

# John J. Hopfield

- J.J. Hopfield después de preguntarse (1982):
  - Los procesos de memoria y generalización en el cerebro, ¿son propiedades emergentes y de origen colectivo tal como los fenómenos que se observan en algunos medios físicos?
- ... demuestra que:
  - propiedades computacionales importantes surgen espontáneamente en un sistema de neuronas interconectadas. Existe un formalismo para analizarlas por analogía a los sistemas físicos.

# Red de Hopfield de estados discretos

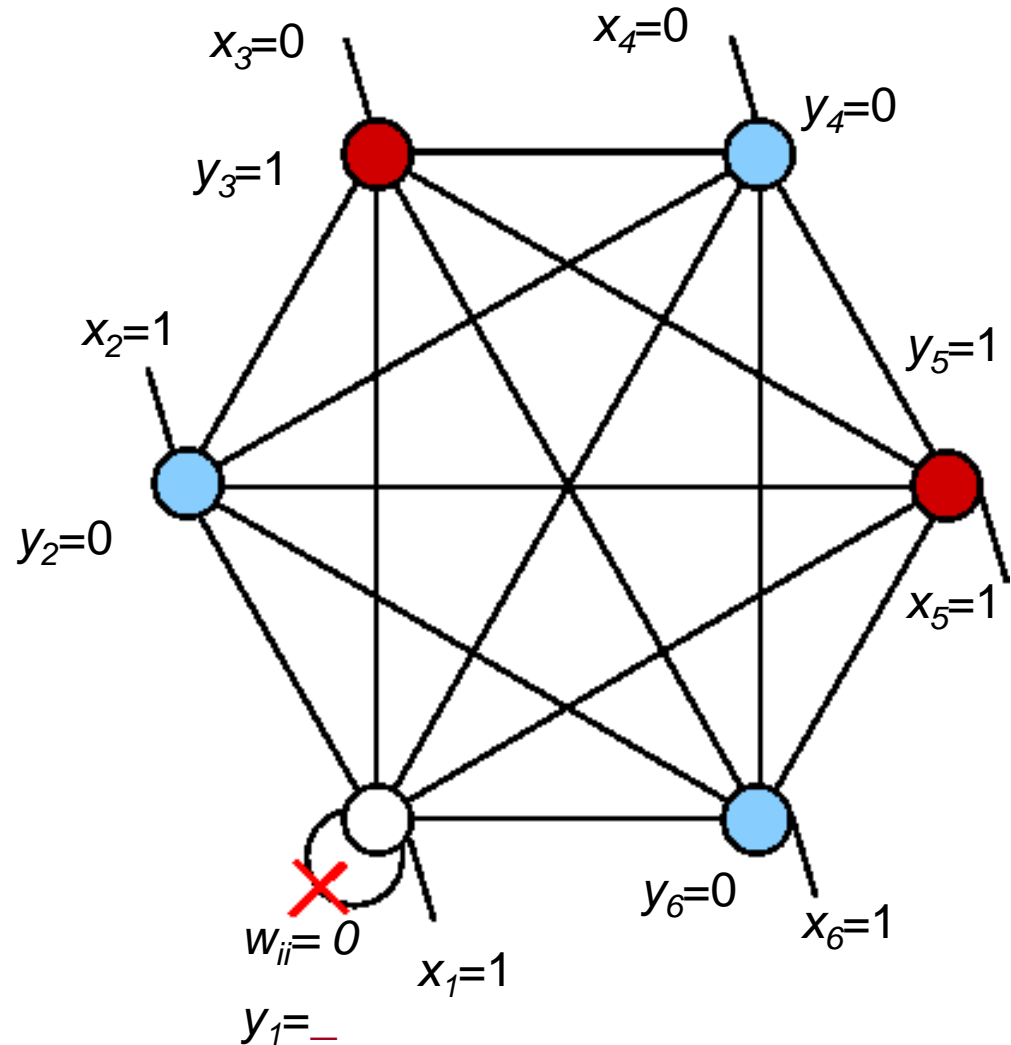
- Red autoasociativa con conexión total (todas con todas).
- La red tiene pesos simétricos:  $w_{ij}=w_{ji}$ .
- No hay autoconexiones:  $w_{ii}=0$ .
- Sólo una unidad  $i$  actualiza su activación  $y_i$  en cada unidad de tiempo (asíncrona).
- El estado de cada neurona, su activación, puede ser:  $y_i = 0$  (no dispara) ó  $y_i = 1$  (dispara).
- Cada unidad recibe una señal externa  $x_i$  además de las señales de las otras neuronas.
- Iteración hasta la convergencia.

