Redes Neuronales Artificiales

Red de Hopfield

***Almacenar un conjunto de* p *patrones de forma tal que cuando se presente un nuevo patrón, la red responda produciendo alguno de los patrones previamente almacenados que más se parezca al presentado*.**

Problema de memoria asociativa

Red de Hopfield

Las memorias asociativas resuelven problemas del tipo de *reconstrucción de patrones y memorias direccionables por contenido.*

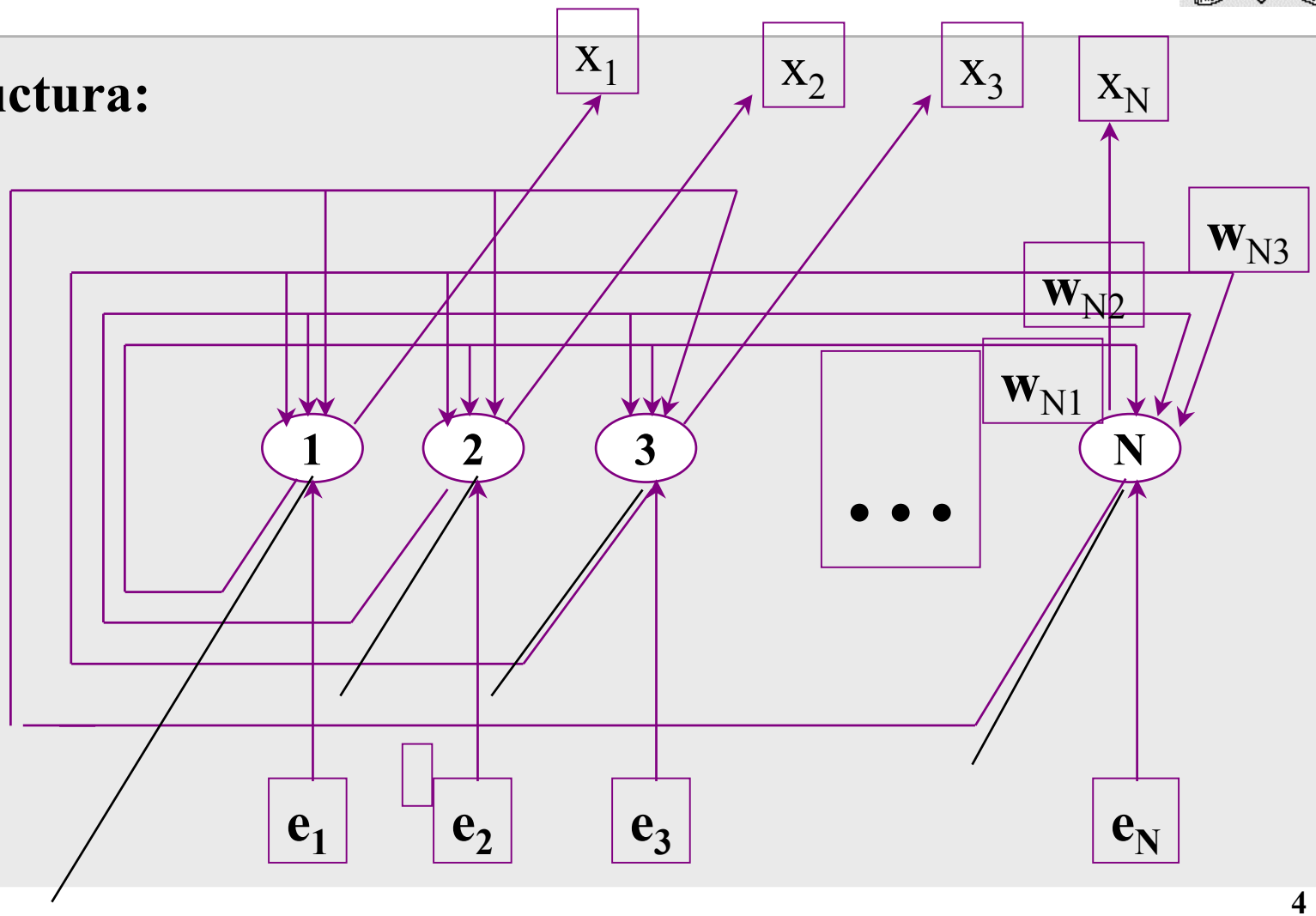
La red de Hopfield consiste de una red monocapa con N neuronas cuyos valores de salida pueden ser binarios o continuos.

