

# BACKPROPAGATION ALGORITHM

intuition:  $\delta_{j\mu}^{(l)}$  = "error" of node  $j$  in range  $\mu$  of layer  $l$  normal

Na última camada, terrenos:

$$\delta_i^{(L)} = \hat{a}_i^{(L)} - u_i$$

Nas. com maderas anteriores, feramos;

$$\delta^{(3)} = (\Theta^{(3)})^T \cdot \delta^{(4)} \cdot \underbrace{\alpha^{(3)} * (1 - \alpha^{(3)})}_{g'(z^{(3)})}$$
$$\delta^{(2)} = (\Theta^{(2)})^T \cdot \delta^{(3)} \cdot \underbrace{\alpha^{(2)} * (1 - \alpha^{(2)})}_{g'(z^{(2)})}$$

Matemática S<sup>1</sup> para a primeira camada, porque vamos assumir que não queremos alterar nada nos valores  $a^T = x$   
<sup>(1)</sup>

De considerarmos por en quanto  
 o termo de regularização (fazendo  
 $\lambda = 0$ ), é possível mostrar que:  $f(\theta)$   
 $= \frac{\partial}{\partial z^{(l+1)}} \text{cost}(\theta)$  not  
 use  
 about index  
 accumulators.  
 why? OK ✓

→ tentar demonstrar (mundia) ( $x^{(i)}, y^{(i)}$ )

→ tentar demonstrar com  $(x, y)$   
Para um caso com  $m$  inputs, no-  
mos começar o algoritmo definindo:

- ①  $\Delta_{ij}^{(l)} = 0$  for all  $l, i, j \in \{1, \dots, n\}$
  - Then, for  $i=1:m$
  - ② Set  $a^{(1)} = x^{(i)}$
  - ③ Perform forward propagation to compute  $a^{(l)}$ , for  $l=2, 3, \dots, L$
  - ④ We'll use  $y^{(i)}$  to compute  $\delta^{(L)} = a^{(L)} - y^{(i)}$
  - ⑤ Compute  $\delta^{(L-1)}, \delta^{(L-2)}, \dots, \delta^{(2)}$   $\rightarrow \delta^{(L+1)} \cdot (a^{(L)})^T$
  - ⑥ Accumulate:  $\Delta_{ij}^{(l)} := \Delta_{ij}^{(l)} + a_j^{(l)} \delta_i^{(l+1)}$