# Red de Hopfield

Vector de entrada:  
 $\mathbf{X} = (1, 1, 0, 1, 1)$

Vector de activaciones:  
 $\mathbf{y} = (\_, 0, 1, 0, 1, 0)$

- Una única capa
- Conexión total salvo a sí mismas
- Actualización asíncrona
- Input presente en todos los ciclos



# Algoritmo

- {1} Hallar la matriz de pesos  $w_{ij}$  (Hebb).
- {2} Para cada vector de entrada  $\mathbf{x}$ :
  - {3} Mientras las activaciones cambien:
    - {4} En el primer ciclo:  $y_i = x_i, \forall i$
    - {5} Para cada unidad  $i$  (elegida aleatoriamente):
      - {6}  $input_i = x_i + \sum_j y_j w_{ij}$
      - {7}  $y_i = \begin{cases} 1 & \text{si } input_i > \theta_i \\ y_i & \text{si } input_i = \theta_i \\ 0 & \text{si } input_i < \theta_i \end{cases}$
      - {8} Enviar el valor de  $y_i$  a todas las unidades

# Variaciones en el algoritmo

- **Necesario** (para asegurar la convergencia):
  - El orden de actualización de las unidades es aleatorio pero es necesario actualizar sólo una unidad por ciclo.
  - $w_{ii} = 0$
- **Opcional:**
  - El umbral  $\theta_i = 0$
  - Activaciones binarias (1,0) , sin input externo  $x_i$
  - Activaciones bipolares (-1,1), sin input externo  $x_i$

# Regla de Hebb para el aprendizaje

{1} Inicializar todos los pesos a cero:

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

{2} Para cada par  $p$  ( $1, \dots, P$ ) de vectores de entrenamiento  $\mathbf{s}, \mathbf{t}$ :

{3} Establecer las activaciones de las unidades de entrada al vector de entrenamiento:

$$x_i = s_i \quad (i=1, \dots, n)$$

{4} Establecer las activaciones de las unidades de salida al vector objetivo:

$$y_j = t_j \quad (j=1, \dots, n)$$

{5} Ajustar los pesos ( $i=1, \dots, n; j=1, \dots, n$ ):

$$w_{ij} = w_{ij} + x_i y_j$$

# Regla *hebbiana* (no supervisada)

- Conjunto de  $P$  patrones  $s(p)$ ,  $p = 1, \dots, P$  donde

$$s(p) = (s_1(p), \dots, s_i(p), \dots, s_n(p))$$

- Para almacenar patrones binarios:

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

$$w_{ii}=0, \quad w_{ij} = \sum_p [2s_i(p)-1][2s_j(p)-1] \quad \forall i \neq j$$

- Para almacenar patrones bipolares:

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

$$w_{ii}=0, \quad w_{ij} = \sum_p s_i(p)s_j(p) \quad \forall i \neq j$$

# Hebb *a priori*: productos externos

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

$$\begin{bmatrix} s_1 \\ \vdots \\ s_i \\ \vdots \\ s_n \end{bmatrix} \begin{bmatrix} t_1 & \cdots & t_j & \cdots & t_n \end{bmatrix} = \begin{bmatrix} s_1 t_1 & \cdots & s_1 t_j & \cdots & s_1 t_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_i t_1 & \cdots & s_i t_j & \cdots & s_i t_n \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ s_n t_1 & \cdots & s_n t_j & \cdots & s_n t_n \end{bmatrix}$$

- Para almacenar  $P$  patrones ( $p=1 \dots P$ ):  $\mathbf{W} = \sum_p \mathbf{s}^t(p) \mathbf{t}(p)$



# Función de Lyapunov

- Es posible demostrar que la red de Hopfield converge a un patrón estacionario.
- Formulación hamiltoniana de la red: función de energía o función de Lyapunov:

$$E = -1/2 \sum_{i \neq j} \sum_j (y_i y_j w_{ij}) - \sum_i (x_i y_i) + \sum_i (\theta_i y_i)$$

Si la activación cambia en una cantidad  $\Delta y_j$ :

$$\Delta E = - [\sum_j (y_j w_{ij}) + x_i - \theta_i] \Delta y_j$$

(sólo una unidad cambia su activación en cada paso)

# La energía no puede aumentar

$$\Delta E = - [\sum_j (y_j w_{ij}) + x_i - \theta_i] \Delta y_j$$

- Si  $y_j > 0$ , cambiará a cero si  $input_i = (x_i + \sum_j y_j w_{ij}) < \theta_i$

$\Delta y_j < 0$ , y por tanto:  $\Delta E < 0$ .

- Si  $y_j = 0$ , cambiará a positivo si  $(x_i + \sum_j y_j w_{ij}) > \theta_i$

$\Delta y_j > 0$ , y por tanto:  $\Delta E < 0$

# La red alcanza un equilibrio

$$\Delta y_j > 0 \text{ sólo si } [\sum_j (y_j w_{ij}) + x_i - \theta_i] > 0$$

$$\Delta y_j < 0 \text{ sólo si } [\sum_j (y_j w_{ij}) + x_i - \theta_i] < 0$$

Por tanto,  $\Delta E = - [\sum_j (y_j w_{ij}) + x_i - \theta_i] \Delta y_j$  no puede ser  $> 0$

- La energía no puede aumentar, y puesto que está acotada, debe alcanzar un estado estable de equilibrio que no cambia con las siguientes iteraciones.
- La demostración no depende de que las activaciones  $y_j$  sean binarias, ni de elecciones precisas de  $x_i$  ó  $\theta_i$ .

