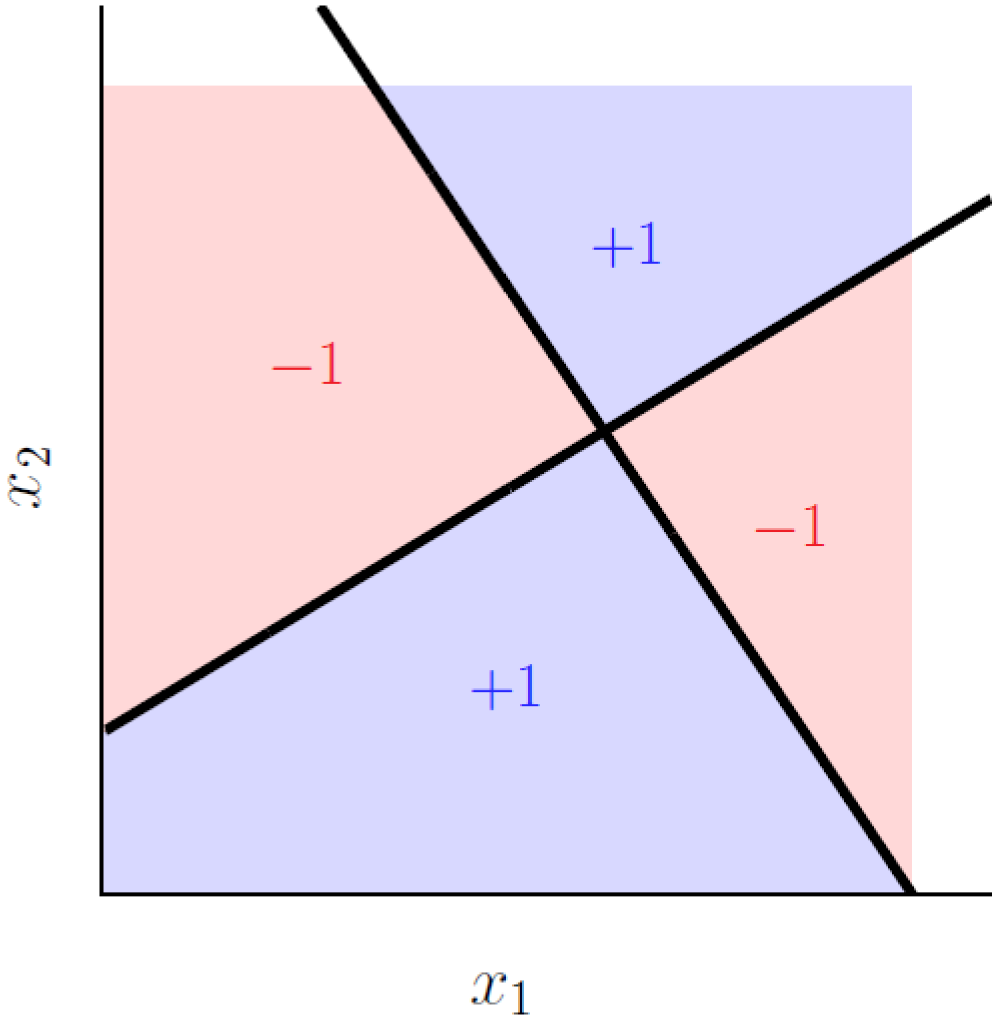
IIC 2433 Minería de Datos

https://github.com/marcelomendoza/IIC2433

- MLP -

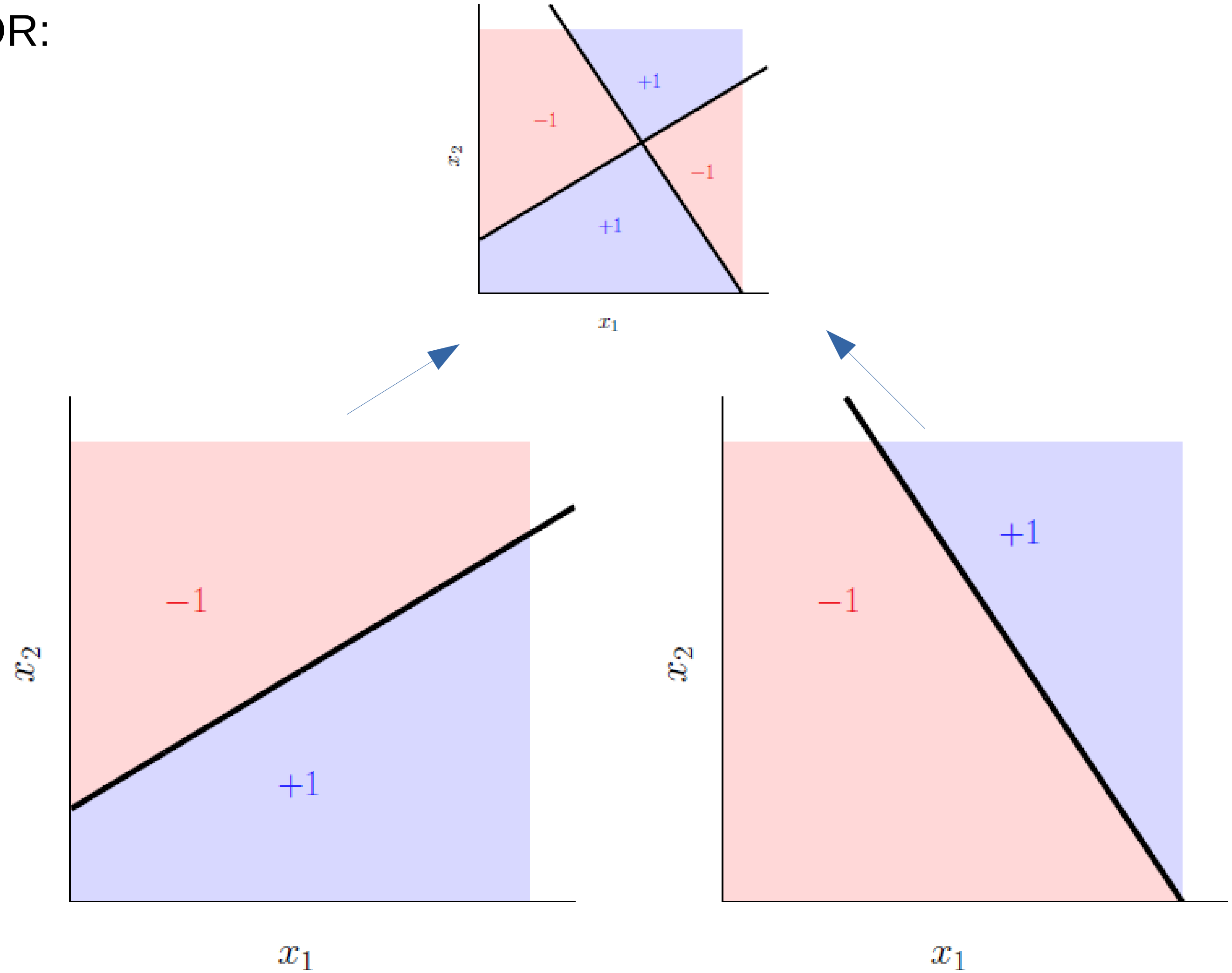
La limitación del modelo lineal

XOR:



Combinación de perceptrones

XOR:

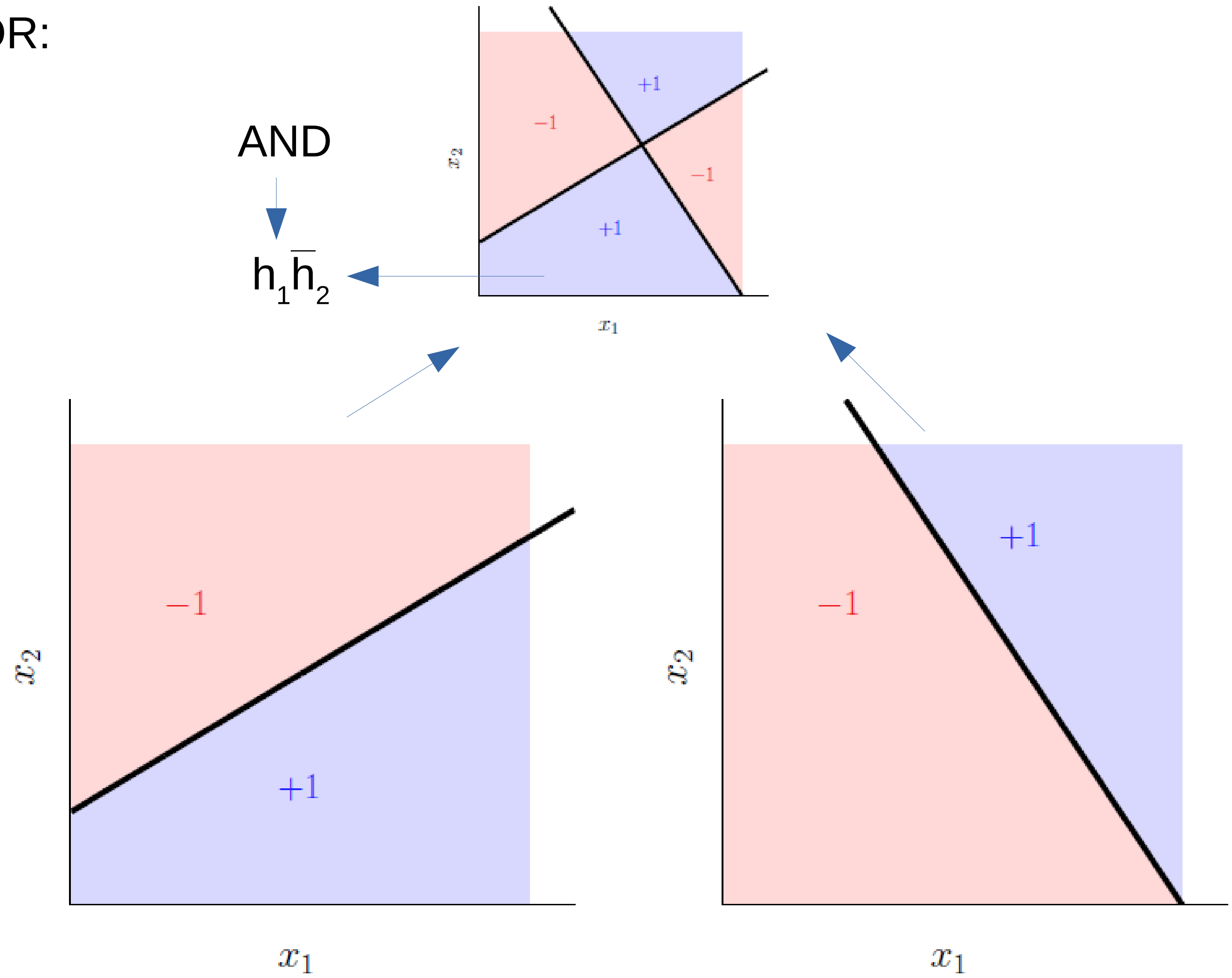


$$h_1(\mathbf{x}) = \text{sign}(\mathbf{w}_1^{\mathrm{T}}\mathbf{x})$$

$$h_2(\mathbf{x}) = \text{sign}(\mathbf{w}_2^{\mathrm{T}}\mathbf{x})$$

Combinación de perceptrones

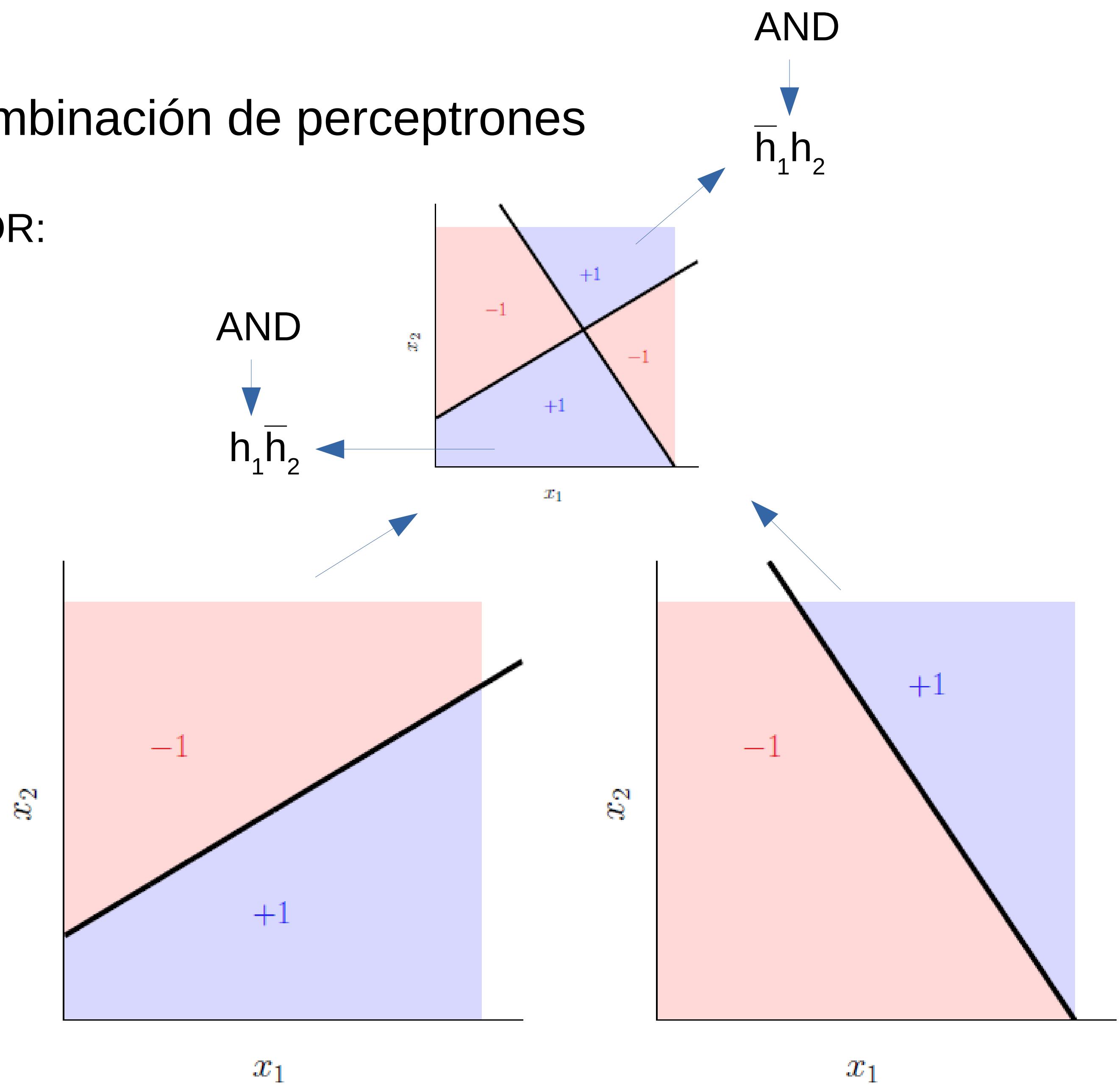
XOR:



$$h_1(\mathbf{x}) = \mathrm{sign}(\mathbf{w}_1^{\mathrm{T}}\mathbf{x}) \qquad h_2(\mathbf{x}) = \mathrm{sign}(\mathbf{w}_2^{\mathrm{T}}\mathbf{x})$$

Combinación de perceptrones

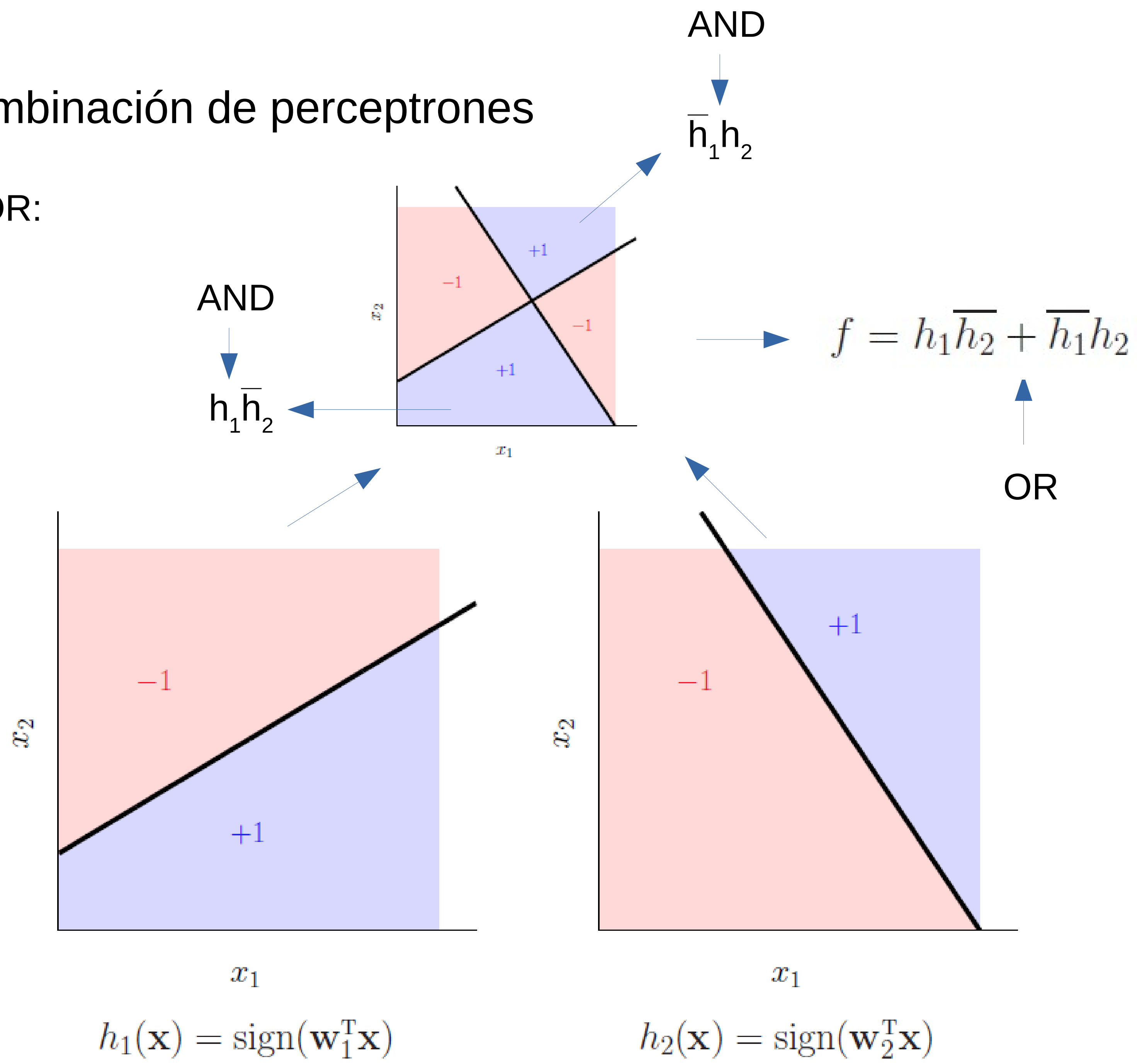
XOR:



$h_1(\mathbf{x}) = \mathrm{sign}(\mathbf{w}_1^{\mathrm{T}}\mathbf{x})$ $h_2(\mathbf{x}) = \mathrm{sign}(\mathbf{w}_2^{\mathrm{T}}\mathbf{x})$

Combinación de perceptrones

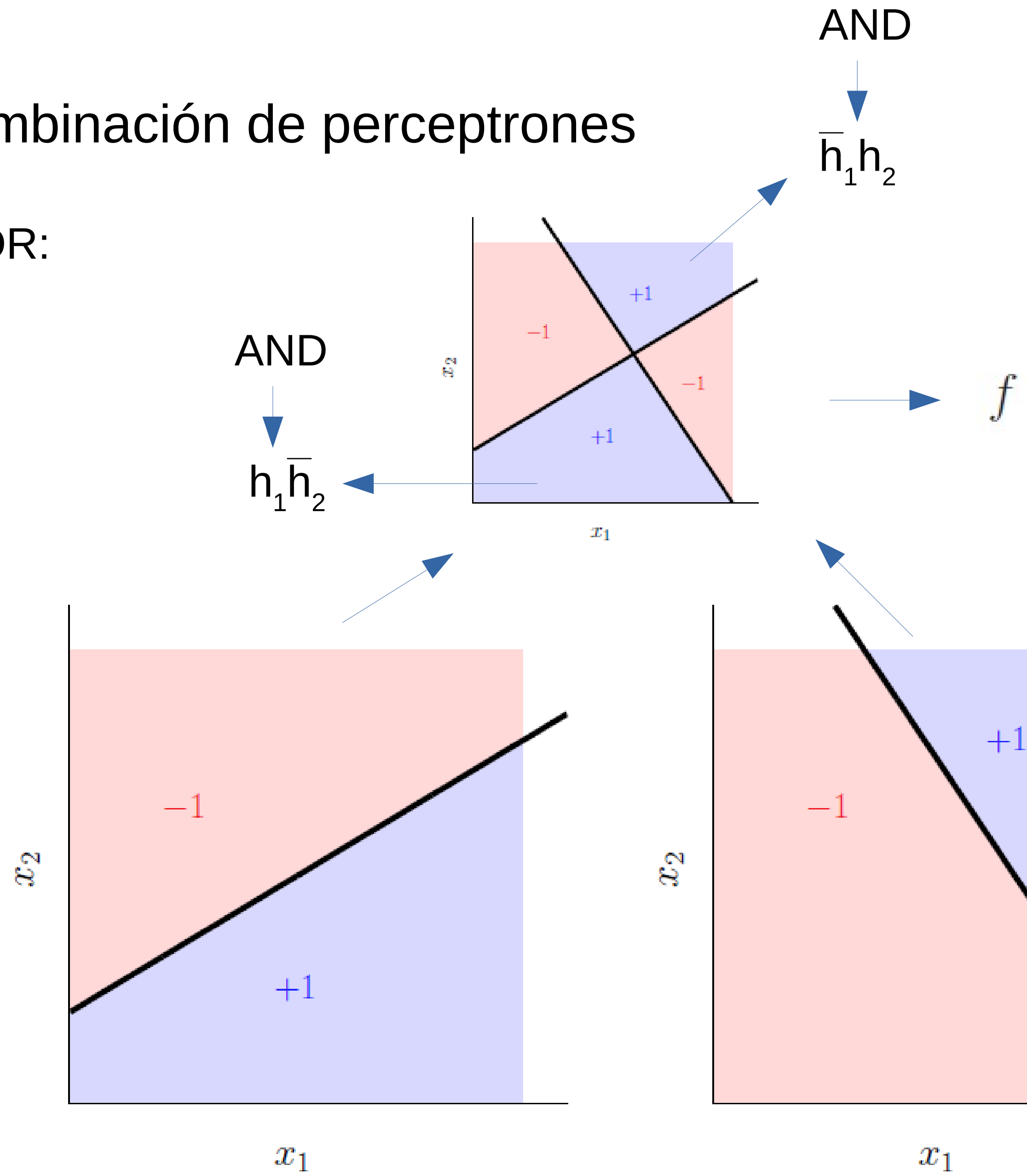
XOR:



Combinación de perceptrones

XOR:

h_1	h_2	XOR
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	-1



$$f = h_1\overline{h_2} + \overline{h_1}h_2$$

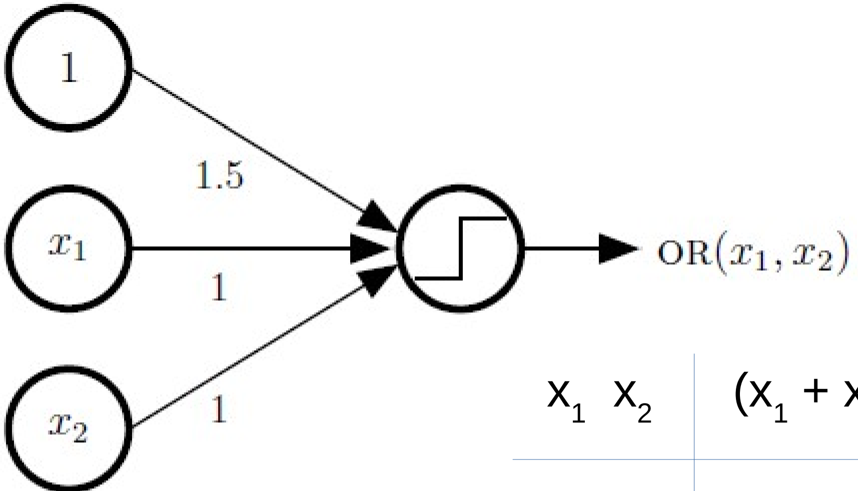
OR

$$h_1(\mathbf{x}) = \text{sign}(\mathbf{w}_1^T\mathbf{x}) \qquad h_2(\mathbf{x}) = \text{sign}(\mathbf{w}_2^T\mathbf{x})$$

