PERCEPTRON MULTICAPA

(Freeman, capt.3)

1. Regla Delta Generalizada (BackPropagation)
    1.1. Características
    1.2. Arquitectura de Pesos
        - Capa de Salida
        - Capas Ocultas
    1.3. Consideraciones Prácticas
        - Datos de Entrada
        - Funciones de Transferencia
        - Dimensionamiento de la estructura
    1.4. Control de Convergencia
        - Mínimos Locales
        - Momento
        - Heurísticas para incrementar la velocidad de aprendizaje

REGLA DELTA GENERALIZADA (BACKPROPAGATION)

- La Regla Delta o LMS: Entrena un PE o una capa de PE.

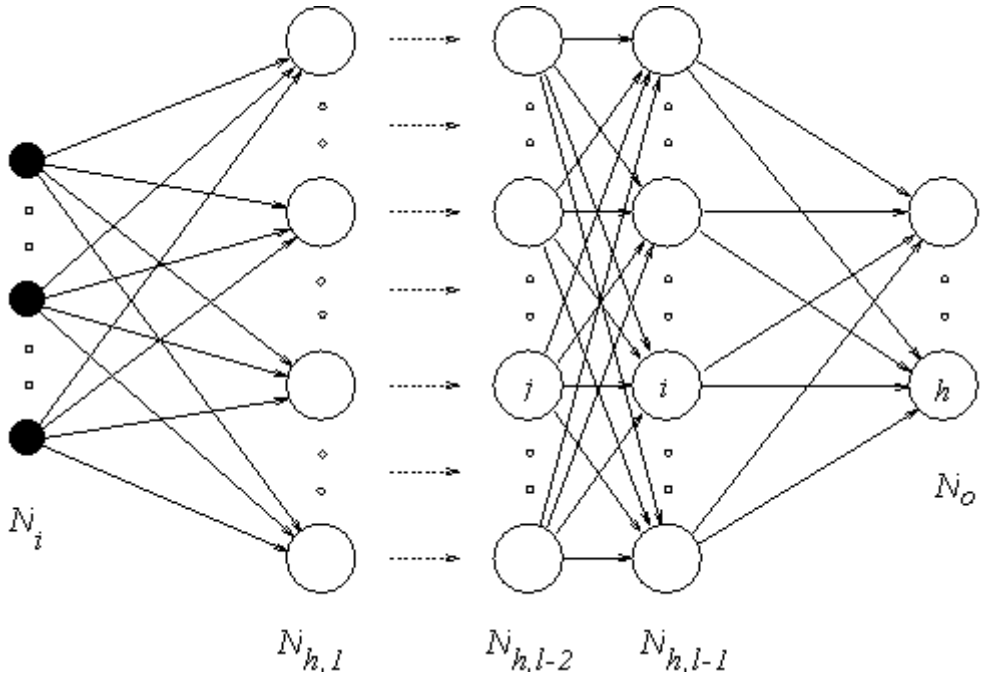
- Existencia de Capas Ocultas: la Regla Delta no serviría para el entrenamiento de la estructura:

    - Conocemos la salida deseada para cada patrón de entrenamiento.
    - No conocemos la salida deseada para los PE de las capas ocultas.

La ***Regla Delta Generalizada o Backpropagation*** fue creada para generalizar la Regla Delta sobre *Redes Neuronales de múltiples capas* y *funciones de transferencia no lineales y diferenciables*.

Características:

- Entrenamiento Supervisado: Corrección de Error.
- Aprendizaje Off Line.
- Capacidad de Generalización.



Notación:

$$neta_{pj}^{h} = \sum_{i=0}^{n} w_{ji}^{h} x_{pi}$$

$$i_{pj}^{h} = f_{j}^{h}(neta_{pj}^{h})$$

$$o_{pj} = f_{j}^{o}(neta_{pj}^{o})$$

n- número de entradas de cada patrón.
o- indica la capa.
p- indica el patrón.
j- PE j.
Wji- conexión PE i (capa h-1) con PE j (capa h)
Opj- salida PE j en la capa de salida.

ARQUITECTURA DE PESOS

Capa de Salida

- Como en la capa de salida puede haber un nº >1 de PE, en este caso no nos basta con un único valor de error:

$$\delta_{pk} = (y_{pk} - o_{pk})$$

- El error que se debe minimizar es la suma de los cuadrados de los errores de todas las unidades de salida.

$$E_p = \frac{1}{2}\sum_{k=1}^{n}\delta_{pk}^2 = \frac{1}{2}\sum_k (y_{pk} - o_{pk})^2$$

$$\nabla E_p = \frac{\delta E_p}{\delta w_{kj}^o} = -(y_{pk} - o_{pk})\frac{\delta f_k^o}{\delta(neta_{pk}^o)}\frac{\delta(neta_{pk}^o)}{\delta w_{kj}^o}$$

$$\frac{\delta(neta_{pk}^o)}{\delta w_{kj}^o} = \frac{\delta}{\delta w_{kj}^o}\left(\sum_j w_{kj}^o i_{pj}^{o-1}\right) = i_{pj}^{o-1}$$

Todo implica que:

$$\nabla E_p = (y_{pk} - o_{pk}) f_k^{o'}(neta_{pk}^o) i_{pk}^{o-1}$$

De tal manera que los pesos en la capa de salida se modificarán de la siguiente manera:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \Delta_p w_{kj}^{o}(t)$$

$$\Delta_p w_{kj}^{o}(t) = \mu(y_{pk} - o_{pk}) f_k^{o'}(neta_{pk}^{o}) i_{pj}^{o-1}$$

