Aprendizaje. Redes Neuronales Artificiales

- Introducción
- La neurona biológica
- Neurona Artificial
  - ✓ Revisión histórica
  - ✓ Modelo
  - ✓ RNA
- Arquitecturas
  - ✓ Supervisado (MLP)
  - ✓ No supervisado (SOM)
  - ✓ Recurrentes (Hopfield)
  - ✓ Jerárquicas (ART)
  - ✓ Híbridas (RBF)
- Implementaciones
- Aplicaciones

"Redes Neuronales Artificiales
Una Introducción"
Luis Alonso Romero
Dpto. Informática y Automática
Universidad de Salamanca (España)

Motivaciones

- Las arquitecturas Von-Neumann no pueden tratar con éxito
  - ✓ la información que se presenta es masiva, imprecisa y ruidosa
- Existen algunos paradigmas de computación que abordan estas tareas
  - ✓ Redes Neuronales Artificiales (RNA), Sistemas borrosos, genéticos, etc.
- Las RNA intentan usar algunos de los principios de funcionamiento de los cerebros biológicos
  - ✓ Un cerebro humano tiene unas 10^{11} neuronas, con unas $6*10^{12}$ conexiones(sinapsis), con una enorme eficiencia energética
  - ✓ El consumo de energía del cerebro es del orden de 10^{-16} julios por operación/segundo, mientras que el mejor computador actual no baja de 10^{-6}.
  - ✓ Un sistema de procesamiento de información altamente complejo, no lineal y con un elevadísimo grado de paralelismo.
- Las RNA son sistemas de computación masivamente paralelos (al menos conceptualmente), que constan de un número elevado de procesadores muy simples ("*neuronas*") interconectados.

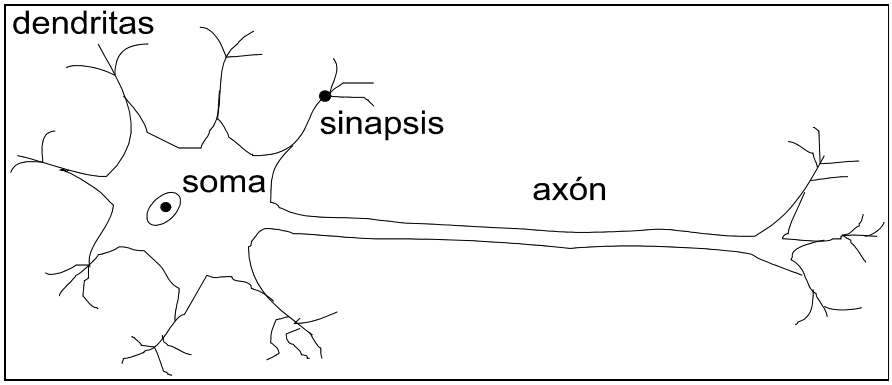
Computadores vs cerebros

Característica	Cerebro	Computador
a.- Procesador Ca	Simple Baja velocidad (ms) Muchos	Complejo Alta velocidad(µs) Unos pocos (1 o más)
b.- Memoria	Integrada en el procesador Distribuida Direccionable por contenido	Separada del procesador Localizada Normalmente no direccionable por contenido
c.-Computación	Distribuida Altamente Paralela Auto-aprendizaje	Centralizada Secuencial/muy poco paralela Programa almacenado
d.- Tolerancia a fallos	Muy robusto	Muy vulnerable
e.- Habilidades	Muy bueno en problemas de percepción	Muy bueno en problemas de manipulación numérica/simbólica
f.- Entorno de operación	No restringido. Poco definido	Bien restringido Bien definido

Neuronas biológicas I

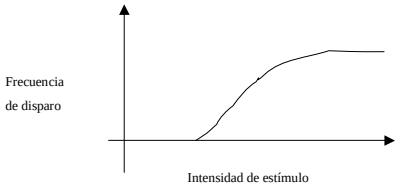
- El "padre" de las RNA es Ramón y Cajal (1888)
- Una neurona tiene un cuerpo celular, o soma, (entre 10 y 80 µm), del que surge un denso árbol de ramificaciones (dendritas) y una fibra tubular (axon, entre 100 µm y 1 metro)
- Una neurona es un procesador de información muy simple
  - Canal de entrada: dendritas
  - Procesador: soma
  - Canal de salida : axon
- Una neurona de la corteza cerebral recibe unas 10000 entradas, y envía a su vez salida a varios cientos de neuronas
- En la corteza cerebral se aprecia una organización en capas (6), y en grupos especializados
  - La unión entre neuronas se llama sinapsis. No existe contacto físico (0.2 µm)
  - Las sinapsis son conexiones direccionales (simplex), en las que la transmisión de información puede hacerse de forma eléctrica (en el interior de la neurona) o química (entre neuronas)

Neurona biológica



Neurona biológica II

- Sinapsis, o áreas de contacto electroquímico entre neuronas, realizado a través de unas sustancias específicas, neurotransmisores, de dos tipos
  - ✓ De excitación: hacen el potencial algo menos negativo (-50 mV)
  - ✓ De inhibición: aumentan el potencial negativo (-75 mV)
- La función de respuesta (frecuencia - intensidad del estímulo)



- Es indudable que la fuerza de las sinapsis constituye una forma de almacenar el conocimiento del sistema nervioso.
  - ✓ Sencillo modelo sirve para sentar las bases de las RNA

Sistemas conexionistas.

- Los tres conceptos clave que una RNA intenta emular son
  - Procesamiento paralelo
    - las diferentes neuronas están operando de forma paralela, permitiendo tratar grandes volúmenes de información en poco tiempo
  - Memoria distribuida
    - la información almacenada se encuentra en las sinapsis
  - Adaptabilidad al entorno
    - las sinapsis van variando con el tiempo, modificando su valor a partir de ejemplos
- El elemento básico de una RNA es la *neurona artificial* o *elemento de proceso (EP)*
- Varias de estas neuronas, organizadas en capas, formarán la RNA. Al añadir las interfases de entrada y salida y algunos módulos adicionales necesarios, tendremos un sistema de proceso neuronal, o sistema *conexionista*, o *neurocomputador*

Algunas definiciones.

- Red Neuronal Artificial
  - Estructura de proceso de información, paralela y distribuida, formada por Elementos de Proceso (EP)
    - Interconectados a través de canales unidireccionales llamados conexiones
    - Tiene una salida única que se distribuye sobre un número de conexiones bilaterales
    - La salida puede ser de cualquier tipo numérico (binario, bipolar, entero, real, etc).
- El proceso de información interno a cada EP puede definirse de forma arbitraria
  - Totalmente local, es decir, depender solamente de los valores de las señales que llegan al EP y de valores almacenados en su memoria local.

Historia

- McCulloch, Pitts (1943)
- Hebb (1949): Organization of Behaviour
- Rosenblatt, Wightman (1957): perceptrón.
- Widrow, Hoff (1960): Adaline
  - Poco rigor, demasiado entusiasmo, demasiado énfasis en la "analogía biológica".
- 1965-82: etapa de silencio:
  - Artículo de Minsky, Papert
  - Falta de soporte tecnológico
- 1982-86 : Ciclo de conferencias de Hopfield.
- 1986: PDP. Rumelhart, Mc. Lelland. Retropopagación.
- 1987: Primera ICNN IEEE San Diego
- 1988: Neural Networks (Primera revista) ...

