REDES ASOCIATIVAS

- El objetivo de estas redes es proporcionar asociaciones entre entradas y respuestas. Entradas similares a las del entrenamiento tienen que producir salidas que corresponden a la asociación del entrenamiento.

- Permiten resolver problemas de almacenamiento y recuperación de información basándose en el contenido en vez del direccionamiento.

- Las redes neuronales de memoria asociativa son redes de una sola capa que pueden almacenar un conjunto de asociaciones de patrones.

REDES ASOCIATIVAS

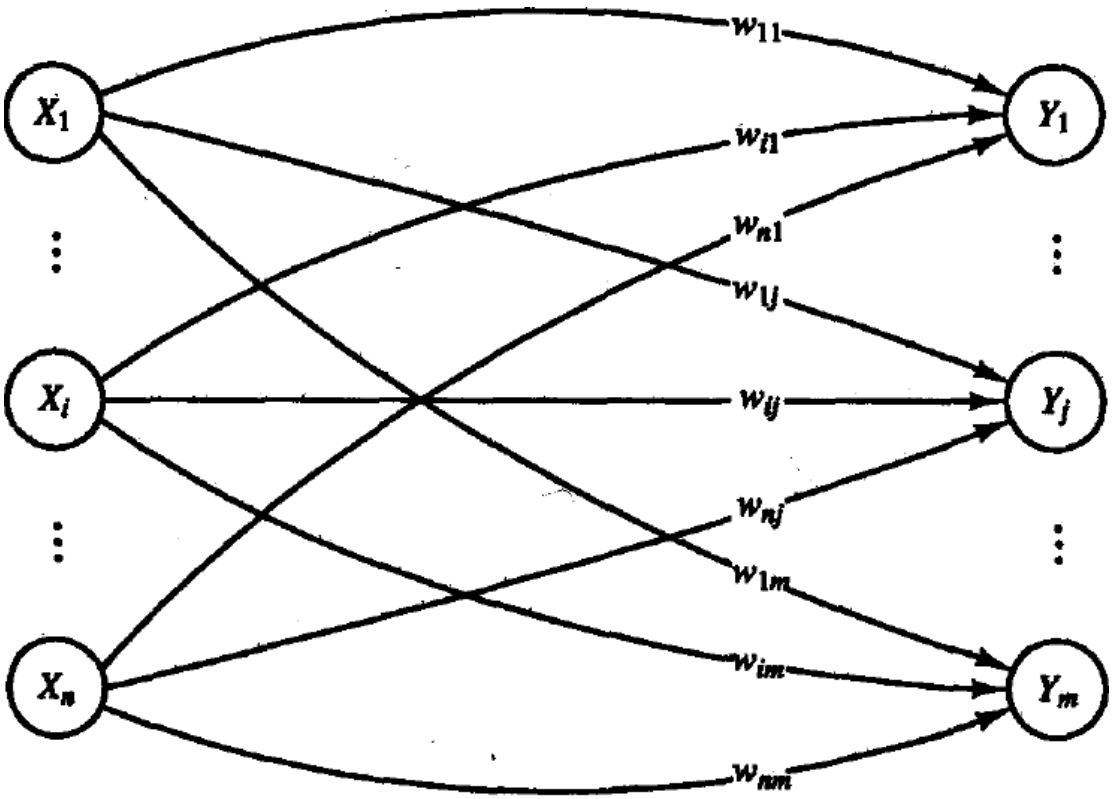
- Cada asociación consiste en un patrón de entrada salida **s:t**

- Si el vector **t** es el mismo que el vector **s** con el cual se asocia, entonces la red se llama autoasociativa.

- Si el vector **t** es distinto a **s** entonces la red se llama heteroasociativa.

- La arquitectura de las redes asociativas puede ser de dos tipos:
    1. De ***flujo directo o feedforward***: la información fluye de las neuronas de entrada a las de salida.
    2. **Recurrentes o interactivas**: las conexiones forman ciclos cerrados y la información fluye de forma recurrente.

Redes heteroasociativas de flujo directo



Neuronas de entrada pesos de las conexiones Neuronas de salida

Para el aprendizaje se puede utilizar la regla de Hebb o la regla delta.
Cada asociación es un par de vectores $(\mathbf{s}(p), \mathbf{t}(p))$, con $p=1,\ldots P$.

Redes heteroasociativas de flujo directo. Fase de explotación

Paso 0: Inicializar todos los pesos y sesgos utilizando la regla de Hebb o la regla delta

Paso 1: Para cada vector de entrada, ejecutar pasos 2-4

Paso 2: Establecer las activaciones a las neuronas de entrada

$x_i = s_i \quad (i=1 \ldots n)$

Paso 3: Calcular la entrada neta a las neuronas de salida:

$$y\_in_j = \sum_i x_i w_{ij}$$

Paso 4: Determinar la activación de cada neurona de salida (por ejemplo con alguna de las siguientes funciones de transferencia)::

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & y\_in_j = 0 \\ -1 & \text{si } y\_in_j < 0 \end{cases}$$

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & \text{si } y\_in_j \le 0 \end{cases}$$

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > \theta_j \\ y_j & \text{si } y\_in_j = \theta_j \\ -1 & \text{si } y\_in_j < \theta_j \end{cases}$$

Regla de Hebb para el aprendizaje

Paso 0: Inicializar todos los pesos a cero:

- $w_{ij} = 0, \quad i = 1,...,n \quad j = 1,...,m$

Paso 1: Para cada par *p* (1,...*P*) de vectores de entrenamiento **s,t**:

