Redes Convolutivas com Python

-- *João Victor Vilela Cassiano* --
joaocassianox7x@gmail.com

Sumário da Apresentação:

1. **O que é uma Imagem?**

2. **Kernel e Convolução;**

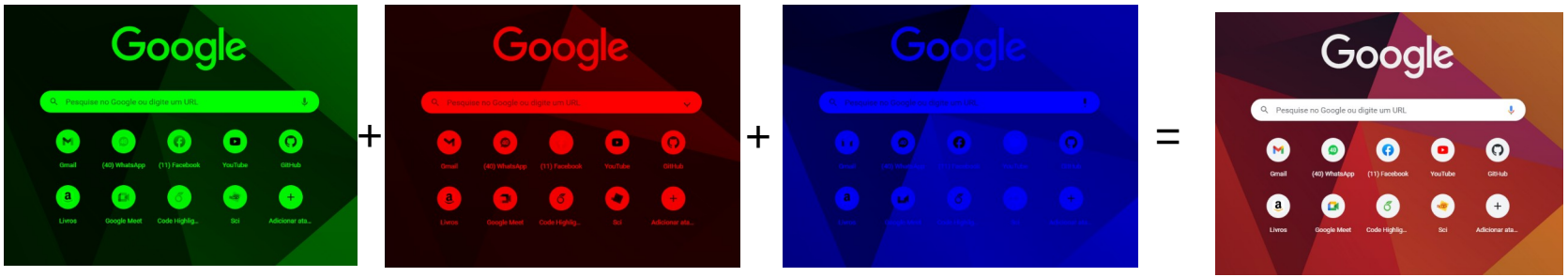
3. **CNN com o Keras;**

4. **Mãos a obra:**
   - **Gatos vs Cachorros;**
   - **CIFAR-10;**
   - **Números escritos a mão.**

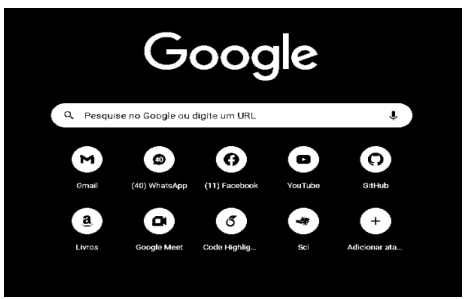
Keras

• O que é uma imagem ?

- **Uma imagem em cores do pontos de vista computacional pode ser interpretada como três matrizes, cujos elementos vão de 1 a 255 em números inteiros ou 0 a 1 em números reais, ou seja**



- **Imagens pode ser em escala cinza também, podendo ser descrita por apenas uma matriz :**



- **Em python :**

In [1]:
```
1 import numpy as np #importar o numpy para manipular imagem
2 import matplotlib.pyplot as plt #matplotlib para visualizar e carregar as imagens
```

In [2]:
```
1 imagem = plt.imread("imagens/imagem_exe.png") # carregar imagem do hd
2 print(np.shape(imagem)) #printar o tamanho da imagem
```

```
(525, 711, 4)
```

In [3]:
```
1 plt.imshow(imagem) #mostrar imagem carregada
```

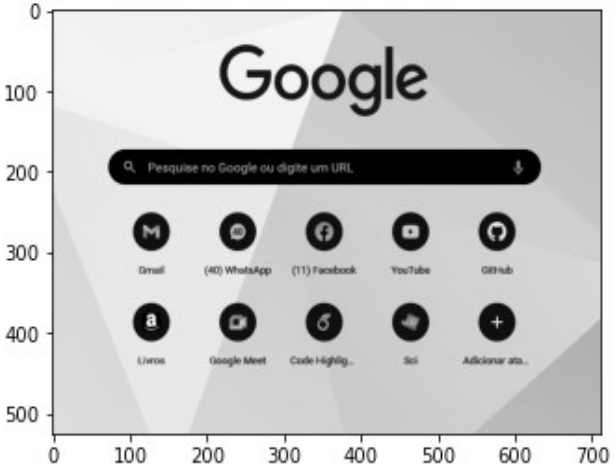
Out[3]:
```
<matplotlib.image.AxesImage at 0x1c2fd624490>
```

- # Como interpretar isso em python ?

