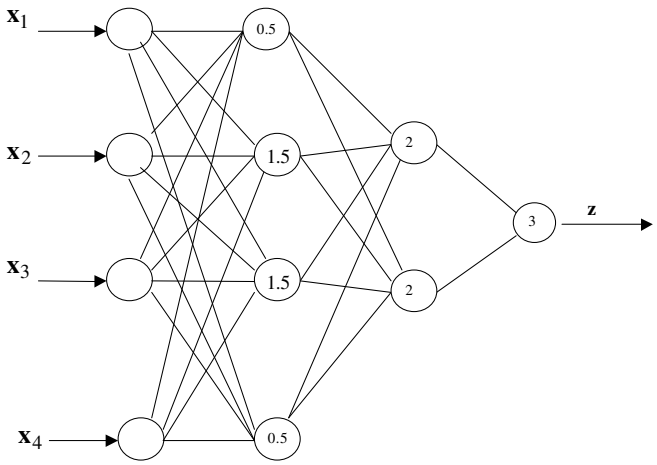
TLU(s) MULTICAPAS

Se pueden implementar funciones en TLU con más de una capa.



No existen, en la actualidad, mecanismos de entrenamiento que permita conocer los pesos sinápticos de las redes TLU multicapas. Algunos mecanismos de aprendizaje, como el Adaline, el Procesador cuadrático, el Madaline, la máquina comité, etc, pretenden resolver la falta de captación no lineal de los algoritmos seminales de entrenamiento. Widrow y Hoff (1960), introducen el algoritmo Least Mean-Squared (LMS) para derivar los adaptadores lineales ADALINE y MADALINE en problemas de clasificación.

Esto por supuesto conduce a pensar:

1. Los TLU resuelven separabilidad en la clasificación de problemas típicamente lineales. Es decir son buenos clasificadores lineales.
2. Los TLU multicapas podrían resolver teóricamente cualquier problema pero no se dispone de procedimientos y mucho menos de algoritmos generales de entrenamiento para estos casos.

Backpropagation (Retropropagación) surge como un método alternativo para resolver la generalidad no alcanzada por los TLU.

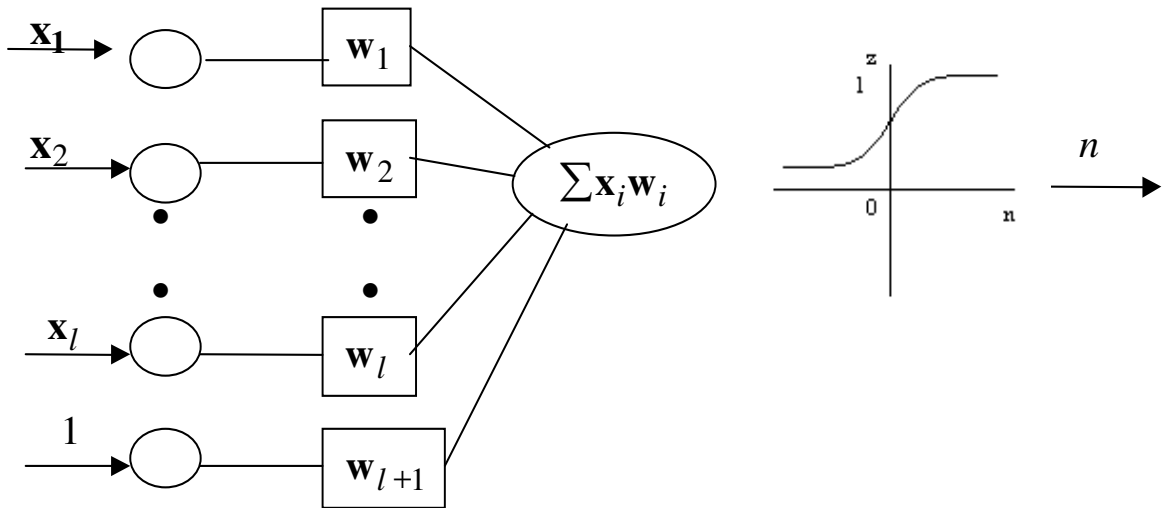
BACKPROPAGATION

Qué es Backpropagation o Retropropagación?

