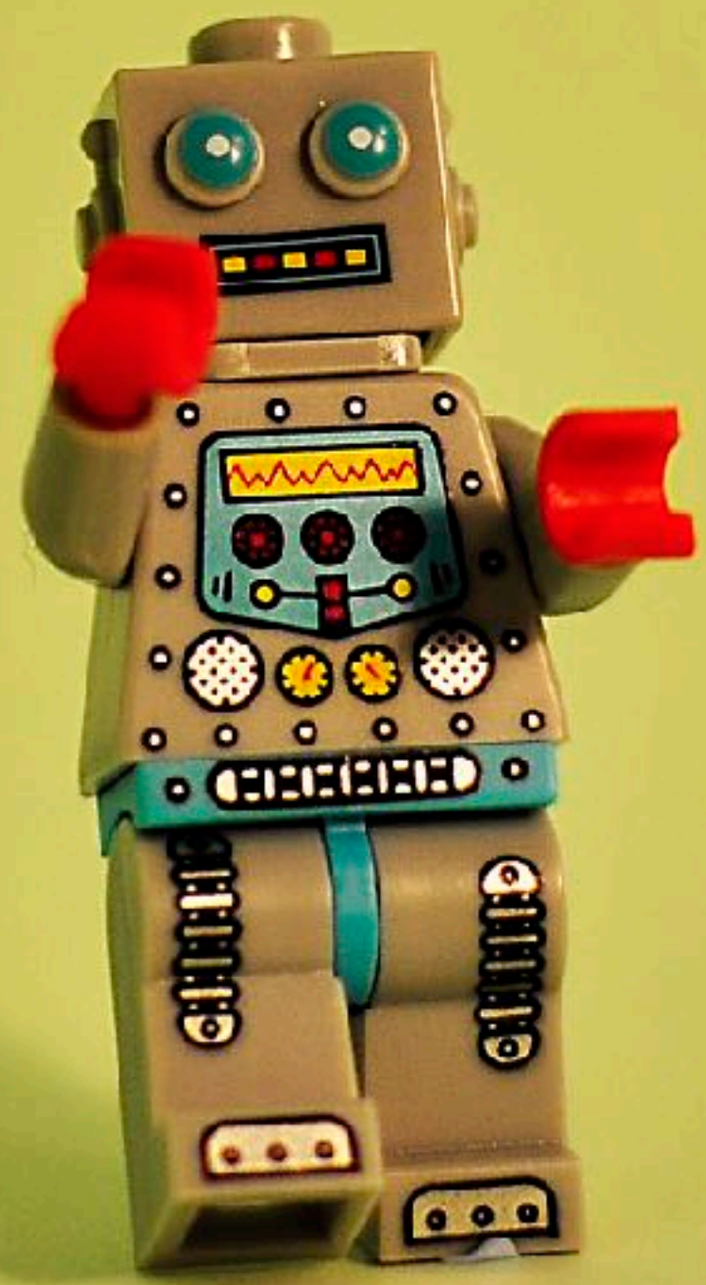


Vision 101

(from ML to DL)

OBJETIVOS:

- INTRO
- TÉCNICAS
- FERRAMENTAS
- APLICAÇÕES
- PRÓXIMOS PASSOS

Disclaimers

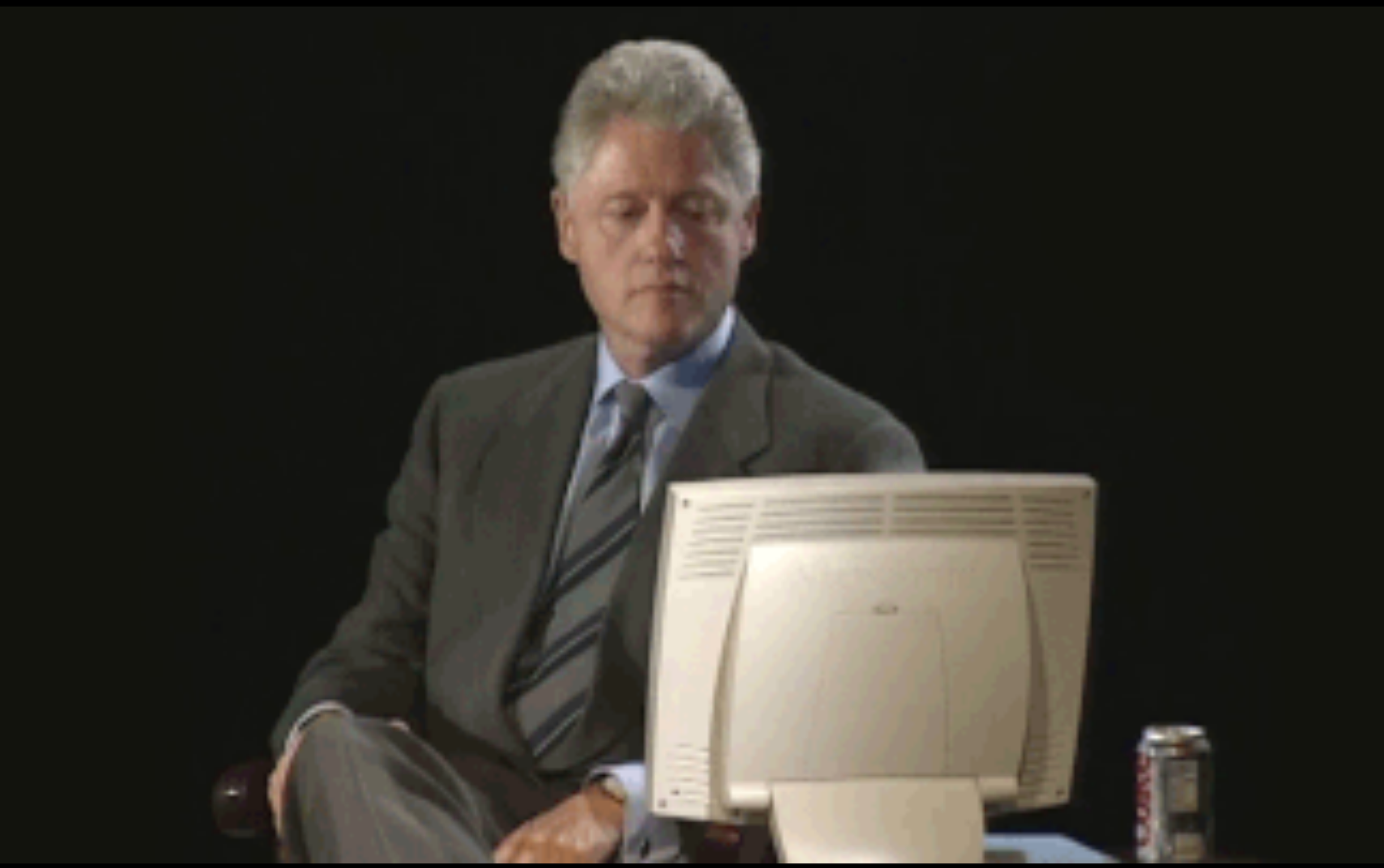
Computeiro

Visão Intuitiva

Minha exp

Perguntem!

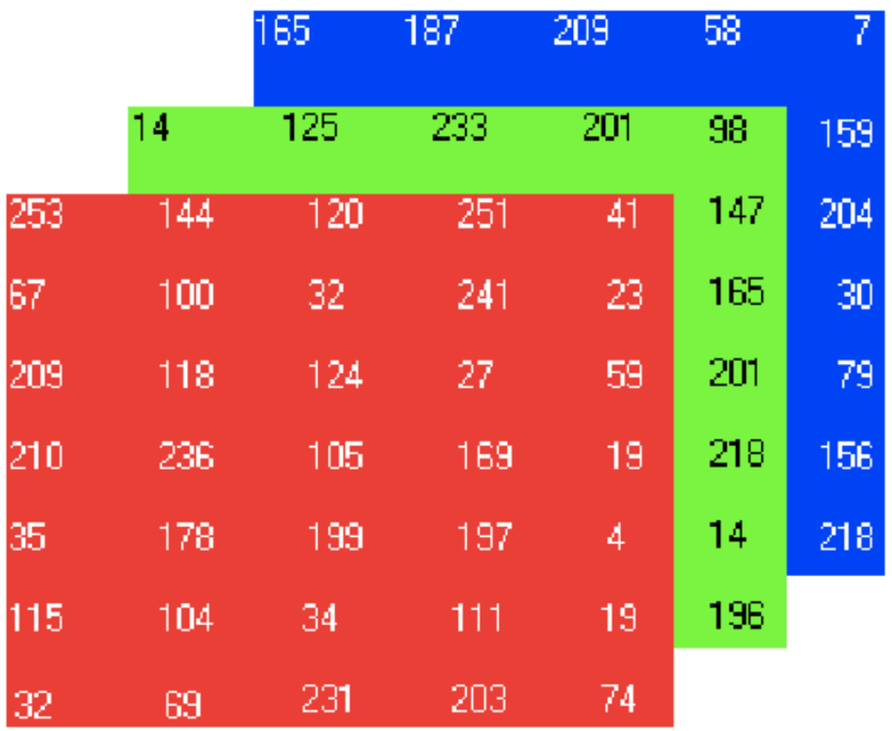
:D



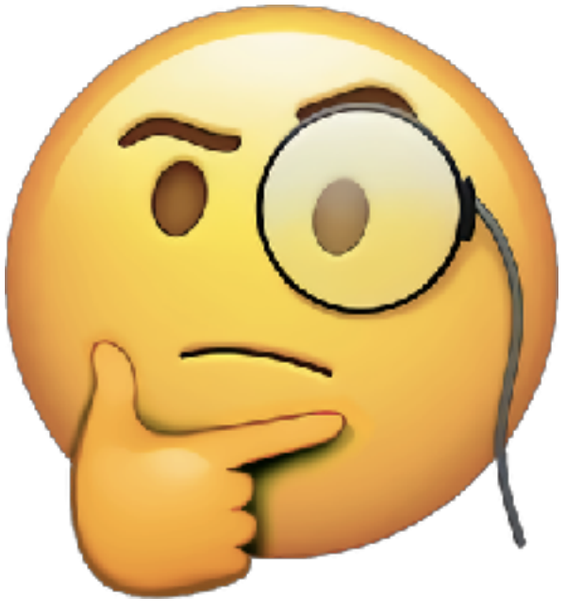
BORA LÁ

		165	187	209	58	7
	14	125	233	201	98	159
253	144	120	251	41	147	204
67	100	32	241	23	165	30
209	118	124	27	59	201	79
210	236	105	169	19	218	156
35	178	199	197	4	14	218
115	104	34	111	19	196	
32	69	231	203	74		

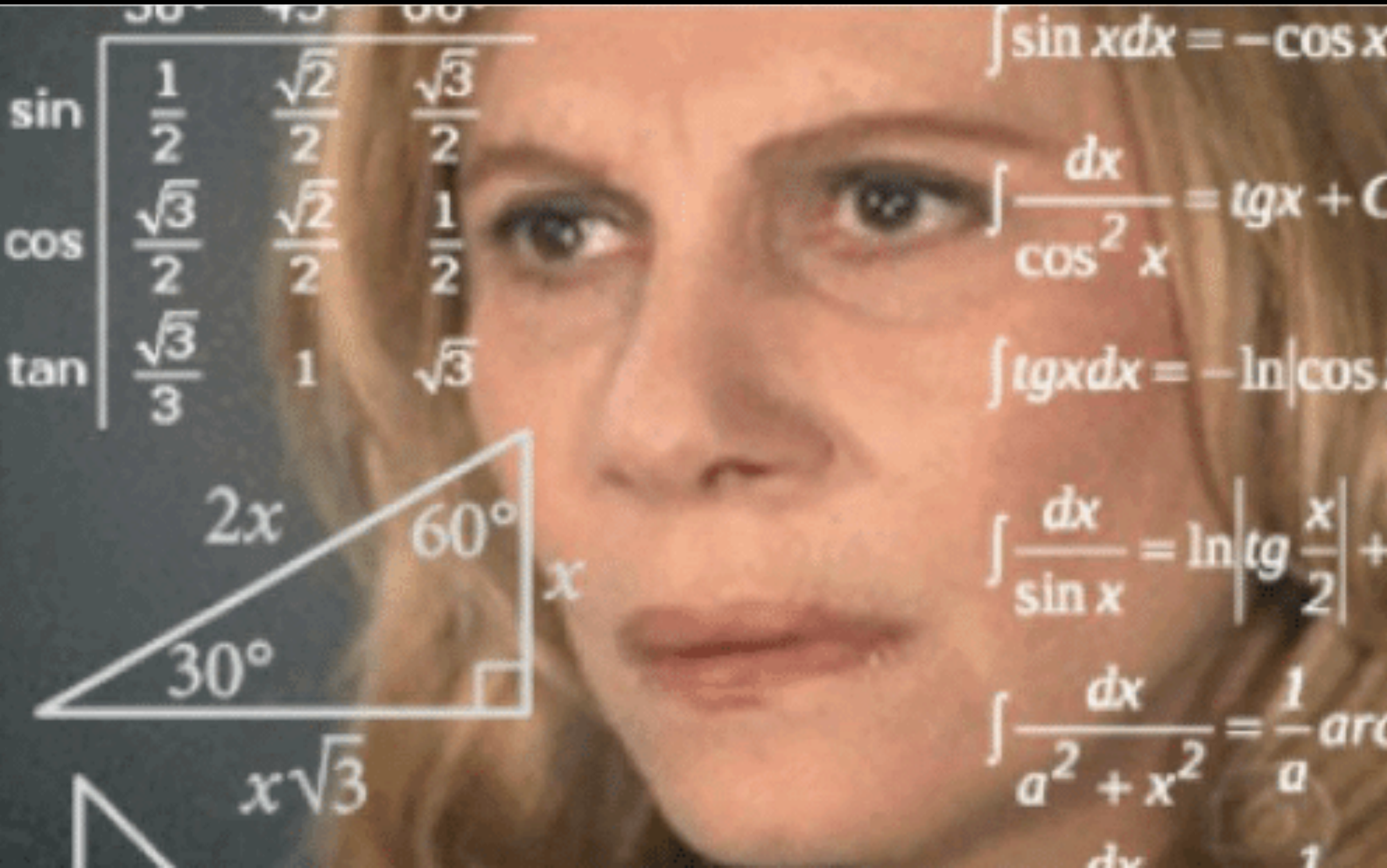
O que é Visão Computacional?



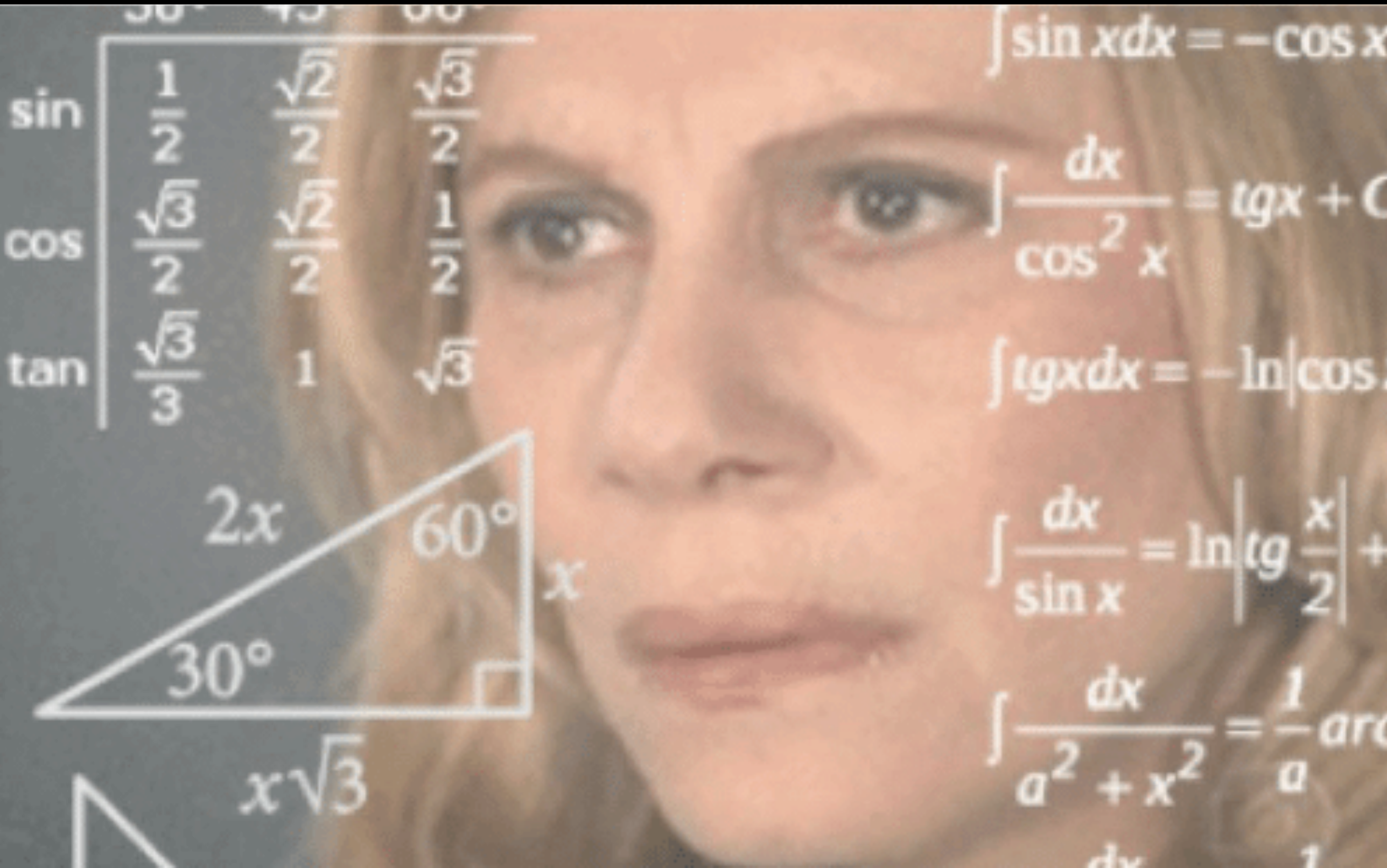
Interpretar!