# Capacidad de almacenamiento

- J. Hopfield estimó numéricamente la capacidad de este tipo de red (número de patrones binarios que se pueden almacenar y recuperar con precisión):

$$P \approx 0.15 N$$

siendo  $N$  el número de neuronas en la red

- Límite superior demostrado analíticamente por Abu-Mostafa y St. Jacques (1985):

$$P < N$$

- McEliece (1987) “capacidad suavizada”:  $P < N / 4 \ln(N)$

# Ejemplo de evolución

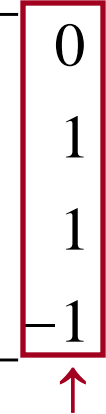
- Se desea una red de Hopfield que almacene el vector (1,1,1,0).
- Una vez entrenada la red se probará con un vector con dos errores (0,0,1,0)
- La matriz de pesos (bipolar) será:

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

# Evolución de la red (I)

- El vector de entrada es  $\mathbf{x} = (0, 0, 1, 0)$ .

Para este vector:

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


- $\mathbf{y} = (0, 0, 1, 0)$
- **Se elige la unidad 1** para actualizar su activación:
  - $input_1 = x_1 + \sum_j y_j w_{i1} = 0 + 1$
  - $input_1 > 0 \Rightarrow y_1 = 1$
  - $\mathbf{y} = (1, 0, 1, 0)$

# Evolución de la red (II)

- $\mathbf{x} = (0,0,1,\mathbf{0})$ .

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

- $\mathbf{y} = (1,0,1,\mathbf{0})$
- **Se elige la unidad 4** para actualizar su activación:
  - $input_4 = x_4 + \sum_j y_j w_{j4} = 0 + (-2)$
  - $input_4 < 0 \Rightarrow y_4 = 0$
  - $\mathbf{y} = (1, 0, 1, \mathbf{0})$

# Evolución de la red (III)

- $\mathbf{x} = (0, 0, \mathbf{1}, 0)$ .

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

- $\mathbf{y} = (1, 0, \mathbf{1}, 0)$
- **Se elige la unidad 3** para actualizar su activación:
  - $input_3 = x_3 + \sum_j y_j w_{j3} = 1 + 1$
  - $input_3 > 0 \Rightarrow y_3 = 1$
  - $\mathbf{y} = (1, 0, \mathbf{1}, 0)$



# Evolución de la red (IV)

- $\mathbf{x} = (0, 0, 1, 0)$ .

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

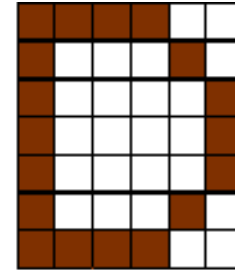
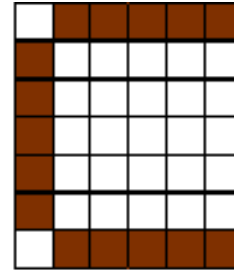
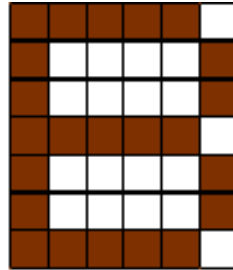
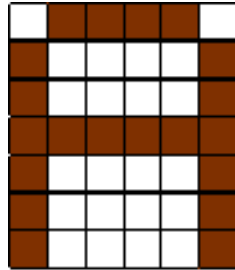
- $\mathbf{y} = (1, 0, 1, 0)$
- **Se elige la unidad 2** para actualizar su activación:
  - $input_2 = x_2 + \sum_j y_j w_{i2} = 0 + 2$
  - $input_2 > 0 \Rightarrow y_2 = 1$
  - $\mathbf{y} = (1, 1, 1, 0)$

# Evolución de la red (V)

- Algunas de las activaciones han cambiado durante este ciclo, por tanto se requiere al menos otro ciclo más:
  - Se elige la neurona 3:  $\mathbf{y} = (1, 1, \mathbf{1}, 0)$
  - Se elige la neurona 1:  $\mathbf{y} = (\mathbf{1}, 1, 1, 0)$
  - Se elige la neurona 4:  $\mathbf{y} = (1, 1, 1, \mathbf{0})$
  - Se elige la neurona 2:  $\mathbf{y} = (1, \mathbf{1}, 1, 0)$
- No hay cambio en las activaciones tras este ciclo y por tanto la red converge al patrón almacenado frente al input con dos errores (0,0,1,0).

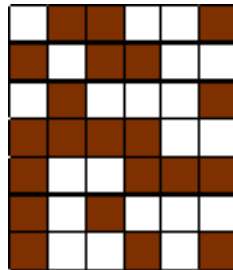
# Ejemplo de reconocimiento de caracteres

patrones  
aprendidos:

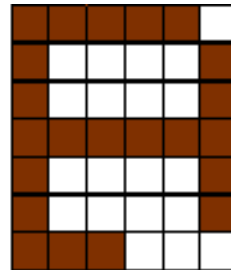


$s(A) = (0,1,1,1,1,0,1,0,0,0,0,1,1,0,0,0,0,1,1,1,1,1,1,0,0,0,0,1,\dots,1,0,0,0,1)$

