# Deep Learning

## **Deeper, Better, ______ , Stronger** than Machine Learning


Valentin Barriere

Universidad de Chile – DCC


CC6204, Primavera 2025

# Perceptron Multi Capas y Backpropagacion

# Perceptrón multicapa

# Outline : Perceptrón multicapa

# MLP – Presentación

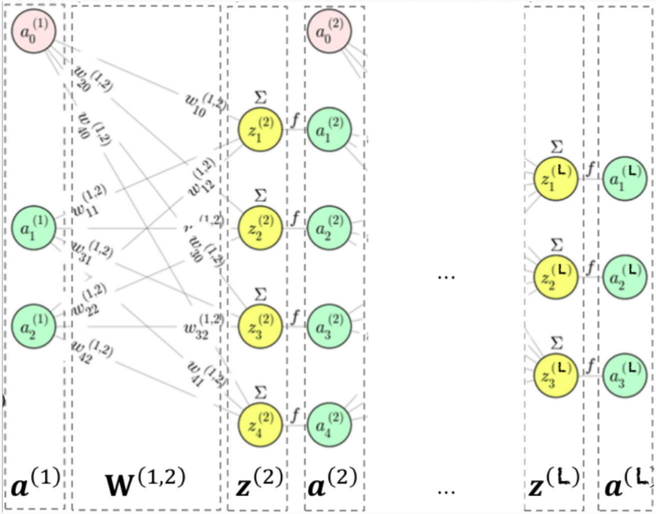



**Figure 1:** Propagación en un MLP

# MLP – Presentación

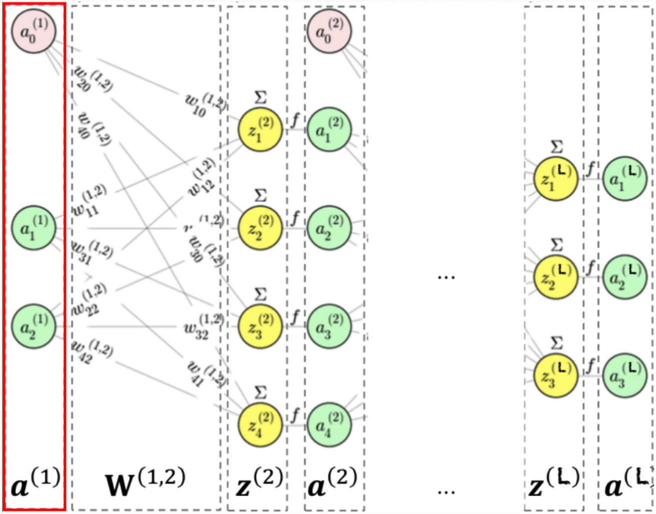



**Figure 1:** Propagación en un MLP: 1ra capa

# MLP – Presentación

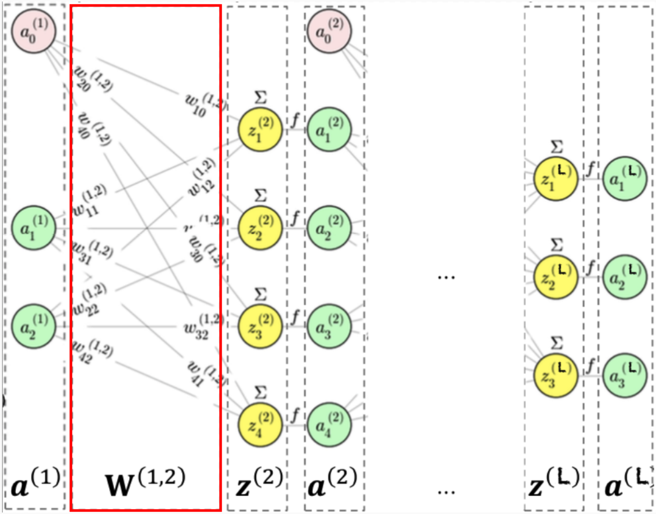



**Figure 1:** Propagación en un MLP: Matriz de paso

# MLP – Presentación

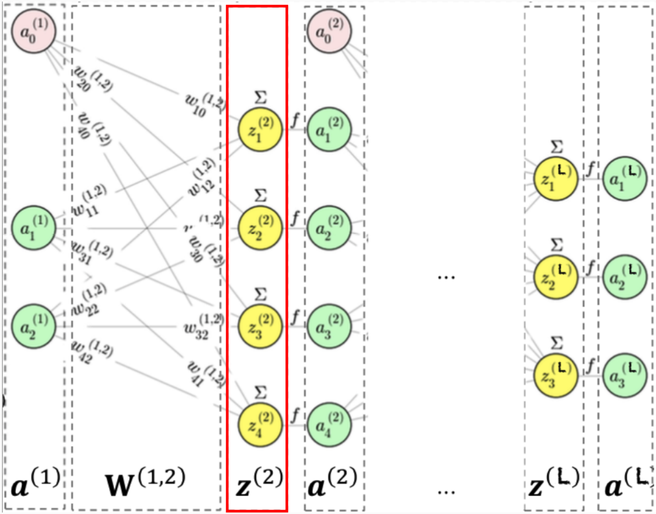



**Figure 1:** Propagación en un MLP: Entrada 2da capa

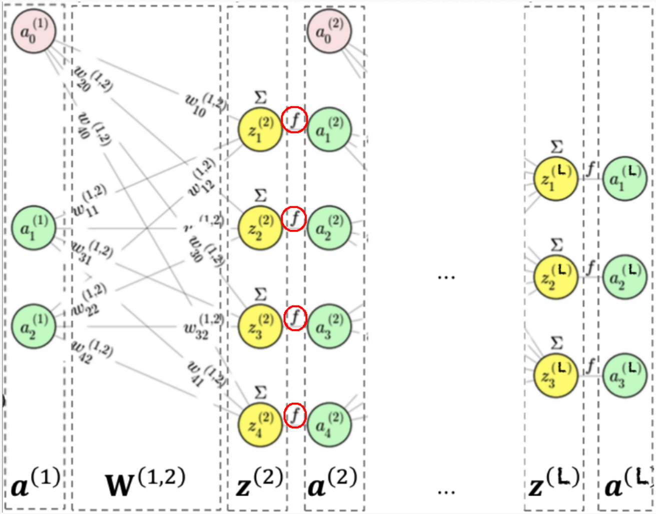



