Implementing McCulloch and Pitts Neuron in Python

Programming Simple "Neural Networks" Using Python to Model Basic Neurons

Hernández Pérez Erick Fernando

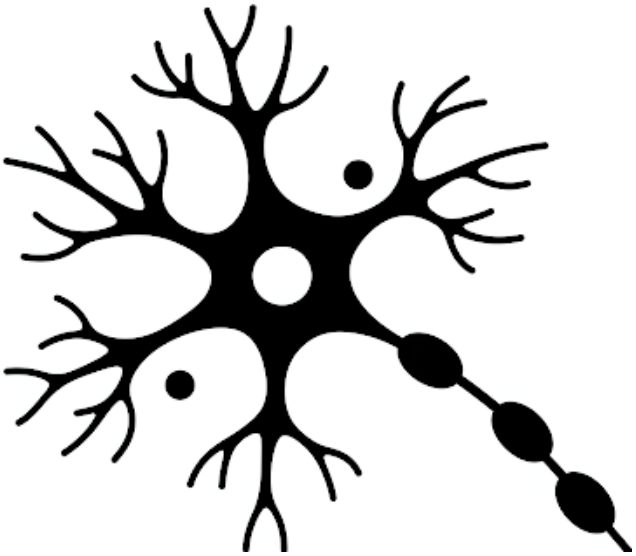


Figure 1: Neuron

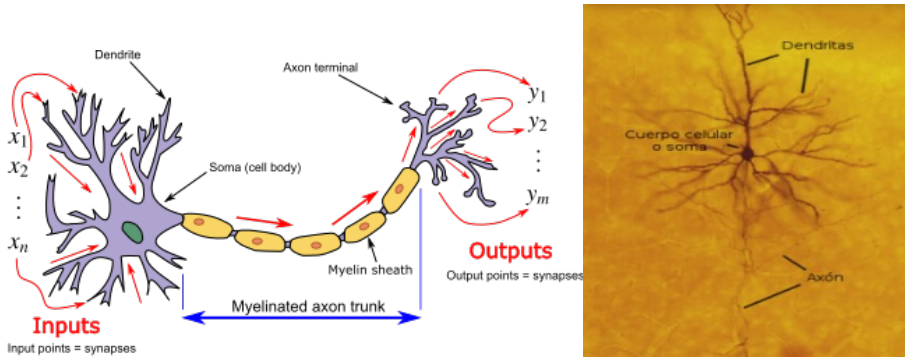
We live. A lot can be said in so little. At this moment our body performs a large number of tasks, we breathe, our heart beats, we read these words, etc. We achieve all this from the brain. It is considered the governing organ of the human body, thanks to it movements can be carried out and through perception significant learning can be achieved (Morerira et al., 2021)[4]. However, it is only one part of the nervous system.

The nervous system is divided into two parts, the central nervous system, made up of the brain and spinal cord, and the peripheral nervous system, whose components are all the nerves that branch out from the spinal cord and extend to all parts of the body. As stated by Megías et al., these two systems allow animals, including humans, to communicate with the outside environment and with the inside of the body, capturing signals, processing information and emitting a response (2023) [3]. At this point we can identify three essential components regarding this process: capturing, processing and responding. The basic components of these systems are neurons and glial cells.

Neurons form more or less numerous groups of neurons connected to each other forming circuits, communicating with each other through specializations in their membranes called synapses. For example, some neurons specialize in communicating with muscle cells through synapses called motor plates (Megías et al., 2023)[3].

