

# Práctica #1 Célula de McCulloch-Pitts

Jorge Gómez Reus

## Índice

<b>1. Introducción</b>	<b>1</b>
1.1. Modelo . . . . .	1
<b>2. Diagrama de Flujo</b>	<b>3</b>
<b>3. Fórmula general para las compuertas AND y OR</b>	<b>4</b>
<b>4. Resultados</b>	<b>4</b>
4.1. Compuerta NOT . . . . .	4
4.2. Compueta AND . . . . .	4
4.3. Compuerta OR . . . . .	4
<b>5. Discusión de Resultados</b>	<b>4</b>
<b>6. Referencias</b>	<b>4</b>
<b>7. Apéndice</b>	<b>4</b>

## 1. Introducción

La célula de McCulloch-Pitts fue el primer modelo de una neurona biológica como un dispositivo de dos estados:

- Apagado(0)
- Encendido(1)

Es la unidad esencial con la cual se construye una red neuronal artificial

En esta práctica usaremos un modelo similar a este ya que el umbral no se aplica a la suma, si no que es parte de la función de activación. Usaremos aprendizaje supervisado ya que  $\forall$  conjunto de valores  $v$   $\exists$  un target  $t$

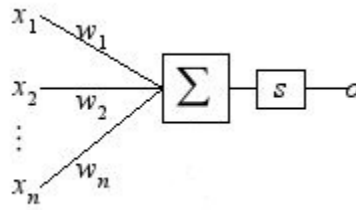
### 1.1. Modelo

Donde  $x_1, x_2, \dots, x_n$  son los valores de entrada,  $w_1, w_2, \dots, w_n$  son los pesos sinápticos,  $\theta$  es un valor de umbral que se usa para activar la señal de entrada.

Matemáticamente podemos representar esta célula con las siguientes expresiones:

$$n = \sum_{i=1}^R W_i * P_i \quad (1)$$

Figura 1: Modelo



$$s = \begin{cases} 1 & \text{si } n > \theta \\ 0 & \text{en otro caso} \end{cases} \quad (2)$$