Combinación de perceptrones

OR:

$$\text{OR}(x_1, x_2) = \text{sign}(x_1 + x_2 + 1.5)$$

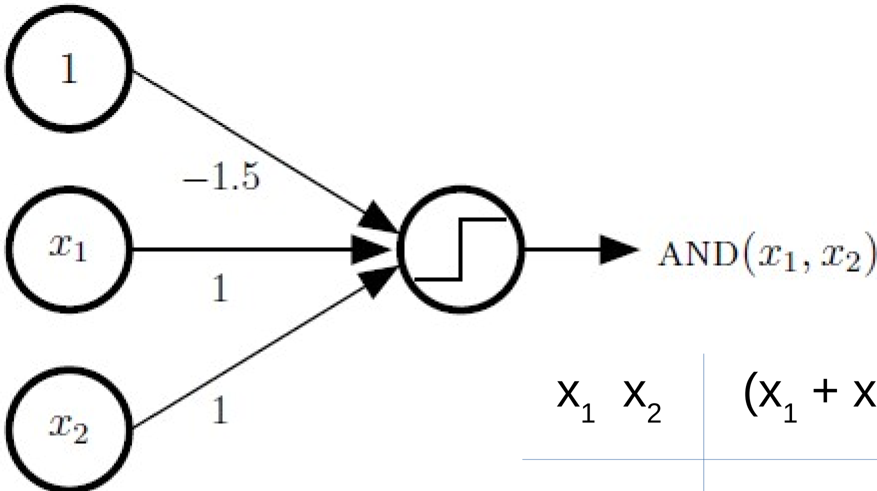


x_1 x_2	$(x_1 + x_2 + 1.5)$	signo
-1 -1	-0.5	-1
-1 +1	1.5	+1
+1 -1	1.5	+1
+1 +1	3.5	+1

Combinación de perceptrones

AND:

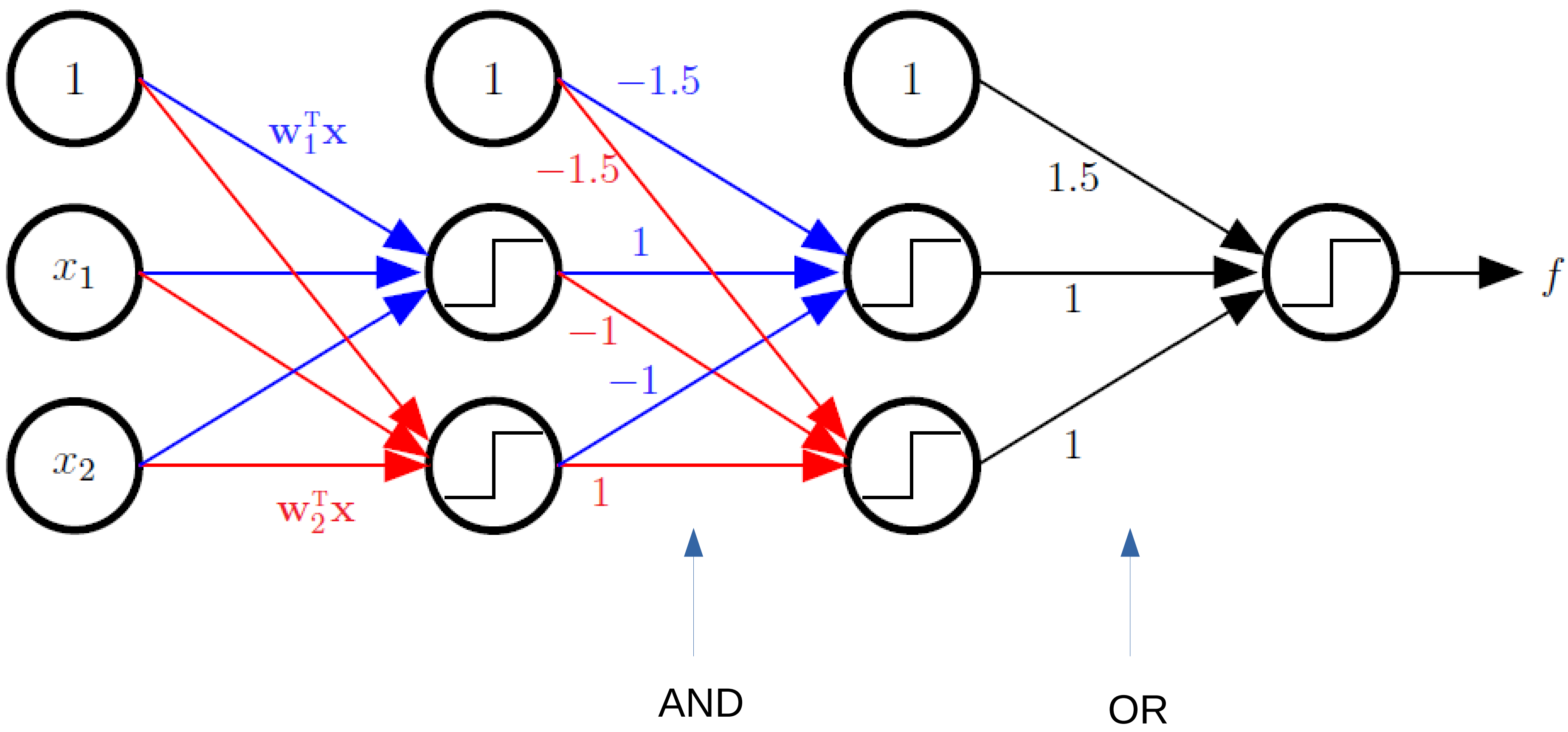
$$\mathrm{AND}(x_1, x_2) = \mathrm{sign}(x_1 + x_2 - 1.5)$$



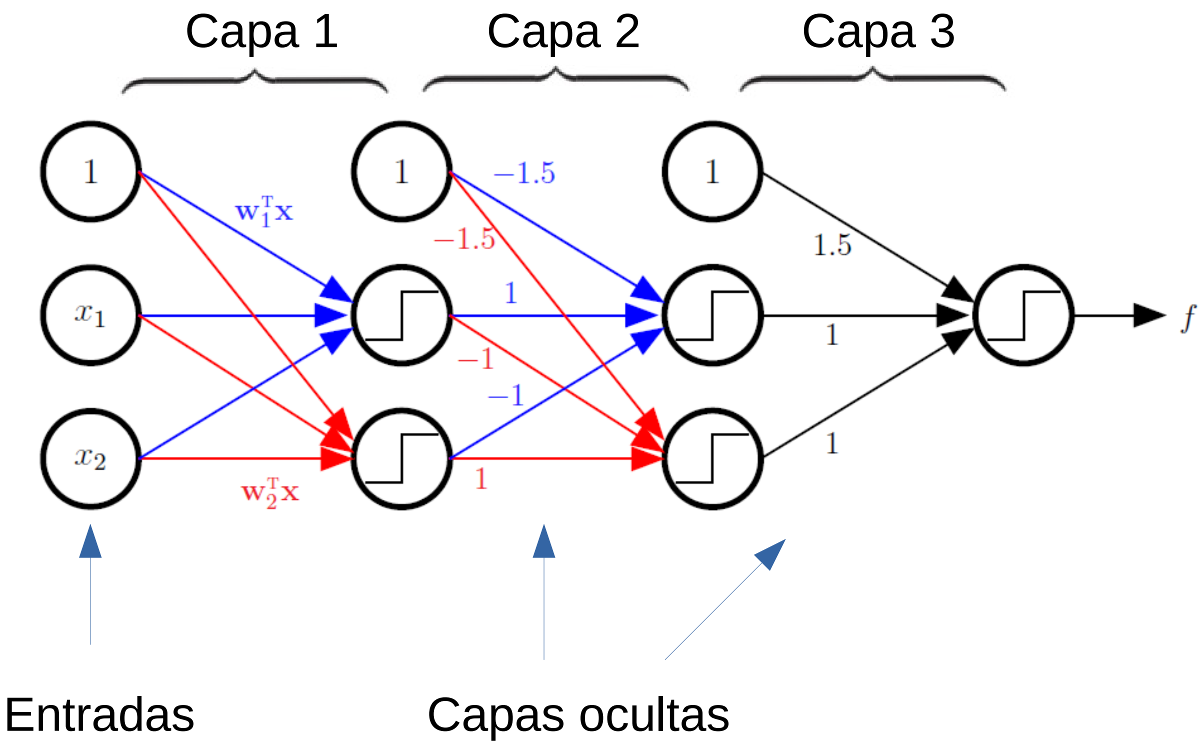
x_1 x_2	$(x_1 + x_2 - 1.5)$	signo
-1 -1	-3.5	-1
-1 +1	-1.5	-1
+1 -1	-1.5	-1
+1 +1	0.5	+1

Combinación de perceptrones

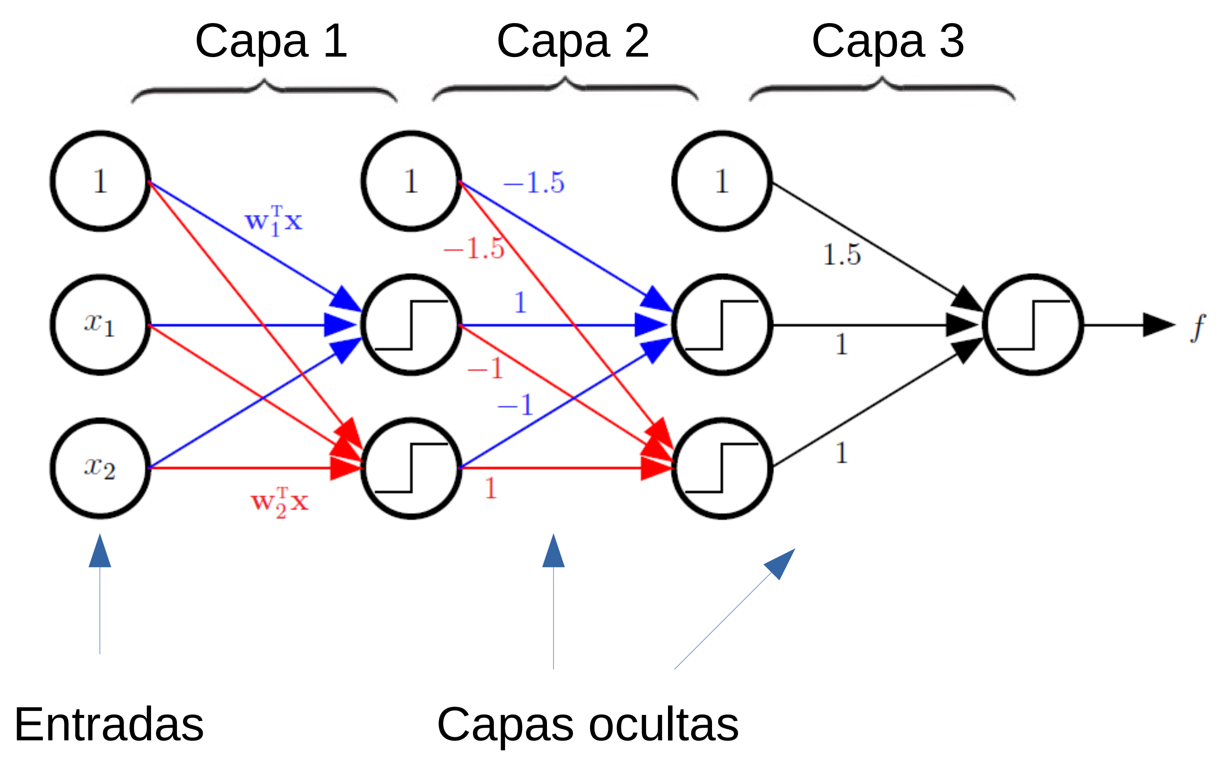
$$f = h_1\overline{h_2} + \overline{h_1}h_2$$



El perceptrón multicapa (Multi-Layer Perceptron)



El perceptrón multicapa (Multi-Layer Perceptron)

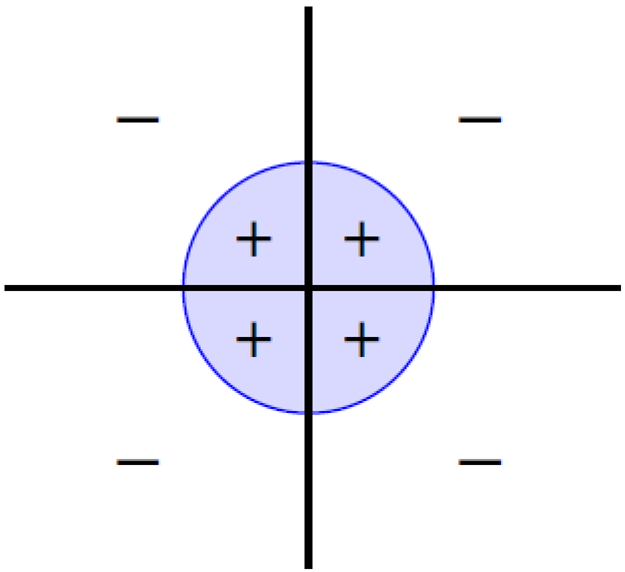


Aproximación universal

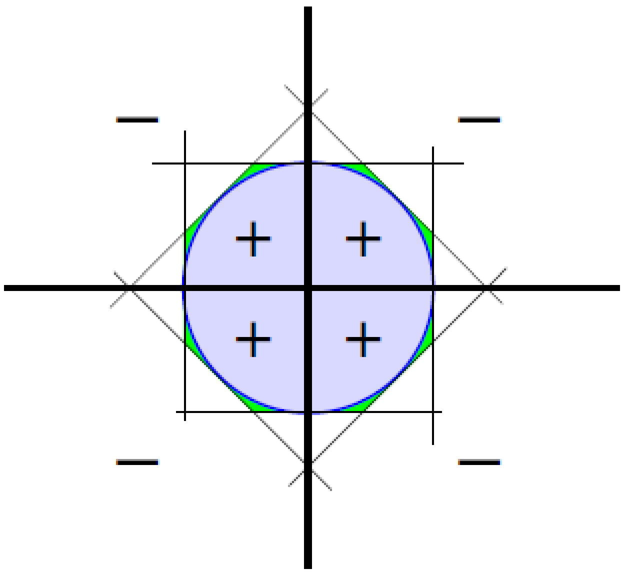
Cualquier función que se puede descomponer en separadores lineales puede ser implementada por un MLP de 3 capas

El perceptrón multicapa (Multi-Layer Perceptron)

Un separador suave puede ser aproximado por N separadores lineales.



