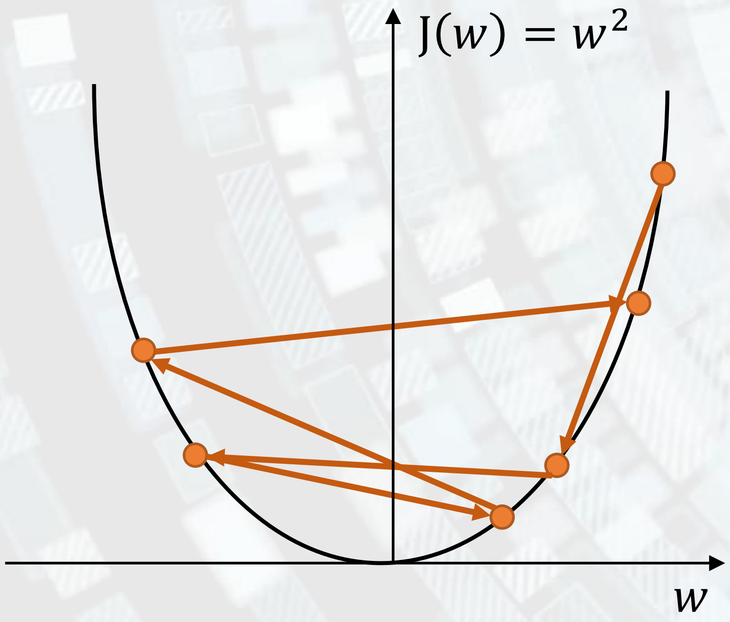


$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- $\frac{\partial J(w)}{\partial w}$ indicates the direction of the steepest increase
- α indicates the amplitude of the modification = learning rate

Gradient descent – learning rate impact

- Same example as before, but with a **high learning rate**.

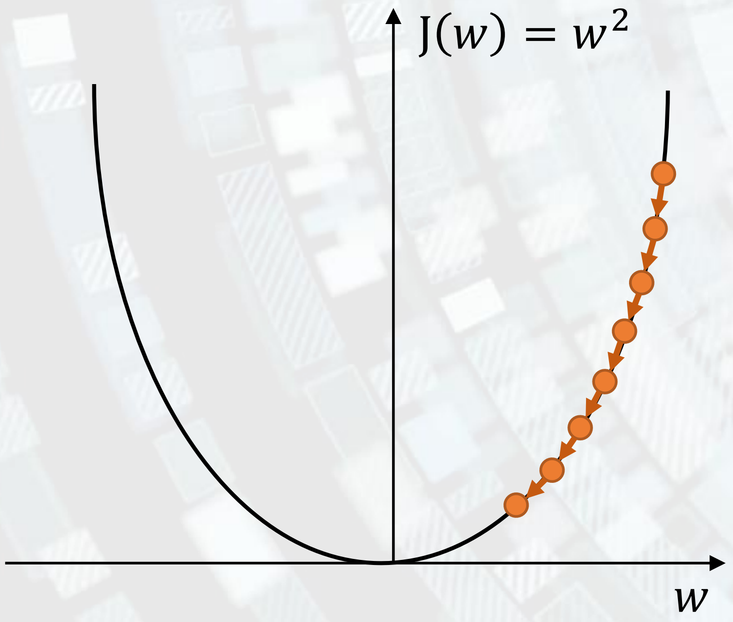


$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- The algorithm may miss the optimum, because each iteration brings a high displacement.
+ Due to the large steps taken, the distance to optimum is travelled faster => faster training.

Gradient descent – learning rate impact

➢Same example as before, but with a **low learning rate**.

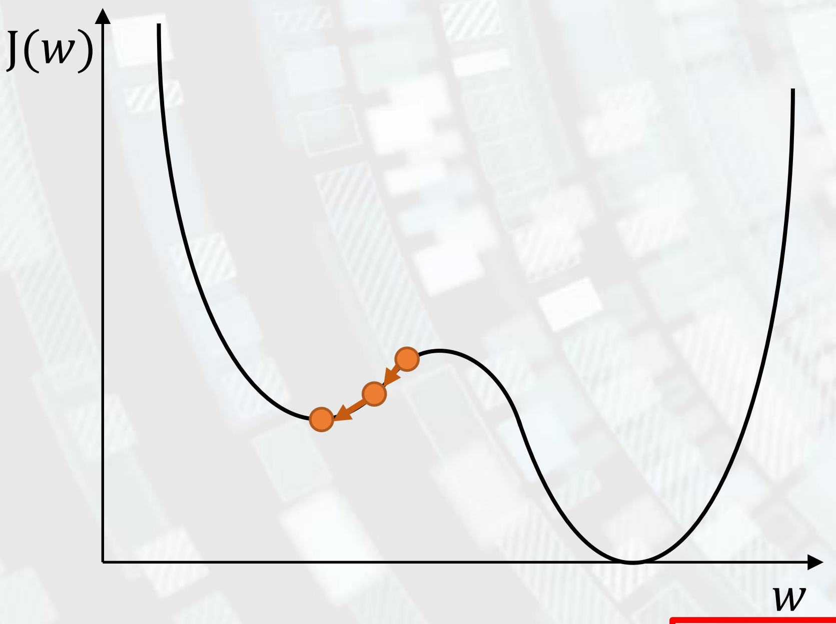


$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

- The algorithm performs small updates => slower training.
+ Low probability to miss the optimum dure to the small steps.
Why not always use small learning rate?

