ISLab

Universidade do Minho
Escola de Engenharia
Departamento de Informática

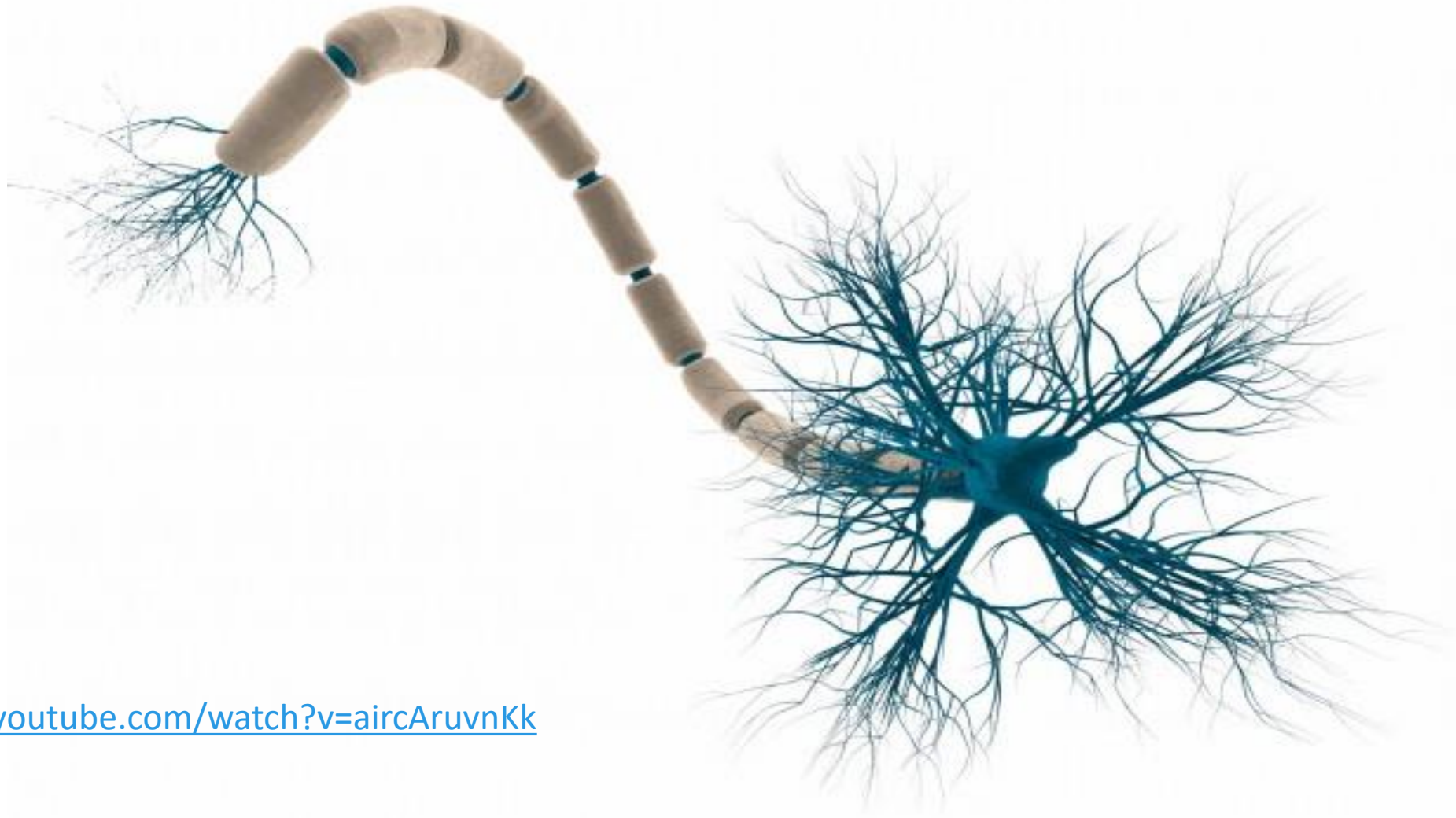
Dados e Aprendizagem Automática

Neural Networks

DAA @ MEI/1º ano – 1º Semestre

ISLab
Synthetic Intelligence Lab

Inspiration

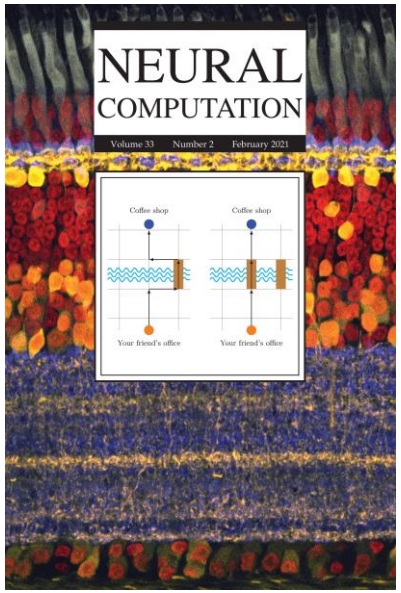


https://www.youtube.com/watch?v=aircAruvnKk

Neural Computation

Neural Computation

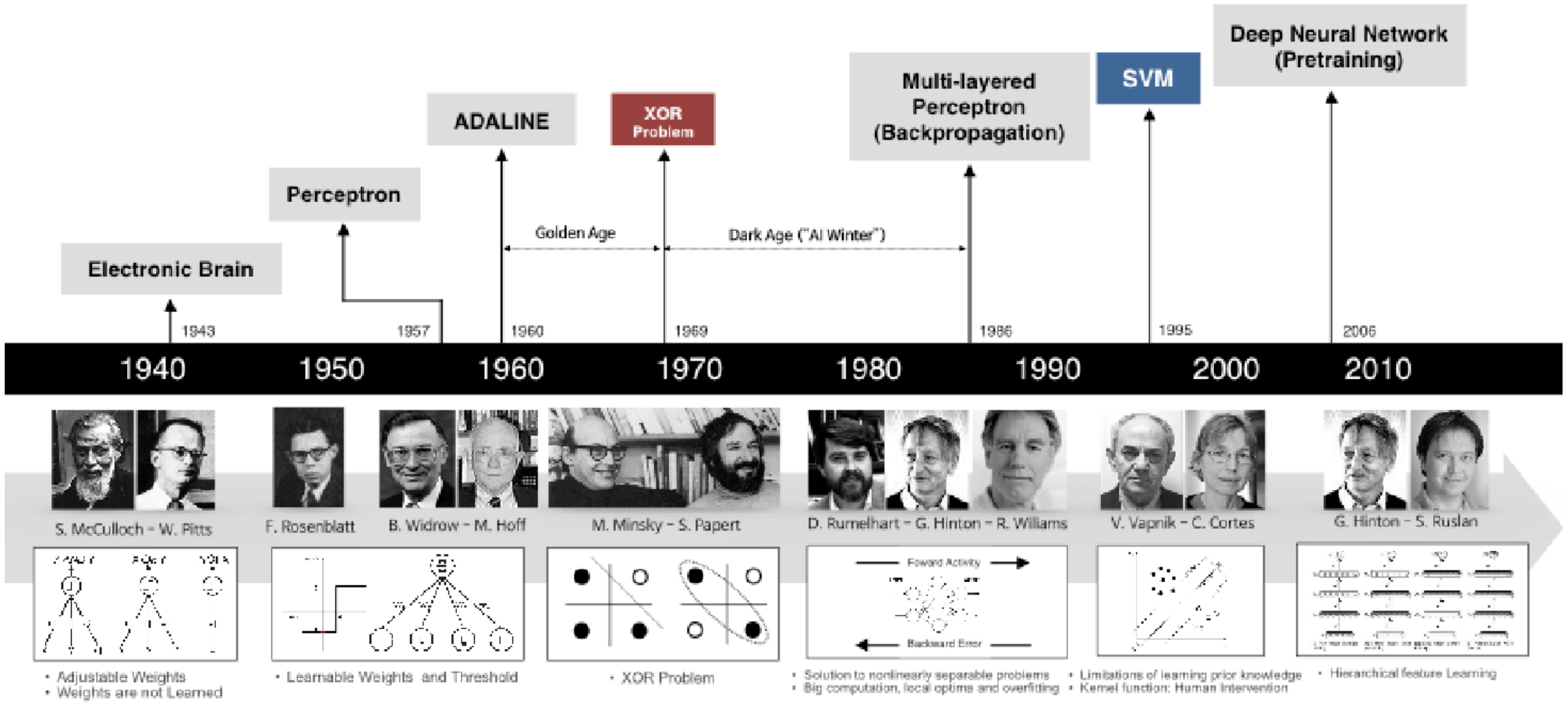
Neural computation is the information processing performed by networks of neurons.



Source: Neural Computation
https://www.mitpressjournals.org/loi/neco

ISLab
Synthetic Intelligence Lab

Timeline



◦https://github.com/qingkaikong/20161202_ANN_basics

ISLab
Synthetic Intelligence Lab

Neural Networks

- An analogy to biological neural systems;
- Understand biological systems through computational shaping;
- Massive parallelism allows for computational efficiency;
- Distributed (neural) representations vs local representation;
- Intelligent behavior - an emergent property of a large number of simple units;
- "**The human brain is a highly complex, non-linear, and parallel computer, responsible for information-processing abilities that humans possess**".

Image source: https://www.roboticshell.com/Tutorial/NeuralNetwork

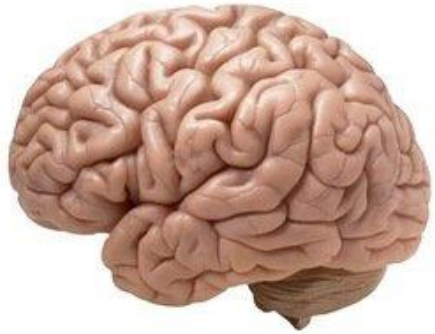
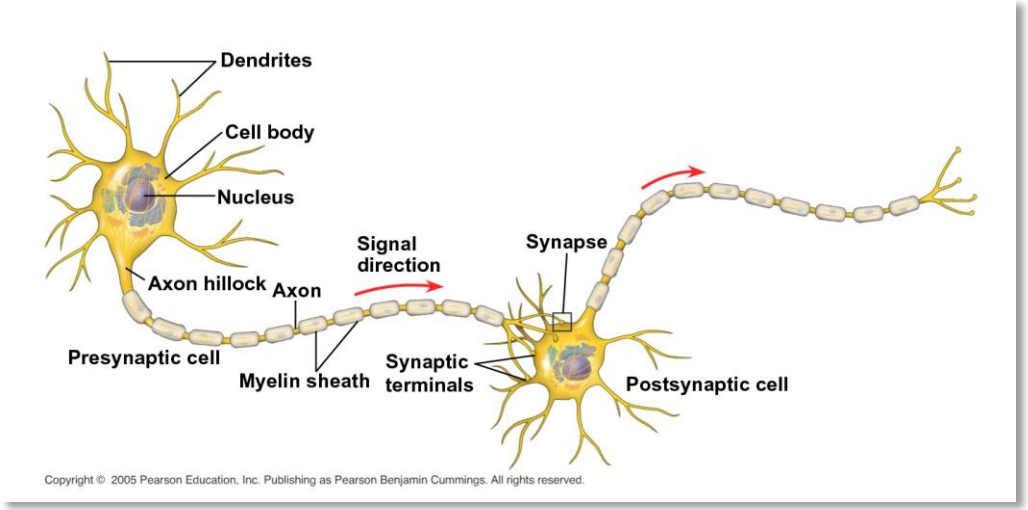


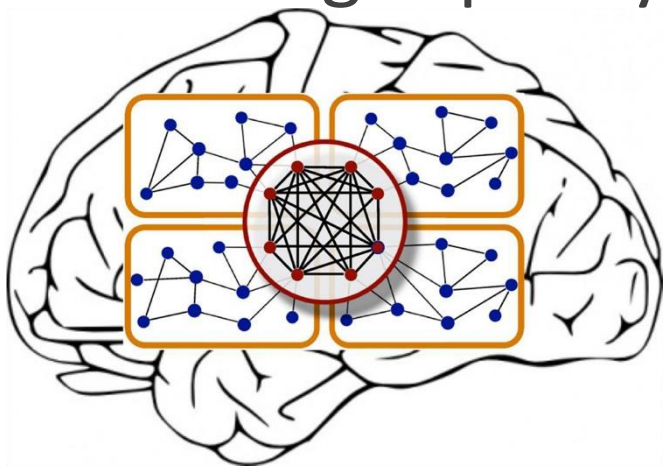
Image Source: https://www.freeimages.com/pt/photo/no-description-1301831

ISLab
Synthetic Intelligence Lab

Artificial Neural Network

- An **Artificial neural network** (ANN) is a connection-based computational system for problem solving.
- An ANN is designed based on a simplified model of human central nervous system.
- An ANN is defined by an interconnected structure of computational units, called neurons, with learning capacity.