Modelo de neurona artificial I

- El modelo de Rumelhart, McClelland (1986) define un elemento de proceso, o neurona artificial, como un dispositivo que a partir de un conjunto de entradas, x_i, i=1,..n (o vector x), genera una única salida y:
  - Conjunto de entradas o vector de entrada x, de n componentes
  - Pesos sinápticos w_{ij} , representan la intensidad de la interacción entre la neurona presináptica j y la postsináptica i.
  - Regla de propagación $\sigma(w_{ij},x_j(t))$, proporciona el potencial postsináptico, $h_i(t)$.
  - Función de activación, $a_i(t) = f_i (a_i(t-1), h_i(t))$, proporciona el estado de activación de la neurona en función del estado anterior $a_i(t-1)$, y del valor postsináptico $h_i(t)$.
  - Función de salida, $F_i(t)$ proporciona la salida $y_i(t) = F_i(a_i(t))$
- En resumen:
  - $y_i(t) = F_i(a_i(t)) = F_i(f_i(a_i(t-1), h_i(t))) = F_i(f_i (a_i(t-1), \sigma(w_{ij} , x_j (t)))$

Modelo de neurona artificial II

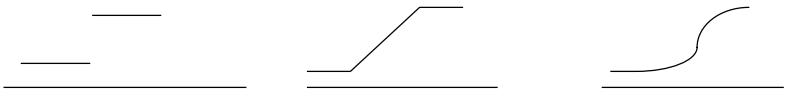
- Entradas y salidas (Tipos de señal):
  - Binarias {(0,1)}, (neurona de McCulloch-Pitts)
  - Bipolares {(-1,1)}, (neurona de Ising)
  - enteros {(…-2,-1,0,1,2,…)}, (n de Potts), variable borrosas, continuas, etc
- Regla de propagación $h_i(t) = \sigma(w_{ij}, x_j(t))$
  - La función más habitual es de tipo lineal, suma ponderada $h_i(t) = \sum w_{ij} x_j(t)$ (producto escalar del vector de entrada x por el vector de pesos w_i):

    $h_i(t) = w_i^t x$
  - La distancia euclídea entre los vectores x y w:

    $h_i(t) = \sum (x_j - w_i)2$,
  - Otros tipos de reglas
    - Distancias de Voronoi
    - Mahalanobis
    - Co-ormas triangulares, etc

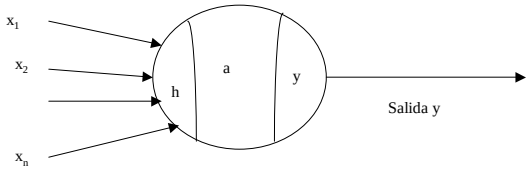
Modelo de neurona artificial III

- Función de activación (o transferencia):
  - Calcula el estado de activación actual, $a_i(t) = f_i(a_i(t-1), h_i(t))$, a partir del estado anterior $a_i(t-1)$, y del potencial postsináptico $h_i(t)$) .
  - Normalmente, se considera que el estado de la neurona no depende del estado anterior y, por tanto, $a_i(t) = f_i(h_i(t))$.
  - La función de activación se suele considerar determinista y, casi siempre, es continua y monótona creciente. Las más habituales son:
    - Signo (+1,-1)
    - Semi-lineal
    - Sigmoide.(1/(1+exp(-kx))
    - Trigonométricas: sin(), cos(), atan(), …



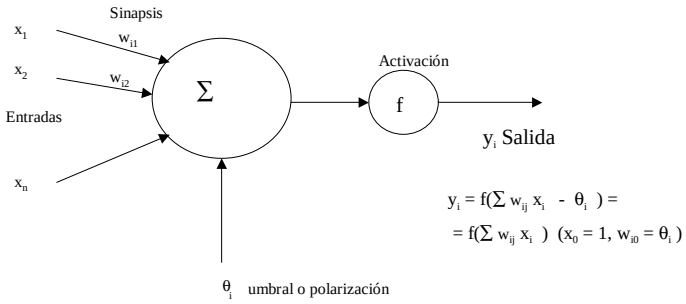
Modelo de neurona artificial IV

- Función de salida
  - Calcula la salida global de la neurona $y_i(t)$ en función del estado de activación ai (t). Casi siempre es la identidad:
    - $y_i(t) = F_i(a_i(t)) = a_i(t)$.
  - En otros casos es una función de umbral (la neurona no se dispara hasta que su estado no supere un cierto valor), o una función estocástica (máquina de Boltzmann)



Modelo estándar de neurona artificial

- El modelo descrito se simplifica



Variantes de la neurona estándar

- Neurona McCulloch-Pitts (43)
  - Tiene entradas binarias (0,1) y la función de activación es el escalón definida entre 0 y 1. Es decir

$$y_i = 1 \text{ si } \sum w_{ij}\, x_j \geq \theta_i$$

$$y_i = 0 \text{ si } \sum w_{ij}\, x_j < \theta_i$$

- Neurona sigmoidea
  - Las entradas pueden ser de cualquier tipo, continuas o discretas, pero la salida es continua con la función de activación de tipo sigmoide:

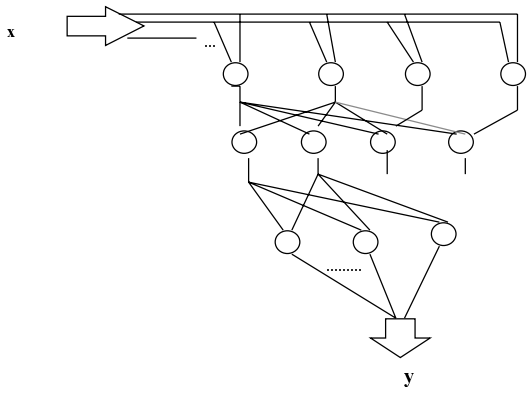
$$y = f(x) = \frac{1}{1+e^{-x}} \quad , y \in [0,1]$$

$$y = f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} = tgh(x) \quad , y \in [-1,1]$$

Arquitecturas de RNA I

- Se llama *arquitectura* de una RNA a la topología, estructura o patrón de conexiones de la red
  - Las conexiones sinápticas son direccionales, es decir, la información solamente se transmite en un sentido
- Las neuronas se agrupan en unidades estructurales llamadas *capas*. Dentro de una capa, las neuronas suelen ser del mismo tipo. Tres tipos de capas
  - De *entrada:* reciben datos o señales procedentes del entorno (pueden estar conectadas a sensores)
  - De *salida:* sus neuronas proporcionan la respuesta de la red (pueden estar conectadas a efectores)
  - *Ocultas*: no reciben, ni suministran, información al entorno
- Puede haber también conexiones intracapa, o *laterales* y conexiones de *realimentación*

Arquitecturas de RNA II



Clasificación de RNA

- Número de capas: Monocapa y Multicapa
- Flujo de datos
  - Unidireccionales , todo-adelante o estáticas (*feedforward*)
  - Recurrente o dinámicas.
- Forma de asociación
  - Heteroasociativas
    - asocian un patrón de entrada A con otro diferente B
  - Autoasociativas
    - asocian un patrón de entrada A -perturbado por ruido- consigo mismo
- Dinámica de actualización del estado
  - Sincronas: los estados se actualizan en función de un cierto reloj común a todas las neuronas. Las neuronas se van actualizando por capas, empezando por la de entrada y prosiguiendo hacia la de salida.
  - Asíncronas: Las neuronas actualizan su estado independientemente de las demás.
  - Estocásticas: tanto en el instante de actualización, como en el valor del estado, existe alguna componente aleatoria.

Aprendizaje I

- Proceso de actualizar los pesos (y eventualmente la arquitectura) de forma que la red pueda llevar a cabo de forma efectiva una tarea determinada
- Normalmente, la modificación se restringe al valor de los pesos, dado que la relación topología-prestaciones no está muy clara
  - *Paradigma de aprendizaje*
    - Información disponible para la red (modelo del entorno de operación)
  - *Regla de aprendizaje*
    - Principios que gobiernan el aprendizaje
  - *Algoritmo de aprendizaje*
    - Procedimiento numérico de ajuste de los pesos.
- Hay cuatro paradigmas de aprendizaje
  - Supervisado, No supervisado, Híbrido, Reforzado

Aprendizaje II

- Supervisado
  - Sea E(W) un funcional del error esperado de la red en función de los pesos sinápticos, W. Si la red tiene n neuronas de entrada y m de salida, se pretende estimar una cierta función

$$f : \mathrm{R}^{n} \rightarrow \mathrm{R}^{m}$$

  - A partir de parejas de ejemplos (x,y)
    - Minimizar E(W) en función de los errores entre las salidas de la red, y las salidas deseadas
- No supervisado o auto-organizado
  - Estimación de la densidad de probabilidad p(x) de la distribución de patrones x del espacio de entrada
    - Se presentan a la red un conjunto de patrones sin adjuntar la respuesta deseada
    - La red deberá extraer rasgos o agrupar patrones (*clustering*) que sean similares.
- Híbrido: combina los dos anteriores.
- Reforzado o de premio-castigo.