Paso 2: Establecer las activaciones de las unidades de entrada al vector de entrenamiento:

$$x_i = s_i \ (i=1,...,n)$$

Paso 3: Establecer las activaciones de las unidades de salida al vector objetivo:

$$y_j = t_j \ (j=1,...,m)$$

Paso 4: Ajustar los pesos (i=1,...n; j=1,...m):

$$w_{ij} \text{ (nuevo)} = w_{ij}\text{(anterior)} + x_i y_j$$

Hebb *a priori*: productos externos

- $\boldsymbol{s} = (s_1,...,s_i,...,s_n)$, $\mathbf{t} = (t_1,...,t_j,...,t_m)$
- La matriz para almacenar la asociación s:t, utilizando la regla de Hebb, es el producto exterior de $\boldsymbol{s}^t$ por $\mathbf{t}$:

$$\begin{bmatrix} s_1 \\ \vdots \\ s_i \\ \vdots \\ s_n \end{bmatrix} \begin{bmatrix} t_1 & \cdots & t_j & \cdots & t_m \end{bmatrix} = \begin{bmatrix} s_1t_1 & \cdots & s_1t_j & \cdots & s_1t_m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_it_1 & \cdots & s_it_j & \cdots & s_it_m \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_nt_1 & \cdots & s_nt_j & \cdots & s_nt_m \end{bmatrix}$$

- Para almacenar P patrones: $w_{ij} = \sum_{p=1}^{P} s_i(p)t_j(p)$

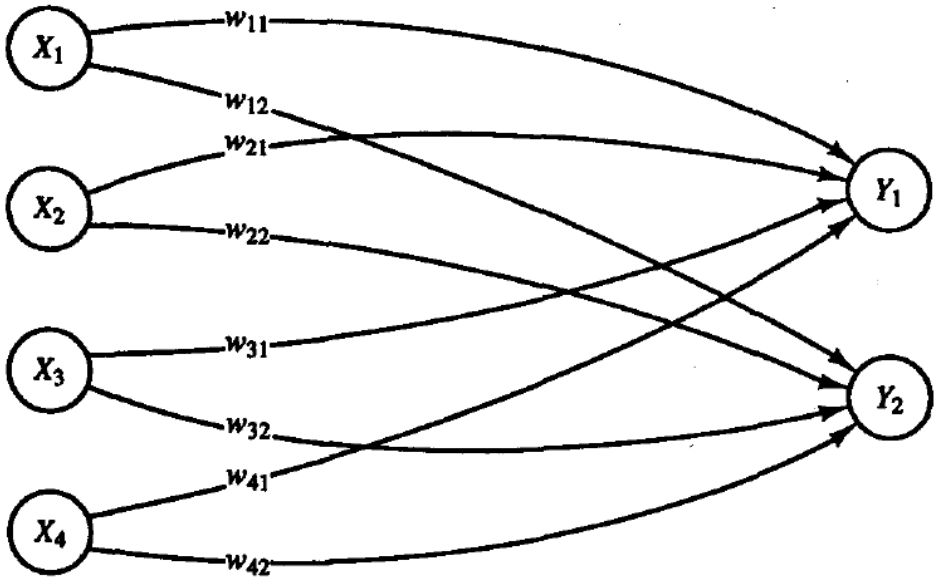
  en notación matricial:

$$\mathbf{W} = \sum_{p=1}^{P} \mathbf{s}^T(p)\mathbf{t}(p)$$

EJERCICIO

- Entrenar una red heteroasociativa con los siguientes patrones de entrada (codificación binaria):

Nª Patrón	s_1	s_2	s_3	s_4	t_1	t_2
1	1	0	0	0	1	0
2	1	1	0	0	1	0
3	0	0	0	1	0	1
4	0	0	1	1	0	1



Algoritmo de entrenamiento para el ejemplo (1/3)

Paso 0: Inicializar todos los pesos a cero:

w_{ij} =0, i = 1,...,n j = 1,...,m

Paso 1: Para cada par p (1,...P) de vectores de entrenamiento **s,t**:

Primer par: (1,0,0,0):(1,0)

Paso 2: x_1=1; x_2=x_3=x_4=0

Paso 3: y_1=1; y_2=0

Paso 4: w_{11}(nuevo) = w_{11}(anterior) + $x_1 y_1$ = 0+1=1

todos los demás pesos son cero (no cambian)

Algoritmo de entrenamiento para el ejemplo (2/3)

Segundo par: (1,1,0,0):(1,0)

Paso 2: $x_1=1$; $x_2=1$; $x_3=x_4=0$

Paso 3: $y_1=1$; $y_2=0$

Paso 4: $w_{11}(\text{nuevo}) = w_{11}(\text{anterior}) + x_1 y_1 = 1+1=2$

$w_{21}(\text{nuevo}) = w_{21}(\text{anterior}) + x_2 y_1 = 0+1=1$

todos los demás pesos son cero (no cambian)

Tercer par: (0,0,0,1):(0,1)

Paso 2: $x_1=x_2=x_3=0$; $x_4=1$

Paso 3: $y_1=0$; $y_2=1$

Paso 4: $w_{42}(\text{nuevo}) = w_{42}(\text{viejo}) + x_4 y_2 = 0+1=1$

todos los demás pesos son cero (no cambian)

Algoritmo de entrenamiento para el ejemplo (3/3)

Cuarto par: (0,0,1,1):(0,1)

Paso 2: $x_1=x_2=0;\ x_3=x_4=1$

Paso 3: $y_1=0;\ y_2=1$

Paso 4: $w_{32}(\text{nuevo}) = w_{32}(\text{anterior}) + x_3 y_2 = 0+1=1$

$w_{42}(\text{nuevo}) = w_{42}(\text{anterior}) + x_4 y_2 = 1+1=2$

todos los demás pesos son cero (no cambian)

La matriz de pesos queda:

$$\mathbf{W} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix}$$

Regla de Hebb a priori:

Cálculo de la matriz de pesos con el producto externo

Primer par: (1,1,0,0):(1,0):

$$\begin{pmatrix}1\\0\\0\\0\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix}=\begin{pmatrix}1 & 0\\0 & 0\\0 & 0\\0 & 0\end{pmatrix}$$

Segundo par: (1,1,0,0):(1,0):

$$\begin{pmatrix}1\\1\\0\\0\end{pmatrix}\begin{pmatrix}1 & 0\end{pmatrix}=\begin{pmatrix}1 & 0\\1 & 0\\0 & 0\\0 & 0\end{pmatrix}$$

