

Perceptrón multicapa

Diego Milone y Leonardo Rufiner
Inteligencia Computacional
Departamento de Informática
FICH-UNL

Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Notas históricas

- 1957 Rosenblatt comienza el desarrollo del Perceptrón (simple).
- 1960 Widrow y Hoff desarrollan el modelo Adaline (ADaptative LINear Elements).
- 1969 Minsky y Papert prueban que el Perceptrón no es capaz de resolver problemas sencillos (XOR).
- 1974 Werbos desarrolla la idea básica del algoritmo de retro-propagación (BP).
- 1986 Rumelhart y Hinton redescubren y mejoran el algoritmo de BP.



Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Perceptrón A: $x_2 = -1 - x_1$

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Perceptrón A: $x_2 = -1 - x_1 = \frac{w_{A0}}{w_{A2}} - \frac{w_{A1}}{w_{A2}}x_1$

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Perceptrón A: $x_2 = -1 - x_1 = \frac{w_{A0}}{w_{A2}} - \frac{w_{A1}}{w_{A2}}x_1$

$$\rightarrow \left\{ \begin{array}{l} w_{A0} = -1 \\ w_{A1} = +1 \\ w_{A2} = +1 \end{array} \right\} \rightarrow y_A = \text{sgn}(x_2 + x_1 + 1)$$

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Perceptrón A: $x_2 = -1 - x_1 = \frac{w_{A0}}{w_{A2}} - \frac{w_{A1}}{w_{A2}}x_1$

$$\rightarrow \left\{ \begin{array}{l} w_{A0} = -1 \\ w_{A1} = +1 \\ w_{A2} = +1 \end{array} \right\} \rightarrow y_A = \text{sgn}(x_2 + x_1 + 1)$$

Perceptrón B: $x_2 = +1 - x_1$

Combinación de perceptrones simples

¿Cómo podemos combinar dos o más PS para resolver el problema XOR?

Perceptrón A: $x_2 = -1 - x_1 = \frac{w_{A0}}{w_{A2}} - \frac{w_{A1}}{w_{A2}}x_1$

$$\rightarrow \left\{ \begin{array}{l} w_{A0} = -1 \\ w_{A1} = +1 \\ w_{A2} = +1 \end{array} \right\} \rightarrow y_A = \text{sgn}(x_2 + x_1 + 1)$$

Perceptrón B: $x_2 = +1 - x_1$

$$\rightarrow \left\{ \begin{array}{l} w_{B0} = +1 \\ w_{B1} = +1 \\ w_{B2} = +1 \end{array} \right\} \rightarrow y_B = \text{sgn}(x_2 + x_1 - 1)$$

Combinación de perceptrones simples

Perceptrón C: $y_A = +1 + y_B$

Combinación de perceptrones simples

Perceptrón C: $y_A = +1 + y_B$

$$\rightarrow \left\{ \begin{array}{l} w_{C0} = +1 \\ w_{C1} = -1 \\ w_{C2} = +1 \end{array} \right\} \rightarrow y_C = \text{sgn}(y_A - y_B - 1)$$

Combinación de perceptrones simples

Perceptrón C: $y_A = +1 + y_B$

$$\rightarrow \left\{ \begin{array}{l} w_{C0} = +1 \\ w_{C1} = -1 \\ w_{C2} = +1 \end{array} \right\} \rightarrow y_C = \text{sgn}(y_A - y_B - 1)$$

¿Cómo es la arquitectura de esta red neuronal?

$$\left. \begin{array}{l} y_A = \text{sgn}(x_2 + x_1 + 1) \\ y_B = \text{sgn}(x_2 + x_1 - 1) \end{array} \right\} \rightarrow y_C = \text{sgn}(y_A - y_B - 1)$$

Combinación de perceptrones simples

Perceptrón C: $y_A = +1 + y_B$

$$\rightarrow \left\{ \begin{array}{l} w_{C0} = +1 \\ w_{C1} = -1 \\ w_{C2} = +1 \end{array} \right\} \rightarrow y_C = \text{sgn}(y_A - y_B - 1)$$

¿Cómo es la arquitectura de esta red neuronal?

$$\left. \begin{array}{l} y_A = \text{sgn}(x_2 + x_1 + 1) \\ y_B = \text{sgn}(x_2 + x_1 - 1) \end{array} \right\} \rightarrow y_C = \text{sgn}(y_A - y_B - 1)$$

¿Resuelve el problema XOR?

Combinación de perceptrones simples

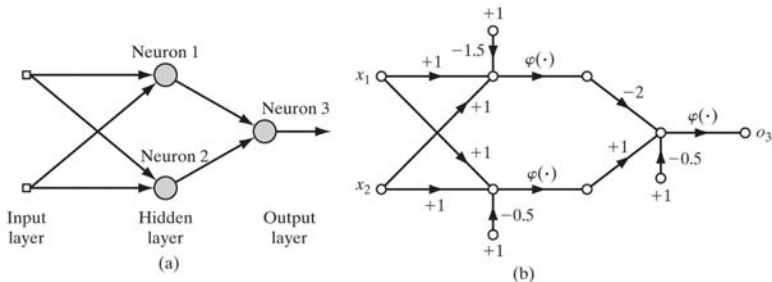


Figura: (a) Arquitectura de una red para resolver el problema del XOR. (b) Gráfico de flujo de señal de la red.