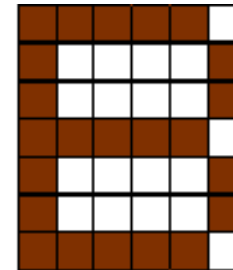
evolución de  
actividades:



input



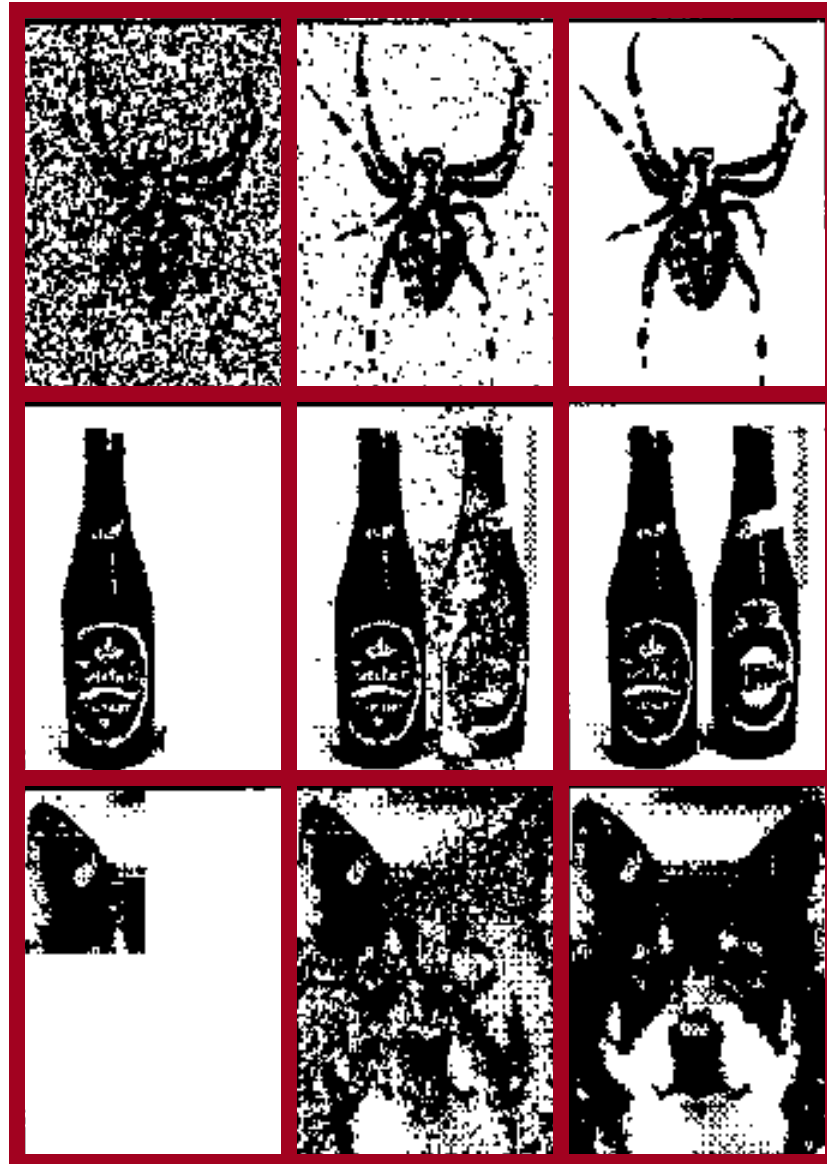
iteración 1



iteración 2

# Ejemplo de reconocimiento de imágenes

130x180 pixels =  
23400 neuronas =  
 $547 \cdot 10^6$   
conexiones



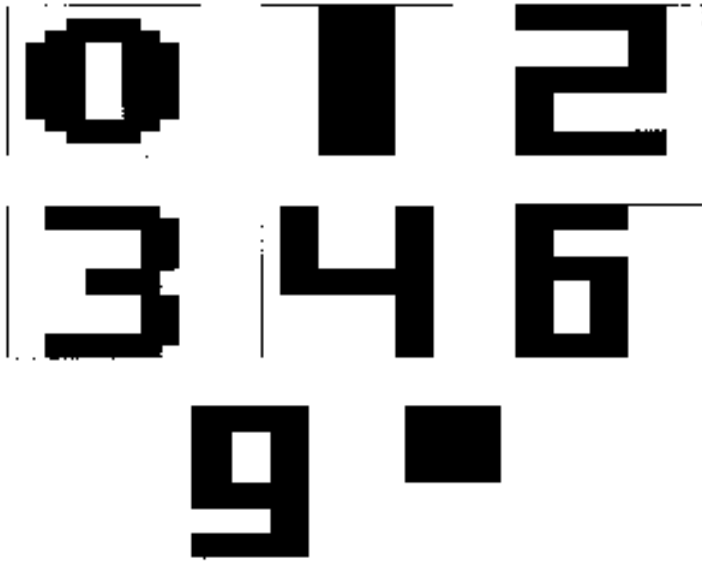
red entrenada  
con 7 imágenes

# Presencia de estados espurios

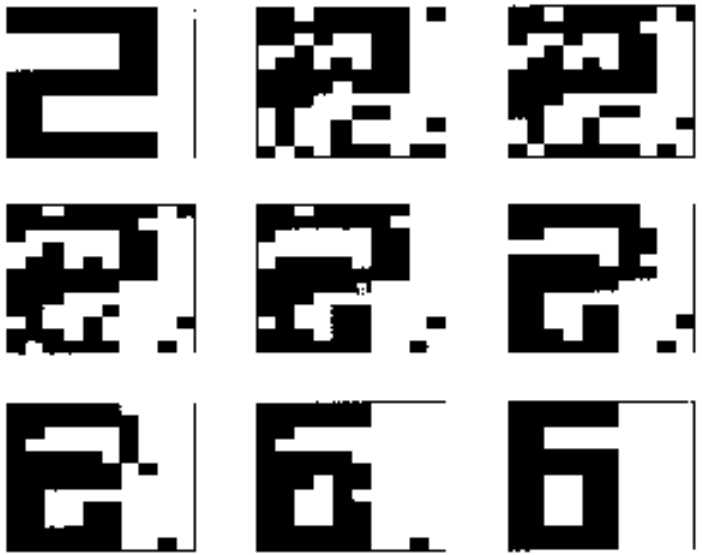
- Si el vector de entrada no es reconocido, la red puede converger a un vector que no es ninguno de los almacenados: *estado estable espurio*.
- Los estados espurios se pueden agrupar en tres tipos:
  - Memorias inversas de las aprendidas ( $E$  simétrica).