ISLab
Synthetic Intelligence Lab

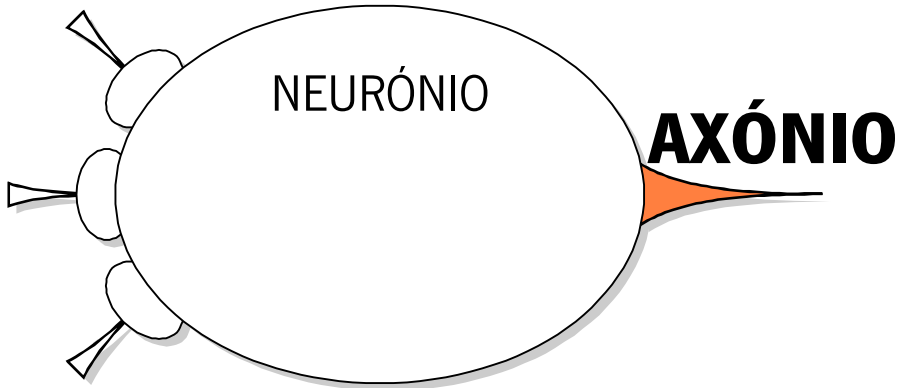
Neuron

- **Computational unit** of ANN composition.
- **Identified** by its **position** in the network.
- Characterized by the **state value**.

ISLab
Synthetic Intelligence Lab

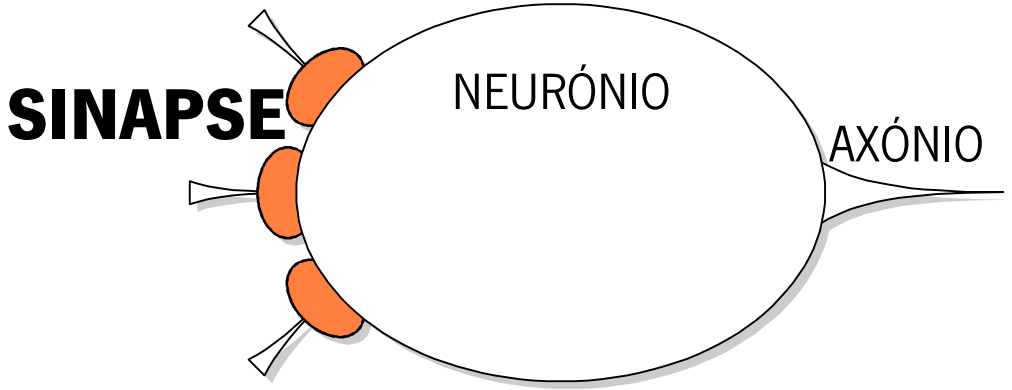
Axon

- **Communication pathway** between neurons.
- It can **connect any neuron,** including itself.
- Connections may **differ over time**.
- Information circulates in **one direction**.



Synapse

- **Connection point** between axons and neurons.
- The **synapse value** determines the **weight** (importance) of the signal to enter the neuron: excitatory, inhibitory or null.
- The **variation in time determines the learning** of ANN.



ISLab
Synthetic Intelligence Lab

Activation

- The activation value is represented by a **single value**.
- The activation value **changes with time**.
- The range of values differs with the model adopted (it is usually dependent on inputs and some memory effect).

ISLab
Synthetic Intelligence Lab

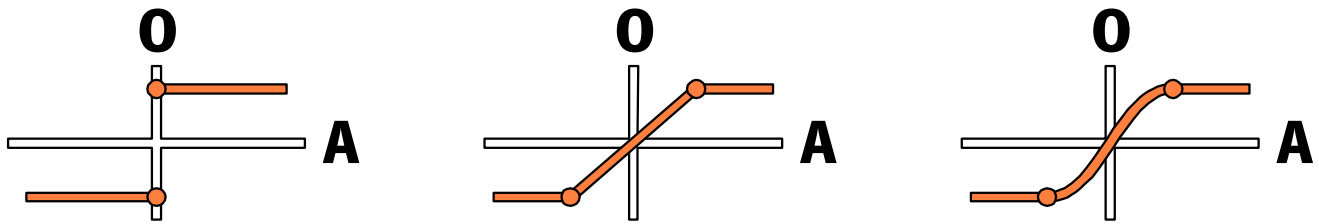
Transfer

- The transfer value of a neuron determines the **value** that is **placed at the exit** (transferred through the axon).
- It is calculated as a function of the activation value (possibly with some memory effect).

Tasks of a neuron

- Calculation of the output value (output = Oi), function of the activation value:

$O_i = f_T (A_i)$



Activation value (A_j).

Varies over time with its own value and that of other entries (w_i ; I):

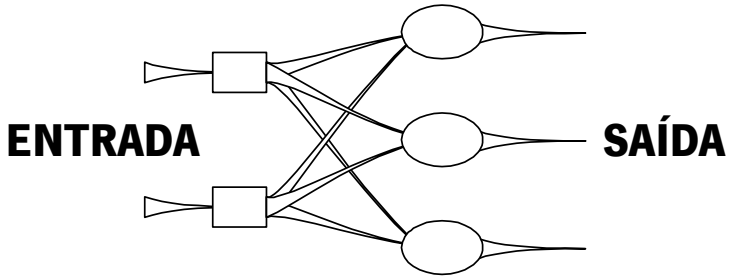
$$A_j = \mathcal{F}(A_{j-1}; I_j; \sum w_{i,j} \times O_i)$$

Learning: weight modification rules (w_i).

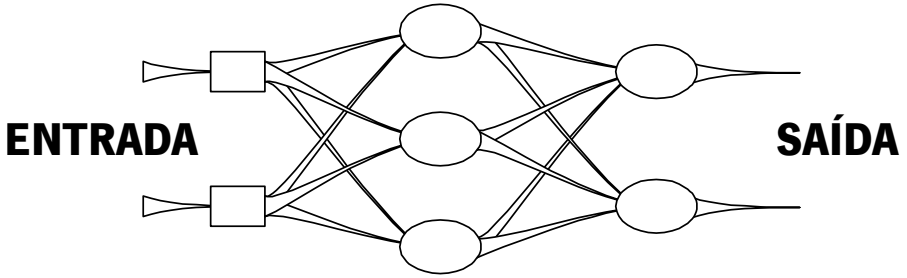
ISLab
Synthetic Intelligence Lab

Artificial Neural Network

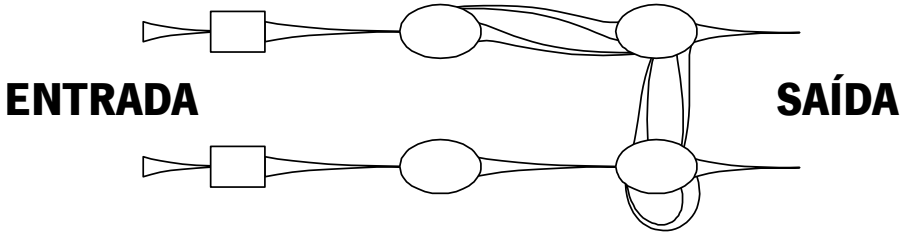
Feedforward, Single layer:



Feedforward, multi-layer:



Recurrent:

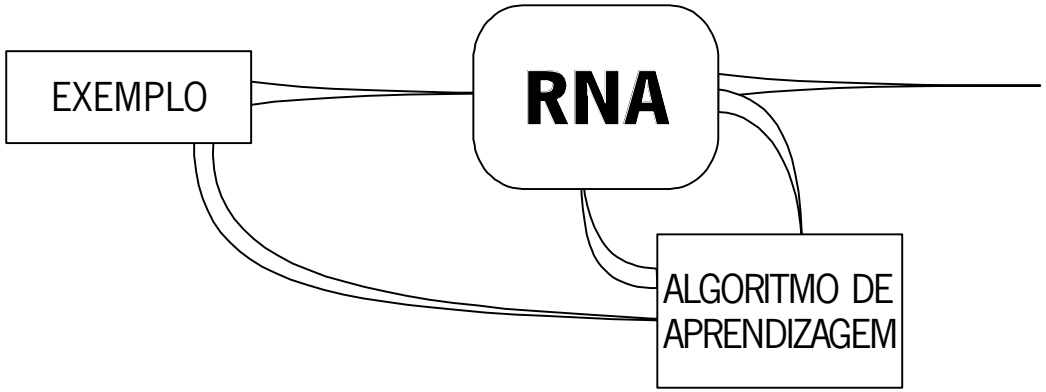


ISLab
Synthetic Intelligence Lab

Learning paradigms

Without supervision

(eg, when two adjacent neurons have variations in activation in the same direction, then the weight of the bond should be progressively increased.)



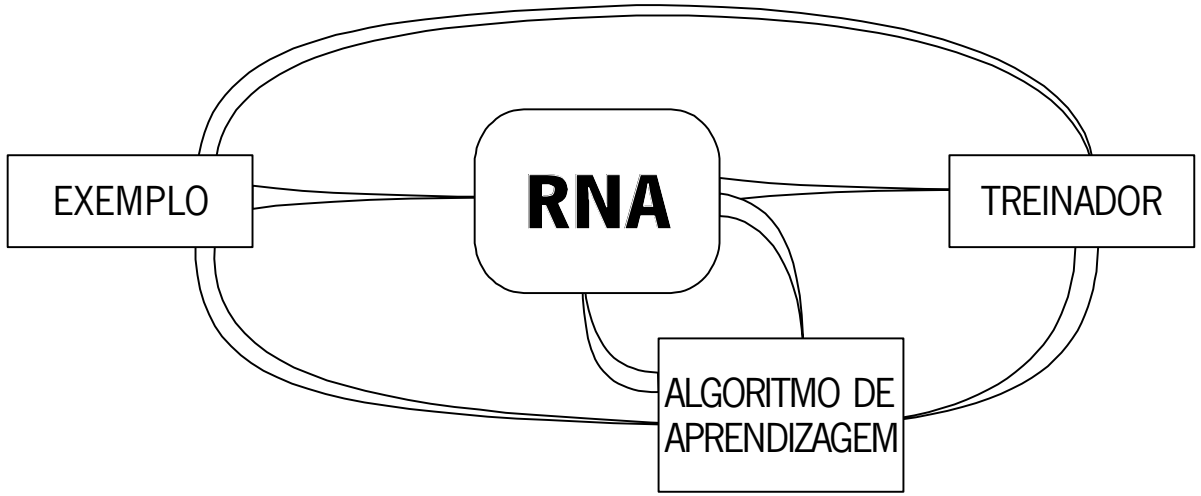
Learning paradigms

With supervision:

(eg, adjustments to the connection weights are made in order to minimize the error produced by the ANN results.)

Reinforcement:

the example contains only an indication of the correction of the result.



ISLab
Synthetic Intelligence Lab

Training rules (learning)

The training of an ANN corresponds to the application of learning rules, in order to vary the connection weights (synapses);

Examples of more common learning rules are:

- Hebbian;
- Competitive;
- Stochastic;
- Memory-based;
- Descending gradient.