Cada neurona se encuentra conectada a todas las demás pero no consigo misma.

Los pesos asociados a las conexiones entre pares de neuronas son simétricos, es decir, $w_{ij}=w_{ji}$.

Red de Hopfield

Estructura:



Red de Hopfield

- Funcionamiento:
  - En el instante (t=0) se aplica la información de entrada (valores $e_1,e_2,...,e_N$)

$s_i(t=0)=e_i$ para $1<=i<=N$

  - La red realiza iteraciones hasta alcanzar la convergencia (hasta que $s_i(t+1) = s_i(t)$).

$s_i(t+1)=f\,[\Sigma_{j=1,N} w_{ij} s_j(t)-t_i]$ para $1<=i<=N$

Red de Hopfield

- Aprendizaje:

  No supervisado de tipo hebbiano.

- Expresión:

- $$w_{ij} = \Sigma_{k=1,M}\ e_i^{(k)}\ e_j^{(k)} \quad \text{para } 1 <= ij <= N; i \neq j$$

  $$0 \quad \text{para } 1 <= ij <= N; i = j$$

Red de Hopfield

Suponiendo K clases a almacenar en una red

1. Se calculan los pesos W_{ij}

2. Se muestra a la red un vector

3. La red itera en pasos discretos hasta la convergencia.

4. El vector mostrado es el vector de salida.

Red de Hopfield

1. Calcule los pesos: $w_{ij}=\Sigma X^k_i X^k_j$

2. Inicialice la red con un vector de entrada X

3. Itera hasta converger:

$$y_j(t+1)=F_h[\Sigma w_{ij}\, y_j(t)]$$

Siendo: $F_h(x)=1$ si $x>0$

-1 si $x<0$

$y_j(t)$ si $x=0$

Red de Hopfield
Ejemplo

1. Almacenar los patrones:

X1=[1,-1,1,1] y X2=[-1,1,1,1]

Multiplicamos cada patrón por su traspuesta

$$\begin{vmatrix} 1 \\ -1 \\ 1 \\ 1 \end{vmatrix} * [1, -1, 1, 1] = \begin{vmatrix} 1 & -1 & 1 & 1 \\ -1 & 1 & -1 & -1 \\ 1 & -1 & 1 & 1 \\ 1 & -1 & 1 & 1 \end{vmatrix}$$

$$\begin{vmatrix} -1 \\ 1 \\ 1 \\ 1 \end{vmatrix} * [-1, 1, 1, 1] = \begin{vmatrix} 1 & -1 & -1 & -1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \\ -1 & 1 & 1 & 1 \end{vmatrix}$$

Sumamos estos productos haciendo la diagonal 0.

$$W = \begin{vmatrix} 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{vmatrix}$$

Red de Hopfield
Ejemplo

Sea que deseamos recuperar el vector X=[1, 1, 1, -1]

Evaluamos la red, X*W

$$[1, 1, 1, -1] * \begin{bmatrix} 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{bmatrix} = [-2, -2, -2, 2]$$

$Y_1=[-1, -1, -1, 1]$ $\quad Y_1 <> X$

Red de Hopfield
Ejemplo

Ahora evaluamos la red, y_1*W

$$[-1, -1, -1, 1] * \begin{bmatrix} 0 & -2 & 0 & 0 \\ -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 \\ 0 & 0 & 2 & 0 \end{bmatrix} = [2, 2, 2, -2]$$

$Y_2=[1, 1, 1, -1]$ $\quad Y_1 <> Y_2$

Red de Hopfield

Aplicaciones:

- **Reconocimiento de imágenes y de voz**
- **Control de motores**
- **Resolución de problemas de optimización**
- **Diseño de conversores análogos**
- **Procesamiento de señales.**