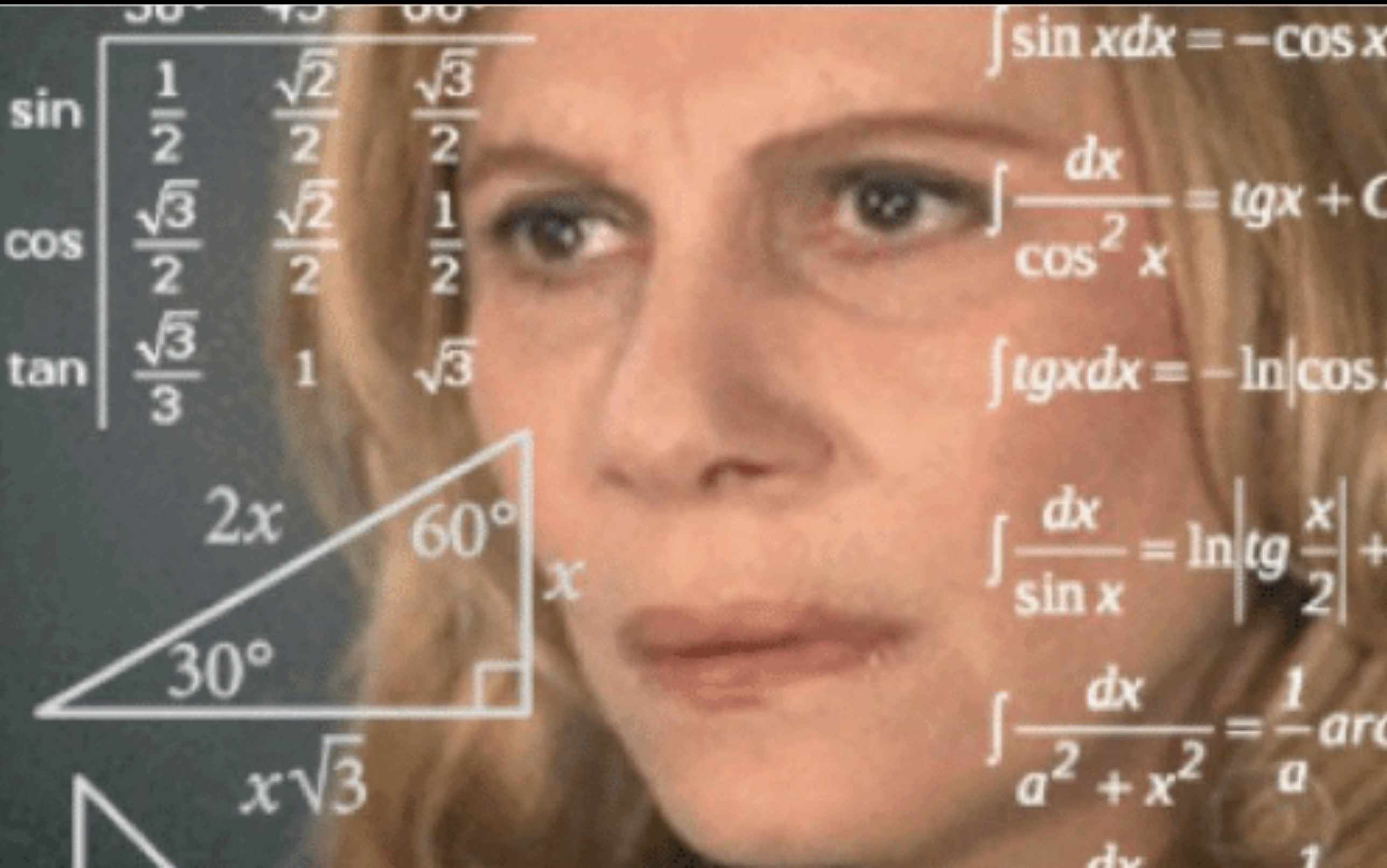
Operando com Imagens



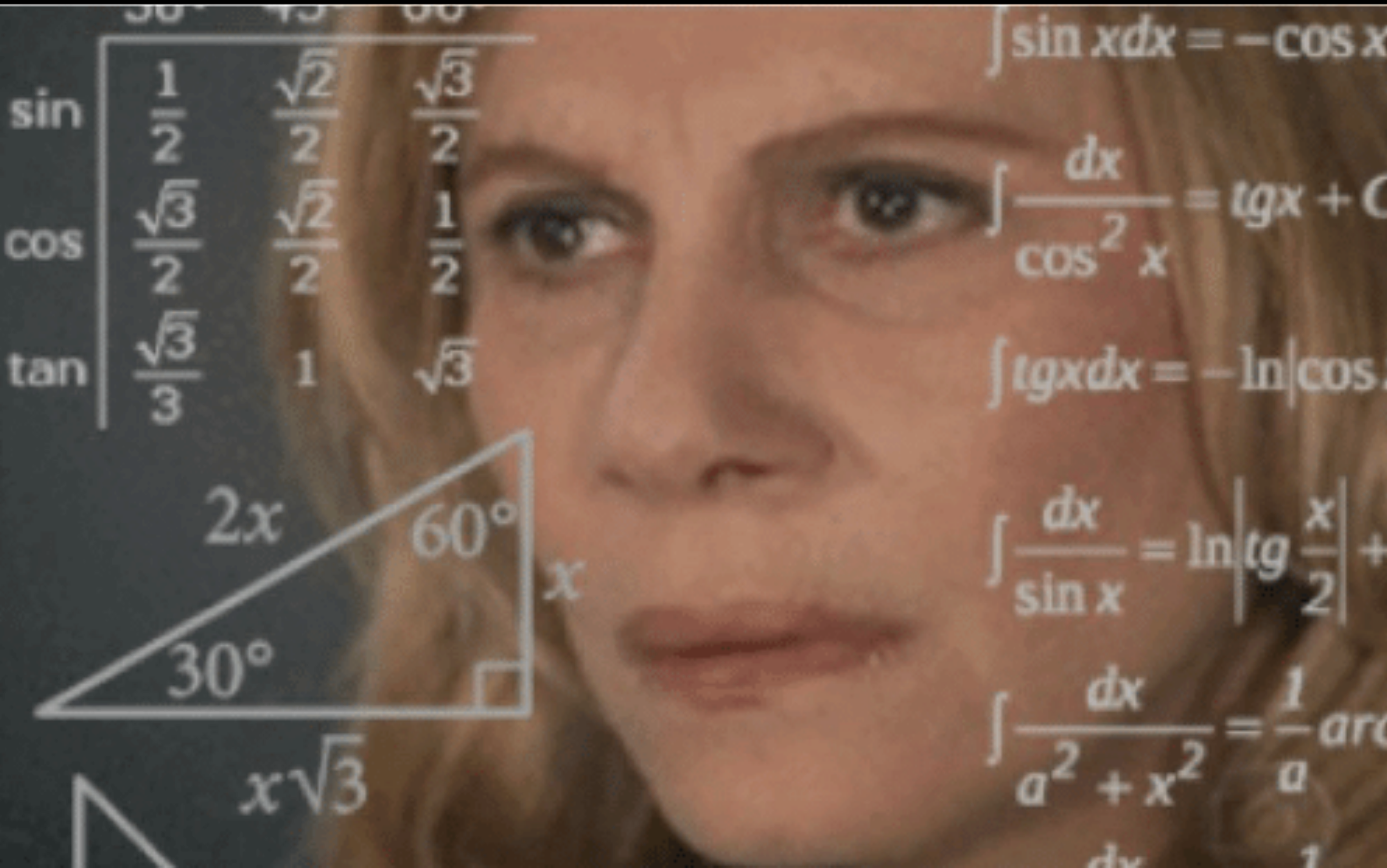
Operando com Imagens



Operando com Imagens

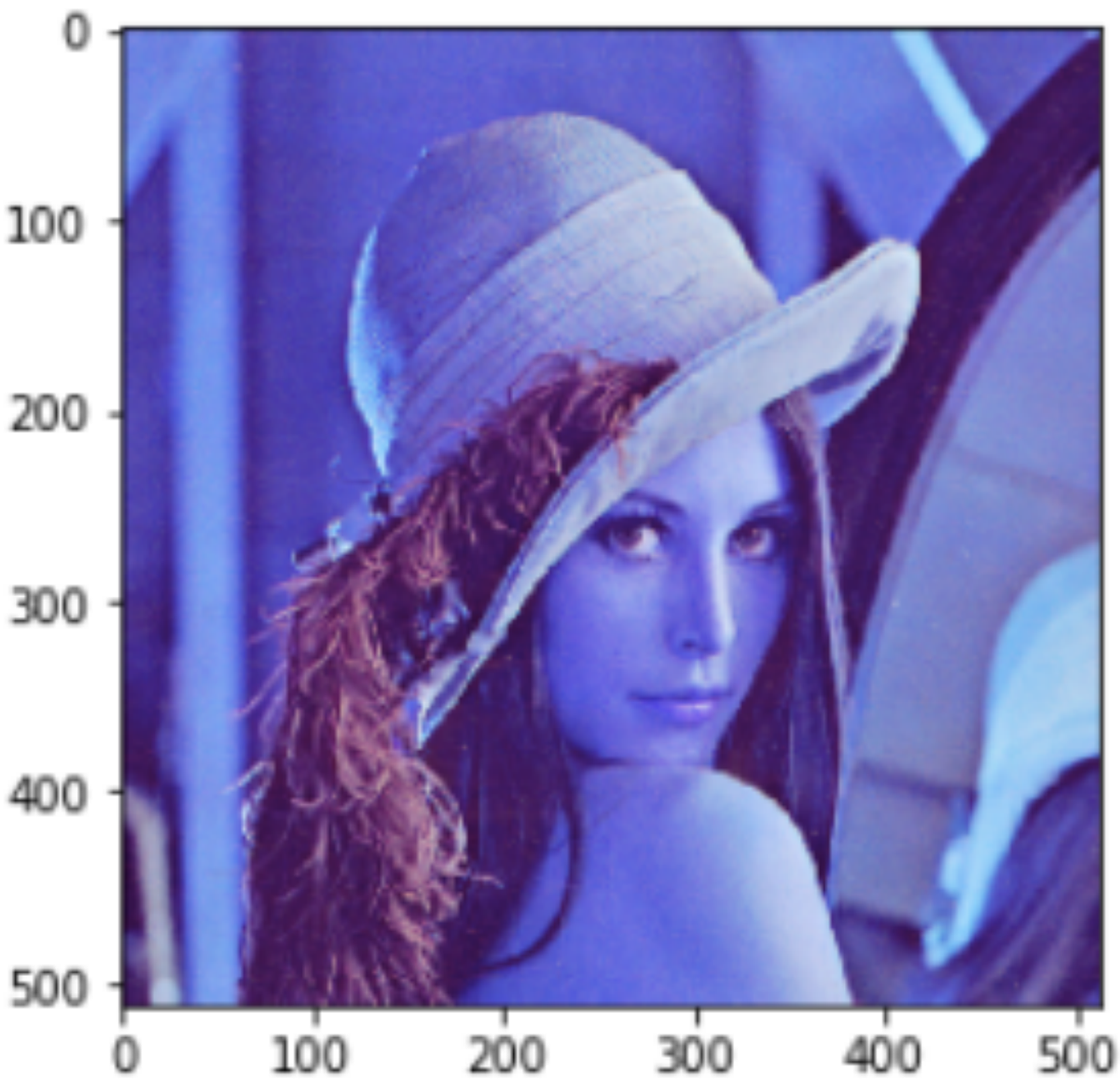


Operando com Imagens



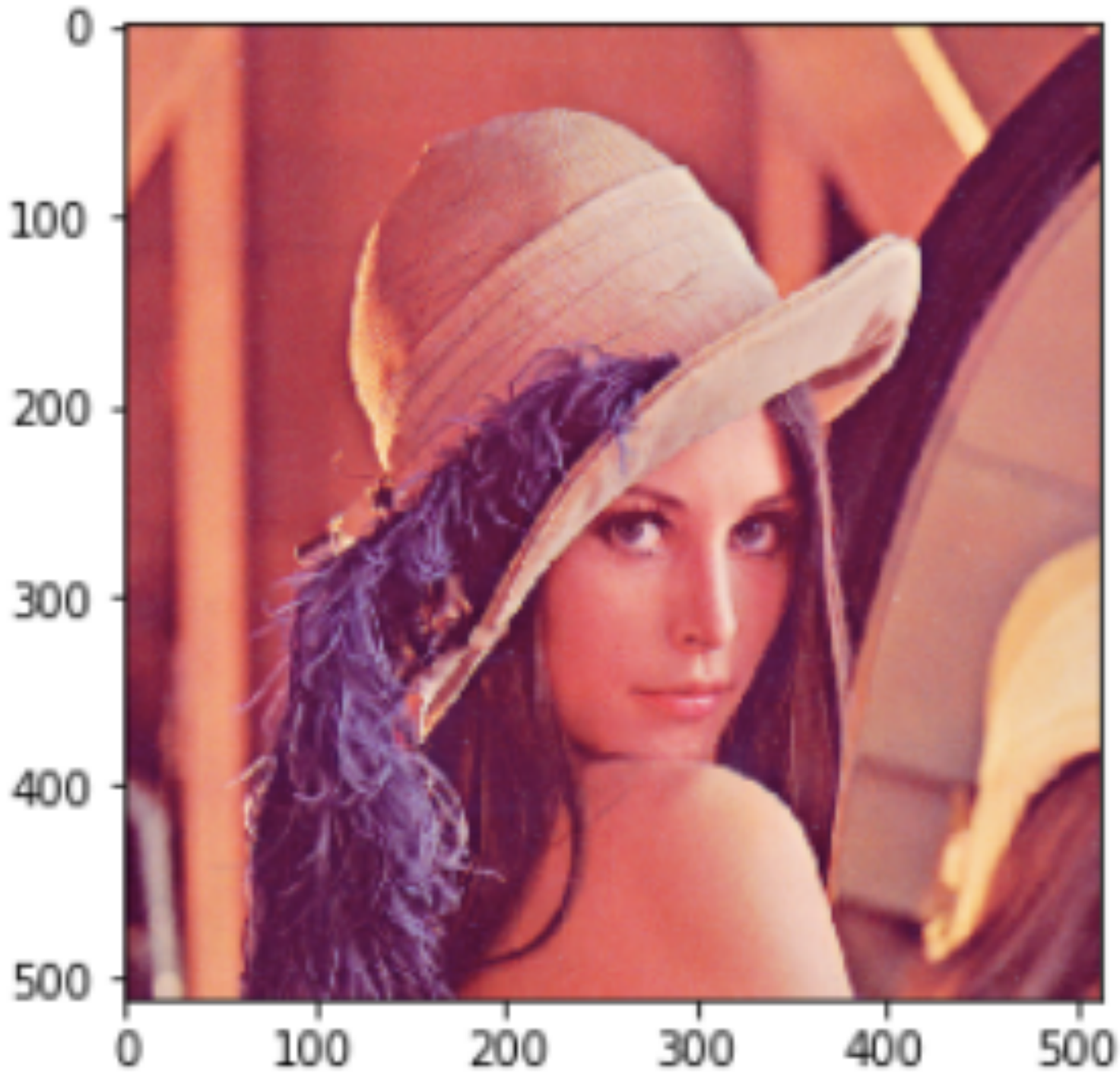
```python
import cv2
import matplotlib.pyplot as plt

img = cv2.imread('lenna.png')
plt.imshow(img);
```

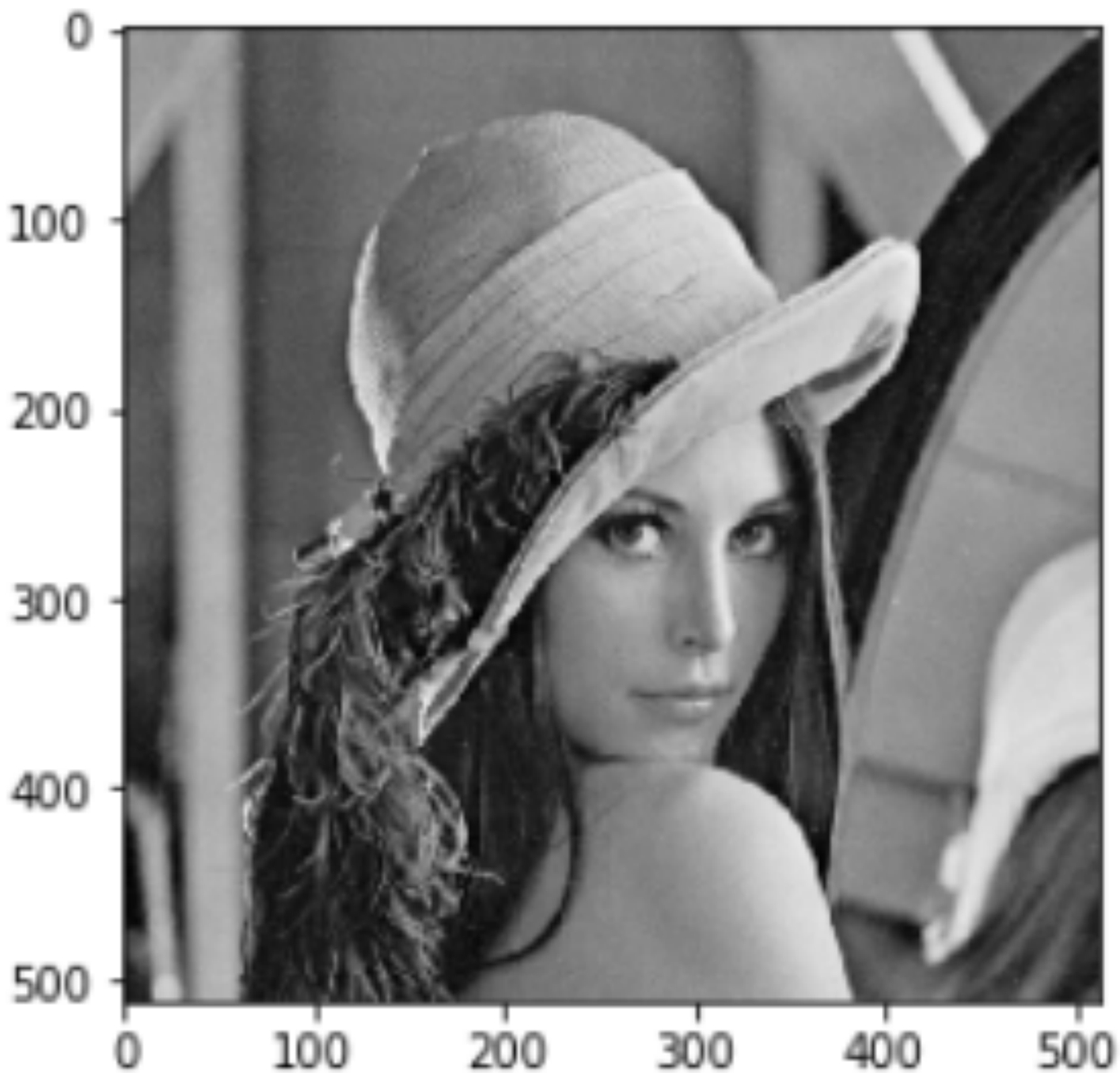


OpenCV

```
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(img);
```



```
img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
plt.imshow(img, cmap='gray');
```



Mas..

COMO?

Exemplo!