Combinación de perceptrones simples

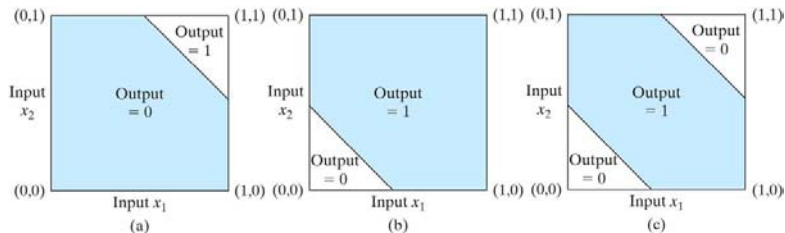


Figura: (a) Límite de decisión construido por la neurona oculta 1 de la red en la fig. anterior. (b) Límite de decisión construido por la neurona oculta 2 de la red. (c) Límite de decisión construido por la red completa.

Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Entrenamiento por el método de gradiente

- Concepto:

Mover los pesos en la dirección en que se reduce el error, dirección que es opuesta a su gradiente con respecto a los pesos

Entrenamiento por el método de gradiente

- Concepto:
Mover los pesos en la dirección en que se reduce el error, dirección que es opuesta a su gradiente con respecto a los pesos
- Interpretación gráfica

Entrenamiento por el método de gradiente

- Concepto:
Mover los pesos en la dirección en que se reduce el error, dirección que es opuesta a su gradiente con respecto a los pesos
- Interpretación gráfica
- Ecuación básica:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu \nabla_{\mathbf{w}} \xi(\mathbf{w}(n))$$

Entrenamiento por el método de gradiente

- Concepto:
Mover los pesos en la dirección en que se reduce el error, dirección que es opuesta a su gradiente con respecto a los pesos
- Interpretación gráfica
- Ecuación básica:

$$\mathbf{w}(n+1) = \mathbf{w}(n) - \mu \nabla_{\mathbf{w}} \xi(\mathbf{w}(n))$$

- Aplicación:
 - Caso sencillo: perceptrón simple (least mean squares)
 - Caso más general: perceptrón multicapa (back-propagation)

Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Extensión del algoritmo a múltiples capas

- Entrenamiento por gradiente en el ADALINE
- Entrenamiento por gradiente en el MADALINE
- Entrenamiento por gradiente en el caso general
- Regiones de decisión

Regiones para varias capas


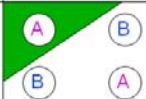
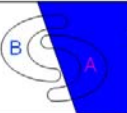


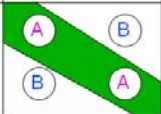
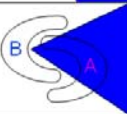


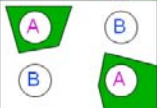
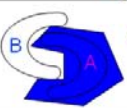
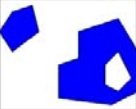
<i>Estructura</i>	<i>Tipos de regiones de decisión</i>	<i>Problema XOR</i>	<i>Separación en clases</i>	<i>Formas regiones más generales</i>
Una capa 	<i>hemiplano limitado por hiperplano</i>			
Dos capas 	<i>Regiones convexas abiertas o cerradas</i>			
Tres capas 	<i>Arbitrarias (Complejidad limitada por N°. de Nodos)</i>			

Figura: Diferentes problemas no-linealmente separables (Lippmann, 1987).

Arquitectura del perceptrón multicapa

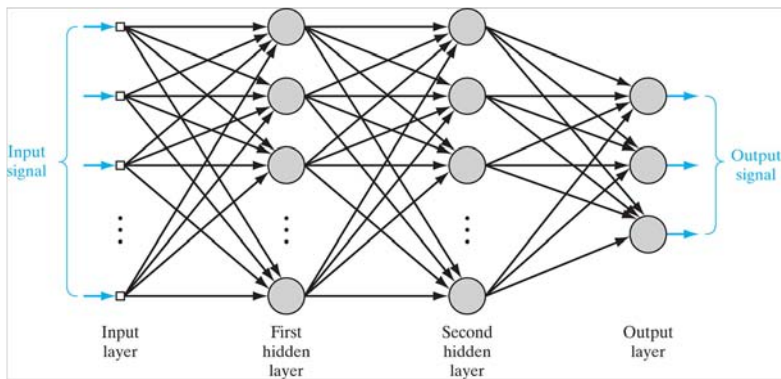


Figura: Arquitectura de un perceptrón multicapa (PMC) con dos capas ocultas.