1. También conocido como retropropagación del error.
2. Método para calcular el gradiente del error

Como ya se se mencionó al principio de este texto, en (Werbos, 1974) (Parker, 1985) (Rumelhart, 1986) se comenta que se desarrolla el algoritmo Backpropagation como un mecanismo de aprendizaje para perceptrones multicapas. De alta popularidad para la solución de problemas de clasificación y pronóstico.

Es un método de entrenamiento general para redes multicapas, requiriendo diferenciabilidad en la capa de salida. Es decir la función debe ser diferenciable (sigmoide). En la siguiente figura podemos observar una neurona básica en backpropagation, con sus capas de entrada, oculta y de salida. La de salida activada con una función sigmoidal que describe valores de **z** entre 0 y 1.

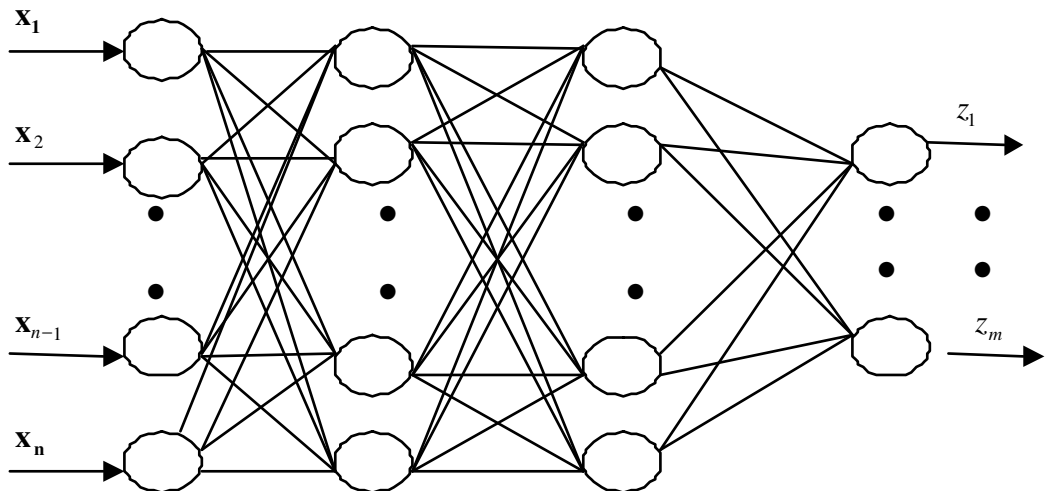


$$\mathbf{z} = \frac{1}{1+e^{-n}}$$

si n >> 0 entonces z → 1
si n << 0 entonces z → 0

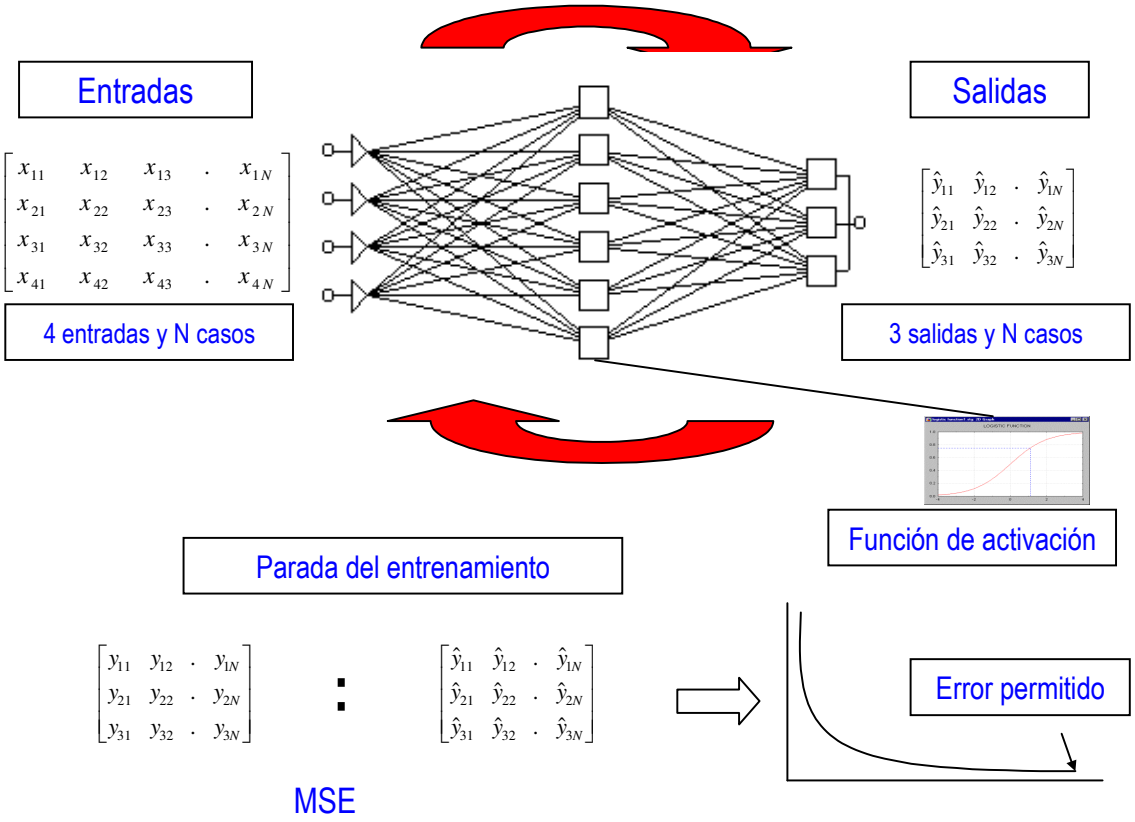
1. Las neuronas transforman una entrada no- restringida en una señal limitada z
2. La función sigmoidal restringe el rango de z entre 0 y 1.
3. Es diferenciable. Hay que tener presente que la no linealidad es una característica resaltante de este algoritmo.
4. Las entradas a la unidad de procesamiento son ejemplos de entrada o salidas de la capa previa