  - Combinaciones lineales de un número impar de estados.
  - Mínimos locales de  $E$  que no están correlacionados con ninguna de las memorias de la red.

# Ej: evolución incorrecta y estados espurios

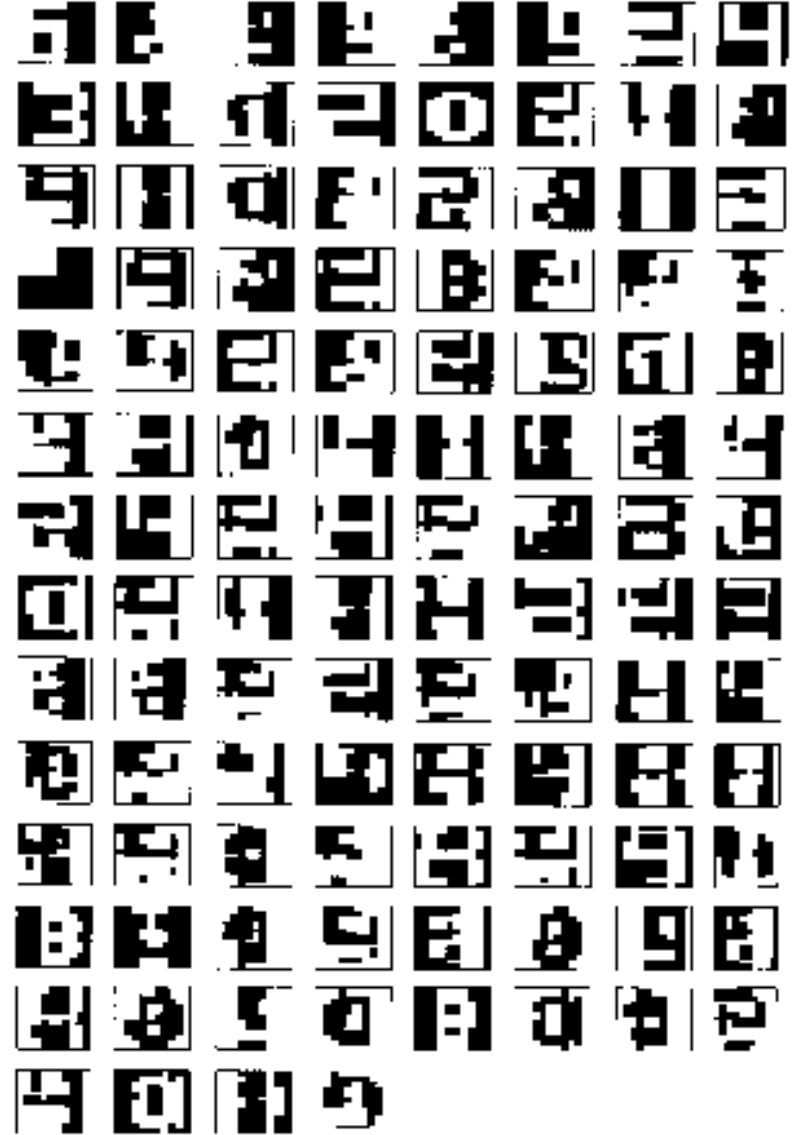
Memorias



Evolución incorrecta



Estados espurios



# Red de Hopfield de estados continuos

$$\tau_i \, dv_i/dt_i = f(\sum_j v_j w_{ij} - \theta_i) - v_i$$

- $f$  es la función de activación (tipo sigmoideal, p.ej.).
- $\tau_i$  es una tasa de retardo.
- Las activaciones  $v_i$  toman valores continuos.
- Las salidas de todas las neuronas se realizan de forma síncrona y continuada en el tiempo.