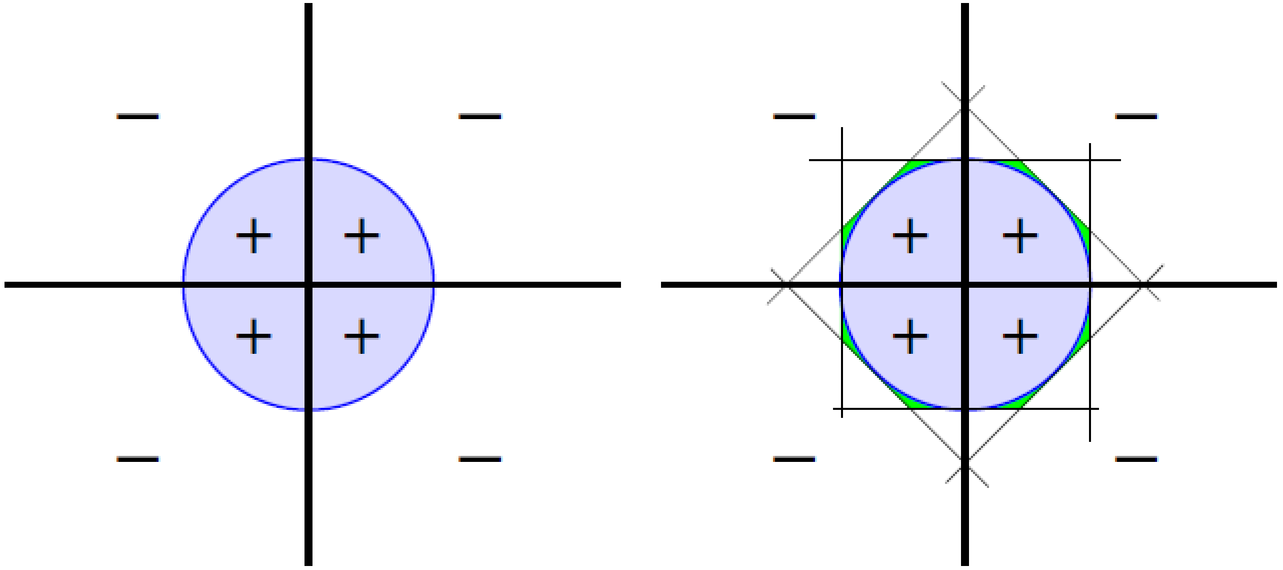
f



¿Cuántos perceptrones hay?

El perceptrón multicapa (Multi-Layer Perceptron)

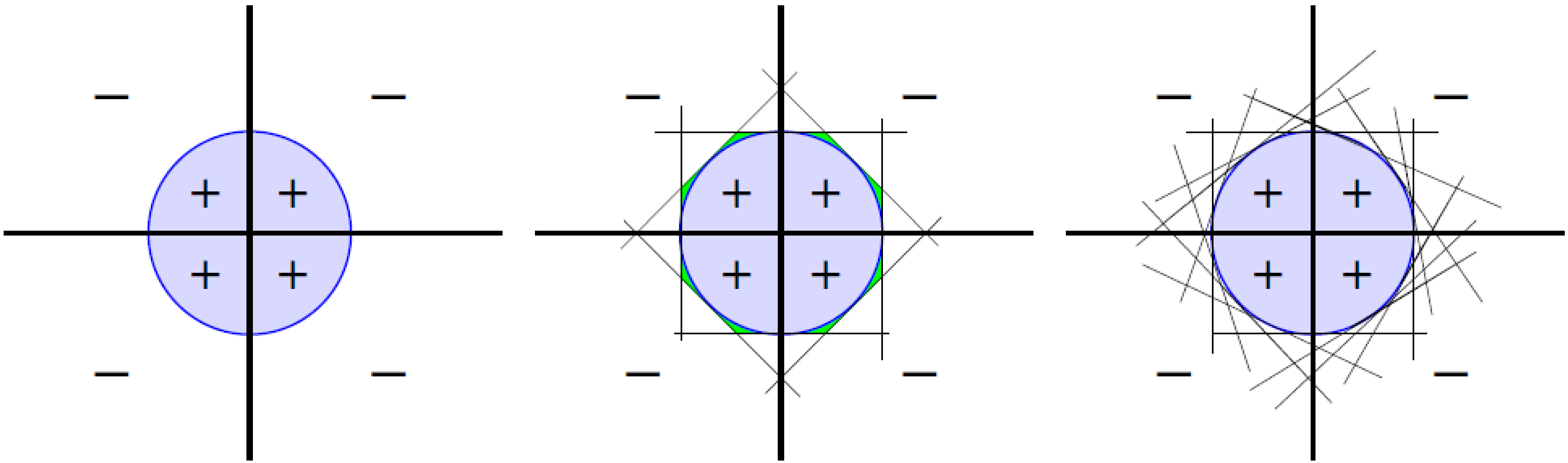
Un separador suave puede ser aproximado por N separadores lineales.



f 8

El perceptrón multicapa (Multi-Layer Perceptron)

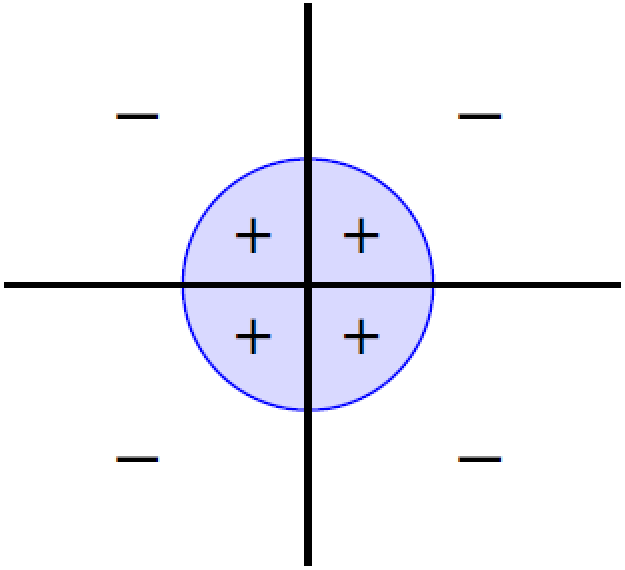
Un separador suave puede ser aproximado por N separadores lineales.



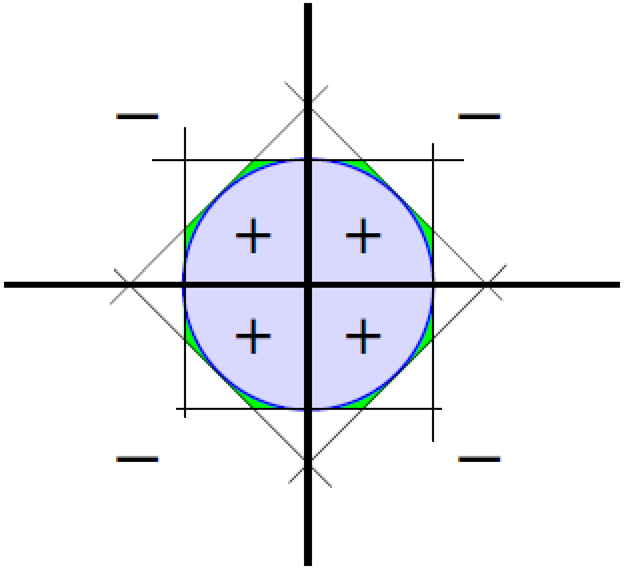
f 8 ¿?

El perceptrón multicapa (Multi-Layer Perceptron)

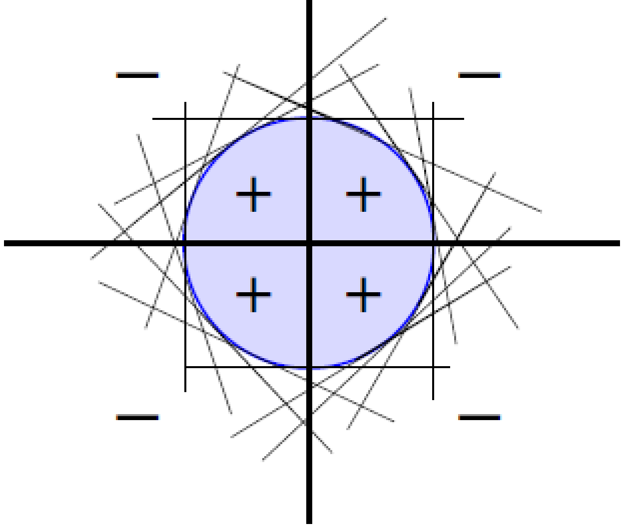
Un separador suave puede ser aproximado por N separadores lineales.



f



8



¿?

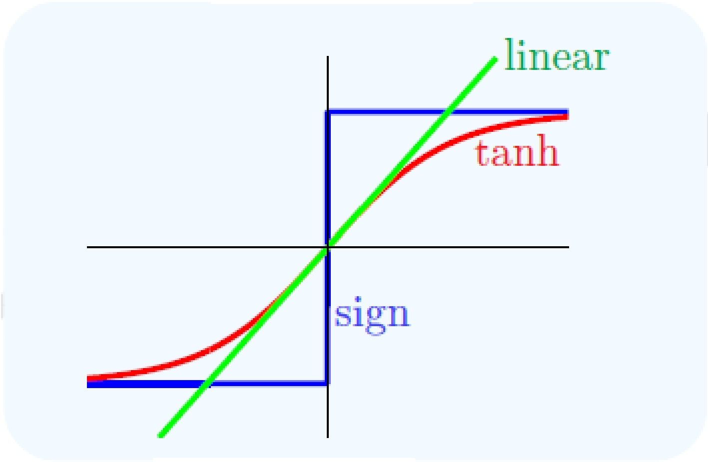
Tradeoff aproximación-generalización:

Más neuronas, mejor aproximación.
Más neuronas, más datos.

El perceptrón multicapa (Multi-Layer Perceptron)

Para entrenar un MLP necesitamos reemplazar la función signo dado que no es diferenciable.

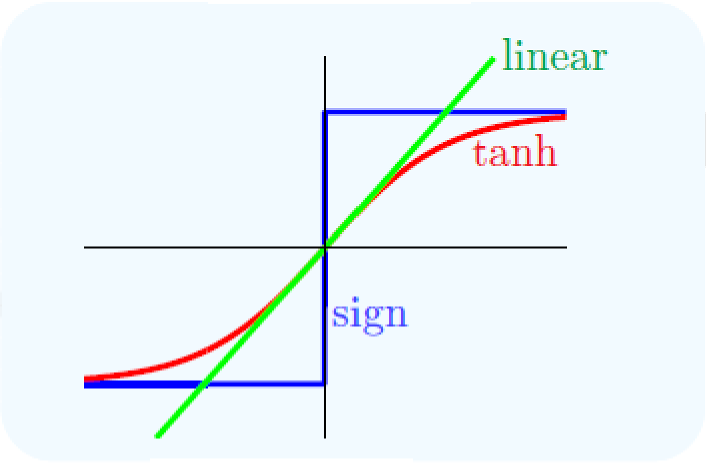
Podemos suavizar la función signo con la tangente hiperbólica, que tiene un comportamiento lineal cerca del origen y es cercana a +1 o -1 para entradas grandes.

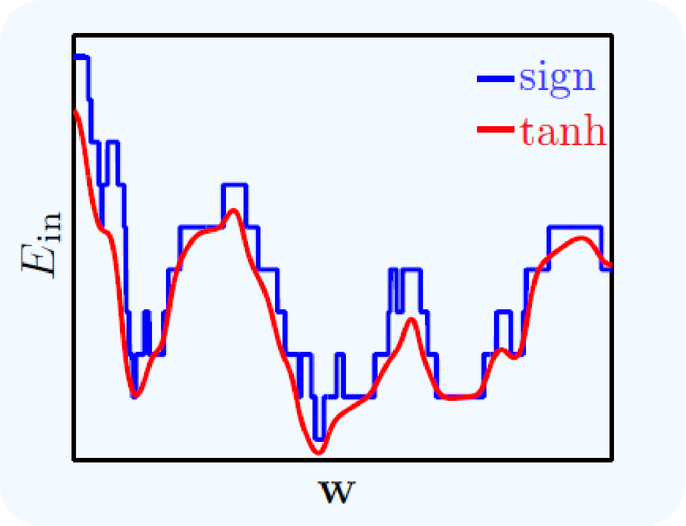


El perceptrón multicapa (Multi-Layer Perceptron)

Para entrenar un MLP necesitamos reemplazar la función signo dado que no es diferenciable.