Tercer par: (1,1,0,0):(1,0):

$$\begin{pmatrix}0\\0\\0\\1\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix}=\begin{pmatrix}0 & 0\\0 & 0\\0 & 0\\0 & 1\end{pmatrix}$$

Cuarto par: (1,1,0,0):(1,0):

$$\begin{pmatrix}0\\0\\1\\1\end{pmatrix}\begin{pmatrix}0 & 1\end{pmatrix}=\begin{pmatrix}0 & 0\\0 & 0\\0 & 1\\0 & 1\end{pmatrix}$$

La suma de las 4 matrices obtenidas es la matriz de pesos resultante del aprendizaje:

$$\mathbf{W}=\begin{pmatrix}2 & 0\\1 & 0\\0 & 1\\0 & 2\end{pmatrix}$$

Test con los patrones de entrenamiento (1/2)

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & \text{si } y\_in_j \le 0 \end{cases}$$

$$\mathbf{W} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix}$$

Primer par: (1,0,0,0):(1,0)

x=(1,0,0,0)

$y\_in_1 = x_1w_{11} + x_2w_{21} + x_3w_{31} + x_4w_{41} = 1(2)+0(1)+0(0)+0(0)=2 \rightarrow y_1=1$

$y\_in_2 = x_1w_{12} + x_2w_{22} + x_3w_{32} + x_4w_{42} = 1(0)+0(0)+0(1)+0(2)=0 \rightarrow y_2=0$ OK

Segundo par: (1,1,0,0):(1,0)

x=(1,1,0,0)

$y\_in_1 = x_1w_{11} + x_2w_{21} + x_3w_{31} + x_4w_{41} = 1(2)+1(1)+0(0)+0(0)=3 \rightarrow y_1=1$

$y\_in_2 = x_1w_{12} + x_2w_{22} + x_3w_{32} + x_4w_{42} = 1(0)+1(0)+0(1)+0(2)=0 \rightarrow y_2=0$ OK

Test con los patrones de entrenamiento (2/2)

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & \text{si } y\_in_j \le 0 \end{cases}$$

$$\mathbf{W} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix}$$

Tercer par: (0,0,0,1):(0,1)

x=(0,0,0,1)

$y\_in_1 = x_1w_{11} + x_2w_{21} + x_3w_{31} + x_4w_{41} = 0(2)+0(1)+0(0)+1(0)=0 \rightarrow y_1=0$

$y\_in_2 = x_1w_{12} + x_2w_{22} + x_3w_{32} + x_4w_{42} = 0(0)+0(0)+0(1)+1(2)=2 \rightarrow y_2=1$ OK

Cuarto par: (0,0,1,1):(0,1)

x=(0,0,1,1)

$y\_in_1 = x_1w_{11} + x_2w_{21} + x_3w_{31} + x_4w_{41} = 0(2)+0(1)+1(0)+1(0)=0 \rightarrow y_1=0$

$y\_in_2 = x_1w_{12} + x_2w_{22} + x_3w_{32} + x_4w_{42} = 0(0)+0(0)+1(1)+1(2)=3 \rightarrow y_2=1$ OK

Test con los patrones de entrenamiento notación vectorial

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & \text{si } y\_in_j \le 0 \end{cases}$$

$$\mathbf{W} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix}$$

Primer par: (1,0,0,0):(1,0)

x=(1,0,0,0)

$$(y\_in_1, y\_in_2) = \mathbf{xW} = \begin{pmatrix} 1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix} = \begin{pmatrix} 2 & 0 \end{pmatrix} \rightarrow \mathbf{y} = (1,0)$$ OK

Segundo par: (1,1,0,0):(1,0) $\begin{pmatrix} 1 & 1 & 0 & 0 \end{pmatrix}\mathbf{W} = \begin{pmatrix} 3 & 0 \end{pmatrix} \rightarrow \mathbf{y} = (1,0)$ OK

Tercer par: (0,0,0,1):(0,1) $\begin{pmatrix} 0 & 0 & 0 & 1 \end{pmatrix}\mathbf{W} = \begin{pmatrix} 0 & 2 \end{pmatrix} \rightarrow \mathbf{y} = (0,1)$ OK

Cuarto par: (0,0,1,1):(0,1) $\begin{pmatrix} 0 & 0 & 1 & 1 \end{pmatrix}\mathbf{W} = \begin{pmatrix} 0 & 3 \end{pmatrix} \rightarrow \mathbf{y} = (0,1)$ OK

Test con otros patrones

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & \text{si } y\_in_j \le 0 \end{cases}$$

$$\mathbf{W} = \begin{pmatrix} 2 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 2 \end{pmatrix}$$

Vectores de entrenamiento:

Primer par: (1,0,0,0):(1,0) Segundo par: (1,1,0,0):(1,0)

Tercer par: (0,0,0,1):(0,1) Cuarto par: (0,0,1,1):(0,1)

- Test con el patrón (0,1,0,0) que se diferencia del primer patrón en 1:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}\mathbf{W} = \begin{pmatrix} 1 & 0 \end{pmatrix} \rightarrow \mathbf{y} = (1,0)$$

- Test con el patrón (0,1,1,0) que no se parece a ninguno del entrenamiento:

$$\begin{pmatrix} 0 & 1 & 1 & 0 \end{pmatrix}\mathbf{W} = \begin{pmatrix} 1 & 1 \end{pmatrix} \rightarrow \mathbf{y} = (1,1)$$

La salida no es ninguna de las utilizadas en el entrenamiento: la red no reconoce el patrón

Otras codificaciones

- Conjunto de *P* patrones **s**(*p*):**t**(*p*), *p* = 1,...,*P* donde

$$\mathbf{s}(p) = (s_1(p),..., s_i(p),...,s_n(p))$$
$$\mathbf{t}(p) = (t_1(p),..., t_i(p),...,t_m(p))$$

  - Para almacenar patrones binarios con matriz de representación bipolar:

    Matriz de pesos $\mathbf{W}=\{w_{ij}\}$:

$$w_{ij} = \Sigma_p[2s_i(p)-1][2t_j(p)-1]$$

  - Para almacenar patrones bipolares:

    Matriz de pesos $\mathbf{W}=\{w_{ij}\}$:

$$w_{ij} = \Sigma_p s_i(p)t_j(p)$$

Otras codificaciones. Ejemplos:

– Para almacenar patrones binarios con matriz de representación bipolar:

Matriz de pesos $\mathbf{W}=\{w_{ij}\}$:

$$w_{ij} = \Sigma_p[2s_i(p)-1][2t_j(p)-1]$$

$$\mathbf{s}(1)=(1\quad 0\quad 0\quad 0)\rightarrow\ \mathbf{t}(1)=(1\quad 0)$$
$$\mathbf{s}(2)=(1\quad 1\quad 0\quad 0)\rightarrow\ \mathbf{t}(2)=(1\quad 0)$$
$$\mathbf{s}(3)=(0\quad 0\quad 0\quad 1)\rightarrow\ \mathbf{t}(3)=(0\quad 1)$$
$$\mathbf{s}(4)=(0\quad 0\quad 1\quad 1)\rightarrow\ \mathbf{t}(4)=(0\quad 1)$$

La matriz de pesos de representación bipolar queda:

$$\mathbf{W}=\begin{pmatrix} 4 & -4 \\ 2 & -2 \\ -2 & 2 \\ -4 & 4 \end{pmatrix}$$

Regla de Hebb a priori (bipolares):

Cálculo de la matriz de pesos con el producto externo

Primer par: (1,-1,-1,-1):(1,-1):

$$\begin{pmatrix}1\\-1\\-1\\-1\end{pmatrix}\begin{pmatrix}1 & -1\end{pmatrix}=\begin{pmatrix}1 & -1\\-1 & 1\\-1 & 1\\-1 & 1\end{pmatrix}$$

Segundo par: (1,1,-1,-1):(1,-1):

$$\begin{pmatrix}1\\1\\-1\\-1\end{pmatrix}\begin{pmatrix}1 & -1\end{pmatrix}=\begin{pmatrix}1 & -1\\1 & -1\\-1 & 1\\-1 & 1\end{pmatrix}$$

Tercer par: (-1,-1,-1,1):(-1,1):

$$\begin{pmatrix}-1\\-1\\-1\\1\end{pmatrix}\begin{pmatrix}-1 & 1\end{pmatrix}=\begin{pmatrix}1 & -1\\1 & -1\\1 & -1\\-1 & 1\end{pmatrix}$$

Cuarto par: (-1,-1,1,1):(-1,1):

$$\begin{pmatrix}-1\\-1\\1\\1\end{pmatrix}\begin{pmatrix}-1 & 1\end{pmatrix}=\begin{pmatrix}1 & -1\\1 & -1\\-1 & 1\\-1 & 1\end{pmatrix}$$

La suma de las 4 matrices obtenidas es la matriz de pesos resultante del aprendizaje:

$$\mathbf{W}=\begin{pmatrix}4 & -4\\2 & -2\\-2 & 2\\-4 & 4\end{pmatrix}$$

Otras codificaciones. Ejemplos:

– Para almacenar patrones bipolares:

Matriz de pesos $\mathbf{W}=\{w_{ij}\}$:

$$w_{ij} = \Sigma_p s_i(p) t_j(p)$$

$$\mathbf{s}(1) = (1 \quad -1 \quad -1 \quad -1) \rightarrow \mathbf{t}(1) = (1 \quad -1)$$
$$\mathbf{s}(2) = (1 \quad 1 \quad -1 \quad -1) \rightarrow \mathbf{t}(2) = (1 \quad -1)$$
$$\mathbf{s}(3) = (-1 \quad -1 \quad -1 \quad 1) \rightarrow \mathbf{t}(3) = (-1 \quad 1)$$
$$\mathbf{s}(4) = (-1 \quad -1 \quad 1 \quad 1) \rightarrow \mathbf{t}(4) = (-1 \quad 1)$$

La matriz de pesos de representación bipolar queda:

$$\mathbf{W} = \begin{pmatrix} 4 & -4 \\ 2 & -2 \\ -2 & 2 \\ -4 & 4 \end{pmatrix}$$

Test con otros patrones (codificación bipolar)

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & y\_in_j = 0 \\ -1 & \text{si } y\_in_j < 0 \end{cases}$$

$$\mathbf{W} = \begin{pmatrix} 4 & -4 \\ 2 & -2 \\ -2 & 2 \\ -4 & 4 \end{pmatrix}$$

Vectores de entrenamiento:

Primer par: (1,-1,-1,-1):(1,-1)

Segundo par: **(1,1,-1,-1):(1,-1)**

Tercer par: (-1,-1,-1,1):(-1,1)

Cuarto par: (-1,-1,1,1):(-1,1)

- Test con el patrón (0,1,0,-1) que tiene dos componentes “desconocidas”:

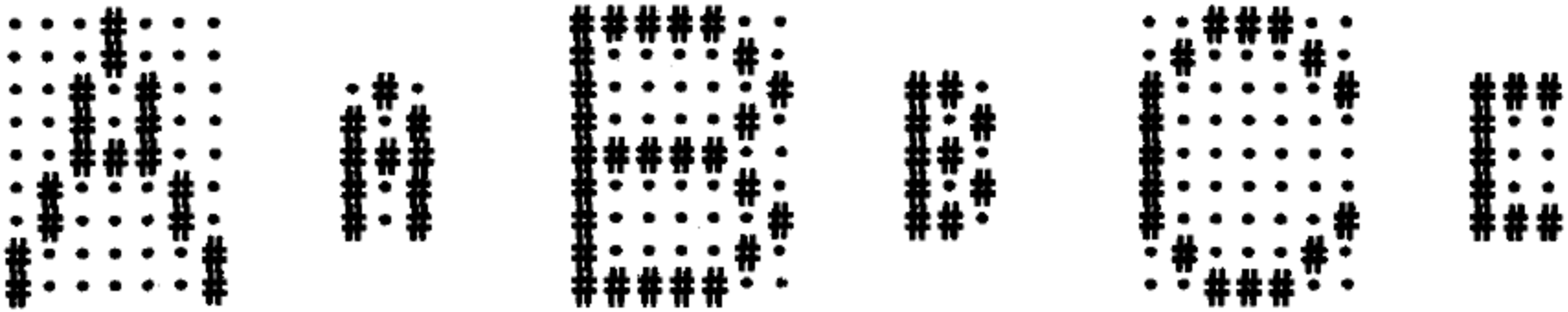
$$(0 \quad 1 \quad 0 \quad -1)\mathbf{W} = (6 \quad -6) \rightarrow \mathbf{y} = (1,-1)$$