Gradient descent – learning rate impact

- Cost function that is not convex (local minimum is not necessarily global minimum), low learning rate.



$$w := w - \alpha \frac{\partial J(w)}{\partial w}$$

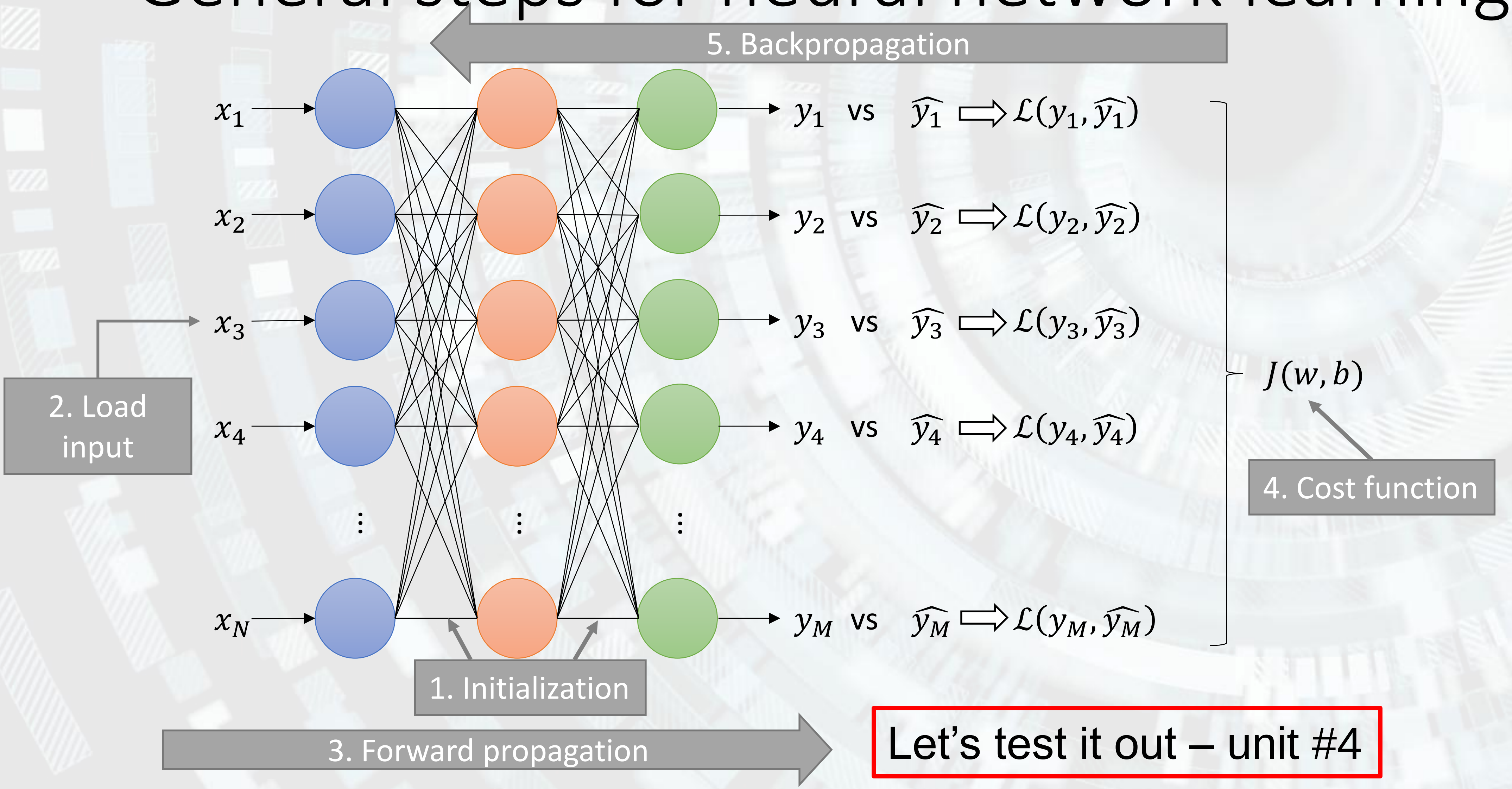
- There is a possibility that the algorithm gets stuck in a local minimum (saddle point)

Let's test it out – unit #3

General steps for neural network learning

1. Initialize network parameters
2. Load input data
3. Forward propagation
4. Compute cost function by comparing output data with groundtruth
5. Backpropagation + parameters update
6. Repeat steps 3-5 until convergence.

General steps for neural network learning



Let's test it out – unit #4