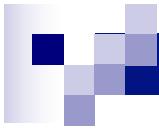


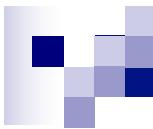
INTELIGENCIA ARTIFICIAL APLICADA AL CONTROL

Tema 5: Redes
Neuronales: Aprendizaje

Dpto.: Arquitectura de Computadores y Automática
Autor: Matilde Santos

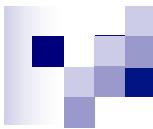


5.3 APRENDIZAJE EN LAS REDES NEURONALES



APRENDIZAJE

- Un sistema aprende cuando es capaz de experimentar modificaciones estructurales y/o funcionales, de acuerdo con la experiencia, en vistas a conseguir una mayor eficacia en su interacción con el medio
 - ◆ Modificación de pesos sinápticos
 - Hasta pesos estables y adecuado
 - ◆ Reglas de aprendizaje
 - Criterios y procedimientos (iterativos) de modificación de los pesos



TIPOS DE APRENDIZAJE

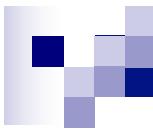
■ MECANISMO:

- Supervisado
 - Redes heteroasociativas
- No supervisado
 - Redes autoasociativas

	<i>Supervised Learning</i>	<i>Unsupervised Learning</i>
<i>Discrete</i>	classification or categorization	clustering
<i>Continuous</i>	regression	dimensionality reduction

■ IMPLEMENTACIÓN

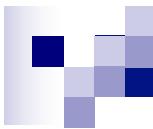
- On-line
- Off-line



REDES HETEROASOCIATIVAS

- La red aprende pares de datos [(A₁,B₁), (A₂,B₂), (A₃,B₃),...]: si se presenta A_i la red responderá B_i.
- Calculan cierta función (no expresable analíticamente) entre la entrada y la salida (caja negra)
- Al menos 2 capas (una para captar y retener la información de entrada y otra para mantener la salida).
- Ej: Perceptron, ART, Backpropagation, ...

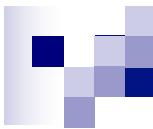
Aprendizaje Supervisado



REDES AUTOASOCIATIVAS

- Se presenta una información de entrada: A₁, A₂, A₃,..., A_n : la red realiza una correlación respondiendo con uno de los datos almacenados, el que más se le parezca.
- Su principal misión es reconstruir la información que se le presenta incompleta o distorsionada.
- Ej: Hopfield, Brain-State-in-a-box, ...

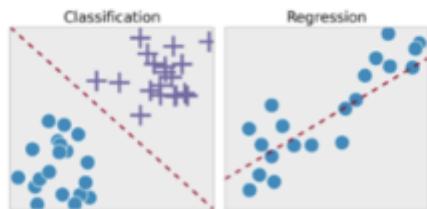
Aprendizaje NO Supervisado



TIPOS DE APRENDIZAJE

Supervised Learning

- Regression – sales forecasting
- Classification – grouping potential customers



Unsupervised Learning

- Clustering – splitting customers by preferences
- Dimensionality Reduction – storing less data



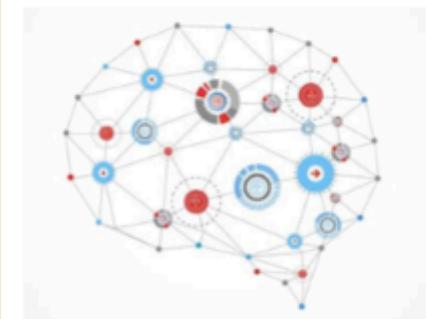
Reinforcement Learning

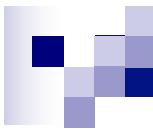
- Reasoning - robotics



Deep Learning

- Neural Networks – image recognition



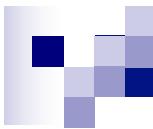


APRENDIZAJE SUPERVISADO

Se le indica la corrección o no de una determinada salida durante el entrenamiento

Tipos:

- Aprendizaje por refuerzo (premio o castigo)
- Aprendizaje estocástico
- Aprendizaje a partir de ejemplos o por corrección de error



APRENDIZAJE NO SUPERVISADO

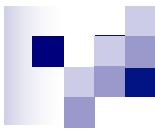
DESCUBRIR REGULARIDADES,
RELACIONES, CARACTERÍSTICAS

No realimentación de las salidas

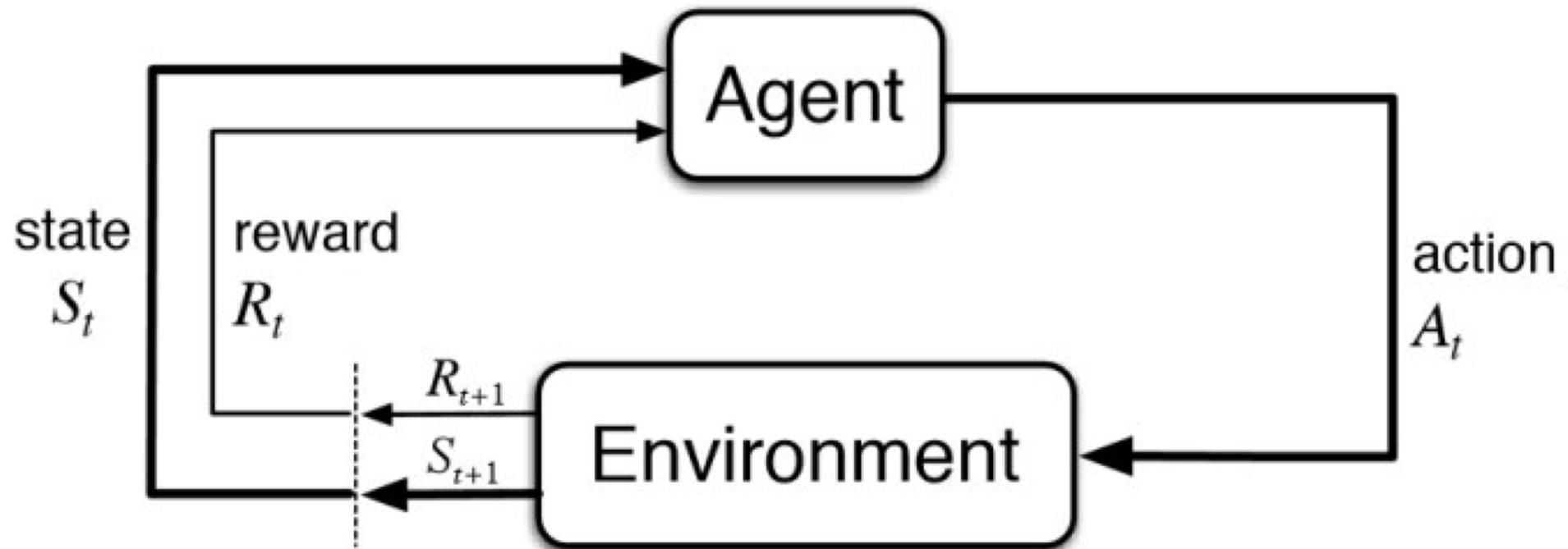
No refuerzo de un crítico

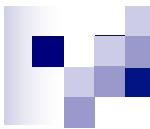
Tipos:

- ◆ Aprendizaje Hebbiano
 - ◆ Red de Hopfield
- ◆ Aprendizaje competitivo y/o cooperativo
 - ◆ RED ART

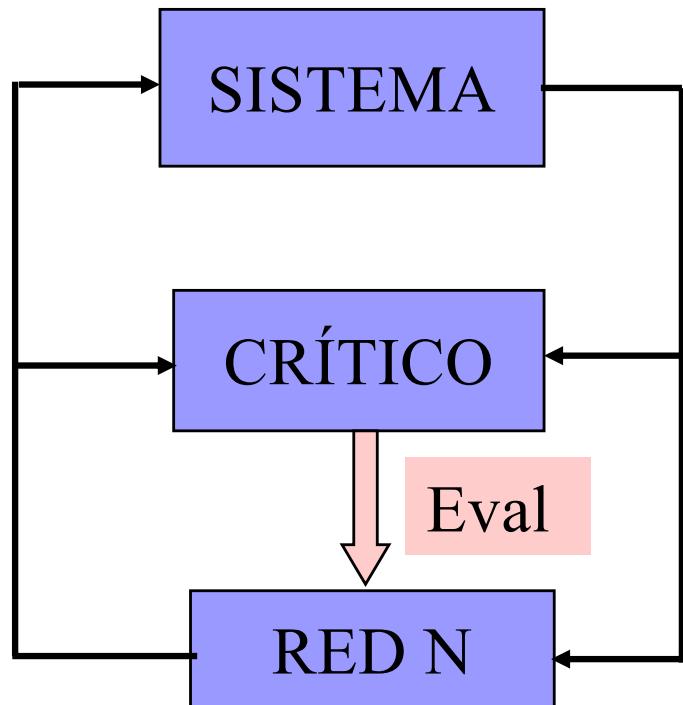


APRENDIZAJE POR REFUERZO





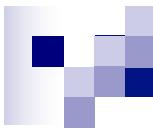
APRENDIZAJE POR REFUERZO



- Evalúa el rendimiento de la red en términos generales de éxito o fracaso
- Refuerzo:
 - Positivo: (+1)
 - Negativo (-1)
- No necesita conocer exactamente la salida deseada

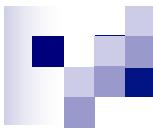
PROBLEMAS:

- Construir un crítico eficaz (Red neuronal)
- Modificación de la red neuronal
- Lento

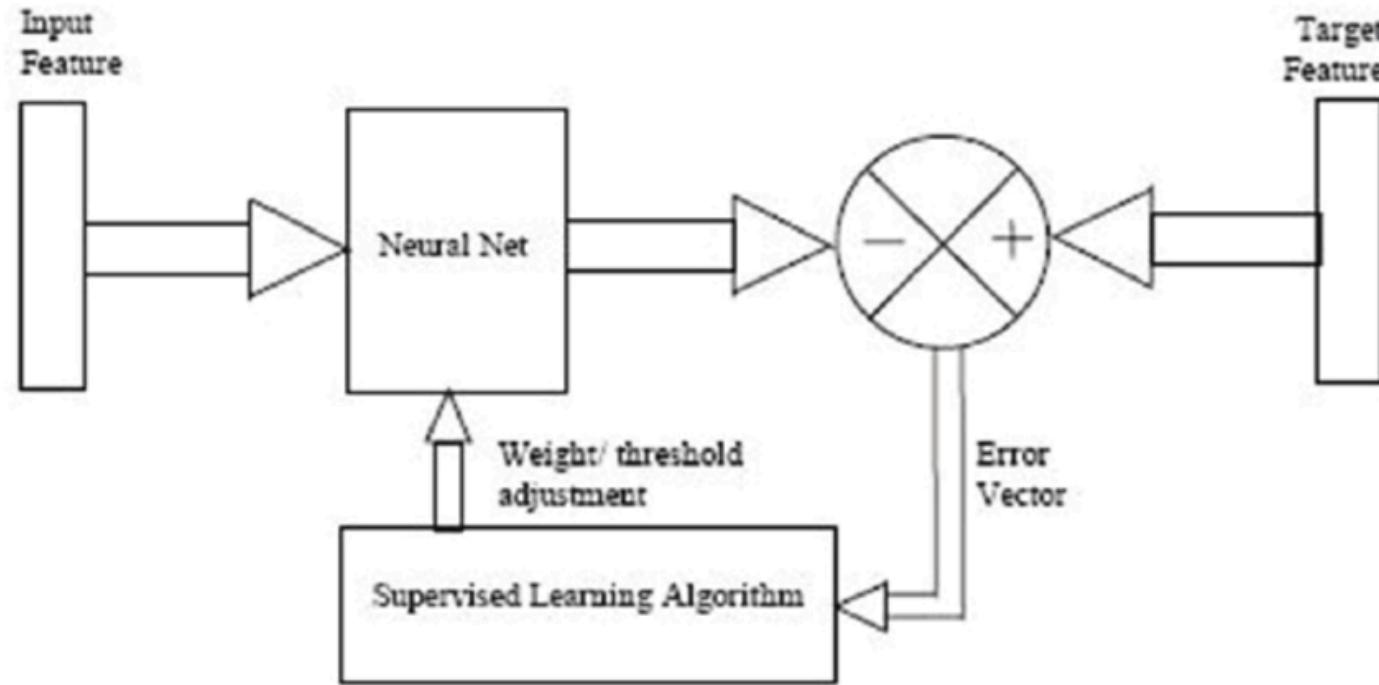


APRENDIZAJE SUPERVISADO ESTOCÁSTICO

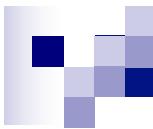
- Cambios aleatorios en los valores de los pesos
 - Evaluar su efecto
 - Si la red se comporta mejor, se acepta
 - Si no, se acepta con una determinada función de probabilidad



APRENDIZAJE SUPERVISADO: POR EJEMPLOS



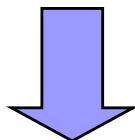
Minimizar el error (distancia entre las salidas deseadas y las de la red) expresado como una función de los pesos