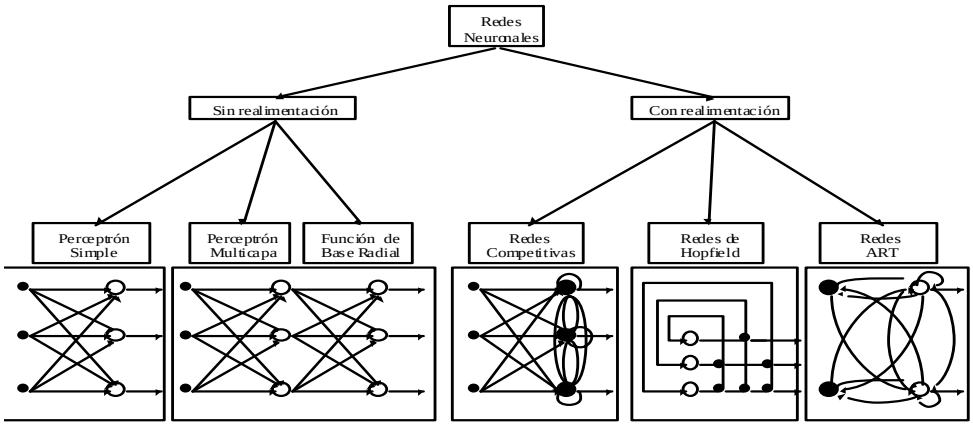
Aprendizaje III

- *La teoría del aprendizaje* contempla tres aspectos
  - Capacidad
    - Nº patrones puede "aprender" la red
  - Complejidad de muestras
    - Nº de patrones de entrenamiento necesarios para entrenar la red de forma que se garantice una generalización válida.
  - Complejidad computacional: referente al algoritmo de aprendizaje
- Hay cuatro tipos básicos de algoritmos de aprendizaje
  - Basados en la corrección del error (gradiente, retro-propagación, etc)
  - Boltzmann, Hebb
  - Competitivo
- Una vez entrenada, la red congela sus pesos y funcionará en modo de *recuerdo o ejecución*
  - Se suministra una entrada y la red responde con una salida

Taxonomía de RNA

Paradigma	Regla	Arquitectura	Algoritmo de aprendizaje	Aplicaciones
Supervisado	Error Boltzmann Hebb Competitivo	Perceptrón Recurrente Multicapa Una capa ART	Perceptrón, retropropagación, Boltzmann Análisis discriminante LVQ ARTMap	Clasificación, aproximación, control, predicción, análisis de datos, compresión,
No supervisado	Error Hebb Hebb Competitivo	Multicapa Estática Hopfield SOM	Sammon Analisis de componentes memoria asociativa SOM	Analisis y compresión de datos, memoria asociativa, agrupamiento, etc
Hibrido	Error y competitivo	RBF	RBF	Clasificación, aproximación, control, predicción,

Taxonomía de las RNA (II)



Perceptrón simple I

- Debido a Rosenblatt (1959), sus neuronas son nolineales y las entradas son binarias.

$$\mathbf{y}_i(t) = f(\sum_{j=1}^{n} \mathbf{w}_{ij}\mathbf{x}_j - \theta_i) \quad \forall i \in [1, m]$$

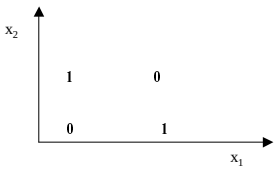
  - La función de transferencia de las neuronas, f(), es del tipo todo-nada.
- El perceptrón simple, al ser binario, realiza una clasificación. Por ejemplo, si n=2 y m=1 , la salida de la red sería

$$y = \begin{matrix} 1 \text{ si } w_1 x_1 + w_2 x_2 \geq \theta \\ 0 \text{ si } w_1 x_1 + w_2 x_2 < \theta \end{matrix}$$

  - de forma que hay una recta (= θ) que separa la clase 0 de la clase 1. En el caso de n>2 la recta sería un hiperplano.

Perceptrón simple II

- Hay problemas en los cuales las clases no pueden separarse por un hiperplano (no linealmente separables).
  - Ejemplo XOR

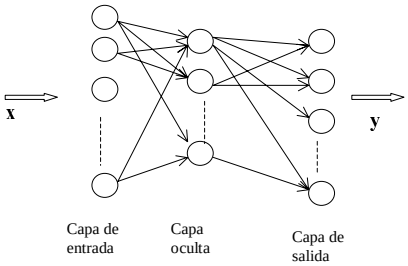


x_1	x_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

  - No existe ninguna recta que separe los 0 de los 1
- El perceptrón solamente sirve para clasificar problemas que sean linealmente separables.
- La regla de aprendizaje
  - $\Delta \mathbf{W}_i = \varepsilon(\mathbf{y}_i' - \mathbf{y}_i)\mathbf{x}_i$ siendo ε el parámetro *factor de aprendizaje*

MLP I

- El Perceptrón Multicapa (MLP)
  - Debido al grupo PDP (1986)
  - Extensión y generalización del perceptrón simple
    - Se añaden una o más capas ocultas, se permiten entradas de cualquier tipo
    - Las funciones de transferencia suelen ser de tipo sigmoide:



Normalmente, las neuronas de la capa oculta son de tipo sigmoide y las de la capa de salida son lineales. Con ello:

$$\mathbf{y} = (\mathbf{w}_{o}{}^{T}\mathbf{z} - \theta_o) = (\mathbf{w}_{o}^{T} f(\mathbf{w}_{e}^{T}\mathbf{x} - \theta_e) - \theta_o)$$

donde $\mathbf{z}$ representa la salida de la capa oculta y $\mathbf{w}_{eo}$ los pesos de la capa de entrada-capa oculta

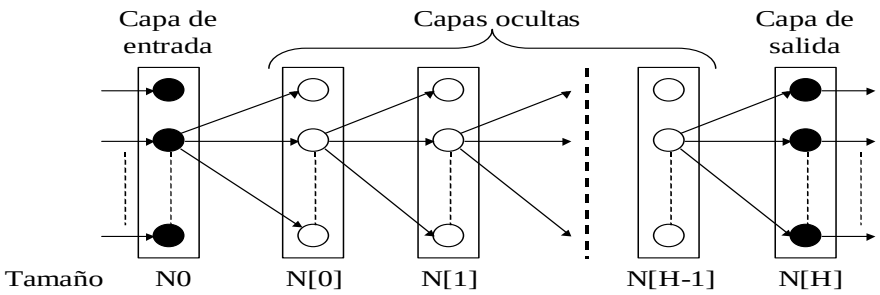
MLP II

- Las redes MLP intentan resolver el problema general de la correspondencia (mapping)
  - ✓ Obtener f : $A \subset R^n \rightarrow R^m$
  - ✓ partiendo de un conjunto de ejemplos (x_i,y_i), i=1,...N, donde y_i=f(x_i)
- En general, una red sometida a una entrada x genera una salida y'(x,W) que será distinta de y=f(x)

$$\mathbf{F}(\mathbf{x},\mathbf{W}) = (\mathbf{f}(\mathbf{x}) - \mathbf{y}'(\mathbf{x},\mathbf{W}))^2$$

- El funcionamiento de la red se suele expresar en función del error cuadrático medio $\mathbf{E}(\mathbf{W}) = lim(1/L)\sum_k \mathbf{F}_k(\mathbf{x}_k, \mathbf{W}_k)$
  - ✓ calculado sobre un conjunto de ejemplos de prueba diferentes del conjunto de aprendizaje
- Son posibles otras medidas del error (absoluto, medio absoluto, etc), pero el MSE tiene grandes ventajas

MLP III (Varias capas ocultas)



MLP IV

- Teorema de Kolmogorov: Dada cualquier función continua
  - **f** :(0,1)^n -->**R^m** , **f(x) = y**
  - puede implementarse de forma exacta por una RNA de 3 capas, de propagación hacia adelante, con n elementos de proceso en la capa de entrada, m en la de salida y (2n+1) en la capa oculta

$$z_k = \sum_{j=1}^{n} \lambda^k \Psi (x_j + \varepsilon k) + k \quad k = 1, \dots 2\,n + 1 \quad \text{Capa oculta}$$

$$y_j = \sum_{k=1}^{2n+1} g_j (z_k) \quad j = 1, \dots m \quad \text{Capa de salida}$$

donde

λ = constante real

Ψ = función real , monótona creciente

ε = constante racional

g_i = función real , continua (transferencia)

Por desgracia , el teorema es solamente de existencia .