ISLab
Synthetic Intelligence Lab

Training rules (learning)

Hebbian:

- If two nodes on each side of a connection are activated simultaneously, then the strength of that connection is progressively increased;
- If two nodes on each side of a connection are activated asynchronously, then the connection is progressively weakened or eliminated;
- Rule used in unsupervised learning

Competitive:

- The outputs of the nodes of the same layer compete with each other to become active, with only one node being activated at any given time. When a pattern is supplied to the network, one of the elements will respond better than the others, is therefore allowed to reinforce the weight of its connections.
- SOM (Self-Organinzing Maps) and the Kohonen

Stochastic:

- Connection weights adjusted in a probabilistic way. Simulated Annealing

ISLab
Synthetic Intelligence Lab

Training rules (learning)

Memory-based:

- Stores all (or almost all) past experiences. We have explicitly a large memory of pairs input/output.
- Radial-Basis Functions

Descending gradient:

- Supervised learning: it is intended to reduce the error between the desired value and the output value. The error as a control mechanism, with successive adjustments;
- Back-propagation algorithm implementation.

ISLab
Synthetic Intelligence Lab

Specification

Number of neurons:
- in the input layer;
- in the output layer;
- in the intermediate layers;

Levels (or layers) of the RNA;

Links between neurons;

Connection topology;

Scheme for assigning and updating weights;

Functions:
- transfer;
- activation;
- of learning;

Training Methods.

ISLab
Synthetic Intelligence Lab

Common Types of Applications

- Associative memory;
- Classification/Diagnosis;
- Pattern recognition;
- Regression;
- Control;
- Optimization;
- Data filtering / compression.

ISLab
Synthetic Intelligence Lab

XOR problem

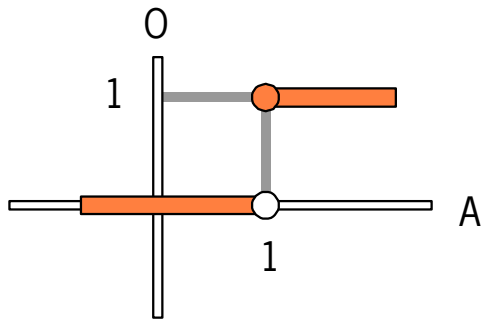
A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Activation function:

$\mathcal{F}_A = \sum$ imputs x weights

Transfer function:

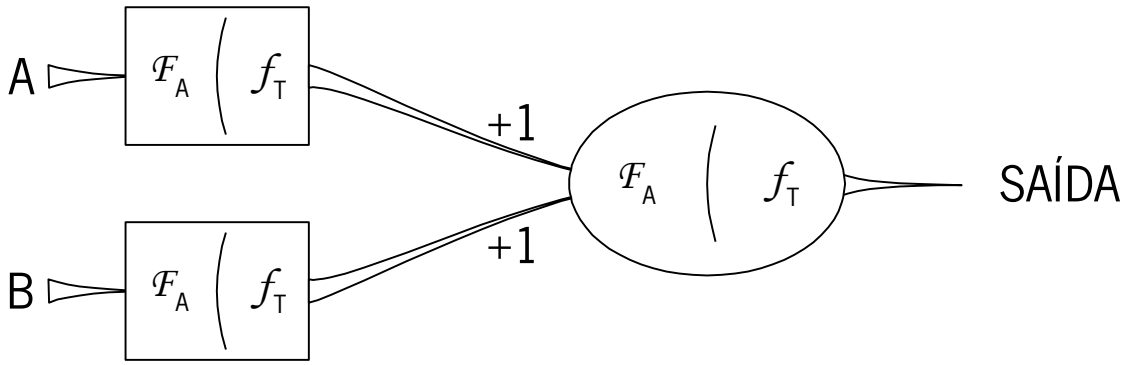
$f_T =$



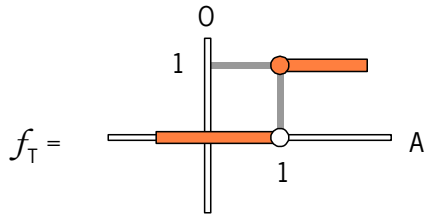
XOR Problem

Feedforward RNA, completely bound, with layers 2-1;

Assume the training result given by::



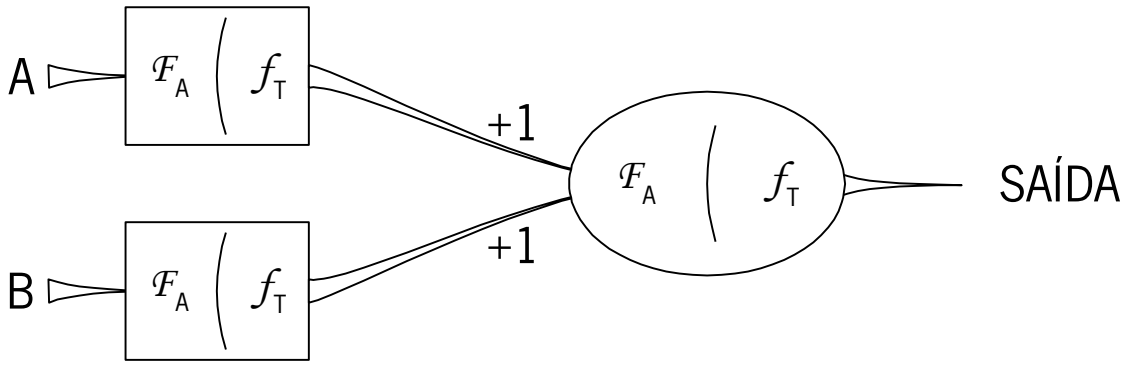
$\mathcal{F}_A$ = $\sum$ imputs x weights



XOR Problem

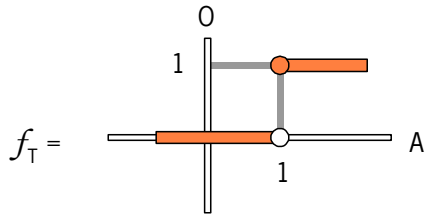
Feedforward RNA, completely bound, with layers 2-1;

Assume the training result given by::



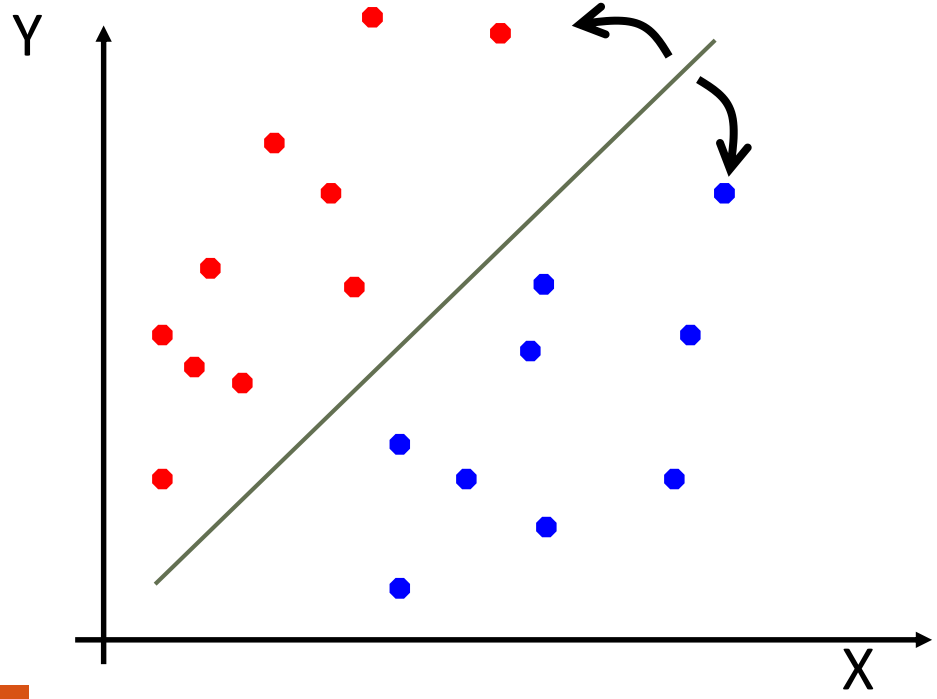
A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	1

$\mathcal{F}_A = \sum$ imputs x weights



Perceptron

Since perceptron uses linear threshold function, it is searching for a linear separator that discriminates the classes.



(As we know) Logistic regression is only able to find hyperplanes (basically straight lines) that separate the subspaces of each class.
linear discriminants

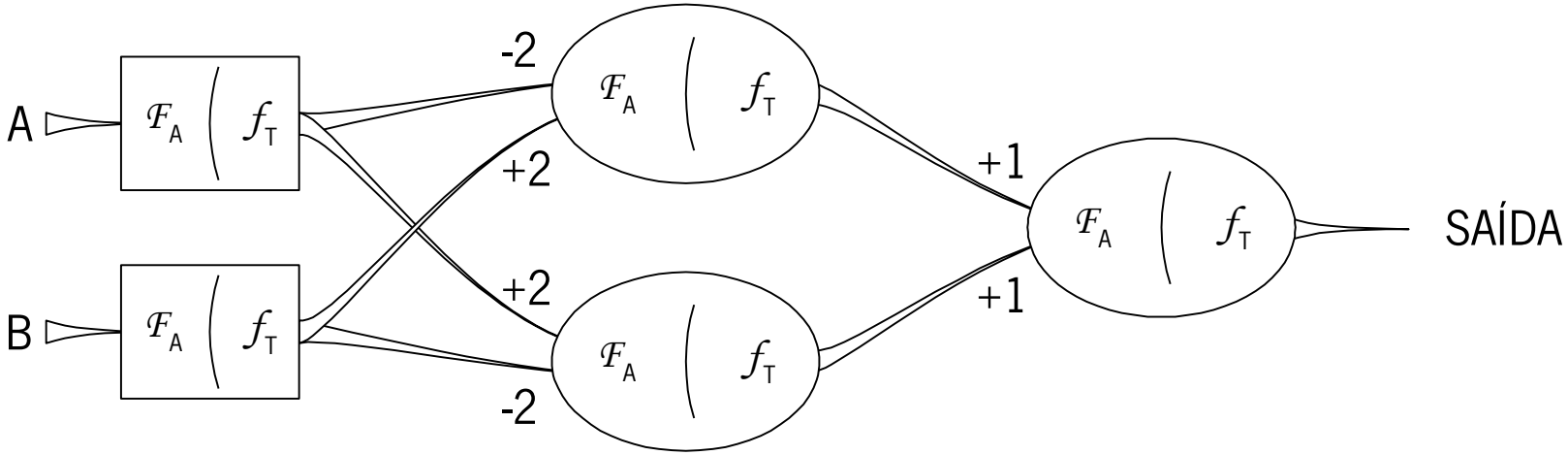
Historically the inefficiency of the Perceptron networks to solve this problem caused the “NN winter”

ISLab
Synthetic Intelligence Lab

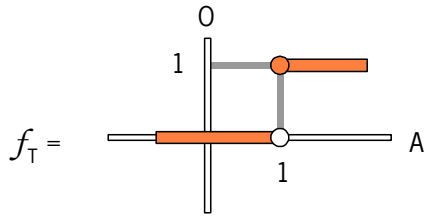
XOR Problem

Feedforward RNA, completely bound, with layers 2-2;

Assume the training result given by::