Arquitectura del perceptrón multicapa

- Representación gráfica de 3 capas
- Cálculo de las salidas en cada capa
- Criterio: suma del error cuadrático instantáneo

Cálculo de las salidas en cada capa

- Capa I:

$$v_j^I = \langle \mathbf{w}^I, \mathbf{x} \rangle = \sum_{i=0}^N w_{ji}^I x_i \quad (\text{completo } \mathbf{v}^I = \mathbf{W}\mathbf{x})$$

Cálculo de las salidas en cada capa

- Capa I:

$$v_j^I = \langle \mathbf{w}^I, \mathbf{x} \rangle = \sum_{i=0}^N w_{ji}^I x_i \quad (\text{completo } \mathbf{v}^I = \mathbf{W}\mathbf{x})$$

$$y_j^I = \phi(v_j^I) = \frac{2}{1 + e^{-bv_j^I}} - 1 \quad (\text{simétrica } \pm 1)$$

Cálculo de las salidas en cada capa

- Capa I:

$$v_j^I = \langle \mathbf{w}^I, \mathbf{x} \rangle = \sum_{i=0}^N w_{ji}^I x_i \quad (\text{completo } \mathbf{v}^I = \mathbf{W}\mathbf{x})$$

$$y_j^I = \phi(v_j^I) = \frac{2}{1 + e^{-bv_j^I}} - 1 \quad (\text{simétrica } \pm 1)$$

- Capa II:

$$v_j^{II} = \langle \mathbf{w}^{II}, \mathbf{y}^I \rangle \rightarrow y_j^{II} = \phi(v_j^{II})$$

- Capa III:

$$v_j^{III} = \langle \mathbf{w}^{III}, \mathbf{y}^{II} \rangle \rightarrow y_j^{III} = \phi(v_j^{III}) = y_j$$

Criterio de error

Suma del error cuadrático instantáneo

$$\xi(n) = \frac{1}{2} \sum_{j=1}^M e_j^2(n)$$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = -\mu \frac{\partial \xi(n)}{\partial w_{ji}(n)}$$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = -\mu \frac{\partial \xi(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \frac{\partial v_j(n)}{\partial w_{ji}(n)}$$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = -\mu \frac{\partial \xi(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \left[\frac{\partial v_j(n)}{\partial w_{ji}(n)} \right]$$

$$\frac{\partial v_j(n)}{\partial w_{ji}(n)} = \frac{\partial \sum_{i=0}^N w_{ji}(n) y_i(n)}{\partial w_{ji}(n)} = y_i(n)$$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = -\mu \frac{\partial \xi(n)}{\partial w_{ji}(n)}$$

$$\frac{\partial \xi(n)}{\partial w_{ji}(n)} = \left[\frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)} \right] y_i(n)$$

Gradiente de error local instantáneo: $\delta_j = \frac{\partial \xi(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)}$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = \mu \delta_j(n) y_i(n)$$

Gradiente de error local instantáneo: $\delta_j = \frac{\partial \xi(n)}{\partial y_j(n)} \boxed{\frac{\partial y_j(n)}{\partial v_j(n)}}$

Derivada de la función de activación simétrica (1/2)

$$\begin{aligned}
 \frac{\partial y_j(n)}{\partial v_j(n)} &= \frac{\partial \left\{ \frac{2}{1+e^{-v_j(n)}} - 1 \right\}}{\partial v_j(n)} \\
 &= 2 \frac{e^{-v_j(n)}}{(1+e^{-v_j(n)})^2} \\
 &= 2 \frac{1}{1+e^{-v_j(n)}} \frac{e^{-v_j(n)}}{1+e^{-v_j(n)}} \\
 &= 2 \frac{1}{1+e^{-v_j(n)}} \frac{\overbrace{-1+1}^0 + e^{-v_j(n)}}{1+e^{-v_j(n)}} \\
 &= 2 \frac{1}{1+e^{-v_j(n)}} \left(\frac{-1}{1+e^{-v_j(n)}} + \frac{1+e^{-v_j(n)}}{1+e^{-v_j(n)}} \right)
 \end{aligned}$$

Derivada de la función de activación simétrica (2/2)

$$\begin{aligned}
 \frac{\partial y_j(n)}{\partial v_j(n)} &= 2 \frac{1}{1 + e^{-v_j(n)}} \left(1 - \frac{1}{1 + e^{-v_j(n)}} \right) \\
 &= 2 \frac{y_j(n) + 1}{2} \left(1 - \frac{y_j(n) + 1}{2} \right) \\
 &= (y_j(n) + 1) \left(1 - \frac{y_j(n) + 1}{2} \right) \\
 &= (y_j(n) + 1) \left(\frac{2 - y_j(n) - 1}{2} \right) \\
 &= \frac{1}{2} (y_j(n) + 1) (y_j(n) - 1)
 \end{aligned}$$

Aplicación del gradiente (caso general)

$$\Delta w_{ji}(n) = \mu \delta_j(n) y_i(n)$$

Gradiente de error local instantáneo: $\delta_j = -\frac{\partial \xi(n)}{\partial y_j(n)} \frac{\partial y_j(n)}{\partial v_j(n)}$

$$\delta_j = \frac{\partial \xi(n)}{\partial y_j(n)} \frac{1}{2} (1 + y_j(n))(1 - y_j(n))$$

Organización

Un poco de historia...

¿Cómo resolver el problema XOR?

