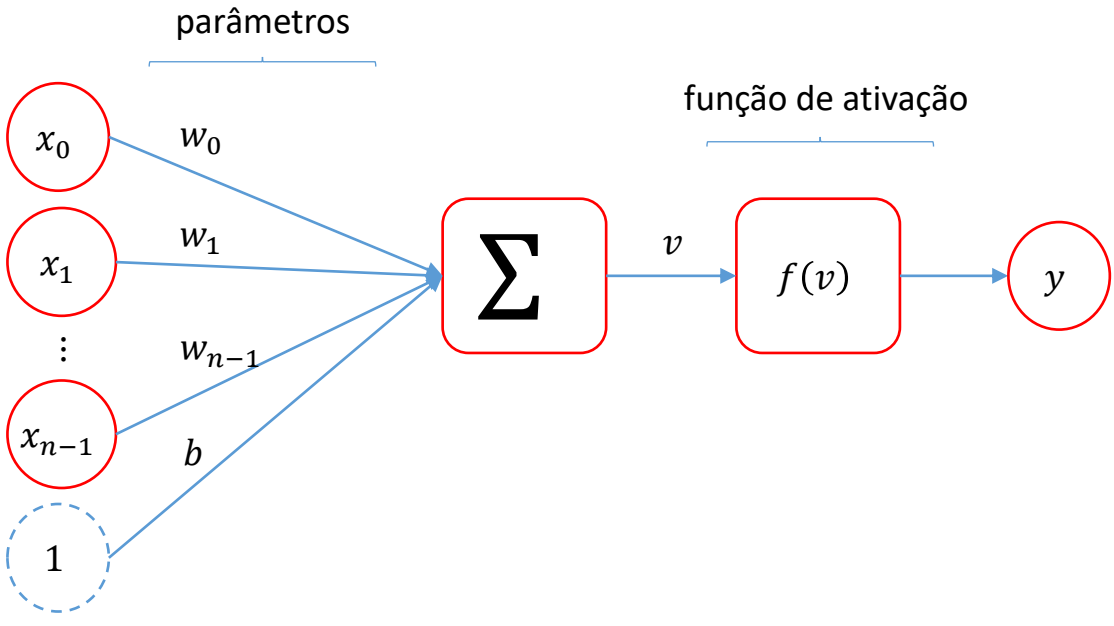
Insper

Machine Learning

Aula 21 – Treinando redes neurais

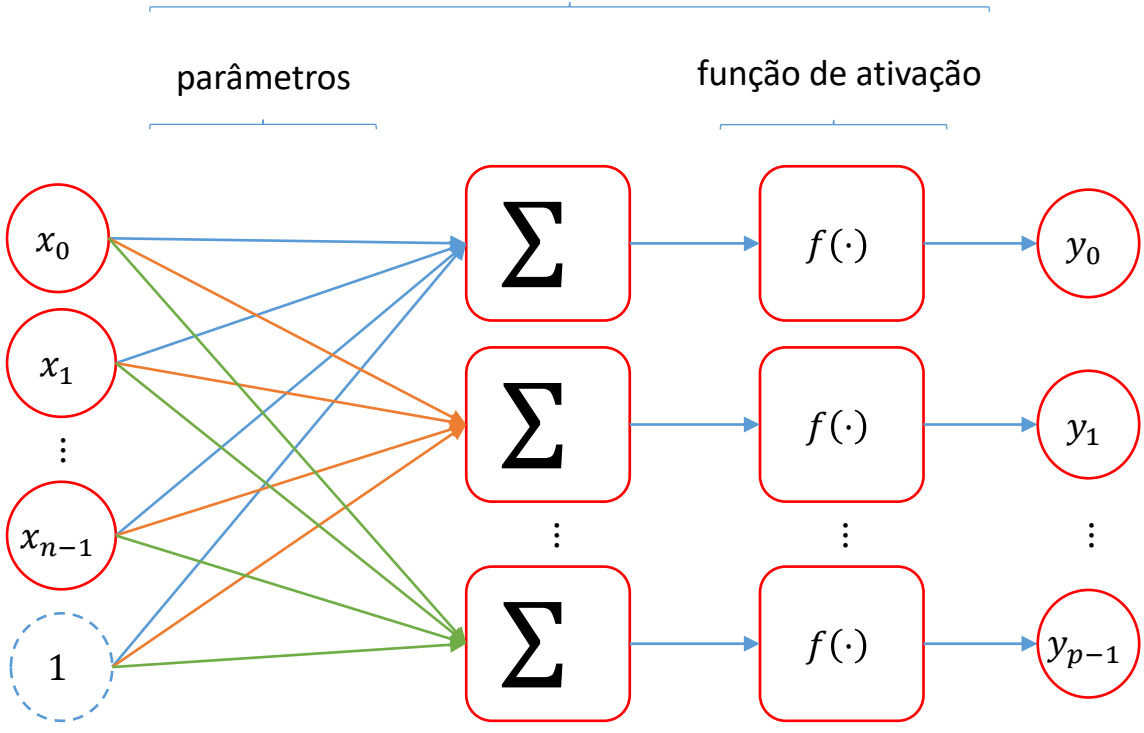
2021 – Engenharia
Fábio Ayres <fabioja@insper.edu.br>