Condición NECESARIA: **f debe ser derivable**. Dos casos:

$$f_k^{o}(neta_k^{o}) = neta_k^{o} \Rightarrow f_k^{o'} = 1$$

$$f_k^{o}(neta_k^{o}) = (1+e^{-neta_k^{o}})^{-1} \Rightarrow f_k^{o'} = f_k^{o}(1-f_k^{o}) = o_{pk}(1-o_{pk})$$

$$\delta_{pk}^{o} = (y_{pk} - o_{pk}) f_k^{o'}(neta_{pk}^{o}) = \delta_{pk} f_k^{o'}(neta_{pk}^{o})$$

Sea:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \mu \delta_{pk}^{o} i_{pj}^{o-1}$$

Capas Ocultas

- ¿Como determinamos los valores esperados de los PE en las capas ocultas?
- Ep está relacionado con la salida de los PE en capas ocultas de la siguiente manera:

$$E_p = \frac{1}{2}\sum_k (y_{pk} - o_{pk})^2 = \frac{1}{2}\sum_k (y_{pk} - f_k^o(neta_{pk}^o))^2 = \frac{1}{2}\sum_k (y_{pk} - f_k^o(\sum_j w_{kj}^o i_{pj}^h))^2$$

Por otra parte:

$$i_{pj}^h = f_j^h(neta_{pj}^h) = f_j^h(\sum_i w_{ji}^h i_{pi}^{h-1})$$

De tal manera:

$$\frac{\delta E_p}{\delta w_{ji}^h} = \frac{1}{2}\sum_k \frac{\delta}{\delta w_{ji}^h}(y_{pk} - o_{pk})^2 = -\sum_k (y_{pk} - o_{pk}) \frac{\delta o_{pk}}{\delta(neta_{pk}^o)} \frac{\delta(neta_{pk}^o)}{\delta i_{pj}^h} \frac{\delta i_{pj}^h}{\delta neta_{pj}^h} \frac{\delta neta_{pj}^h}{\delta w_{ji}^h} =$$

$$= -\sum_k (y_{pk} - o_{pk}) f_k^{o'}(neta_{pk}^o) w_{kj}^o f_j^{h'}(neta_{pj}^h) i_{pi}^{h-1}$$

Con lo que:

$$\Delta w_{ji}^h = \mu f_j^{h'}(neta_{pj}^h) i_{pi}^{h-1} \sum_k (y_{pk} - o_{pk}) f_k^{o'}(neta_{pk}^o) w_{kj}^o$$

$$\Delta w_{ji}^h = \mu f_j^{h'}(neta_{pj}^h) i_{pi}^{h-1} \sum_k \delta_{pk}^o w_{kj}^o$$

- Las actualizaciones de los pesos en la capa oculta dependen de todos los términos de errores de la capa de salida. Esto es a lo que se refiere uno con el nombre de propagación hacia atrás.

$$\delta_{pj}^{h} = f_{j}^{h'}(neta_{pj}^{h})\sum_{k}\delta_{pk}^{o} w_{kj}^{o}$$

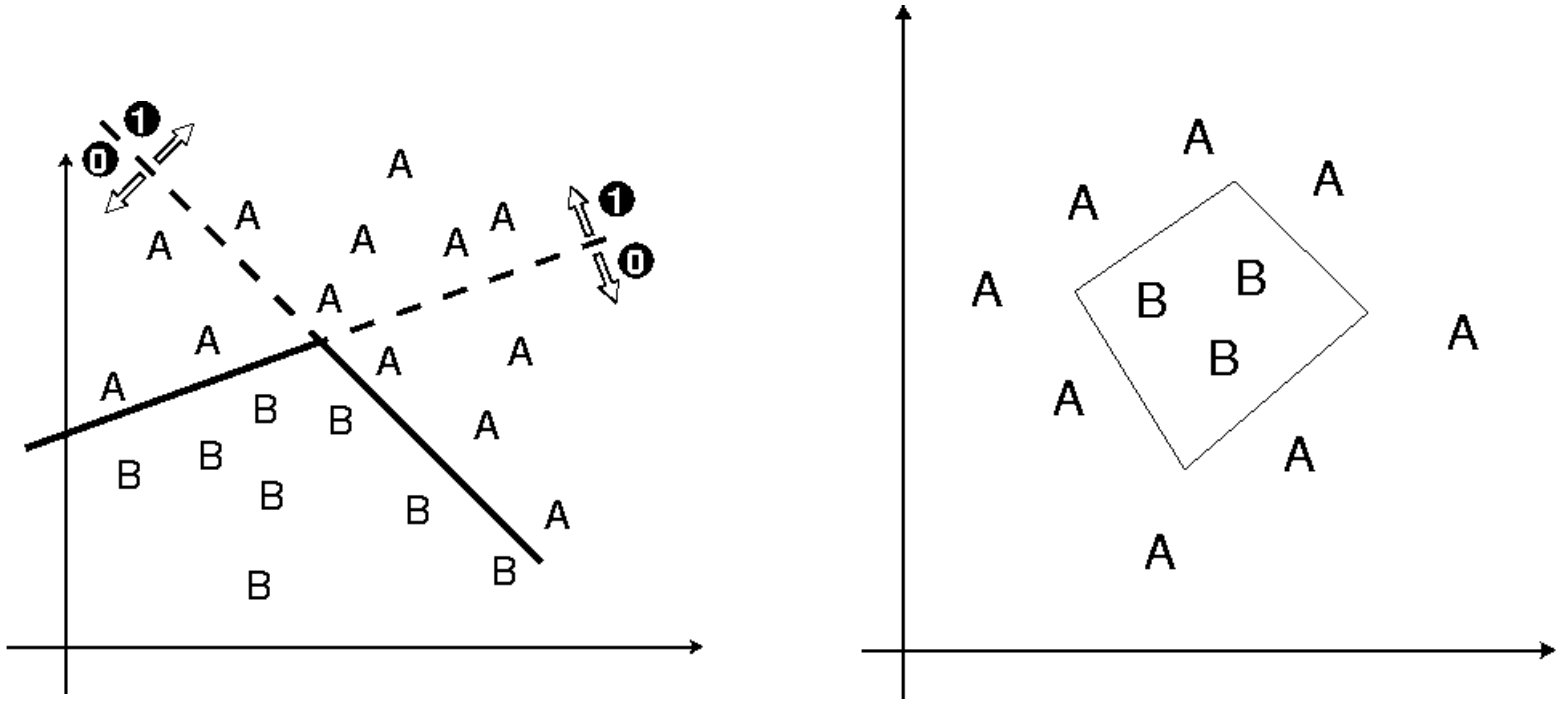
Entonces:

$$w_{ji}^{h}(t+1) = w_{ji}^{h}(t) + \mu\delta_{pj}^{h} i_{i}^{h-1}$$

- Los términos de error de las unidades ocultas, se calculan antes de que hayan sido modificado los pesos de conexiones con las unidades de la capa de salida.

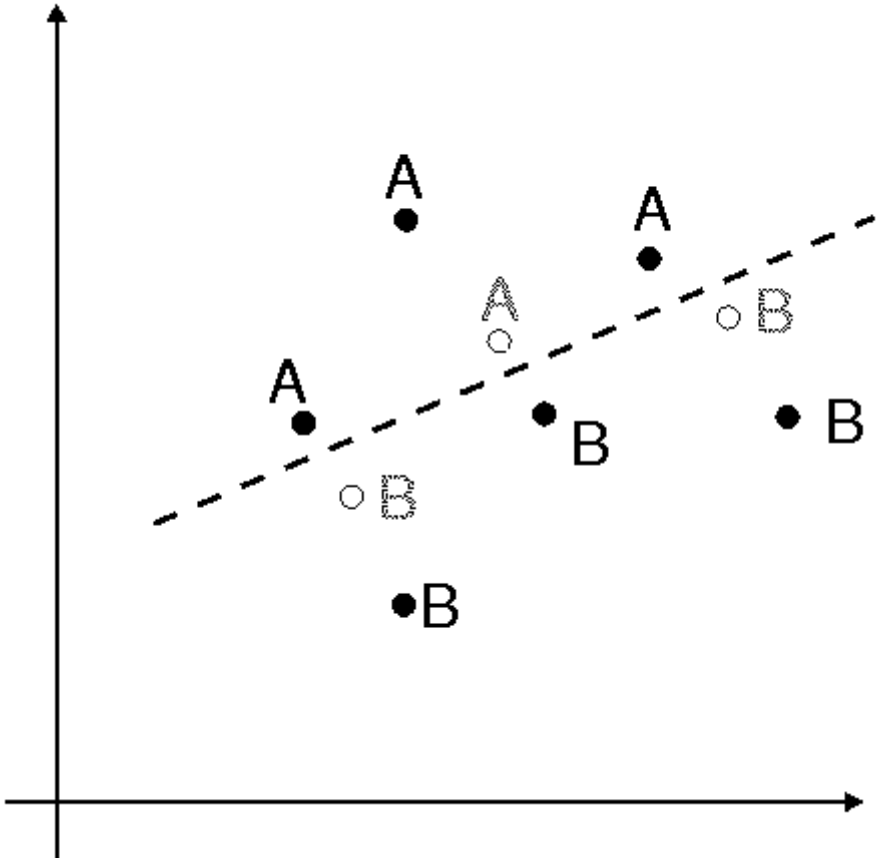
CONSIDERACIONES PRÁCTICAS

- Este tipo de estructuras se introducen para resolver problemas que no son linealmente separables.



Datos de entrada (entrenamiento)

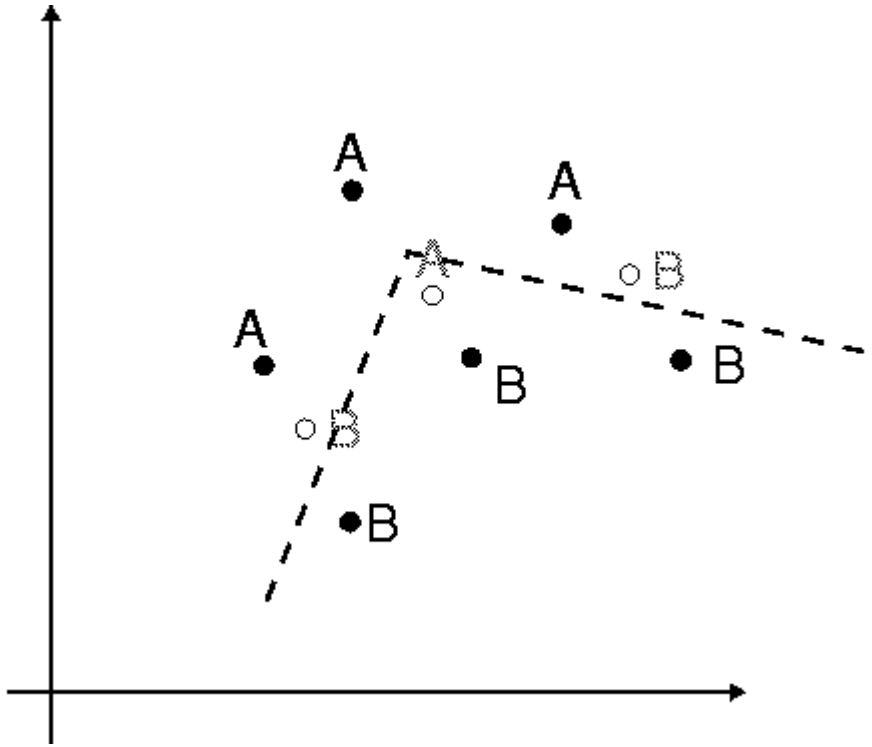
- Se pueden emplear todos los datos disponibles para entrenar la red. Lo que se necesita es: ***Subconjunto de datos que cubran todo el espacio de los mismos***.