# Función de energía con estados continuos

$$E = -\frac{1}{2} \sum_{i \neq j} \sum_j (v_i v_j w_{ij}) + \sum_i \int_0^{v_i} f^{-1}(v) dv$$

Si  $f(x) = 1/(1 + \exp(-\alpha(x - \theta_i)))$ :

$$f^{-1}(v) = -1/\alpha \ln(1/v - 1) + \theta_i, \text{ y}$$

$$E = -\frac{1}{2} \sum_{i \neq j} \sum_j (v_i v_j w_{ij}) + \sum_i (\theta_i v_i) - 1/\alpha (\sum_i \int_0^{v_i} \ln(1/v - 1) dv)$$

• Los mínimos de  $E$  dependen de  $\alpha$ :

• Si  $\alpha = \infty$ , el último término es cero y se recupera la  $E$  de la red de actividades discretas.

• Si  $\alpha$  toma valores pequeños, disminuye el número de mínimos de  $E$ : el número de estados estables de la red.



# Aplicaciones de las redes de Hopfield

- Memoria asociativa de acceso por contenido.
- Reconocimiento de patrones (caracteres, imágenes, voz).
- Resolución de problemas de optimización con restricciones (modelo de estados continuos):
  - Resolución del problema del viajante.
  - Resolución de ecuaciones.
  - Manipulación de grafos.
  - Procesado de señales (diseño de conversores analógico digitales).

# Limitaciones de las redes de Hopfield

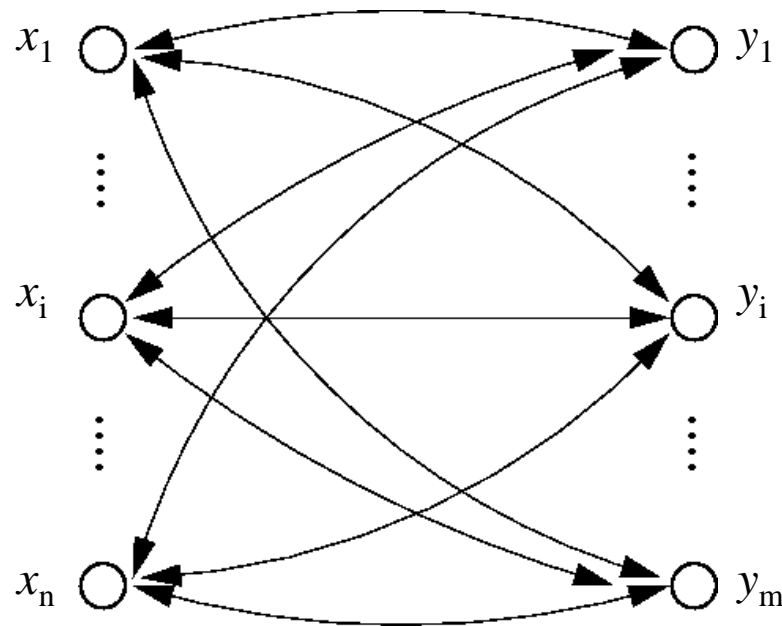
- El número de memorias está severamente limitado:  
Red de 100 neuronas, 9900 conexiones: 15 memorias.
- Si se almacenan demasiadas memorias, la red puede converger a valores de salida diferentes a los aprendidos (estados espurios).
- No siempre se puede garantizar que la red realice una asociación correcta si las memorias no tienen un alto grado de ortogonalidad.
- *Brain-State-In-A-Box* es una red asociativa alternativa con mayor capacidad de almacenamiento.

# Referencias para profundizar en las redes de Hopfield:

- J.J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences* 79:2554-2558. 1982.
- J.J. Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences* 81: 3088-3092. 1984.

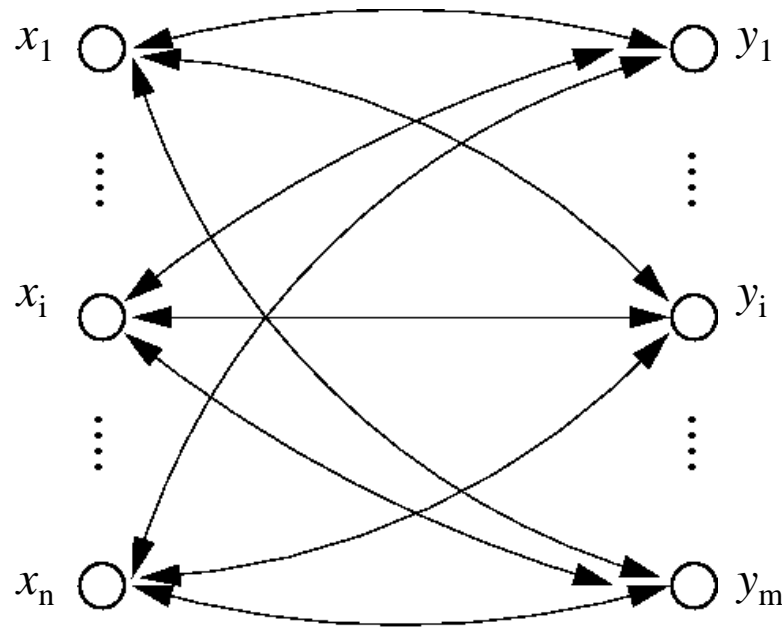
# Redes BAM

- Un tipo de red derivado de las redes de Hopfield: *Bidirectional Associative Memory* (B. Kosko, 1988).
- Son redes heteroasociativas de dos capas.
- La red itera de una capa a otra hasta el equilibrio.
- Responden al input en cualquiera de las dos capas.