Neurônio artificial



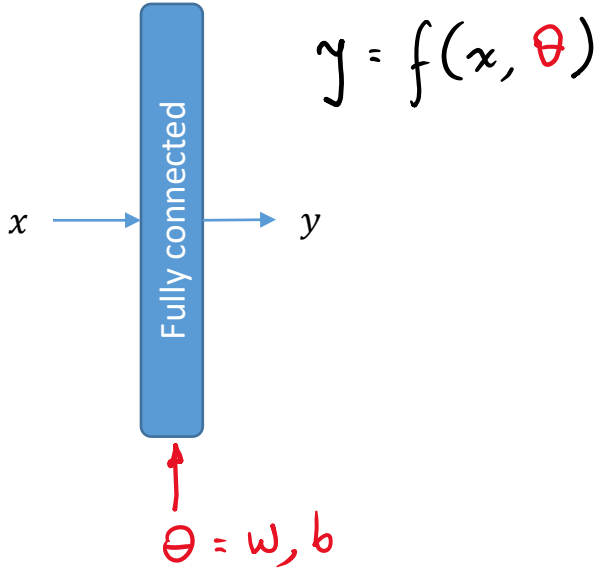
Camada (layer) de neurônios

"Fully-connected layer" → Dense

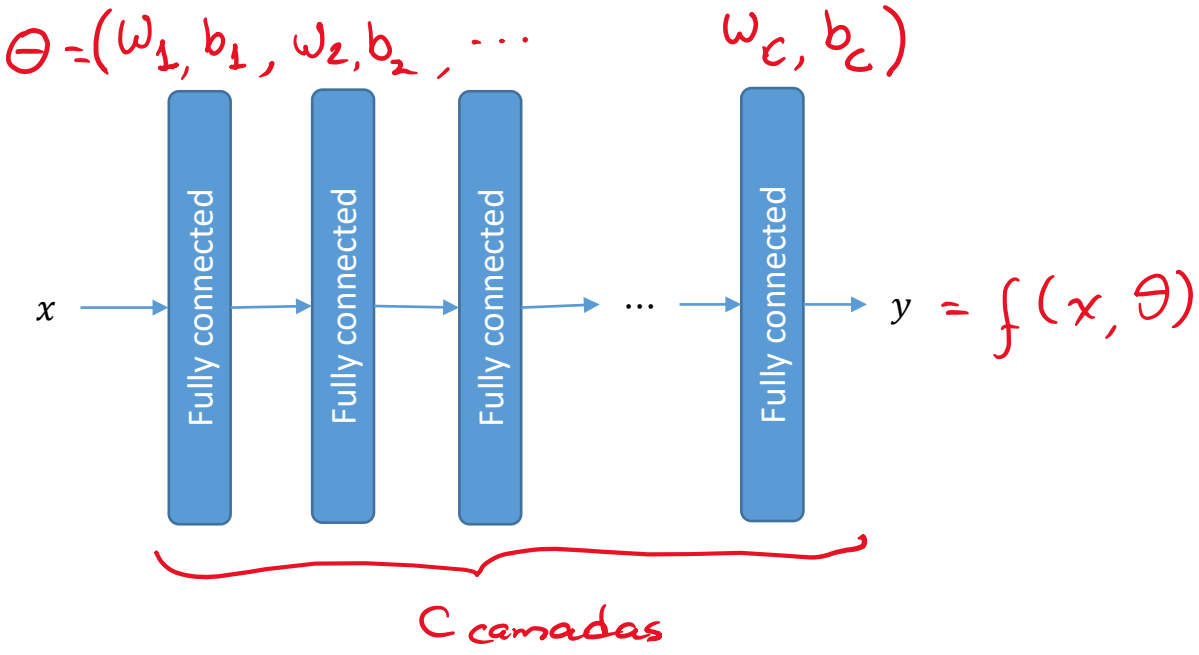


Camada de neuronios

- Geralmente representada de modo simples:



Rede neural multicamadas



Modelo: $y = f(x, \theta)$

← parâmetros treináveis

Dados: X_{train}, y_{train}

Função de perda: $L(\theta, X_{train}, y_{train}) \rightarrow L(\theta)$

fixos

← a verdadeira variável desse problema

Algoritmo de otimização

$$\theta_{opt} = \arg\min_{\theta} L(\theta)$$

Gradient descent

$$\theta^{(i+1)} = \theta^{(i)} - \eta \nabla_{\theta} L(\theta^{(i)})$$

learning rate

Problemas com as redes neurais

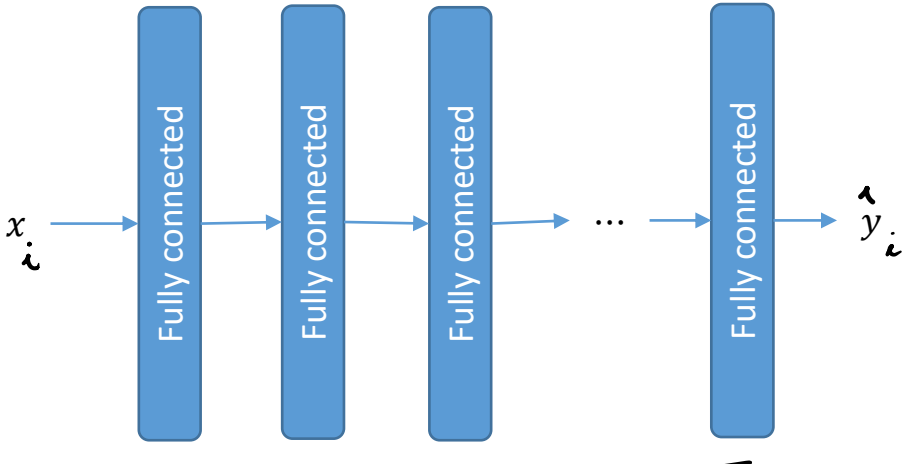
- Difícil de fazer convergir
- Overfitting
- Demora muito para treinar

Estes são os desafios que acabaram com os esforços de estudar redes neurais nos anos 90, inicio dos anos 2000.