APRENDIZAJE DEL PERCEPTRÓN

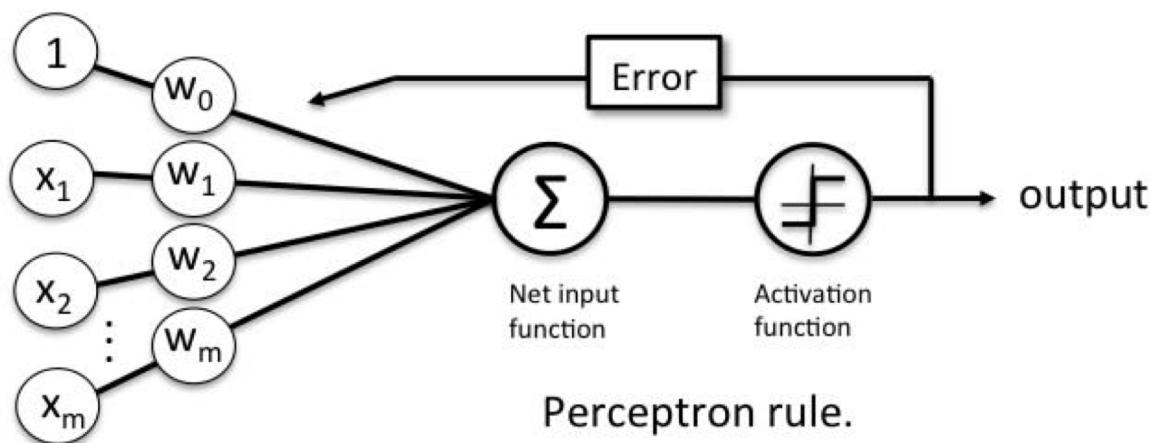
MODIFICACIÓN DE PESOS SINÁPTICOS

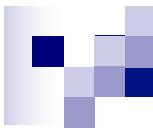
Encontrar un vector de pesos óptimo



Colocación de la superficie de decisión

Inicialización aleatoria: $\mathbf{w} = (w_0, w_1, w_2, \dots, w_n)$





APRENDIZAJE DEL PERCEPTRÓN

FUNCIÓN CRITERIO DEL PERCEPTRÓN $J(\mathbf{w})$

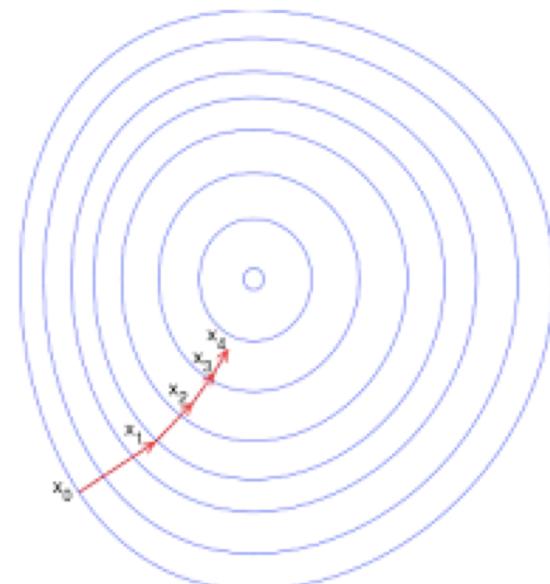
$$J(\mathbf{w}) = \sum_{\mathbf{x} \in X} |\mathbf{w} \cdot \mathbf{x}|$$

Suma de las distancias de las entradas mal clasificadas a la superficie de decisión

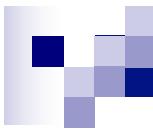
- Minimizar $J(\mathbf{w}) = 0$
- Dirección del gradiente negativo
- Actualizar pesos
 - ✓ iterativo