Everything we are is owed to approximately 100 billion neurons in our brain according to the Eunice Kennedy Shriver National Institute of Child Health and Human Development (n.d.)[5]. These neurons are divided into soma, dendrites and axon. Inside the soma is the nucleus, the endoplasmic reticulum, the Golgi apparatus, etc. On the other hand, dendrites are the main information-receiving element of neurons; they look like branches and hence their name, dendron in Greek means tree. Finally, we have the axon, which is a thin extension that starts from the soma and is normally branched. It is responsible for transporting and transmitting the information, collected and integrated by the dendrites and the soma, to other neurons. Cellular materials are also transported along the axon between the soma and the synaptic terminals. In Figure 2a we can see a diagram of a neuron, while in Figure 2b we can see a neuron from the cerebral cortex of a rat impregnated with the Golgi technique.

In a very simplified way, and for practical purposes, we can say that the dendrites are the way in which the neuron receives information and processes it, through the direction of the soma, to then give a response-output



(a) Diagram of a neuron

(b) Neuron from the cerebral cortex of a rat

Figure 2

from the axon through which other neurons will receive said information, repeating the cycle. As we will see later, these three parts have a clear reference in the McCulloch-Pitts neuron model to try to replicate the behavior of a natural neuron.

McCulloch-Pitts neuron

In 1943, the state of neurophysiology was very different from today. The ionic and electrical bases of neuronal activity were not clear [...] The observed cellular activity was the 'all or nothing' action potential. It was not possible to take intracellular records (Prieto et al., 2000)[6]. It was also not possible to see how postsynaptic potentials were understood for a good number of milliseconds and also, it was not understood that neurons function more as frequency voltage converters.

This was, broadly speaking, the neurophysiological context that surrounded Warren S. McCulloch and Walter Pitts at the time they published their neuronal model. As mentioned by Prieto et al., this was an attempt to explain the functioning of the nervous system from small computational mechanisms based on abstractions of what was thought to be how neurons and their connections functioned (2000) [6].

In their paper "A logical calculus of the ideas immanent in nervous activity", they focus solely on studying neuronal computation rather than focusing on its physiology and morphology as such. They focus on studying the computational features and capabilities of their model, characterizing it as a logical device. The McCulloch-Pitts model is based on five assumptions (McCulloch & Pitts, 1943)[2]:

1. The activity of a neuron is an "all or nothing" process.
2. A certain fixed number of synapses must be excited within a latent addition period in order to excite a neuron in any time interval (threshold), and this number is independent of the previous activity and position of the neuron.
3. The only significant delay within the system is the synaptic delay.
4. The activity of any inhibitory synapse absolutely prevents the excitation of the neuron in that time interval.
5. The structure of the network does not change over time.

We can draw some conclusions from these assumptions. The McCulloch-Pitts neuron is a binary device that can only be in one of two possible states. Each neuron can receive inputs from excitatory synapses that have the same weight, just as it can receive inputs from inhibitory synapses that, if present, the neuron cannot be active. There is a fixed given time for the integration of synaptic inputs, which gives it its discrete-time character.

It is also important to mention that the authors start from a series of simplifications in which they consider that the firing threshold of the neurons adopts discrete values, and that these thresholds also remain unchanged. With this inmind, we can account for the mode of operation of the neuron.

The mode of operation of the McCulloch-Pitts neuron is simple. During the integration time, the neuron

responds to the activity of its synapse, which reflects the state of the presynaptic cells. If there are no active inhibitory synapses, the neuron sums its synaptic inputs and checks whether the sum of them meets or exceeds its threshold level. If it does, then the neuron becomes active. If not, the neuron remains inactive. In case there are any active inhibitory synapses, the neuron remains inactive. (Prieto et al., 2000) [6].

And as McCulloch and Pitts put it in their introduction:

> The "all-or-none" law of nervous activity is sufficient to insure that the activity of any neuron may be represented as a proposition. Physiological relations existing among nervous activities correspond, of course, to relations among the propositions; and the utility of the representation depends upon the identity of these relations with those of the logic of propositions. (McCulloch & Pitts, 1943)[2]

We can say that the McCulloch-Pitts neuron does threshold logic. In this way, complex propositions can be created from connections between simple propositions, which leads us to the idea of connecting neurons and creating a network for this purpose; as long as we are working with finite logical expressions.