Opção 1) Média

$$\text{Gray}: Y \leftarrow 0.333 \cdot R + 0.333 \cdot G + 0.333 \cdot B$$

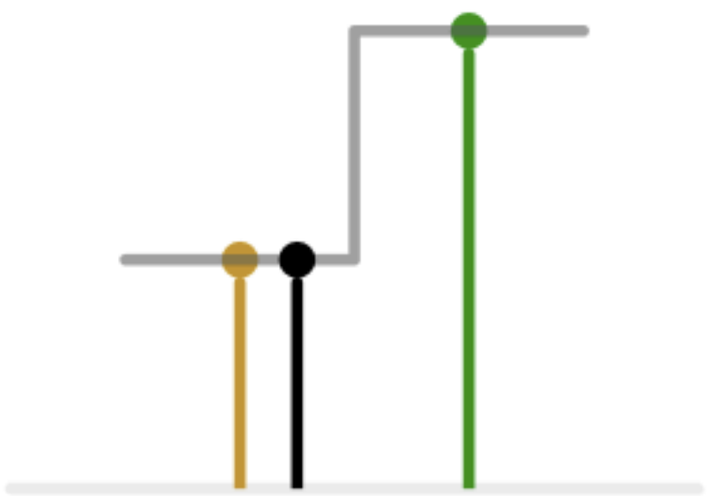
Opção 2) Weighted Luminance

$$\text{Gray}: Y \leftarrow 0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B$$

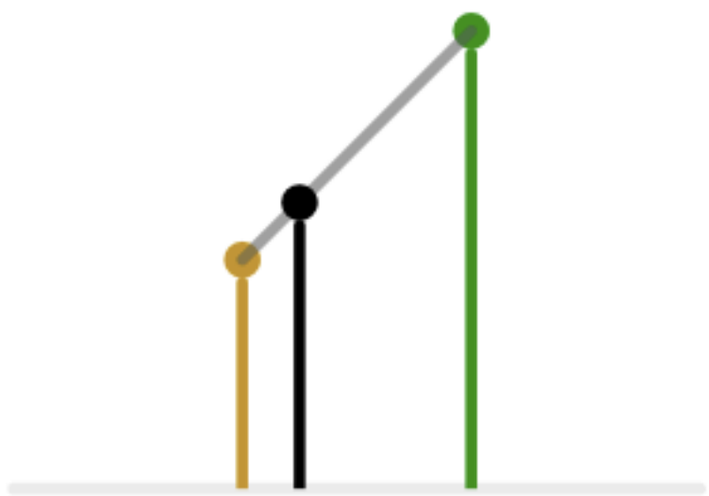
Interpolação

(1024x1024) (64x64) (32x32)

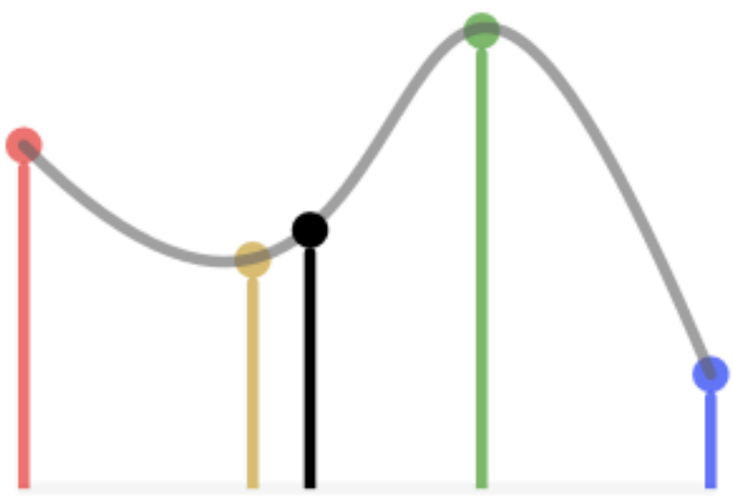
Interpolação



1D nearest-neighbour

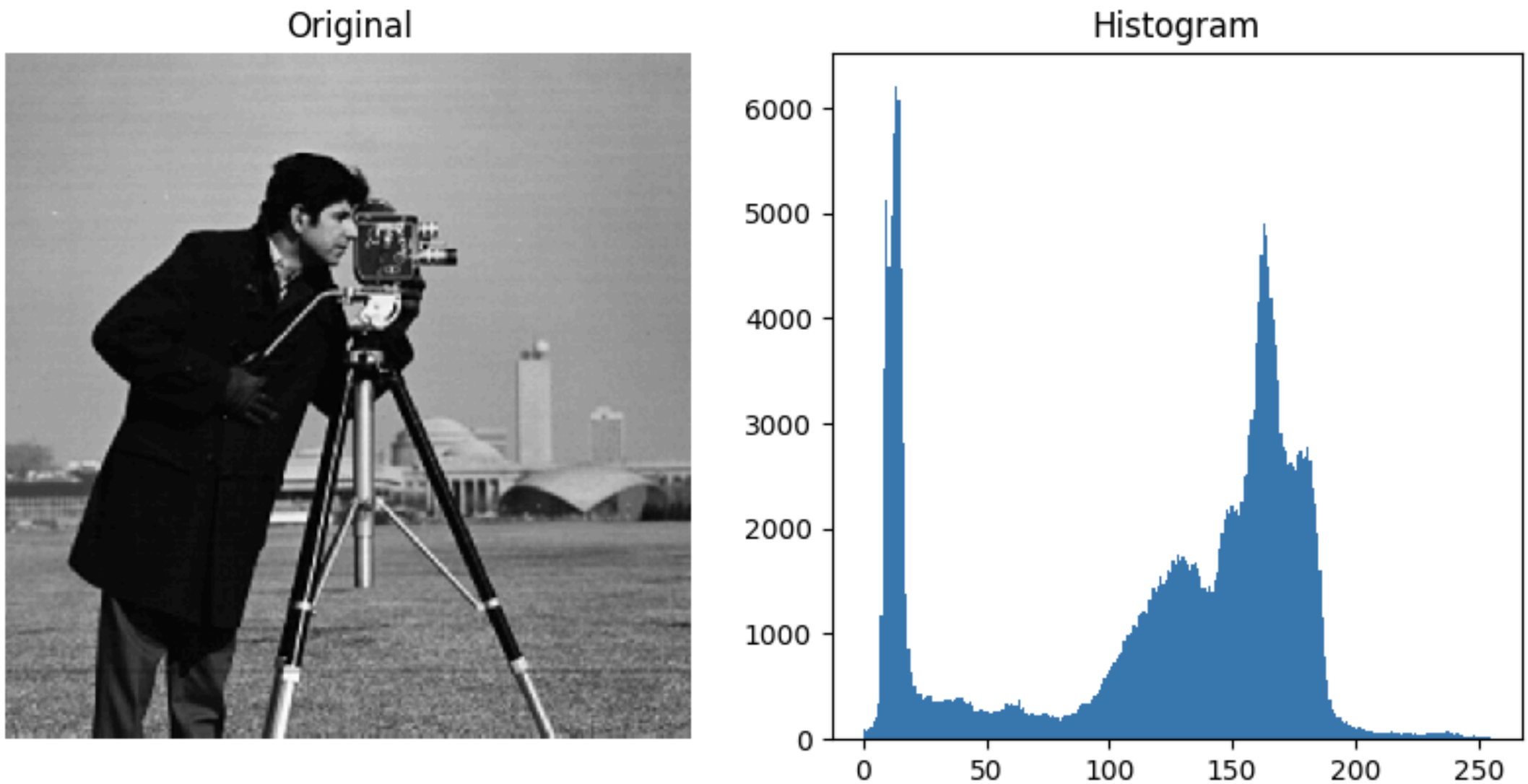


Linear

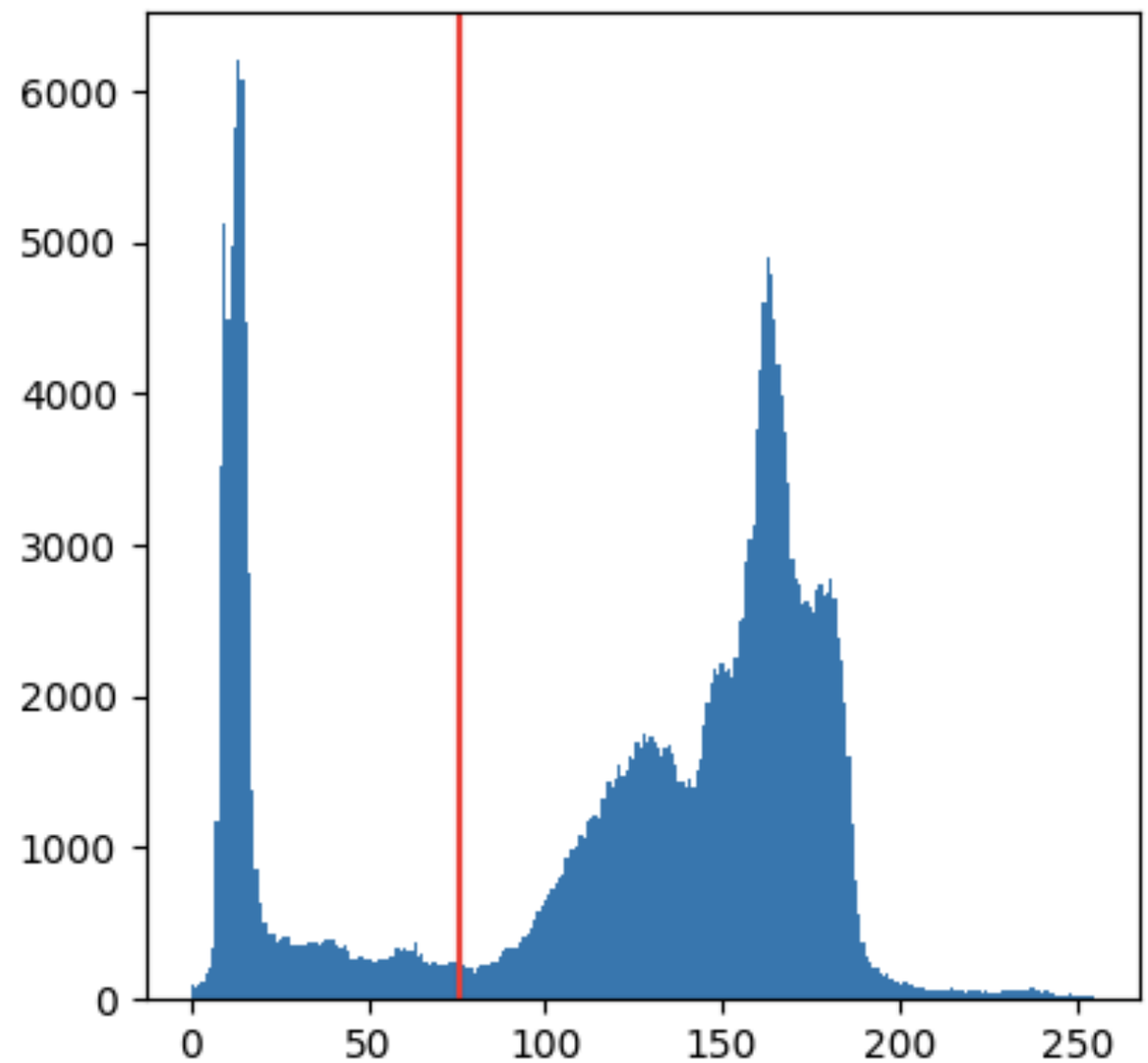


Cubic

Segmentação de Imagens

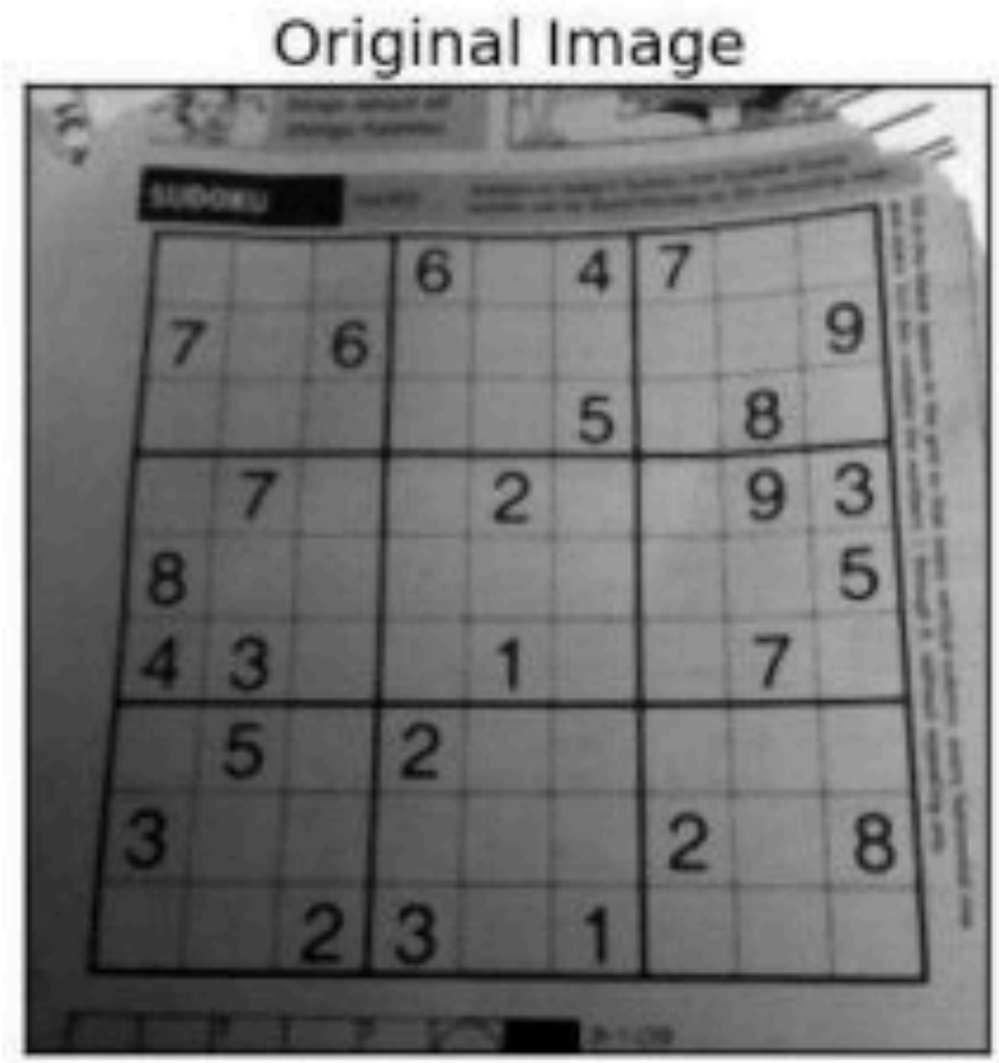


Segmentação de Imagens



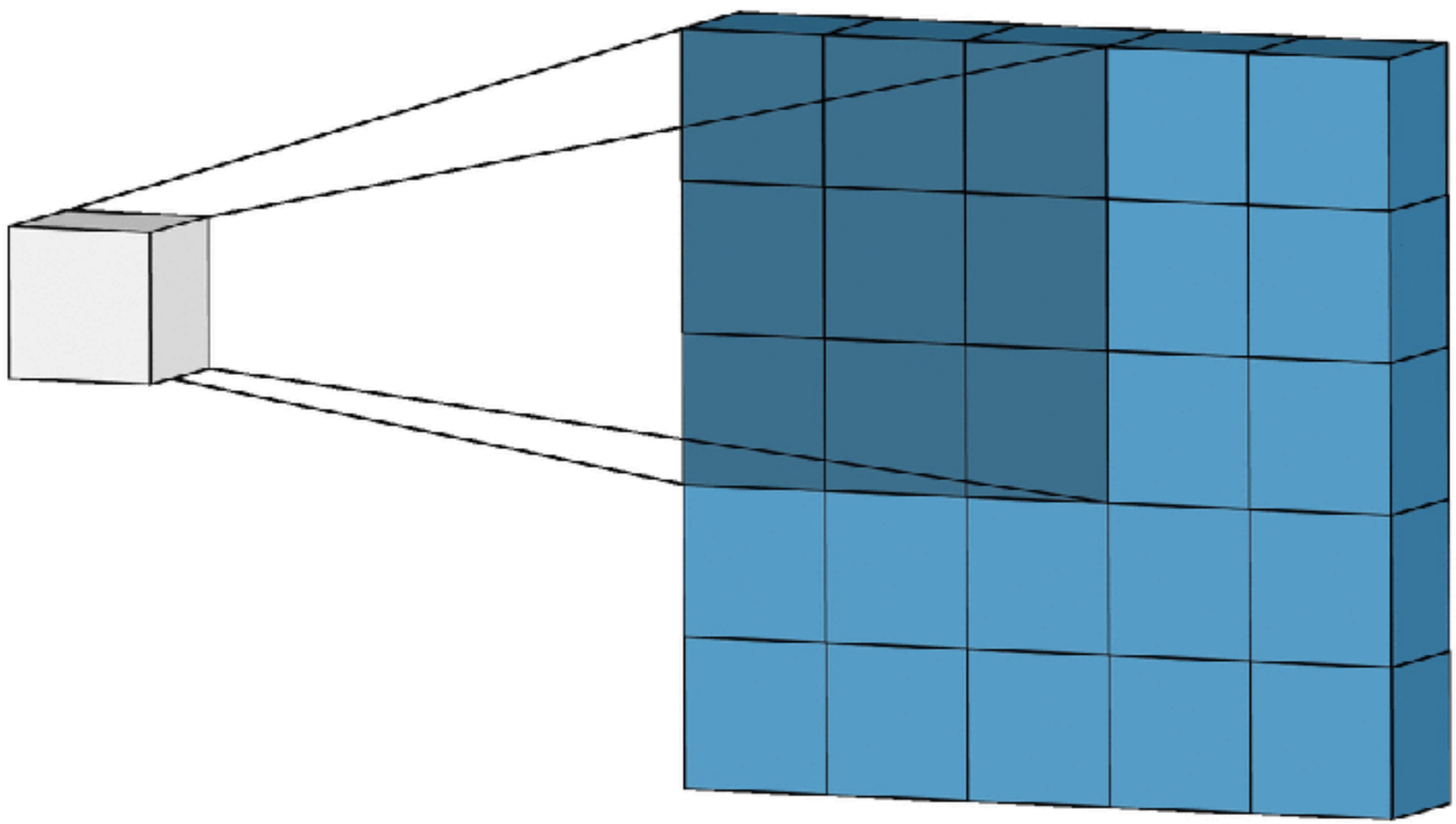
- Como determinar o limiar?
- Otsu (e Outras Estratégias)