$$\mathbf{w}_{t+1} = \mathbf{w}_t + \rho \cdot \text{grad}(J)$$

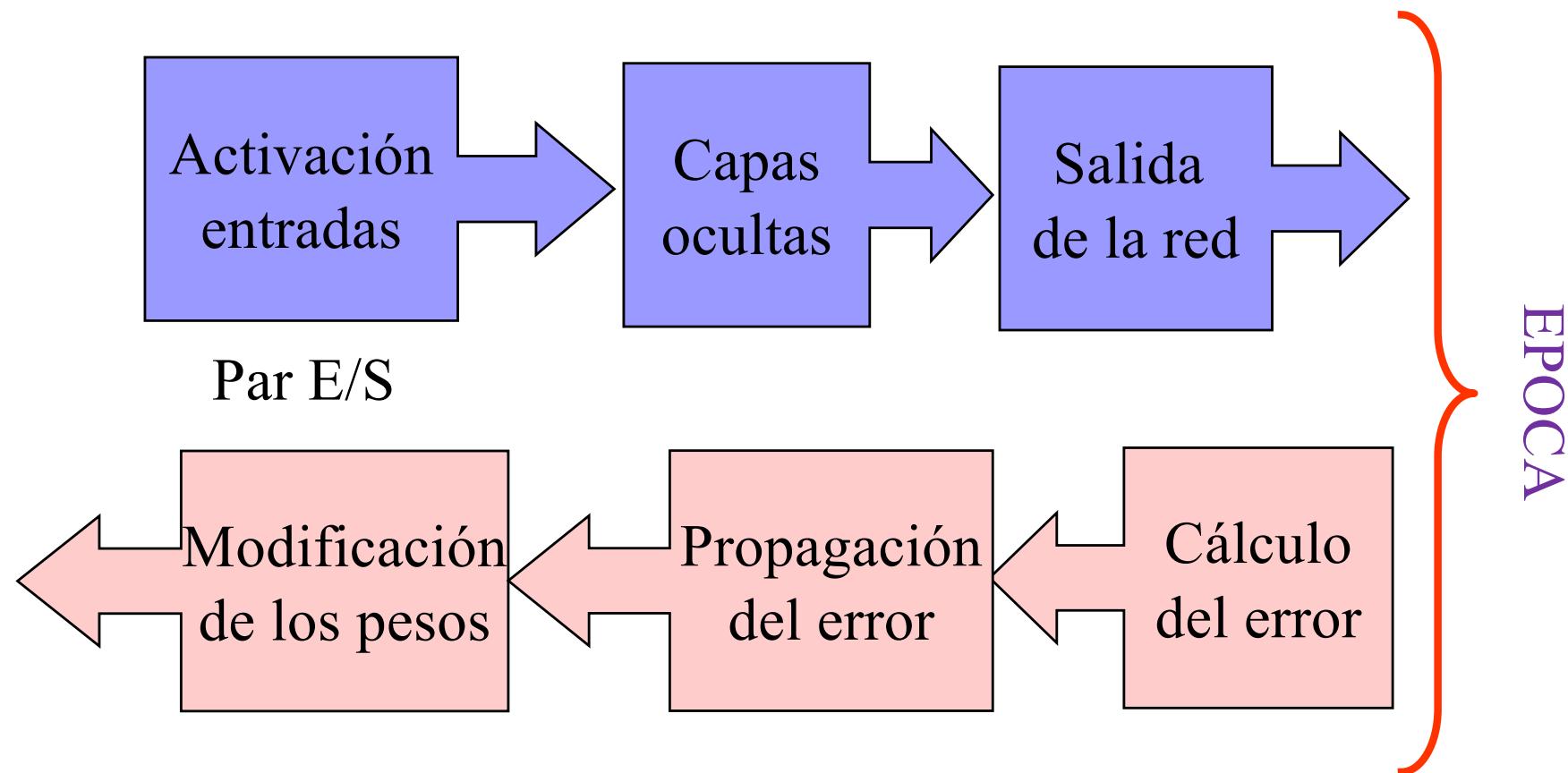
velocidad de aprendizaje



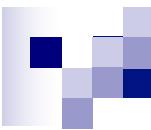
SUPERVISADO



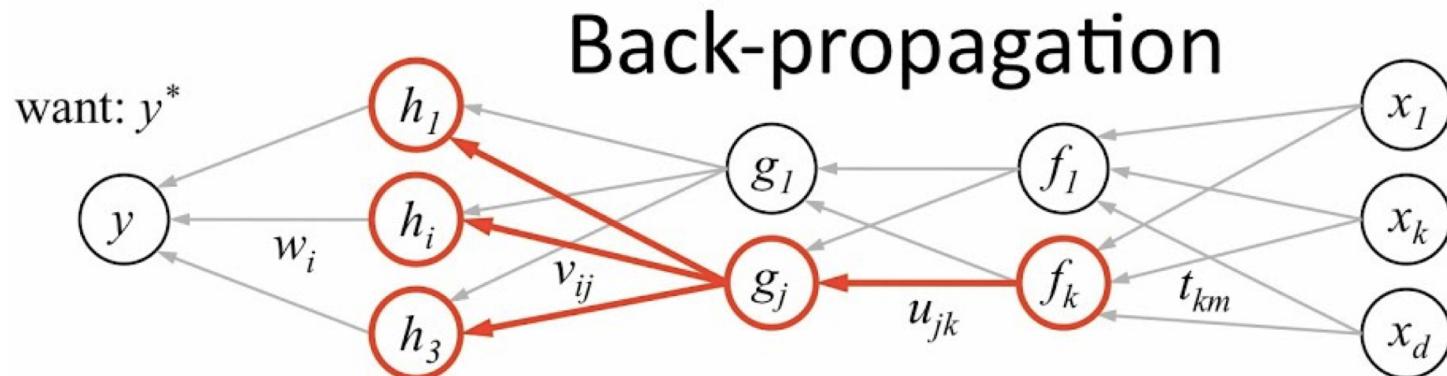
PERCEPTRÓN MULTICAPA: BACKPROPAGATION



SUPERVISADO

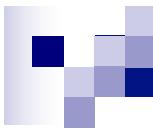


BACKPROPAGATION



1. receive new observation $\mathbf{x} = [x_1 \dots x_d]$ and target y^*
2. **feed forward:** for each unit g_j in each layer $1 \dots L$
compute g_j based on units f_k from previous layer: $g_j = \sigma\left(u_{j0} + \sum_k u_{jk} f_k\right)$
3. get prediction y and error $(y - y^*)$
4. **back-propagate error:** for each unit g_j in each layer $L \dots 1$

$$y_i = g_1\left(\sum_{j=1}^M w_{ij} s_j\right) = g_1\left(\sum_{j=1}^M w_{ij}\left(g_2\left(\sum_{r=1}^L t_{jr} x_r\right)\right)\right)$$



BACKPROPAGATION

Valores $\delta_j = S_j - O_j$

$$Error = \sum_{ejemplos} \sum_j (S_j - O_j)^2$$

$$\Delta w_{ij} = -\alpha \frac{\partial Error}{\partial w_{ij}} + \mu w_{ij}$$

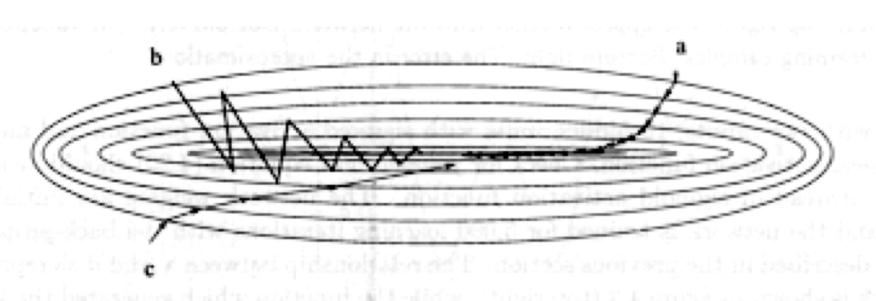
S_j : salida deseada de la neurona j

O_j : salida real de la neurona j

α : velocidad de aprendizaje

μ : momentum

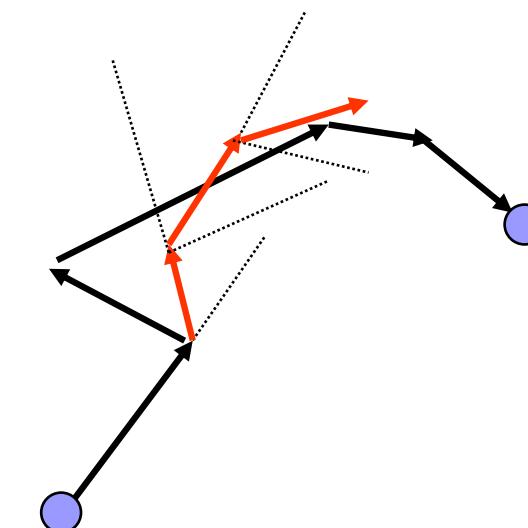
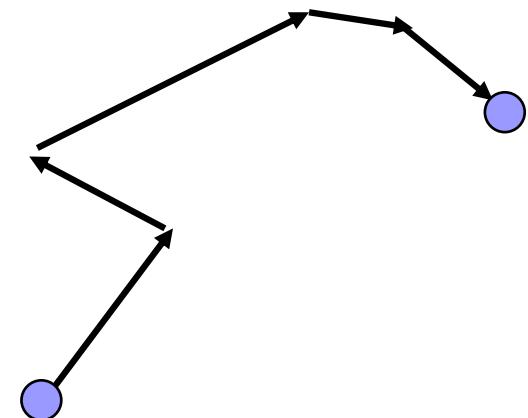
acelera el aprendizaje
evita mínimos locales

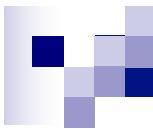


- a) Utilización de α baja
- b) Utilización de α alta
- c) Utilización de α alta y momento μ

MOMENTOS

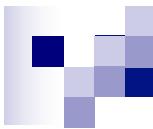
- Acelerar la bajada en las direcciones similares al descenso al ir acumulando dichos valores
- Si tenemos direcciones con oscilaciones de signo en iteraciones consecutivas se ajustaran los pesos en cantidades pequeñas, es decir, tendrá un efecto estabilizador





PROBLEMAS

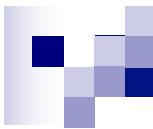
- Mínimos locales
- Convergencia
 - No extraña bien, no generaliza (memoriza)
 - El número de neuronas por capa generalmente limitado por el nº de ejemplos
- Número de patrones de entrenamiento
 - Dejar uno fuera (LOO)
 - K-fold
- Dispersión de los patrones