No nos dice cómo implementar las funciones

MLP V

- La funcionalidad de los MLP solamente fue posible cuando se descubrió la forma de entrenarlos: algoritmo de retropropagación (back-propagation, BP) Es uno de los desarrollos más importantes en neurocomputación, debido a Werbos (84), Rumelhart (85) y Parker (89)
- El peso w_{ij} (E.P. j al E.P i) se ajusta por

$$\Delta w_j (t) = \gamma \delta_i (t) a_j (t) \quad \text{donde}$$

a_j = Salida del elemento de proceso j

δ_i = Derivada de la señal de error en la capa de salida con relación a la función de activación en el elemento de proceso i (producto escalar de vector de entrada por vector de pesos)

- Si el elemento de proceso i-ésimo es de salida

$$\delta_i (t) = (y'_i (t) - y_i (t)) f'_i$$

donde f'_i es la derivada de la función de transferencia

MLP VI

- El problema es propagar estas expresiones de la capa de salida hacia las capas anteriores

$$\delta_i(t) = (y'_i(t) - y_i(t))\, y_i(t)(1 - y_i(t))$$

En las capas interiores , la señal de error es :

$$\delta_i(t) = f'_i \sum_{h=1}^{N} \delta_h(t) w_{hl}$$

con N = número de elementos de proceso en la capa destino.

Si la función de transfere ncia es sigmoide :

$$\delta_i(t) = a_i(t)(1 - a_i(t)) \sum_{h=1} \delta_h(t) w_{hi}$$

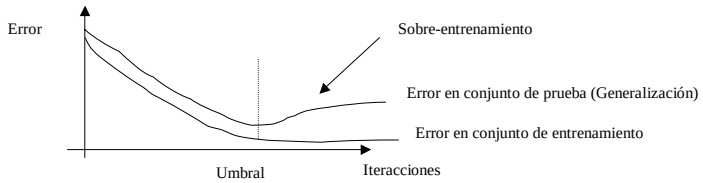
donde a_i es la salida del sigmoide del elemento i - esimo

- Estas ecuaciones forman la llamada regla delta generalizada.
- En la práctica, para acelerar la convergencia, se suele introducir un término de momento calculado de forma más o menos empírica.

$$\Delta w_{ij}(t) = \gamma_i \delta_i a_i(t) + \alpha \Delta w_{ij}(t-1)$$

MLP VII

- La regla delta generalizada no es más que un algoritmo del gradiente, con todas sus ventajas e inconvenientes.
- La forma de actuación es:
  - Aplicar una pareja de ejemplo (**x,y**)
  - Propagar hacia adelante y calcular la salida real **y'**
  - Calcular el error, diferencia entre **y** e **y'**.
  - Aplicar el algoritmo e ir modificando los pesos hacia atrás.
  - Repetir hasta que se alcance un cierto valor mínimo del error, bien con el conjunto de entrenamiento o con el de prueba.



MLP VIII

- Si la red se entrena "demasiado" pierde capacidad de generalización
  - "Sobreentrenamiento"
- Si el número de neuronas en la capa oculta es excesivo
  - "Sobre-ajuste" que implica también pérdida de capacidad de generalización.
- El número de patrones de entrenamiento necesarios depende del número de pesos y del error deseado. (Baum 89)
  - Nw el número de pesos de la red y ε el error deseado
    - Nº patrones es Nw/ ε
    - Así, si se desea un error del 1% (0.01) y se tienen 200 pesos
      - el número teórico de patrones necesarios sería de 20000 (muy conservador)
  - Formas de disminuir esta cifra
    - Reducir el número de entradas (Técnicas estadísticas:análisis de componentes principales)
    - Podado de pesos (pruning) , etc
- No existe todavía una técnica que indique la arquitectura de red óptima para un problema dado
  - Los algoritmos constructivos, o las arquitecturas evolutivas parecen ser un buen camino para este problema.

MLP IX

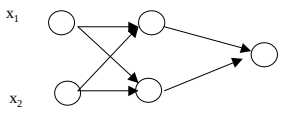
- Problemas importantes del MLP y el algoritmo de BP
  - El tiempo de computación necesario para el entrenamiento
  - La parálisis de la red
    - si los pesos alcanzan valores muy altos, la forma de la función sigmoide hace que la salida de la capa oculta sea muy próxima a 0 o 1
    - De las ecuaciones del algoritmo de BP, se deduce que el incremento de pesos en esos casos es casi nulo
      - Una parálisis del entrenamiento
      - Una posible solución consiste en añadir una pequeña cantidad de ruido a la salida de la neurona.
  - Minímos locales
    - La función de error de una red compleja está llena de valles y picos
      - La propia naturaleza del algoritmo puede producir la caída en uno de los valles que no es, necesariamente, el mínimo
      - Una posible solución parece ser el aumentar el número de neuronas ocultas pero esto, como se vió, produce otros efectos secundarios indeseables.

Aplicación del MLP. OR exclusivo

- Sea el problema del XOR entre dos entradas:

x1	0	0	1	1
x2	0	1	0	1
y=XOR	0	1	1	0

- Un posible red para este problema tendría 2 neuronas de entrada, 1 de salida (y) y una capa oculta por ejemplo con 2. Los patrones de entrenamiento serían parejas ((x1 , x2),y) siguiendo la tabla anterior



Hay un total de 6 pesos a entrenar.

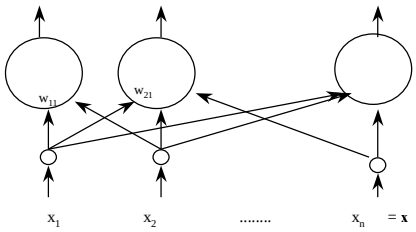
Una vez entrenada, la red responde con una salida , y, que es el XOR de las dos entradas.

Redes Autoorganizadas

- En las redes autoorganizadas (no supervisadas)
  - El entrenamiento se realiza presentando solamente entradas
  - La red debe descubrir patrones o características significativas en los datos de entrada, e incorporarlas a sus pesos (auto-organización)
- El tipo de procesamiento que una red autoorganizada
  - Análisis de similitud de patrones de entrada, componentes principales (PCA)
  - Agrupamiento (clustering), clasificación
  - Memoria asociativa
  - Codificación, Mapas de características
- En las redes autoorganizadas, se emplean dos reglas de aprendizaje:
  - aprendizaje hebbiano (muy poco usado)
  - aprendizaje competitivo : es la base de las redes autoorganizadas más empleadas: ART (Grossberg 88), Neo-cognitron (Fukushima 80) y mapas autoorganizados (SOM, Kohonen 82, 89)

Aprendizaje Competitivo I

- La formulación original es muy discutida
  - ✓ Aparentemente debida a T.Kohonen (1988).
- La idea básica es que una capa de elementos de proceso modifique sus pesos en R^n de forma que su distribución de frecuencias sea proporcional a la densidad de probabilidad de los vectores de entrada



Todas las neuronas reciben las mismas entradas .