Métodos de gradiente para el entrenamiento

Perceptrón multicapa

Retropropagación en el perceptrón multicapa

Retropropagación en la capa III (salida)

$$\Delta w_{ji}^{III}(n) = \mu \delta_j^{III}(n) y_i^{II}(n)$$

Retropropagación en la capa III (salida)

$$\Delta w_{ji}^{III}(n) = \mu \delta_j^{III}(n) y_i^{II}(n)$$

$$\delta_j^{III}(n) = -\frac{\partial \xi(n)}{\partial y_j^{III}(n)} \frac{1}{2} (1 + y_j^{III}(n)) (1 - y_j^{III}(n))$$

Retropropagación en la capa III (salida)

$$\Delta w_{ji}^{III}(n) = \mu \delta_j^{III}(n) y_i^{II}(n)$$

$$\delta_j^{III}(n) = -\frac{\partial \xi(n)}{\partial y_j^{III}(n)} \frac{1}{2} (1 + y_j^{III}(n)) (1 - y_j^{III}(n))$$

$$\delta_j^{III}(n) = -\frac{\partial \xi(n)}{\partial e_j(n)} \frac{\partial e_j(n)}{\partial y_j^{III}(n)} \frac{1}{2} (1 + y_j^{III}(n)) (1 - y_j^{III}(n))$$

Retropropagación en la capa III (salida)

$$\delta_j^{III}(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_j e_j^2(n) \right\}}{\partial e_j(n)} \cdot \frac{\partial \left\{ d_j^{III}(n) - y_j^{III}(n) \right\}}{\partial y_j^{III}(n)} \cdot \frac{1}{2} (1 + y_j^{III}(n))(1 - y_j^{III}(n))$$

Retropropagación en la capa III (salida)

$$\delta_j^{III}(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_j e_j^2(n) \right\}}{\partial e_j(n)} \cdot \frac{\partial \left\{ d_j^{III}(n) - y_j^{III}(n) \right\}}{\partial y_j^{III}(n)} \cdot \frac{1}{2}(1 + y_j^{III}(n))(1 - y_j^{III}(n))$$

$$\delta_j^{III}(n) = \frac{1}{2}e_j(n)(1 + y_j^{III}(n))(1 - y_j^{III}(n)) \star$$

Retropropagación en la capa III (salida)

$$\delta_j^{III}(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_j e_j^2(n) \right\}}{\partial e_j(n)} \cdot \frac{\partial \left\{ d_j^{III}(n) - y_j^{III}(n) \right\}}{\partial y_j^{III}(n)} \cdot \frac{1}{2}(1 + y_j^{III}(n))(1 - y_j^{III}(n))$$

$$\delta_j^{III}(n) = \frac{1}{2}e_j(n)(1 + y_j^{III}(n))(1 - y_j^{III}(n)) \star$$

$$\Delta w_{ji}^{III}(n) = \eta e_j(n)(1 + y_j^{III}(n))(1 - y_j^{III}(n))y_i^{II}(n)$$

Retropropagación en la capa II (oculta)

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

Retropropagación en la capa II (oculta)

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

$$\delta_j^H(n) = -\frac{\partial \xi(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

$$\delta_j^H(n) = -\frac{\partial \xi(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_k e_k^2(n) \right\}}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

$$\delta_j^H(n) = -\frac{\partial \xi(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_k e_k^2(n) \right\}}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\frac{1}{2} \sum_k \frac{\partial e_k^2(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

$$\delta_j^H(n) = -\frac{\partial \xi(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\frac{\partial \left\{ \frac{1}{2} \sum_k e_k^2(n) \right\}}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\frac{1}{2} \sum_k \frac{\partial e_k^2(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

$$\delta_j^H(n) = -\sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n))(1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^{\text{II}}(n) = - \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_k^{\text{III}}(n)} \frac{\partial y_k^{\text{III}}(n)}{\partial v_k^{\text{III}}(n)} \frac{\partial v_k^{\text{III}}(n)}{\partial y_j^{\text{II}}(n)} \frac{1}{2} (1 + y_j^{\text{II}}(n)) (1 - y_j^{\text{II}}(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^H(n) = - \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_k^H(n)} \frac{\partial y_k^H(n)}{\partial v_k^H(n)} \frac{\partial v_k^H(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

$$\delta_j^H(n) = - \sum_k e_k(n) \cdot \frac{\partial \{d_k^H(n) - y_k^H(n)\}}{\partial y_k^H(n)} \cdot \frac{1}{2} (1 + y_k^H(n)) (1 - y_k^H(n)) \cdot$$

$$\cdot \frac{\partial \left\{ \sum_j w_{kj}^H y_j^H(n) \right\}}{\partial y_j^H(n)} \cdot \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^H(n) = - \sum_k e_k(n) \frac{\partial e_k(n)}{\partial y_k^H(n)} \frac{\partial y_k^H(n)}{\partial v_k^H(n)} \frac{\partial v_k^H(n)}{\partial y_j^H(n)} \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

$$\delta_j^H(n) = - \sum_k e_k(n) \cdot \frac{\partial \{d_k^H(n) - y_k^H(n)\}}{\partial y_k^H(n)} \cdot \frac{1}{2} (1 + y_k^H(n)) (1 - y_k^H(n)) \cdot$$