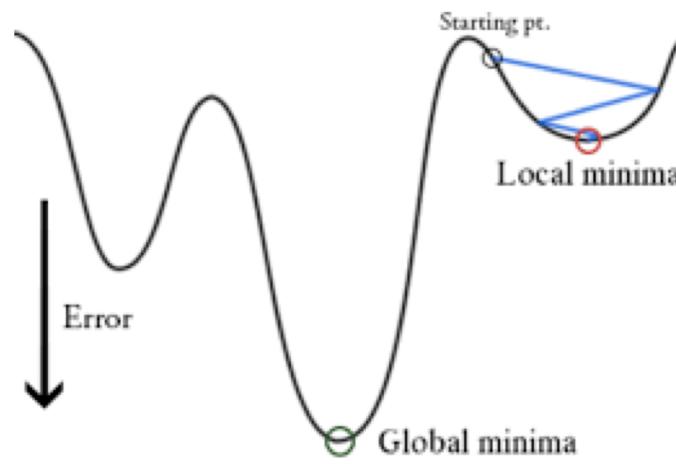
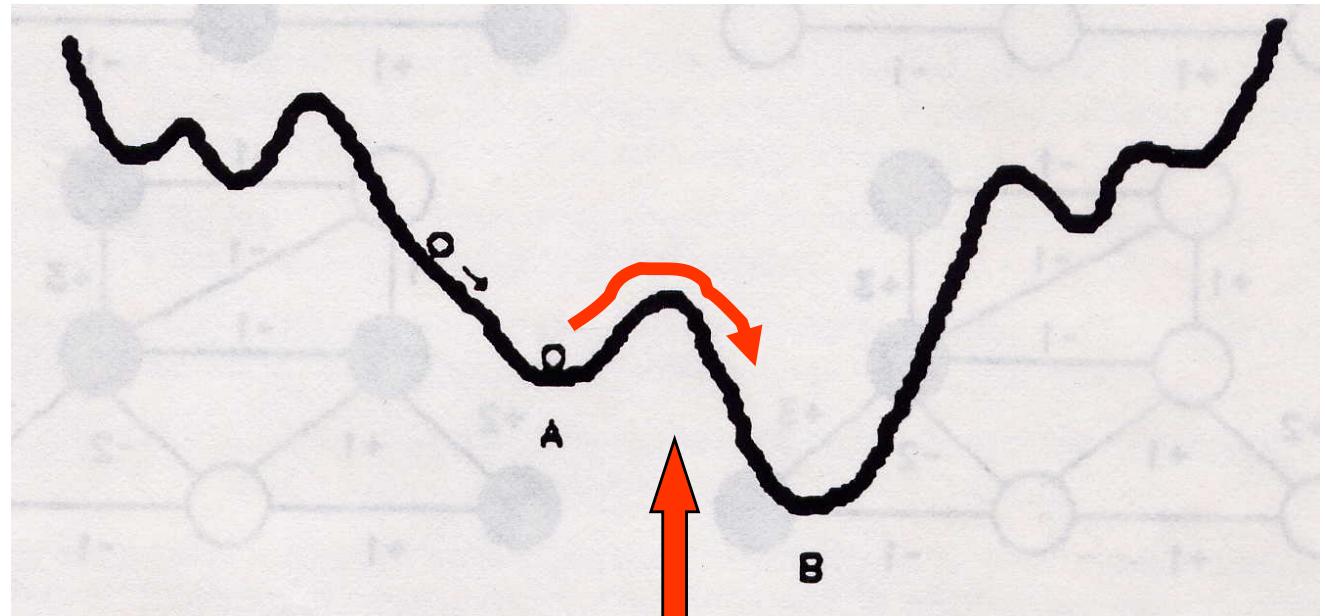
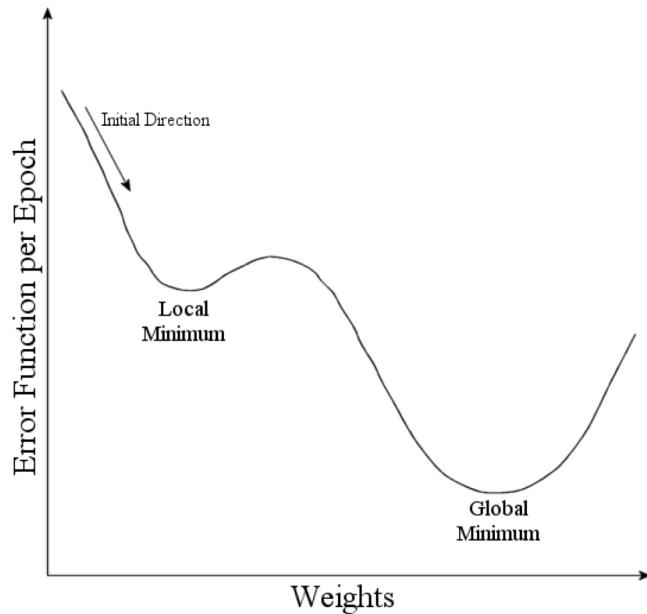
MÍNIMOS LOCALES

■ Mínimos locales en la función de error

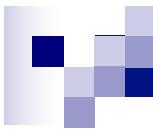
- El entrenamiento se detiene aunque no se haya alcanzado la tasa de convergencia fijada
- Soluciones:
 - Inicializar de nuevo los pesos de la red de forma aleatoria y volver a entrenarla
 - cambiar la topología de la red (número de capas y número de neuronal)
 - modificar el algoritmo de aprendizaje
 - modificar los parámetros de aprendizaje
 - modificar el conjunto de entrenamiento
 - presentar los patrones en otro orden



MÍNIMOS LOCALES



**SIMULATED
ANNEALING**



SIMULATED ANNEALING (SA)

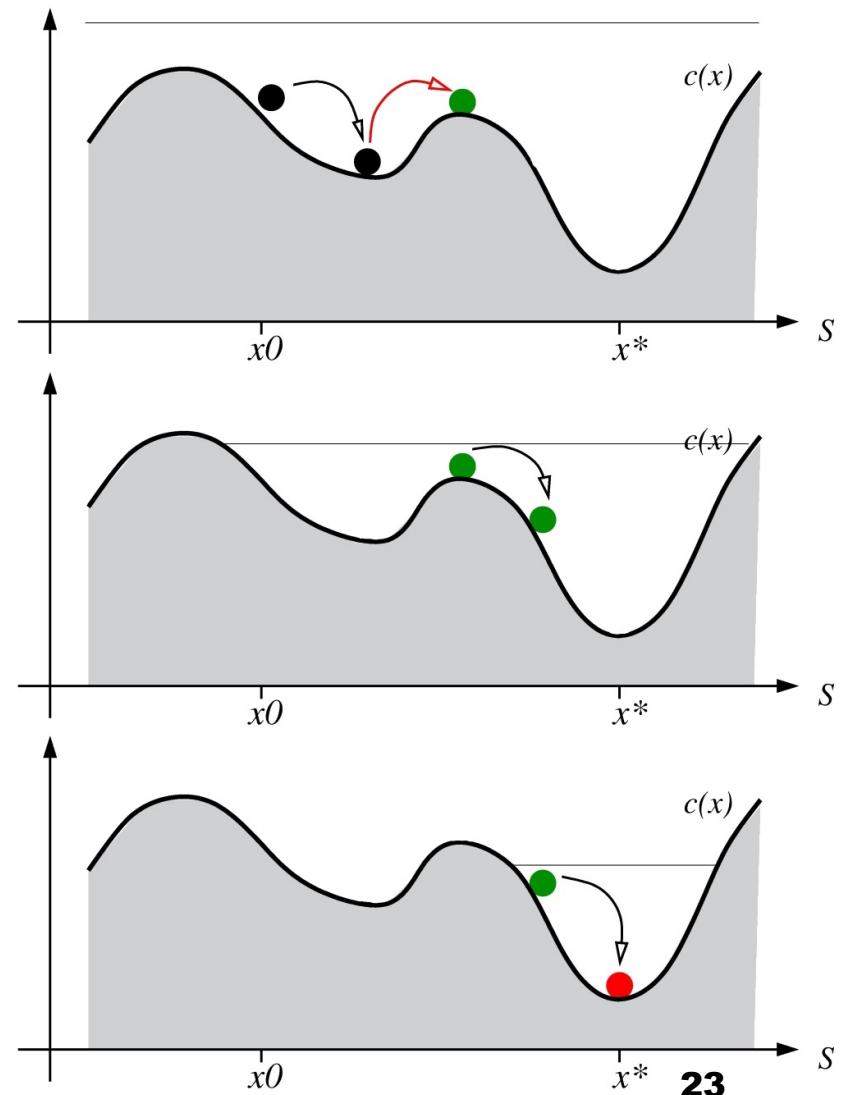
recocido, templado o enfriamiento simulado