La BPN admite la ***Generalización***: Dados varios vectores de entrada (no pertenecientes al conjunto de entrenamiento), similares a patrones existentes en el conjunto de entrenamiento, la red reconocerá las similitudes entre dichos patrones.

- ***La BNP no extrapola bien es decir***: Si la red se entrena mal o insuficientemente, las salidas pueden ser imprecisas.

- ***Región de Incertidumbre***: (vectores de entrenamiento A,B). Red con 2 unidades ocultas (1 capa oculta). Al minimizar el error los planos que se generan se alinean tan cerca de los patrones de entrenamiento como sea posible.



Otras Funciones de Transferencia

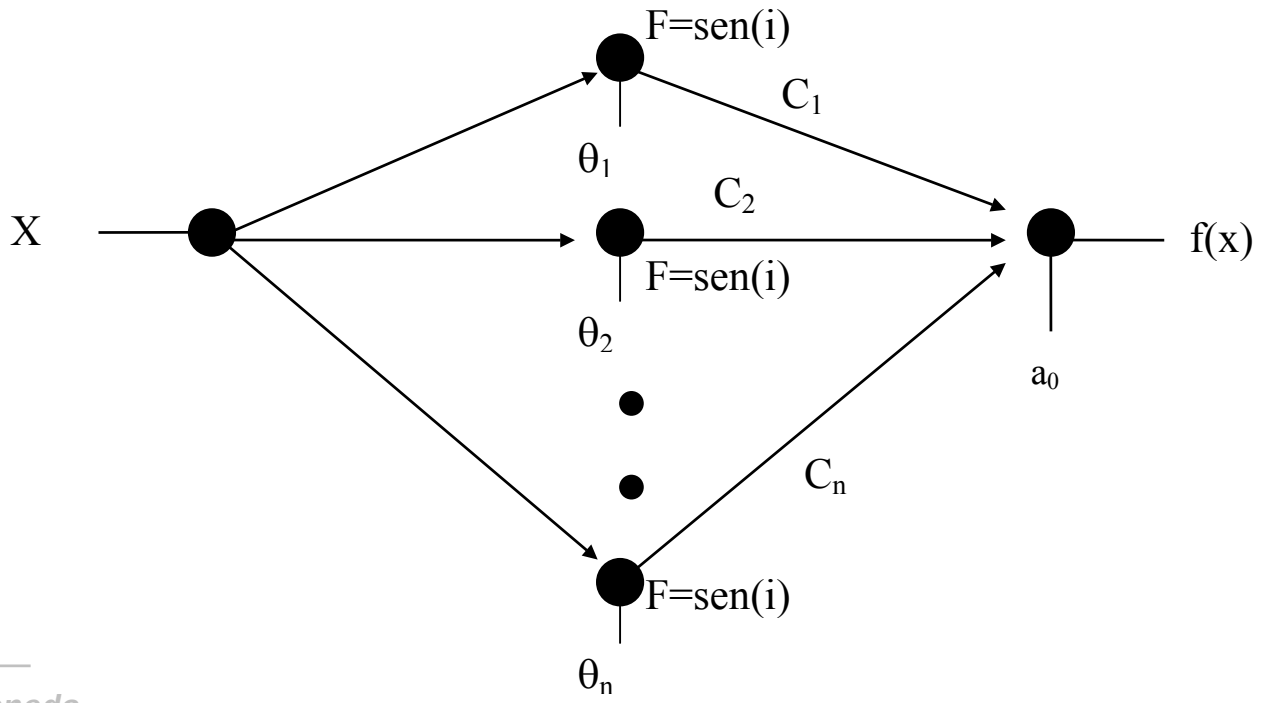
- La elección de la función de transferencia para los diferentes elementos de una red es un aspecto importante a tener en cuenta para el funcionamiento correcto de la red.