Segmentação de Imagens

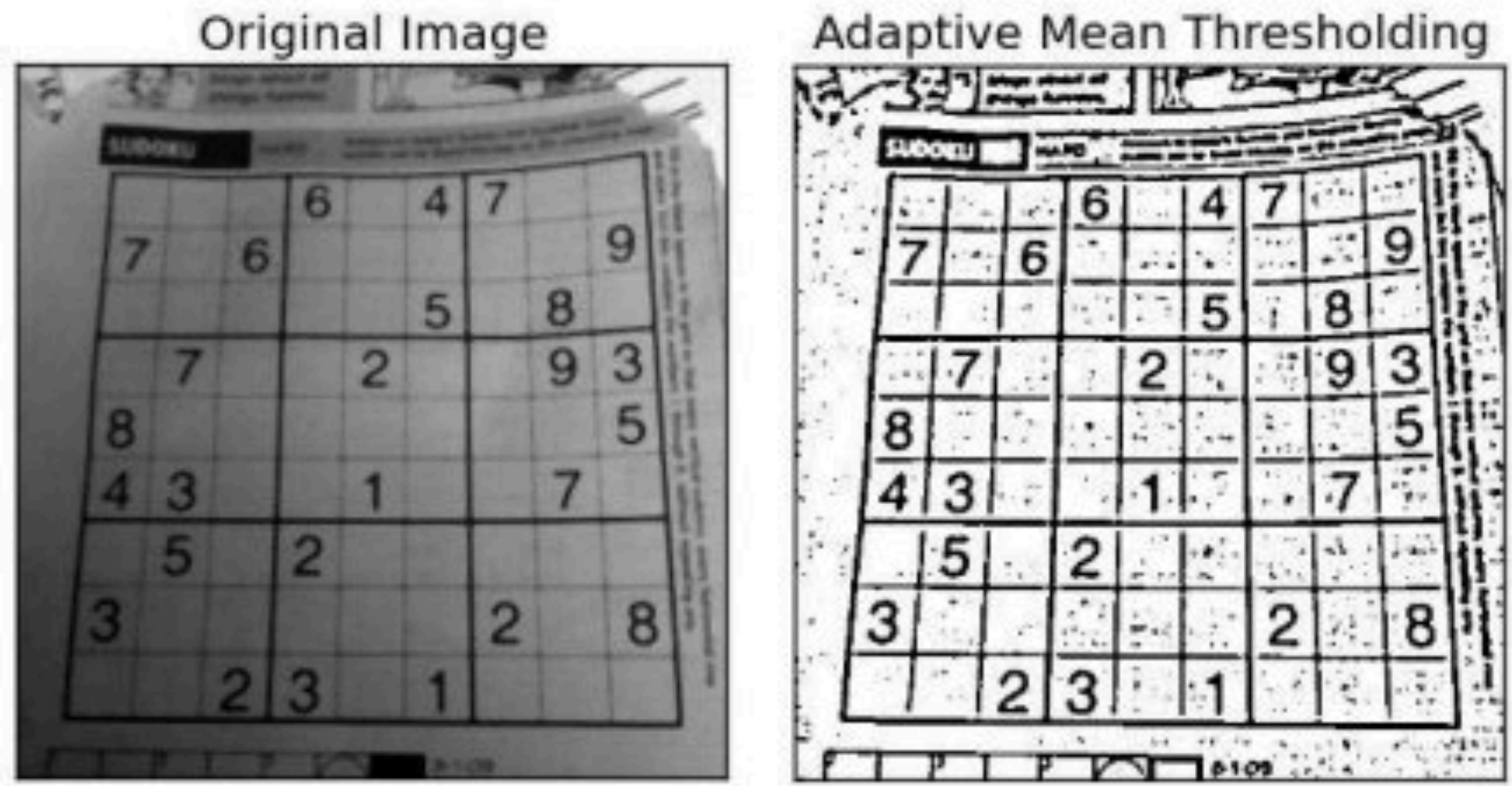


Segmentação de Imagens

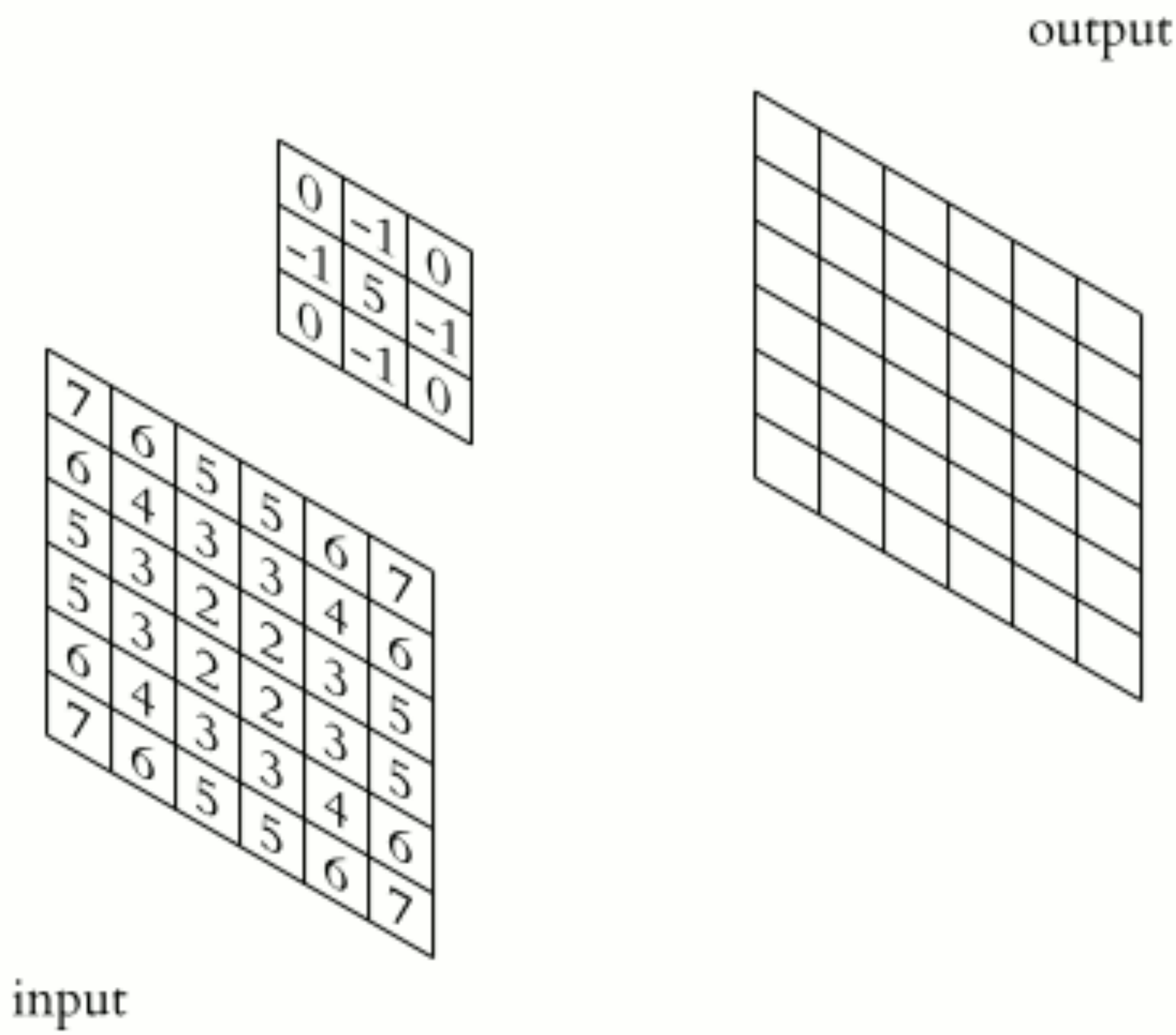
(Convoluções)



Convoluções!

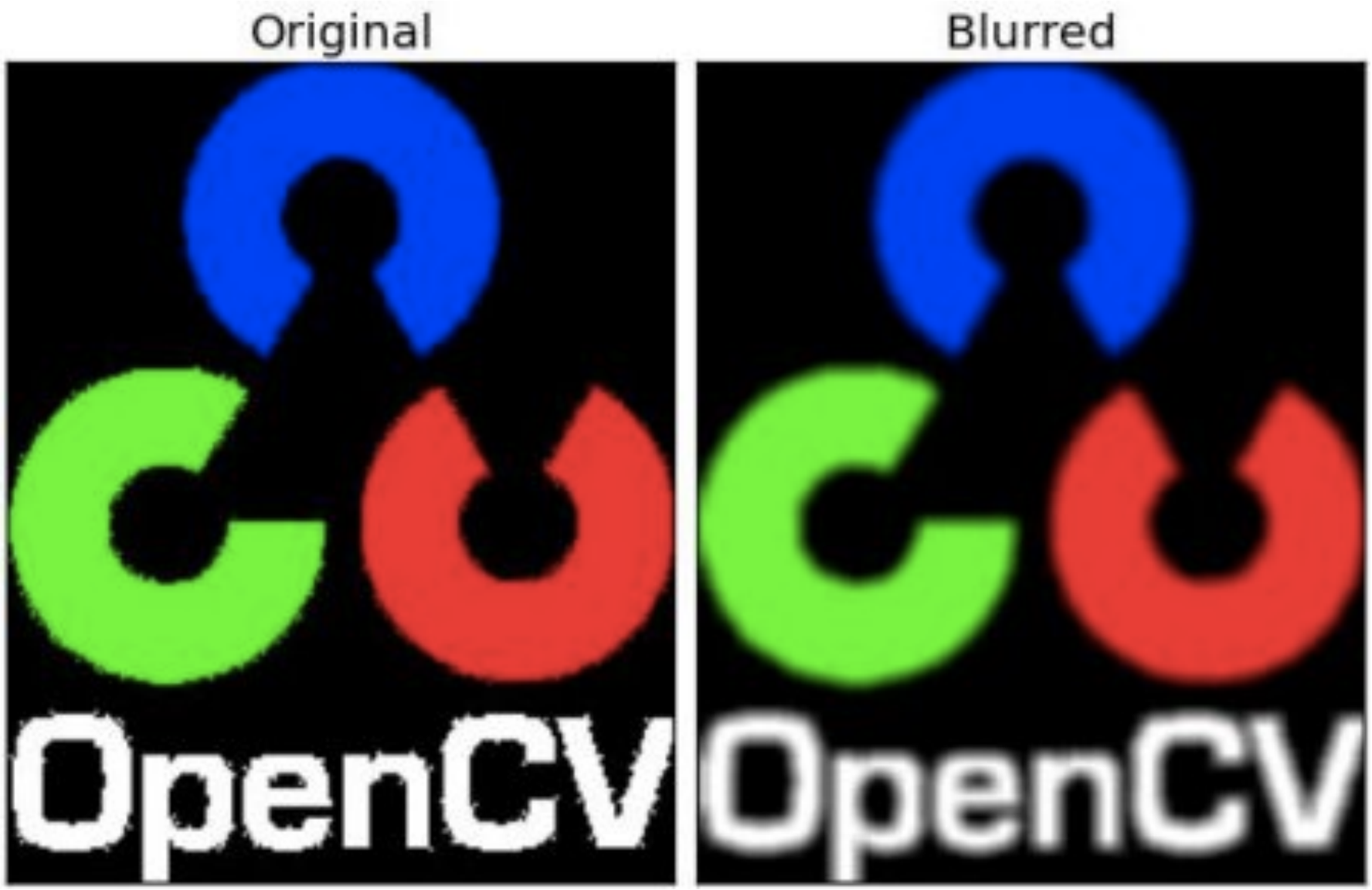


Convoluções e Filtros!



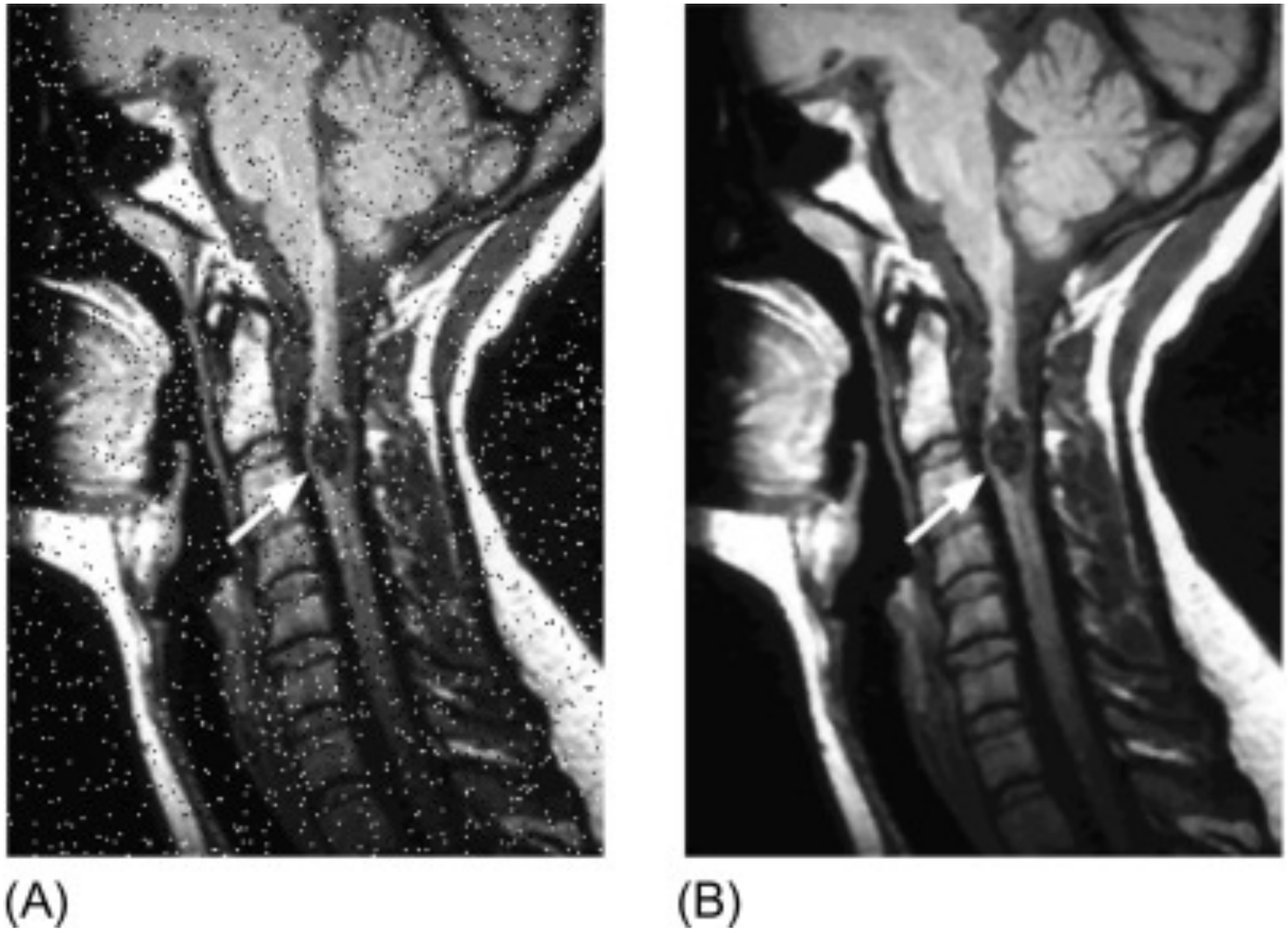
Stride
Kernel
Padding

Convoluções e Filtros!



$$K = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Convoluções e Filtros!



filtro da mediana

Detecção de Bordas (Sobel Operator)

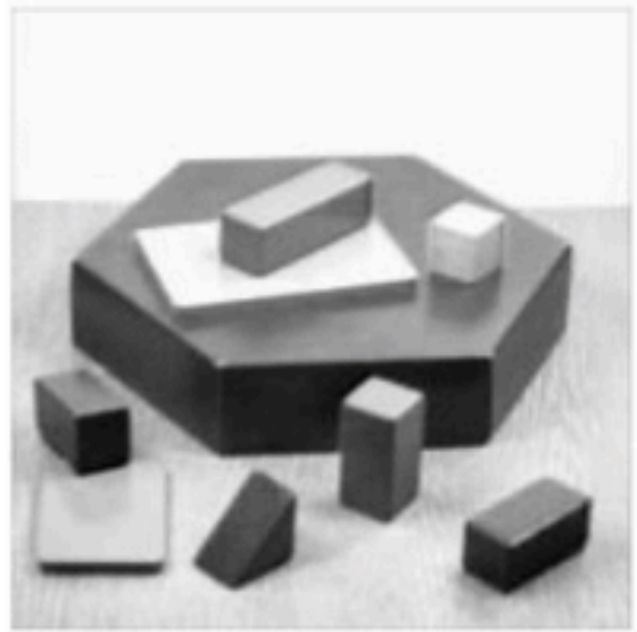
$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Detecção de Bordas

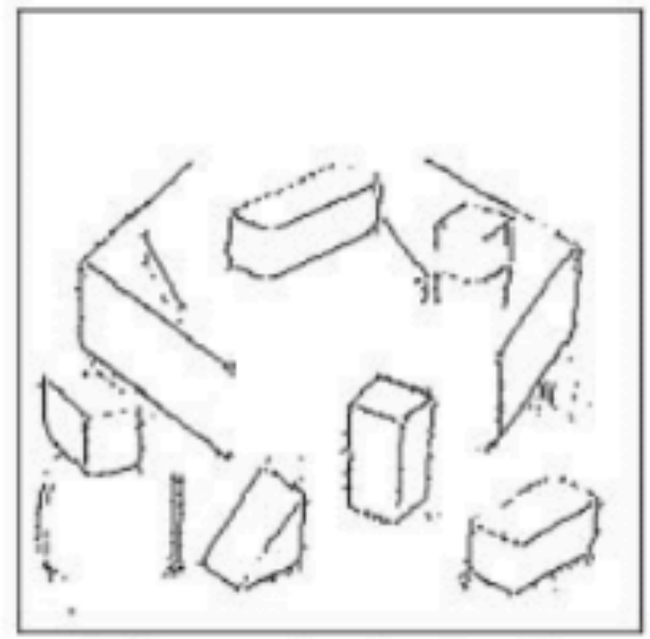
Algoritmo 1 Determinação de pontos de borda em uma imagem

entradas: uma imagem de entrada **f** com dimensões $M \times N$ pixels e um limiar T.
for $x = 0$ até $M - 1$ **do**
  for $y = 0$ até $N - 1$ **do**
    // calcular a magnitude do gradiente $\nabla f(x, y)$
    $$\nabla f(x,y) = \sqrt{\left(\frac{\partial f(x,y)}{\partial x}\right)^2 + \left(\frac{\partial f(x,y)}{\partial y}\right)^2}$$
    // efetuar a limiarização
    if $\nabla f(x, y) > T$ **then**
      (x, y) é um ponto da borda
    end if
  end for
end for