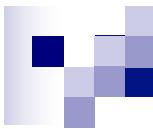
■ Algoritmo

- S : conjunto soluciones posibles (“estados”)
- Función de coste en S (“energía”)
- Encontrar un elemento en S que minimize el coste (un estado de mínima energía)

■ Los estados: función de distribución de probabilidad de Boltzman:

- T : temperatura del sistema





ALGORITMO ITERATIVO SA

Initialize:

- initial solution \mathbf{x} ,
- highest temperature T_h ,
- and coolest temperature T_l

$T = T_h$

When the temperature is higher than T_l

While not in equilibrium

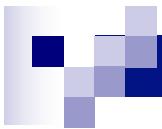
 Search for the new solution \mathbf{X}'

 Accept or reject \mathbf{X}' according to Metropolis Criterion

End

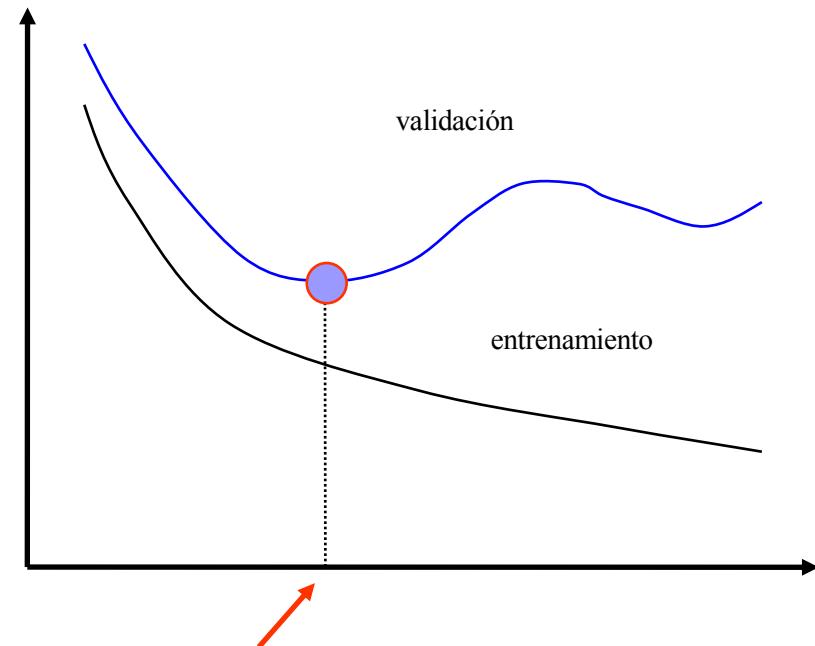
Decrease the temperature T

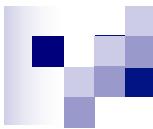
End



GENERALIZACIÓN

- Error de entrenamiento:
 - Decrece monótonamente durante la fase de entrenamiento
- Error de validación:
 - Decrece hasta un punto a partir del cual crece: *superajuste*
- El proceso de entrenamiento debe finalizar cuando se alcance el primer mínimo de la función del error de validación

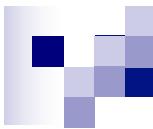




TAMAÑO DE LA NN

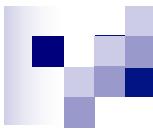
- NN pequeñas son preferibles a las grandes:
 - menor número de parámetros
 - el entrenamiento es más rápido
 - mayor capacidad de generalización al utilizar nuevos patrones

 1. Partir de una NN de gran tamaño y podarla eliminando unidades de proceso y conexiones
 2. Comenzar con una NN muy pequeña e ir añadiendo unidades de proceso, conexiones o capas
 3. Partir de una red de tamaño suficiente y podar las conexiones y unidades poco relevantes. A continuación se añaden nuevas unidades de proceso con pesos aleatorios y se vuelve a entrenar la red



PODA DE UNA NN

- Eliminar conexiones cuyos pesos sinápticos sean de pequeña magnitud
- Eliminar conexiones cuya existencia no afecte significativamente a las salidas de la red
 - Ir comparando las salidas de la red cuando un peso sináptico es reducido a cero.
- Eliminar sensores de entrada que producen cambios insignificantes en la salida de la red
 - Reducir la dimensión de los patrones de entrada al detectar aquellas componentes que son innecesarias (reducción de características)

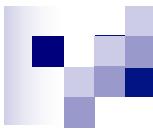


APRENDIZAJE NO SUPERVISADO

- La red no recibe información exterior
- El ajuste de los pesos no se hace en función del conocimiento de si la respuesta es correcta o no

APLICACIONES:

- OBTENCIÓN DE SIMILITUDES, RELACIONES: entre la información presentada y la mostrada en el pasado
- CLASIFICACIÓN O AGRUPACIÓN (Clustering)
- RECONOCIMIENTO DE PATRONES: clases representantes de las entradas
- MAPPING: la salida es un mapa topográfico de las características de las entrada



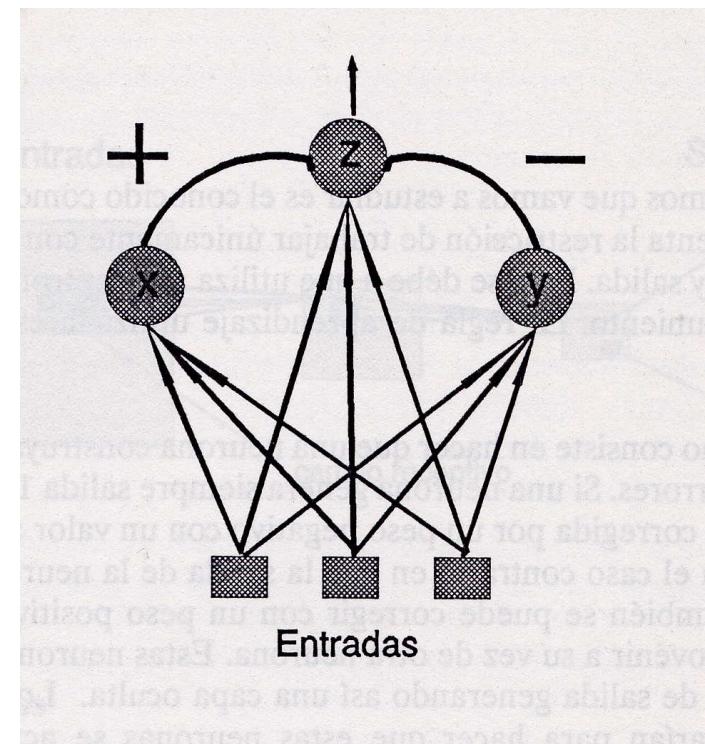
APRENDIZAJE NO SUPERVISADO

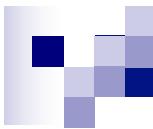
■ COMPETITIVO

- Activación inicial
- Conexiones inhibitorias (o de excitación) de las salidas
- The winner take all