$$\cdot \frac{\partial \{ \sum_j w_{kj}^H y_j^H(n) \}}{\partial y_j^H(n)} \cdot \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

$$\delta_j^H(n) = - \sum_k e_k(n) \cdot (-1) \cdot \frac{1}{2} (1 + y_k^H(n)) (1 - y_k^H(n)) \cdot$$

$$\cdot w_{kj}^H \cdot \frac{1}{2} (1 + y_j^H(n)) (1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^H(n) = \sum_k e_k(n) \cdot \frac{1}{2}(1 + y_k^{III}(n))(1 - y_k^{III}(n)) \cdot w_{kj}^{III} \cdot \frac{1}{2}(1 + y_j^H(n))(1 - y_j^H(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^H(n) = \sum_k e_k(n) \cdot \frac{1}{2}(1 + y_k^{III}(n))(1 - y_k^{III}(n)) \cdot w_{kj}^{III} \cdot \frac{1}{2}(1 + y_j^H(n))(1 - y_j^H(n))$$

Pero de la capa III★ sabemos que:

$$\delta_k^{III}(n) = \frac{1}{2}e_k(n)(1 + y_k^{III}(n))(1 - y_k^{III}(n))$$

Retropropagación en la capa II (oculta)

$$\delta_j^{\text{II}}(n) = \sum_k e_k(n) \cdot \frac{1}{2}(1 + y_k^{\text{III}}(n))(1 - y_k^{\text{III}}(n)) \cdot w_{kj}^{\text{III}} \cdot \frac{1}{2}(1 + y_j^{\text{II}}(n))(1 - y_j^{\text{II}}(n))$$

Pero de la capa III★ sabemos que:

$$\delta_k^{\text{III}}(n) = \frac{1}{2}e_k(n)(1 + y_k^{\text{III}}(n))(1 - y_k^{\text{III}}(n))$$

Reemplazando:

$$\delta_j^{\text{II}}(n) = \sum_k \delta_k^{\text{III}}(n)w_{kj}^{\text{III}} \cdot \frac{1}{2}(1 + y_j^{\text{II}}(n))(1 - y_j^{\text{II}}(n))$$

Retropropagación en la capa II (oculta)

Volviendo a:

$$\Delta w_{ji}^H(n) = \mu \delta_j^H(n) y_i^I(n)$$

Retropropagación en la capa II (oculta)

Volviendo a:

$$\Delta w_{ji}^{II}(n) = \mu \delta_j^{II}(n) y_i^I(n)$$

Por lo tanto:

$$\Delta w_{ji}^{II}(n) = \eta \left[\sum_k \delta_k^{III} w_{kj}^{III}(n) \right] (1 + y_j^{II}(n))(1 - y_j^{II}(n)) y_i^I(n)$$

Generalizando para la capa “ p ”

$$\Delta w_{ji}^H(n) = \eta \left[\sum_k \delta_k^H w_{kj}^H(n) \right] (1 + y_j^H(n))(1 - y_j^H(n)) y_i^I(n)$$

⇓

$$\Delta w_{ji}^{(p)}(n) = \eta \left\langle \delta^{(p+1)}, \mathbf{w}_j^{(p+1)} \right\rangle (1 + y_j^{(p)}(n))(1 - y_j^{(p)}(n)) y_i^{(p-1)}(n)$$

Resumen del algoritmo de retropropagación (BP)

1. Inicialización aleatoria
2. Propagación hacia adelante (de la entrada)
3. Propagación hacia atrás (del error)
4. Adaptación de los pesos
5. Iteración: vuelve a 2 hasta convergencia o finalización

Ejemplo gráfico BP con PMC 3 capas

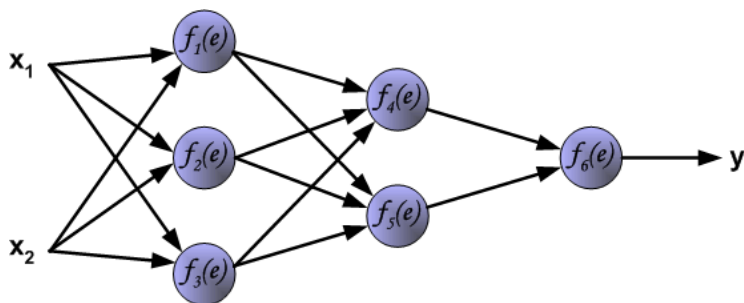


Figura: Ejemplo de un PMC de 3 capas.

Ejemplo: Cálculo de las salidas en cada capa

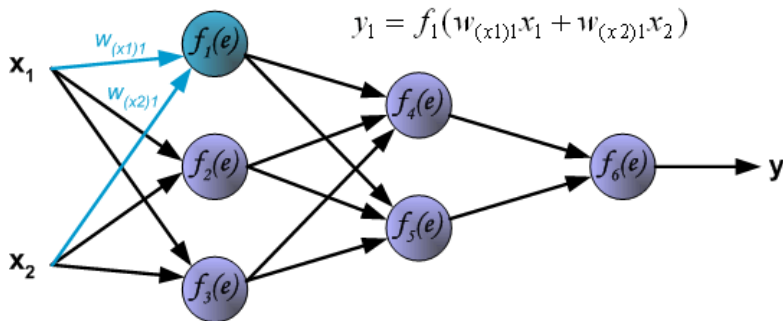


Figura: Cálculo salida capa I, neurona 1.