Nos anos 2010 esses desafios foram enfrentados de novo, com sucesso!

Convergência

- Vanishing/exploding gradients



$$\hat{y} = (F_c \circ F_{c-1} \circ \dots \circ F_1)(x, \theta_1, \theta_2, \dots, \theta_c)$$

$\ell(y, \hat{y})$: perda por amostra

Exemplo: $\ell(y, \hat{y}) = (y - \hat{y})^2$ no m.s.e

$\ell(y, \hat{y}) = -y \log \hat{y} - (1-y) \log(1-\hat{y})$ na entropia cruzada

$$L(\underset{\sim}{y}, \underset{\sim}{\hat{y}}) = \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, \hat{y}_i)$$

$$\frac{\partial L}{\partial \theta_1} = \frac{1}{m} \sum_{i=1}^{m} \frac{\partial \ell}{\partial \hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial \theta}$$

$$\frac{\partial \hat{y}_i}{\partial \theta_1} = \frac{\partial}{\partial \theta_1} F_c(F_{c-1}(F_{c-2}(\cdots F_1(x_i, \theta_1) \cdots), \theta_{c-1}), \theta_c)$$

$$\frac{\partial \hat{y}_i}{\partial \theta_1} = \frac{\partial}{\partial \theta_1} F_c(F_{c-1}(F_{c-2}(\cdots F_1(x_i, \theta_1)\cdots), \theta_{c-1}), \theta_c)$$

$$\frac{\partial \hat{y}_i}{\partial \theta_1} = \frac{\partial F_c}{\partial F_{c-1}} \cdot \frac{\partial F_{c-1}}{\partial F_{c-2}} \cdot \ldots \cdot \frac{\partial F_2}{\partial F_1} \cdot \frac{\partial F_1}{\partial \theta_1}$$

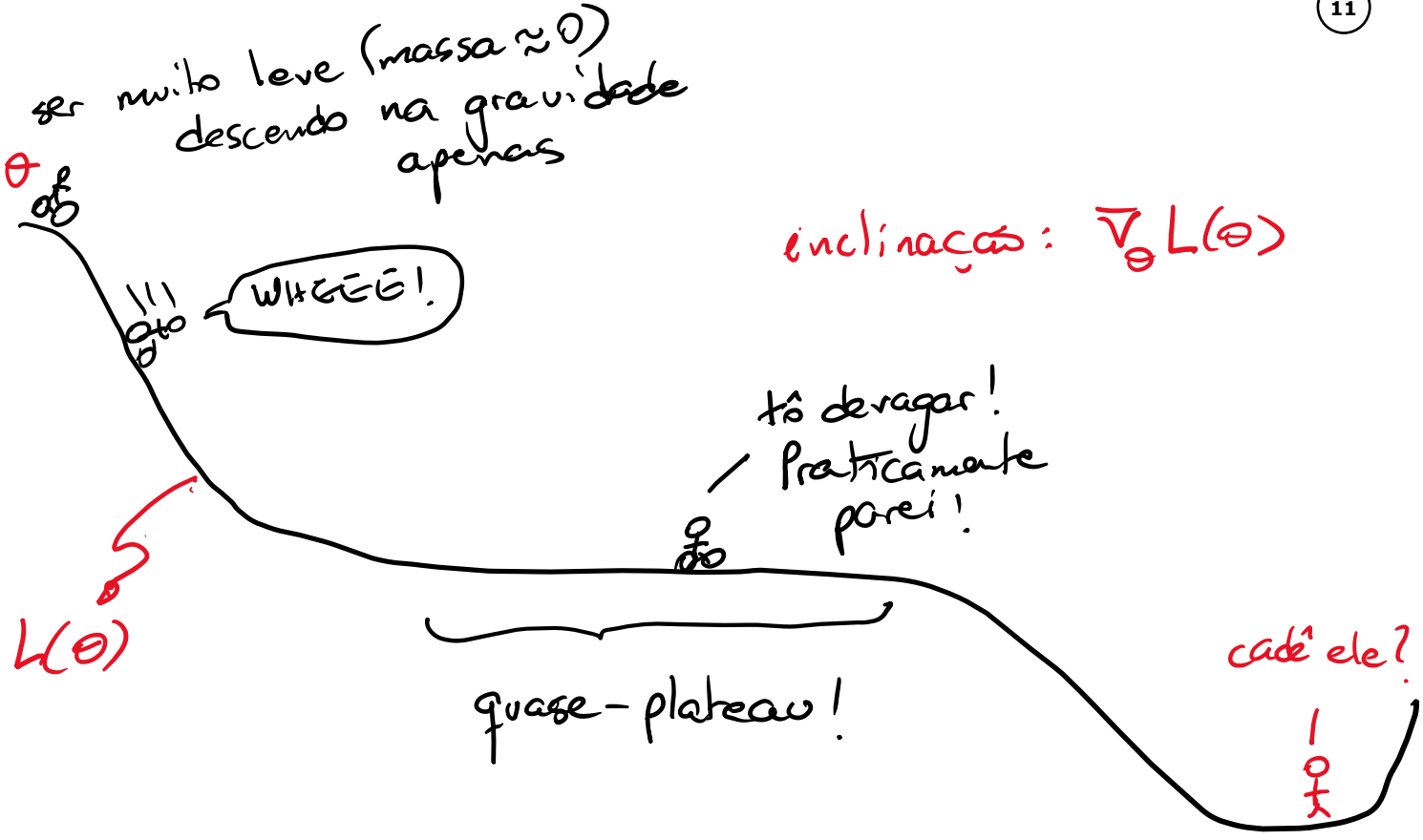
E se $\left|\frac{\partial F_k}{\partial F_{k-1}}\right| \approx 0$?

