Rede Neural em Python

Marcos Vinicius Strieder

GRR20195857

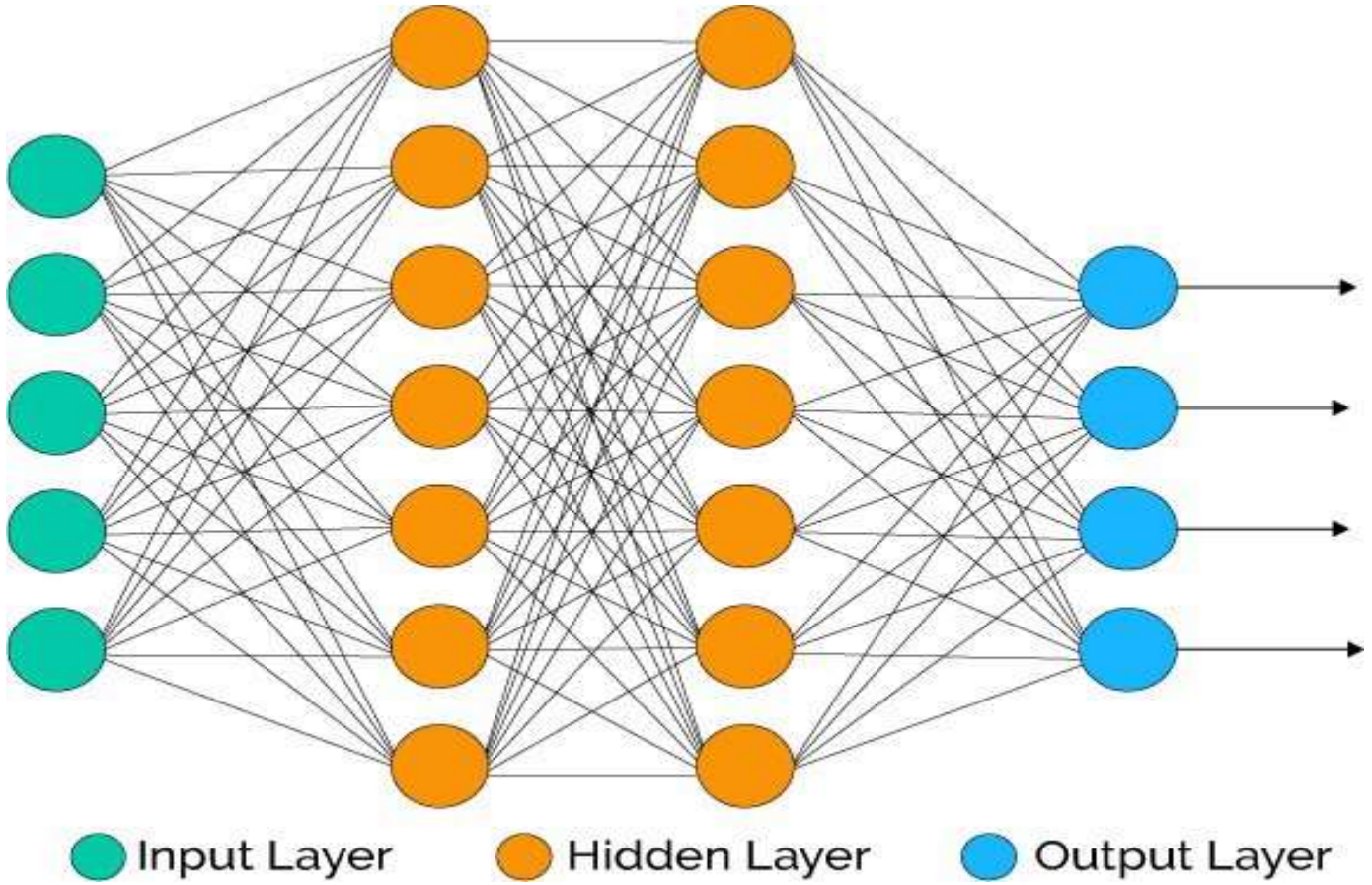
Objetivo

Criar uma rede neural artificial e treina-la para reconhecimento de dígitos escritos a mão, usando para tal dados retirados do MNIST database