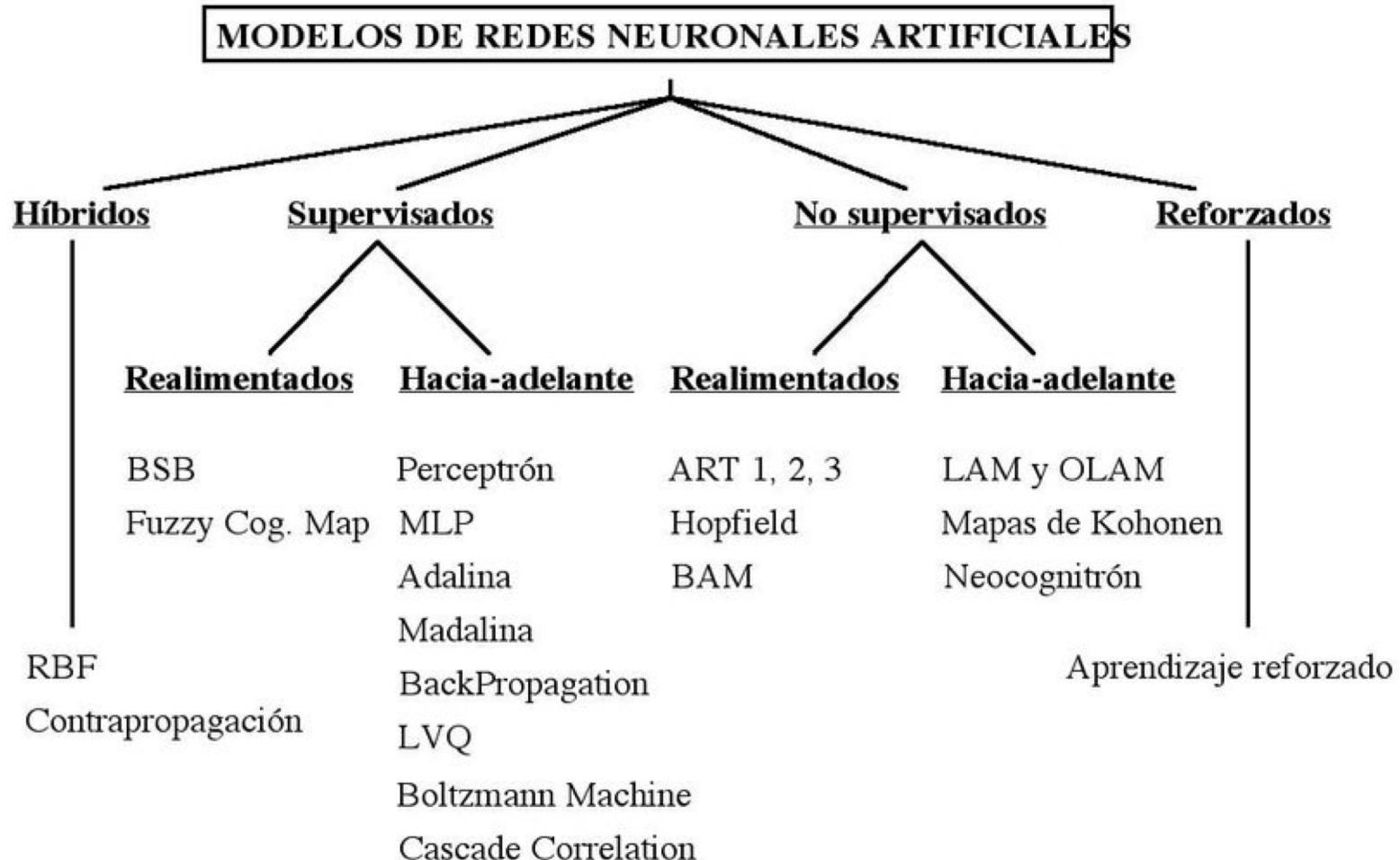
■ HEBBIANO

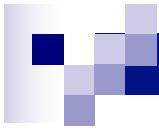
- Correlación (multiplicación por -1 o +1) de los valores de activación (salidas) de las neuronas conectadas



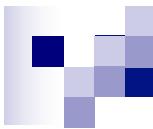


CLASIFICACIÓN POR APRENDIZAJE



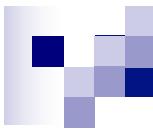


5.4 CONTROL NEURONAL



CONTROL NEURONAL

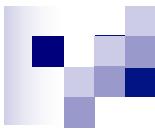
- En el modelado se desprecian muchos datos debido a la ***no-linealidad*** de los mismos
 - un motor: el desgaste de máquina
 - son valores importantes, pero la solución se haría muy compleja o imposible
- Con un modelado tradicional, si el sistema varía, el modelo ya no sirve
- Con las redes neuronales el sistema después de haber recibido unos patrones iniciales comienza a identificar, acepta, aprende y responde ante diferentes señales



NN EN ÁUTOMÁTICA

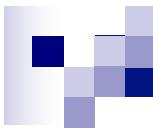
- Las NN, recurrentes y multicapa, muy útiles para identificación y control de sistemas dinámicos no lineales complejos

- Problema de la estabilidad con NN tanto para identificación como para control de sistemas no lineales, no existiendo condiciones algebraicas simples que garanticen la estabilidad total del sistema



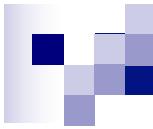
IDENTIFICACIÓN

- Una NN de dos capas con un número de nodos arbitrariamente grande en la capa oculta puede aproximar cualquier función continua $f \in C(R^n, R^m)$ con cualquier grado de aproximación
- Las NN son adecuadas para propósitos de caracterización (identificación de sistemas)
- Sus ventajas frente a expansiones ortogonales y en forma de polinomios no son obvias y tienen que ser justificadas en base a consideraciones prácticas



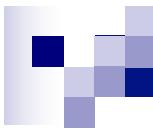
EJEMPLOS

- Red de Hopfield para estimación paramétrica de un sistema, donde cada neurona se corresponde con uno de los coeficientes del modelo de la planta
 - Definir una función de energía en función de los parámetros estimados
- Aprender la dinámica inversa de una planta, bien su representación interna o externa, mediante un perceptrón multicapa



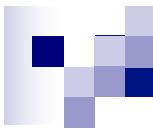
CONTROL NEURONAL

- Las NN son muy adecuadas para problemas de **control no lineal**
 - capacidad de aproximación de funciones no lineales con precisión arbitraria
- Las NN tienen estructura altamente paralela
 - Procesamiento rápido (**tiempo real**)
 - Tolerantes a fallos de nodos individuales
- Las redes son procesadores adaptables a las variaciones en el proceso a controlar (**adaptación**)



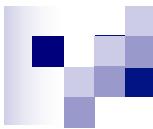
APLICACIONES BASICAS DE NN EN CONTROL

- Control supervisado
- Control inverso directo
- Control adaptativo
- Back-propagación a través del tiempo
- Métodos críticos adaptativos
(aprendizaje por refuerzo)