**Figure 1:** Propagación en un MLP: Activación

# MLP – Presentación

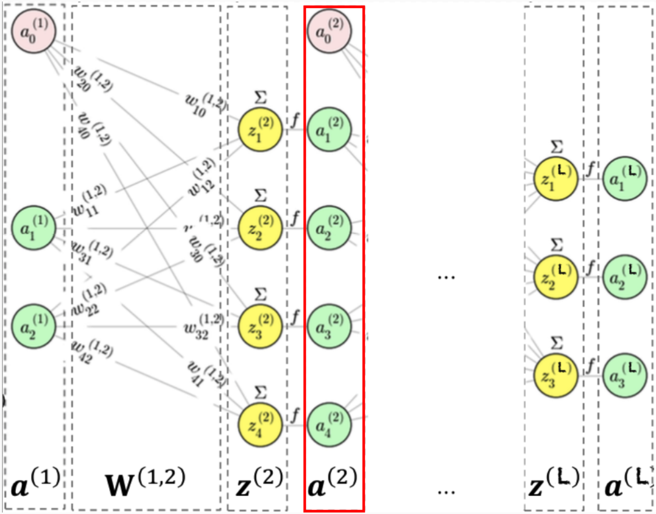



**Figure 1:** Propagación en un MLP: Salida 2da capa

# MLP – Presentación

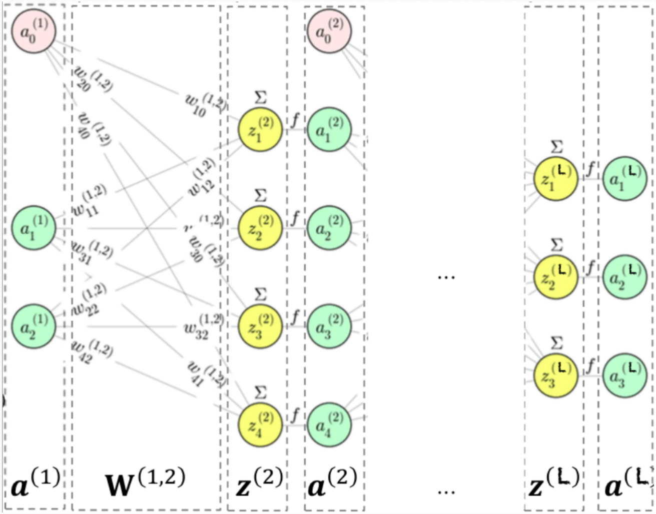
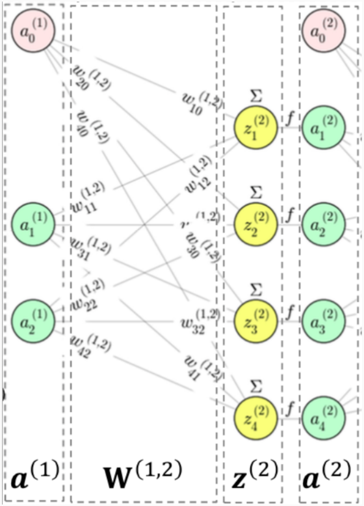



**Figure 1:** Propagación en un MLP: $\mathbf{a}^{(\ell+1)} = f(\mathbf{W}_{(\ell,\ell+1)}\mathbf{a}^{(\ell)})$

# Outline : Ecuaciones y Composicion

# MLP: Principio de una capa




$$a_1^{(2)} = f(\sum_j w_{1j}^{(1,2)} a_j^{(1)}) = f(\langle \mathbf{W}_1^{(1,2)}, \mathbf{a}^{(1)} \rangle)$$

$$a_2^{(2)} = f(\sum_j w_{2j}^{(1,2)} a_j^{(1)}) = f(\langle \mathbf{W}_2^{(1,2)}, \mathbf{a}^{(1)} \rangle)$$

$$a_3^{(2)} = f(\sum_j w_{3j}^{(1,2)} a_j^{(1)}) = f(\langle \mathbf{W}_3^{(1,2)}, \mathbf{a}^{(1)} \rangle)$$

$$a_4^{(2)} = f(\sum_j w_{4j}^{(1,2)} a_j^{(1)}) = f(\langle \mathbf{W}_4^{(1,2)}, \mathbf{a}^{(1)} \rangle)$$

Por la cappa $\ell$, con $n_\ell$ neuronas:

$$\mathbf{a}^{(\ell)} = (a_0^{(\ell)} \cdots a_{n_\ell}^{(\ell)})$$

$$\mathbf{W}^{(\ell,\ell+1)} = (\mathbf{W}_1^{(\ell,\ell+1)} \cdots \mathbf{W}_{n_k}^{(\ell,\ell+1)})$$

**Calculo matricial de una capa, similar a LogReg!**

Tenemos $\mathbf{a}^{(\ell+1)} = f(\mathbf{z}^{(\ell+1)}) = f(\mathbf{W}^{(\ell,\ell+1)} \mathbf{a}^{(\ell)})$

# MLP: Principio

Apilamos capas una sobre otra: la salida de la $\ell^{\text{ésima}}$ capa es la entrada de la $\ell+1^{\text{ésima}}$ capa. Si consideramos la función $h^{(1,2)}$ que representa el paso de la capa 1 a la capa 2.