As Prieto et al. point out, and as the reader will be able to see for himself, the formal proofs and notation used by McCulloch and Pitts are exceptionally difficult to understand, since their mathematical notation is based on temporal propositional calculus. A better introduction to McCulloch-Pitts neurons is Marvin Minsky's book "Computation: Finite and Infinite Machines." Here the authors' results are extended and neurons are placed in the perspective of later work on automata theory and the theory of computation (2000) [6]. In what follows, Minsky's work will be used.

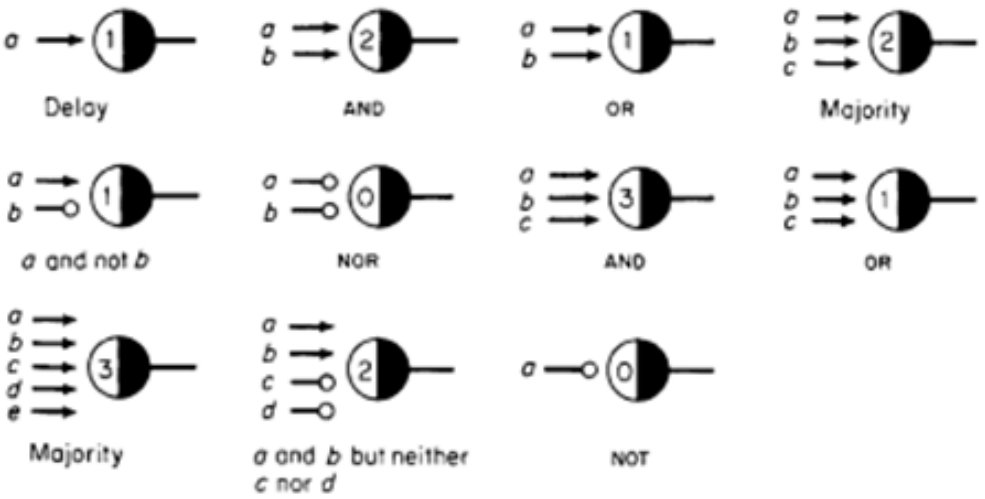


Figure 3: Examples of McCulloch-Pitts neurons

We can see that the circular figure in Figure 3 is the representation of the neuron, from which emerges a single line called the "output fiber" that will be the input connection of another neuron or perhaps to itself. We can also see the two inputs: the excitatory inputs that are represented by an arrow and the inhibitory inputs that are represented by a line ending in a circle. In this way, neurons become finite state machines.

As García points out in his degree work: A Finite State Machine (FSM) is a behavioral model of a system in which the output value depends on the current value and the previous values of the input. The model is based on the definition of a finite set of states that determine the operation of the FSM, that is, its response to the input values. The state of the machine is determined by the evolution of the input values; in this way, the output depends only on the current input and state. (García, 2016)[1].

Neurons only have two possible states, on or off. Each state corresponds to an output signal and is transmitted through its output fiber. Neurons change state depending on the inputs they receive. In the center of the circle, which represents our neuron, we can see the value of the neuron's threshold. In the following illustration we can see some behaviors of neurons.

We will formalize the behavior of a McCulloch-Pitts neuron in order to better understand it and to use it as a starting point when programming it in Python. We will represent a neuron with the letter C and we will define that the input of the neuron, which is called input, is a tuple (E, I), where E is the set of excitatory signals and I is the set of inhibitory signals. We will have as a restriction that both the input and the output