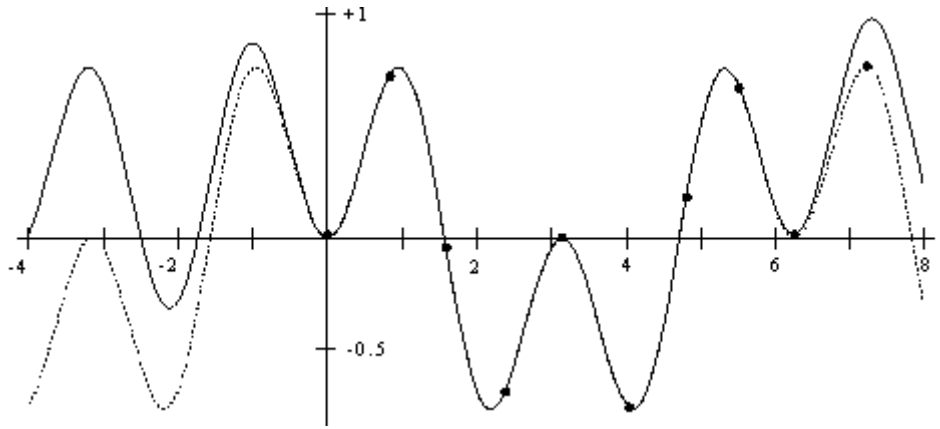
Ejemplo: **Análisis de Fourier.**

$$f(x) = \sum_{n=0}^{\infty} (a_n \bullet \cos nx + b_n \bullet \operatorname{sen} nx) = a_0 + \sum_{n=1}^{\infty} c_n \operatorname{sen}(nx + \theta_n)$$

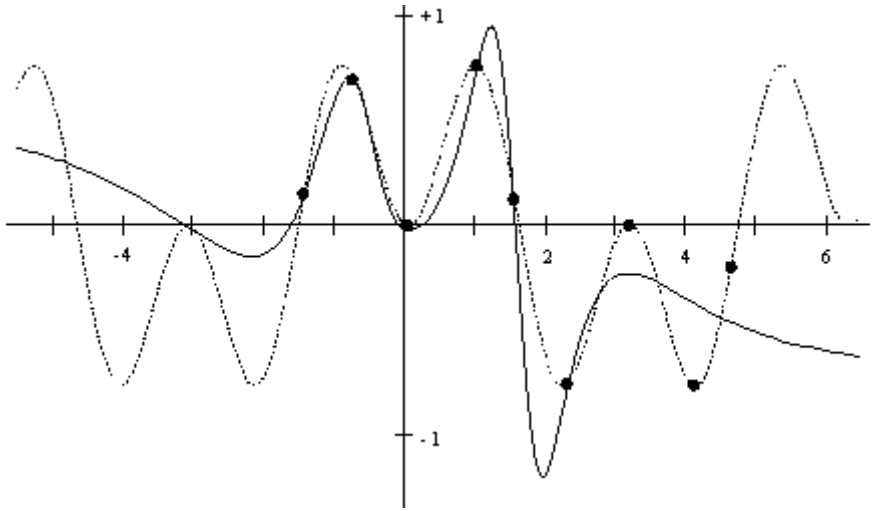
$$c_n = \sqrt{(a_n^2 + b_n^2)}$$

$$\theta_n = arctang(b/a)$$





FunciónTransferencia F=sen(i).



Función Transferencia sigmoidal.

CONTROL DE CONVERGENCIA

- En las técnicas de gradiente descendente es conveniente avanzar por la superficie de error con incrementos pequeños de los pesos.
- Información local de la superficie.
- Incrementos grandes: se corre el riesgo de pasar por encima del punto mínimo sin conseguir estacionarse en él.
- Incrementos pequeños: aunque se tarde más en llegar, se evita que ocurra esto.

- El elegir un incremento adecuado influye en la velocidad con la que converge el algoritmo. Sabemos que este control lo podemos realizar mediante el parámetro denominado ***ganancia***. Normalmente se le asigna un valor pequeño (0,05-0,25) para asegurar que la red llegue a asentarse en una solución.

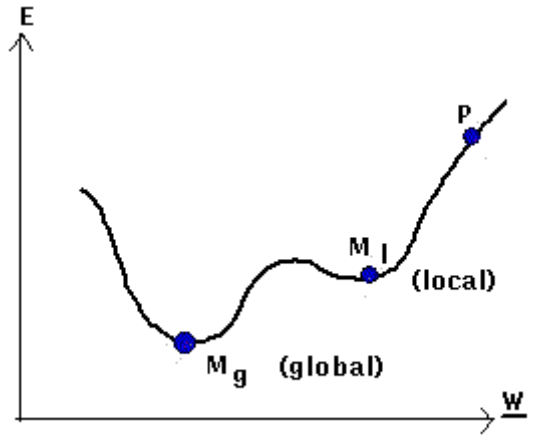
- Otra manera de incrementar la velocidad de aprendizaje, consiste en utilizar otro parámetro llamado ***Momento***:

$$w_{kj}^{o}(t+1) = w_{kj}^{o}(t) + \mu \delta_{pk}^{o} i_{pk}^{o-1} + \alpha \Delta_p w_{kj}^{o}(t-1)$$

- Un último aspecto a tener en cuenta es la posibilidad de convergencia hacia alguno de los ***mínimos locales*****.**

  - No se puede asegurar en ningún momento que el mínimo que se encuentre sea global.

  - Una vez que la red se asienta en un mínimo, sea local o global, cesa el aprendizaje, aunque el error siga siendo demasiado alto, si se ha alcanzado un mínimo local.



- Si se alcanza un mínimo local y el error es satisfactorio, el entrenamiento ha sido un éxito, si no sucede así, puede realizarse varias acciones para solucionar el problema:

  - Cambio de arquitectura (más capas ocultas o más PE)
  - Modificación de parámetros de aprendizaje.
  - Emplear un conjunto de pesos iniciales diferentes.
  - Modificar el conjunto de entrenamiento o presentar los patrones en distinta secuencia.

Procedimientos para incrementar la velocidad de aprendizaje.

- En este apartado describiremos diferentes procedimientos que permitan incrementar la velocidad de aprendizaje manteniendo intacto la propiedad de ***Localidad*** que tienen este tipo de redes, referido a que la computación de un PE está solamente influenciado por aquellos PEs que están físicamente conectados con él.