Cada neurona tiene su propio vector de pesos

Aprendizaje Competitivo II

- La capa consta de N elementos de proceso, cada uno de los cuales recibe las mismas n señales x_i, i=1,2,...N
- Cada EP calcula su intensidad de entrada I_i :
  - ✓ $I_i = D(w_i,x)$ donde D( ) es una función de medida de distancia entre los vectores de entrada x y de pesos w_i. En la práctica, se suele usar la distancia euclídea en el espacio R^n.
- Una vez que cada unidad ha calculado su intensidad de entrada I_i, arranca una competición para ver qué E.P tiene la mínima intensidad
  - ✓ El vector de pesos es el más próximo a la entrada
- Existen varios métodos de implementar esta competencia
  - ✓ Por inhibición lateral
  - ✓ Por unidad de planificación
- Una vez que se ha determinado el elemento de proceso ganador, su salida z_i se hace igual a 1, y el resto igual a 0

Aprendizaje Competitivo III

- Se aplica entonces la ley de Kohonen

$$w_i(k+1) = w_i(k) + a(x - w_i(k))z_i$$

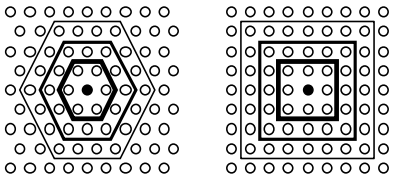
  - donde a es una constante entre [0,1] (factor de aprendizaje)
- Solamente el EP ganador actualiza sus pesos
  - El vector de pesos de la neurona ganadora se "mueve" un poco hacia el vector de entrada
  - A medida que se introducen entradas, los vectores de pesos van formando nubes que siguen la distribución de las entradas.
- Sin embargo, la ley de Kohonen no produce, en general, un conjunto de vectores de peso equiprobables, i.e., un vector de entrada x no tiene la misma probabilidad de estar más cercano a cualquiera de los vectores de peso.
  - Es decir, sería deseable que dado x en R^n todas las neuronas tuviesen la misma probabilidad de resultar ganadoras

Aprendizaje Competitivo IV

- El Mapa Autoorganizado (SOM) utiliza una variante del aprendizaje competitivo
  - Actualizar los pesos de la neurona ganadora
  - Actualiza también los de sus vecinas, aplicando una cierta función de vecindad, H(i,g) , entre la neurona ganadora g y sus vecinas i.
- H(i,g) decrece con la distancia (i-g) y depende de un parámetro llamado radio de vecindad, R(t), que es un número entero
  - Las neuronas situadas a una distancia mayor a R(t) no modifican sus pesos.
- Además, tanto el factor de aprendizaje, a como R(t) van decreciendo a medida que progresa el entrenamiento.

Aprendizaje Competitivo V

- Tipos de vecindad en los SOM
  - En el SOM, las neuronas se disponen en una capa pero con estructura de 1 dimensión (lineal), o de 2 (planar) o de 3 (espacial). Todas las neuronas reciben las mismas entradas.
  - Como resultado del entrenamiento, las neuronas se van sintonizando de forma selectiva a los patrones de entrada
    - Un SOM forma un mapa topográfico de los patrones de entrada, en el cual las coordenadas de las neuronas corresponden a características intrínsecas en los patrones de entrada. (Kohonen, 90).



(a) Hexagonal **(b)** Rectangular

Aprendizaje Competitivo VI

- Parece existir una analogía entre la forma de operar de un SOM y el cerebro: las diferentes áreas del cerebro están organizadas en mapas computacionales organizados topológicamente
- En esencia, un SOM realiza la proyección de los datos de entrada de un espacio n-dimensional a una rejilla de dimensión baja (1, 2 o 3) pero de forma que los patrones cercanos en el espacio inicial se corresponden con neuronas cercanas en el mapa
- Algunas aplicaciones de los mapas de Kohonen son
  - Reducción de dimensionalidad de datos
  - Monitorización de procesos
  - Agrupamiento de datos (clustering)
  - Cuantificación vectorial

Redes Recurrentes I

- Redes neuronales recurrentes o dinámicas
  - ✓ Existe realimentación entre las neuronas
  - ✓ Al introducir una entrada, la información se propaga hacia adelante y hacia atrás
  - ✓ Eventualmente, la evolución se detendrá en algún estado estable, punto fijo o atractor
    - ✓ En otros casos, es posible que la red no se detenga (inestabilidad)
- Las redes recurrentes deberían cumplir tres objetivos:
  - ✓ Dado cualquier estado inicial, deben converger siempre a un estado estable.
  - ✓ El dominio de atracción de cada estado estable debe estar perfectamente delimitado, y cumplir algún criterio de métrica (por ejemplo, que el estado final sea el más cercano al inicial).
  - ✓ Debe poder tener cualquier número de estados estables.
- Dado que las RNA recurrentes son sistemas dinámicos no lineales, su análisis es difícil. Existen teoremas de estabilidad (Cohen-Grossberg 89, Kosko 92)

Redes Recurrentes II

- Red de Hopfield
  - ✓ Es una red diseñada por Hopfield en 1982. Completamente conectada, con una matriz de pesos fija, W, simétrica y de diagonal nula

    $w_{ii} = 0 \quad w_{ij} = w_{ji}$
  - ✓ La salida de los elementos de proceso es bipolar, con función de transferencia dada por

$$x_i(t+1) = 1 \text{ si } \sum w_{ij} x_j(t) > T_i$$
$$x_i(t+1) = x(t) \text{ si } \sum w_{ij} x_j(t) = T_i$$
$$x_i(t+1) = -1 \text{ si } \sum w_{ij} x_j(t) < T_i$$

  - ✓ Los elementos de proceso se actualizan de uno en uno, de forma asíncrona o serie (*Glauber*) o bien en forma paralela (*Little*) actualizándose varias neuronas a la vez. Ambos métodos producen resultados diferentes
  - ✓ La red arranca en un estado inicial, x_0 , y va evolucionando hasta llegar, eventualmente, a un estado estable

Redes jerárquicas I

- Se denominan así a redes con un gran número de capas
  - ✓ Conexiones muy escasas y localizadas
  - ✓ Los elementos de proceso reciben conexiones de un subconjunto de elementos de proceso de capas anteriores.
- Se derivan de una idea original de Fukushima (cognitron, 1975), modificada en 1988 (neocognitron).
- El objetivo del Neocognitron es el reconocimiento de caracteres escritos, independiente de su posición.
- La red original tiene unos 50000 EP, y unas 14 millones de conexiones, organizadas en 156 capas.
- La capa de entrada contiene una imagen digitalizada (19x19) del carácter a reconocer.
- La capa de salida contiene tantos elementos de proceso como caracteres a reconocer (10, en la versión original)
- Existen, entre ellas, un total de 154 capas ocultas.

Redes jerárquicas II

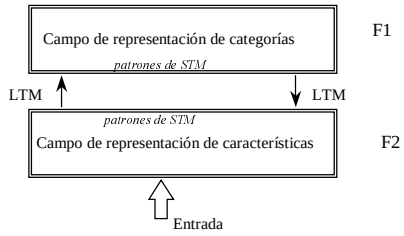
- Se distinguen dos tipos de elementos de proceso: excitadores e inhibidores.
- La idea básica es que cada capa se "especialice" en el reconocimiento de determinadas características o propiedades de la imagen de entrada: trazos horizontales, verticales, oblicuos, etc.
- A medida que se "avanza" por las capas, esas características van siendo más y más globales, de suerte que pueden llegar a abstraerse de la posición física en que se encuentran sobre la imagen.
- La ley de aprendizaje solamente afecta a los elementos excitadores, cuyos pesos se modifican según la llamada ley de "pinocho", por la cual sus valores absolutos pueden crecer de forma indefinida, pero no la salida de los EP.
- El neocognitrón ha demostrado buenos resultados en reconocimiento de caracteres independiente de posición, y con buena inmunidad al ruido. Su mayor inconveniente es la complejidad de su implementación debido al elevado número de neuronas.