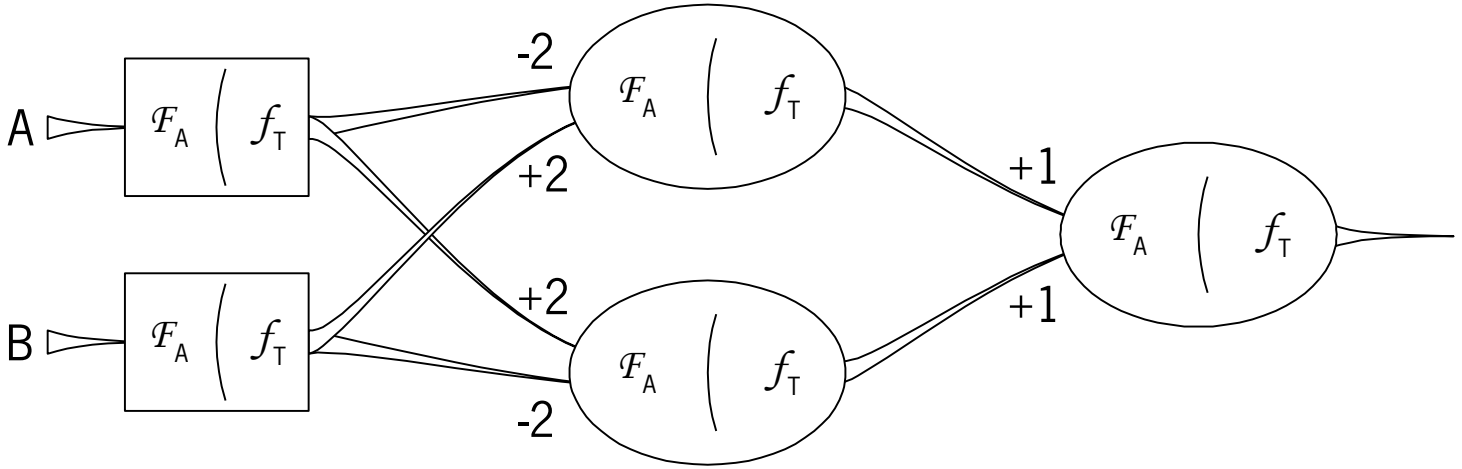
$\mathcal{F}_A = \sum$ imputs x weights



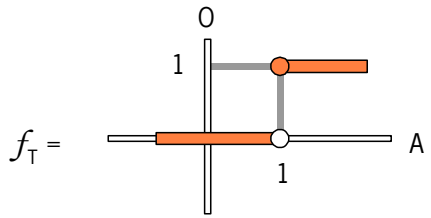
XOR Problem

Feedforward RNA, completely bound, with layers 2-2;

Assume the training result given by::



$\mathcal{F}_A = \sum$ imputs x weights



A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0

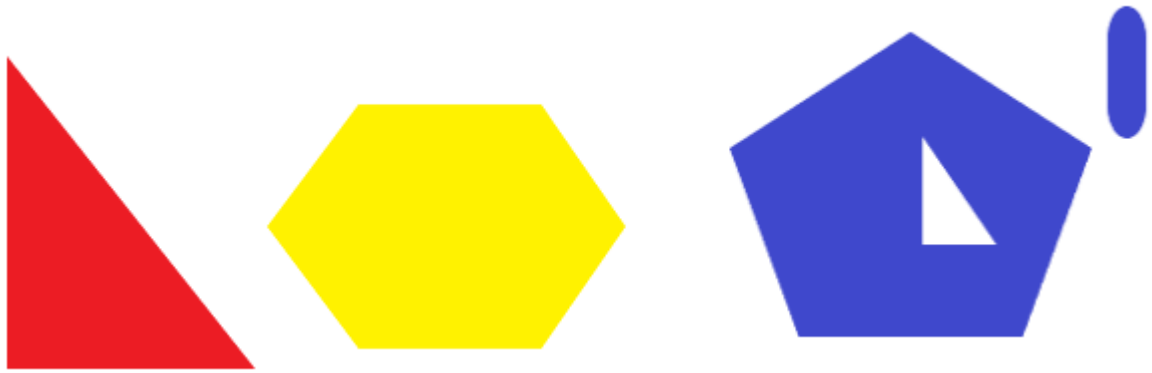
ANN: issues

The number of hidden layers, typically:

- one layer has the ability to approximate any linear decision area;
- two layers approximate any continuous decision area;
- three layers approximate any decision area.

In the hidden layers, the number of neurons:

- A short number might not be enough to model the decision area desired;
- A high value might lead to overfitting
- Values between half and double of the number of neurons in the input layer should test.



ISLab
Synthetic Intelligence Lab

RNA's training

Consider the following Artificial Neuronal Network ...

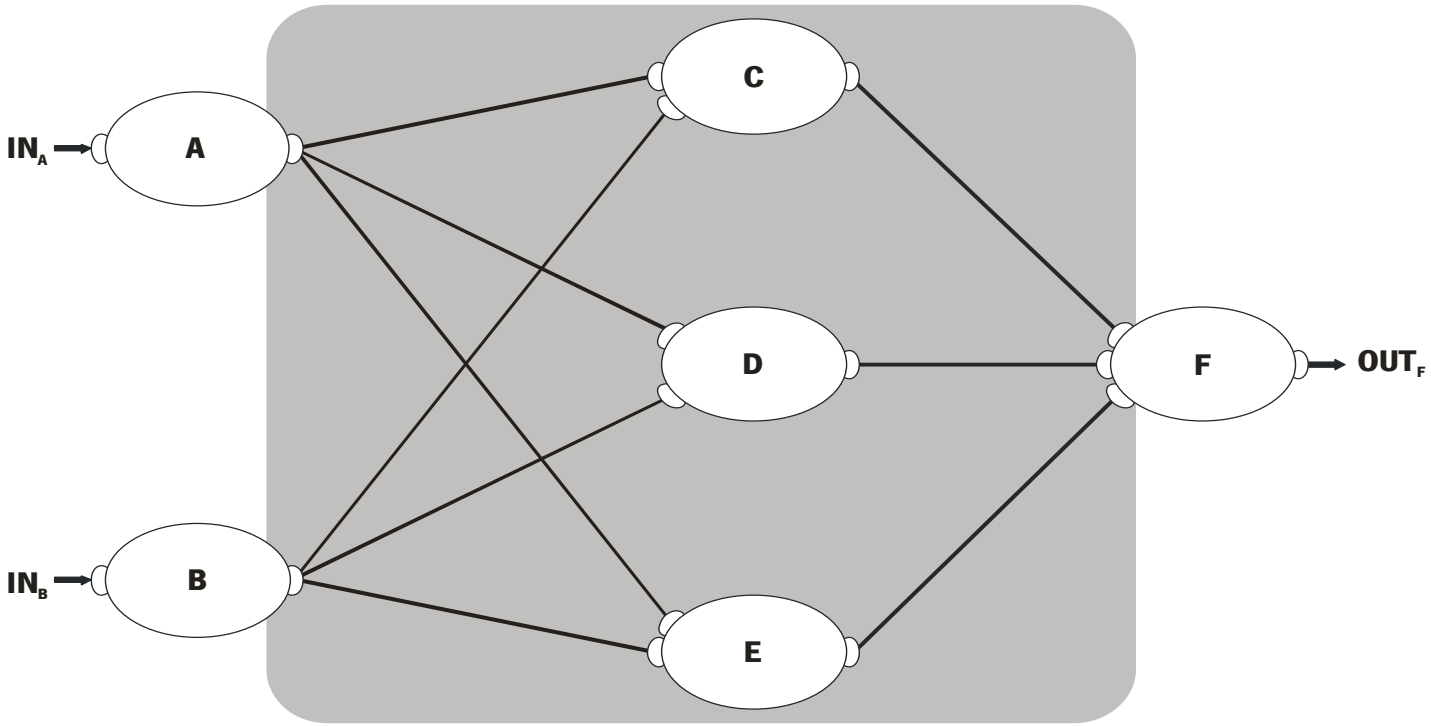
RNA's training

... structure: 2 neurons at the input and 1 at the output ...