- $\mathbf{a}^{(2)} = h^{(1,2)}(\mathbf{a}^{(1)})$
- $\mathbf{a}^{(3)} = h^{(2,3)}(\mathbf{a}^{(2)})$
- ...
- $\mathbf{a}^{(L)} = h^{(L-1,L)}(\mathbf{a}^{(L-1)})$
- $\mathbf{a}^{(L)} = h^{(L-1,L)} \circ h^{(L-2,L-1)} \circ ... \circ h^{(1,2)}(\mathbf{a}^{(1)})$
- $MLP = h^{(L-1,L)} \circ h^{(L-2,L-1)} \circ ... \circ h^{(1,2)}$ tal que $MLP(\mathbf{a}^{(1)}) = \mathbf{a}^{(L)}$

**Una función final como una composición de funciones**

El MLP puede verse como una composición de funciones no lineales. Esto permite obtener una función que pasa de los datos de entrada a la salida deseada, que es lo más compleja posible.

# Backpropagacion

# Outline : Backpropagacion

**Perceptrón multicapa**

Principio

Ecuaciones y Composicion

**Backpropagacion**

Principio

Funciones de activación

# Outline : Principio

**Perceptrón multicapa**

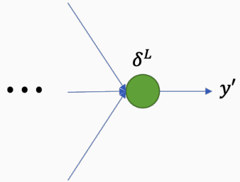
- Principio
- Ecuaciones y Composicion

**Backpropagacion**

- Principio
- Funciones de activación

# Backpropagacion: Príncipe y regla del Delta

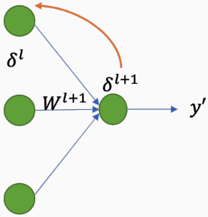
El error en la capa de salida




$$\delta^L = \frac{\partial C}{\partial y'} \sigma'(z^L)$$

$$= (y' - y) \cdot \sigma'(z^L)$$

El error de una capa en términos del error de la siguiente capa




$$\delta^l = (\delta^{l+1} (W^{l+1})^T) \odot \sigma'(z^l)$$

Error va hacia atrás a través de los pesos

**Figure 2:** El error inicial $\delta_L$ se calcula en la salida, y se propaga inversamente

# Backpropagacion: Príncipe y regla del Delta

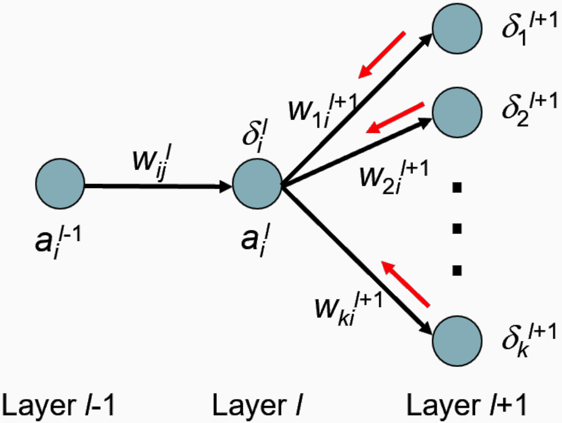
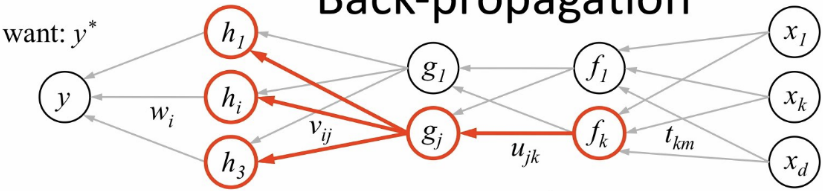