- Test con el patrón (-1,1,1,-1) con dos fallos:

$$(-1 \quad 1 \quad 1 \quad -1)\mathbf{W} = (0 \quad 0) \rightarrow \mathbf{y} = (0,0)$$

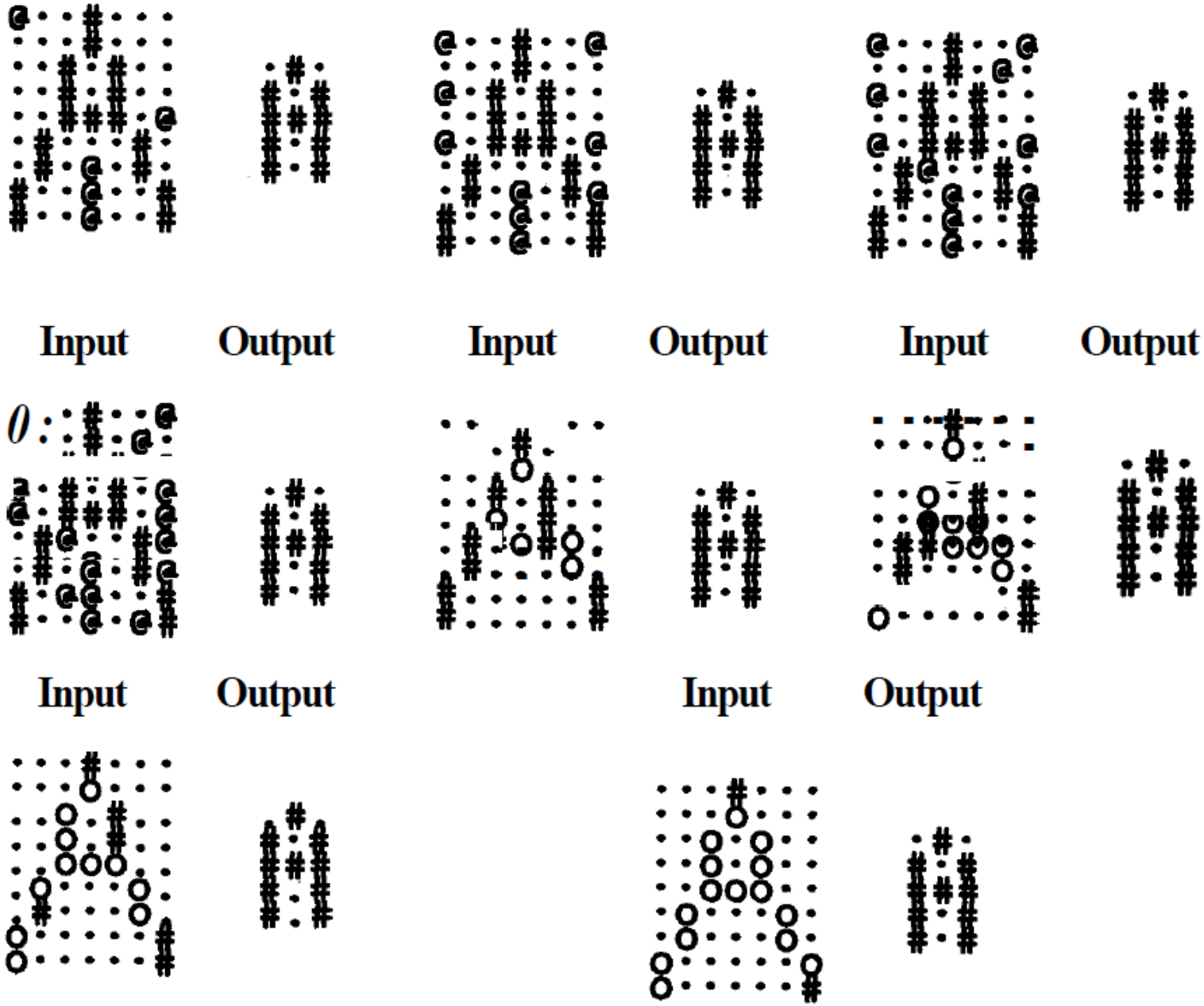
La salida no es ninguna de las utilizadas en el entrenamiento: la red no reconoce el patrón