En una red cuya arquitectura incluye capas ocultas con varias neuronas (nodos) ocultas, podemos observar las siguientes características de diseño.



- Está conectada hacia delante.

- Esta red en particular tiene la capa de entrada, dos capas ocultas y la capa de salida.
- La capa de entrada sirve solamente como un punto de distribución.
- Una neurona esta asociada con un conjunto de pesos que conecta a sus entradas. Los pesos en la capa uno terminan en las neuronas de la capa uno.
-



En backpropagation, el método general de entrenamiento se resume a cuatro pasos:

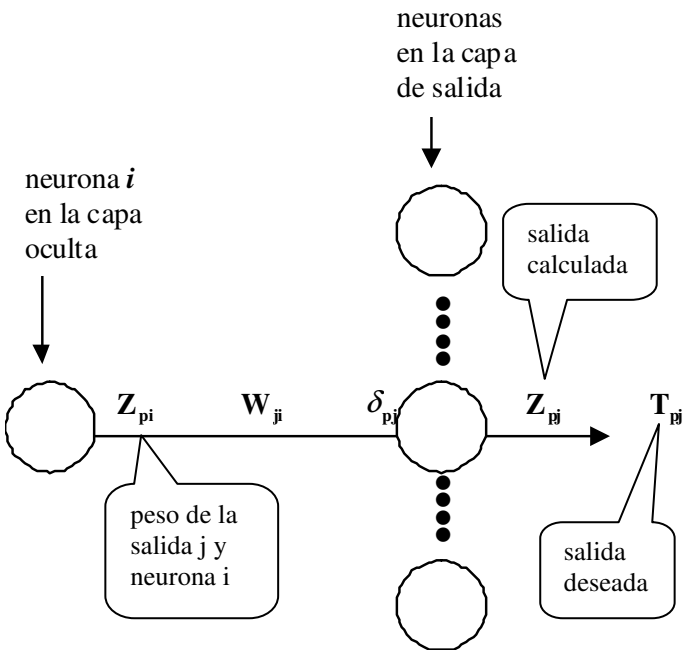
Pasos hacia delante:

1. Selecciona un vector de entrada desde el conjunto de entrenamiento.
2. Aplica esta entrada a la red y calcula la salida.

Pasos hacia atrás:

3. Calcular el error entre la salida calculada y la salida deseada de la entrada usada.
4. Ajustar los pesos para que el error cometido entre la salida calculada y la salida deseada sea disminuido.
5. Repetir los pasos 1 al 5 para todas las entradas del conjunto de entrenamiento, hasta que el error global sea aceptablemente bajo.

COMO AJUSTA LOS PESOS DE LA CAPA DE SALIDA



Como el valor de la salida deseado $\mathbf{T}_{pj}$ está disponible en la capa de salida, entonces los pesos $\mathbf{w}_{ji}$ son ajustados usando algún algoritmo tal como regla Delta y las constantes de aprendizaje.