(I) Num produto, se um termo é zero, o produto é zero!

$$\theta^{(i+1)} = \theta^{(i)} - \eta \nabla_\theta L(\theta^{(i)})$$

(II) Logo, $|\nabla_\theta| \approx 0 \Rightarrow \theta^{(i+1)} \approx \theta^{(i)}$

a rede para de evoluir!


ser muito leve (massa ≈ 0)
descendo na gravidade
apenas
θ
WHEEE!
inclinação: $\nabla_\theta L(\theta)$
tô devagar!
Praticamente
parei!
$L(\theta)$
quase-platô!
cadê ele?

$$\frac{\partial \hat{y}_i}{\partial \theta_1} = \frac{\partial}{\partial \theta_1} F_c(F_{c-1}(F_{c-2}(\cdots F_1(x_i, \theta_1)\cdots), \theta_{c-1}), \theta_c)$$

$$\frac{\partial \hat{y}_i}{\partial \theta_1} = \frac{\partial F_c}{\partial F_{c-1}} \cdot \frac{\partial F_{c-1}}{\partial F_{c-2}} \cdot \ldots \cdot \frac{\partial F_2}{\partial F_1} \cdot \frac{\partial F_1}{\partial \theta_1}$$

— " —

E se $\left| \frac{\partial F_k}{\partial F_{k-1}} \right| \to +\infty$?

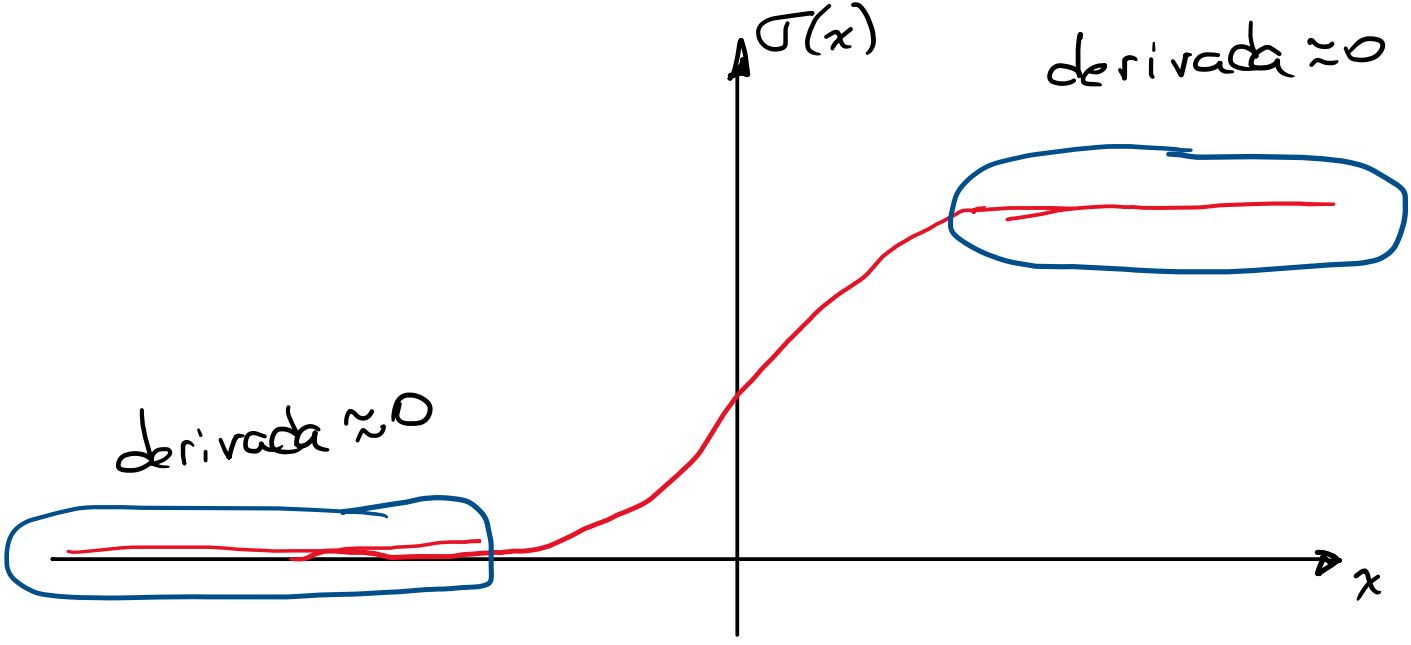
Num produto, se um termo é $+\infty$ a magnitude do produto é $+\infty$

$$\underset{\sim}{\theta}^{(i+1)} = \theta^{(i)} - \eta \nabla_\theta L(\theta^{(i)})$$

↑ passo exagerado => vede diverge!