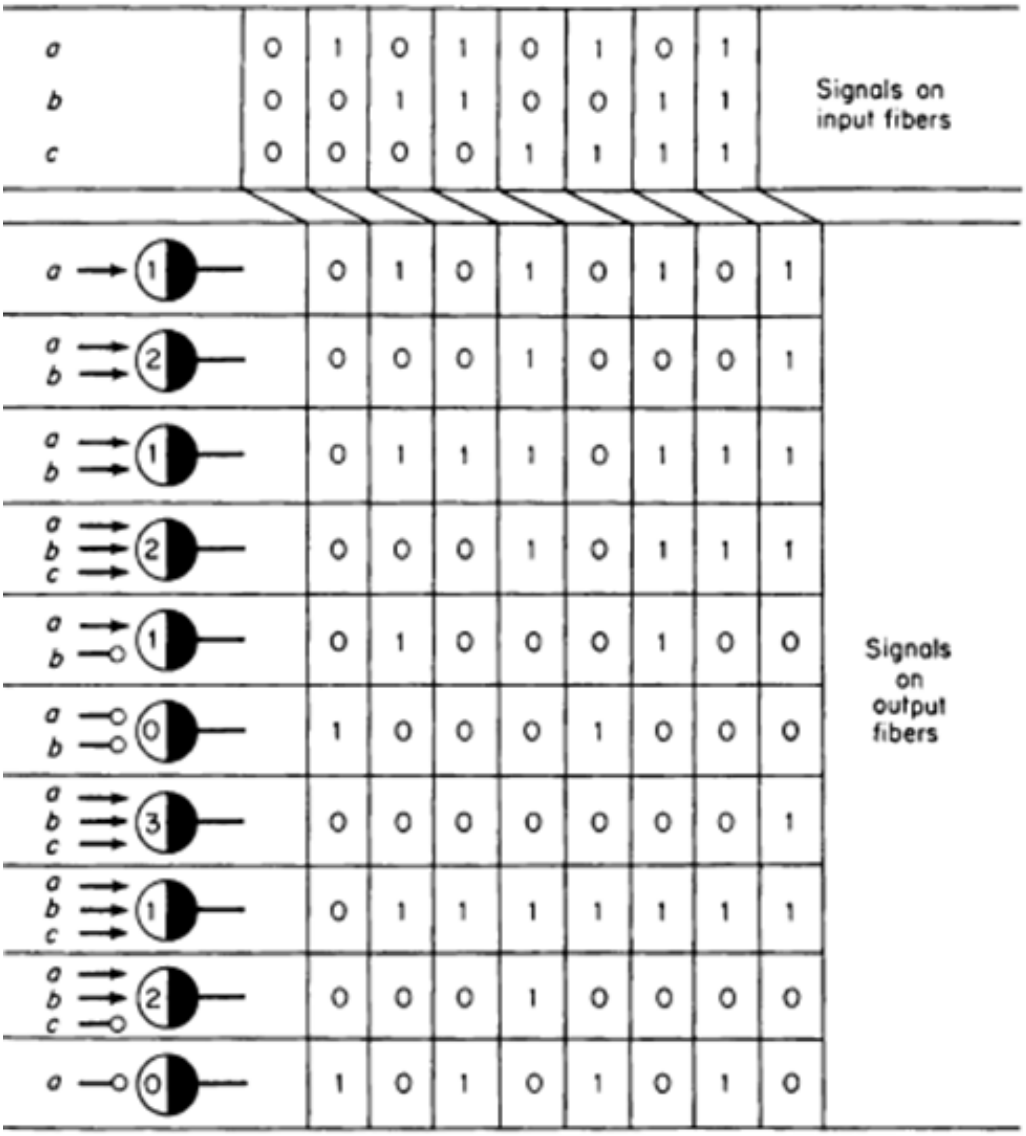


Figure 4: Behavior of some McCulloch-Pitts neurons

are binary variables, this means that $input, output \in 0, 1$. Likewise, we will define a threshold U that the neuron uses to determine the cases of excitation.