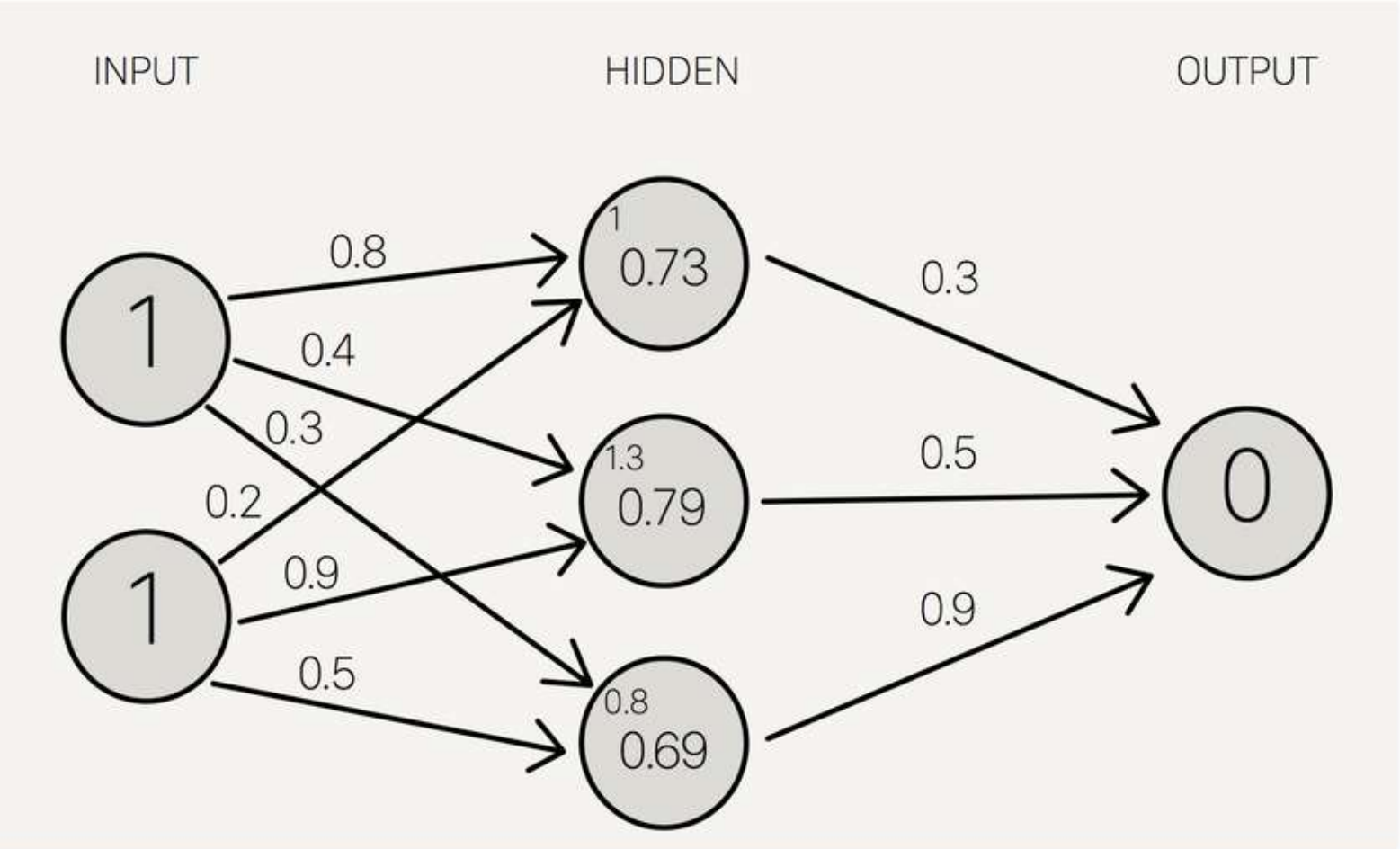
Bibliotecas utilizadas

- Pickle
- Random
- Numpy
- Matplotlib

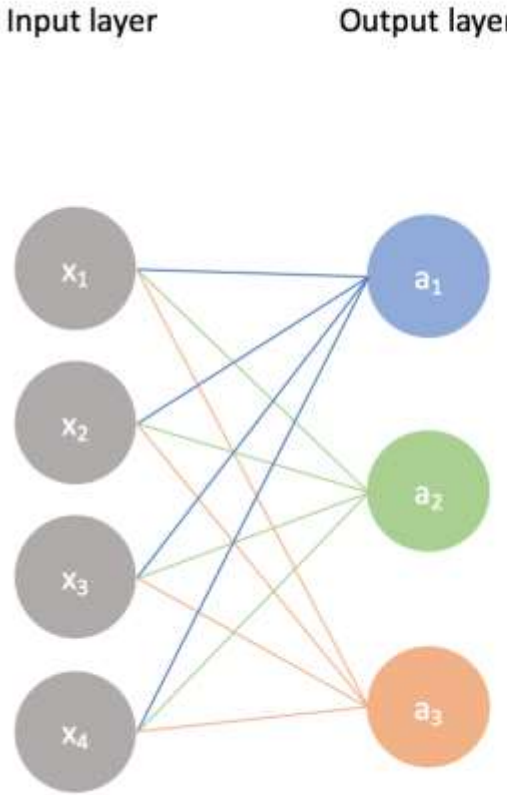
Rede Neural - Estrutura



Rede Neural - Estrutura



Rede Neural - Estrutura



A simple neural network

$$\begin{bmatrix} w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \\ w_1 & w_2 & w_3 & w_4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} + \begin{bmatrix} b \\ b \\ b \end{bmatrix} = \begin{bmatrix} w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \\ w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + b \end{bmatrix} \xrightarrow{activation} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