Heurística 1: Cada parámetro ajustable de la red que determina la función de coste debería tener su propio parámetro de control de velocidad de aprendizaje.

- Esta heurística reconoce el hecho de la posible existencia de diferentes ganancias para cada conexión ajustable de la estructura.

Heurística 2: Cada parámetro de control de velocidad debería variar de un paso a otro.

- Normalmente la superficie de error tiene diferentes formas en un mismo espacio. En función de dichas diferencias, esta heurística establece que así como cambia la forma del error, debería cambiar la velocidad de aprendizaje.

Heurística 3: Cuando la derivada de la función error (coste) con respecto a una conexión determinada, tiene el mismo signo algebraico durante varios pasos del algoritmo, el parámetro ganancia para dicha conexión debería ser incrementado.

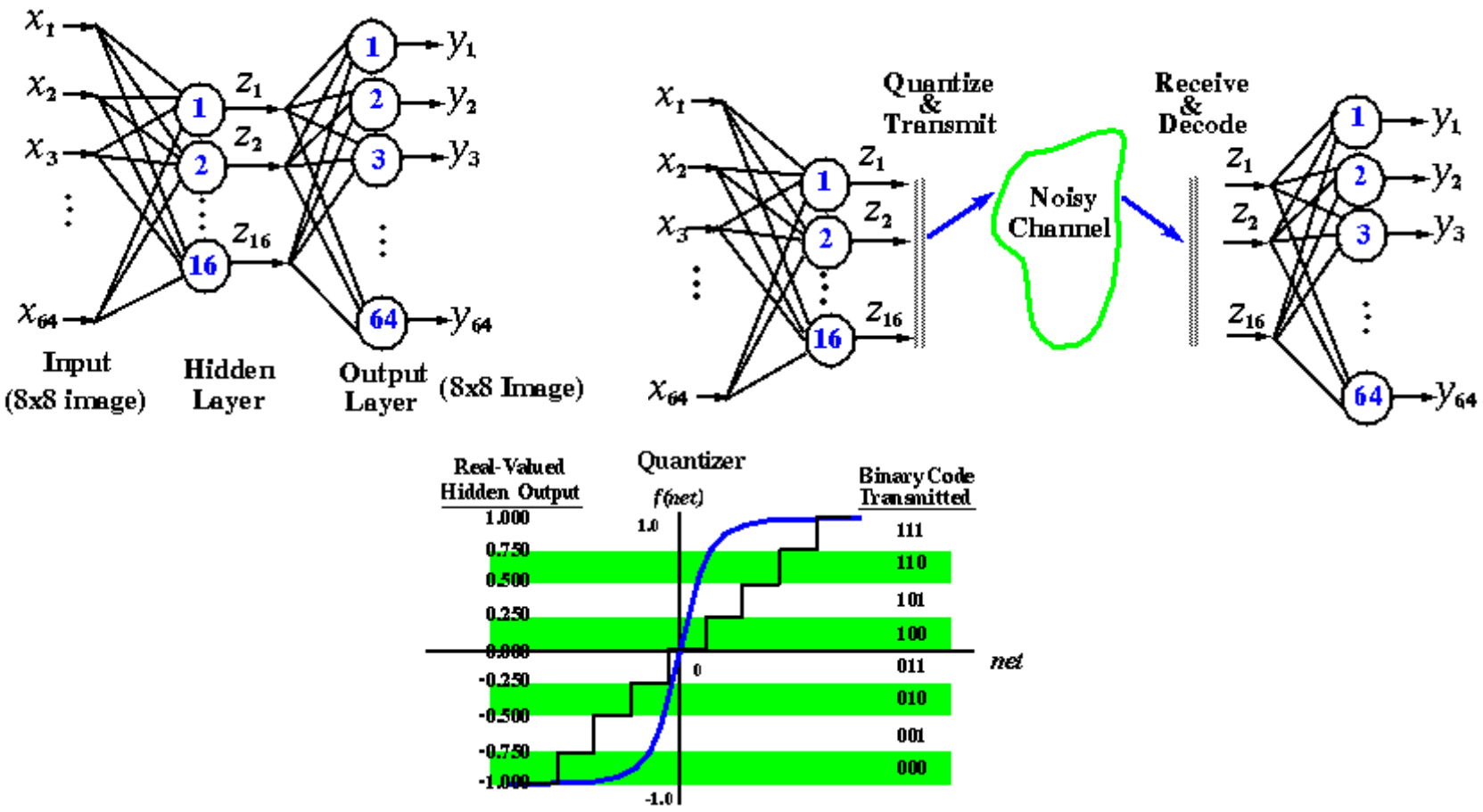
- Esta heurística, establece que en las condiciones mencionadas anteriormente, el número de pasos requeridos para moverse en una cierta parte de la superficie de error, puede ser reducido, incrementando convenientemente la ganancia.

Heurística 4: Cuando el signo algebraico de la derivada de la función error con respecto a una particular conexión cambia durante pasos consecutivos, la ganancia para el peso en cuestión debería decrecer.