In [4]:
```
1 imagem_cinza = imagem[:,:,0]*0.33+imagem[:,:,1]*0.33+imagem[:,:,2]*0.33 #passar para escala cinza
```

In [5]:
```
1 plt.imshow(imagem_cinza,cmap="Greys") #plotar a figura na escala cinza
```

Out[5]: <matplotlib.image.AxesImage at 0x1c2fd712610>



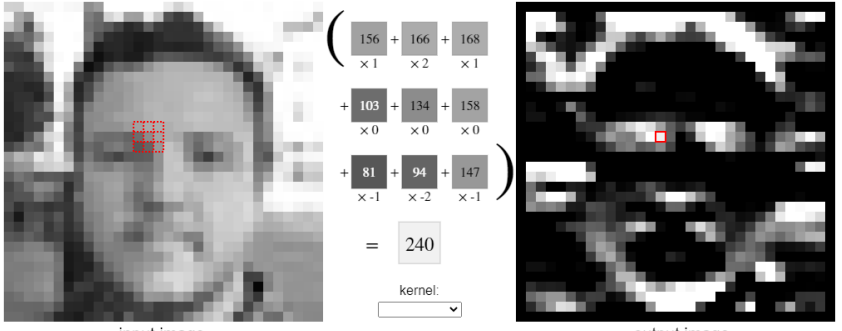
• Kernel e Convolução :

- **A expressão geral para uma convolução é:** $g(x,y) = \omega * f(x,y) = \sum_{dx=-a}^{a} \sum_{dy=-b}^{b} \omega(dx,dy) f(x+dx, y+dy)$

- **De maneira mais clara, seja um kernel 3x3 (esquerda) atuando sobre uma imagem :**

$$\left(\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \right) [2,2] = (i \cdot 1) + (h \cdot 2) + (g \cdot 3) + (f \cdot 4) + (e \cdot 5) + (d \cdot 6) + (c \cdot 7) + (b \cdot 8) + (a \cdot 9)$$

- **Caso ainda não esteja claro o bastante, vamos visitar o site setosa.io/ev/image-kernels/ que explica de maneira bastante competende e intuitiva a aplicação de um kernel sobre uma imagem :**



CNN + Keras:

- **Agora, como usar os kernels junto à convolução para extrair relações, comportamentos e padrões entre imagens? Por que não usar os modelos densos de redes neurais mais clássicos e simples de implementar ?**
  1) **Os parâmetros dos kernels se ajustarão para extrair os comportamentos, aí entra a parte de Machine Learning;**

  2) **Modelos convolutivos são expressivamente mais rápidos quando se trata de imagens (compararemos!);**