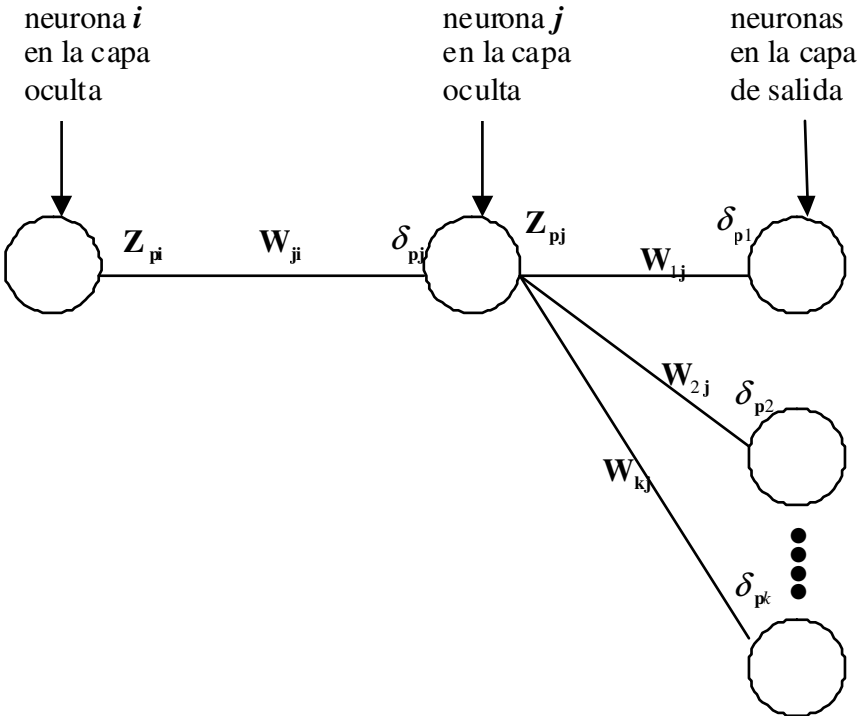
Para un patrón de entrada *p* determinado, si $\mathbf{w}_{ji}$ es el peso que se corresponde con la salida ***j*** y la neurona oculta ***i***, entonces el valor de corrección entre la salida deseada $\mathbf{T}_{pj}$ y la calculada $\mathbf{z}_{pi}$ para ese patrón ***p*** expresada por δ_{pj} y la

salida correspondiente a la neurona ***i*** dada por $\mathbf{z}_{pi}$, podría corregir el valor del peso $\mathbf{w}_{ji}$. Es decir,

$$\Delta_{\mathbf{p}}\mathbf{W}_{\mathbf{ji}} = \eta\delta_{\mathbf{pj}}\mathbf{Z}_{\mathbf{pi}}\text{, donde}$$

$$\delta_{\mathbf{pj}} = \mathbf{Z}_{\mathbf{pj}}(1-\mathbf{Z}_{\mathbf{pj}})(\mathbf{T}_{\mathbf{pj}}-\mathbf{Z}_{\mathbf{pj}})$$

COMO AJUSTA LOS PESOS ENTRE CAPAS OCULTAS



Por otro lado, para la actualización de los pesos entre las capas ocultas se calcula δ propagando hacia atrás desde la capa de salida.

$$\delta_{\mathbf{pj}} = \mathbf{Z}_{\mathbf{pj}}(1-\mathbf{Z}_{\mathbf{pj}})\left(\sum_{\mathbf{k}}\delta_{\mathbf{pk}}\mathbf{w}_{\mathbf{kj}}\right)\text{, y luego } \Delta_{\mathbf{p}}\mathbf{w}_{\mathbf{ji}} = \eta\delta_{\mathbf{pj}}\mathbf{Z}_{\mathbf{pi}}$$

Este cálculo se repite en las capas ocultas de la red.

