

# 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>3</b> |
| 3.1. AND . . . . .                                     | 3        |
| 3.2. OR . . . . .                                      | 4        |
| <b>4. Resultados</b>                                   | <b>4</b> |
| 4.1. Compuerta NOT . . . . .                           | 4        |
| 4.2. Compuerta AND . . . . .                           | 4        |
| 4.3. Compuerta OR . . . . .                            | 6        |
| <b>5. Discusión de Resultados</b>                      | <b>7</b> |
| <b>6. Conclusiones</b>                                 | <b>8</b> |
| <b>7. Referencias</b>                                  | <b>8</b> |
| <b>8. Apéndice</b>                                     | <b>8</b> |

## 1. Introducción

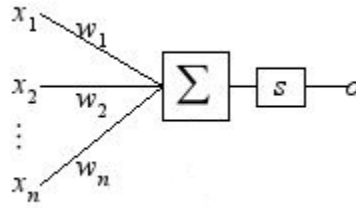
En 1943 Warren S. McCulloch, un neurocientífico, y Walter Pitts, un lógico, publicaron “A logical calculus of the ideas immanent in nervous activity” en el “Bulletin of Mathematical Biophysics” 5: 115-133. En este documento, McCulloch y Pitts intentaron comprender cómo el cerebro puede producir patrones altamente complejos mediante el uso de muchas células básicas que están conectadas entre sí, estas se llaman neuronas, McCulloch y Pitts dieron un modelo muy simplificado de una neurona en su papel, este ha hecho una contribución importante al desarrollo de redes neuronales artificiales, que modelan las características clave de las neuronas biológicas. La célula de McCulloch-Pitts fue el primer modelo de un 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 no tenemos bias. Usaremos aprendizaje supervisado ya que  $\forall$  conjunto de valores  $v \exists$  un target  $t$

Figura 1: Modelo



### 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:

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

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

## 2. Diagrama de Flujo

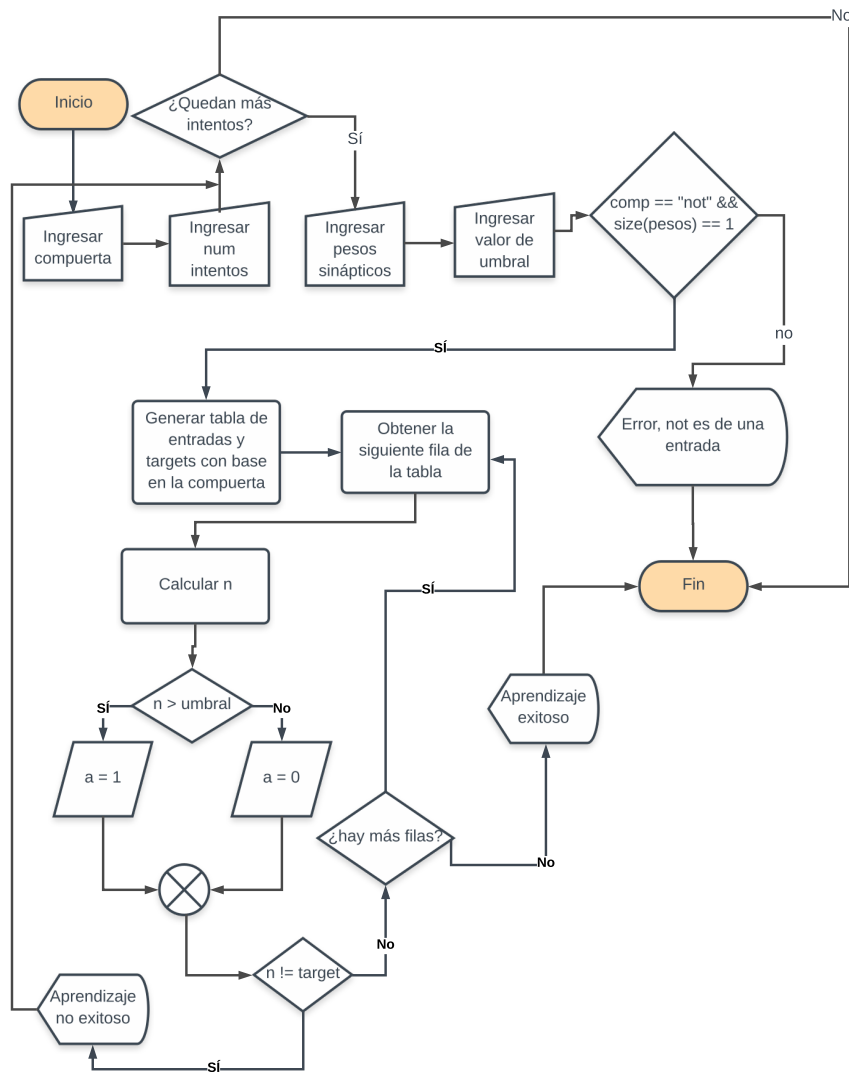


Figura 2: Diagrama de Flujo

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

Debido a que trabajamos con valores lógicos se propone que los pesos sinápticos siempre sean unos.