Podemos suavizar la función signo con la tangente hiperbólica, que tiene un comportamiento lineal cerca del origen y es cercana a +1 o -1 para entradas grandes.





- BACKPROPAGATION -

El perceptrón multicapa (Multi-Layer Perceptron)

Forward propagation

$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{\mathrm{W}^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{\mathrm{W}^{(2)}} \mathbf{s}^{(2)} \xrightarrow{\theta} \mathbf{x}^{(2)} \cdots \xrightarrow{\mathrm{W}^{(L)}} \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x}).$$

Forward propagation

$\mathbf{x}^{(0)} \leftarrow \mathbf{x}$

for $\ell = 1$ to L **do**

$\quad \mathbf{s}^{(\ell)} \leftarrow (\mathrm{W}^{(\ell)})^{\mathrm{T}} \mathbf{x}^{(\ell-1)}$

$\quad \mathbf{x}^{(\ell)} \leftarrow \begin{bmatrix} 1 \\ \theta(\mathbf{s}^{(\ell)}) \end{bmatrix}$

end for

$h(\mathbf{x}) = \mathbf{x}^{(L)}$

Objetivo supervisado:

$$E_{\text{in}}(h) = E_{\text{in}}(\mathrm{W}) = \frac{1}{N} \sum_{n=1}^{N} (h(\mathbf{x}_n) - y_n)^2$$

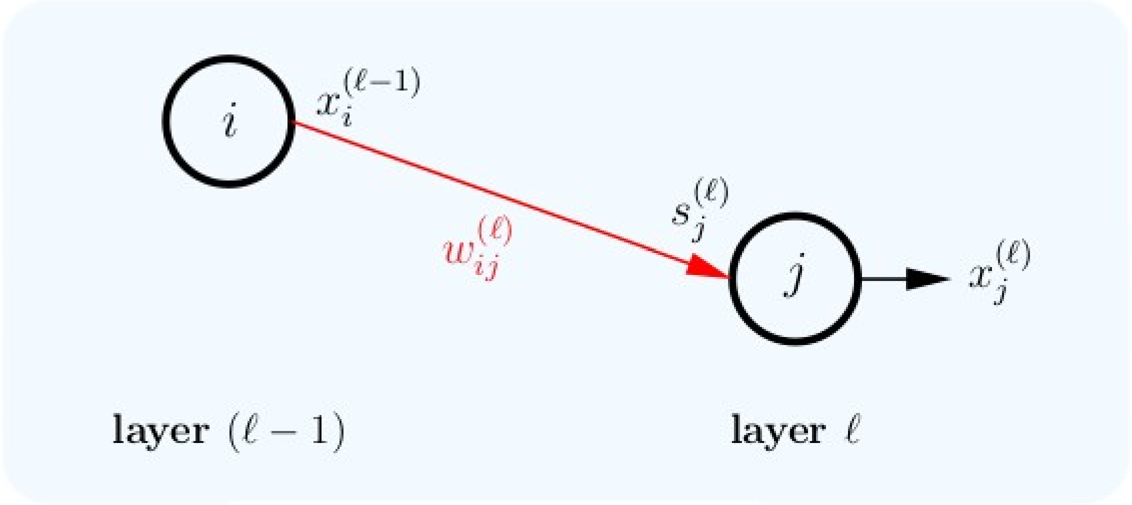
Dado que $\theta = \tanh$, E_{in} es diferenciable usando GD sobre

$$\underbrace{\mathrm{W} = \{\mathrm{W}^{(1)}, \mathrm{W}^{(2)}, \ldots, \mathrm{W}^{(L)}\}}_{\text{Parámetros del modelo}}$$

Feed-Forward

$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{\mathrm{W}^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{\mathrm{W}^{(2)}} \mathbf{s}^{(2)} \xrightarrow{\theta} \mathbf{x}^{(2)} \cdots \xrightarrow{\mathrm{W}^{(L)}} \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x}).$$

Nodo a nodo:

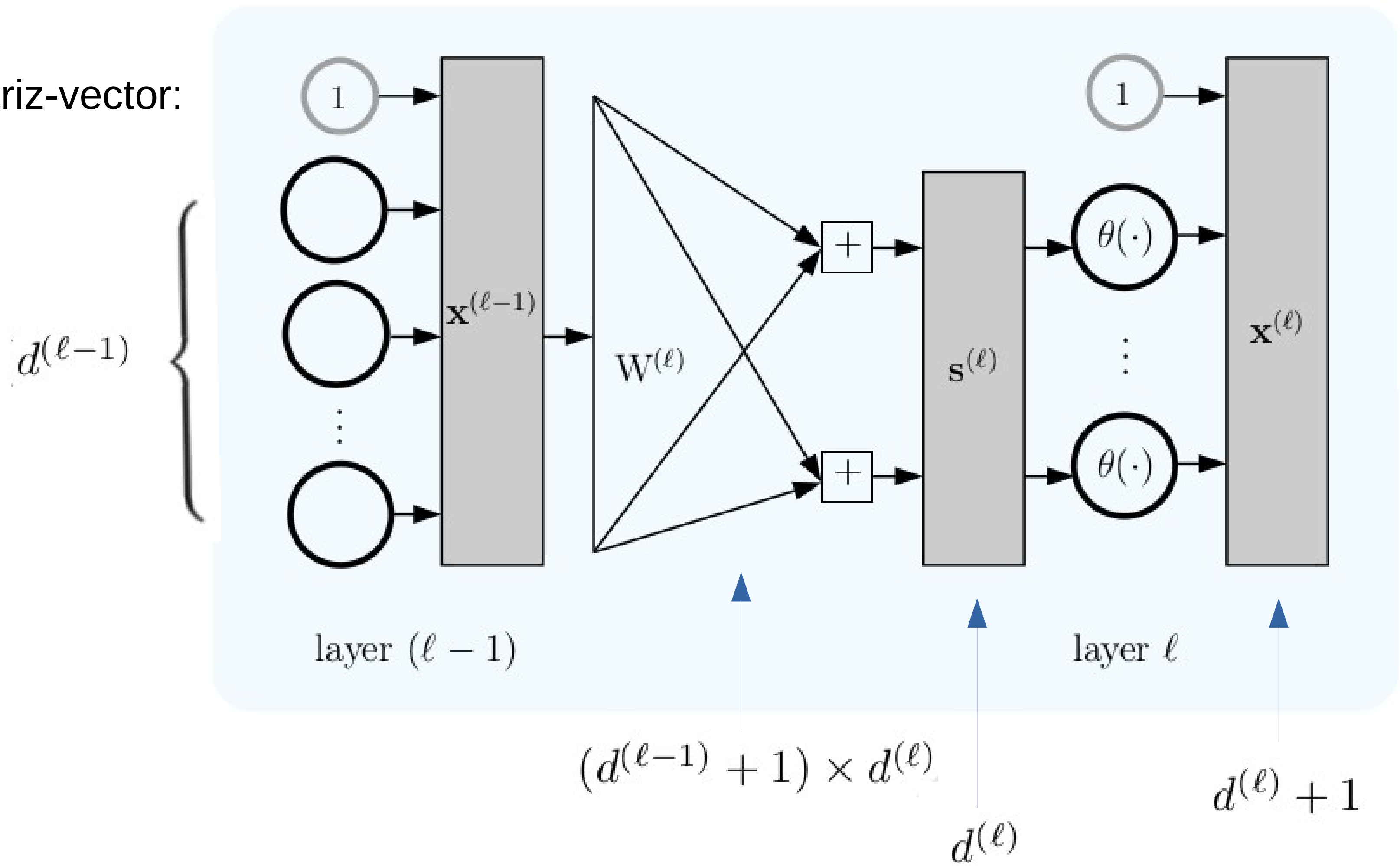


Feed-Forward

$$\mathbf{x} = \mathbf{x}^{(0)} \xrightarrow{\mathrm{W}^{(1)}} \mathbf{s}^{(1)} \xrightarrow{\theta} \mathbf{x}^{(1)} \xrightarrow{\mathrm{W}^{(2)}} \mathbf{s}^{(2)} \xrightarrow{\theta} \mathbf{x}^{(2)} \cdots \xrightarrow{\mathrm{W}^{(L)}} \mathbf{s}^{(L)} \xrightarrow{\theta} \mathbf{x}^{(L)} = h(\mathbf{x}).$$

$$\mathbf{s}^{(\ell)} \leftarrow (\mathrm{W}^{(\ell)})^{\mathrm{T}} \mathbf{x}^{(\ell-1)}$$

Matriz-vector:



Backpropagation

Minimizar: $$E_{\text{in}}(h) = E_{\text{in}}(\mathrm{W}) = \frac{1}{N}\sum_{n=1}^{N}(h(\mathbf{x}_n) - y_n)^2$$

Podemos usar la idea de gradiente descendente:

$$\mathrm{W}(t+1) = \mathrm{W}(t) - \eta \nabla E_{\text{in}}(\mathrm{W}(t))$$

Dado que:

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}\overset{\mathsf{e}_n}{\mathsf{e}}(h(\mathbf{x}_n), y_n)$$

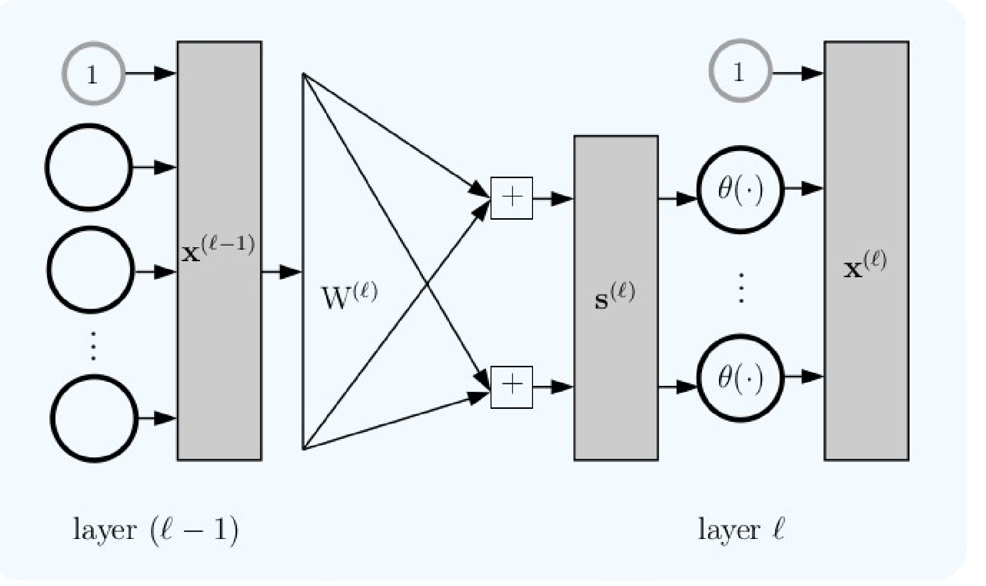
$$\frac{\partial E_{\text{in}}(\mathbf{w})}{\partial \mathrm{W}^{(\ell)}} = \frac{1}{N}\sum_{n=1}^{N}\frac{\partial \mathsf{e}_n}{\partial \mathrm{W}^{(\ell)}}$$

Necesitamos:

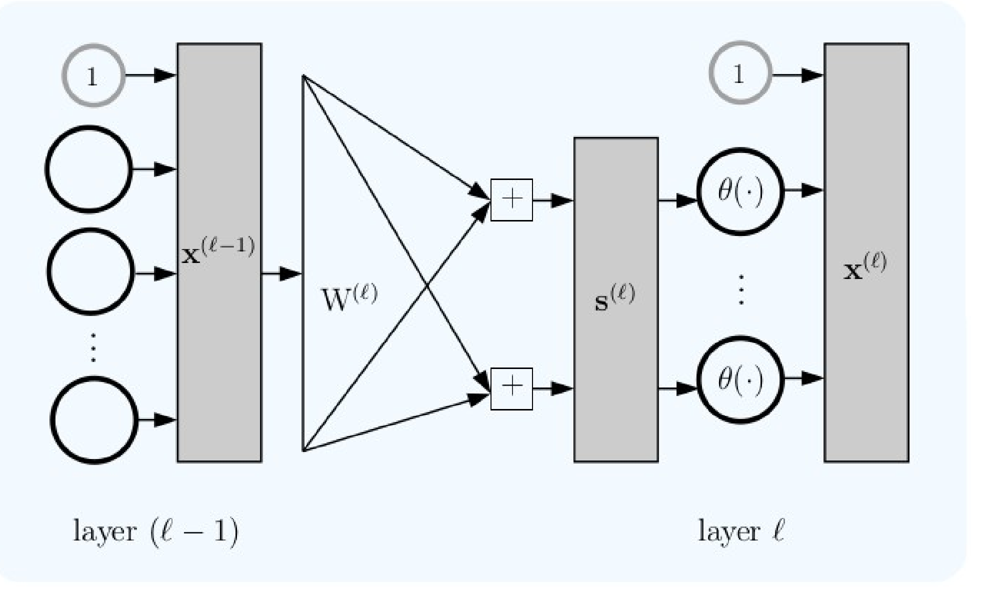
$$\frac{\partial \mathsf{e}(\mathbf{x})}{\partial \mathrm{W}^{(\ell)}}$$

Backpropagation

Vamos a usar la **regla de la cadena** para expresar las derivadas parciales de la capa $(\ell-1)$ en función de las derivadas parciales de la capa ℓ .