Red heteroasociativa para reconocer letras

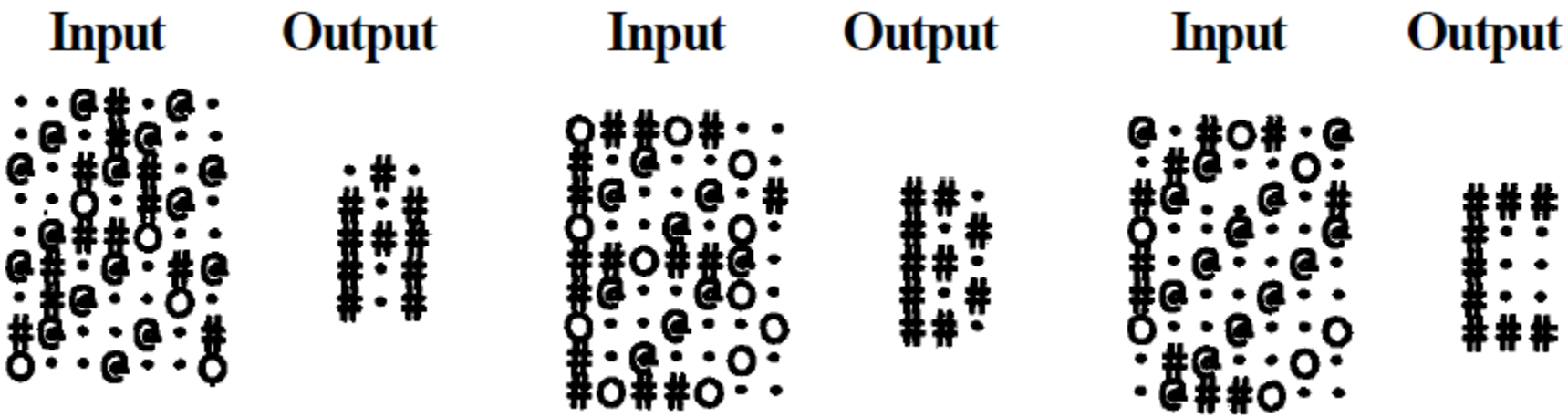


- Los patrones de entradas tienen 9x7 componentes, los de salida 5x3
- Codificación bipolar

Respuesta a patrones con ruido



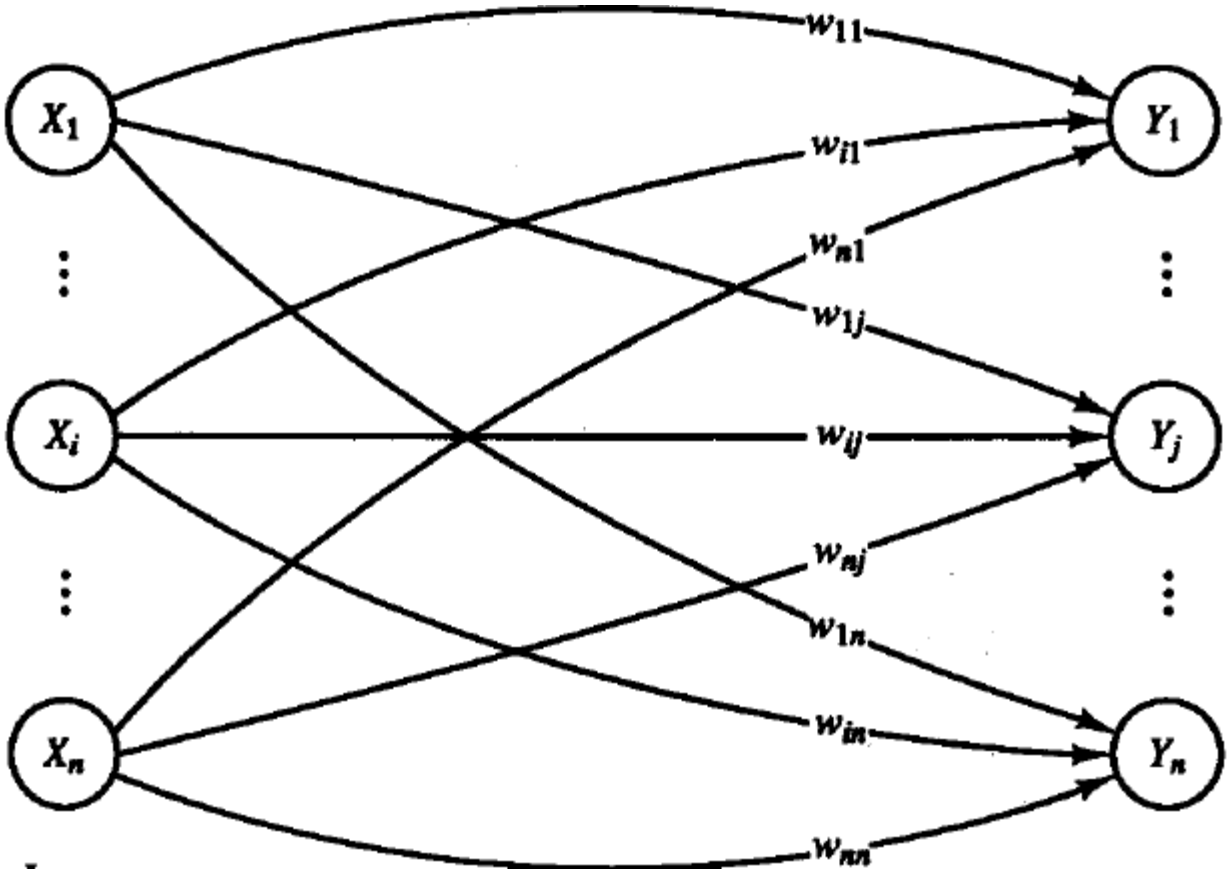
Respuesta a patrones con fallos



- Se reconocen patrones incluso con el 30% de ruido

- Si los patrones de entrada son ortogonales (descorrelacionados), la regla de Hebb producirá pesos correctos. Si no, la respuesta puede incluir una parte de cada uno de las salidas objetivo.

Redes Autoasociativas



Neuronas de entrada pesos de las conexiones Neuronas de salida

- Se trata de un caso especial de las redes heteroasociativas cuando los patrones de entrenamiento de entrada y salida son idénticos.
- El proceso de entrenamiento se llama de almacenamiento de los patrones.
- Un patrón almacenado se puede recuperar con una entrada distorsionada.

Regla de Hebb para el aprendizaje

Paso 0: Inicializar todos los pesos a cero:

$w_{ij} = 0, \quad i = 1,...,n \quad j = 1,...,n$

Paso 1: Para cada par *p* (1,...*P*) de vectores de entrenamiento **s,t**:

Paso 2: Establecer las activaciones de las unidades de entrada al vector de entrenamiento:

$x_i = s_i \ (i=1,...,n)$

Paso 3: Establecer las activaciones de las unidades de salida al vector objetivo:

$y_j = t_j \ (j=1,...,n)$

Paso 4: Ajustar los pesos (i=1,...n; j=1,...n):

$w_{ij}\ (\text{nuevo}) = w_{ij}(\text{anterior}) + x_i y_j$

ALTERNATIVA:

Producto externo y suma

$$\mathbf{W} = \sum_{p=1}^{P} \mathbf{s}^t(p)\mathbf{s}(p)$$

Redes autoasociativas. Fase de explotación

Paso 0: Inicializar todos los pesos y sesgos con la regla de Hebb o el producto externo.