ISLab
Synthetic Intelligence Lab

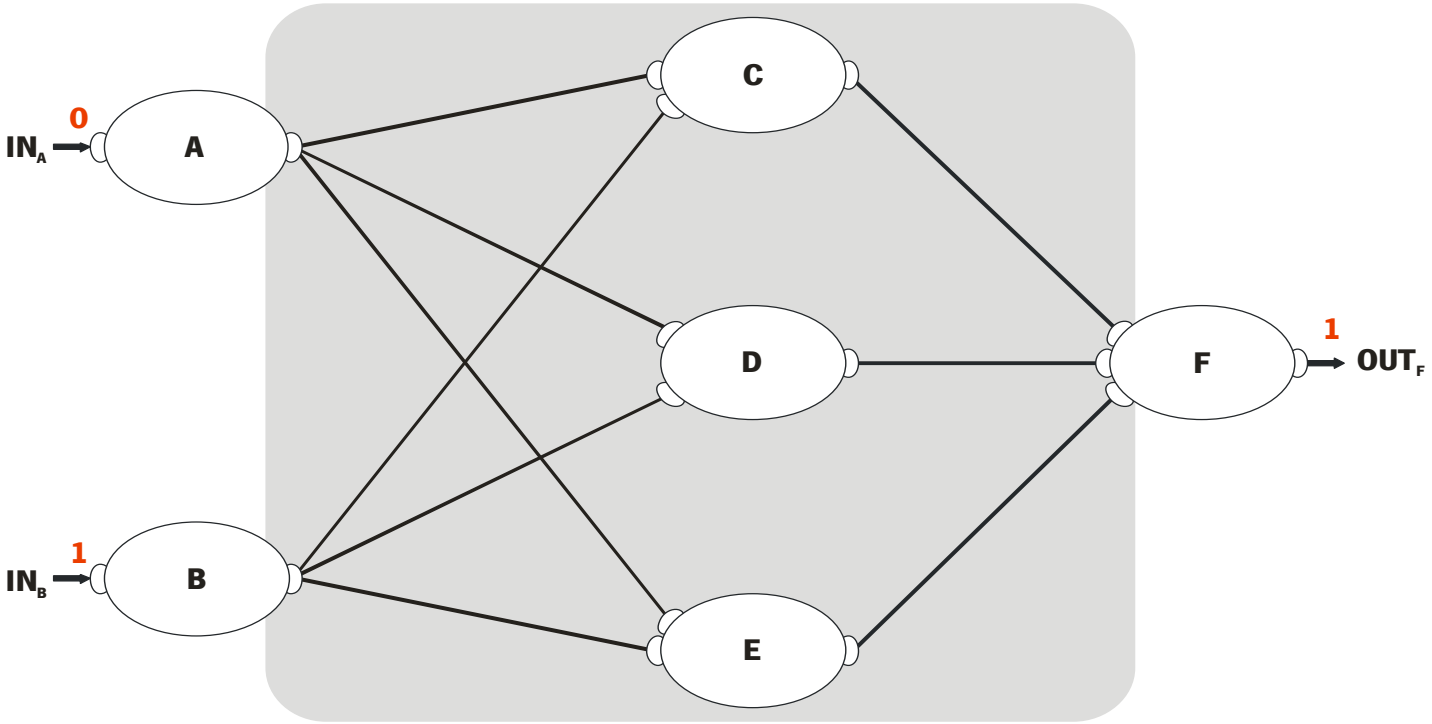
RNA's training

... feedforward, completely connected



RNA's training

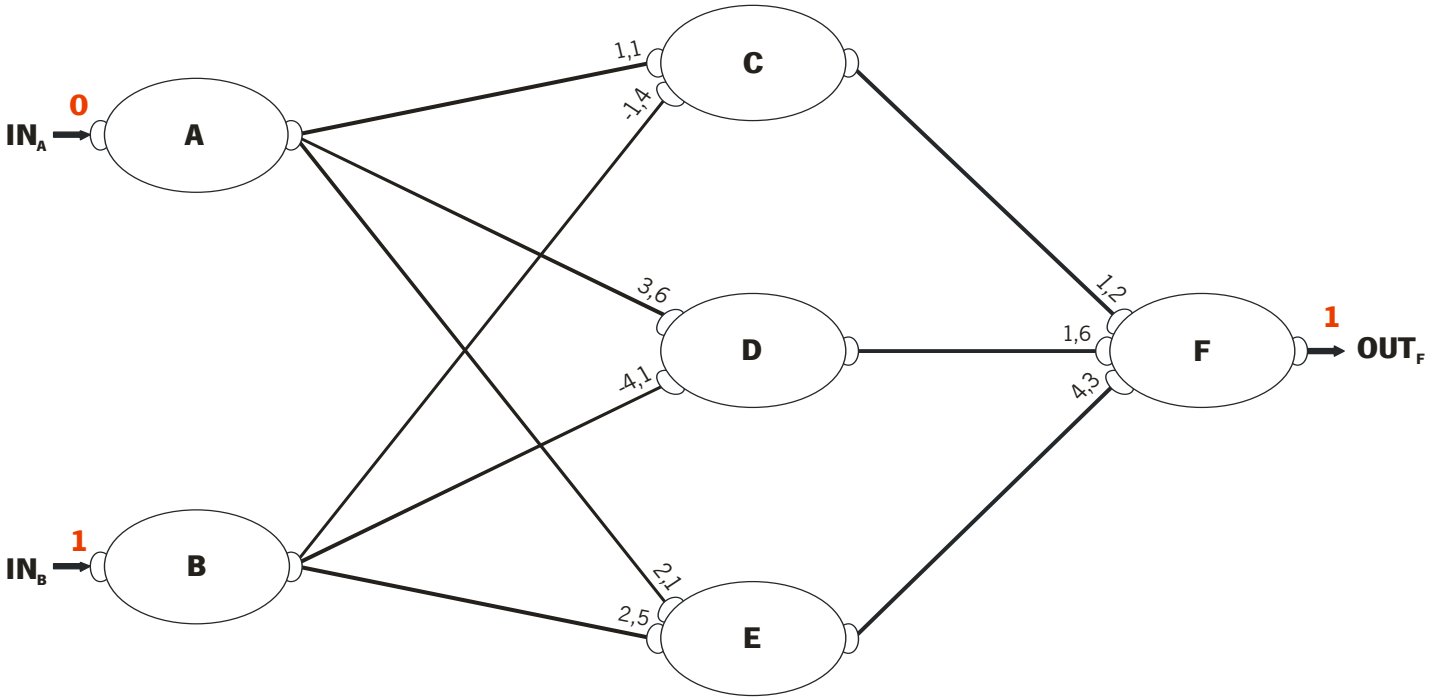
The training examples contain the desired results.



ISLab
Synthetic Intelligence Lab

RNA's training

Random assignment of weights to synapses.

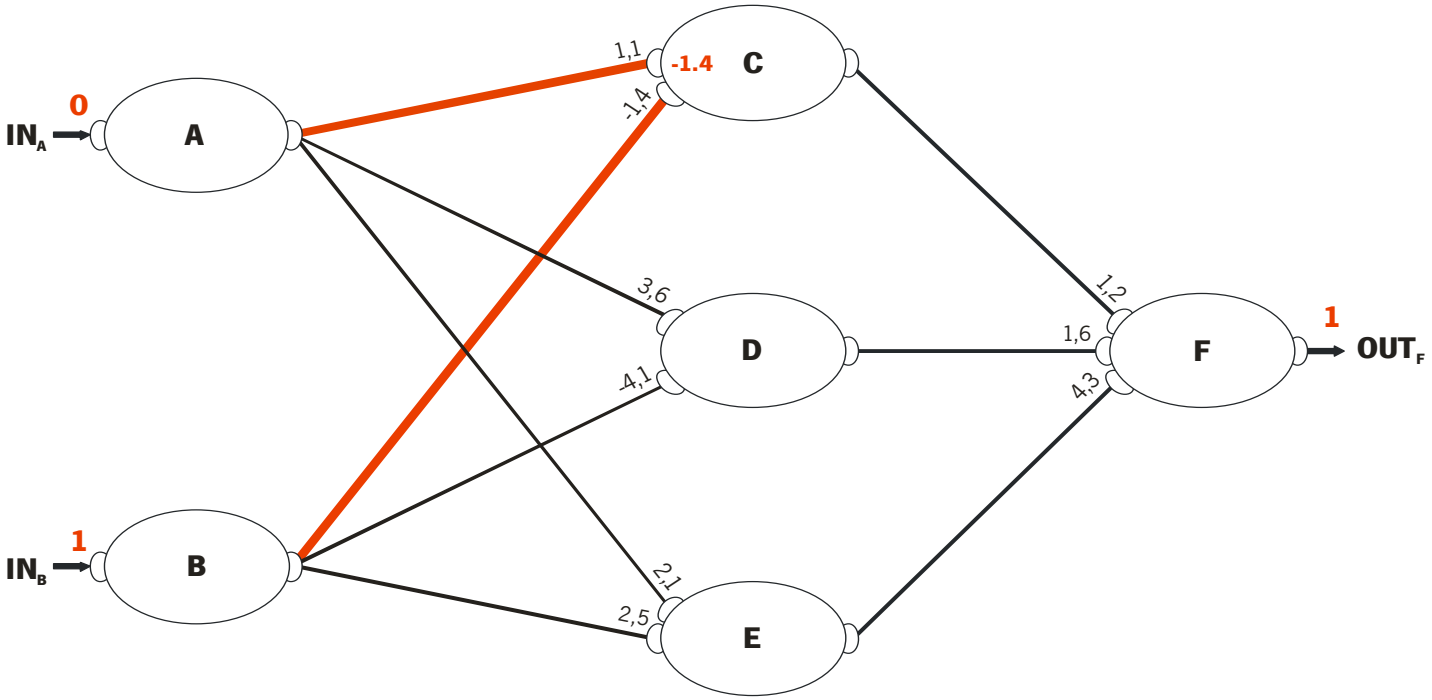


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

Computation of activation value ...

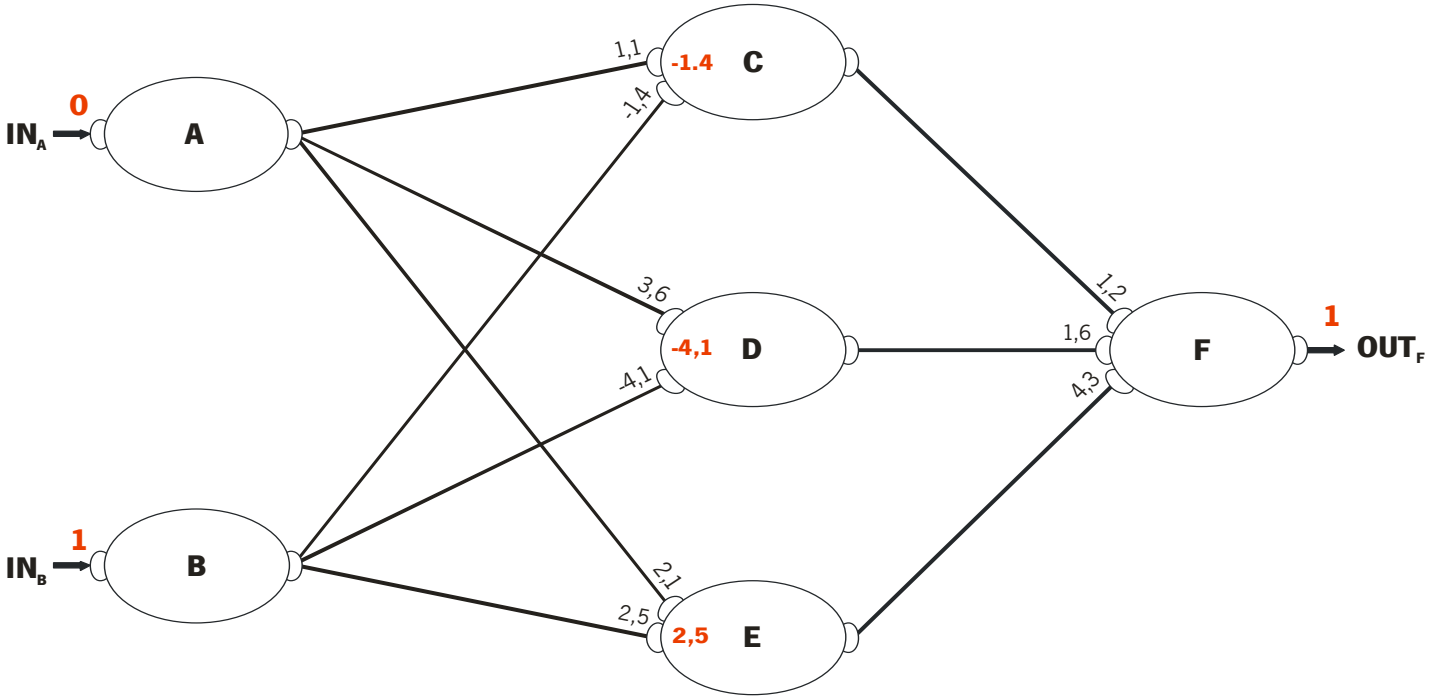


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

... for all neurons in the middle layer.

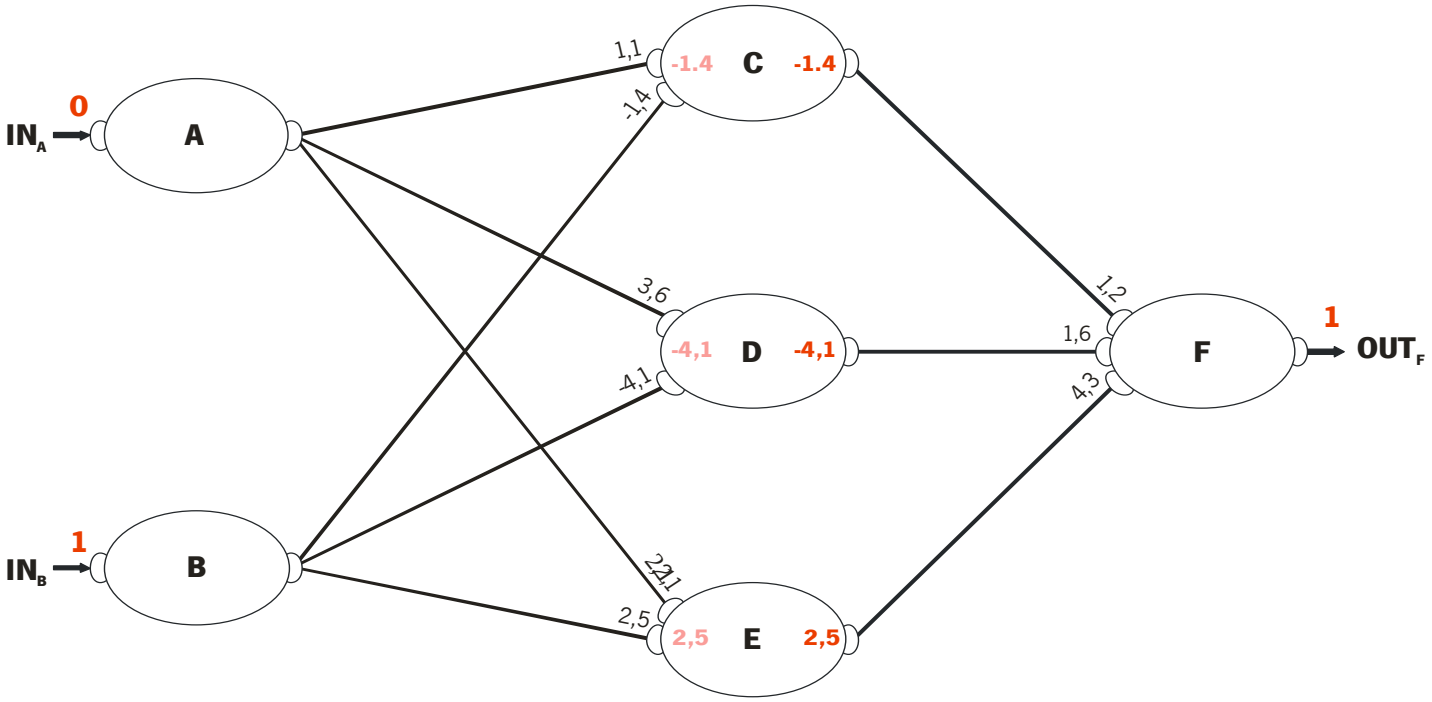


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

Estimate of the transfer amount.