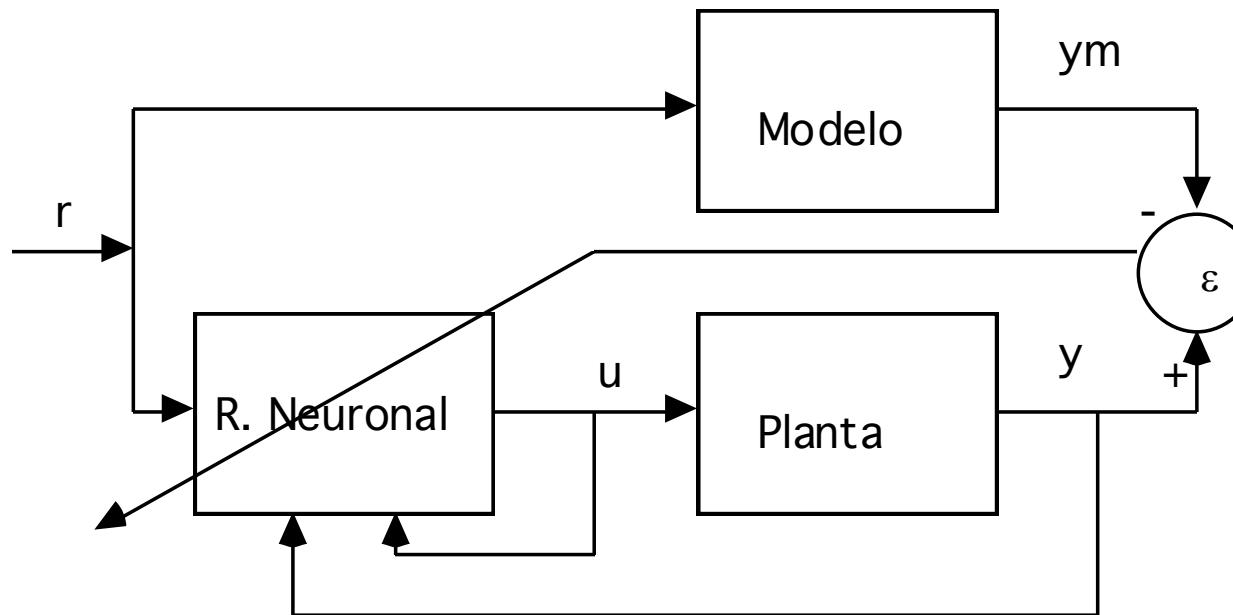
CONTROL SUPERVISADO

- Sistema experto neuronal
- Los controladores supervisados aprenden a partir de acciones y de entradas de los sensores
 - Pueden aprender cómo varía el comportamiento del sistema en función de las entradas proporcionadas de los sensores, y no sólo una serie prefijada de acciones
- Para implementar este tipo de control, se elige cualquiera de los conocidos métodos de aprendizaje supervisado
- Es el método de aprendizaje el que adapta a la red, incluso en tiempo real



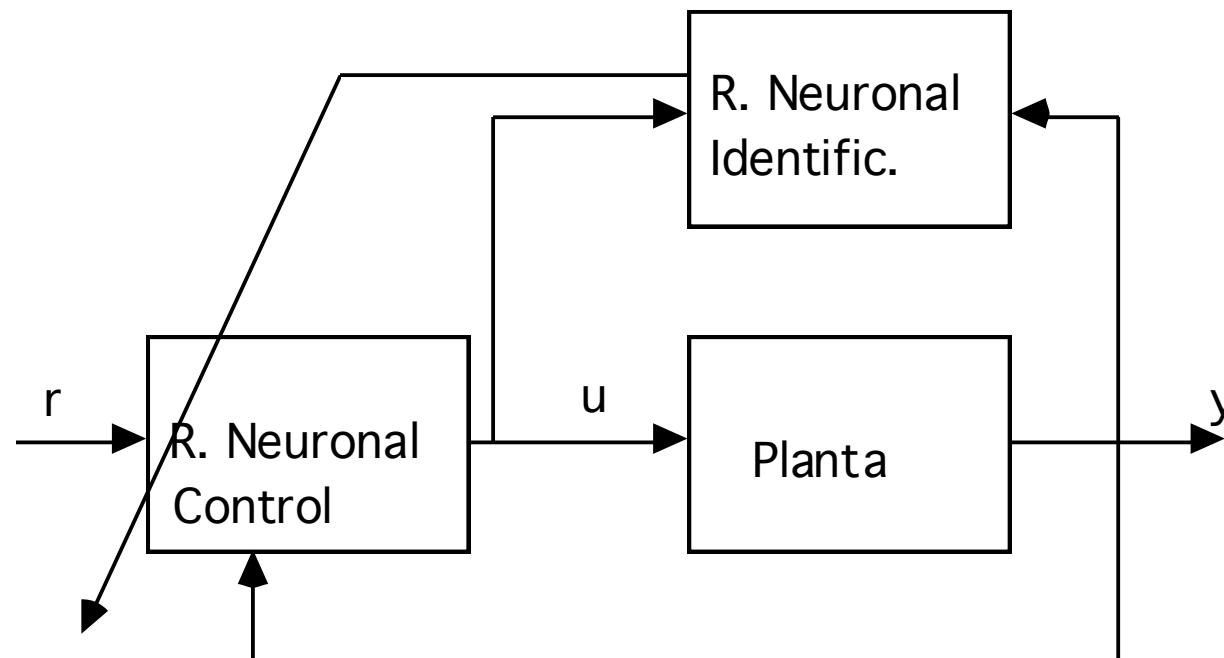
CONTROL DIRECTO

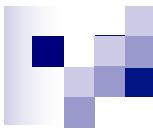
- Los parámetros del controlador se ajustan directamente basándose en el error de la salida
- No existen métodos para hacerlo debido a la naturaleza no lineal tanto de la planta como del controlador
- Back-propagation no puede ser usado directamente al ser la planta desconocida (no se pueden obtener las derivadas parciales necesarias)



CONTROL INDIRECTO

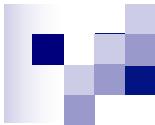
- Se utiliza un modelo parametrizado de la planta
 - El modelo se identifica con una NN
 - El controlador se sintoniza con otra NN en función del modelo





DINÁMICA INVERSA

- La red neuronal que sirve de controlador debe actuar como la inversa de la planta
- A partir de la respuesta deseada (referencia r) se calcula la señal de control u que lleva a la salida real de la planta y a ser la referencia
- El objetivo del aprendizaje es seleccionar los valores de los pesos de la NN de forma que ésta produzca las aplicaciones $r \rightarrow u$ correctas
- Tres métodos de aprendizaje (MLP, backpropagation):
 - Aprendizaje Indirecto (ILA)
 - Aprendizaje Generalizado (GLA)
 - Aprendizaje Especializado (SLA)



EJEMPLOS

- Entrenar una red para que tomando como entrada las variables de estado de la planta, obtenga como salida los coeficientes de un controlador PID (K_p , T_i , T_d)

- Entrenar off-line una red con la dinámica inversa de la planta, y una vez ajustados sus pesos se emplea para ajustar los coeficientes del controlador