Análisis Matemático para Inteligencia Artificial

Martín Errázquin (merrazquin@fi.uba.ar)

Especialización en Inteligencia Artificial

Backpropagation

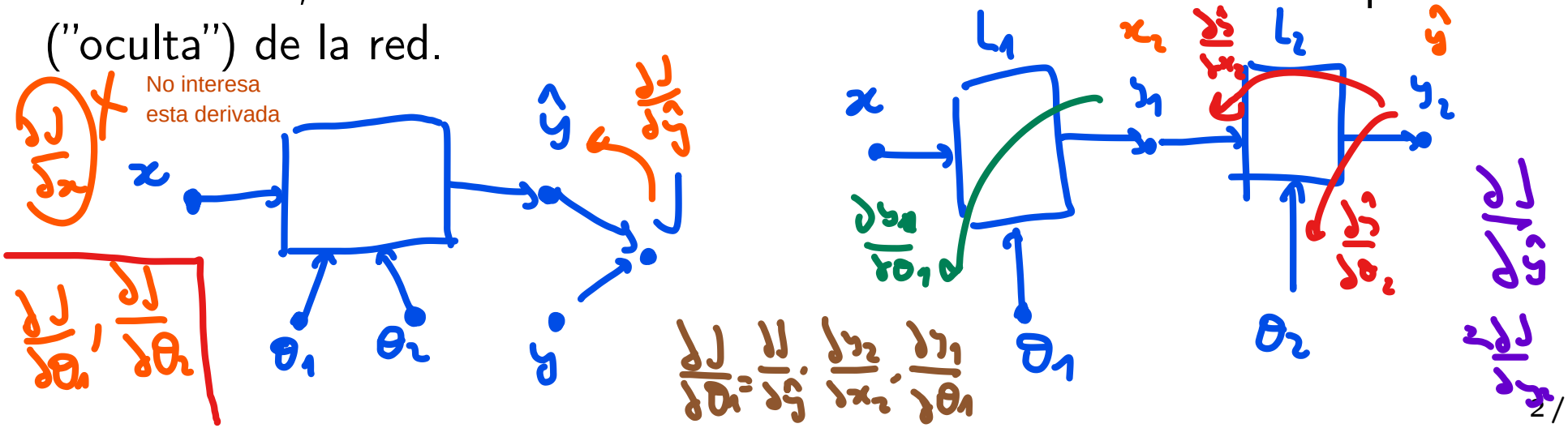
Backpropagation

¿Dónde se aplica la diferenciación automática? En Backpropagation (o simplemente Backprop), el algoritmo utilizado para entrenar redes neuronales.

Dada una entrada x y una salida esperada y, se puede calcular una función de error J. Para optimizar la red es necesario obtener la derivada de J respecto de *cada parámetro*.

¿Qué función cumple? Obtener $\frac{dJ}{d\theta}$ para cada parámetro θ de la red, a través del grafo de cómputo.

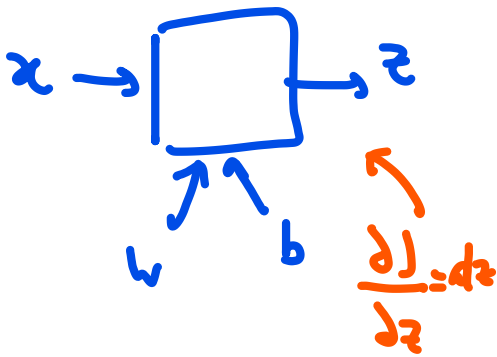
En este caso, las variables intermedias son cada salida de cada capa interna ("oculta") de la red.



Redes Neuronales: capa lineal

La base de las redes neuronales es la capa lineal, que no es otra cosa que una transformación afín $z : \mathbb{R}^n \to \mathbb{R}^m$:

$$z(W, x, b) = W \cdot x + b$$



donde $x \in \mathbb{R}^n$, $W \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$!

No es dificil llegar a que dado $\frac{dJ}{dz} = dz$, resulta:

$$\frac{dJ}{db} = dz$$

$$\frac{dJ}{dW} = dz \cdot x^T$$

$$\frac{dJ}{dx} = W^T \cdot dz$$

Redes Neuronales: función de activación

- Composición de funciones lineales es lineal $\rightarrow z_2(z_1(x)) = z_3(x) \rightarrow$ se coloca "algo no lineal" en el medio.
- Suele ser una función escalar *barata* aplicada elemento a elemento.

Dada $g : \mathbb{R} \rightarrow \mathbb{R}$ no lineal, se define el campo vectorial $y : \mathbb{R}^n \rightarrow \mathbb{R}^n$ donde $y_i = g(x_i)$ o equivalentemente $\vec{y} = (g(x_1), \ldots, g(x_n))$.

El jacobiano es una matriz diagonal, con diag. $D_y = (g'(x_i), \ldots, g'(x_n))$.

Observar que entonces dado $\frac{dJ}{dy} = dy$, resulta:

$$\frac{dJ}{dx} = \begin{pmatrix} g'(x_1) & \ldots & 0 \\ 0 & \ddots & 0 \\ 0 & \ldots & g'(x_n) \end{pmatrix} \cdot dy = D_y \odot dy$$

El caso más conocido: $y = \text{ReLU}(x) = \max(0, x) \rightarrow Dy = 1 \cdot (x > 0)$.

Redes neuronales: resumen

- FC: lineal + act
- CONV: conv + act + pool
- ATT: att. + norm + dropout

- Una red neuronal es una función compuesta *diferenciable punta a punta*. } end 2 end
- Backpropagation es usar grafo de cómputo para obtener las derivadas del error para cada parámetro.
- Cada *capa* en una red neuronal es un bloque diferenciable. Existen distintos tipos de capa.
- La capa más conocida es la *Fully Connected* (FC): capa lineal + función de activación. La arquitectura MLP es muchas FC en serie.

Es muy fácil de plantear como código!

```
z1 = W1 @ x + b1
y1 = g(z1)
z2 = W2 @ y1 + b2
y2 = g(z2)
loss = L(y2,y)
```

```
dy2 = dL(y2,y)
dz2 = dy2 * g'(z2)
dy1 = dx2 = W2.T @ dz2
...
dz1 = dx2 * g'(z1)
...
```