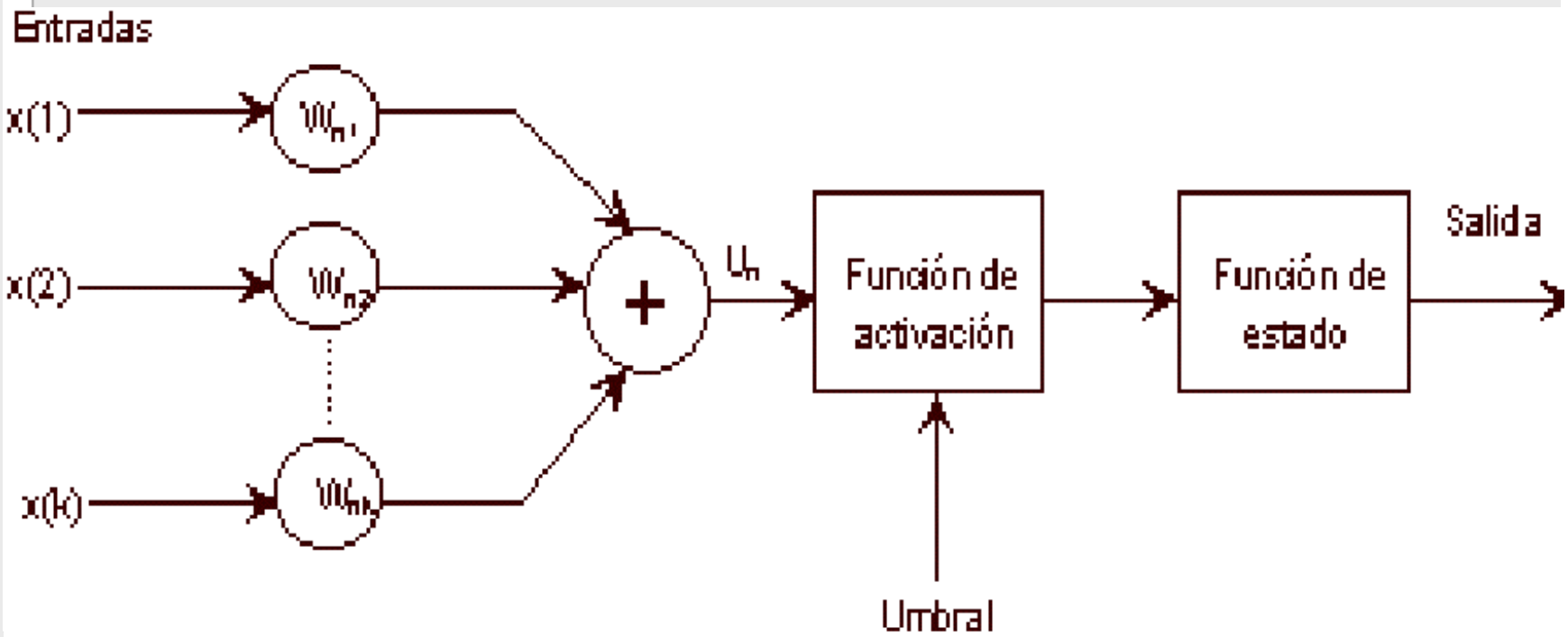
Arquitectura de las redes

En teoría de redes neuronales podemos distinguir tres niveles en su arquitectura:

- **Microestructura.** Hace referencia a los elementos más pequeños de las redes neuronales: las neuronas.
- **Mesoestructura.** Resultado de la combinación de las neuronas. Serían las redes neuronales propiamente dichas.
- **Macroestructura.** Combinación de redes, se podría denominar a este nivel "comité de expertos". Existen diferentes tipos de combinación: paralelo, jerárquica, etc. dependiendo de la aplicación que se quiera implementar.

Microestructura

La neurona puede tener diferentes formas dependiendo de la aplicación:



Mesoestructura

La combinación de las neuronas se puede realizar de muchas formas diferentes. En esta combinación se habla de capas y dependiendo del número de éstas y de la conexión entre ellas tenemos diferentes clasificaciones.