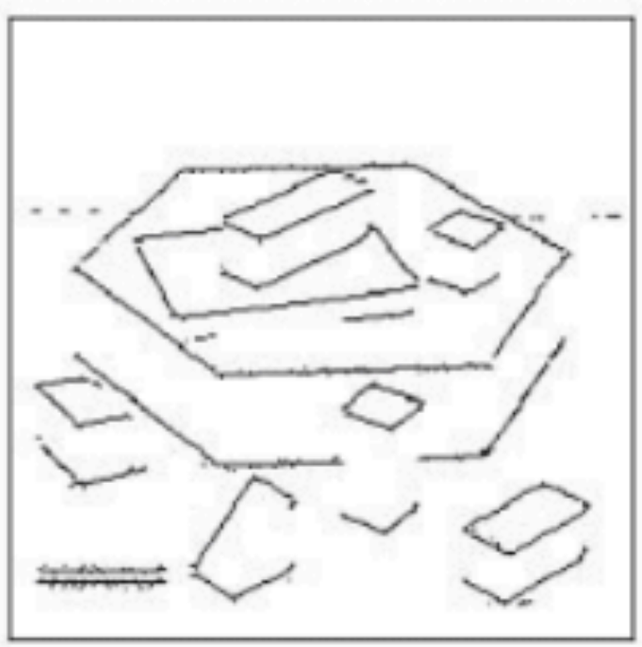
Detecção de Bordas



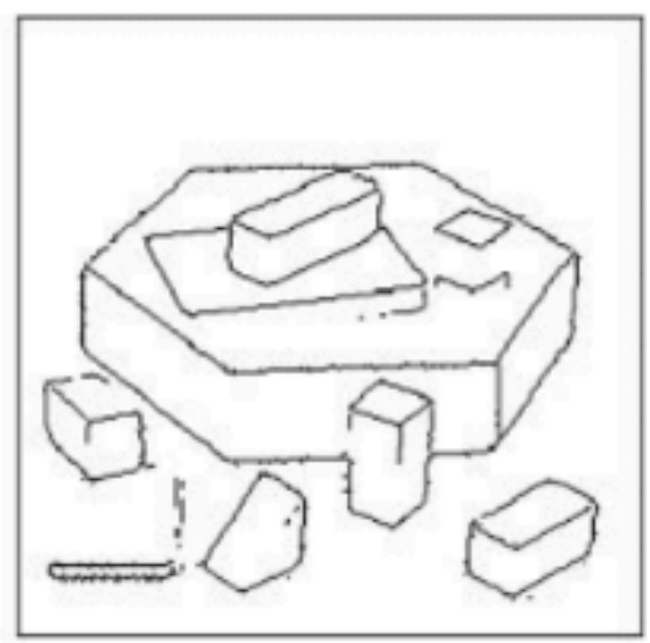
(a) imagem original



(b) resultado de G_x



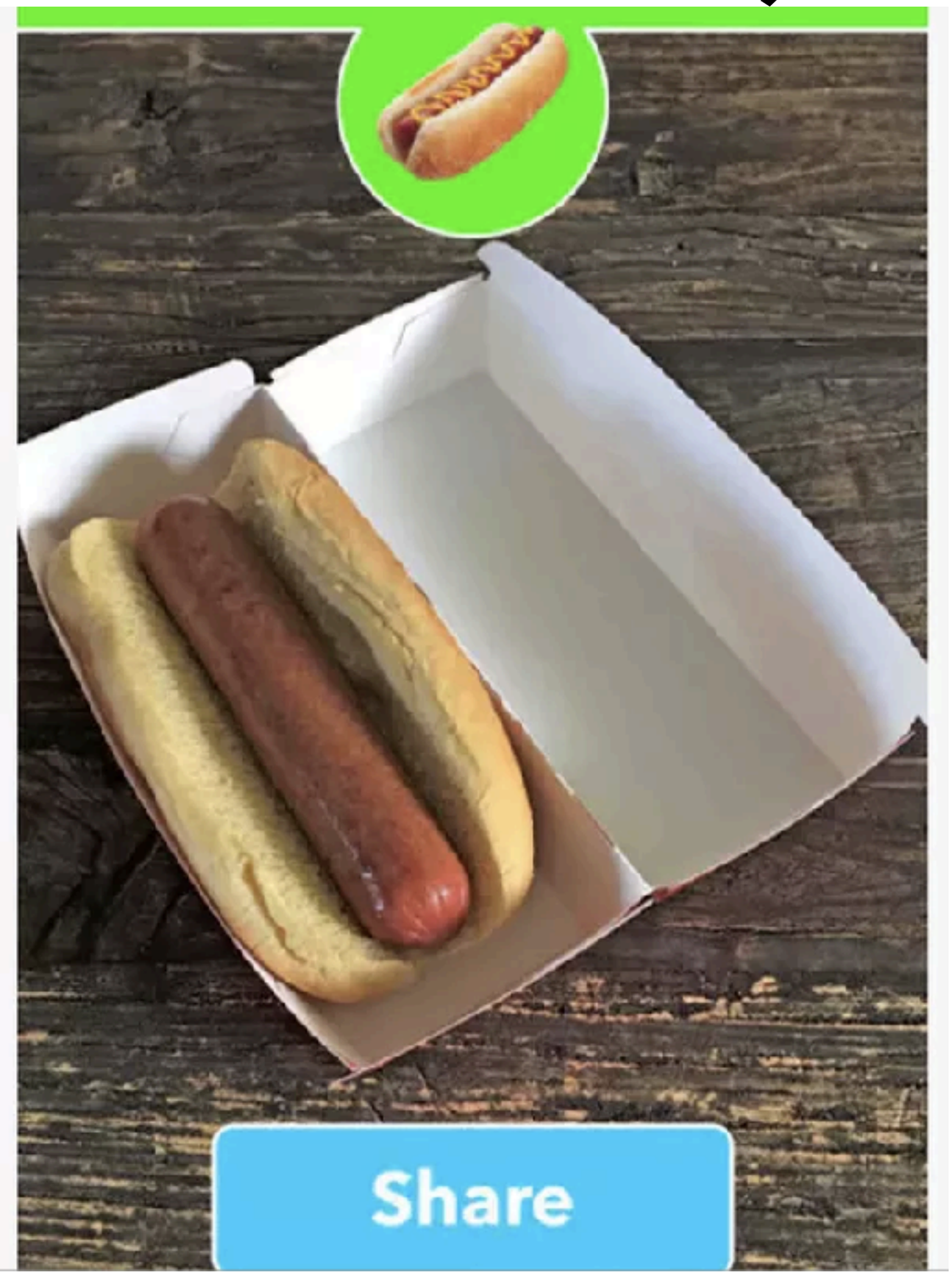
(c) resultado de G_y



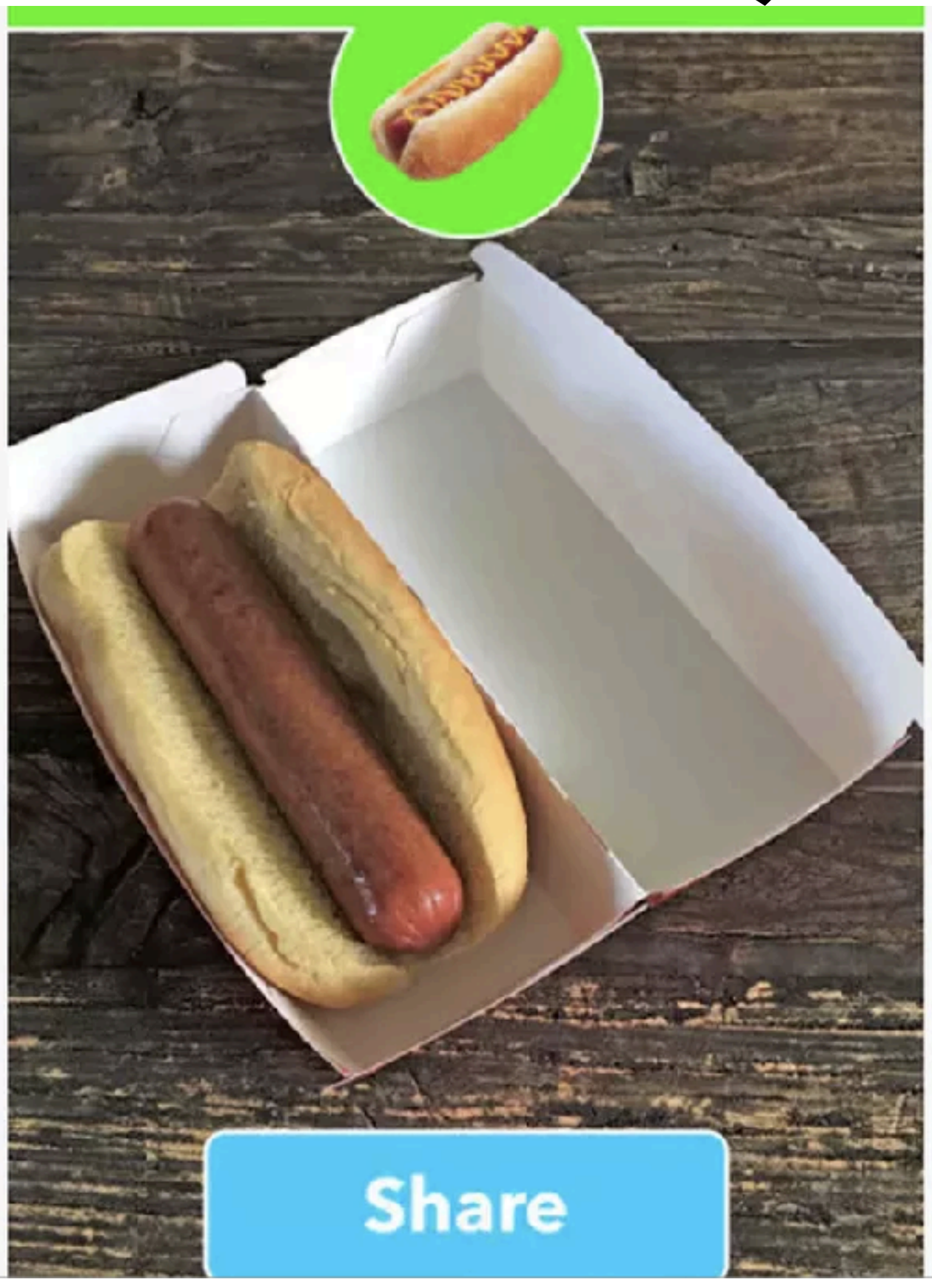
(d) combinação de G_x e G_y

SO WHAT

Classificação de Imagens



Classificação de Imagens



Opção 1)

→ Classifier

Muitos pixels

Resize!

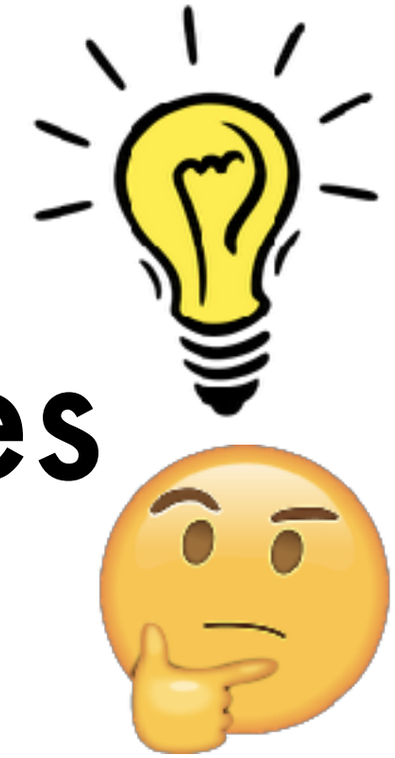
Espaço para muita variância (0-255)

Filtros!

OK se depender de poucos pixels

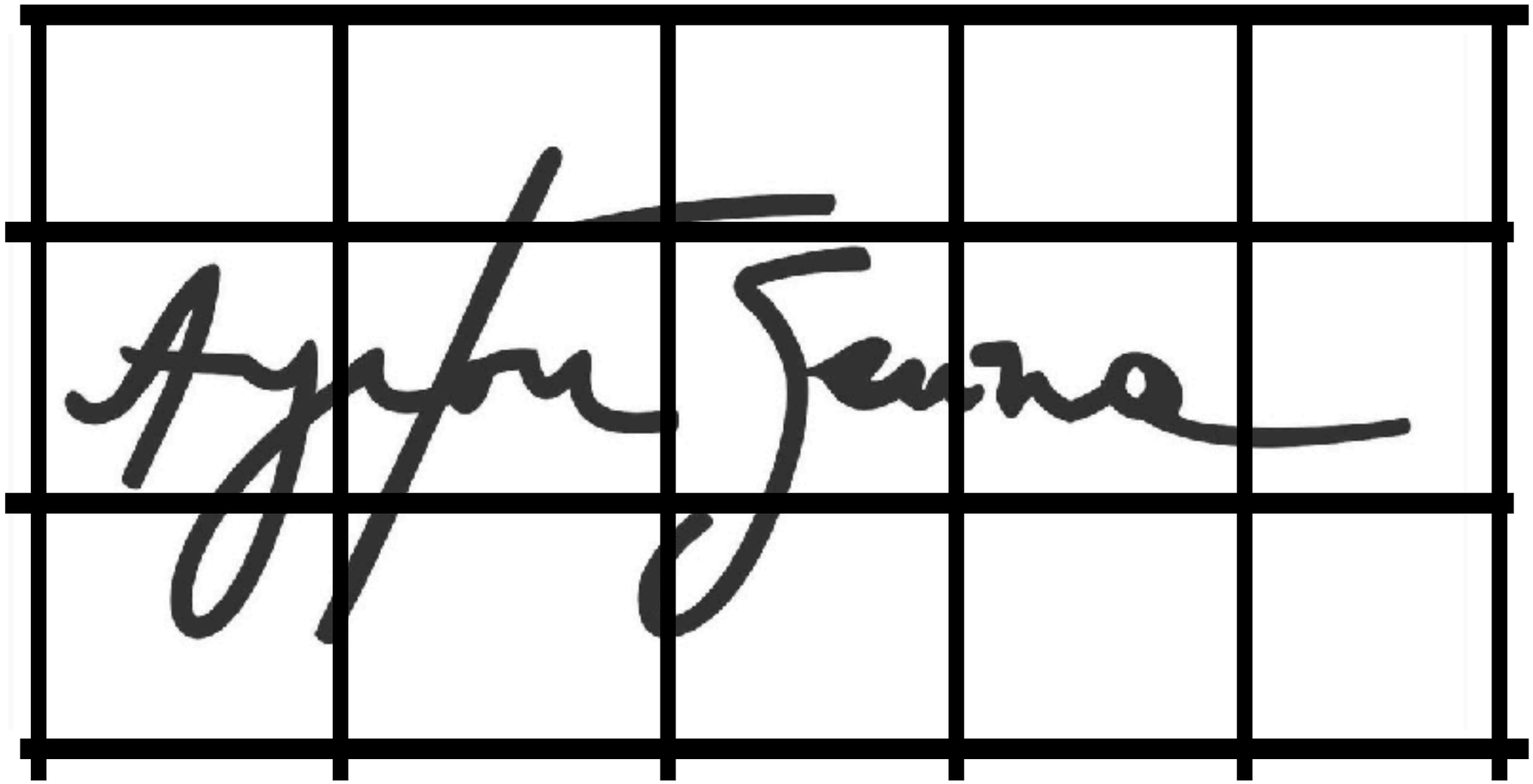
Opção 2)