Redes jerárquicas III. ART I

- La Teoría de Resonancia Adaptativa, fué desarrollada por Grossberg (76). Da lugar a un tipo de red jerárquica, recurrente y con aprendizaje no supervisado
- Tiene una fuerte inspiración biológica, en el sentido de que resalta tres propiedades importantes
  - Normalización
    - Los sistemas biológicos se adaptan muy bien a grandes cambios del entorno.
  - Realce de contraste
    - La posibilidad de distinguir pequeñas diferencias en patrones de entrada puede ser de importancia vital.
  - Memoria de corto plazo (STM)
    - Antes de decodificar un patrón de entrada, se almacena en la memoria de corto plazo. La memoria de largo plazo (LTM) implementa un mecanismo de estimulación (por ejemplo, clasificación), mientras que la de corto plazo se usa para producir cambios graduales en la LTM.
- La red ART consta de dos capas, F1 y F2, que se conectan entre sí a través de la LTM. La entrada se recibe en F1, y la clasificación se hace en F2

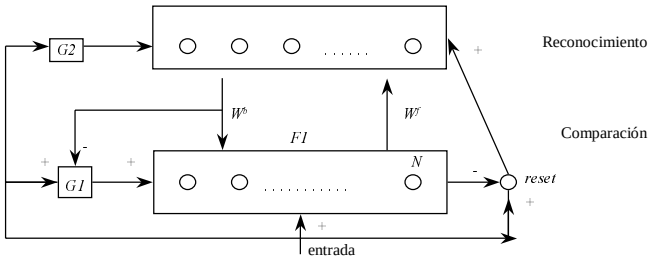
ART II

- La clasificación no se hace directamente. Primero, tiene lugar una extracción de características, que da lugar a una activación en el *campo de representación de características.*
- Las *expectativas*, almacenadas en las conexiones LTM, trasladan el patrón de entrada al *campo de representación de categorías*. La clasificación se compara con la expectativa. Si coincide, se aumentan las expectativas (pesos). Si no, se rechaza la clasificación.



ART III

- El modelo simple ART1 tiene dos capas de neuronas binarias (0 o 1):
  - La capa F1 o de *comparación.*
  - La capa F2 o de *reconocimiento.*
- Cada neurona en F1 se conecta a todas las de F2 a través de una matriz de pesos con valores *reales* ,LTM, W^f. Y , viceversa, las de F2 se conectan a F1 a través de una matriz de pesos *binarios* W^b.



ART IV

- Cada neurona en la capa F1 recibe tres entradas
  - Una componente del patrón de entrada.
  - Una componente del patrón de realimentación.
  - Una ganancia G1
- La salida de la neurona es uno solamente si al menos dos de las entradas son 1 (regla de los dos tercios)
- Las neuronas de F2 calculan el producto escalar de su vector de pesos $\mathbf{w}^f$ y el patrón de salida de F1. La neurona ganadora inhibe al resto.
- G2 es el "or" de todos los elementos del patrón de entrada **x**
- G1 es igual a G2, excepto si la salida de F2 contiene al menos un 1, en cuyo caso vale 0.
- La señal de reset se envía a la neurona ganadora de F2 si el vector **x** y la salida de F1 difieren más de un cierto valor de *vigilancia*.

ART V

- La red arranca fijando la entrada en F1. Como F2 es cero, tanto G1 como G2 son 1 y por tanto , la salida de F1 es igual a la entrada:
  - 1.- Inicialización

$$w_{ji}^{b}(0) = 1$$

$$w_{ji}^{f}(0) = \frac{1}{1+N}$$

Fijar $\rho \in (0,1)$

  - 2.- Aplicar un patrón de entrada **x**
  - 3.- Calcular los valores de activación de las neuronas de F2

$$a'_i = \sum_{j=0}^{N-1} w_{ij}^{f}(t) x_j$$

  - 4.- Seleccionar la neurona ganadora de F2, k
  - 5.- Prueba de vigilancia. Si

$$\frac{\mathbf{w}_k^b(t)\cdot\mathbf{x}}{\mathbf{x}\cdot\mathbf{x}} > \rho \text{ ir a paso } 7$$

  - 6.- Desactivar la neurona k. Ir a paso 3

ART VI

  - 7.- Ajustar para todo l, entre 0 y N:

$$w_{lk}^{b}(t+1) = w_{lk}^{b}(t) x_l$$

$$w_{kl}^{f}(t+1) = \frac{w_{kl}^{b}(t) x_l}{\frac{1}{2} + \sum_{j=0}^{N-1} w_{jk}^{b}(t) x_j}$$

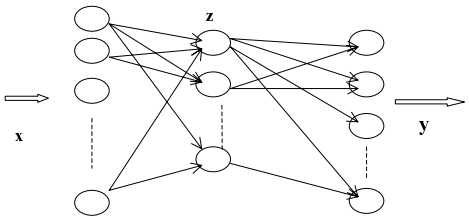
  - 8.- Reactivar las neuronas de F2. Ir a paso 2
- De esta forma, la red ART puede almacenar y recuperar patrones binarios.
  - ART2 y 3
    - Admiten patrones analógicos continuos.
    - La red no solamente clasifique patrones ya conocidos (Clustering) sino que en caso de mala clasificación puede "crear" nuevas clases, resolviendo así el dilema de la *plasticidad-estabilidad.*
- Últimamente, se ha extendido la teoría para incluir lógica borrosa
  - Fuzzy ArtMap

Funciones de Base Radial (RBF) I

- Las RBF (Moody, Poggio 90) son un modelo híbrido que incorpora aprendizaje supervisado y no supervisado
- Como los MLP con BP, las RBF permiten realizar modelados de sistemas no lineales arbitrarios. Pero los tiempos de entrenamiento son mucho más reducidos (Warwick 95)
- Una red RBF tiene tres capas
  - Entrada
    - Simplemente envían la entrada a la capa oculta
  - Salida
    - Lineales
  - Oculta
    - Operan en base a la distancia que separa el vector de entrada del vector de pesos de la neurona (*centroide*) . A esta distancia aplican una función *gaussiana*.
- En el RBF, la respuesta de las neuronas de la capa oculta es localizada
  - Solamente responden a entradas situadas cerca de su centroide

Funciones de Base Radial (RBF) II

- La estructura general es:



Capa de entrada

Capa oculta (Neuronas radiales)

Capa de salida: Neuronas lineales.

Las neuronas de la capa oculta calculan la *distancia r* de su vector de pesos (centroide) al vector de entrada **x**. La salida de la neurona se calcula a partir de una función de activación llamada función radial. La más empleada es la función gaussiana:

$$\Phi(r) = e^{-r^2/2\sigma^2}$$

El parámetro σ (*de escala*) mide la anchura de la gaussiana y equivale al radio de influencia de la neurona en el espacio de entradas: cuanto mayor sea σ mayor es el radio de influencia.

Funciones de Base Radial (RBF) III

- Así, la salida de la neurona j de la capa oculta será

$$z_j = e^{-r_j^2/2\sigma_j^2}$$

- Si el vector de entradas x coincide con el centroide de la neurona j, r_j será muy pequeño y la salida z_j será 1: la neurona se activa. Por el contrario si x está muy alejado del centroide, r_j será grande y la salida z_j será casi nula
- Una vez calculadas las salidas de la capa oculta, las salidas de la red se calculan de forma lineal

$$y_k = \sum_j w_{kj} z_j + \theta_k = \sum_j w_{kj} \Phi(r_j) + \theta_k$$

- Cada neurona de la capa oculta se ocupa de una zona del espacio de entradas. El conjunto de neuronas debería cubrir totalmente la zona de interés

Aprendizaje en RBF-I

- Dado que cada neurona radial cubre una parte del espacio de entradas, el número de neuronas de la capa oculta depende de la distribución y de la dimensión del espacio de entradas
- El Algoritmo jerárquico Auto-organizado, parte de un número reducido de neuronas. Si algún patrón de entrada no activa suficientemente ninguna de las neuronas, se añade otra nueva hasta disminuir un cierto umbral de error.
- No obstante, el número de neuronas de la capa oculta suele fijarse por método de prueba y error.
- El aprendizaje se puede hacer por un método del gradiente, de forma análoga al algoritmo de retropropagación. Sin embargo, se suele emplear un aprendizaje por etapas: se entrena primero la capa oculta y a continuación la capa de salida.
- El entrenamiento de la capa oculta supone determinar:
  - ✓ Los vectores de peso de las neuronas (centroides)
  - ✓ Los parámetros de escala σ