# Redes BAM: matriz de pesos

- Las conexiones son bidireccionales: si la matriz de pesos de la capa  $X$  a la capa  $Y$  es  $\mathbf{W}$ , la matriz de pesos de la capa  $Y$  a la capa  $X$  es  $\mathbf{W}^t$ .



# Otras codificaciones

- Conjunto de  $P$  patrones  $\mathbf{s}(p) \cdot \mathbf{t}(p)$ ,  $p = 1, \dots, P$  donde

$$\mathbf{s}(p) = (s_1(p), \dots, s_i(p), \dots, s_n(p))$$

$$\mathbf{t}(p) = (t_1(p), \dots, t_i(p), \dots, t_m(p))$$

- Para almacenar patrones binarios con matriz de representación bipolar:

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

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

- Para almacenar patrones bipolares:

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

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

# Redes BAM: matriz de pesos

- Conjunto de  $P$  patrones  $\mathbf{s}(p) \cdot \mathbf{t}(p)$ ,  $p = 1, \dots, P$  donde

$$\mathbf{s}(p) = (s_1(p), \dots, s_i(p), \dots, s_n(p))$$

$$\mathbf{t}(p) = (t_1(p), \dots, t_i(p), \dots, t_m(p))$$

- Para almacenar patrones binarios con matriz de representación bipolar:

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

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

- Para almacenar patrones bipolares:

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

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

# Redes BAM: funciones de transferencia

- Para vectores de entrada binarios, las funciones de transferencia para la capa  $Y$  y la capa  $X$  respectivamente son:

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

- Para vectores de entrada bipolares, las funciones de transferencia para la capa  $Y$  y la capa  $X$  respectivamente son:

$$y_j = \begin{cases} 1 & \text{si } y\_in_j > \theta_j \\ y_j & \text{si } y\_in_j = \theta_j \\ -1 & \text{si } y\_in_j < \theta_j \end{cases} \quad x_i = \begin{cases} 1 & \text{si } x\_in_i > \theta_i \\ x_i & \text{si } x\_in_i = \theta_i \\ -1 & \text{si } x\_in_i < \theta_i \end{cases}$$



# Redes BAM: algoritmo

**Paso 0:** Inicializar todos los pesos para almacenar los P patrones. Inicializar todas las activaciones a 0.

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

**Paso 1:** Para cada vector de test, ejecutar los pasos 2 a 5:

**Paso 2a:** Presentar el patrón de entrada  $\mathbf{x}$  a la capa  $X$   
(las activaciones de  $X$  se establecen según ese patrón)

**Paso 2b:** Presentar el patrón  $\mathbf{y}$  a la capa  $Y$

**Paso 3:** Mientras las activaciones no converjan,  
ejecutar los pasos 4-6:

**Paso 4:** Actualizar las activaciones de la capa  $Y$ :

$$y\_in_j = \sum_i w_{ij} x_i \quad y_j = f(y\_in_j) \quad \text{y enviarlas a la capa } X$$

**Paso 5:** Actualizar las activaciones de la capa  $X$ :

$$x\_in_i = \sum_j w_{ij} y_j \quad x_i = f(x\_in_i) \quad \text{y enviarlas a la capa } Y$$

# Redes BAM: ejemplo (1/4)

- Reconocimiento de letras con codificación bipolar:



$(-1, 1)$



$(1, 1)$

- Las matrices de pesos son:  
para almacenar  $A \rightarrow (-1, 1)$

$$\begin{bmatrix} 1 & -1 \\ -1 & 1 \\ 1 & -1 \\ -1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ -1 & 1 \\ 1 & -1 \\ -1 & 1 \\ -1 & 1 \\ 1 & -1 \\ -1 & 1 \end{bmatrix}$$

$C \rightarrow (1, 1)$

$$\begin{bmatrix} -1 & -1 \\ 1 & 1 \\ 1 & 1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \\ 1 & 1 \\ -1 & -1 \\ -1 & -1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

la matriz final  $W$ :

$$\begin{bmatrix} 0 & -2 \\ 0 & 2 \\ 2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ -2 & 0 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}$$

# Redes BAM: ejemplo (2/4)

- Prueba con los patrones A y C:

$$(-1 \ 1 \ -1 \ 1 \ -1 \ 1 \ 1 \ 1 \ 1 \ 1 \ -1 \ 1 \ 1 \ -1 \ 1) \mathbf{W} = (-14, 16) \rightarrow (-1, 1)$$

$$(-1 \ 1 \ 1 \ 1 \ -1 \ -1 \ 1 \ -1 \ -1 \ 1 \ -1 \ -1 \ -1 \ 1 \ 1) \mathbf{W} = (14, 16) \rightarrow (1, 1)$$

- Los vectores de la capa  $Y$  también proporcionan adecuadamente los vectores de la capa  $X$ . La matriz de pesos es la transpuesta de  $\mathbf{W}$ .