Features



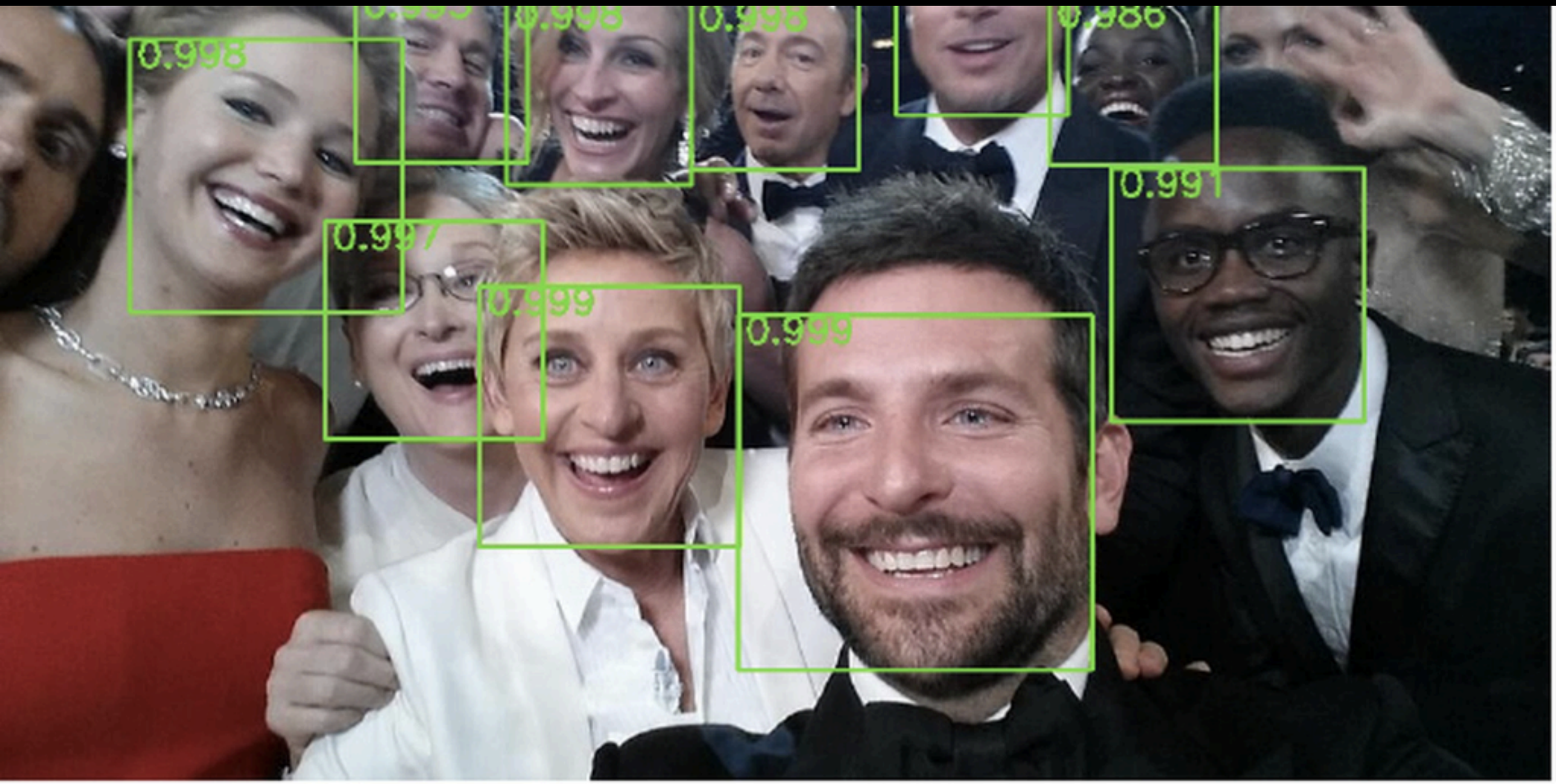
Classifier

% Pixels Brancos

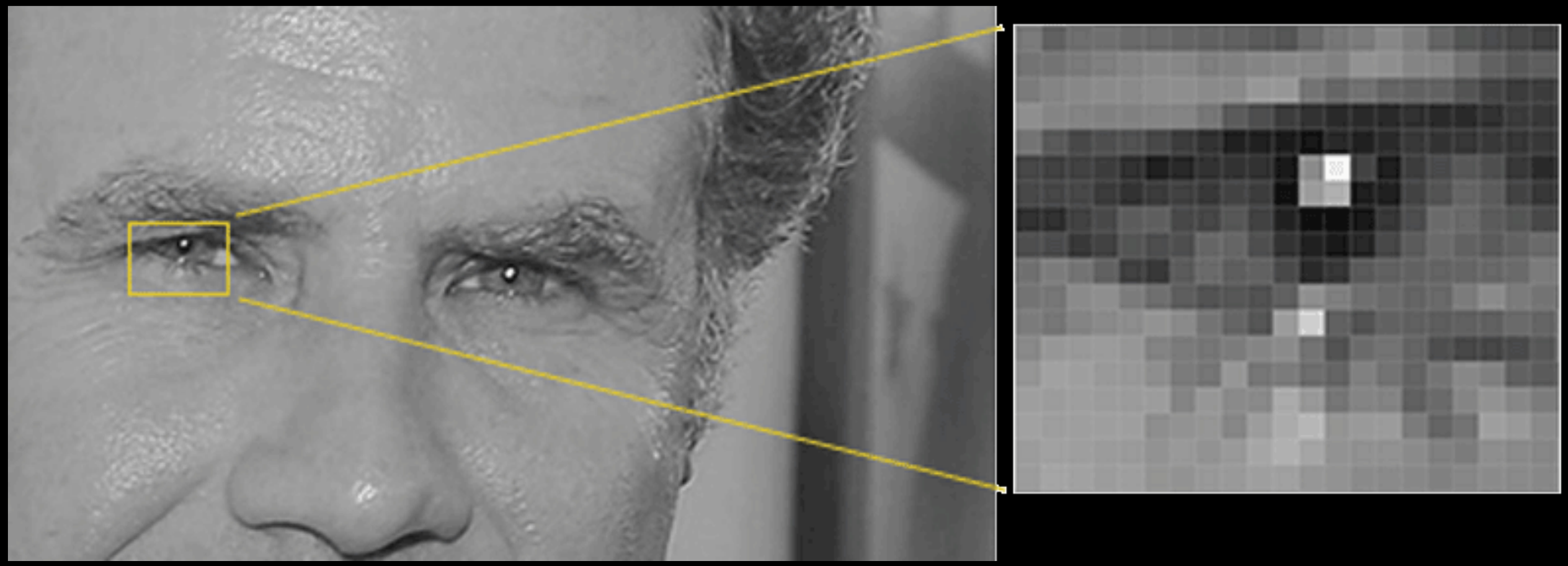


% Pixels Brancos

Grid

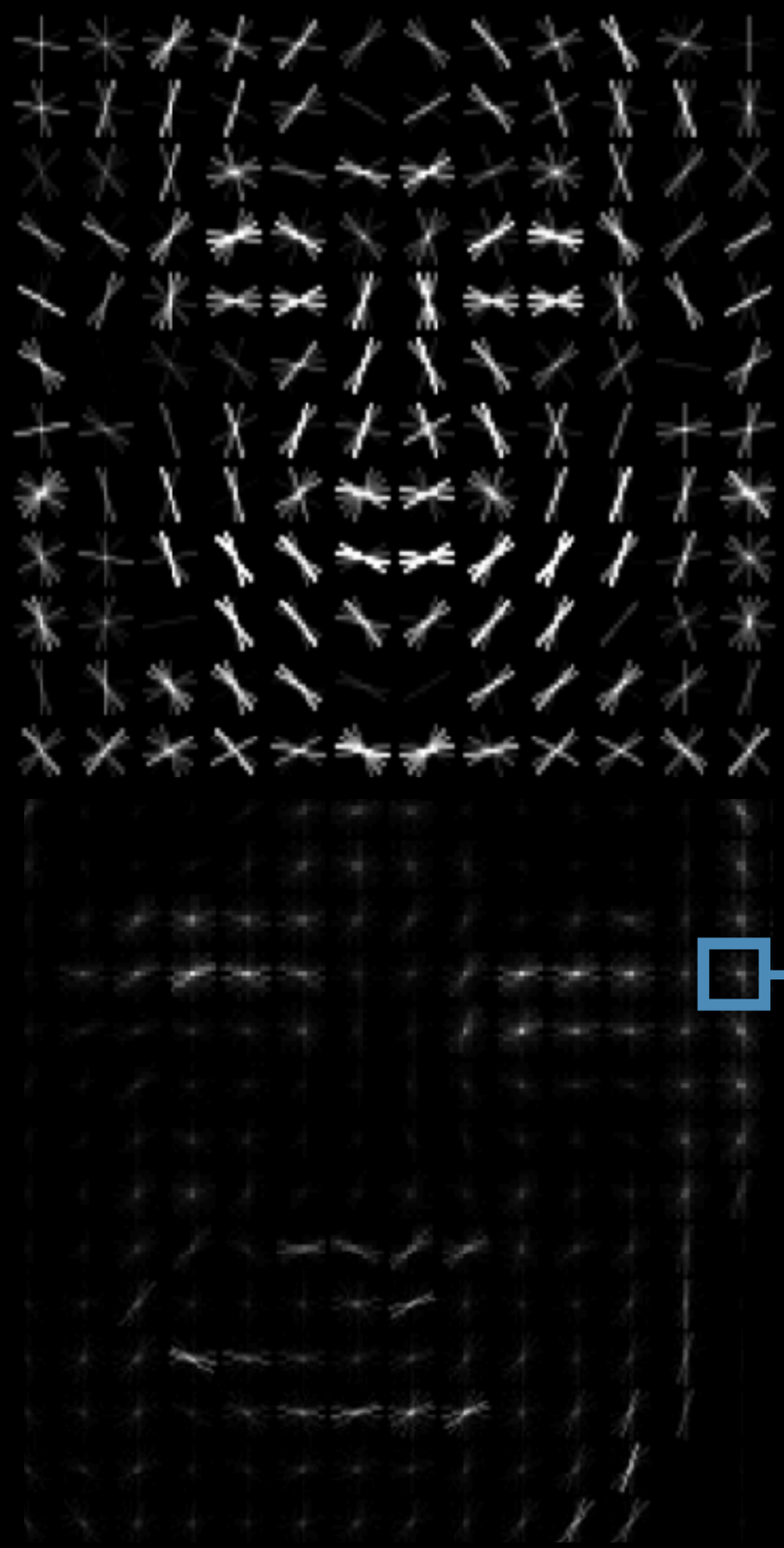

0.998
0.998
0.998
0.997
0.999
0.999
0.991

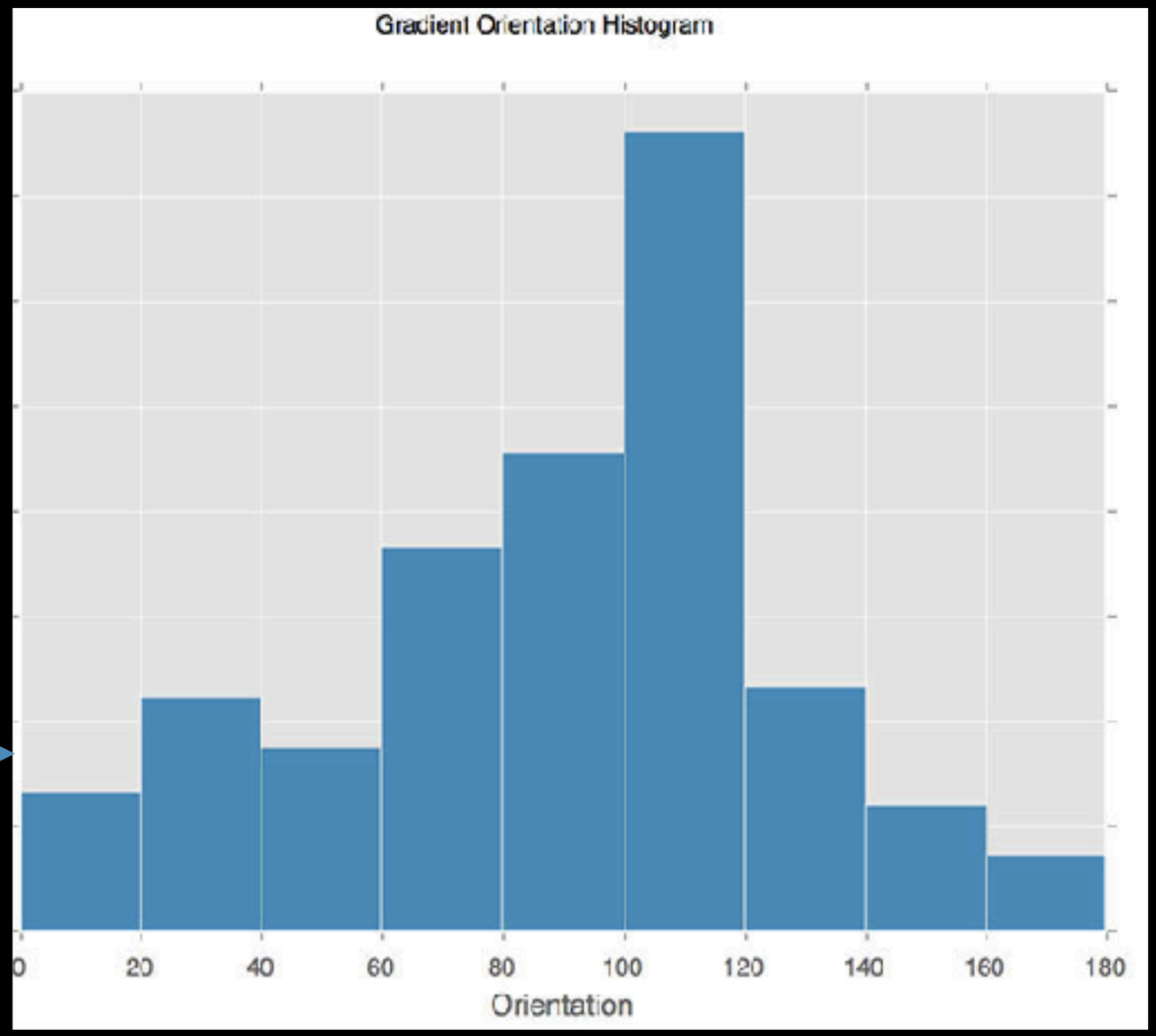
HoG: Histogram of Gradients

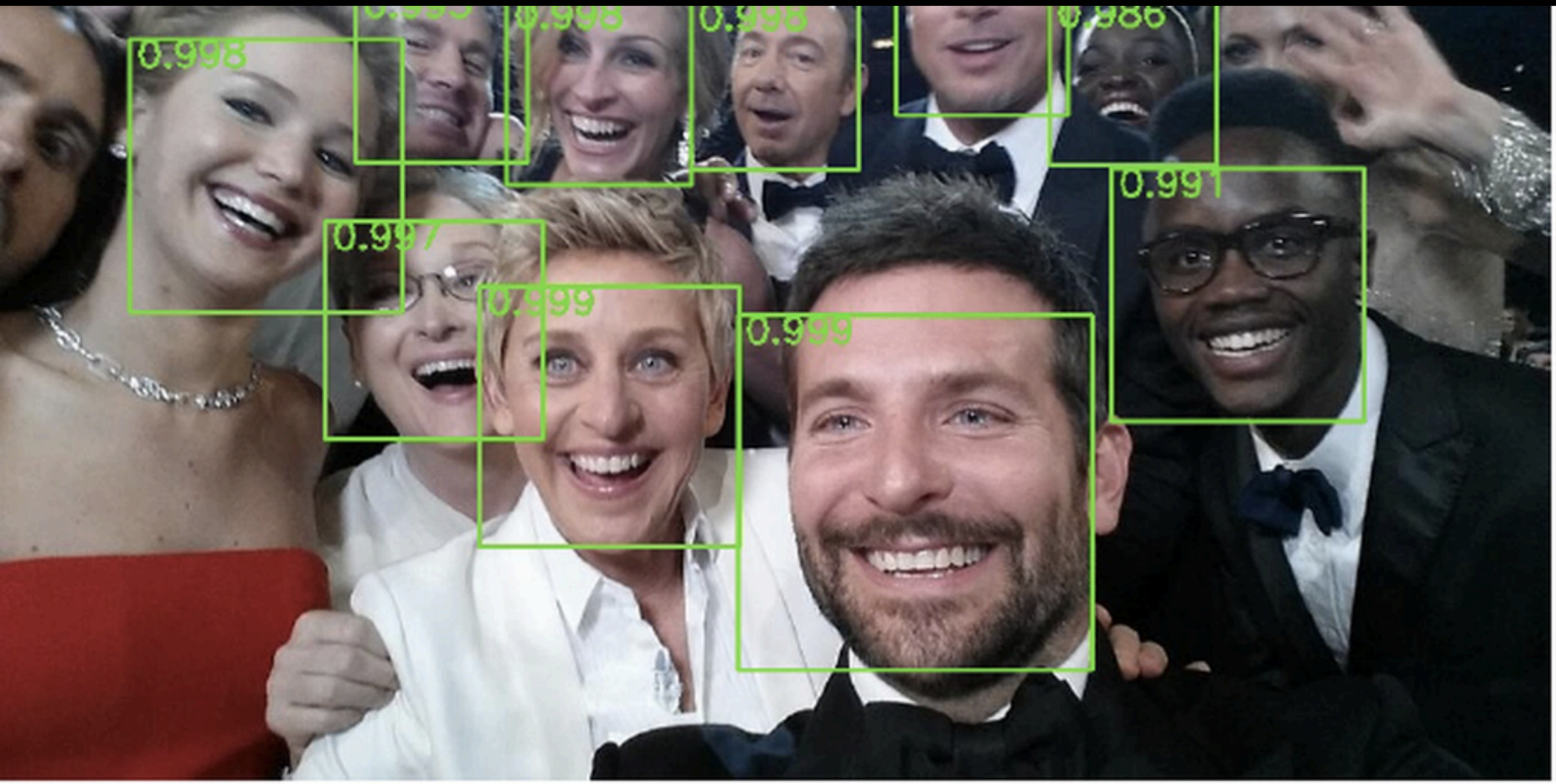


HoG: Histogram of Gradients

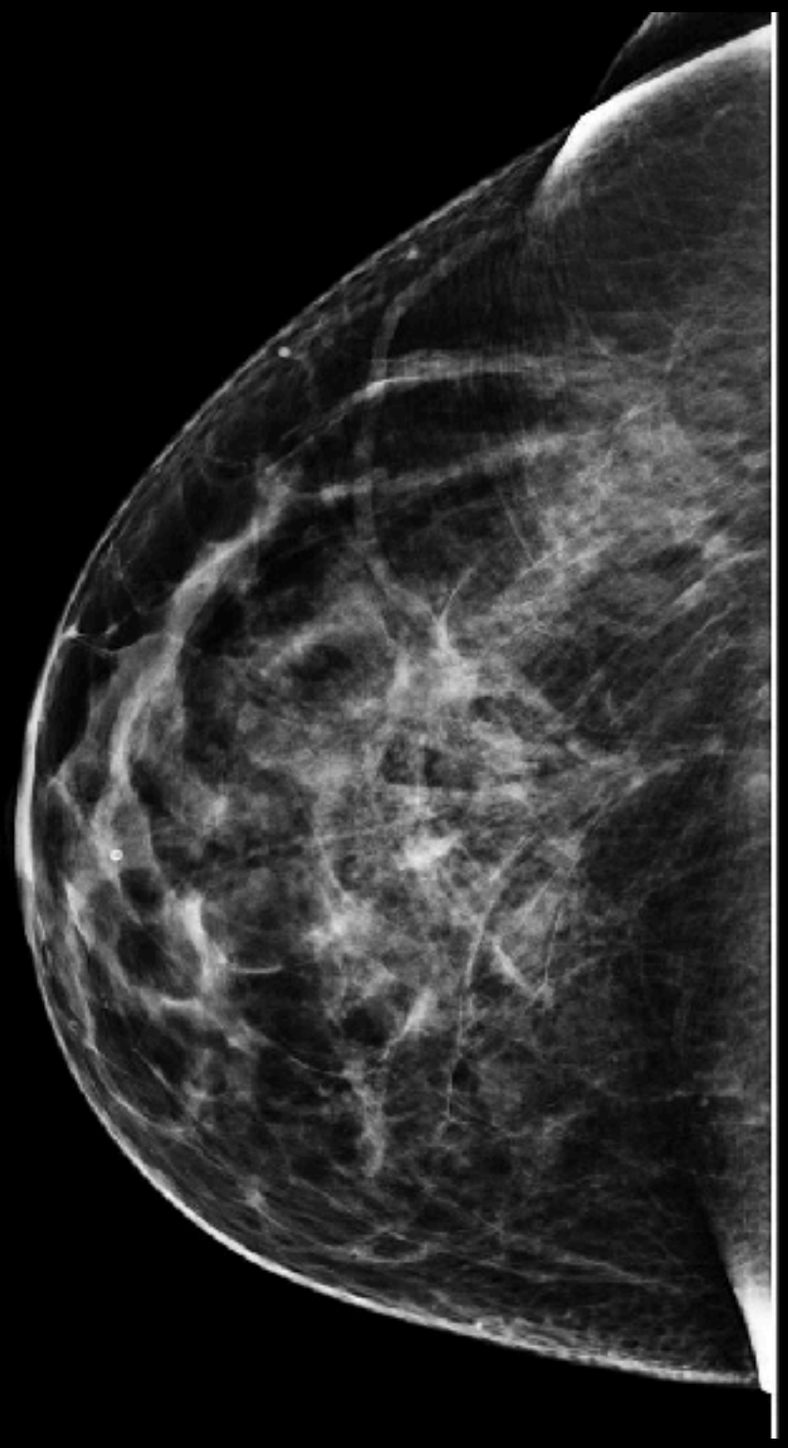
HoG: Histogram of Gradients






0.998
0.998
0.998
0.997
0.999
0.999
0.991

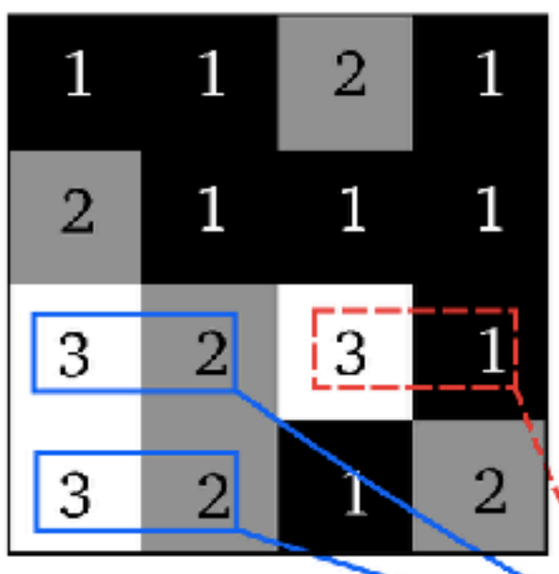
Descritores de Textura!