Convergência

- Vanishing/exploding gradients



Convergência

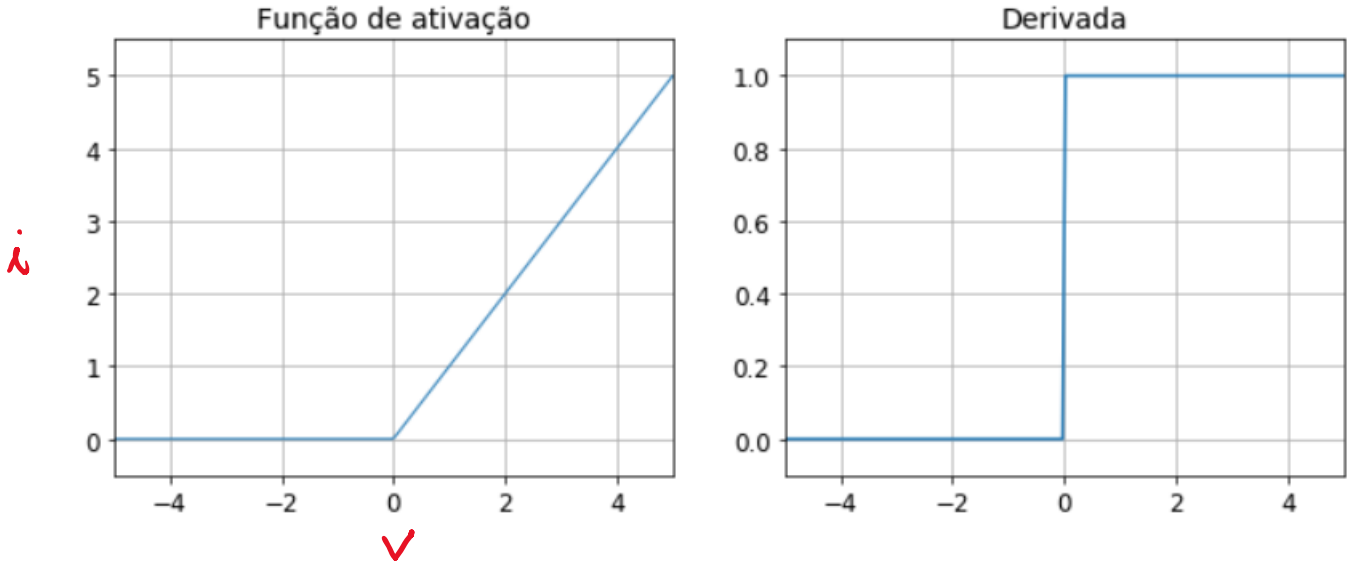
- Solução: novas funções de ativação

"Família -lu"

- relu : Rectified Linear Unit → também parecidas com SVM!
- elu : Exponential Linear Unit
- selu
- gelu
- prelu
- etc

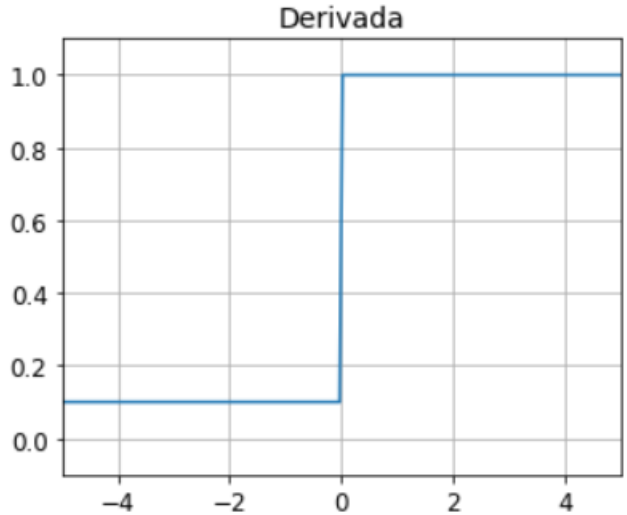
- simples de calcular
- Pelo menos em metade do domínio a deriv. não se anula!

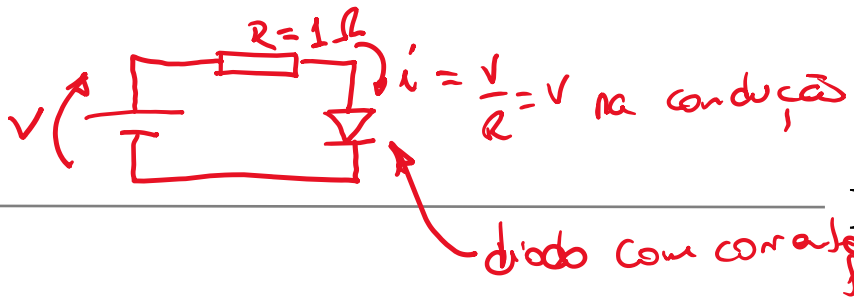
Rectified linear unit



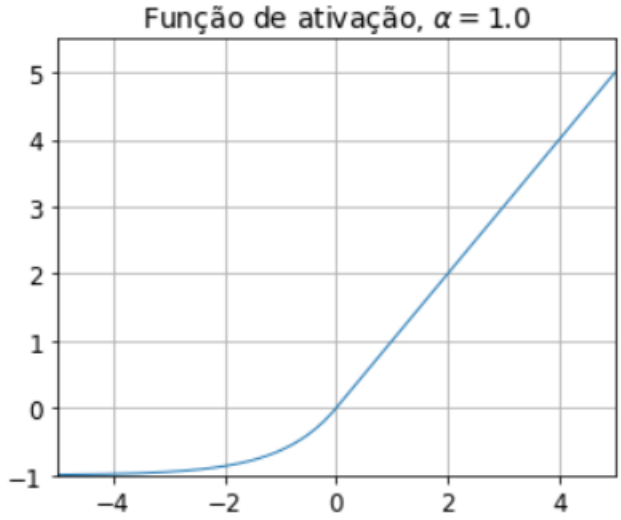
Leaky relu

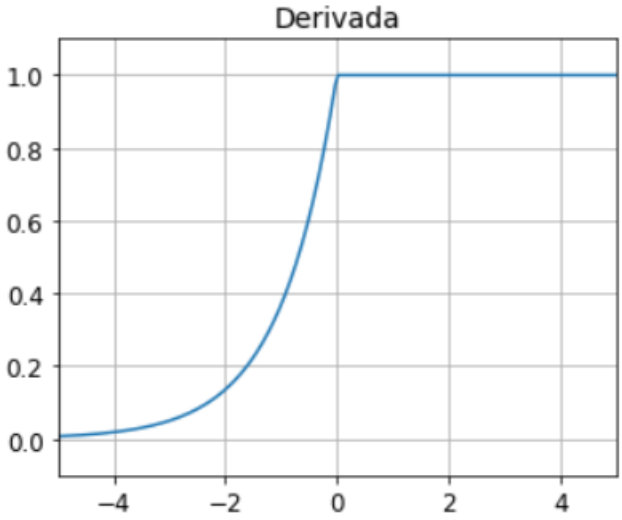

Função de ativação
5
4
3
2
1
0
−1
−4
−2
0
2
4


Derivada
1.0
0.8
0.6
0.4
0.2
0.0
−4
−2
0
2
4


R = 1 Ω
i = V/R = V na condução
V
diodo com corrente de fuga

Exponential linear unit (elu)