Função de custo

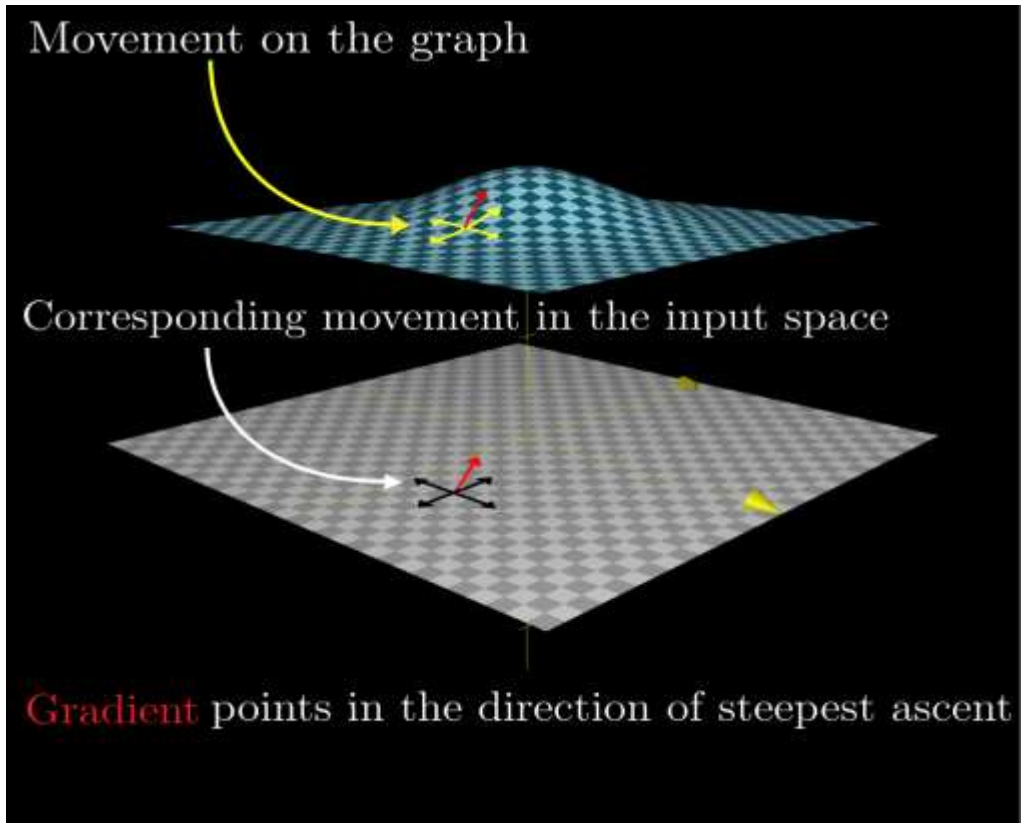
$$C(w,b) \equiv \frac{1}{2n} \sum_x \|y(x) - a\|^2. \qquad (6)$$

- **Ex:** Output = (0.1 , 0.1, 0.3, 0.9, 0.8, 0.2, 0.1, 0.3, 0.1, 0.1)

  Output ideal = (0, 1.0, 0, 0, 0, 0, 0, 0, 0, 0)

  Função de custo = ½ ((0-0.1)^2 + (1.0-0.1)^2 + (0-0.3)^2
  + ... + (0-0.1)^2)

Otimização – Descida de Gradiente estocástico



Función multivariable con valores escalares

$$\nabla f(x_0, y_0, \dots) = \begin{bmatrix} \frac{\partial f}{\partial x}(x_0, y_0, \dots) \\ \frac{\partial f}{\partial y}(x_0, y_0, \dots) \\ \vdots \end{bmatrix}$$

∇f tiene el mismo tipo de entradas que f

Notación para el vector gradiente

∇f es un vector con todas las derivadas parciales posibles de f

$$v \to v' = v - \eta \nabla C. \tag{11}$$

Referências

Michel A. Nielsen. ‘Neural Networks and Deep Learning’, Determination Press, 2015

MNIST database. Disponível em <http://yann.lecun.com/exdb/mnist/>

Biblioteca numpy. Disponível em <https://numpy.org/>