Ejemplo: Cálculo de las salidas en cada capa

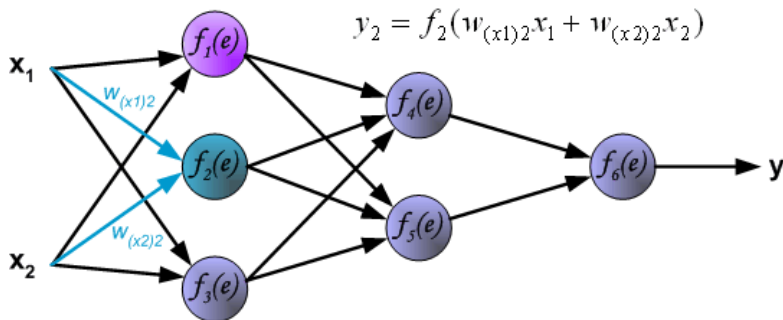


Figura: Cálculo salida capa I, neurona 2.

Ejemplo: Cálculo de las salidas en cada capa

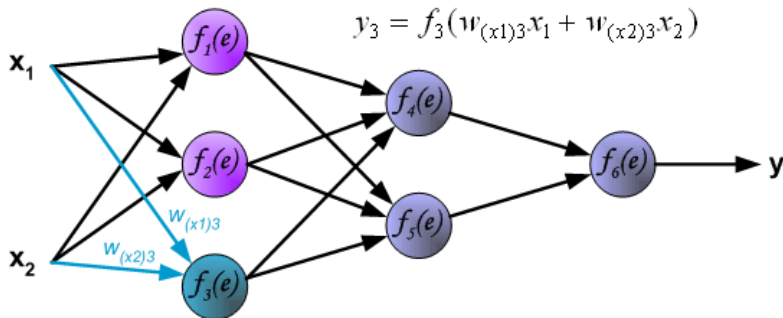


Figura: Cálculo salida capa I, neurona 3.

Ejemplo: Cálculo de las salidas en cada capa

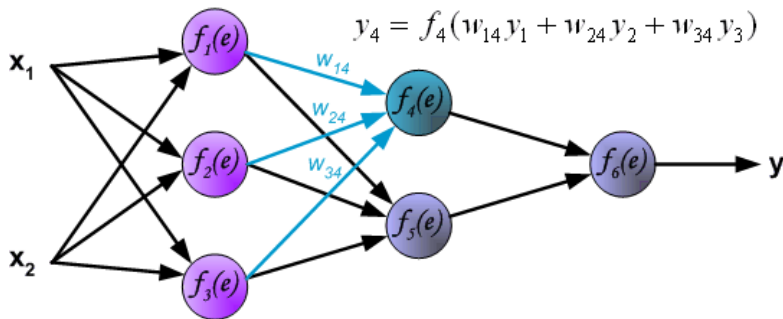


Figura: Cálculo salida capa II, neurona 1.

Ejemplo: Cálculo de las salidas en cada capa

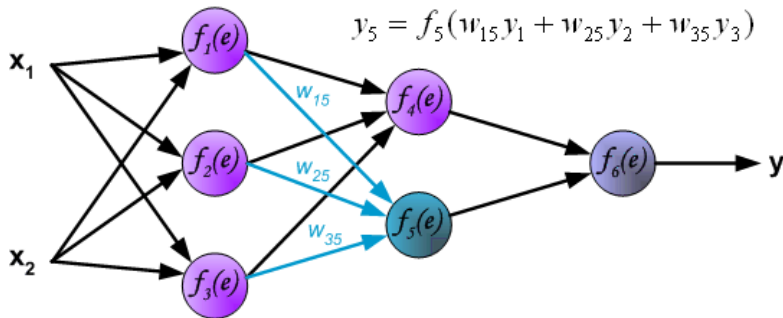


Figura: Cálculo salida capa II, neurona 2.

Ejemplo: Cálculo de las salidas en cada capa

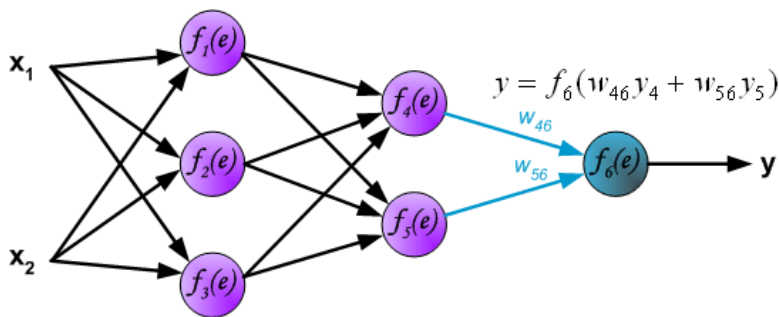


Figura: Cálculo salida capa III, neurona 1.