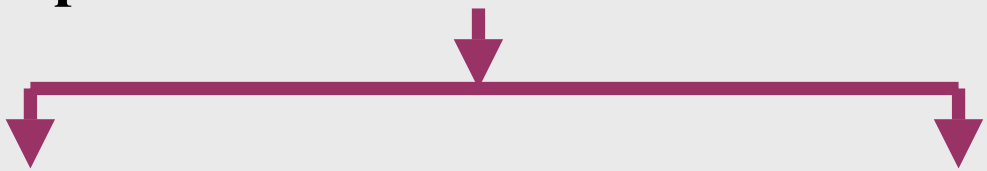
Número de capas	**Monocapa (1 capa)**
	Multicapa
Tipos de conexiones	**Recurrente (Hay realimentación)**
	No recurrente
Número de conexiones	**Totalmente conectada**
	Parcialmente conectada

Macroestructura

Existen problemas donde una combinación de redes da un mejor comportamiento que usar una sola red. Esta combinación puede ser en paralelo (todas tienen el mismo peso), en serie (la salida de una red es la entrada a otra mayor), jerárquica (en problemas de clasificación, existen redes más expertas que otras), etc. o variaciones de ellas dependiendo de la aplicación concreta.

Algoritmos de aprendizaje

Los procedimientos para determinar las conexiones entre neuronas reciben el nombre de algoritmos de aprendizaje ya que es en los pesos donde reside el "conocimiento" de una red.



NO-Supervisado

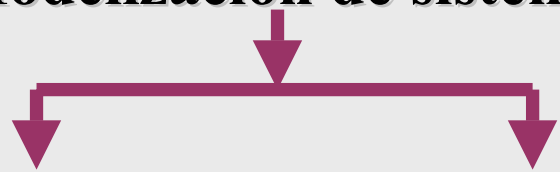
Extracción de características

Agrupamiento de datos

Supervisado

Clasificación de patrones

Modelización de sistemas



Por Refuerzo **Por Corrección**

El supervisado dispone de información sobre la salida deseada de la red.

Aprendizaje por corrección. Se dispone de información adecuada. Al disponer de información lógica (es o no la señal deseada) estamos en un aprendizaje por refuerzo.

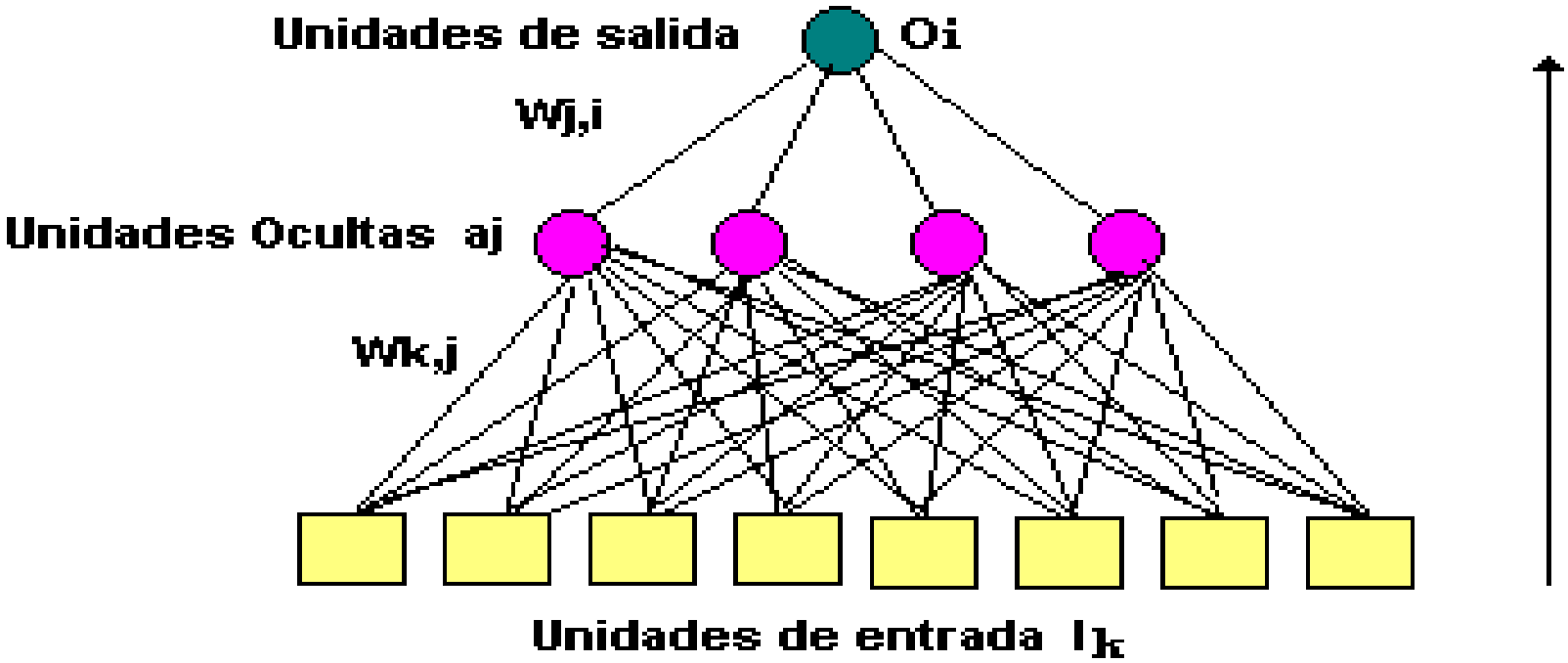
La función debe ser siempre una función monótona creciente de la diferencia entre la señal deseada (señal que debería dar la red) y la salida proporcionada por la red. El problema es, pues, de optimización: búsqueda del mínimo de una función .