Backpropagation



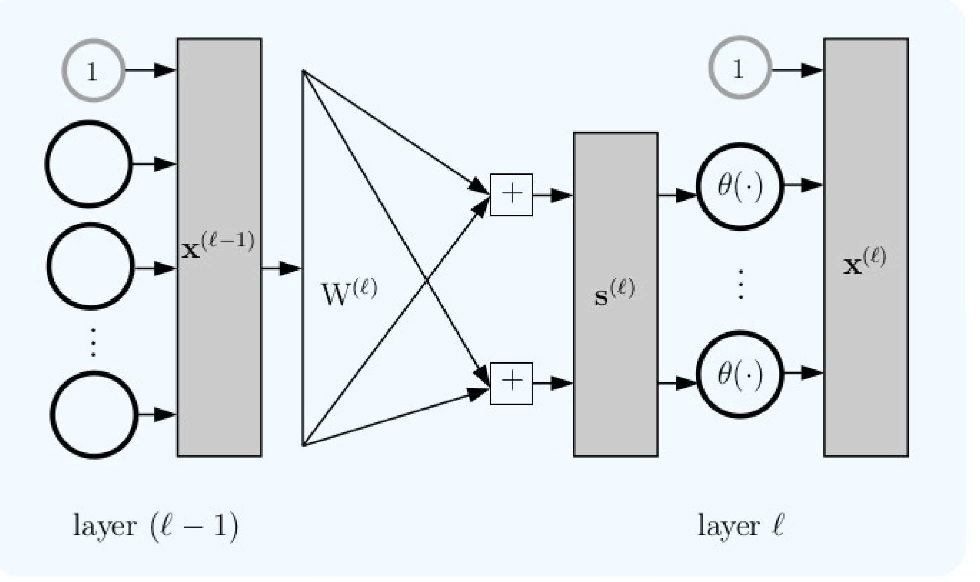
Vamos a usar la **regla de la cadena** para expresar las derivadas parciales de la capa $(\ell-1)$ en función de las derivadas parciales de la capa ℓ .

Tenemos: $\mathbf{s}^{(\ell)} = (\mathrm{W}^{(\ell)})^{\mathrm{T}}\mathbf{x}^{(\ell-1)}$

Definimos la sensibilidad de la capa ℓ :

$$\boldsymbol{\delta}^{(\ell)} = \frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}$$

Backpropagation



Vamos a usar la **regla de la cadena** para expresar las derivadas parciales de la capa $(\ell-1)$ en función de las derivadas parciales de la capa ℓ .

Tenemos: $\mathbf{s}^{(\ell)} = (\mathrm{W}^{(\ell)})^{\mathrm{T}}\mathbf{x}^{(\ell-1)}$

Definimos la sensibilidad de la capa ℓ :

$$\boldsymbol{\delta}^{(\ell)} = \frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}$$

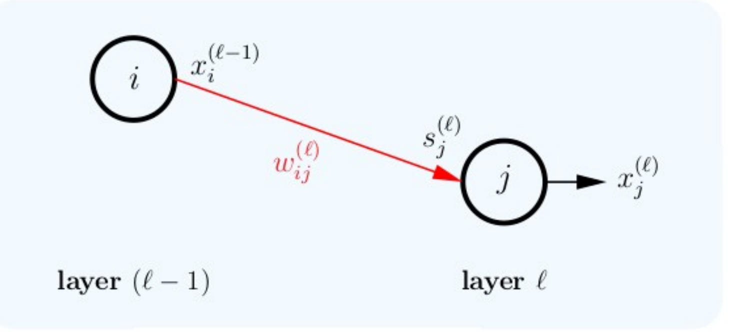
Aplicando la regla de la cadena:

$$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

?

Backpropagation



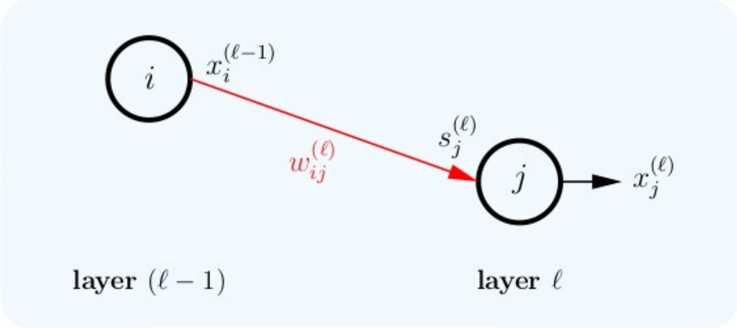
Miremos esto:

$$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$

a nivel de un enlace.

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

Backpropagation

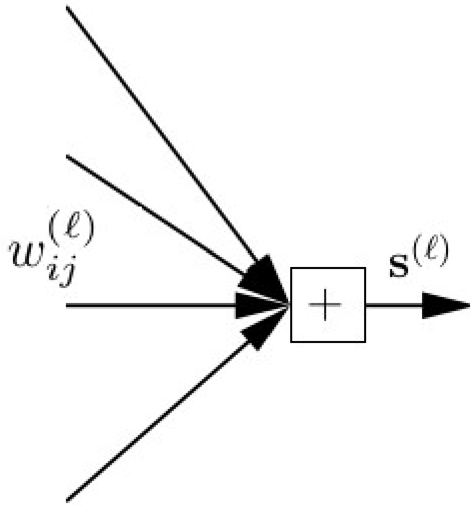


Miremos esto: $$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$ a nivel de un enlace.

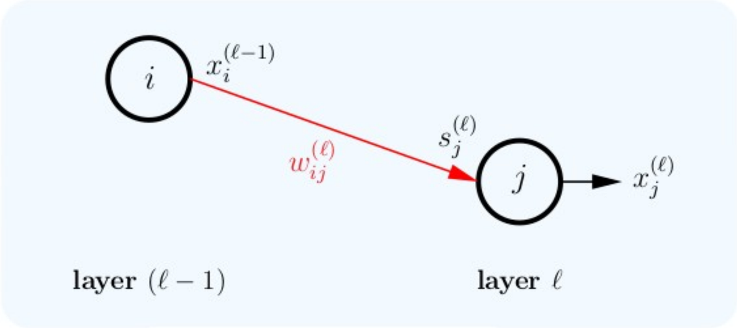
$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

Tenemos: $$\frac{\partial \mathrm{e}}{\partial\, w_{ij}^{(\ell)}} = \frac{\partial \mathbf{s}_j^{(\ell)}}{\partial w_{ij}^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$

y sabemos que: $$\mathbf{s}_j^{(\ell)} = \sum_{\alpha=0}^{d^{(\ell-1)}} w_{\alpha j}^{(\ell)} \mathbf{x}_\alpha^{(\ell-1)}$$



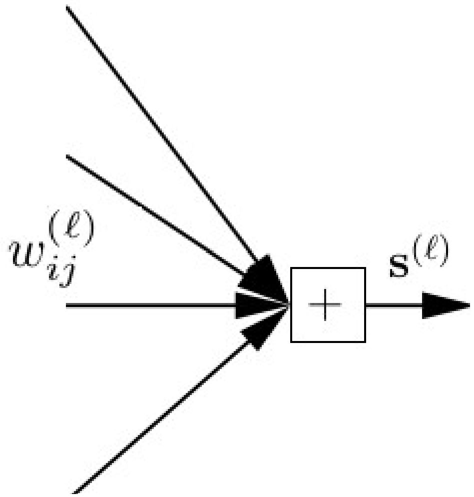
Backpropagation



Miremos esto: $$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$ a nivel de un enlace.

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

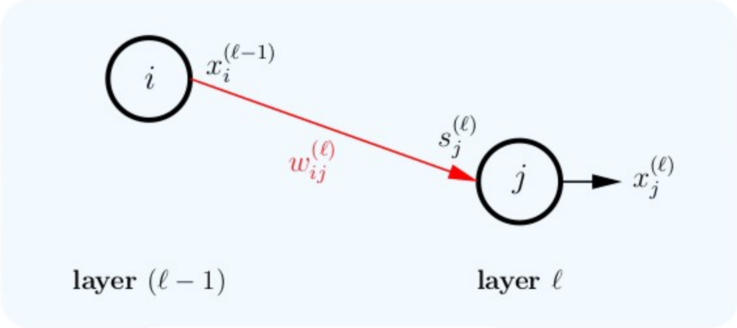
Tenemos: $$\frac{\partial \mathrm{e}}{\partial\, w_{ij}^{(\ell)}} = \frac{\partial \mathbf{s}_j^{(\ell)}}{\partial w_{ij}^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$



y sabemos que: $\mathbf{s}_j^{(\ell)} = \sum_{\alpha=0}^{d^{(\ell-1)}} w_{\alpha j}^{(\ell)} \mathbf{x}_\alpha^{(\ell-1)}$

por lo que al derivar con respecto a $w_{ij}^{(\ell)}$, queda: $\mathbf{x}_i^{(\ell-1)} \cdot \boldsymbol{\delta}_j^{(\ell)}$

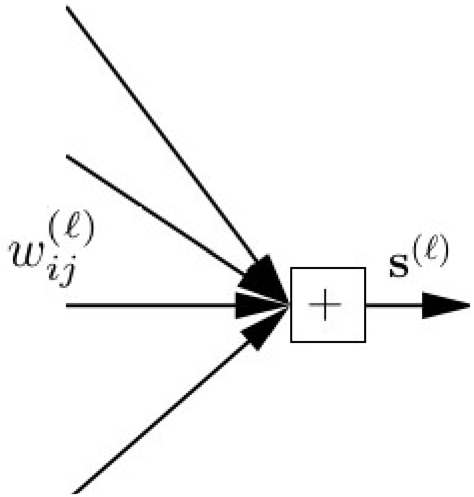
Backpropagation



Miremos esto: $$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$ a nivel de un enlace.

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

Tenemos: $$\frac{\partial \mathrm{e}}{\partial\, w_{ij}^{(\ell)}} = \frac{\partial \mathbf{s}_j^{(\ell)}}{\partial w_{ij}^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$



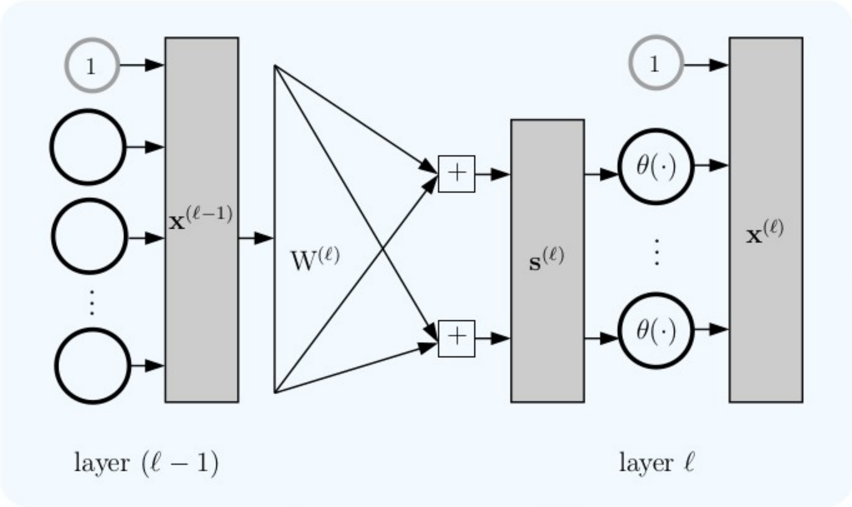
y sabemos que: $\mathbf{s}_j^{(\ell)} = \sum_{\alpha=0}^{d^{(\ell-1)}} w_{\alpha j}^{(\ell)} \mathbf{x}_\alpha^{(\ell-1)}$

por lo que al derivar con respecto a $w_{ij}^{(\ell)}$, queda: $\mathbf{x}_i^{(\ell-1)} \cdot \boldsymbol{\delta}_j^{(\ell)}$

Luego, haciendo lo mismo para cada parámetro, encontramos que: $$\frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)} \quad \checkmark$$

Backpropagation

Ahora trabajaremos con: $$\boldsymbol{\delta}_j^{(\ell)} = \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$