The output of C is calculated with the following rules:

1. If there is an active inhibitory input, then the neuron is not excited:

$$C(E, I) = 0 \text{ if } \exists i \in I : i = 1$$

2. The neuron is excited if the sum of the excitatory inputs is equal to or greater than the threshold, that is:

$$C(E, I) = 1 \text{ if } \sum e_i \geq U, e_i \in E$$

Python implementation

McCulloch Pitts Neuron

```python
import numpy as np

class neuronMP:
    def __init__(self, E, I, U):
        self.E = np.array(E)
        self.I = np.array(I)
        self.U = U

    def getOutput(self):
        sumE = self.E.sum()
        sumI = self.I.sum()

        if sumI == 0 and sumE >= self.U:
            return 1
        else:
            return 0

def AND(A,B):
    return neuronMP(E=[A, B], I=[], U=2).getOutput()

def OR(A,B):
    return neuronMP(E=[A, B], I=[], U=1).getOutput()

def NOT(A):
    return neuronMP(E=[], I=[A], U=0).getOutput()

def NAND(A,B):
    return NOT(AND(A,B))

def NOR(A,B):
    return NOT(OR(A,B))
```

In this way, three basic logic gates have been implemented using a single McCulloch-Pitts neuron, demonstrating the ability of this model to represent fundamental logical operations. Additionally, two more complex logic gates have been built by sequentially interconnecting two neurons, combining the negation operation (NOT) with the conjunction (AND) and disjunction (OR) operations.

Taking McCulloch and Pitts' work as a reference, it is possible to build a structure that could be called a "neural network". It is important to note that this does not correspond to a modern perceptron-based neural network, but to a conceptual model in which multiple McCulloch-Pitts neurons are interconnected to perform logical calculations. An illustrative example of this approach is the implementation of the XOR logic gate, which cannot be represented by a single McCulloch-Pitts neuron due to its non-linear nature. However, by combining simpler logic gates, such as AND, OR and NOT, it is possible to build a circuit that reproduces their behavior, demonstrating the ability of these artificial neurons to model more complex functions through appropriate interconnections.

There are three well-known methods to implement the XOR logic gate from combinations of more basic gates:

1. Using only NAND gates: A minimum of four NAND gates are required to build the XOR function. Figure 5a.

$$A \oplus B = \overline{(\overline{A \cdot (\overline{A \cdot B})})} \cdot \overline{((\overline{A \cdot B}) \cdot (B))}$$

2. Using only NOR gates: In this case, at least five NOR gates are needed. Figure 5b.

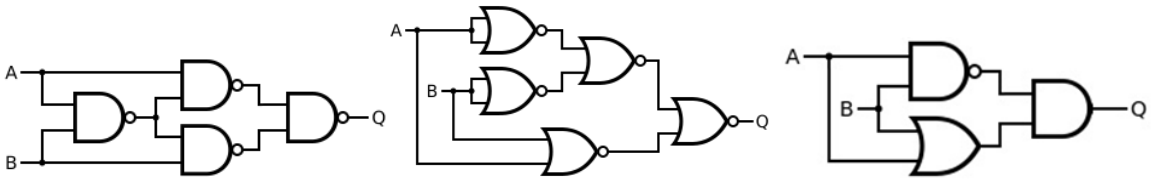
$$A \oplus B = \overline{\overline{(\overline{A+A})+(\overline{B+B})}+\overline{(A+B)}}$$

3. Using a combination of AND, OR and NAND: A more direct way to obtain XOR, Figure 5c, is by using the equation:

$$A \oplus B = (A+B)\cdot\overline{(A\cdot B)}$$

XOR gate

```
def XOR3(A,B):
    return AND(NAND(A,B),OR(A,B))

def XOR4(A,B):
    return NAND(NAND(A,NAND(A,B)),NAND(NAND(A,B),B))

def XOR5(A,B):
    return NOR(NOR(NOR(A,A),NOR(B,B)),NOR(A,B))

print(XOR3(0,0)) # 0
print(XOR3(0,1)) # 1
print(XOR3(1,0)) # 1
print(XOR3(1,1)) # 0

print(XOR4(0,0)) # 0
print(XOR4(0,1)) # 1
print(XOR4(1,0)) # 1
print(XOR4(1,1)) # 0

print(XOR5(0,0)) # 0
print(XOR5(0,1)) # 1
print(XOR5(1,0)) # 1
print(XOR5(1,1)) # 0
```



(a) 4 NAND (b) 5 NOR (c) AND, NAND, OR

Figure 5

If we wanted a representation similar to the one used by Minsky.