**Figure 2:** El error se propaga desde la salida hasta la entrada

$$\delta_i^l = f'(z_i^l) \sum_j w_{ij}^{l+1} \delta_j^{l+1}$$

# Back-propagation




1. receive new observation $\boldsymbol{x} = [x_1 \ldots x_d]$ and target $y^*$
2. **feed forward:** for each unit $g_j$ in each layer 1…L
   compute $g_j$ based on units $f_k$ from previous layer: $g_j = \sigma\left(u_{j0} + \sum_k u_{jk} f_k\right)$
3. get prediction $y$ and error $(y\text{-}y^*)$
4. **back-propagate error:** for each unit $g_j$ in each layer L…1

(a) compute error on $g_j$

$$\underbrace{\frac{\partial E}{\partial g_j}}_{\text{should } g_j \text{ be higher or lower?}} = \sum_i \underbrace{\sigma'(h_i) v_{ij}}_{\text{how } h_i \text{ will change as } g_j \text{ changes}} \underbrace{\frac{\partial E}{\partial h_i}}_{\text{was } h_i \text{ too high or too low?}}$$

(b) for each $u_{jk}$ that affects $g_j$

(i) compute error on $u_{jk}$

$$\frac{\partial E}{\partial u_{jk}} = \underbrace{\frac{\partial E}{\partial g_j}}_{\text{do we want } g_j \text{ to be higher/lower}} \underbrace{\sigma'(g_j) f_k}_{\text{how } g_j \text{ will change if } u_{jk} \text{ is higher/lower}}$$

(ii) update the weight

$$u_{jk} \leftarrow u_{jk} - \eta \frac{\partial E}{\partial u_{jk}}$$

Mas detalles sobre backpropagacion en [este video](), y generalidades [aca]()

# Outline : Funciones de activación

# Gradiente Evanescente

La función de activación $f$ tiene un rol importante en la propagación inversa del gradiente:

$$\delta_i^l = f'(z_i^l) \sum_j w_{ij}^{l+1} \delta_j^{l+1}$$

Si $f'(z_i^l)$ es pequeño, $\delta_i^l$ sera pequeño, y el error va a disminuir de manera exponencial.

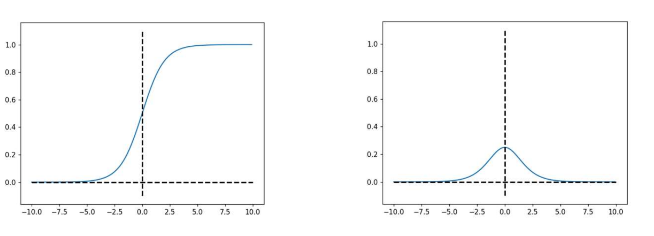
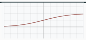
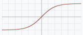
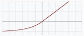



**Figure 3:** El máximo valor de la derivada ocurre en 0 con valor 0.25, asi que el error se divida per 4 en cada capa

## Función de Heaviside

- En teoría $f$ es la función de Heaviside: $f(z) = \mathbb{1}_{\mathbb{R}_+} = \begin{cases} 0 & si \quad z < 0 \\ 1 & si \quad z \geq 0 \end{cases}$
- Problema: esta función no es derivable.

En la práctica se utilizan funciones derivables como:

| Nombre | Gráfico | Ecuación | Valores en $\pm\infty$ | Valores derivada en $\pm\infty$ |
|---|---|---|---|---|
| Sigmoide |  | $f(z) = \frac{1}{1+e^{-z}}$ | $0; 1$ | $0; 0$ |
| Tangente hiperbólica |  | $f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$ | $-1; 1$ | $0; 0$ |
| ReLU[1] |  | $f(z) = \begin{cases} 0 & si \quad z < 0 \\ z & si \quad z \geq 0 \end{cases}$ | $0; z$ | $0; 1$ |
| ELU[2] |  | $f(z) = \begin{cases} \alpha(e^z - 1) & si \quad z < 0 \\ z & si \quad z \geq 0 \end{cases}$ | $-\alpha; z$ | $0; 1$ |

[1]Unidad de Rectificación Lineal

[2]Unidad Exponencial Lineal

# Questions?

# References i