Aprendizaje en RBF-II

- Para determinar los centroides se suele aplicar el algoritmo de las k-medias (*k-means*) que es un algoritmo no supervisado
  - ✓ Elegir los valores de los k centroides de partida. Pueden ser, por ejemplo, los primeros k patrones de aprendizaje
  - ✓ Para cada patrón de entrada, determinar la neurona más cercana y asignársela
  - ✓ Recalcular el nuevo centroide de la neurona como promedio de los vectores de entrada que se le han asignado
  - ✓ Parar si los cambios en todos los centroides no superan un valor de parada. Volver a 2 en caso contrario
- Puede demostrarse que este algoritmo es equivalente a la regla de aprendizaje competitivo de Kohonen (sin relaciones de vecindad)
- Una vez calculados los vectores de peso, los parámetros de escala, σ se calculan por un método heurístico
  - ✓ Promedio de las distancias de los patrones asignados al centroide

Aprendizaje en RBF-III

- Una vez entrenada la capa oculta, la capa de salida (lineal) se entrena usando la regla del asociador lineal. Si es y' la salida de la red e y la salida deseada

$$w_{kj}(t+1) = w_{kj}(t) + \varepsilon(y_k - y'_k)\Phi(r_j)$$

- Este aprendizaje por etapas produce una notable aceleración
  - ✓ Factor de 3
- Las RBF se aplican al mismo tipo de tareas que las MLP
  - ✓ Comparación
    - ✓ Tiempos de ejecución/Entrenamiento
      - RBF mayor rapidez de entrenamiento
      - RBF más lentas en ejecución debido a que, normalmente, necesitan muchas más neuronas ocultas que los MLP
    - ✓ MLP construyen aproximaciones globales a la correspondencia no lineal entrada-salida
      - Pueden hacer buenas generalizaciones a regiones del espacio de entrada donde hay pocos patrones
    - ✓ RBF construyen aproximaciones locales
      - Capacidad de generalización es pobre.

Implementaciones

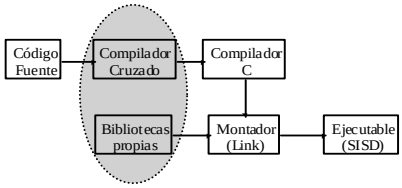
- Se suele hablar de tres niveles de implementación
  - ✓ Simulación por software en máquinas SISD (*neurosoftware*)
  - ✓ Emulación hardware en máquinas MIMD (*neurocomputadores)*
  - ✓ Implementación hardware (*neurochips*)
- Mayoría del neurosoftware
  - ✓ Ejecutado sobre máquinas paralelas
  - ✓ La distinción entre las dos primeras categorías no está tan clara

Neurofore caster (Comercial) Accel Infortech	Clasificación y predicción de series temporales.
Nestor Learning Systems (Comercial) Nestor	reconocimiento de imágenes y caracteres
WAND (comercial) Novel Technical Solutions & Imperial College of London	Reconocimiento de imágenes
SNNS (libre distribución) Univ. De Stuttgart, Facultad de Informatica ftp.informatik.uni-stuttgart.de/pub/SNNS	El software de simulación de RN más completo hasta la fecha. Disponible para sistemas Unix con X. Existe una versión para W95 y NT. Exige un servidor X en el PC.

Neurosoftware I

- La mayoría del neurosoftware es de libre disposición
  - ✓ aspirine : pt.cs.cmu.edu (128.2.254.155)
  - ✓ Xerion: me.uta.edu (129.107.2.10)
  - ✓ planet: me.uta.edu (129.107.2.10)
  - ✓ genesis: genesis.cns.caltech.edu (131.215.137.64)
  - ✓ winNN:ftp.cc.monash.edu.au
- El neurosoftware es la técnica de simulación más simple y menos costosa
  - ✓ No supone un gran esfuerzo la construcción de códigos fuente para los tipos más habituales de redes
  - ✓ Sirve de banco de pruebas
    - ✓ Antes de manejar (adquirir) otro tipo de hardware más específico, conviene asegurarse de si una solución basada en RNA es apropiada para un problema concreto.

Neurosoftware II



En algunos casos (aspirine, planet, etc) el neurosoftware incorpora un lenguaje de definición de la red neuronal y un compilador cruzado que los traduce a un lenguaje de propósito general (normalmente C). Un vez compilado y montado, se obtiene un código para ejecutar o entrenar la red.

La máquina destino puede, en algunos casos, ser una máquina paralela (CM-1) o bien puede adaptarse el software a entornos paralelos (PVM, MPI, etc).

Neurohardware

- Existen una serie de procesadores paralelos de propósito general que pueden emplearse en RNA. Se están usando tres tipos de enlaces con el host
  - Como periférico (SCSI, RS-232,etc)
  - Orientado a bus, con mapa de memoria (ISA, EISA, S-BUS)
  - Como nodo en Red
- Como ejemplo, la tarjeta Balboa860, de Hecht-Nielsen Co, basada en el I860, tiene un precio de unos 10000 US$
- Todos ellos pueden considerarse, todavía, en un estadio inicial de desarrollo, con costes muy elevados para las prestaciones que ofrecen.
- La mejor relación precio/prestaciones está siendo ofrecida por los sistemas basados en transputers (Inmos)

Neurochips I

- En principio, la regularidad y simplicidad en el planteamiento de las neuronas artificiales las hace muy adecuadas para su integración en chips
- Como siempre, existe aquí un compromiso entre
  - Dispositivos de propósito general, obligatorios para desarrollos rápidos, evaluaciones y educación
  - Dispositivos especializados, necesarios para investigación básica y aplicada
- El cálculo de los valores de activación de la RNA
  - Vector de estado x matriz de pesos puede paralelizarse con relativa facilidad.
- El cálculo del vector de estado de las neuronas
  - Función de transferencia (valores de activación) puede hacerse en paralelo sin demasiados problemas

Neurochips II

- Programa WISARD en el Imperial College (Londres)
  - Pioneros en la línea de neurochips digitales
  - Dedicadas al procesamiento de imagen
- WSI de Hitachi
  - Integrar en un solo chip 576 neuronas y 36K pesos
  - Un sistema con ocho de estos chips, en la fase de aprendizaje de un MLP puede conseguir hasta 2.3G CUPS (actualizaciones de pesos/sg).
- Neurochip ETANN (Electrically Trainable Analog Neural Network)
  - Tecnología analógica. (Intel)
    - 64 neuronas con 10K pesos y un rendimiento de hasta 2G CPS (conexiones/sg) en la fase de recuperación.
- Estimaciones año 98
  - 30% neurochips, un 20% con neurocomputadores y todavía quede un 40% con neurosoftware. El 10% restante con dispositivos ópticos.

Campos de Aplicación

- Clasificación de patrones
  - Asignar a cada patrón de entrada (vectores en un espacio de características) una etiqueta de clase. Ejemplos: reconocimiento de habla, de caracteres, clasificación de ECG, EEG, células , etc
- Agrupamiento, categorización
  - Explorar las posibles similitudes entre los patrones de entrada y crear las correspondientes agrupaciones. Ejemplo: compresión de datos, análisis exploratorio de datos, minería de datos.
- Aproximación funcional
- Predicción
  - Dadas n muestras y_i, de una serie temporal, predecir la siguiente (o siguientes)
- Optimización
- Control
- Memorias asociativas,...

Algunos ejemplos reales

- Lectura de códigos postales
- Clasificación de patatas
- OCR, MICR
  - OCR japonés: Sharp (prototipo)
  - MICR inglés: Verifone(1992)
- Identificación de ecos de sonar
- Sensores virtuales: partiendo de medidas directas de planta, obtener una estimación del valor de alguna variable no medible.
  - Ejemplos: Eastman, espesor de la capa sensible del carrete fotográfico.
- Cancelación de vibraciones
- Reconocimiento de habla
- Reconocimiento de locutor
- Diagnóstico de fallos: Nippon Steel-Fujitsu

Aplicaciones en Robótica (Manipuladores)

- Se emplean para resolver problemas de
  - Cinemática directa: dadas las posiciones de cada eje del robot , q1,..qn, determinar la posición y orientación del extremo efector.
  - Cinemática inversa: dadas la posición y orientación del extremo efector, calcular las posibles combinaciones de ángulos de los ejes que las generan.
    - Dado el carácter no lineal de las ecuaciones, este es, en general, un problema de solución difícil.
  - Dinámica: calcular las fuerzas que hay que aplicar a cada eje para llevar el extremo de una posición a otra, a una cierta velocidad y con una cierta aceleración.
  - Generación de trayectorias: calcular las ecuaciones de movimiento de cada uno de los ejes del robot, para obtener el efecto deseado.
- En todos estos casos, las RN pueden utilizarse para compensar los efectos de desgaste, modelos poco precisos de sensores y manipuladores, y estrategias de control flexibles.