COMO SE DERIVA LAS FORMULAS

1. Backpropagation es un método gradiente descendente.

2. Minimiza el cuadrado de las diferencias entre las valores calculados y deseados de las salidas sumados sobre todas las unidades de salida y todo los pares, entradas y salidas, del conjunto de entrenamiento

3. Para un conjunto de entrenamiento *p* (entrada y salidas) y *n* unidades de salida,

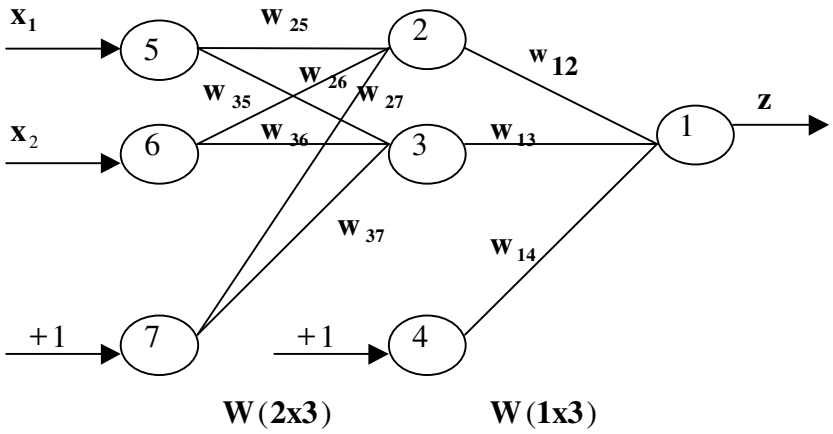
$$\mathbf{E_p} = \frac{1}{2}\sum_{\mathbf{j=1}}^{\mathbf{n}}\left(\mathbf{T_{pj}} - \mathbf{Z_{pj}}\right)^2$$

4. El error total sobre todo el conjunto sería

$$\mathbf{E} = \sum_{\mathbf{p}} \mathbf{E_p}$$

UN EJEMPLO USANDO BACKPROPAGATION

Sea la red



En esta red se muestran dos matrices de pesos $\mathbf{W}_{2x3}$ (3 entradas (incluyendo bias) y 2 salidas en la capa oculta) y otra, $\mathbf{W}_{1x3}$ que incluye 3 entradas en la capa oculta y una salida

Se asume:

Para comodidad del ejercicio se dispuso la siguiente nomenclatura en el índice de las neuronas. En la capa de entrada, las neuronas 5, 6 y 7; en la capa oculta son 2, 3 y 4 y en la de salida es la neurona 1.

- una función sigmoide tal que $f_j'(red) = Z_{pi}(1-Z_{pi})$, donde $red = \sum w_j x_j$
- $\eta = 1$ y
- todos los pesos inicialmente son puestos a 1.

El conjunto de entrenamiento es

x_1	x_2	**z**
0	0	1
0	1	0

Cómo sería un ciclo de aprendizaje?

Para el primer patrón de entrada ampliada [x_1 x_2 bias]= [0 0 1]

a) Pasos hacia delante:

$n_2 = w_{25}\, x_1 + w_{26}\, x_2 + w_{27}\,(+1) = 1*0 + 1*0 + 1*1 = 1$

$n_3 = w_{35}\, x_1 + w_{36}\, x_2 + w_{37}\,(+1) = 1*0 + 1*0 + 1*1 = 1$

$z_2 = 1 / (1 + e^{-n2}) = 1 / (1 + e^{-1}) = 1 / (1 + 0.368) = 0.731$

$z_3 = 0.731$

$n_1 = w_{12}\, z_2 + w_{13}\, z_3 + w_{14}\,(+1) = 1*.731 + 1*.731 + 1*1 = 2.462$

$\mathbf{z_1 = 1 / (1 + e^{-n_1}) = 1 / (1 + e^{-2.462}) = 0.921}$

b) Pasos hacia atrás, para ajustar los pesos.

b.1) Como la neurona 1 es de salida, entonces

$$\delta_1 = \mathbf{Z}_1(1-\mathbf{Z}_1)(\mathbf{T}_1 - \mathbf{Z}_1)$$

$$\delta_1 = 0.921\ (1 - 0.921)\ (1 - 0.921) = 0.00575$$

$$\Delta_{\mathbf{p}}\mathbf{W}_{\mathbf{ji}} = \eta\delta_{\mathbf{pj}}\mathbf{Z}_{\mathbf{pi}}$$

$$\Delta\mathbf{W}_{12} = \eta\delta_1\mathbf{Z}_2 = 1*0.00575*0.731 = 0.0042$$

$$\Delta \mathbf{W}_{13} = \eta \delta_1 \mathbf{Z}_3 = 1*0.0575*0.731 = 0.0042$$