### 3.1. AND

La única forma de que la salida de esta sea "1" es que todas las entradas sean "1's", entonces  $n = w_1 * p_1 + w_2 * p_2 + \dots + w_n * p_n$  se puede simplificar a  $n = \text{tam}(w)$ , por lo tanto, n tiene que ser mayor al umbral solo en ese caso, entonces, el umbral sería  $\theta = \text{tam}(w) - 1$ .

Explicado matemáticamente para cuando la salida de AND es 1:

$$p = (p_1, p_2, \dots, p_r) = (1, 1, \dots, 1)$$

$$w = (w_1, w_2, \dots, w_r) = (1, 1, \dots, 1)$$

$$r > 0$$

$$n = w_1 * p_1 + w_2 * p_2 + \dots + w_r * p_r = \text{tam}(w)$$

$$f(n) = 1 \rightarrow (n > \theta)$$

$$\therefore \theta = \text{tam}(w) - 1$$

### 3.2. OR

La única forma en que la salida sea '0' es que todas las entradas sean '0', entonces  $n = w_1 * p_1 + w_2 * p_2 + \dots + w_n * p_n$  se puede simplificar a  $n = 0$ .  $n$  tiene que ser mayor al umbral para todos los demás casos, entonces el umbral sería 0.

Explicado matemáticamente para cuando la salida de OR es 0:

$$p = (p_1, p_2, \dots, p_r) = (0, 0, \dots, 0)$$

$$w = (w_1, w_2, \dots, w_r) = (1, 1, \dots, 1)$$

$$r > 0$$

$$n = w_1 * p_1 + w_2 * p_2 + \dots + w_r * p_r = 0$$

$$f(n) = 0 \rightarrow (n \leq \theta)$$

Dejando a un lado a los enteros negativos:  $\theta = 0$

## 4. Resultados

### 4.1. Compuerta NOT

Ingrese la compuerta (and, or, not): not  
 Ingrese el número de intentos: 1  
 Ingrese el valor de los pesos sinápticos separados por espacios(e.g. 1 2 3 4): -1  
 Ingrese el valor del umbral: -1

model =

2x2 logical array

```
0  1
1  0
```

a\_1 = 1 -> t\_1 = 1

a\_2 = 0 -> t\_2 = 0

El aprendizaje fue exitoso

### 4.2. Compueta AND

Ingrese la compuerta (and, or, not): and  
 Ingrese el número de intentos: 1  
 Ingrese el valor de los pesos sinápticos separados por espacios(e.g. 1 2 3 4): 1 1 1 1 1  
 Ingrese el valor del umbral: 4

model =

32x6 logical array

```
0  0  0  0  0  0
```

```

0  0  0  0  1  0
0  0  0  1  0  0
0  0  0  1  1  0
0  0  1  0  0  0
0  0  1  0  1  0
0  0  1  1  0  0
0  0  1  1  1  0
0  1  0  0  0  0
0  1  0  0  1  0
0  1  0  1  0  0
0  1  0  1  1  0
0  1  1  0  0  0
0  1  1  0  1  0
0  1  1  1  0  0
0  1  1  1  1  0
1  0  0  0  0  0
1  0  0  0  1  0
1  0  0  1  0  0
1  0  0  1  1  0
1  0  1  0  0  0
1  0  1  0  1  0
1  0  1  1  0  0
1  0  1  1  1  0
1  1  0  0  0  0
1  1  0  0  1  0
1  1  0  1  0  0
1  1  0  1  1  0
1  1  1  0  0  0
1  1  1  0  1  0
1  1  1  1  0  0
1  1  1  1  1  1