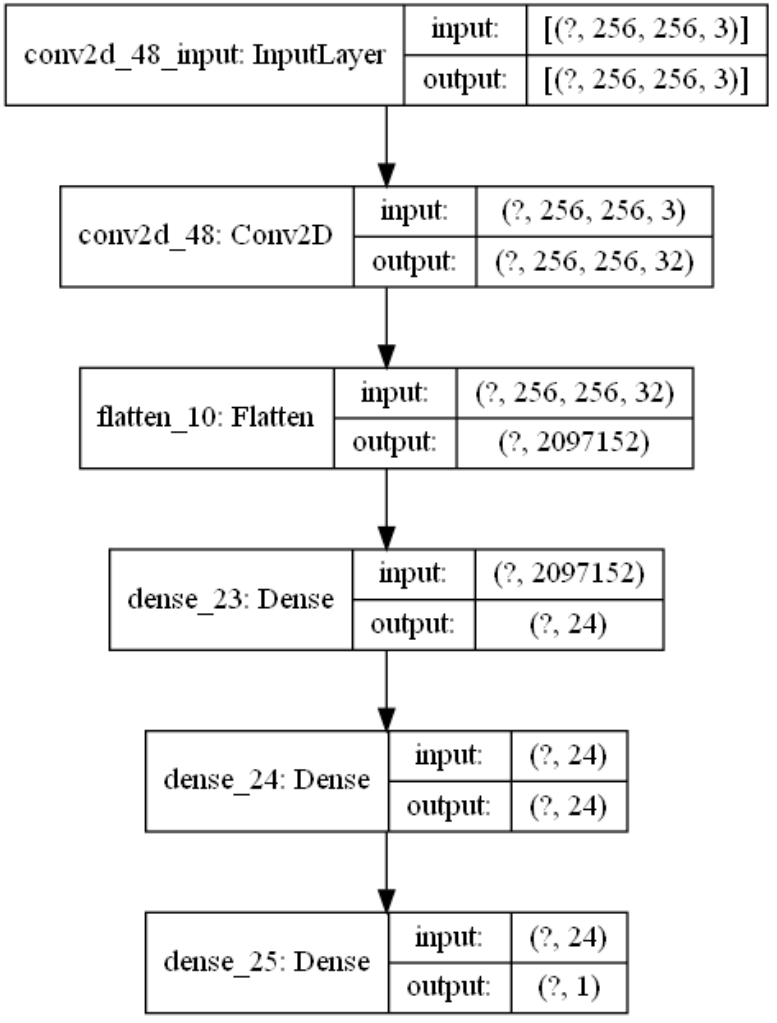
  3) **Exigem pouco processamento prévio;**

  4) **Quase sempre combinaremos modelos do tipo CNN e Densos;**

  5) **Redes convolutivas funcionam excepcionalmente bem com placas de vídeo.**

- # CNN + Keras:

```
model = keras.Sequential()

model.add(keras.layers.Conv2D(filters = 32,kernel_size=(5,5)
, activation='relu', padding="same", input_shape=(256,256,3)))

model.add(keras.layers.Flatten())

model.add(keras.layers.Dense(24,activation="relu"))

model.add(keras.layers.Dense(24,activation="relu"))

model.add(keras.layers.Dense(1,activation="sigmoid"))

```



• Mãos a obra no Jupyter Notebook !

- **Por agora, usaremos dois datasets já embutidos no tensorflow, o primerio para a classificação binária entre gatos e cachorros, chamado dogs_vs_cats;**

- **O segundo dataset também vem imbutido no tensorflow e se chama Mnist, com dezenas de milhares de imagens com os números de 0 a 9 escritos a mão.**

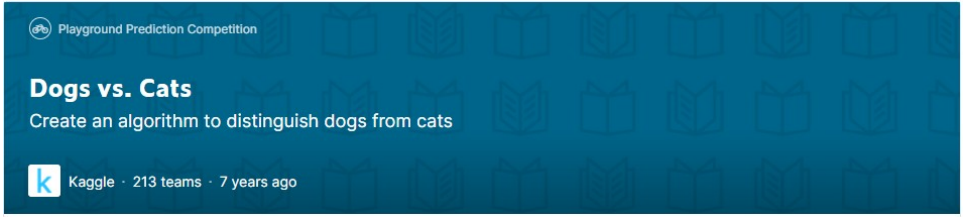
Mnist :

THE MNIST DATABASE

of handwritten digits

Yann LeCun, Courant Institute, NYU
Corinna Cortes, Google Labs, New York
Christopher J.C. Burges, Microsoft Research, Redmond

Dogs vs Cats:



- # Algumas dicas de leitura :

1)Patter Recognition and Machine Learning (Christopher Bishop);

2) The Elements of Statistical Learning (Jerome H. Friedman);

3)Deep Learning (Ian Goodfellow) ;

4) Deep Learning with Python (François Chollet)

5)Neural networks and deep learning (Michael Nielsen)