Densidade Mamária

Descritores de Textura! (Robert) Haralick

Image with numeric gray levels



Right neighbor GLCM

Reference pixel value (i) \ Neighbor pixel value (j)	1	2	3
1	3	2	0
2	3	0	1
3	1	2	0

Normalized GLCM p(i,j)

Reference pixel value (i) \ Neighbor pixel value (j)	1	2	3
1	0.25	0.17	0
2	0.25	0	0.08
3	0.08	0.17	0

Texture features

Homogeneity:

$$\sum_{ij} \frac{p(i,j)}{1+(i-j)^2}$$

Contrast:

$$\sum_{ij} p(i,j)(i-j)^2$$

NOT BAD

Precisamos de mais?

Limitações

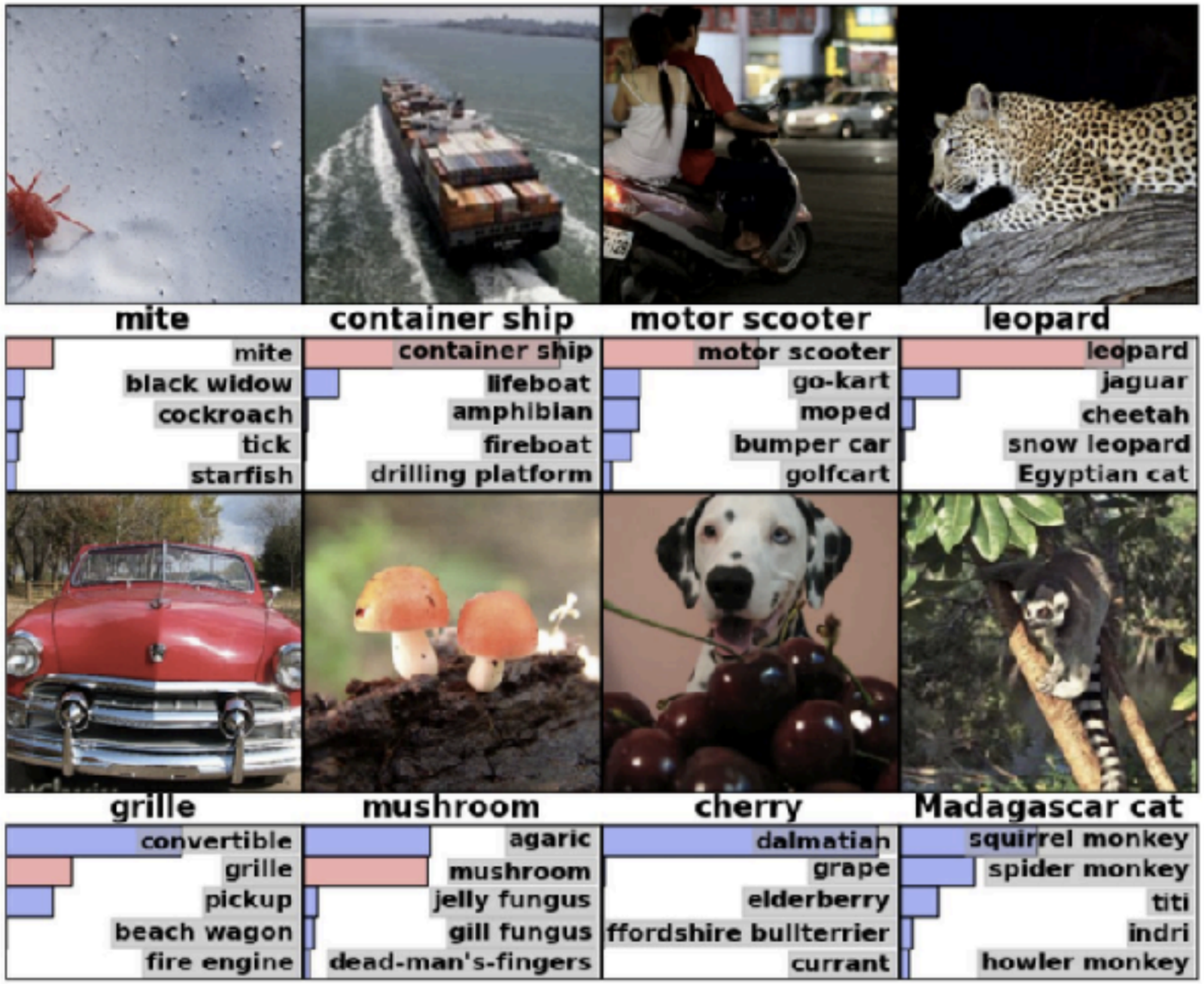
Como escolher as features?

Dependendo da feature,
mais exemplos ou overfitting.

...Hard Problems.

IMAGENET

- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.



Variância

Background Clutter

Iluminação

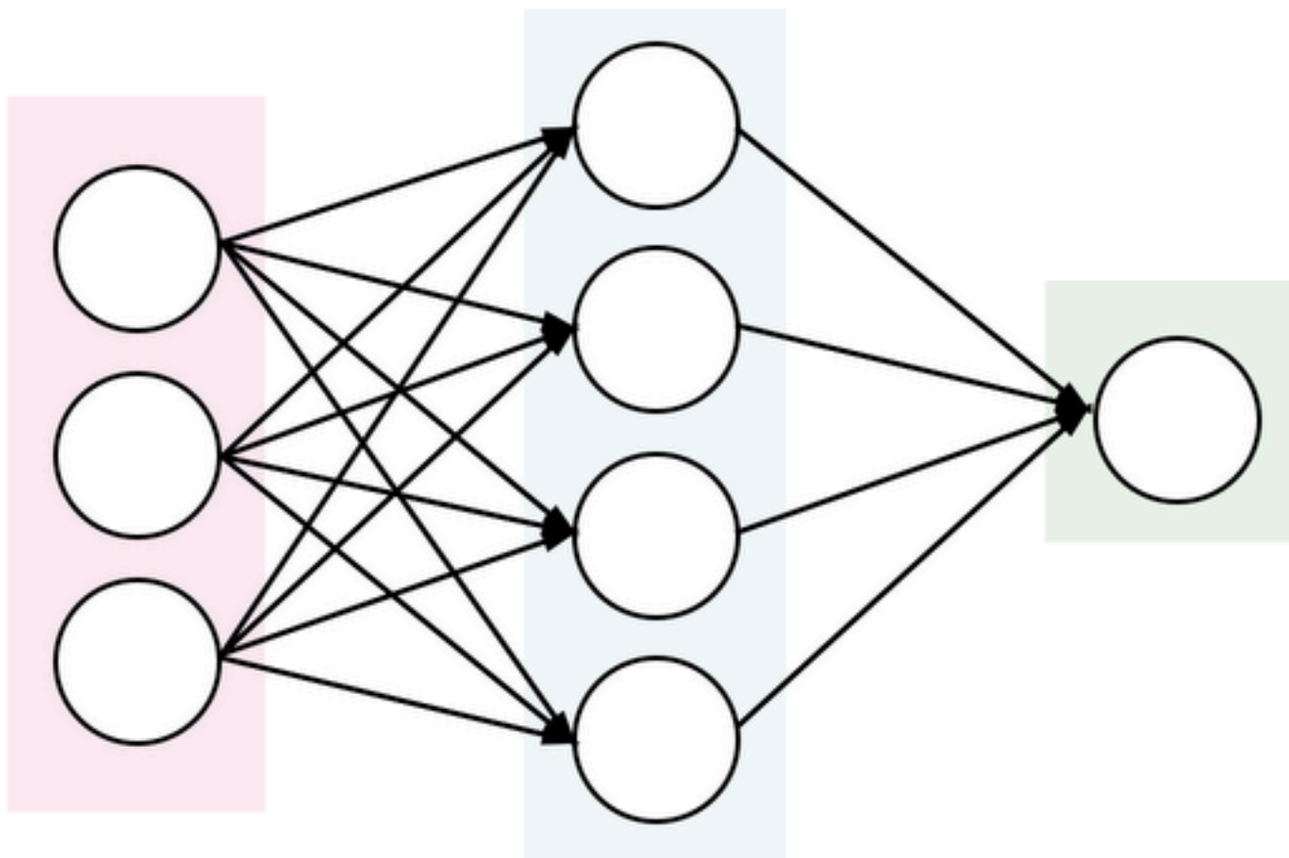
This image is CC0 1.0 public domain

This image is CC0 1.0 public domain

Deformação

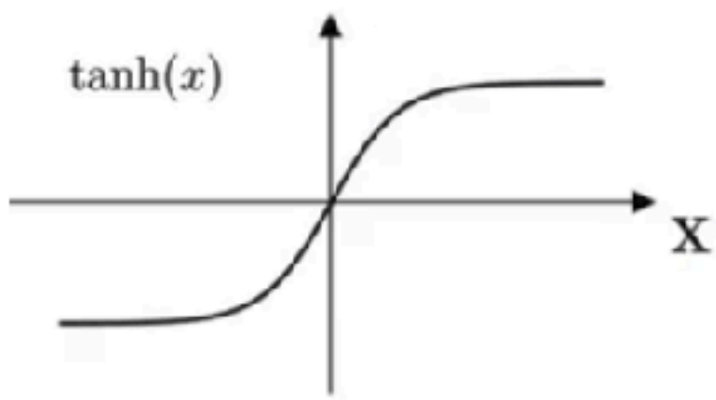
Oclusão

Redes Neurais
(fully connected)

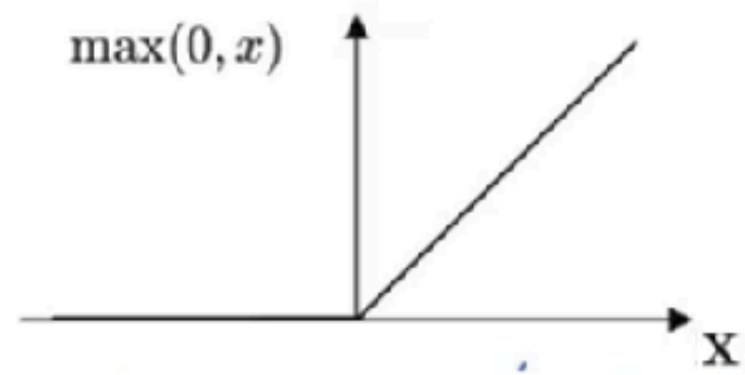


Redes Neurais
(funções de ativação)

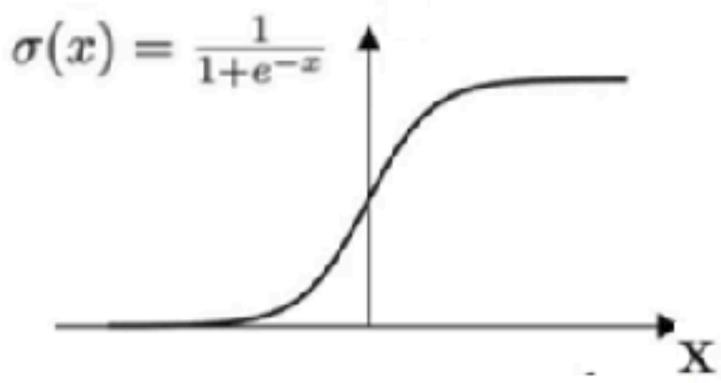
Hyper Tangent Function



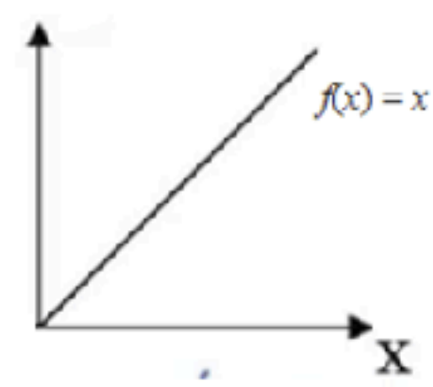
ReLU Function



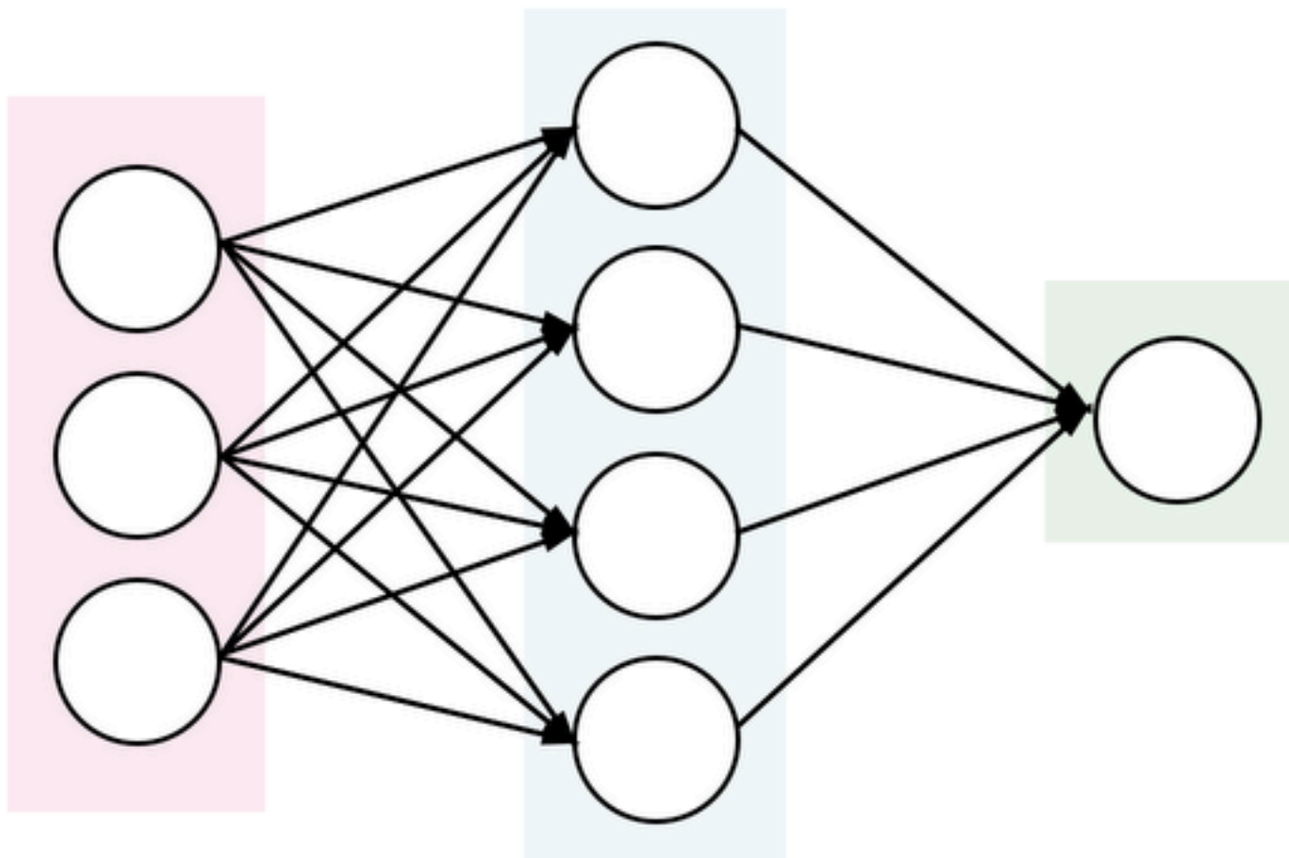
Sigmoid Function



Identity Function



Redes Neurais
(fully connected)



Redes Neurais
(backpropagation)

Derivative of Error with respect to weight

Old weight

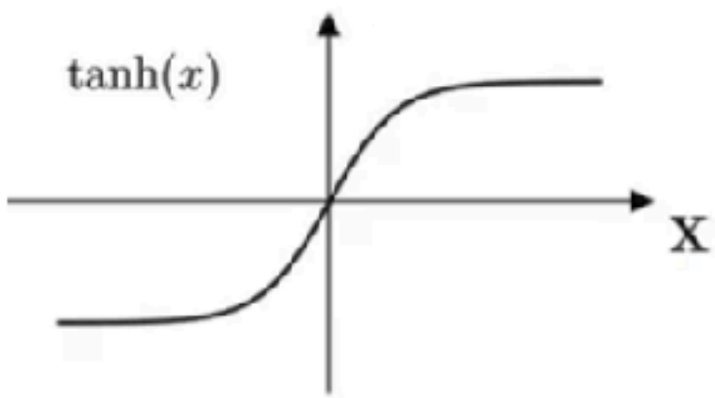
$$^{*}W_x = W_x - a\left(\frac{\partial Error}{\partial W_x}\right)$$

New weight

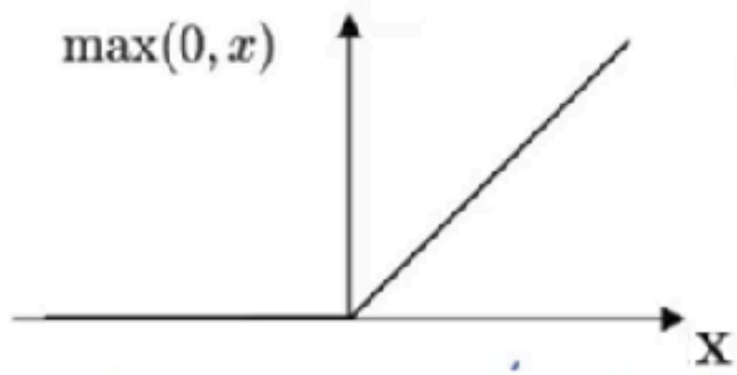
Learning rate

Redes Neurais
(funções de ativação)