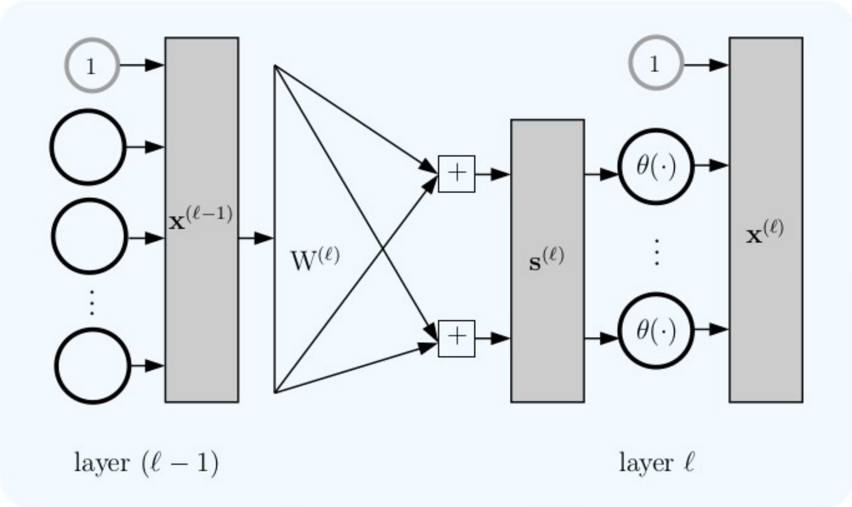
$$\frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)} \quad \checkmark$$

Backpropagation

Ahora trabajaremos con: $$\boldsymbol{\delta}_j^{(\ell)} = \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$

Aplicamos regla de la cadena:

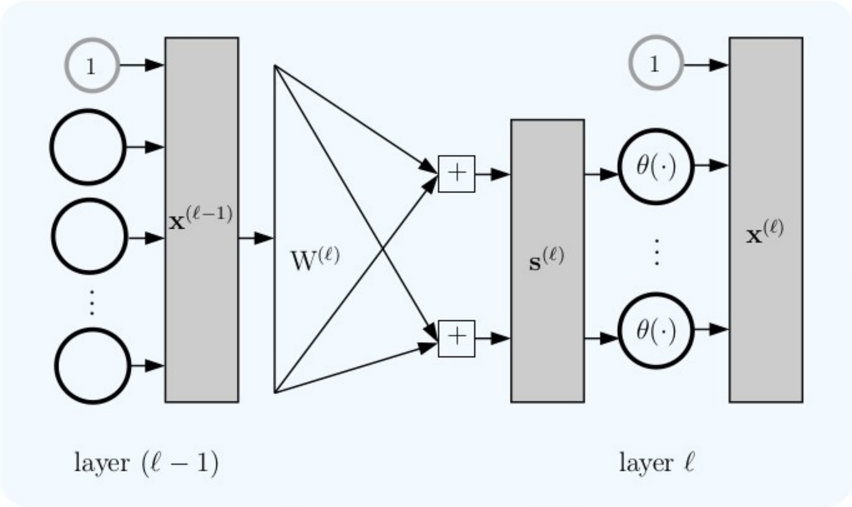
$$\frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}} = \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathbf{x}_j^{(\ell)}}{\partial \mathbf{s}_j^{(\ell)}}$$



$$\frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)} \quad \checkmark$$

Backpropagation

Ahora trabajaremos con: $$\boldsymbol{\delta}_j^{(\ell)} = \frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$$



$$\frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)} \quad \checkmark$$

Aplicamos regla de la cadena:

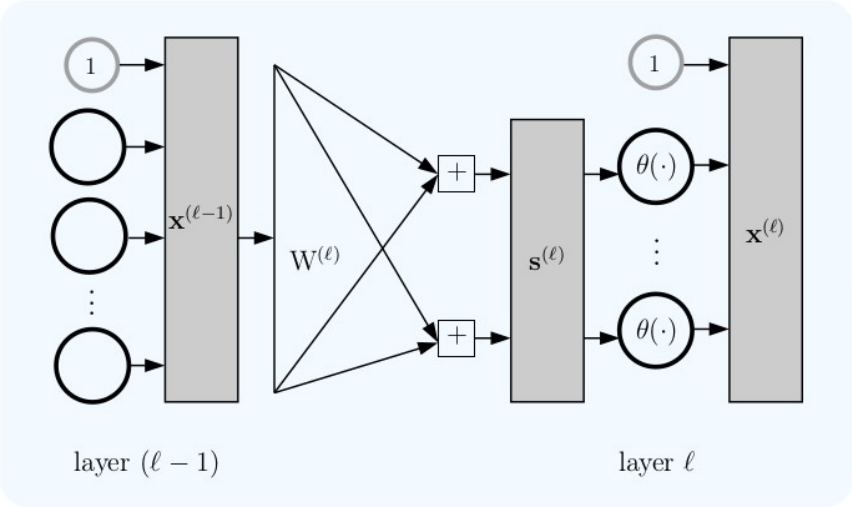
$$\frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}} = \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathbf{x}_j^{(\ell)}}{\partial \mathbf{s}_j^{(\ell)}}$$

$$= \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \theta'\left(\mathbf{s}_j^{(\ell)}\right)$$

Derivada de la función de activación

Backpropagation

Ahora trabajaremos con: $\delta_j^{(\ell)} = \dfrac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}}$



$$\frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} = \mathbf{x}^{(\ell-1)} \quad \checkmark$$

Aplicamos regla de la cadena:

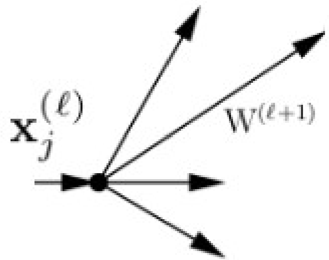
$$\frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}} = \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathbf{x}_j^{(\ell)}}{\partial \mathbf{s}_j^{(\ell)}}$$

$$= \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \theta'\left(\mathbf{s}_j^{(\ell)}\right)$$

Derivada de la función de activación

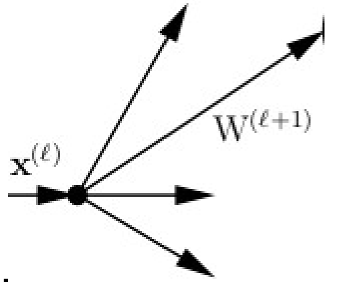
Ahora veamos que ocurre en:

$$\frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}}$$



Hay una multiplexión

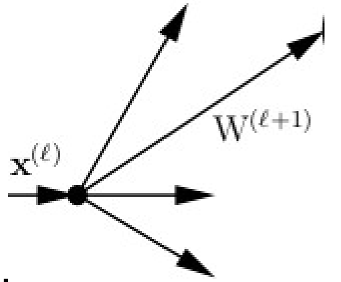
Backpropagation



Dado que una componente de $\mathbf{x}^{(\ell)}$ afecta a todas las componentes de $\mathbf{s}^{(\ell+1)}$, necesitamos sumar estas dependencias:

$$\frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} = \sum_{k=1}^{d^{(\ell+1)}} \frac{\partial \mathbf{s}_k^{(\ell+1)}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_k^{(\ell+1)}}$$

Backpropagation

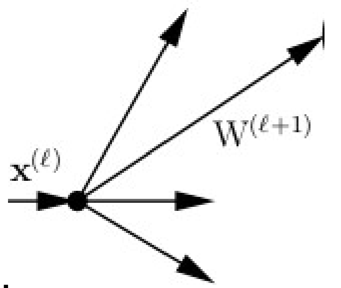


Dado que una componente de $\mathbf{x}^{(\ell)}$ afecta a todas las componentes de $\mathbf{s}^{(\ell+1)}$, necesitamos sumar estas dependencias:

$$\frac{\partial \cancel{\mathbf{x}_j^{(\ell)}} \cdot w_{jk}^{(\ell+1)}}{\cancel{\partial \mathbf{x}_j^{(\ell)}}}$$

$$\begin{aligned}\frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} &= \sum_{k=1}^{d^{(\ell+1)}} \frac{\partial \mathbf{s}_k^{(\ell+1)}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_k^{(\ell+1)}} \\ &= \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}.\end{aligned}$$

Backpropagation



Dado que una componente de $\mathbf{x}^{(\ell)}$ afecta a todas las componentes de $\mathbf{s}^{(\ell+1)}$, necesitamos sumar estas dependencias:

$$\frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} = \sum_{k=1}^{d^{(\ell+1)}} \frac{\partial \mathbf{s}_k^{(\ell+1)}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathrm{e}}{\partial \mathbf{s}_k^{(\ell+1)}}$$

$$\frac{\partial \, \cancel{\mathbf{x}_j^{(\ell)}} \cdot w_{jk}^{(\ell+1)}}{\partial \mathbf{x}_j^{(\ell)}}$$

$$= \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}.$$

Luego:

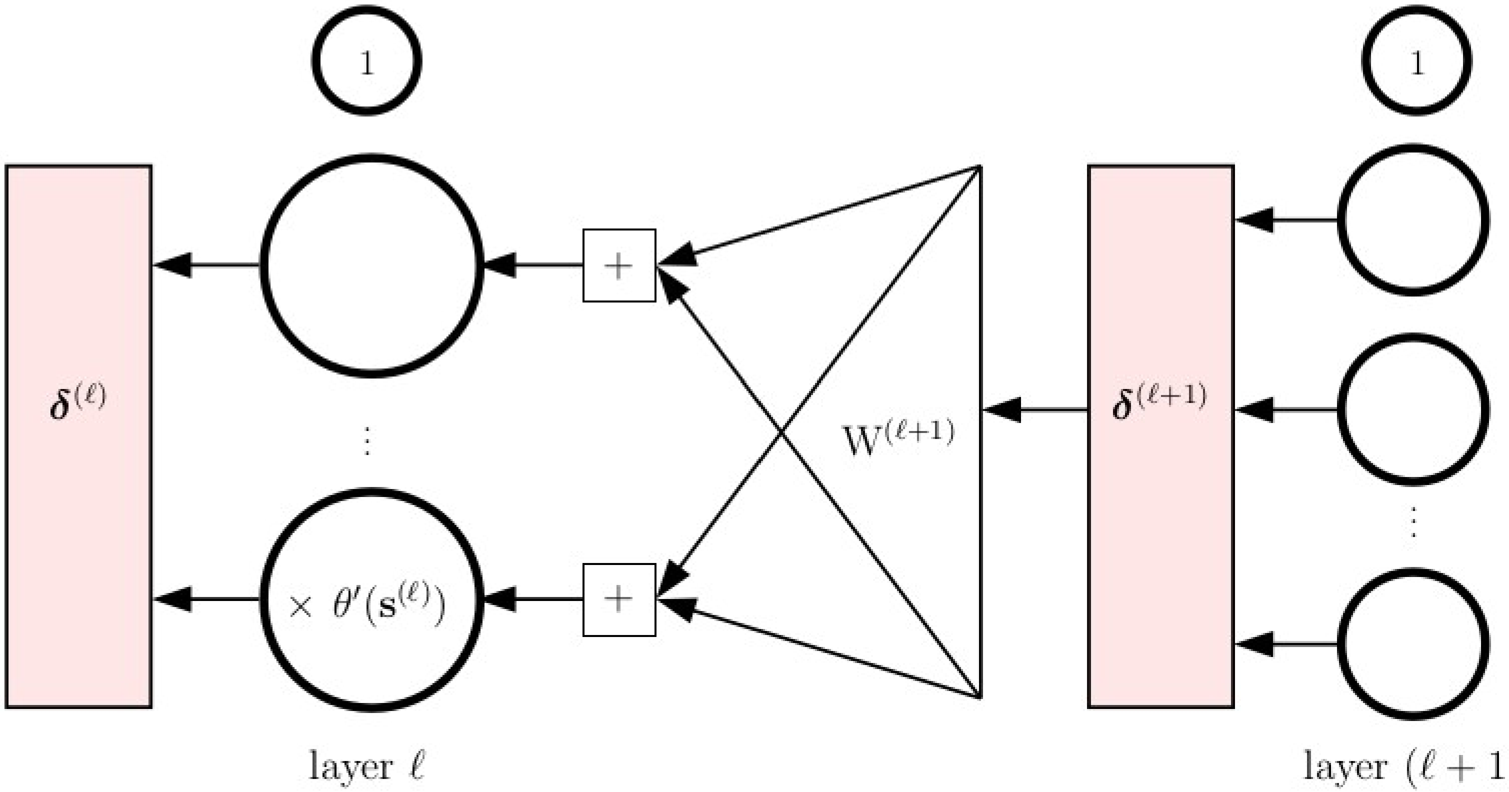
$$\frac{\partial \mathrm{e}}{\partial \mathbf{s}_j^{(\ell)}} = \frac{\partial \mathrm{e}}{\partial \mathbf{x}_j^{(\ell)}} \cdot \frac{\partial \mathbf{x}_j^{(\ell)}}{\partial \mathbf{s}_j^{(\ell)}}$$

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$

Backpropagation

$$\tanh'(\mathbf{s}^{(\ell)}) = \mathbf{1} - \tanh^2(\mathbf{s}^{(\ell)})$$

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$



$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Backpropagation

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$

Backpropagation de sensibilidad:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Nos falta calcular $\boldsymbol{\delta}^{(L)}$.