## 2. Diagrama de Flujo

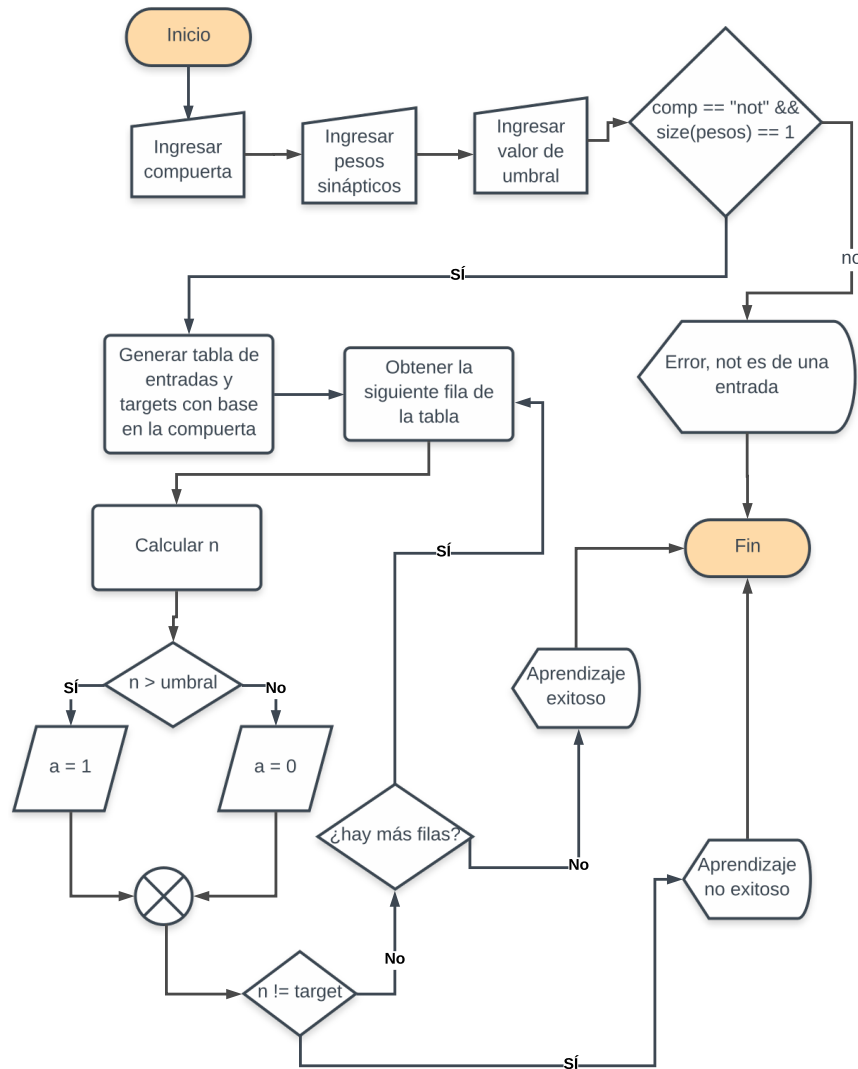


Figura 2: Diagrama de Flujo

### 3. Fórmula general para las compuertas AND y OR

### 4. Resultados

#### 4.1. Compuerta NOT

#### 4.2. Compueta AND

#### 4.3. Compuerta OR

### 5. Discusión de Resultados

### 6. Referencias

### 7. Apéndice

Listing 1: Código

```
1 % User input
2 gate = input('Ingrese la compuerta (and, or, not): ', 's');
3 syn_prompt = 'Ingrese el valor de los pesos sinapticos separados por espacios
4 (e.g. 1 2 3 4): ';
5 w = str2num(strip(input(syn_prompt, 's')));
6 theta = input('Ingrese el valor del umbral: ');
7 if (gate == "not" && size(w, 2) > 1)
8     fprintf("Error, la compuerta NOT es de una sola entrada");
9 else
10     % Generation of table for inputs and targets
11     model = logicalModel(size(w, 2), gate)
12     error = false;
13     % Iteration process
14     for i = 1:size(model, 1)
15         row = model(i, :);
16         % Calculation of n
17         n = sum(row(1:end-1).*w);
18         % Calculation of a
19         if(n > theta); a_n = 1; else; a_n=0; end
20         % Comparison between a and the threshold
21         fprintf("n_ %i = %i -> t_ %i = %i\n", i, a_n, i, row(end));
22         if(a_n ~= row(end)); error = true; break; end
23     end
24     if(~error); fprintf("El aprendizaje fue exitoso\n");
25     else; fprintf("El aprendizaje no fue exitoso\n"); end
26 end
27
28
29 function [table] = logicalModel(i, gate)
30     % logicalModel(I, gate) returns a matrix representing a truth table and
31     % the last column represents the ouput base on all the previous columns
```

```
32     % based on the (gate) parameter
33     % INPUT: (I) shall be an integer >= 1
34     % INPUT: (gate) shall be 'and' or 'or'
35     % OUTPUT: logicalModel is a binary matrix of size [2^I,I + 1]
36     % Heavily inspired in Paul Metcalf's CONDVECTS
37     % Acknowledgements: Paul Metcalf
38
39     g = 2;
40     i2 = 2^i;
41     table = false(i2,i + 1);
42     for m = 1 : 1 : i
43         m2 = 2^m;
44         m3 = (m2/2)-1;
45         i3 = i-m+1;
46         for g = g : m2 : i2
47             for k = 0 : 1 : m3
48                 table(g+k,i3) = true;
49             end
50         end
51         g = m2+1;
52     end
53     if (gate == "and")
54         for row_index = 1:size(table, 1)
55             row = table(row_index,:);
56             res = row(1);
57             for e_index = 1:size(row, 2)-1
58                 res = res & row(e_index);
59             end
60             table(row_index, end) = res;
61         end
62     elseif (gate == "or")
63         for row_index = 1:size(table, 1)
64             row = table(row_index,:);
65             res = row(1);
66             for e_index = 1:size(row, 2)-1
67                 res = res | row(e_index);
68             end
69             table(row_index, end) = res;
70         end
71     elseif (gate == "not")
72         for row_index = 1:size(table, 1)
73             row = table(row_index,:);
74             res = ~row(1);
75             table(row_index, end) = res;
76         end
77     end
78 end
```

??