

# Práctica #2 Red de Hamming

Jorge Gómez Reus

## Índice

<b>1. Introducción</b>	<b>1</b>
1.1. Modelo . . . . .	2
<b>2. Diagrama de Flujo</b>	<b>3</b>
<b>3. Resultados</b>	<b>4</b>
3.1. Ejemplo 1 . . . . .	4
3.2. Ejemplo 2 . . . . .	5
<b>4. Discusión de Resultados</b>	<b>6</b>
<b>5. Conclusiones</b>	<b>6</b>
<b>6. Referencias</b>	<b>6</b>
<b>7. Apéndice</b>	<b>6</b>

## 1. Introducción

La red de Hamming es una red de tipo sin supervisión, su objetivo es calcular la distancia de Hamming entre los vectores prototipo y los de entrada para encontrar la distancia mínima, se conforma dos capas:

1. Capa FeedFoward Esta capa realiza el producto interno, entre cada uno de los prototipos y el patrón de entrada sumado con el vector de bias, después se le aplica una función de transferencia lineal.
2. Capa Recurrente Esta capa también es conocida como la capa competitiva, las neuronas de esta capa son inicializadas con las salidas de la capa feedforward, en esta capa las neuronas “compiten”, al terminar solo una neurona tendrá una salida diferente de zero, lo que indica en que categoría está el prototipo de entrada.

El objetivo de la red es decidir cual vector prototipo está más cercano al vector de entrada.