Backpropagation

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$

Backpropagation de sensibilidad:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Nos falta calcular $\boldsymbol{\delta}^{(L)}$.

Sabemos que: $\mathrm{e} = (\mathbf{x}^{(L)} - y)^2 = (\theta(\mathbf{s}^{(L)}) - y)^2$

Backpropagation

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$

Backpropagation de sensibilidad:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Nos falta calcular $\boldsymbol{\delta}^{(L)}$.

Sabemos que: $\mathrm{e} = \underline{(\mathbf{x}^{(L)} - y)^2} = (\theta(\mathbf{s}^{(L)}) - y)^2$

Luego:

$$\begin{aligned} \boldsymbol{\delta}^{(L)} &= \frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(L)}} \\ &= \frac{\partial}{\partial \mathbf{s}^{(L)}} (\mathbf{x}^{(L)} - y)^2 \end{aligned}$$

Backpropagation

$$\boldsymbol{\delta}_j^{(\ell)} = \theta'(\mathbf{s}_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \boldsymbol{\delta}_k^{(\ell+1)}$$

Backpropagation de sensibilidad:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \cdots \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Nos falta calcular $\boldsymbol{\delta}^{(L)}$.

Sabemos que: $\mathrm{e} = \underline{(\mathbf{x}^{(L)} - y)}^2 = (\theta(\underline{\mathbf{s}^{(L)})} - y)^2$

Luego:

$$\begin{aligned}\boldsymbol{\delta}^{(L)} &= \frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(L)}} \\ &= \frac{\partial}{\partial \mathbf{s}^{(L)}} (\mathbf{x}^{(L)} - y)^2 \\ &= 2(\mathbf{x}^{(L)} - y) \frac{\partial \mathbf{x}^{(L)}}{\partial \mathbf{s}^{(L)}} \\ &= 2(\mathbf{x}^{(L)} - y)\underline{\theta'(\mathbf{s}^{(L)})}.\end{aligned}$$

Esto es 0 si la red acierta

$$\tanh'(\mathbf{s}^{(\ell)}) = \mathbf{1} - \tanh^2(\mathbf{s}^{(\ell)})$$
$$= 1 - (x^{(L)})^2$$

Backpropagation

Backpropagation

$\delta^{(L)} \leftarrow 2(x^{(L)} - y) \cdot \theta'(s^{(L)})$

for $\ell = L-1$ to 1 **do**

Compute $\theta'(\mathbf{s}^{(\ell)}) = \left[1 - \mathbf{x}^{(\ell)} \otimes \mathbf{x}^{(\ell)}\right]_1^{d^{(\ell)}}$

$\boldsymbol{\delta}^{(\ell)} \leftarrow \theta'(\mathbf{s}^{(\ell)}) \otimes \left[\mathrm{W}^{(\ell+1)}\boldsymbol{\delta}^{(\ell+1)}\right]_1^{d^{(\ell)}}$

end for

Backpropagation nos permite obtener la cadena de sensibilidades:

$$\boldsymbol{\delta}^{(1)} \longleftarrow \boldsymbol{\delta}^{(2)} \;\cdots\; \longleftarrow \boldsymbol{\delta}^{(L-1)} \longleftarrow \boldsymbol{\delta}^{(L)}$$

Backpropagation

Recordar que:

$$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

Luego, podemos calcular los gradientes para aplicar GD:

Algorithm to Compute $E_{\text{in}}(\mathbf{w})$ and $\mathbf{g} = \nabla E_{\text{in}}(\mathbf{w})$:
Input: weights $\mathbf{w} = \{\mathrm{W}^{(1)}, \ldots, \mathrm{W}^{(L)}\}$; data $\mathcal{D}$.
Output: error $E_{\text{in}}(\mathbf{w})$ and gradient $\mathbf{g} = \{\mathrm{G}^{(1)}, \ldots, \mathrm{G}^{(L)}\}$.

```
Initialize: E_in = 0; for ℓ = 1, ..., L, G^(ℓ) = 0 · W^(ℓ) .
for Each data point x_n (n = 1, ..., N) do
    Compute x^(ℓ) for ℓ = 0, ..., L.   [forward propagation]
    Compute δ^(ℓ) for ℓ = 1, ..., L.   [backpropagation]
    for ℓ = 1, ..., L do
        G^(ℓ)(x_n) = [x^(ℓ-1)(δ^(ℓ))^T]
        G^(ℓ) ← G^(ℓ) + (1/N) G^(ℓ)(x_n).
    end for
end for
```

Backpropagation

Recordar que:

$$\frac{\partial \mathrm{e}}{\partial \mathrm{W}^{(\ell)}} = \frac{\partial \mathbf{s}^{(\ell)}}{\partial \mathrm{W}^{(\ell)}} \cdot \left(\frac{\partial \mathrm{e}}{\partial \mathbf{s}^{(\ell)}}\right)^{\mathrm{T}}$$

Luego, podemos calcular los gradientes para aplicar GD:

$$= \mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}$$

Algorithm to Compute $E_{\text{in}}(\mathbf{w})$ and $\mathbf{g} = \nabla E_{\text{in}}(\mathbf{w})$:

Input: weights $\mathbf{w} = \{\mathrm{W}^{(1)}, \ldots, \mathrm{W}^{(L)}\}$; data $\mathcal{D}$.

Output: error $E_{\text{in}}(\mathbf{w})$ and gradient $\mathbf{g} = \{\mathrm{G}^{(1)}, \ldots, \mathrm{G}^{(L)}\}$.

Initialize: $E_{\text{in}} = 0$; for $\ell = 1, \ldots, L$, $\mathrm{G}^{(\ell)} = 0 \cdot \mathrm{W}^{(\ell)}$.

for Each data point $\mathbf{x}_n$ $(n = 1, \ldots, N)$ **do**

  Compute $\mathbf{x}^{(\ell)}$ for $\ell = 0, \ldots, L$. [forward propagation]

  Compute $\boldsymbol{\delta}^{(\ell)}$ for $\ell = 1, \ldots, L$. [backpropagation]

  for $\ell = 1, \ldots, L$ **do**

    $\mathrm{G}^{(\ell)}(\mathbf{x}_n) = [\mathbf{x}^{(\ell-1)}(\boldsymbol{\delta}^{(\ell)})^{\mathrm{T}}]$

    $\mathrm{G}^{(\ell)} \leftarrow \mathrm{G}^{(\ell)} + \frac{1}{N}\mathrm{G}^{(\ell)}(\mathbf{x}_n)$.

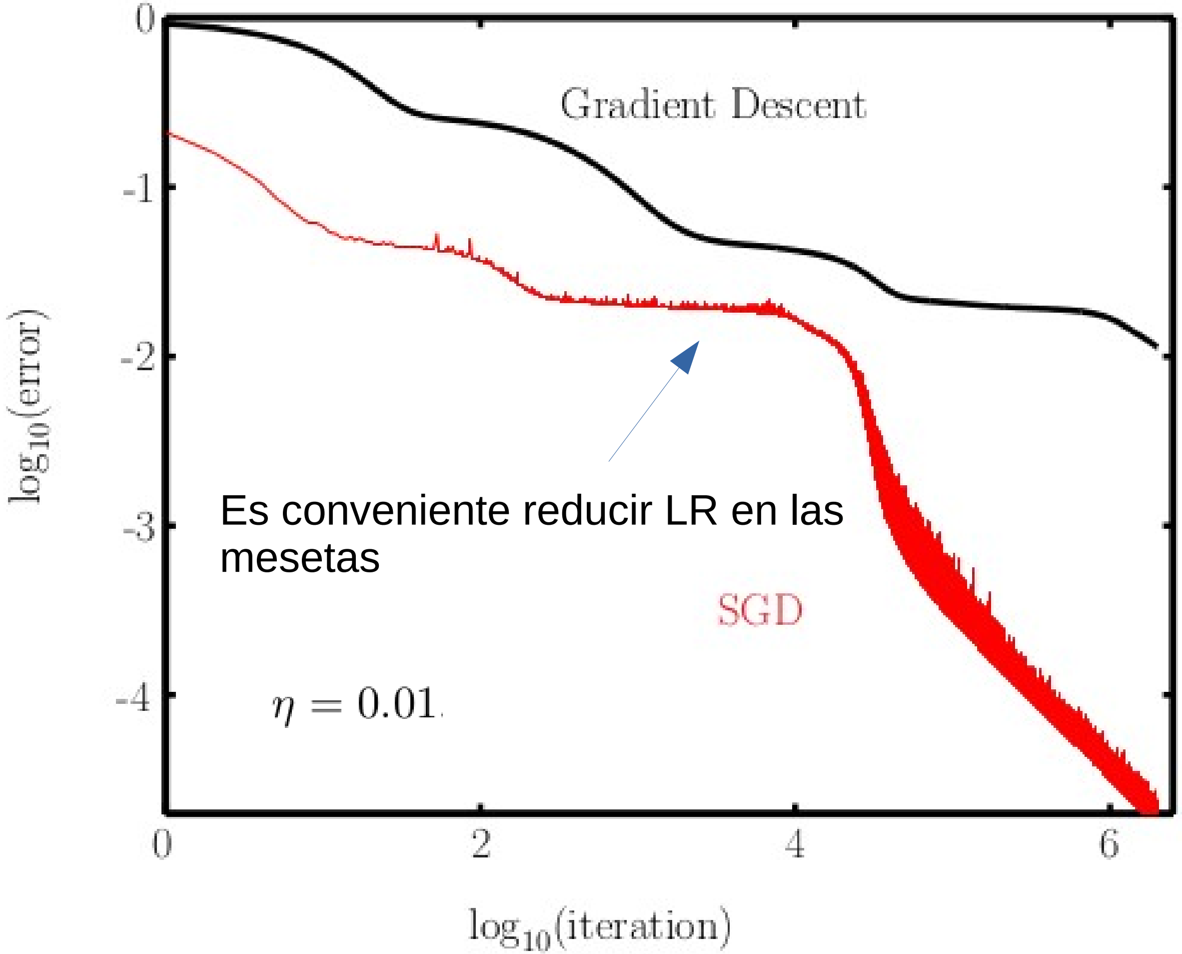
  end for

end for

$$E_{\text{in}} \leftarrow E_{\text{in}} + \tfrac{1}{N}(\mathbf{x}_n^{(L)} - y_n)^2.$$

Backpropagation

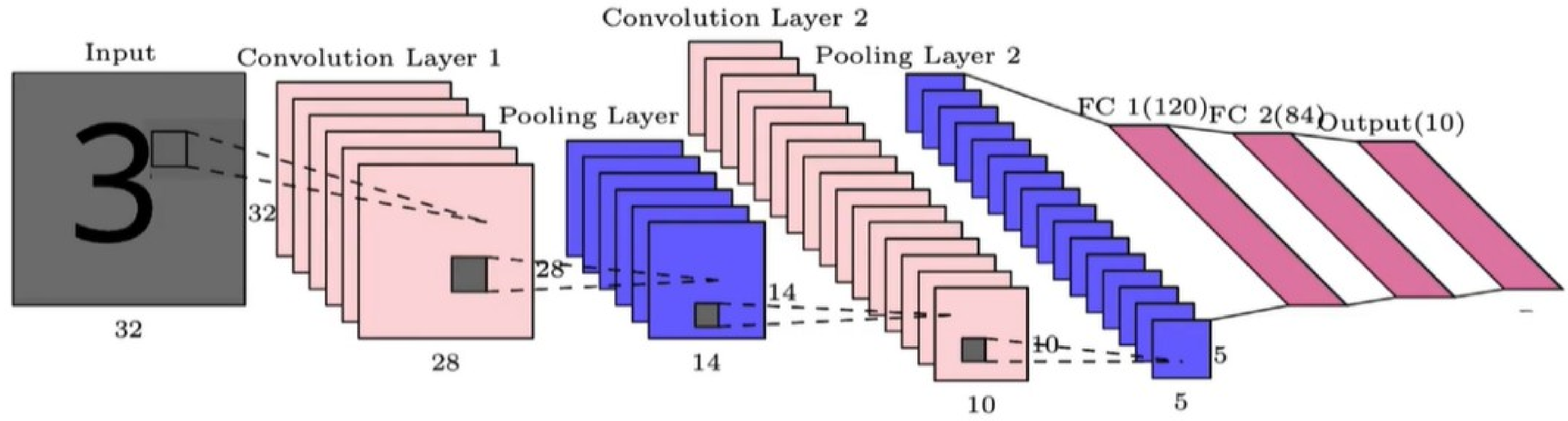
GD para redes feed-forward: $\mathrm{W}^{(\ell)} = \mathrm{W}^{(\ell)} - \eta \mathrm{G}^{(\ell)}(\mathbf{x}_n).$



Es conveniente reducir LR en las mesetas

Digits

Las redes neuronales son útiles para trabajar sobre datos raw



En este caso, la red trabaja directamente sobre la imagen. Usa dos operadores (filtro convolucional y pooling) para extraer patches desde la entrada.

Al final, usa capas densas y colapsa a una softmax.

Si la red aprende, debiera existir un gap pequeño entre accuracy de validación y training (idem para pérdida)