prueba de los dos patrones:

$$(-1, 1) \mathbf{W}^T =$$

**A**

$$(-1, 1) \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & -2 & 0 & -2 & -2 & 0 & 0 & -2 & -2 & 2 & 0 \\ -2 & 2 & 0 & 2 & -2 & 0 & 2 & 0 & 0 & 2 & -2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$= (-2 \ 2 \ -2 \ 2 \ -2 \ 2 \ 2 \ 2 \ 2 \ 2 \ -2 \ 2 \ 2 \ -2 \ 2)$$

$$(1, 1) \mathbf{W}^T =$$

**C**

$$(1, 1) \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & -2 & 0 & -2 & -2 & 0 & 0 & -2 & -2 & 2 & 0 \\ -2 & 2 & 0 & 2 & -2 & 0 & 2 & 0 & 0 & 2 & -2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$= (-2 \ 2 \ 2 \ 2 \ -2 \ -2 \ 2 \ -2 \ -2 \ 2 \ -2 \ -2 \ -2 \ 2 \ 2)$$

$$\rightarrow (-1 \ 1 \ 1 \ 1 \ -1 \ -1 \ 1 \ -1 \ -1 \ 1 \ -1 \ -1 \ -1 \ 1 \ 1)$$

# Redes BAM: ejemplo (3/4)

- Prueba con los patrones en la capa  $Y$  ruidosos:

$$(0, 1)W^T =$$

$$(0, 1) \begin{bmatrix} 0 & 0 & 2 & 0 & 0 & -2 & 0 & -2 & -2 & 0 & 0 & -2 & -2 & 2 & 0 \\ -2 & 2 & 0 & 2 & -2 & 0 & 2 & 0 & 0 & 2 & -2 & 0 & 0 & 0 & 2 \end{bmatrix}$$

$$= (-2 \ 2 \ 0 \ 2 \ -2 \ 0 \ 2 \ 0 \ 0 \ 2 \ -2 \ 0 \ 0 \ 0 \ 2)$$

$$\rightarrow (-1 \ 1 \ 0 \ 1 \ -1 \ 0 \ 1 \ 0 \ 0 \ 1 \ -1 \ 0 \ 0 \ 0 \ 1).$$

- El vector  $x$  se envía de vuelta a la capa  $Y$  con la matriz  $W$ :

$$(-1 \ 1 \ 0 \ 1 \ -1 \ 0 \ 1 \ 0 \ 0 \ 1 \ -1 \ 0 \ 0 \ 0 \ 1) \begin{bmatrix} 0 & -2 \\ 0 & 2 \\ 2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ -2 & 0 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}$$

La red converge pero a un estado espurio, una solución que no es uno de los patrones de entrenamiento.

$$\rightarrow (0 \ 1).$$

# Redes BAM: ejemplo (4/4)

- Prueba con los patrones y ruidosos y alguna información sobre  $\mathbf{x}$ . Por ejemplo  $\mathbf{y}=(0 \ 1)$ ,  $\mathbf{x}=(0 \ 0 \ -1 \ 0 \ 0 \ 1 \ 0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0)$

$$\begin{aligned}
 (0, 1)W^T &= \\
 (0, 1) &\begin{bmatrix} 0 & 0 & 2 & 0 & 0 & -2 & 0 & -2 & -2 & 0 & 0 & -2 & -2 & 2 & 2 & 0 \\ -2 & 2 & 2 & 0 & 2 & -2 & 0 & 2 & 0 & 0 & 2 & -2 & 0 & 0 & 0 & 2 \end{bmatrix} \\
 &= (-2 \ 2 \ 2 \ 0 \ 2 \ -2 \ 0 \ 2 \ 0 \ 0 \ 2 \ -2 \ 0 \ 0 \ 0 \ 2) \\
 &\rightarrow (-1 \ 1 \ -1 \ 1 \ -1 \ 1 \ 1 \ 1 \ 0 \ 1 \ -1 \ 0 \ 0 \ 0 \ 1),
 \end{aligned}$$

No es el patrón A, pero el vector  $\mathbf{x}$  se envía de vuelta a la capa  $Y$  con la matriz  $\mathbf{W}$ :

$$(-1 \ 1 \ -1 \ 1 \ -1 \ 1 \ 1 \ 1 \ 0 \ 1 \ -1 \ 0 \ 0 \ 0 \ 1) \begin{bmatrix} 0 & -2 \\ 0 & 2 \\ 2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ 0 & 2 \\ -2 & 0 \\ -2 & 0 \\ 0 & 2 \\ 0 & -2 \\ -2 & 0 \\ -2 & 0 \\ 2 & 0 \\ 0 & 2 \end{bmatrix}$$

Si se envía este patrón de vuelta a la capa  $X$  una vez más y se recupera el patrón A

$$\rightarrow (-6, 10) \rightarrow (-1, 1).$$

# Función de Energía para las BAM

$$L = -1/2 (x W y^T + y W^T x^T) = x W y^T = - \sum_i \sum_j (x_i w_{ij} y_j)$$

- $L$  decrece conforme la red itera tanto para actualizaciones de las actividades sincrónicas como asíncronas.
- Capacidad: hasta  $\min(2^n, 2^m)$ .
- Una red BAM se puede considerar como una red de Hopfield compuesta de todas las neuronas de las dos capas pero sin interconexiones entre los miembros de cada una de las capas.
- Las actualizaciones de las activaciones dentro de cada capa pueden ser simultáneas porque no afectan el estado de las otras neuronas que comparten capa.