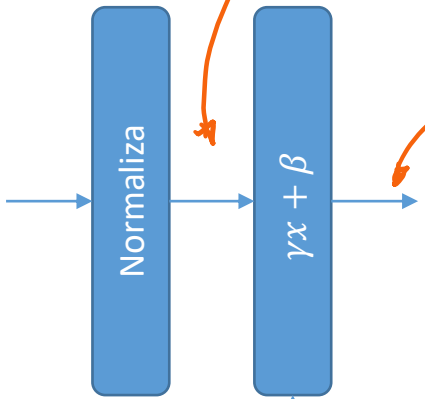


Convergência

- Solução: Batch normalization

treinamento: normaliza
teste ou uso: $\bar{x}, s$ fixos

Para cada batch:



média zero
desvio padrão 1

média β
desvio padrão γ

$$z_i = \frac{x_i - \bar{x}_B}{s_B}$$

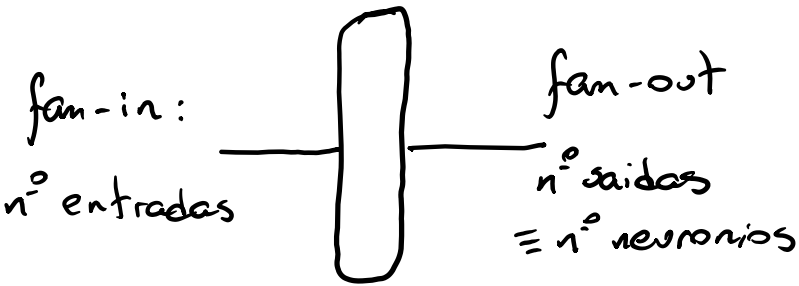
keras.layers.BatchNormalization()

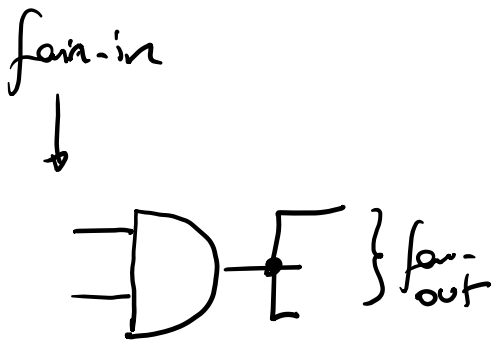
Convergência

- Solução: melhor inicialização dos pesos

→ Como definir $\Theta^{(0)}$?

Antigamente : $\Theta^{(0)} \sim N(0, 1)$





Convergência

- Solução: gradient clipping

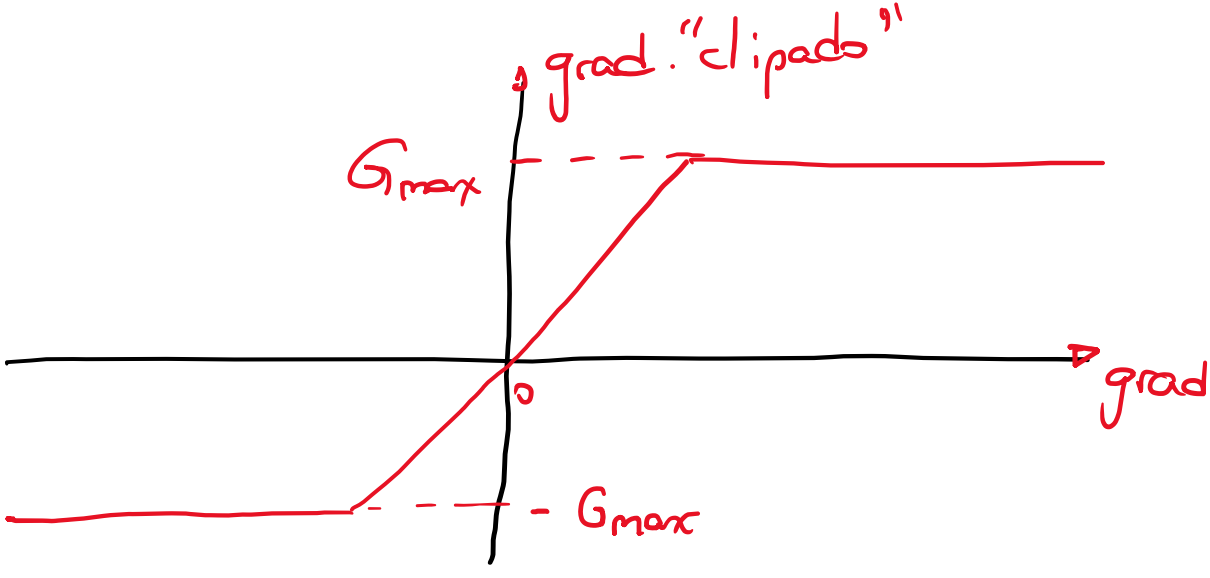
$$fan\text{-}avg = \frac{fan\text{-}in + fan_{out}}{2}$$

Glorot e Bengio

$$pesos \sim N\left(0, \frac{1}{fan\text{-}avg}\right)$$

Convergência

- Solução: gradient clipping



Overfitting

- Solução: regularização
  - Cada camada de rede neural não é uma regressão logística metida a besta? Então tasca Lasso e Ridge nelas!