```

```

a_1 = 0 → t_1 = 0
a_2 = 0 → t_2 = 0
a_3 = 0 → t_3 = 0
a_4 = 0 → t_4 = 0
a_5 = 0 → t_5 = 0
a_6 = 0 → t_6 = 0
a_7 = 0 → t_7 = 0
a_8 = 0 → t_8 = 0
a_9 = 0 → t_9 = 0
a_10 = 0 → t_10 = 0
a_11 = 0 → t_11 = 0
a_12 = 0 → t_12 = 0
a_13 = 0 → t_13 = 0
a_14 = 0 → t_14 = 0
a_15 = 0 → t_15 = 0
a_16 = 0 → t_16 = 0
a_17 = 0 → t_17 = 0

```

```
a_18 = 0 -> t_18 = 0
a_19 = 0 -> t_19 = 0
a_20 = 0 -> t_20 = 0
a_21 = 0 -> t_21 = 0
a_22 = 0 -> t_22 = 0
a_23 = 0 -> t_23 = 0
a_24 = 0 -> t_24 = 0
a_25 = 0 -> t_25 = 0
a_26 = 0 -> t_26 = 0
a_27 = 0 -> t_27 = 0
a_28 = 0 -> t_28 = 0
a_29 = 0 -> t_29 = 0
a_30 = 0 -> t_30 = 0
a_31 = 0 -> t_31 = 0
a_32 = 1 -> t_32 = 1
El aprendizaje fue exitoso
```

### 4.3. Compuerta OR

Ingrese la compuerta (and, or, not): or

Ingrese el número de intentos: 1

Ingrese el valor de los pesos sinápticos separados por espacios(e.g. 1 2 3 4): 1 1 1 1 1

Ingrese el valor del umbral: 0

model =

32x6 logical array

```
0  0  0  0  0  0
0  0  0  0  1  1
0  0  0  1  0  1
0  0  0  1  1  1
0  0  1  0  0  1
0  0  1  0  1  1
0  0  1  1  0  1
0  0  1  1  1  1
0  1  0  0  0  1
0  1  0  0  1  1
0  1  0  1  0  1
0  1  0  1  1  1
0  1  1  0  0  1
0  1  1  0  1  1
0  1  1  1  0  1
0  1  1  1  1  1
1  0  0  0  0  1
1  0  0  0  1  1
1  0  0  1  0  1
1  0  0  1  1  1
1  0  1  0  0  1
1  0  1  0  1  1
```

```

1  0  1  1  0  1
1  0  1  1  1  1
1  1  0  0  0  1
1  1  0  0  1  1
1  1  0  1  0  1
1  1  0  1  1  1
1  1  1  0  0  1
1  1  1  0  1  1
1  1  1  1  0  1
1  1  1  1  1  1

```

```

a_1 = 0 -> t_1 = 0
a_2 = 1 -> t_2 = 1
a_3 = 1 -> t_3 = 1
a_4 = 1 -> t_4 = 1
a_5 = 1 -> t_5 = 1
a_6 = 1 -> t_6 = 1
a_7 = 1 -> t_7 = 1
a_8 = 1 -> t_8 = 1
a_9 = 1 -> t_9 = 1
a_10 = 1 -> t_10 = 1
a_11 = 1 -> t_11 = 1
a_12 = 1 -> t_12 = 1
a_13 = 1 -> t_13 = 1
a_14 = 1 -> t_14 = 1
a_15 = 1 -> t_15 = 1
a_16 = 1 -> t_16 = 1
a_17 = 1 -> t_17 = 1
a_18 = 1 -> t_18 = 1
a_19 = 1 -> t_19 = 1
a_20 = 1 -> t_20 = 1
a_21 = 1 -> t_21 = 1
a_22 = 1 -> t_22 = 1
a_23 = 1 -> t_23 = 1
a_24 = 1 -> t_24 = 1
a_25 = 1 -> t_25 = 1
a_26 = 1 -> t_26 = 1
a_27 = 1 -> t_27 = 1
a_28 = 1 -> t_28 = 1
a_29 = 1 -> t_29 = 1
a_30 = 1 -> t_30 = 1
a_31 = 1 -> t_31 = 1
a_32 = 1 -> t_32 = 1

```

El aprendizaje fue exitoso

## 5. Discusión de Resultados

Para cada uno de los resultados se muestra:

1. Los datos ingresados por el usuario

2. La tabla de verdad que contiene las entradas y su target de la siguiente manera:

```
model =  
P11 P12 ... P1n target1  
P21 P22 ... P2n target2  
.....  
Pm1 Pm2 ... Pmn targetm
```

Donde  $n$  es el tamaño del vector de entradas y  $m = 2^n$

3. Si la neurona está activada con base en el índice del modelo y el target
4. Si el aprendizaje fue exitoso o no

En la tabla de verdad los target se calcularon mediante la aplicación de la operación correspondiente (AND, OR o NOT), por ejemplo, para la AND se puede ver que el target es 1 solo cuando todas las entradas son 1.