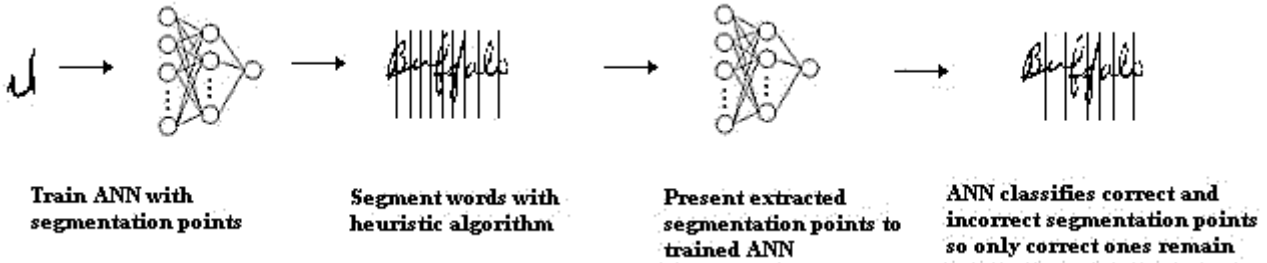
- El cambio de signo en pasos consecutivos viene a significar la existencia de picos y valles. En orden de tratar de localizar el punto mínimo del valle y así evitar esas oscilaciones, la ganancia debería ser reducido (ajuste de pesos más fino).

APLICACIONES

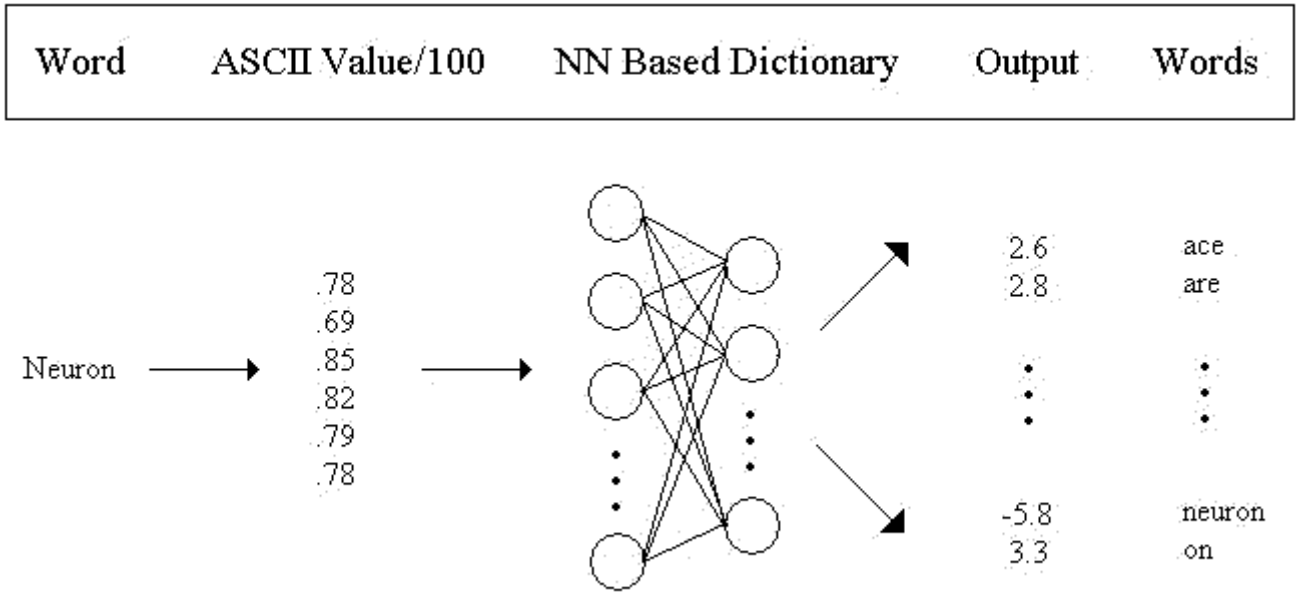
Compresión y Codificación de Información



Clasificación de Caracteres (Segmentación)