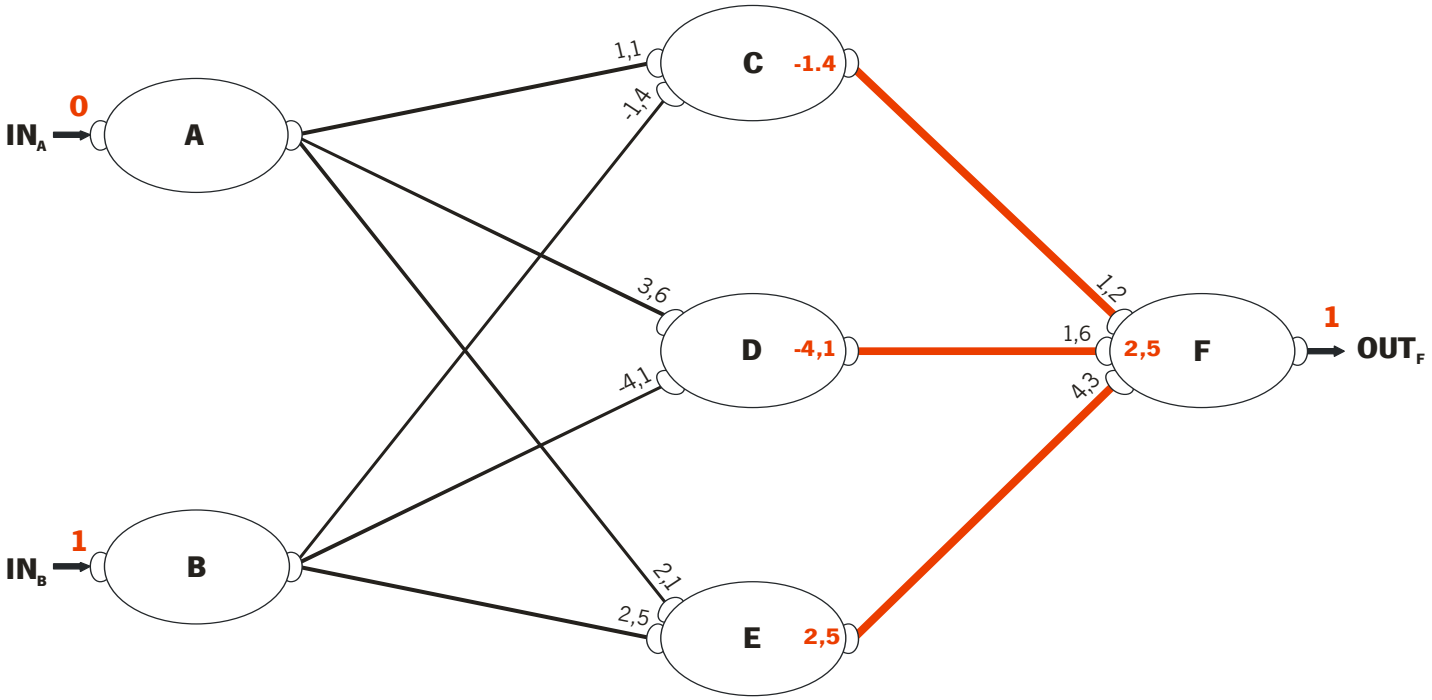


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

Activation value in the output layer ...

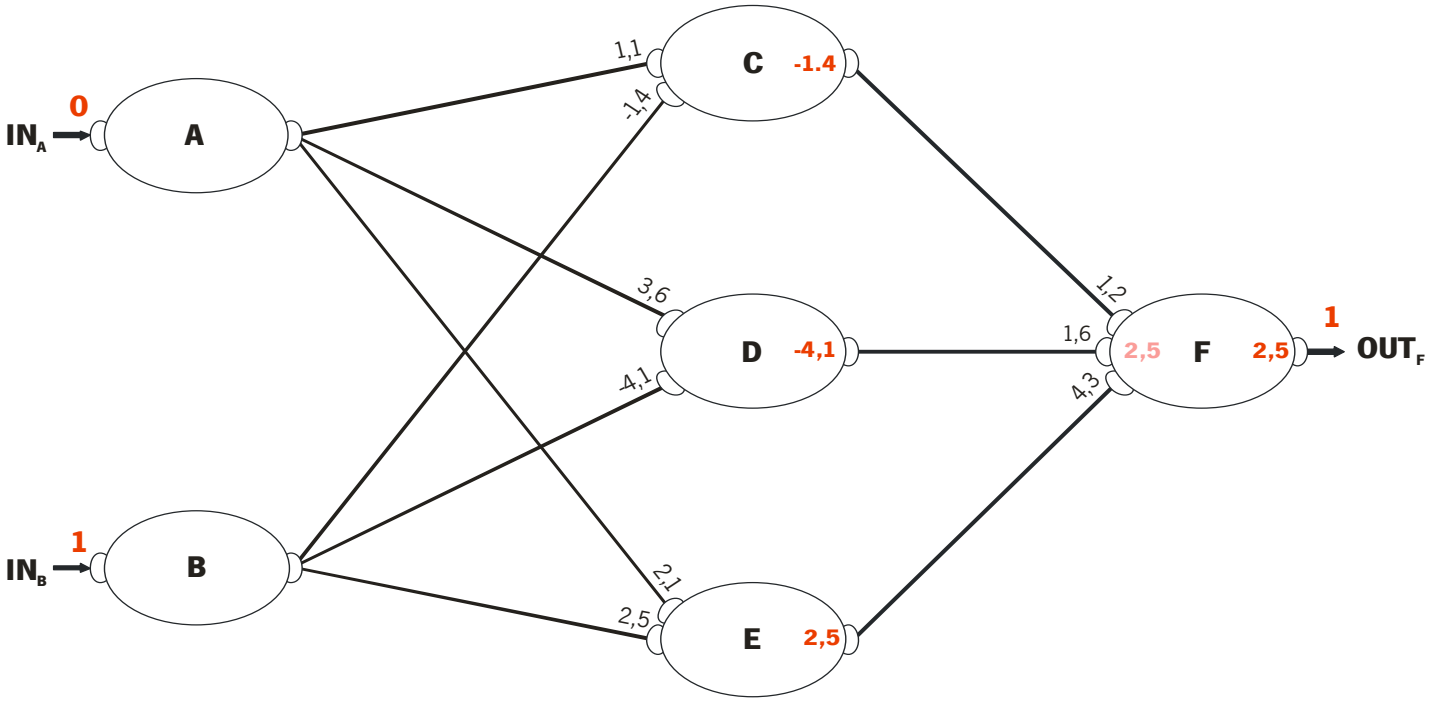


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

... and its transfer value.

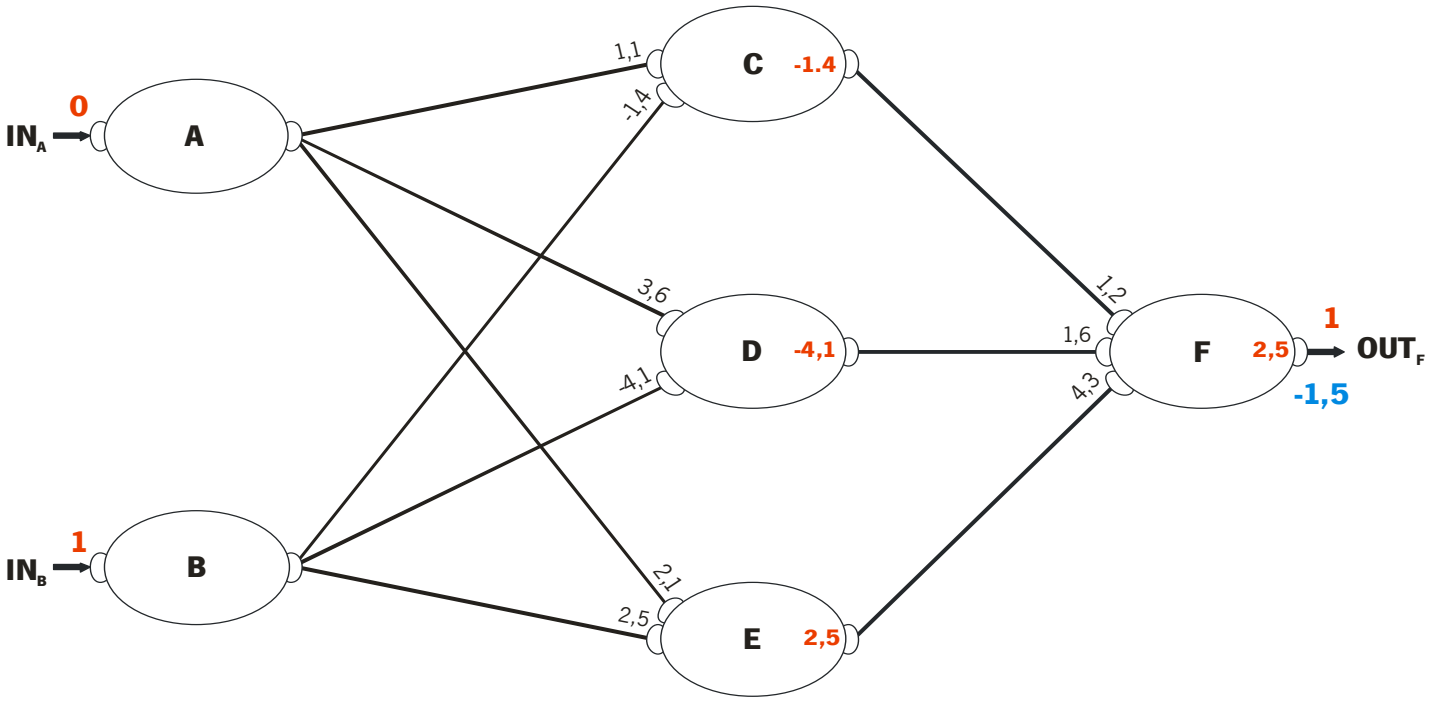


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

Computation of the error in the output layer ...

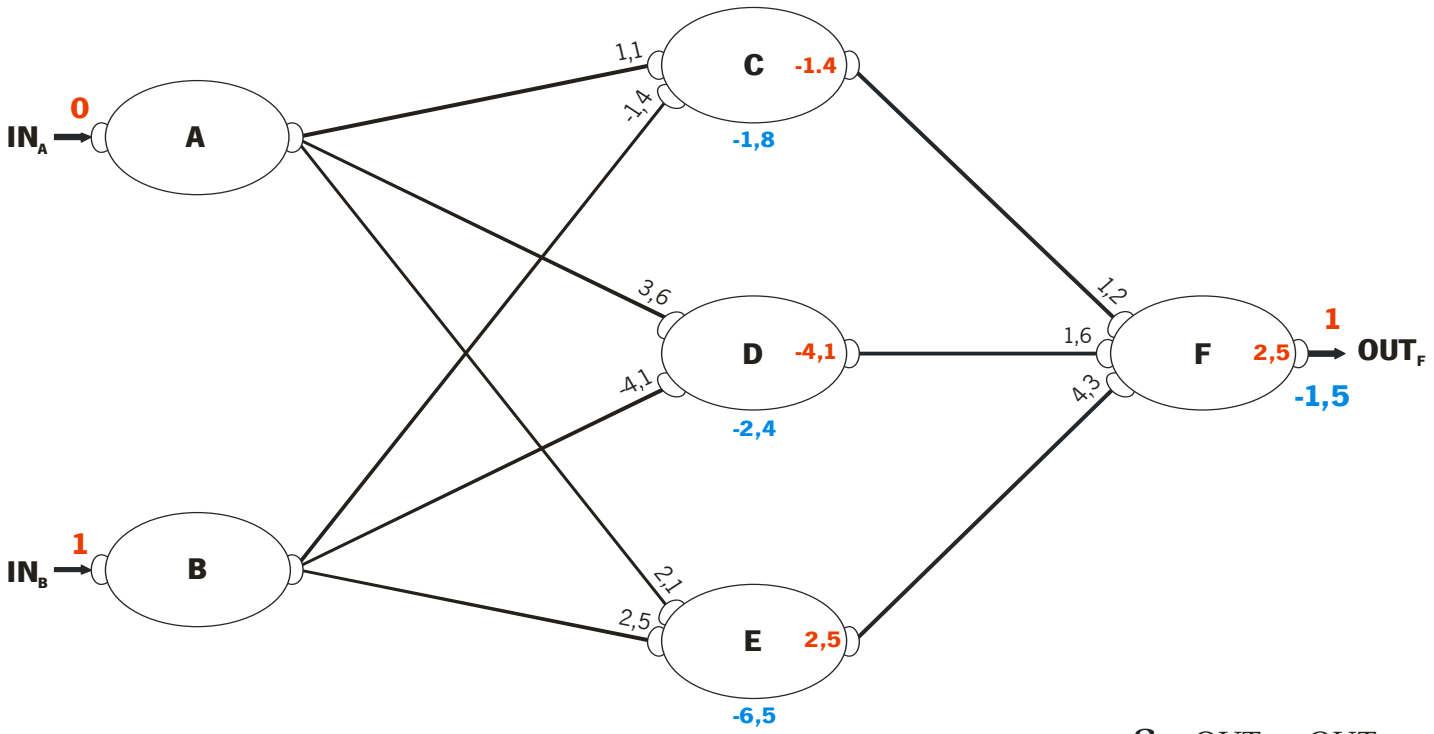


$$\mathcal{E} = OUT_D - OUT_C$$

$$\mathcal{E}_{\leftarrow} = \mathcal{E} \times P$$

RNA's training

... and calculating the estimated value of the error in the middle layer.