Paso 1: Para cada vector de entrada, ejecutar pasos 2-4

Paso 2: Establecer las activaciones a las neuronas de entrada

$$x_i = s_i \quad (i=1\ldots n)$$

Paso 3: Calcular la entrada neta a las neuronas de salida:

$$y\_in_j = \sum_i x_i w_{ij}$$

Paso 4: Determinar la activación de cada neurona de salida (por ejemplo con alguna de las siguientes funciones de transferencia)::

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ 0 & y\_in_j = 0 \\ -1 & \text{si } y\_in_j < 0 \end{cases}$$

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 0 \\ -1 & \text{si } y\_in_j \le 0 \end{cases}$$

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > \theta_j \\ y_j & \text{si } y\_in_j = \theta_j \\ -1 & \text{si } y\_in_j < \theta_j \end{cases}$$

Ejemplos: almacenamiento de un vector en una red autoasociativa

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix}$$

– Comprobación de la recuperación del vector:

$$(1 \quad 1 \quad 1 \quad -1)\mathbf{W} = (4 \quad 4 \quad 4 \quad -4) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

Ejemplos: comprobación con vectores con errores en una red autoasociativa

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix}$$

– Comprobación de la recuperación del vector con entradas con errores:

$$(-1 \quad 1 \quad 1 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad -1 \quad 1 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 1 \quad -1 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 1 \quad 1 \quad 1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

Ejemplos: comprobación con vectores con componentes desconocidas

- La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix}$$

- Comprobación de la recuperación del vector con entradas con una componente desconocida (0):

$$(0 \quad 1 \quad 1 \quad -1)\mathbf{W} = (3 \quad 3 \quad 3 \quad -3) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 0 \quad 1 \quad -1)\mathbf{W} = (3 \quad 3 \quad 3 \quad -3) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 1 \quad 0 \quad -1)\mathbf{W} = (3 \quad 3 \quad 3 \quad -3) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 1 \quad 1 \quad 0)\mathbf{W} = (3 \quad 3 \quad 3 \quad -3) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

Ejemplos: comprobación con vectores con componentes desconocidas

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix}$$

– Comprobación de la recuperación del vector con entradas con 2 ausencias:

$$(0 \quad 0 \quad 1 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$
$$(0 \quad 1 \quad 0 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$
$$(0 \quad 1 \quad 1 \quad 0)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$
$$(1 \quad 0 \quad 0 \quad -1)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$
$$(1 \quad 0 \quad 1 \quad 0)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$
$$(1 \quad 1 \quad 0 \quad 0)\mathbf{W} = (2 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

Ejemplos: comprobación con vectores con dos errores

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix} \qquad y_j = \begin{cases} 1 \text{ si } y\_in_j > 0 \\ -1 \text{ si } y\_in_j \leq 0 \end{cases}$$

– Comprobación de la recuperación del vector con entradas con 2 errores:

$$(-1 \quad -1 \quad 1 \quad -1)\mathbf{W} = (0 \quad 0 \quad 0 \quad 0) \rightarrow (-1 \quad -1 \quad -1 \quad -1)$$

La red no reconoce este vector de entrada.

Red autoasociativa sin autoconexiones

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ sin autoconexiones es:

$$\mathbf{W}_0 = \begin{pmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{pmatrix}$$

– Comprobación de la recuperación del vector con entradas con 1 error:

$$(1 \quad -1 \quad 1 \quad -1)\mathbf{W}_0 = (1 \quad 1 \quad 1 \quad -1) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

– Comprobación de la recuperación del vector con entradas con 2 ausencias:

$$(0 \quad 0 \quad 1 \quad -1)\mathbf{W}_0 = (2 \quad 2 \quad 1 \quad -1) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

$$(1 \quad 0 \quad 1 \quad 0)\mathbf{W}_0 = (1 \quad 2 \quad 1 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

La red reconoce estos vectores.

Ejemplo de almacenamiento de dos vectores en una red autoasociativa

- Se quiere almacenar (1,1,-1,-1) y (-1,1,1,-1)
- La matriz de pesos resultante será la suma de las dos matrices para cada uno de los vectores, con los pesos de la diagonal establecidos a cero (si no los pesos de la diagonal –establecidos al nº de vectores almacenados) dominarían reproduciendo el vector de entrada siempre).

$$\underset{\mathbf{W}_1}{\begin{pmatrix} 0 & 1 & -1 & -1 \\ 1 & 0 & -1 & -1 \\ -1 & -1 & 0 & 1 \\ -1 & -1 & 1 & 0 \end{pmatrix}} + \underset{\mathbf{W}_2}{\begin{pmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 0 \end{pmatrix}} = \underset{\mathbf{W}_1 + \mathbf{W}_2}{\begin{pmatrix} 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \\ -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \end{pmatrix}}$$

- Ejercicio: comprobar que la red reconoce los dos vectores de entrenamiento.

Ejemplo de almacenamiento de dos vectores no-ortogonales en una red autoasociativa

- Se quiere almacenar (1,-1,-1,1) y (1,1,-1,1), no ortogonales
- La matriz de pesos resultante será la suma de las dos matrices para cada uno de los vectores, con los pesos de la diagonal establecidos a cero:

$$\underset{\mathbf{W}_1}{\begin{pmatrix} 0 & -1 & -1 & 1 \\ -1 & 0 & 1 & -1 \\ -1 & 1 & 0 & -1 \\ 1 & -1 & -1 & 0 \end{pmatrix}} + \underset{\mathbf{W}_2}{\begin{pmatrix} 0 & 1 & -1 & 1 \\ 1 & 0 & -1 & 1 \\ -1 & -1 & 0 & -1 \\ 1 & 1 & -1 & 0 \end{pmatrix}} = \underset{\mathbf{W}_1+\mathbf{W}_2}{\begin{pmatrix} 0 & 0 & -2 & 2 \\ 0 & 0 & 0 & 0 \\ -2 & 0 & 0 & -2 \\ 2 & 0 & -2 & 0 \end{pmatrix}}$$

- Ejercicio: comprobar que la red no es capaz de distinguir los dos vectores de entrenamiento.