Ejemplo: Retropropagación en la capa III (salida)

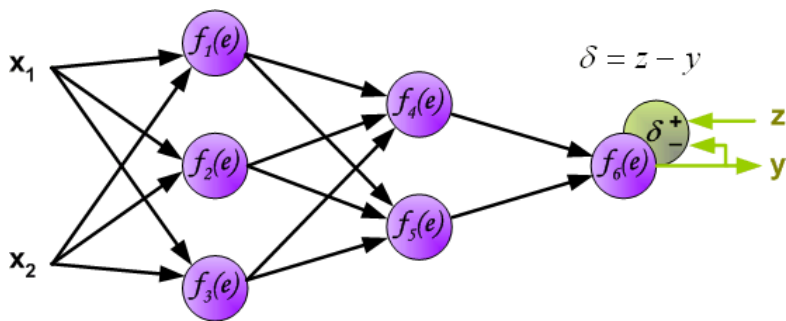


Figura: Cálculo del error en capa III, neurona 1.

Ejemplo: Retropropagación en la capa III (salida)

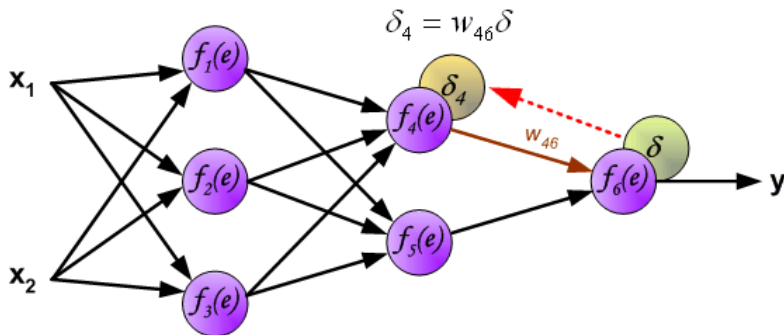


Figura: Propagación del error a la capa II, neurona 1.

Ejemplo: Retropropagación en la capa III (salida)

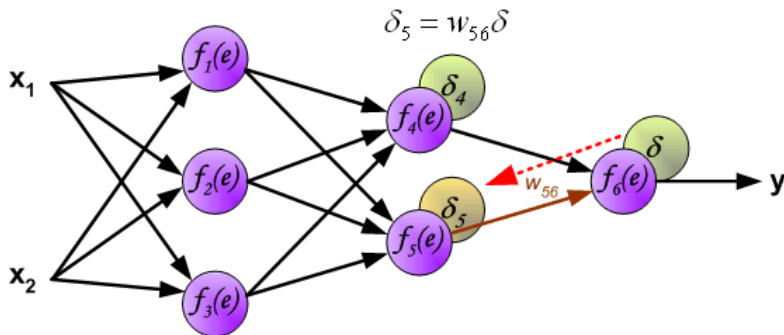


Figura: Propagación del error a la capa II, neurona 2.

Ejemplo: Retropropagación en la capa II (oculta)

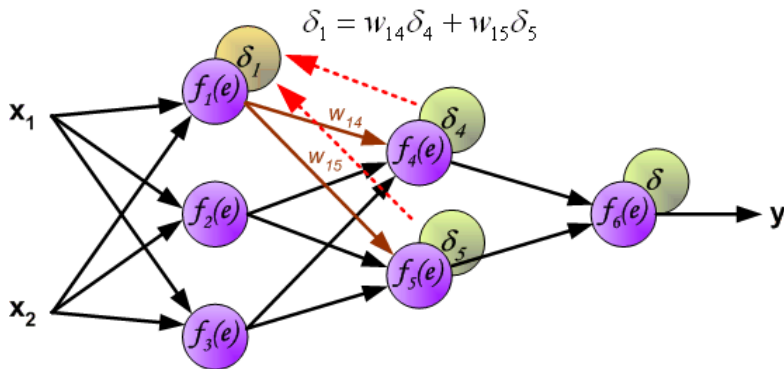


Figura: Propagación del error a la capa I, neurona 1.

Ejemplo: Retropropagación en la capa II (oculta)

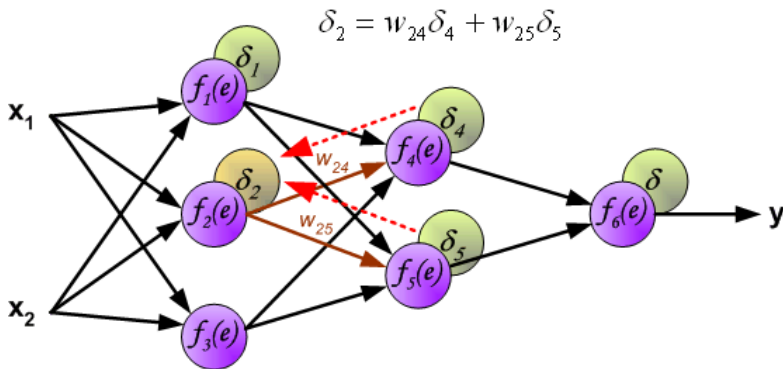


Figura: Propagacion del error a la capa I, neurona 2.