$l_1 : \Sigma |w_i|$ $\quad$ $l_2 = \Sigma w_i^2$

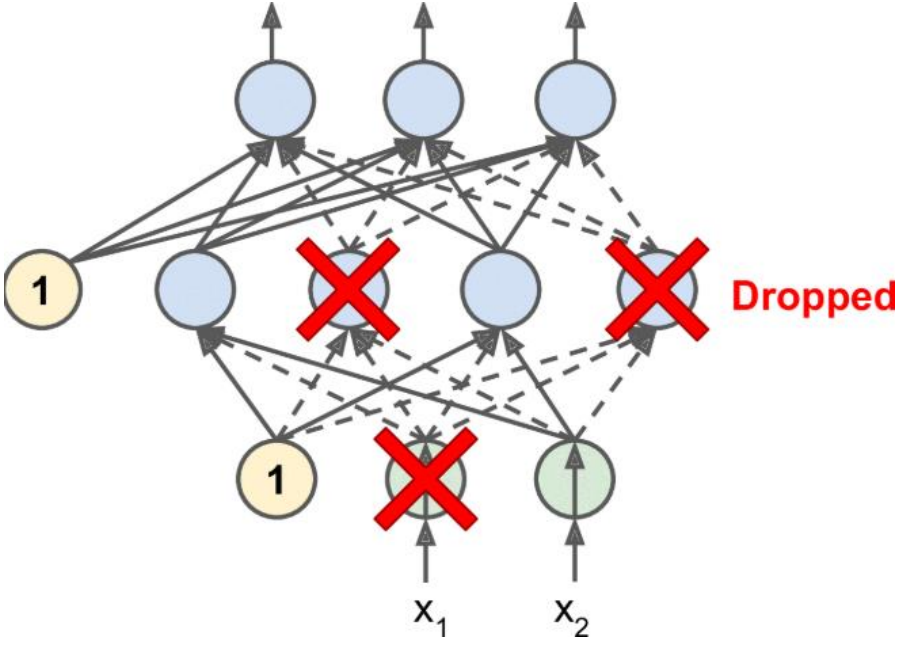
Joga como penalidade extra

Overfitting

- Solução: Early stopping

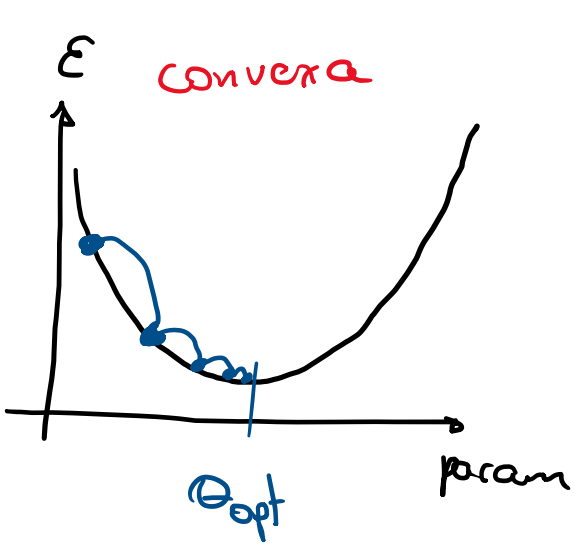
Overfitting

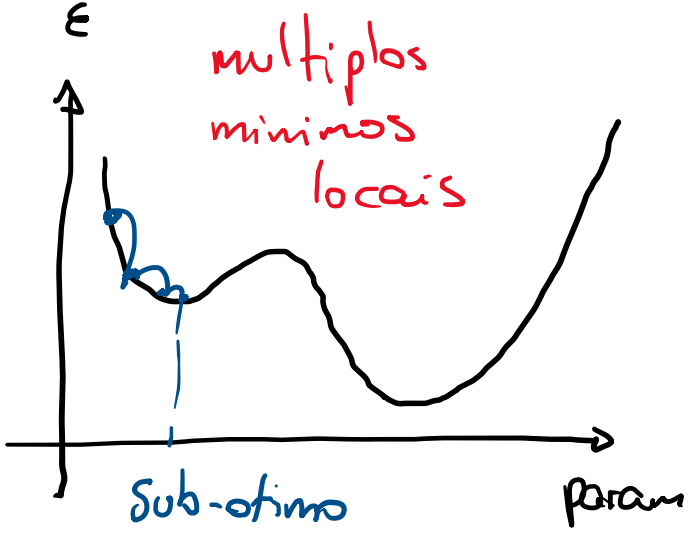
- Solução: Dropout!