$$\Delta \mathbf{W}_{14} = \eta \delta_1 \mathbf{Z}_4 = 1*0.00575*1 = 0.00575$$

b.2) Las neuronas 2 y 3 son unidades ocultas, entonces

$\delta_{\mathbf{pj}} = \mathbf{Z}_{\mathbf{pj}}(1-\mathbf{Z}_{\mathbf{pj}})\left(\sum_{\mathbf{k}} \delta_{\mathbf{pk}} \mathbf{w}_{\mathbf{kj}}\right)$, donde el rango de *k* es 1 por existir solamente una neurona de salida. Luego,

$$\delta_2 = \mathbf{Z}_2(1-\mathbf{Z}_2)\left(\delta_1 \mathbf{w}_{12}\right) = 0.731*(1-0.731)*0.00575*1 = 0.00113$$

$$\delta_3 = \mathbf{Z}_3(1-\mathbf{Z}_3)\left(\delta_1 \mathbf{w}_{13}\right) = 0.731*(1-0.731)*0.00575*1 = 0.00113$$

$$\Delta_{\mathbf{p}} \mathbf{w}_{\mathbf{ji}} = \eta \delta_{\mathbf{pj}} \mathbf{Z}_{\mathbf{pi}}$$

$$\Delta \mathbf{w}_{25} = \eta \delta_2 \mathbf{Z}_5 = 1*0.00113*0 = 0$$

$$\Delta \mathbf{w}_{35} = \eta \delta_3 \mathbf{Z}_5 = 1*0.00113*0 = 0$$

$$\Delta \mathbf{w}_{26} = \eta \delta_2 \mathbf{Z}_6 = 1*0.00113*0 = 0$$

$$\Delta \mathbf{w}_{36} = \eta \delta_3 \mathbf{Z}_6 = 1*0.00113*0 = 0$$

$$\Delta \mathbf{w}_{27} = \eta \delta_2 \mathbf{Z}_7 = 1*0.00113*1 = 0.00113$$

$$\Delta \mathbf{w}_{37} = \eta \delta_3 \mathbf{Z}_7 = 1*0.00113*1 = 0.00113$$

De aquí los nuevos pesos después del primer conjunto de entrenamiento son:

$W_{12} = 1.0042$, $W_{13} = 1.0042$, $W_{14} = 1.00575$

$W_{25} = 1$, $W_{26} = 1$, $W_{27} = 1.00113$

$W_{35} = 1$, $W_{36} = 1$, $W_{37} = 1.00113$

Cuáles serían para el segundo conjunto de entrenamiento?

Desarróllelo……

LA VARIACION MOMENTO EN BACKPROPAGATION

Después que el error es calculado sobre cada conjunto de entrenamiento (patrón), cada peso es ajustado en proporción al gradiente del error calculado propagado hacia atrás desde la salida hacia la entrada, tal como se vio en el algoritmo y el ejemplo. Los cambios en los pesos reducen el error total de la red neuronal. Para evitar que en la superficie del error los pesos estimen una minimización del error con un óptimo local, se aplica el momento como un mecanismo alternativo para cambiar la dirección de los pesoso y eventualmente logre un mínimo global. En consecencia, mejora los tiempos de entrenamiento al algoritmo backpropagation.

E$n esta caso al aplicar momento, en lugar de calcular los pesos para un conjunto de entrenamiento mediante $\Delta_{\mathbf{p}}\mathbf{w}_{\mathbf{ji}} = \eta\delta_{\mathbf{pj}}\mathbf{Z}_{\mathbf{pi}}$, se afecta esta expresión mediante un término que indica la contribución proporcional (término momento) $\alpha\cdot$ de los pesos previos o anteriores. Es decir,

$\Delta_{\mathbf{p}}\mathbf{w}_{\mathbf{ji}} = \eta\delta_{\mathbf{pj}}\mathbf{Z}_{\mathbf{pi}} + \alpha\cdot\Delta_{\mathbf{p}}\mathbf{w}_{\mathbf{ji}}(anterior)$, el valor de $\alpha\cdot$debe estar entre 0 y 1.