El tamaño de las entradas se calculan con base en los pesos que ingresó el usuario, por ejemplo, si el usuario ingresa los pesos: 1 1 1 1, se generará una tabla de  $2^4$  filas y 5 columnas, donde las primeras 4 son las entradas y la última su target.

En cada caso se muestra que el aprendizaje fue exitoso y se puede comprobar con la salida del programa para cada índice.

## 6. Conclusiones

La célula de McCulloch-Pitts fue una muy importante aportación para las Ciencias de la Computación ya que fue la base para resolver problemas que no tenían solución, con base en la aportación se desarrollaron diferentes arquitecturas, cada una de ellas contiene como su unidad a la célula de McCulloch-Pitts. En esta práctica logré comprender el funcionamiento individual de la célula y su comportamiento como las tres compuertas lógicas básicas, al igual que aprendí a usar la plataforma MATLAB, no hay duda de que es una gran herramienta cualquier Académico.

## 7. Referencias

D. Michie, D.J. Spiegelhalter, C.C. Taylor (eds). Machine Learning, Neural and Statistical Classification, 1994.

<http://www.mind.ilstu.edu/curriculum/modOverview.php?modGUI=212>

## 8. Apéndice

Listing 1: Código

```
1 %Datos proporcionados por el usuario  
2 gate = input('Ingrese la compuerta (and, or, not): ', 's');  
3 tries = input('Ingrese el número de intentos: ');  
4 for i = 1:tries  
5     syn_prompt = 'Ingrese el valor de los pesos sinápticos separados por espacios(  
6         e.g. 1 2 3 4): ';  
7     w = str2num(strip(input(syn_prompt, 's')));  
     theta = input('Ingrese el valor del umbral: ');
```



```
8      if (gate == "not" && size(w, 2) > 1)
9          fprintf("Error, la compuerta NOT es de una sola entrada");
10     else
11         % Generación de la tabla de entradas y targets
12         model = logicalModel(size(w, 2), gate)
13         error = false;
14         % Época
15         for i = 1:size(model, 1)
16             row = model(i, :);
17             % Obtención de n
18             n = sum(row(1:end-1).*w);
19             % Obtención de a
20             if(n > theta); a_n = 1; else; a_n=0; end
21             % Comparación de a con el target
22             fprintf("a_ %i = %i -> t_ %i = %i\n", i, a_n, i, row(end));
23             if(a_n ~= row(end)); error = true; break; end
24         end
25         if(~error); fprintf("El aprendizaje fue exitoso\n");break; else
26             fprintf("El aprendizaje no fue exitoso\n"); end
27     end
28 end
29
30
31 function [table] = logicalModel(i, gate)
32 % logicalModel(I, gate) returns a matrix representing a truth table and
33 % the last column represents the output base on all the previous columns
34 % based on the (gate) parameter
35 % INPUT: (I) shall be an integer >= 1
36 % INPUT: (gate) shall be 'and' or 'or'
37 % OUTPUT: logicalModel is a binary matrix of size [2^I,I + 1]
38 % Heavily inspired in Paul Metcalf's CONDVECTS
39 % Acknowledgements: Paul Metcalf
40
41 g = 2;
42 i2 = 2^i;
43 table = false(i2,i + 1);
44 for m = 1 : 1 : i
45     m2 = 2^m;
46     m3 = (m2/2)-1;
47     i3 = i-m+1;
48     for g = g : m2 : i2
49         for k = 0 : 1 : m3
50             table(g+k,i3) = true;
51         end
52     end
53     g = m2+1;
54 end
55     if (gate == "and")
56         for row_index = 1:size(table, 1)
```

```
57         row = table(row_index,:);
58         res = row(1);
59         for e_index = 1:size(row, 2)-1
60             res = res & row(e_index);
61         end
62         table(row_index, end) = res;
63     end
64 elseif (gate == "or")
65     for row_index = 1:size(table, 1)
66         row = table(row_index,:);
67         res = row(1);
68         for e_index = 1:size(row, 2)-1
69             res = res | row(e_index);
70         end
71         table(row_index, end) = res;
72     end
73 elseif (gate == "not")
74     for row_index = 1:size(table, 1)
75         row = table(row_index,:);
76         res = ~row(1);
77         table(row_index, end) = res;
78     end
79 end
80 end
```