Recordatorio: dos vectores $\mathbf{x}$ e $\mathbf{y}$ son ortogonales si $\mathbf{x}\mathbf{y}^t=\Sigma_i x_i y_i=0$

Ejemplo de almacenamiento de tres vectores ortogonales en una red autoasociativa

- Se quiere almacenar (1,1,-1,-1), (-1,1,1,-1) y (-1,1,-1,1), ortogonales.
- La matriz de pesos resultante será la suma de las tres matrices para cada uno de los vectores, con los pesos de la diagonal establecidos a cero:

$$\underset{\mathbf{W}_1+\mathbf{W}_2}{\begin{pmatrix} 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & -2 \\ -2 & 0 & 0 & 0 \\ 0 & -2 & 0 & 0 \end{pmatrix}} + \underset{\mathbf{W}_3}{\begin{pmatrix} 0 & -1 & 1 & -1 \\ -1 & 0 & -1 & 1 \\ 1 & -1 & 0 & -1 \\ -1 & 1 & -1 & 0 \end{pmatrix}} = \underset{\mathbf{W}_1+\mathbf{W}_2+\mathbf{W}_3}{\begin{pmatrix} 0 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 \\ -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{pmatrix}}$$

- Ejercicio: comprobar que la red es capaz de distinguir los tres vectores de entrenamiento.

Ejemplo de almacenamiento de 4 vectores ortogonales en una red autoasociativa

- Se quiere almacenar (1,1,-1,-1), (-1,1,1,-1), (-1,1,-1,1), (1,1,1,1) ortogonales.
- La matriz de pesos resultante será la suma de las tres matrices para cada uno de los vectores, con los pesos de la diagonal establecidos a cero:

$$\begin{matrix} \mathbf{W}_1+\mathbf{W}_2+\mathbf{W}_3 & & \mathbf{W}_4 & & \mathbf{W}_1+\mathbf{W}_2+\mathbf{W}_3+\mathbf{W}_4 \end{matrix}$$

$$\begin{pmatrix} 0 & -1 & -1 & -1 \\ -1 & 0 & -1 & -1 \\ -1 & -1 & 0 & -1 \\ -1 & -1 & -1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

que no puede reconocer ningún vector.

Capacidad de almacenamiento en las redes autoasociativas

- Una red autoasociativa con 4 neuronas puede almacenar tres vectores ortogonales. Sin embargo la matriz de pesos para 4 vectores ortogonales es siempre singular, y por tanto la red no puede almacenar los 4 vectores.

- La capacidad de una red autoasociativa depende del número de componentes que tengan los vectores y de las relaciones entre los vectores: se puede almacenar más vectores si son ortogonales mutuamente.

- Teorema: se puede probar que n-1 vectores bipolares, cada uno con n componentes se pueden siempre almacenar utilizando la suma de las matrices obtenidas con los productos externos de cada vector (con la diagonal a cero).

Redes autoasociativas iterativas

– La matriz de pesos $\mathbf{W}=\{w_{ij}\}$ para almacenar $\mathbf{s} = (1,1,1,-1)$ es:

$$\mathbf{W} = \begin{pmatrix} 0 & 1 & 1 & -1 \\ 1 & 0 & 1 & -1 \\ 1 & 1 & 0 & -1 \\ -1 & -1 & -1 & 1 \end{pmatrix}$$

– Comprobación de la recuperación del vector con entradas con 2 componentes desconocidas: (1,0,0,0):

$$(1 \quad 0 \quad 0 \quad 0)\mathbf{W} = (0 \quad 1 \quad 1 \quad -1) \rightarrow \text{iterar}$$

$$(0 \quad 1 \quad 1 \quad -1)\mathbf{W} = (3 \quad 2 \quad 2 \quad -2) \rightarrow (1 \quad 1 \quad 1 \quad -1)$$

La red reconoce el vector en la segunda iteración.

En este caso las neuronas son a la vez de entrada y de salida.

Autoasociador lineal recurrente

- Tipo de red autoasociativa iterativa con n neuronas conectadas todas entre sí.
- La matriz de pesos es simétrica proporcionada por la ley de Hebb.
- Su comportamiento se puede analizar con conceptos básicos de álgebra lineal: Una matriz simétrica no singular de $n \times n$ elementos tienen n autovectores mutuamente ortogonales.

Brain-State-in-a-Box

- Tipo de red autoasociativa iterativa n neuronas conectadas todas entre sí.
- Las unidades de esta red actualizan sus activaciones simultáneamente (como el autoasociador lineal).
- Evita los problemas del autoasociador linear recurrente respecto al crecimiento de la respuesta mediante una función de activación que toma valores dentro de un cubo (cada componente se restringe entre -1 y 1.
- En esta red hay autoconexiones (los pesos de la diagonal principal no son cero) .

Brain-State-in-a-Box

Paso 0: Inicializar todos los pesos a valores pequeños aleatorios, inicializar las tasas de aprendizaje α y β.

Paso 1: Para cada vector de entrada, ejecutar pasos 2-6

Paso 2: Establecer las activaciones a los valores de entrada

$$y_i = x_i \quad (i=1 \ldots n)$$

Paso 3: Mientras cambien las activaciones ejecutar los pasos 4 y 5:

Paso 4: Calcular la entrada neta a las neuronas:

$$y\_in_i = y_i + \alpha \sum_j y_j w_{ji}$$

Paso 5: Determinar la activación de cada neurona:

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > 1 \\ y\_in_i & \text{si } -1 \le y\_in_i \le 1 \\ -1 & \text{si } y\_in_i < -1 \end{cases}$$

Paso 6: Actualizar los pesos: $w_{ij}(\text{nuevo}) = w_{ji}(\text{anterior}) + \beta y_i y_j$