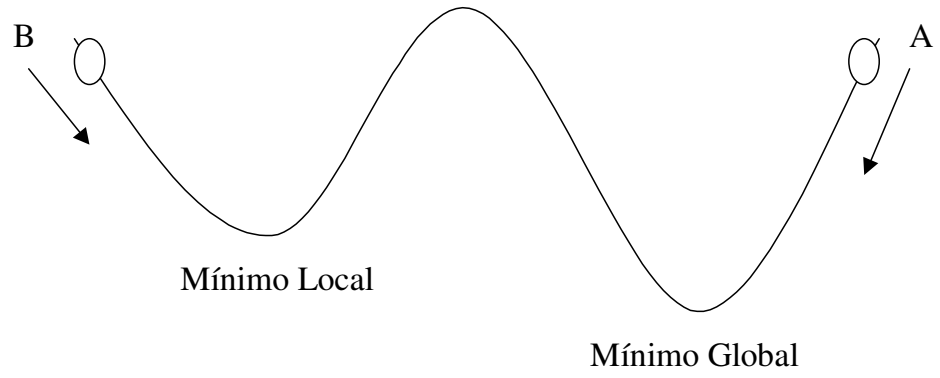
Con respecto a esta variación, se ha encontrado que en algunos casos puede producir buenos resultados cuando en la superficie del error alcanza un área plana o un mínimo local puntual (profundidad estrecha y bastante grande). En estos casos valores cercano a uno (por ej.: 0.9) podrían ayudar a mejorar la solución (permiten obviar este mínimo local). En otros casos, en la superficie del error, por ejemplo áreas ondulantes amplias y pocas profundas, no ayudarían en mayor grado.

De ahí que la solución pueda estar afectada por varios mínimos locales, haciendo inalcanzable un mínimo global.

ALGUNOS COMENTARIOS SOBRE BACKPROPAGATION

a) Acerca del entrenamiento y el error.

La regla Delta (generalizada) empleada por el algoritmo backpropagation puede decrecer el error cuadrático. Sin embargo puede conducir a un mínimo local más que a un mínimo global.



Siempre en una superficie del error cuadrático el mínimo local existe. Sin embargo, no se sabe si ellos están cerca de los mínimos globales.

La superficie del error puede presentar muchas áreas planas donde los cambios de los pesos son mínimos y no ayudan a conseguir los mínimos globales. De ahí, los mecanismos de ayuda al algoritmo original, como, por ejemplo la aplicación del momento, como medida de corrección de la dirección de los pesos.

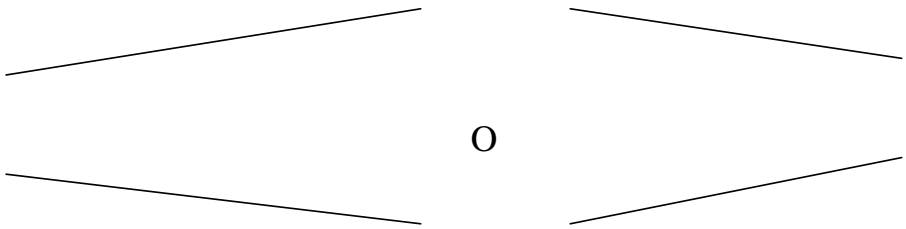
Actualmente se dispone de variantes numéricas que contribuyen a mejorar la eficiencia del algoritmo original. Tal es el caso de los algoritmos del gradiente conjugado, Levenberg-Marquardt y Quasi-Newton, disponibles MATLAB, Statistica Neural Networks, etc.

b) Acerca de las capas y el número de nodos ocultos.

No existe alguna metodología que indique como seleccionar el número de capas, número de nodos (neuronas) por capas en el diseño de una red.

Sin embargo, en la actualidad existen métodos alternativos para mejorar la eficiencia del algoritmo, tales como el uso de los algoritmos genéticos para establecer a priori un diseño de la red, análisis de regresión como recurso de preprocesamiento para conocer los pesos iniciales, transformación de variables, etc.

Eventualmente se ha estimado que el promedio de la suma del número de nodos de entrada y de salida indicaría un valor inicial del número de nodos ocultos. Algo así como un diseño de trompeta en ambos sentidos.



Teóricamente una capa oculta es suficiente para capturar la no linealidad del problema.

Simon Haykin recomienda dos capas: la primera para capturar las características locales y la segunda para las características globales. Una de las principales dificultades en este tipo de redes es el tiempo de entrenamiento: en ocasiones, son demasiados extensos como producto de la complejidad del diseño de la red neuronal.

c) La **selección del valor de la constante de aprendizaje** a veces resulta un arte.

d) La especificidad aplicativa de un modelo de redes neuronales justifica más aún el **cuidado que se debe tener en las fases de preparación y diseño del modelo**. Tal es el caso del pre-procesamiento de los datos y la capacidad de generalización alcanzada por el modelo.