Aplicaciones en Robótica

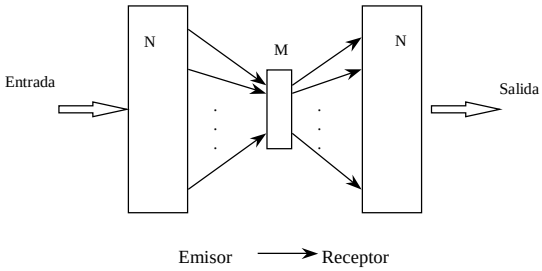
- Partiendo de las imágenes de dos cámaras de TV desplazadas, se emplean las RN para determinar la posición del robot y del objetivo y generar el movimiento hacia el objetivo.
- De forma análoga, alimentando al sistema con la secuencia de posiciones deseadas, se pueden emplear RN para generar las fuerzas sobre los ejes que produzcan el movimiento.
- Robótica móvil
  - Empleo de RN acoplados a dispositivos de visión para "navegación" entre obstáculos , aunque de forma muy rudimentaria, de momento.
  - Soluciones al problema de navegación se puede obtener con otro tipo de sensores de proximidad: ultrasonidos, por ejemplo.

Aplicaciones en visión artificial I

- En visión artificial hay tres problemas básicos:
  - *Reconocimiento*
    - Clasificación de los datos de entrada (pixels) sobre un conjunto de clases posibles.
  - *Información geométrica*
    - Determinación de características y parámetros geométricos .
  - *Compresión* de la imagen
    - Almacenamiento y transmisión, con un mínimo de pérdidas.
- Las RNA se están empleando en los tres campos:
  - Reconocimiento : Congnitrón, neocognitrón, Hopfield, etc
  - Información geométrica: visión estereoscópica.
  - Compresión: MLP, con una capa oculta de menor dimensión que la entrada y la salida.

Aplicaciones en visión artificial II

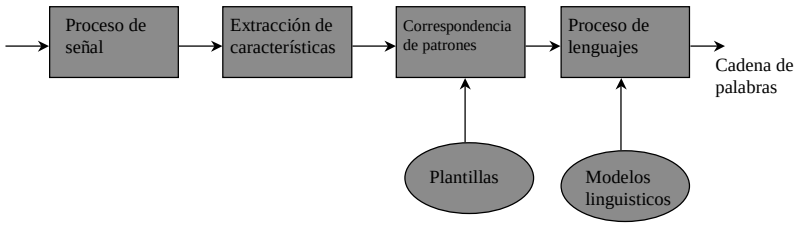
- Compresión de datos



Una vez que la red está entrenada, si el emisor y el receptor disponen de copias de la red, basta enviar las salidas de la capa oculta y el receptor puede reconstruir los datos.

Aplicaciones en Reconocimiento de Habla I

- El RAH puede definirse, en pocas palabras, como la descodificación de la información presente en una señal hablada y su transcripción en un conjunto de caracteres
  - Estos caracteres pueden usarse, posteriormente, en otras tareas (texto escrito, acceso a base de datos, etc)
- Modelo general del problema



RNA en predicción de series temporales I

- La predicción de series temporales es un problema de gran incidencia
  - Carga eléctrica, meteorología, mercados de valores, EEG, ECG, etc.
- En el caso de series estacionarias (cuyas propiedades estadísticas no varían con el tiempo), las redes de retropropagación multicapa (MLP) dan muy buenos resultados
- Para series no estacionarias (tanto aleatorias como caóticas) es preciso emplear algún tipo de RNA con propiedades dinámicas:
  - El MLP -FIR, que sustituye los pesos por filtros FIR, se emplea con éxito en aplicaciones de control adaptable, cancelación de ruidos, identificación de sistemas dinámicos, modelado de series no estacionarias, etc.
  - Las redes recurrentes de tiempo real (RTRN) se emplean en tareas lingüísticas, modulación diferencial adaptable de código de pulsos (ADPCM) en habla, etc

RNA en predicción de series temporales II

- Las RNA se emplean, en general, cuando no se conoce el modelo matemático subyacente (o es demasiado complejo), pero se admite la hipótesis fenomenológica
  - ✓ Existe una relación bien definida entre los valores observables pasados y los futuros.
- En predicción de carga eléctrica, se están empleando tratamientos híbridos, neuronales-expertos-borrosos con resultados prometedores
  - ✓ Las redes recurrentes ofrecen la ventaja de que el volumen de datos para entrenamiento es menor que en los MLP,s.
- Entre las redes recurrentes, la de Elman es de las más empleadas por su sencillez

RNA en predicción de series temporales III

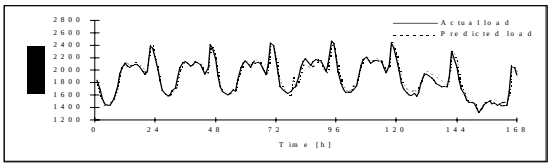


Figura 2: Actual and predicted load. February 15 to 21, 1993.

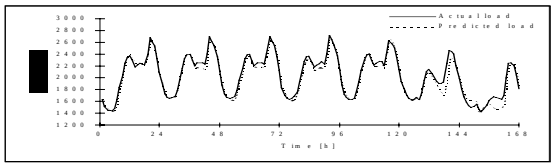


Figura 3: Actual and predicted load. August 23 to 29, 1993.

Resultados de predicción de carga en el sistema de Interconexión de Chile con una red neuronal tipo Elman

RNA en predicción de series temporales IV

- Comparación con otros métodos de predicción

Table 1: Mean Absolute percentage errors.

Week	Elman network	Perceptron network	Expert system	Operators
February 15 to 21	1.89	2.14	1.40	1.31
April 19 to 25	2.90	3.59	2.36	2.22
June 21 to 27	1.98	1.91	2.15	1.86
August 23 to 29	2.42	2.34	2.34	2.87
October 18 to 24	3.98	3.43	2.62	1.91
December 13 to 19	1.67	2.02	2.37	1.38
Average	2.47	2.57	2.21	1.93

Limitaciones prácticas de las RNA

- El entrenamiento puede llevar mucho tiempo.
  - ✓ Las implementaciones paralelas son una solución a este problema, aunque todavía son caras y no muy bien resueltas.
- No existe un método de evaluar las prestaciones de la red.
  - ✓ En cada caso se emplean medidas de funcionamiento diferentes: tasa de acierto, porcentaje de recubrimiento, etc..
- No existe un procedimiento sistemático de visualizar el conocimiento "adquirido" por la RNA.
  - ✓ Existen algunas herramientas gráficas desarrolladas (diagramas de Hinton, bond-graphs, etc) pero su utilidad es muy limitada.

Algunas conclusiones

- Las RNA forman una tecnología emergente que ha crecido muy rápidamente en los últimos 20 años
  - ✓ Sus bases teóricas y técnicas de diseño se han ido elaborando por investigadores procedentes de muy diversos campos
- Se pueden considerar ya como una herramienta más a considerar por los diseñadores de sistemas
- Su capacidad de entrenamiento, supervisado o no, las hace muy adecuadas para problemas de procesamiento de señal o de reconocimiento de patrones considerados difíciles.
- Su mayor ventaja estriba en la capacidad de aprendizaje a partir de ejemplos, formalmente equivalente a una inferencia estadística no paramétrica. De esta forma, las soluciones que proporcionan son buenas, aunque no se pueda garantizar que sean óptimas.
- En muchas ocasiones, funcionan mejor que las alternativas clásicas.