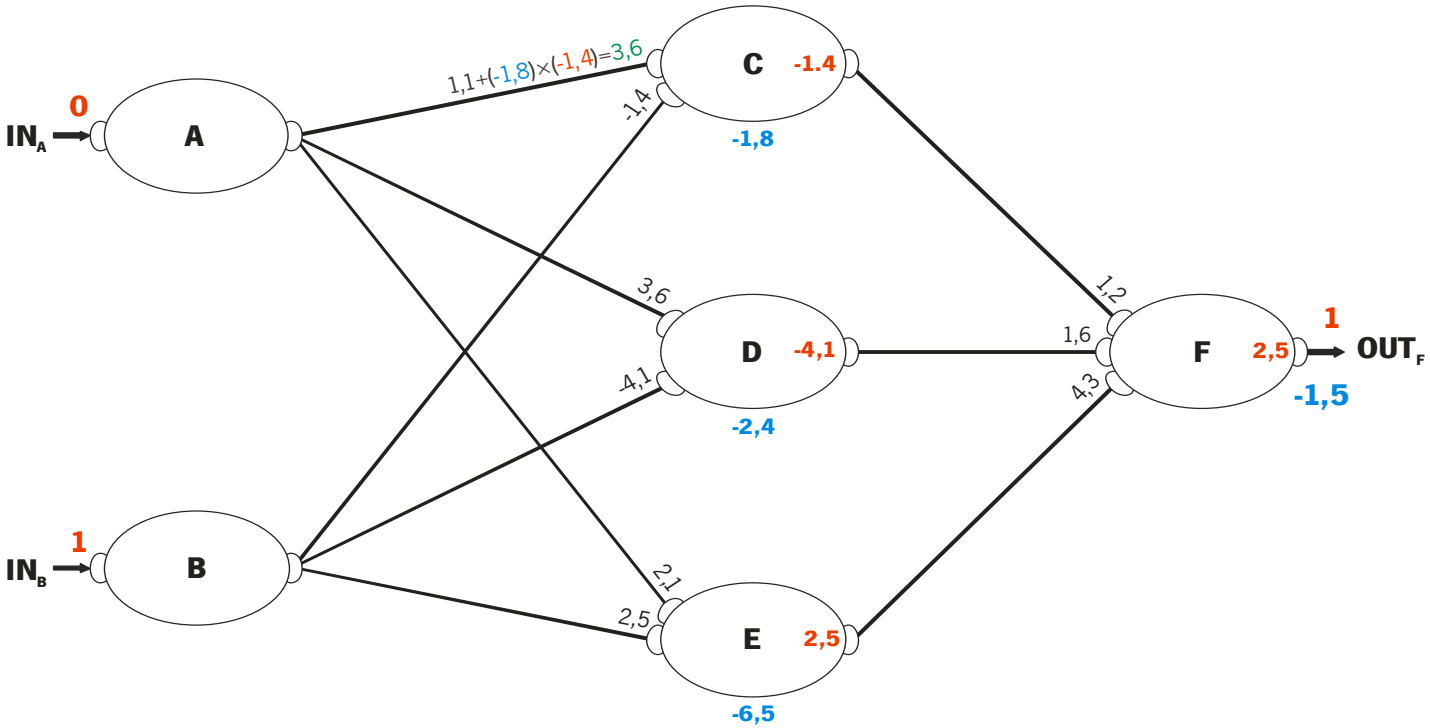


$$\mathcal{E} = OUT_D - OUT_C$$

$$\mathcal{E}_{\leftarrow} = \mathcal{E} \times P$$

RNA's training

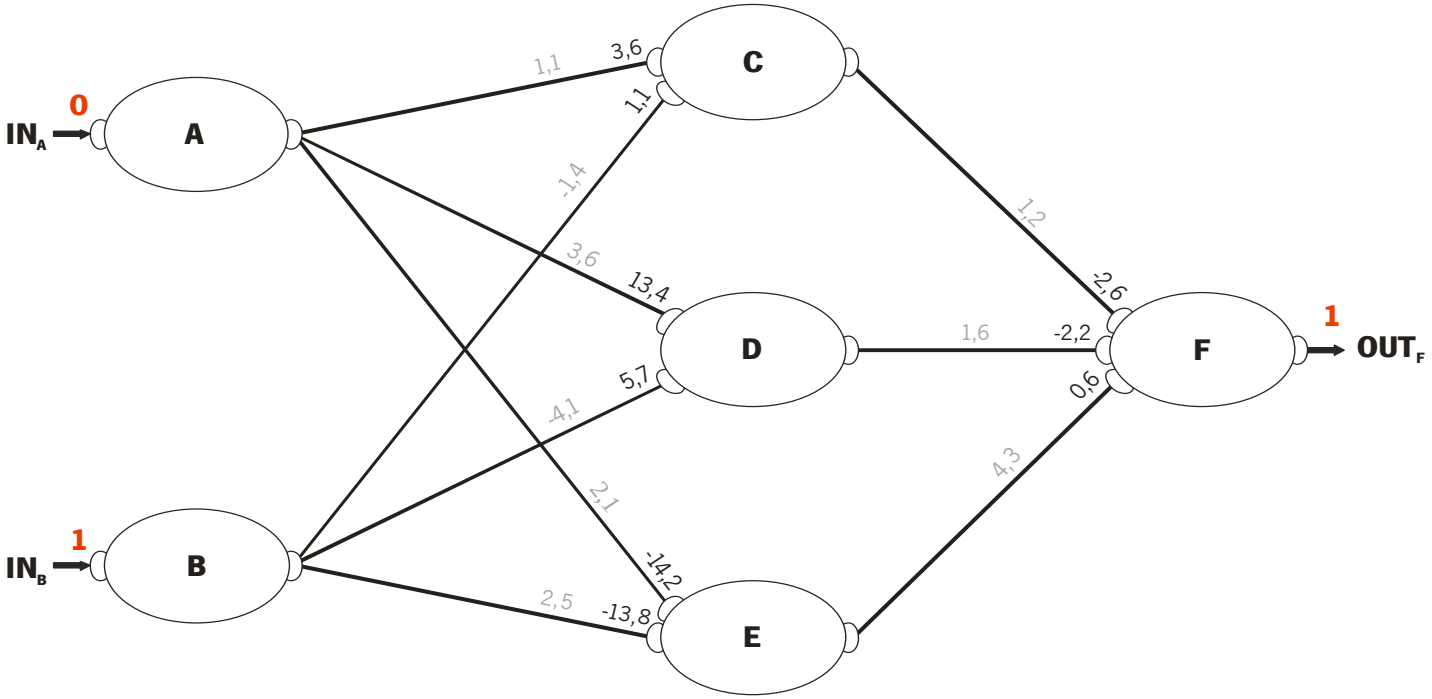
Application of a rule for updating synapse weights ...



$$P_{i+1} = P_i + \mathcal{E} \times f_T$$

RNA's training

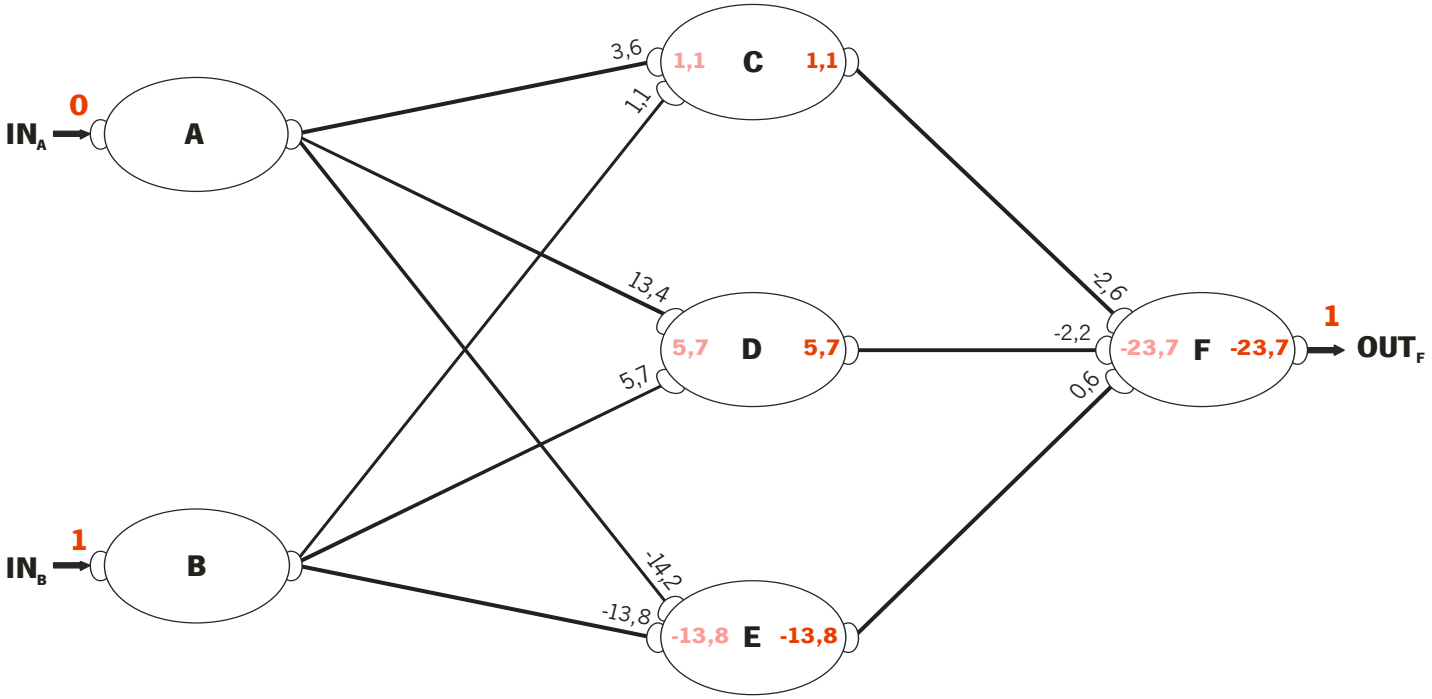
... to update the synapse values of all neurons.



$$P_{i+1} = P_i + \mathcal{E} \times f_T$$

RNA's training

Second iteration of the spread of the training case ...