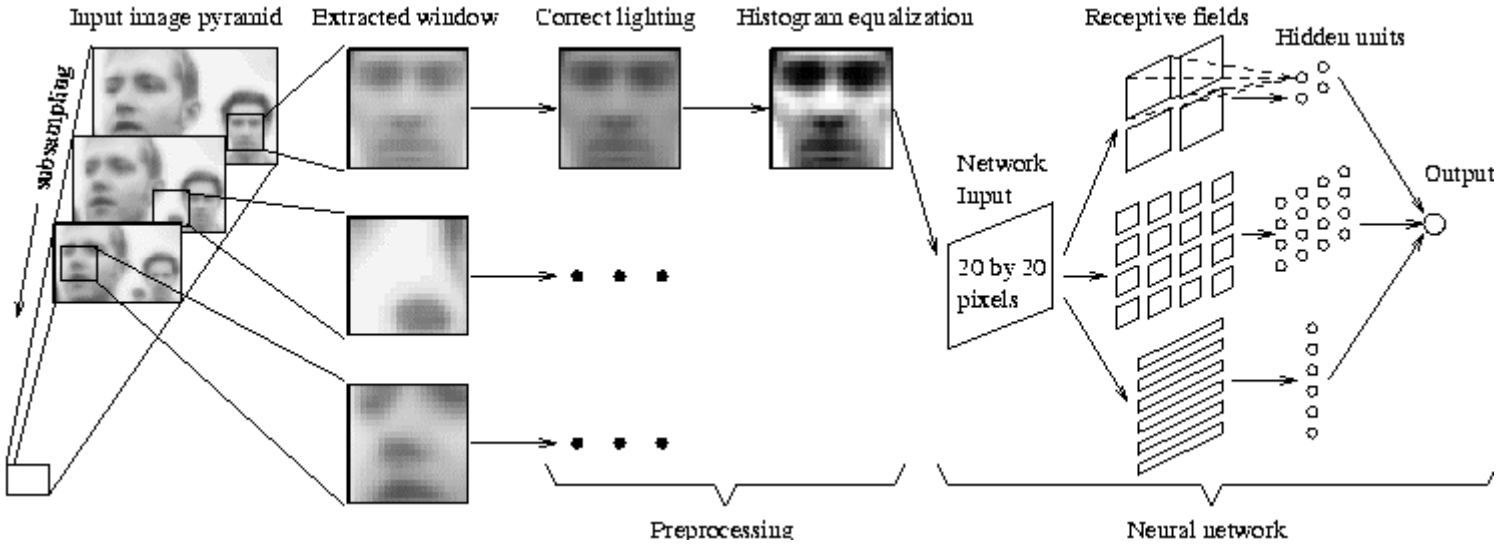


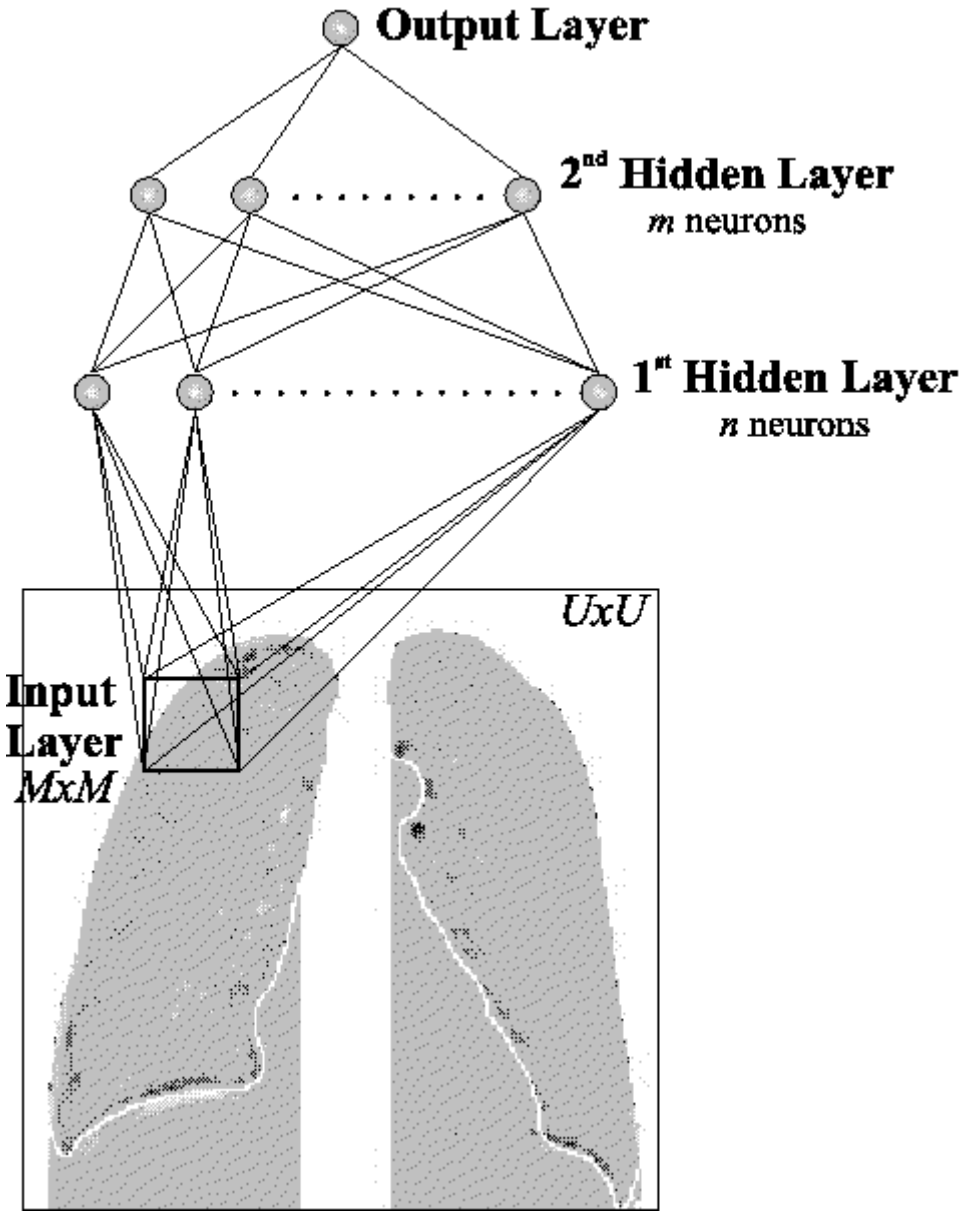
Reconocimiento de Palabras.

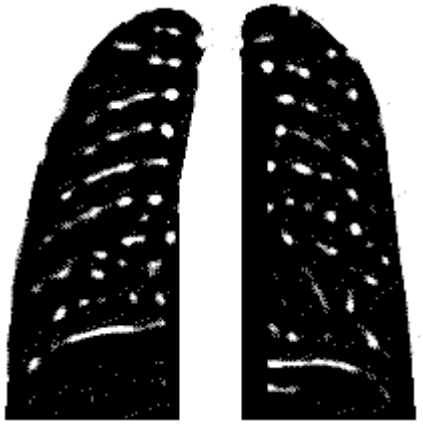


Reconocimiento de Objetos.

Detección de Caras.







Reconocimiento de Patrones

Segmentación de Imágenes