Hay que tener en cuenta que una de la principales características de las redes neuronales, que las hacen especiales frente a otros métodos, es su capacidad de generalización; es decir, ante entradas desconocidas son capaces de dar salidas aproximadas a las deseadas.

Aprendizaje por propagación posterior

Se le presentan a la red las entradas (ejemplos), y si esta calcula un vector de salida que coincida con la meta, no hay nada que hacer. Pero si existe un error (diferencia entre la salida y la meta) los pesos se ajustan para disminuir el error. El secreto consiste en evaluar las consecuencias del error y dividirlas entre todos los pesos contribuyentes.

Regla de actualización



$$W_{i,j} \leftarrow W_{i,j} + \alpha \times a_j \times Err_i \times g'(Ini)$$

$$W_{i,j} \longleftarrow W_{i,j} + \alpha \times a_j \times Err_i \times g'(Ini)$$

$W_{i,j}$ = Peso del enlace que está entre la unidad j y la unidad i

g' = Derivada de la función de activación

α = Velocidad de aprendizaje

a_j = Unidades ocultas

T_i = Salida Meta

Err_i = Ti - Oi

I_i = Activación de un a unidad i en el nivel de entrada

Aprendizaje por refuerzo

- **Un ejemplo. El ambiente propone pares de entrada/salida y el objetivo consiste en aprender la función que haya producido tales pares.**

- **En el aprendizaje por refuerzo no se le proporcionan ejemplos, se empieza sin contar con un modelo de ambiente ni con una función de utilidad.**

El aprendizaje por refuerzo es otra forma de plantear todo el problema de la inteligencia artificial.

- Un agente en un ambiente recibe percepciones, los correlaciona con utilidades positivas o negativas y luego decide que acción emprender.
- La utilidad de un estado es la suma esperada de recompensas recibidas entre el momento presente y el final de la secuencia.

Para aprender la utilidades pueden utilizarce 3 metodos

1- PMC (Promedio de mínimos cuadrados)

2- PDA (Programación dinámica adaptativa)

3- DT (Desviaciones temporales)

Conclusiones

El aprendizaje es la adaptación al ambiente.

El aprendizaje por refuerzo es una área muy activas en la investigación del aprendizaje maquinal. Sus aplicaciones en la robótica son prometedoras, el aprendizaje por refuerzo en ambientes inaccesibles es también tema de investigaciones actuales.