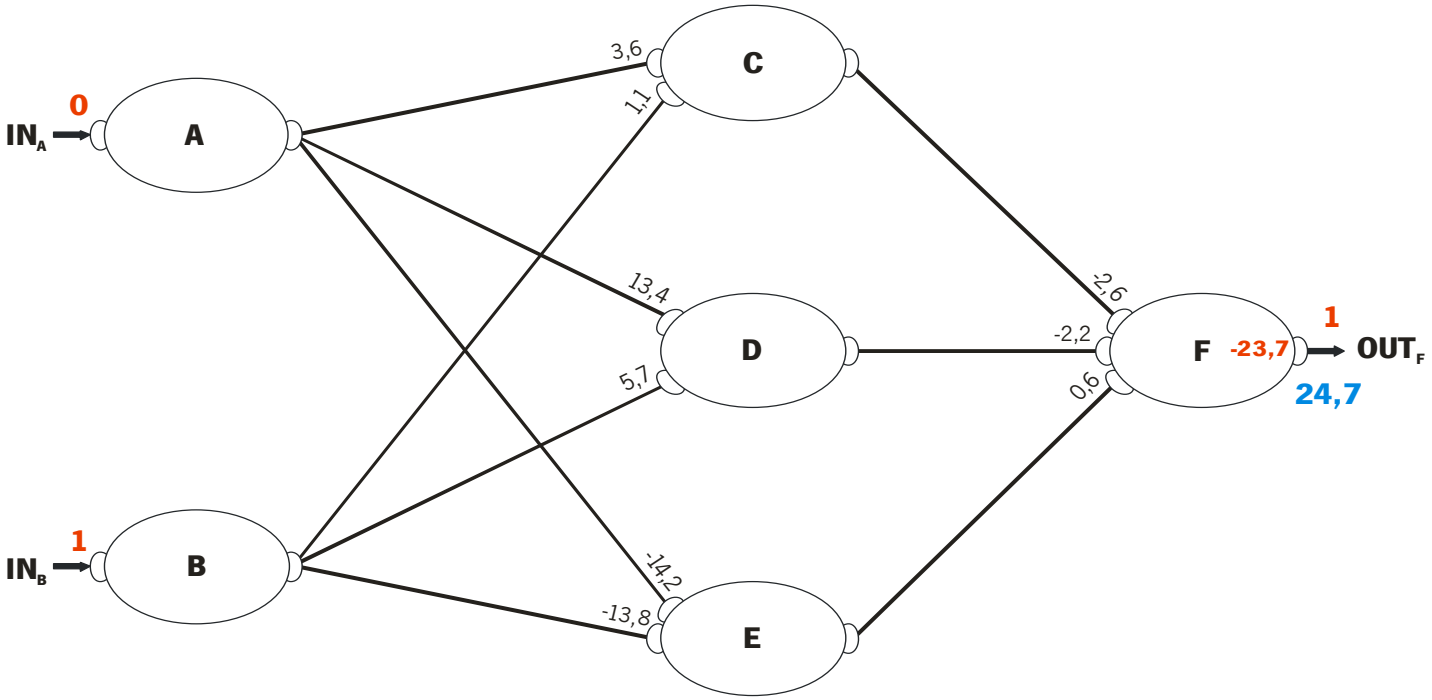


$$f_A(P,E) = \sum P \times E$$

$$f_T(A) = A$$

RNA's training

... and calculating the error produced by RNA in the second iteration.



Summary

Although there are numerous variants of neural networks, each can be defined in terms of:

- An **activation function**, which transforms a neuron's net input signal into a single output signal to be broadcasted further in the network;
- A **network topology** (or architecture), which describes the number of neurons in the model as well as the number of layers and manner in which they are connected;
- The **training algorithm** that specifies how connection weights are set in order to inhibit or excite neurons in proportion to the input signal.

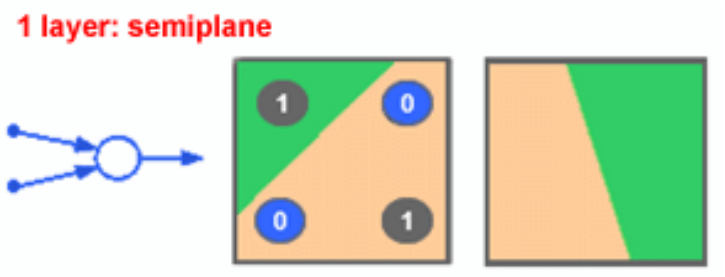
ISLab
Synthetic Intelligence Lab

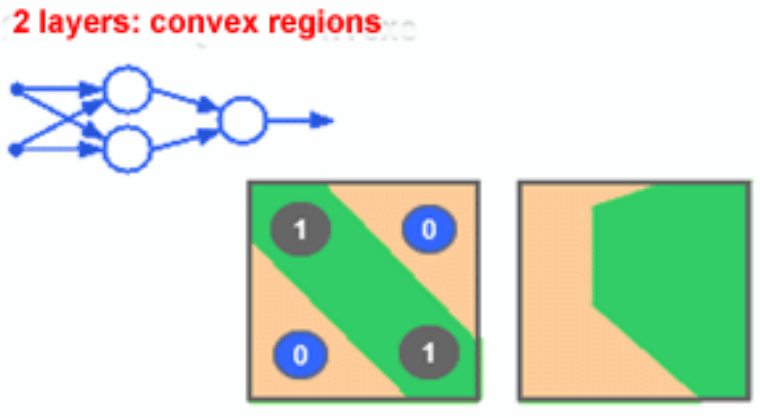
Network Topology

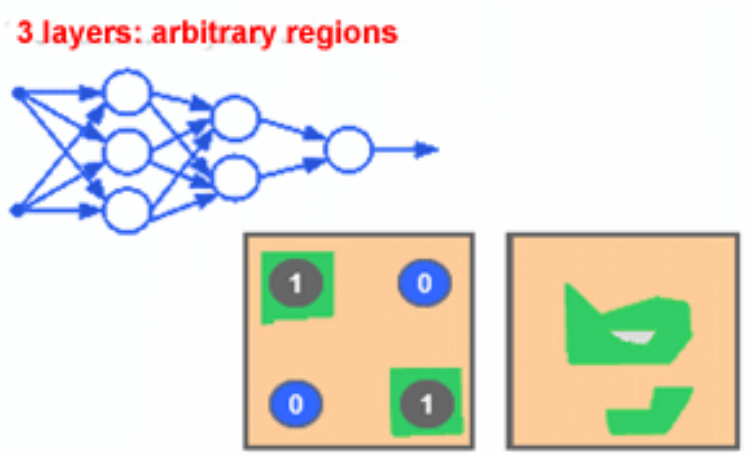
- The capacity of a neural network to learn is rooted in its topology, or the patterns and structures of interconnected neurons;
- Determines the complexity of tasks that can be learned by the network;
  - generally, larger and more complex networks are capable of identifying more subtle patterns and complex decision boundaries
  - however, the power of a network is not only a function of the network size, but also the way units are arranged

  - The number of layers
  - The direction of information flow
  - The number of nodes within each layer of the network

Decision regions of multilayer perceptrons







ISLab
Synthetic Intelligence Lab

Number of nodes in each layer

- The number of input nodes is predetermined by the number of features in the input data;
- The number of output nodes is predetermined by the number of outcomes to be modeled or the number of class levels in the outcome;
- The number of hidden nodes is left to the user to decide prior to training the model there is no reliable rule to determine the number of neurons in the hidden layer:
  - A greater number of neurons will result in a model that more closely mirrors the training data, but this runs a risk of overfitting -> it may generalize poorly to new data
  - Large neural networks can also be computationally expensive and slow to train

Use the fewest nodes that result in adequate performance on a validation dataset

ISLab
Synthetic Intelligence Lab

Different techniques of ANNs

- **Classification Neural Network**
  - In a Classification Neural Network, the network can be trained to classify any given patterns or datasets into a predefined class. It uses Feedforward Networks to do this.
- **Prediction Neural Network**
  - In a Prediction Neural Network, the network can be trained to produce outputs that are expected from a given input. The network 'learns' to produce outputs that are similar to the representation examples given in the input.
- **Clustering Neural Network**
  - The Neural network can be used to identify a unique feature of the data and classify them into different categories without any prior knowledge of the data.
  - Following networks are used for clustering:
    - Competitive networks;
    - Adaptive Resonance Theory Networks;
    - Kohonen Self-Organising Maps.
- **Association Neural Network**
  - An Association Neural Network can be trained to remember a particular pattern, which enables any noise patterns presented to the network to be associated with the closest one in the memory or discard it.

```
# Load Data

(x_train, y_train), (x_test, y_test) =
tensorflow.keras.datasets.mnist.load_data()

# Scale images to the [0, 1] range
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255

# Make sure images have shape (28, 28, 1)
x_train = numpy.expand_dims(x_train, -1)
x_test = numpy.expand_dims(x_test, -1)

# Convert class vectors to binary class matrices
y_train = tensorflow.keras.utils.to_categorical(y_train, num_classes)
y_test = tensorflow.keras.utils.to_categorical(y_test, num_classes)
```

```
# Build the model
model = keras.Sequential(
   [keras.Input(shape=input_shape),
      layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
      layers.MaxPooling2D(pool_size=(2, 2)),
      layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
      layers.MaxPooling2D(pool_size=(2, 2)),
      layers.Flatten(),
      layers.Dropout(0.5),
      layers.Dense(num_classes, activation="softmax")])

# Train the model
model.compile(loss="categorical_crossentropy",
optimizer="adam", metrics=["accuracy"])
model.fit(x_train, y_train, batch_size=128, epochs=15,
validation_split=0.1)
```

ISLab
Synthetic Intelligence Lab

Deep Learning

Training **deep neural networks** (more than 5-10 layers) could only be possible in recent times with:

- Faster computing resources (GPU)
- Larger labeled training sets

Specialized ANN Architectures:

- Convolutional Neural Networks (for image data)
- Recurrent Neural Networks (for sequence data)
- Residual Networks (with skip (shortcut)) connections)

Generative Models: Generative Adversarial Networks

Source: Introduction to Data Mining, Pang Ning Tan, Michael Steinbach, Anuj Karpatne, Vipin Kumar, ISBN 9780133128901, 2018.

ISLab
Synthetic Intelligence Lab

Artificial Neural Networks

Advantages

- Classification accuracy is usually high, even for complex problems;
- Distributed processing, knowledge is distributed through connection weights;
- Robust in handling examples even if they contain errors;
- Handle redundant attributes well, since the weights associated with them are usually very small;
- Results can be discrete, real values, or a vector of values (discrete or real).

Disadvantages

- Difficult to determine the optimal network topology for a problem;
- Difficult to use - have many parameters to define;
- Requires specific data pre-processing;
- Requires long training time;
- Difficult to understand the learning function (weights);
- Discovered knowledge is not readable;
- Do not provide explanations of the results;
- Domain knowledge incorporation is not easy.

ISLab
Synthetic Intelligence Lab

References

Minsky, M., & Papert, S., Perceptrons. M.I.T. Press, 1969.

R. P. Lippman, An Introduction to Computing with Neural Nets, IEEE Acoustics, Speech and Signal Processing Magazine, 1987.

Christopher Bishop, Neural Networks for Pattern Recognition, Clarendon Press;, 1 edition, 1996.

Paulo Cortez e José Neves, Redes Neuronais Artificiais, Universidade do Minho, 2000.

Pang-Ning Tan; Michael Steinbach; Anuj Karpatne; Vipin Kuma, Introduction to Data Mining, 2nd Edition, Pearson, ISBN: 9780133128901, 2019.

Prechelt L. (1998) Early Stopping - But When?. In: Orr G.B., Müller KR. (eds) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, vol 1524. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-49430-8_3

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. 15, 1 (January 2014), 1929–1958.

ISLab

Universidade do Minho
Escola de Engenharia
Departamento de Informática

Dados e Aprendizagem Automática

Neural Networks

DAA @ MEI/1º ano – 1º Semestre