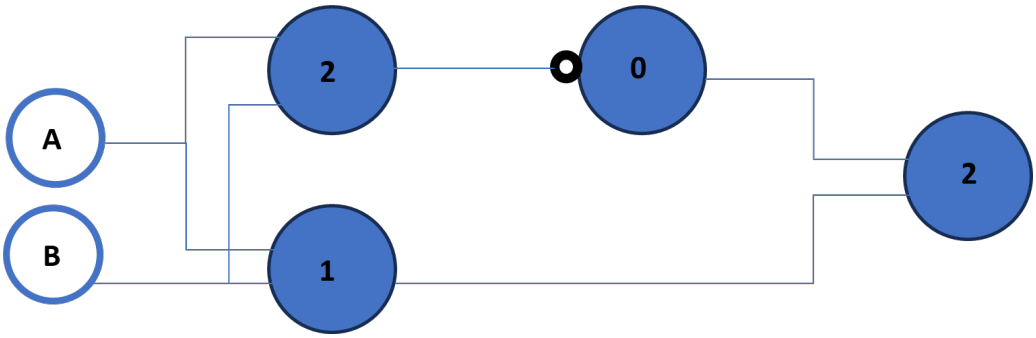


Figure 6: Minksy representation

References

[1] García, I. (2016). Máquinas de estados finitos con multiplexión de entradas: una contribución al diseño e implementación electrónica de máquinas de estados. [Finite state machines with input multiplexing: a contribution to the electronic design and implementation of state machines] (PhD thesis). Universidad de Sevilla, Departamento de Arquitectura y Tecnología de Computadores. Sevilla. https://idus.us.es/items/6338b753-0543-4893-89ba-25302808e7c0

[2] McCulloch, W. & Pitts, W. (1943) *A logical calculus of the ideas immanent in neurons activity*, Bull. Math. Biophys., 5, 115-133.
https://www.cs.cmu.edu/~./epxing/Class/10715/reading/McCulloch.and.Pitts.pdf

[3] Megías, M., Molist, P. & Pombal, A. (2023). *Tipos celulares. Neuronas* [Cell types. Neurons]. Facultad de Biología, Universidad de Vigo.
https://atlashistologia.webs7.uvigo.es/pdfs-descargas/tipos-cel-neurona.pdf

[4] Moreria, M., Morales, F., Zambrano, G., & Rodríguez, M. (2021). El cerebro, funcionamiento y la generación de nuevos aprendizajes a través de la neurociencia. [The brain, its functioning and the generation of new learning through neuroscience.] *Revista Dominio de las Ciencias,* 7(1), 50-67.

[5] Eunice Kennedy Shriver National Institute of Child Health and Human Development. ¿Cuáles son las partes del sistema nervioso? [What are the parts of the nervous system?]
https://espanol.nichd.nih.gov/salud/temas/neuro/informacion/partes

[6] Prieto, R., Herrera, A., Pérez, J. L., & Padrón, A. (2000). *El modelo neuronal de McCulloch y Pitts: Interpretación comparativa del modelo* [The McCulloch-Pitts neural model: Comparative interpretation of the model]. Laboratorio de Computación Adaptativa, Centro de Instrumentos, UNAM.
https://www.researchgate.net/publication/343141076_EL_MODELO_NEURONAL_DE_McCULLOCH_Y_PITTS_Interpretacion_Comparativa_del_Modelo