## 1.1. Modelo

Figura 1: Modelo

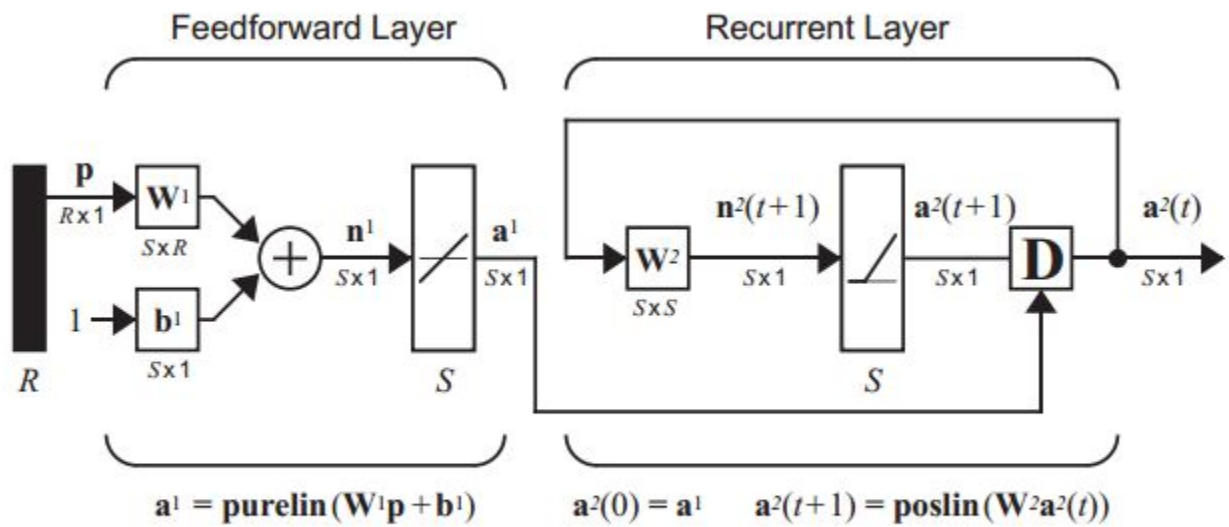


Figure 3.5 Hamming Network

## 2. Diagrama de Flujo

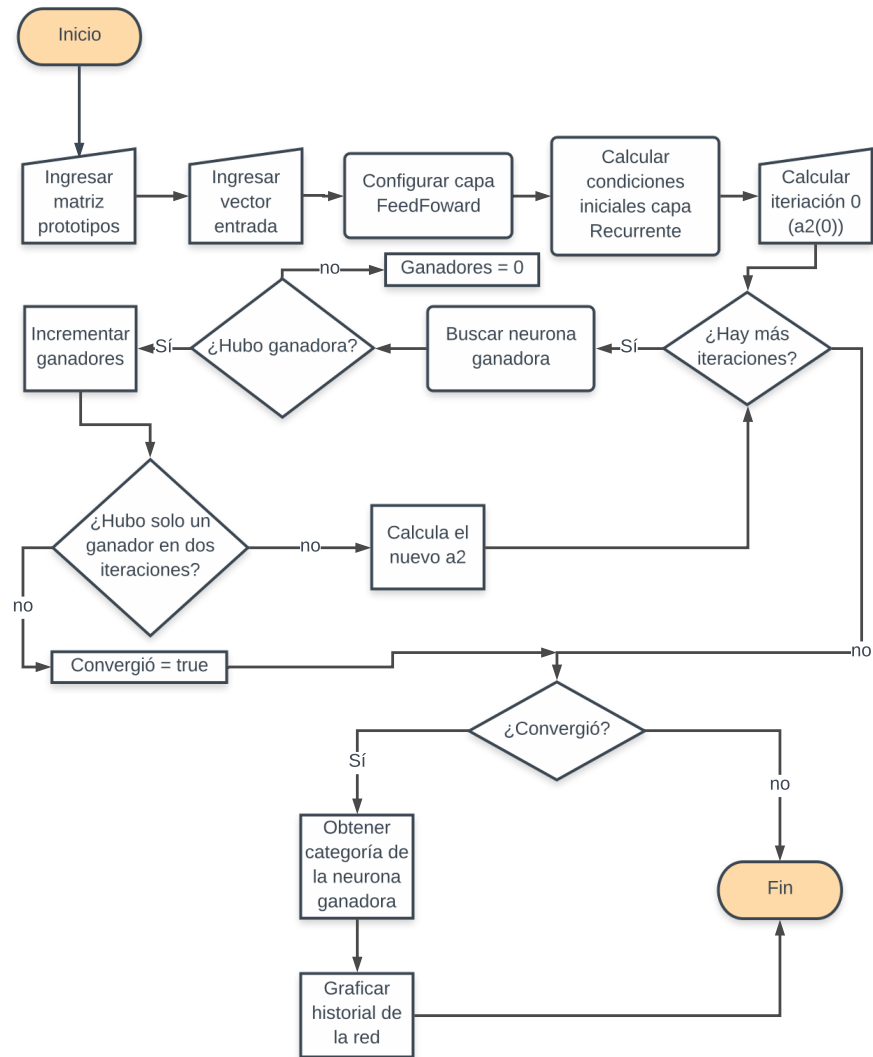


Figura 2: Diagrama de Flujo

### 3. Resultados

#### 3.1. Ejemplo 1

Ingrese la matriz de los valores prototipos.

Las columnas separados por espacios y las filas con ;

(e.g. 1 2 3;4 5 6;7 8 9)

```
>>> 1 -1 -1;1 1 -1
```

Ingrese el vector de entrada. Los valores separados por espacios (e.g. 1 2 3)

```
>>> -1 -1 -1
```

El vector de entrada pertenece a la clase: 1

Vector final:

3

0

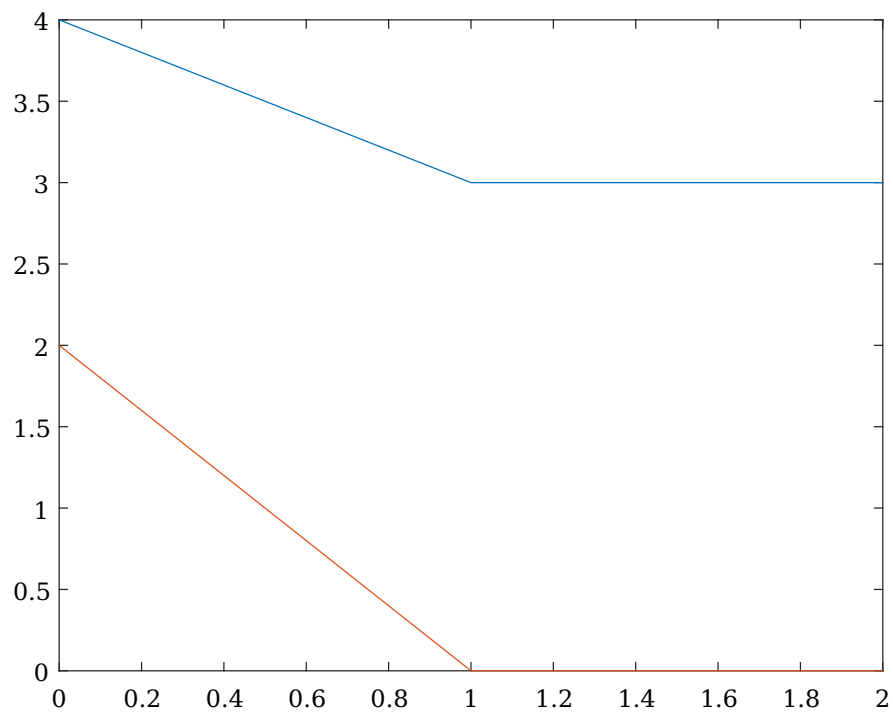


Figura 3: Gráfica del ejemplo 1

### 3.2. Ejemplo 2

Ingrese la matriz de los valores prototipos.

Las columnas separados por espacios y las filas con ;

(e.g. 1 2 3;4 5 6;7 8 9)

```
>>> 1 -1 -1 1;1 -1 1 -1;-1 1 1 -1;-1 1 -1 -1
```

Ingrese el vector de entrada. Los valores separados por espacios (e.g. 1 2 3)

```
>>> -.7 -.1 .2 -.5
```

El vector de entrada pertenece a la clase: 3

Vector final:

```
0
0
1.1904
0
```

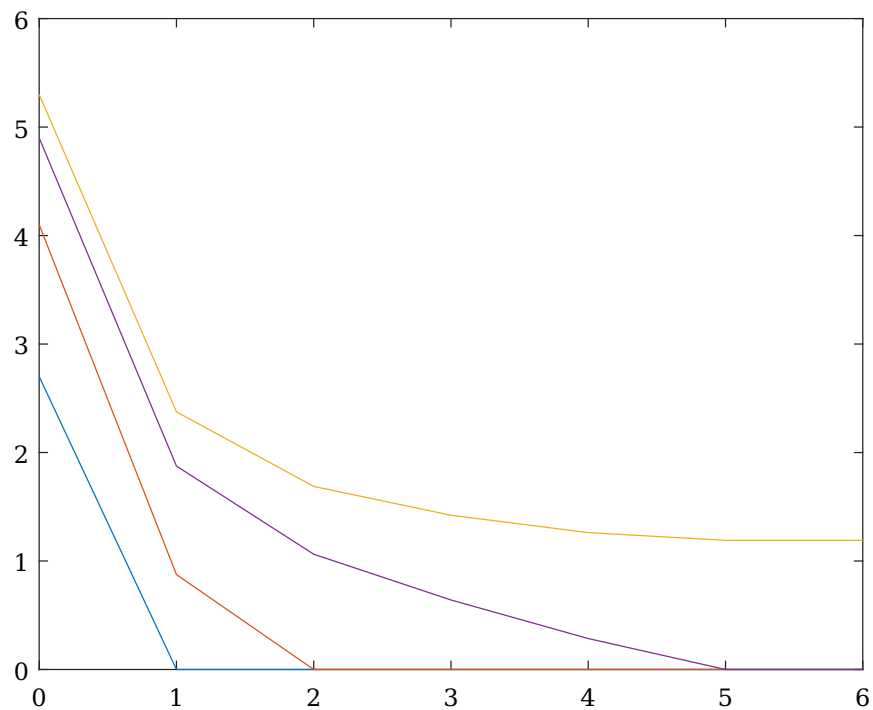


Figura 4: Gráfica del ejemplo 2

## 4. Discusión de Resultados

Para cada uno de los resultados se muestra:

1. A cual categoría más se acercó el vector de entrada (solo si convergió).
2. El vector final ( $a^2(t)$  donde  $0 \leq t \leq 40$ ) (solo si convergió).
3. La gráfica del "historial" de la evolución de  $a^2$  (solo si convergió).
4. Si convergió o no.

## 5. Conclusiones

La red de Hamming es una red interesante, ya que, usa dos capas(FF y recurrente) fue creada para separar patrones dado su vector de entrada, el número de neuronas son igual en las dos capas. Fue una gran aportación para el campo de las redes neuronales. Aprendí nuevas cosas investigando la teoría y práctica(matlab), fue divertida e interesante.

## 6. Referencias

Martin T Hagan. Machine Learning, Neural Network Design (2nd Edition), 2014.  
[http://home.agh.edu.pl/~vlsi/AI/hamming\\_en/](http://home.agh.edu.pl/~vlsi/AI/hamming_en/)

## 7. Apéndice

Listing 1: Código

```
1 % Datos ingresados por el usuario
2 proto_prompt = ['Ingrese la matriz de los valores prototipos. ' ...
3 'Las columnas separados por espacios y las filas con ; (e.g. 1 2 3;4 5 6;7 8 9)\n'];
4 cell_arr = regexp(input(proto_prompt, 's'), ';', 'split');
5 p = str2num(input('Ingrese el vector de entrada. Los valores separados por espacios (e
   .g. 1 2 3)\n', 's'));
6 % Capa Feed Foward
7 W1=[];
8 for i=1:length(cell_arr)
9     % Agregar la columna "parseada" nueva al final de la matriz
10    W1 = [W1; str2num(cell_arr{i})];
11 end
12 n = size(W1, 1);
13 b1 = ones(n,1) * size(W1, 2);
14 a1 = W1 * p + b1;
15 % Capa Recurrente
16 epsilon = 1 / n;
17 W2(1:n, 1:n)=-epsilon;
18 W2(1:n+1:end)=1;
19
20 % Condiciones Iniciales
```

```
21 a2 = a1;
22 h_values = []
23 h_values = [h_values;a1'];
24 a2 = poslin(W2*a2);
25 %Valores Extra
26 h_values = [h_values; a2'];
27 tries = 40;
28 winners = 0;
29 has_converged = false;
30 %Iteraciones
31 for i = 1:tries
32     has_winner = find_winner(a2);
33     if(has_winner)
34         winners = winners + 1;
35     else
36         has_winners = 0;
37     end
38     if(winners == 2)
39         has_converged = true;
40         break;
41     end
42     a2 = poslin(W2*a2);
43     h_values = [h_values; a2'];
44 end
45
46 if (has_converged)
47     for proto = 1:length(a2)
48         if (a2(proto) > 0)
49             fprintf('El vector de entrada pertenece a la clase: %d\n',
                    proto);
50             fprintf('Vector final:\n');
51             disp(a2);
52             break;
53         end
54     end
55     plot(0:i, h_values');
56 else
57     fprintf("La red no convergió\n");
58 end
59
60 %Función para encontrar las neuronas encendidas en el vector a
61 function [is_winner] = find_winner(vector)
62     counter = 0;
63     for i = 1:size(vector, 1)
64         if(vector(i) > 0)
65             counter = counter + 1;
66         end
67     end
68     if counter ~= 1
```

```
69         is_winner = false;
70     else
71         is_winner = true;
72     end
73 end
```