Ejemplo: Retropropagación en la capa II (oculta)

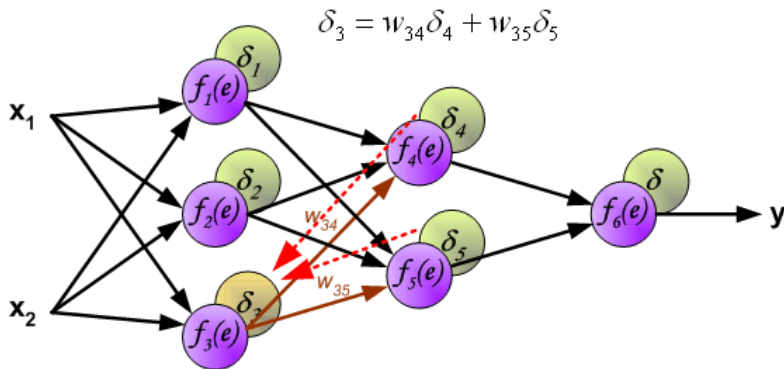


Figura: Propagación del error a la capa I, neurona 3.

Ejemplo: Actualizando los pesos de la red

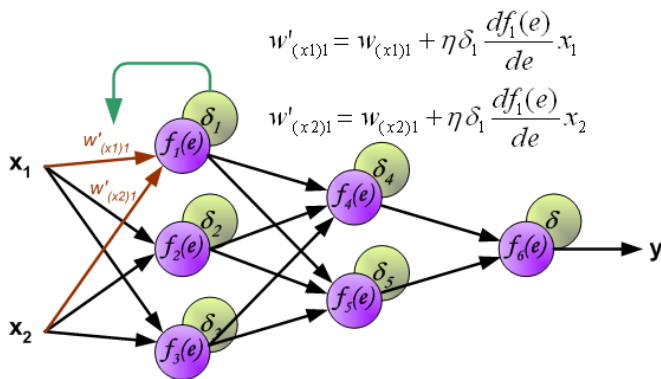


Figura: Actualización de pesos capa I, neurona 1.

Ejemplo: Actualizando los pesos de la red

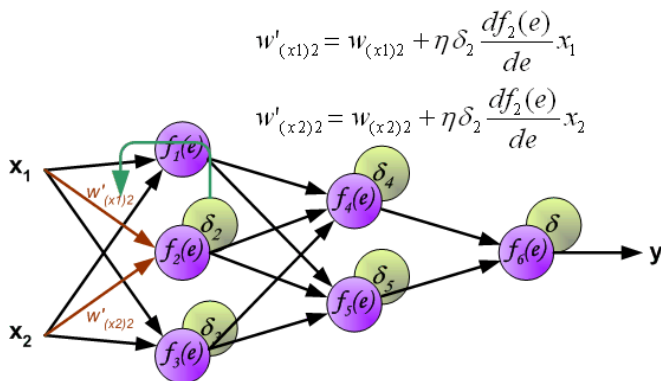


Figura: Actualización de pesos capa I, neurona 2.

Ejemplo: Actualizando los pesos de la red

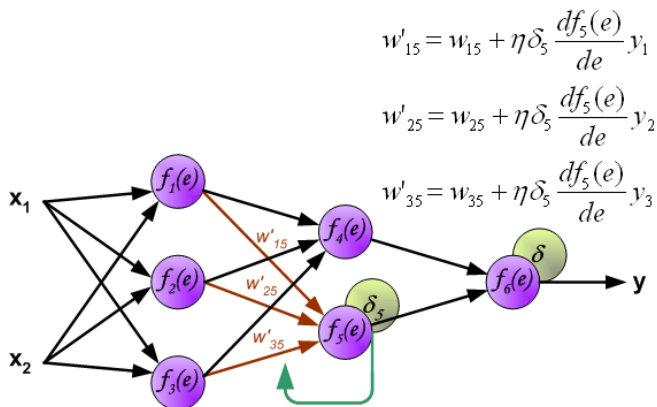


Figura: Actualización de pesos capa II, neurona 2.

Ejemplo: Actualizando los pesos de la red

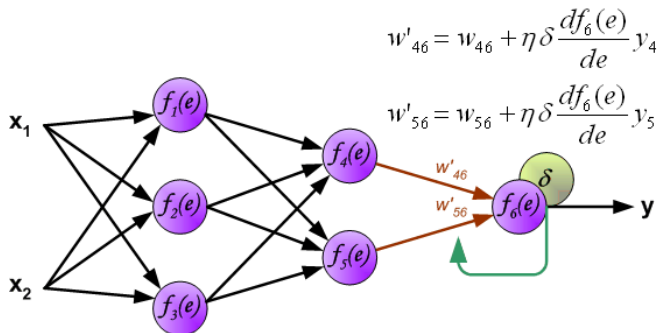


Figura: Actualización de pesos capa III, neurona 1.

Término de momento

Modificación adaptativa de la velocidad de aprendizaje.
(ver Haykin Sección 6.3)