Treinamento

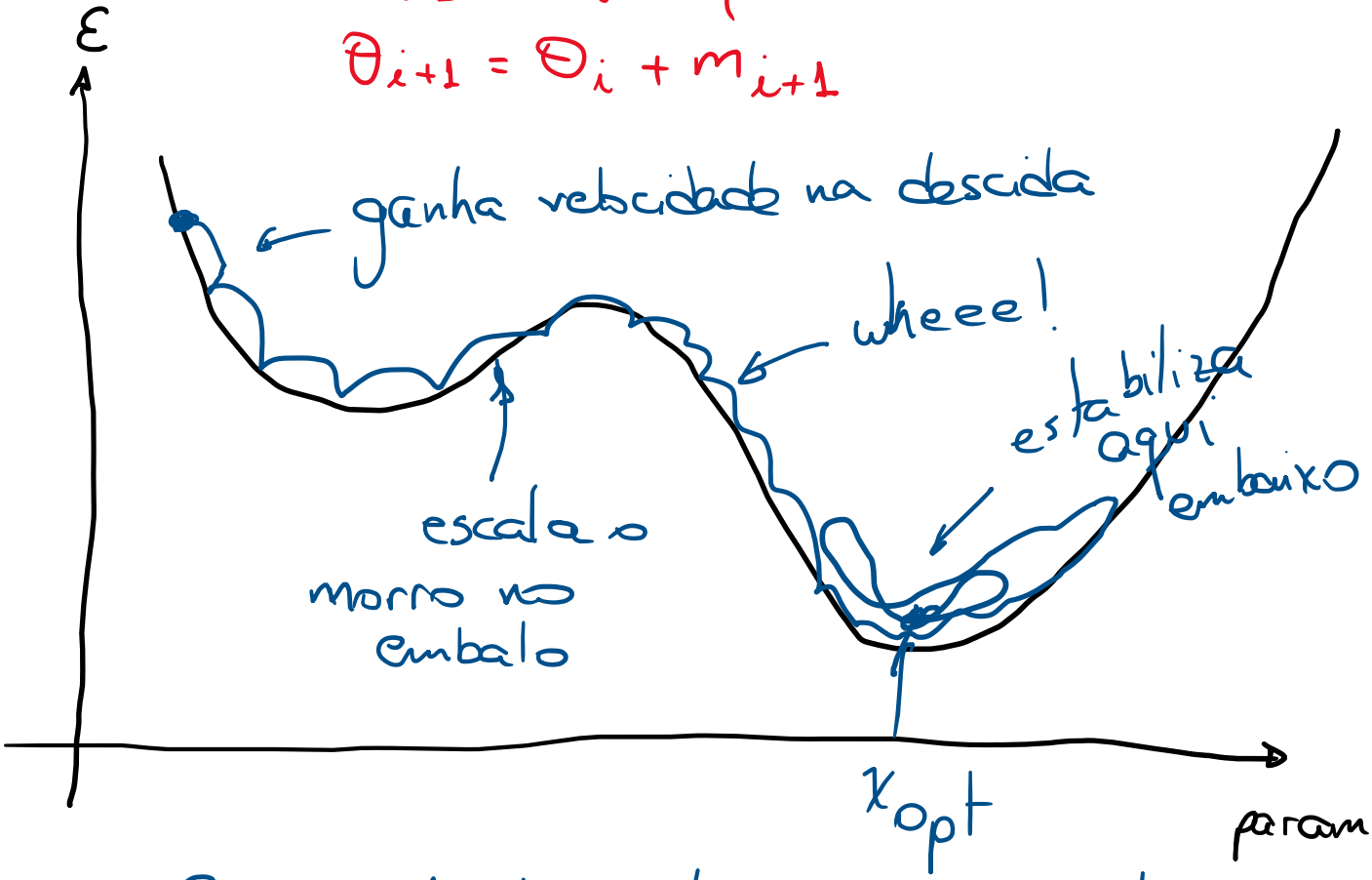
- Solução: melhores otimizadores
  - Melhora também o problema dos ótimos locais da função de erro médio





$$m_{i+1} = m_i - \eta \nabla \mathcal{E}(\theta_i)$$

$$\theta_{i+1} = \theta_i + m_{i+1}$$

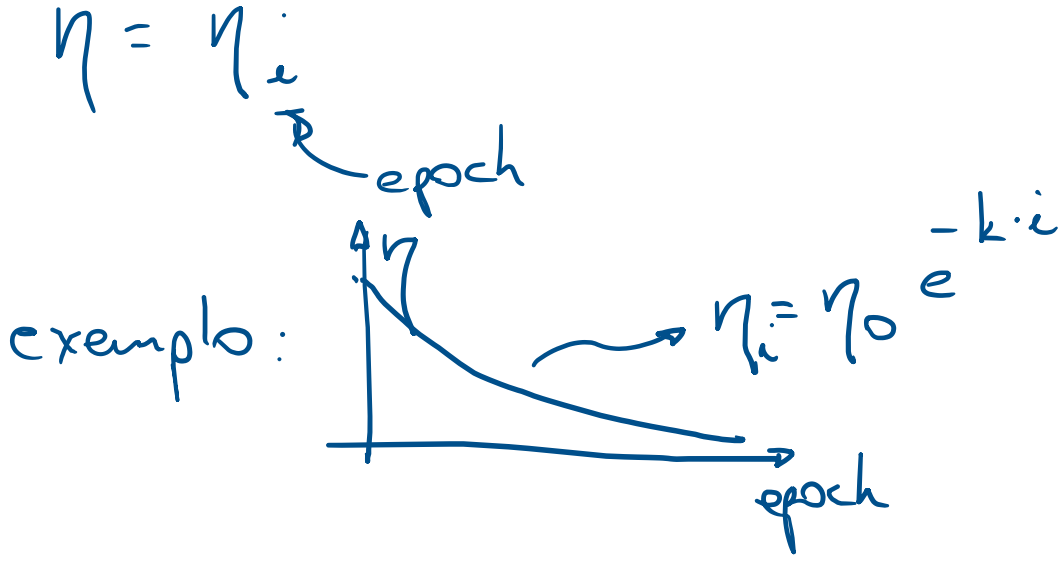


Gradient descent com momento

massa x velocidade

Treinamento

- Solução: mexer na taxa de aprendizado




ε
no início, grandes saltos
& mais pra frente,
saltos pequenos.
param

Insper