Hyper Tangent Function

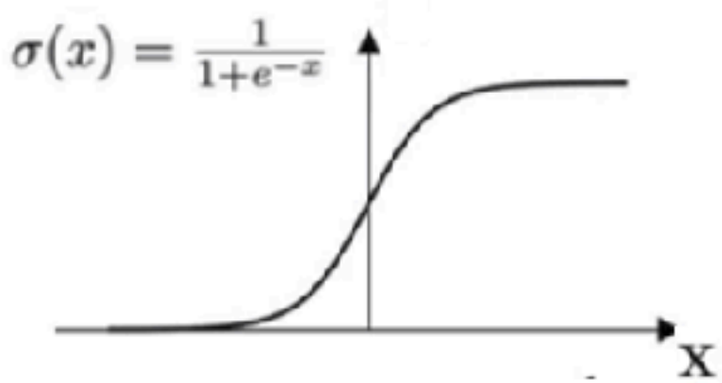


ReLU Function

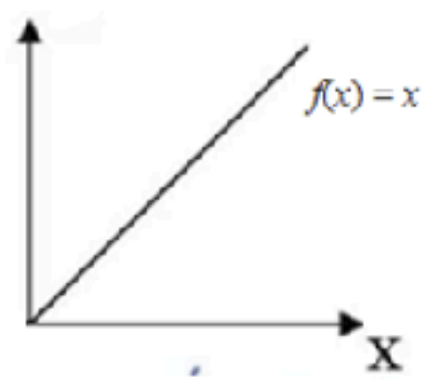


Última ativação de acordo com o problema a ser resolvido

Sigmoid Function

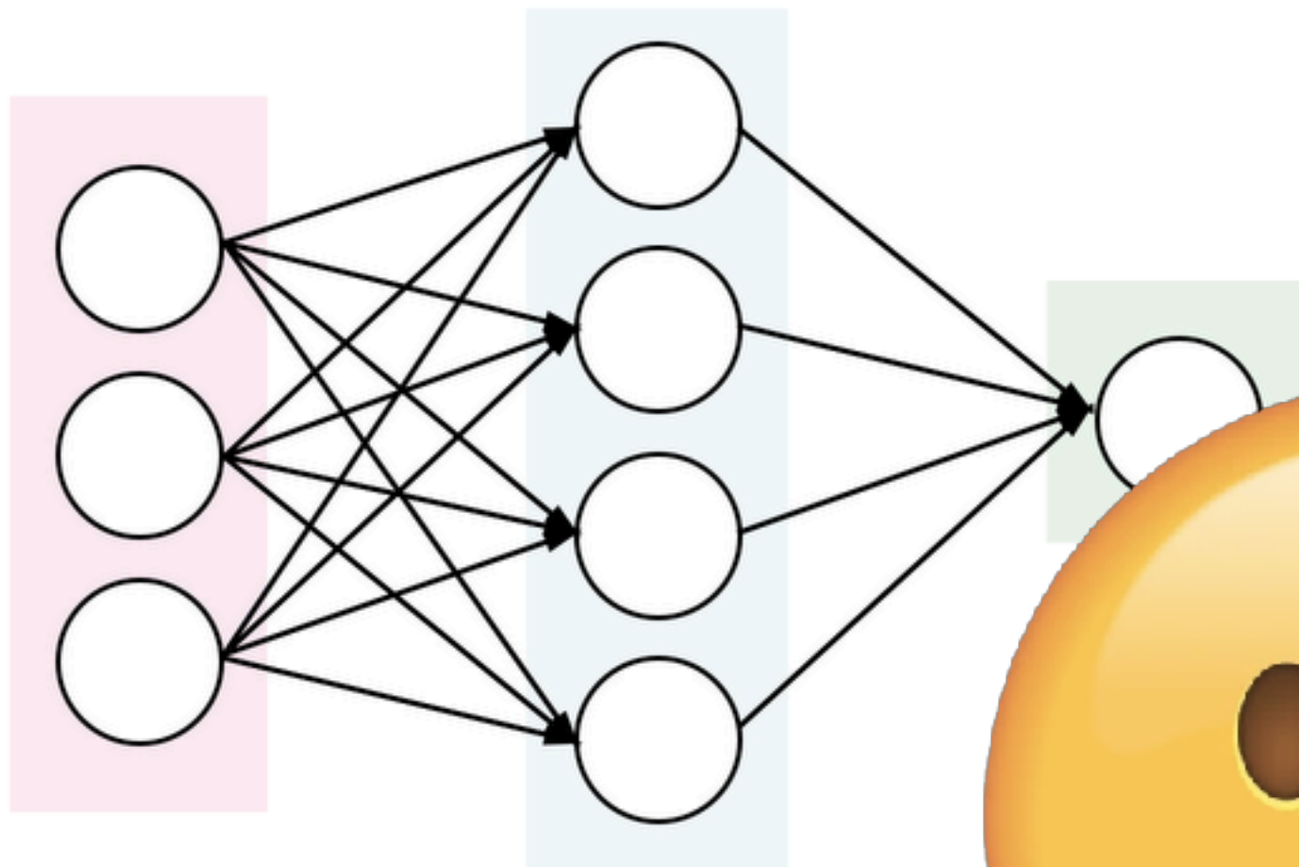


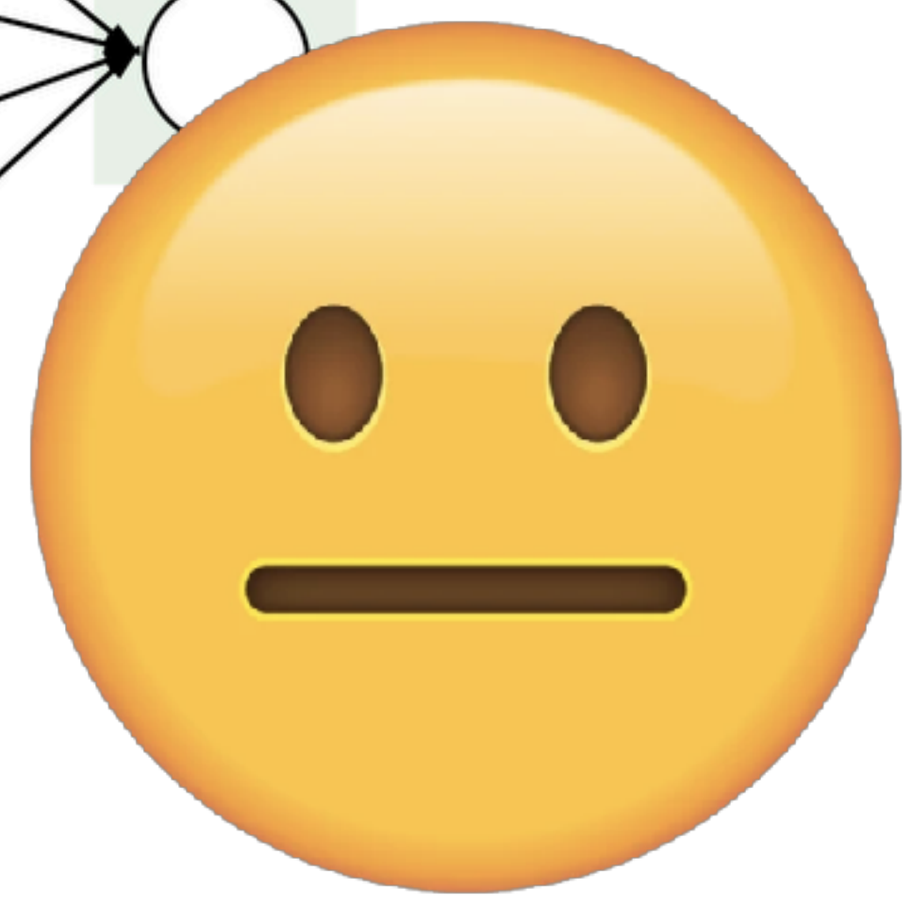
Identity Function



ReLU é muito usada
(Ajuda vanishing gradient)
(Mais rápida)
(Mas.. pode não propagar)

Redes Neurais
(funções de ativação)

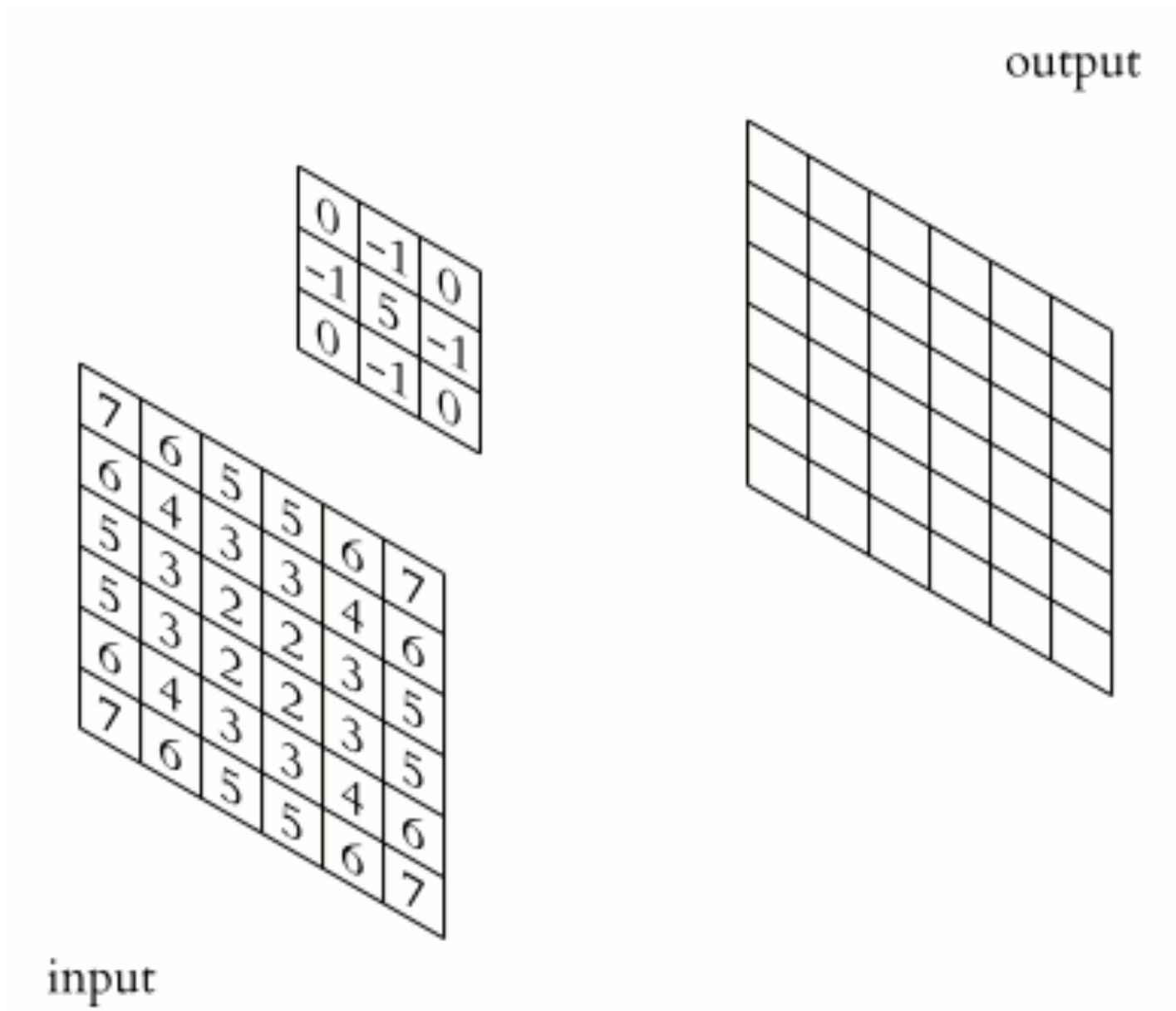




Muitos parâmetros

Não aproveita informações estruturais.

Convoluções e Filtros!



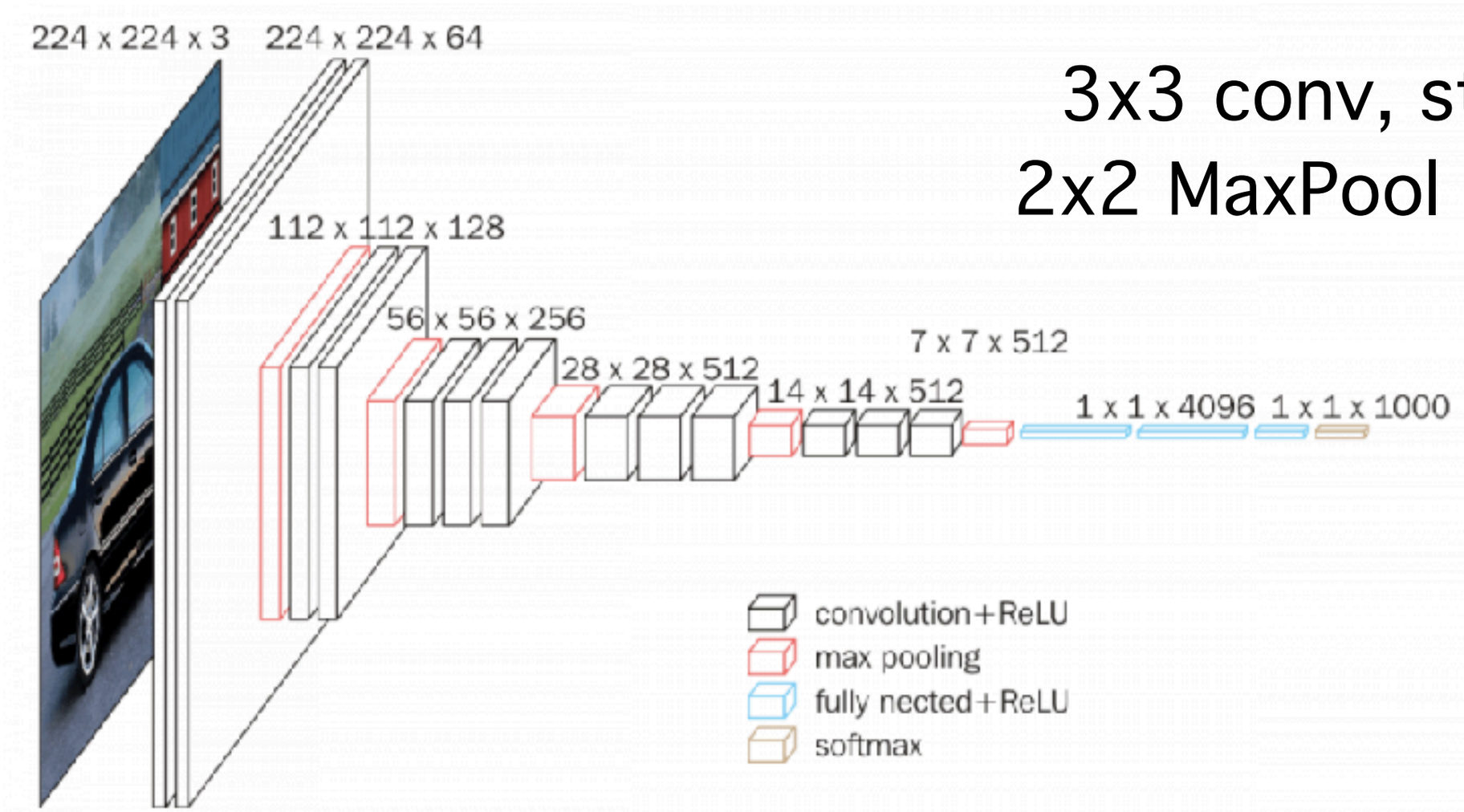
Agora podemos fazer convoluções com os pesos de neurônios.

(Convolutional Neural Networks)

CNN - Classificação

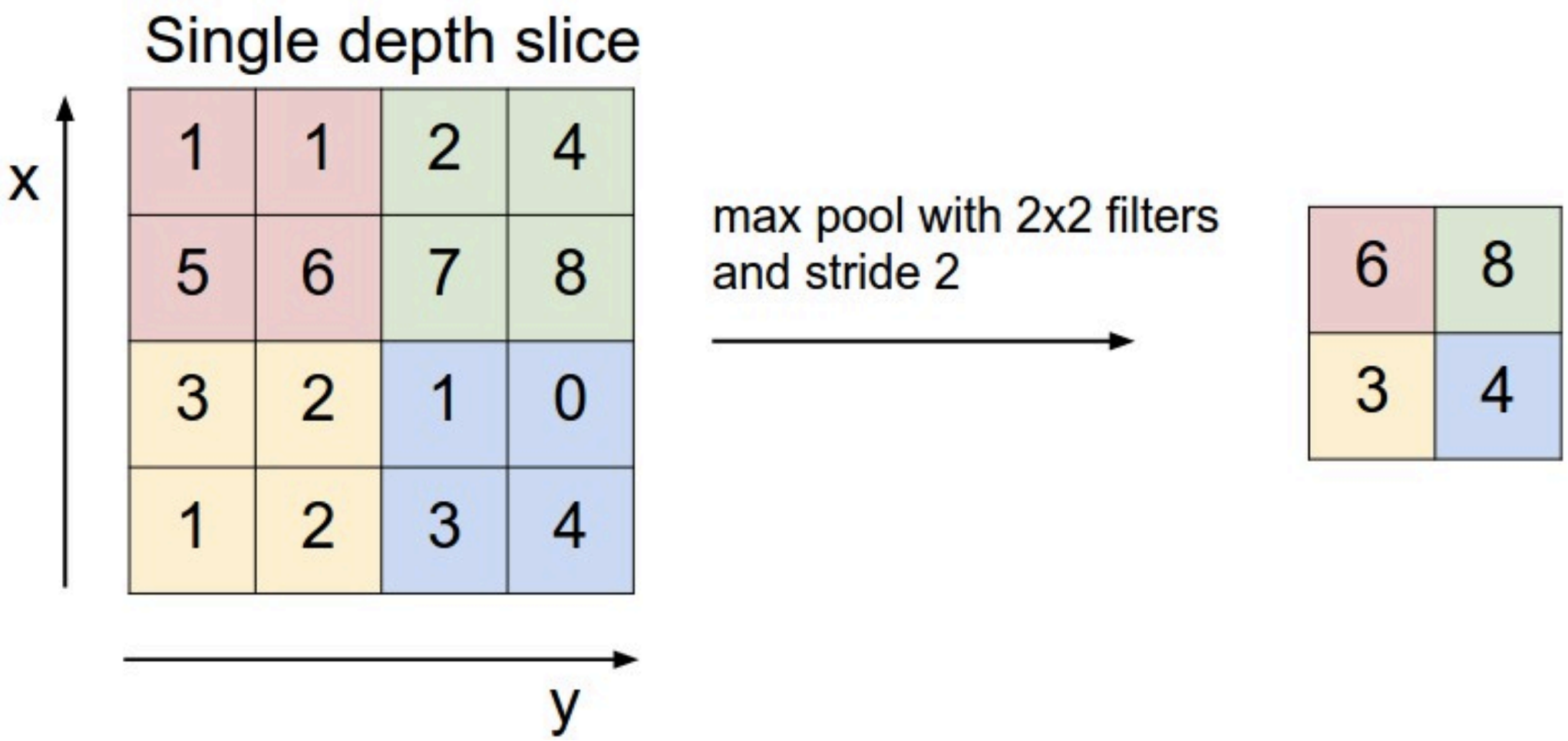
3x3 conv, stride 1

2x2 MaxPool stride 2



VGG-16

Pooling



Não possui pesos ajustáveis (parâmetros)

Redução de Dimensionalidade

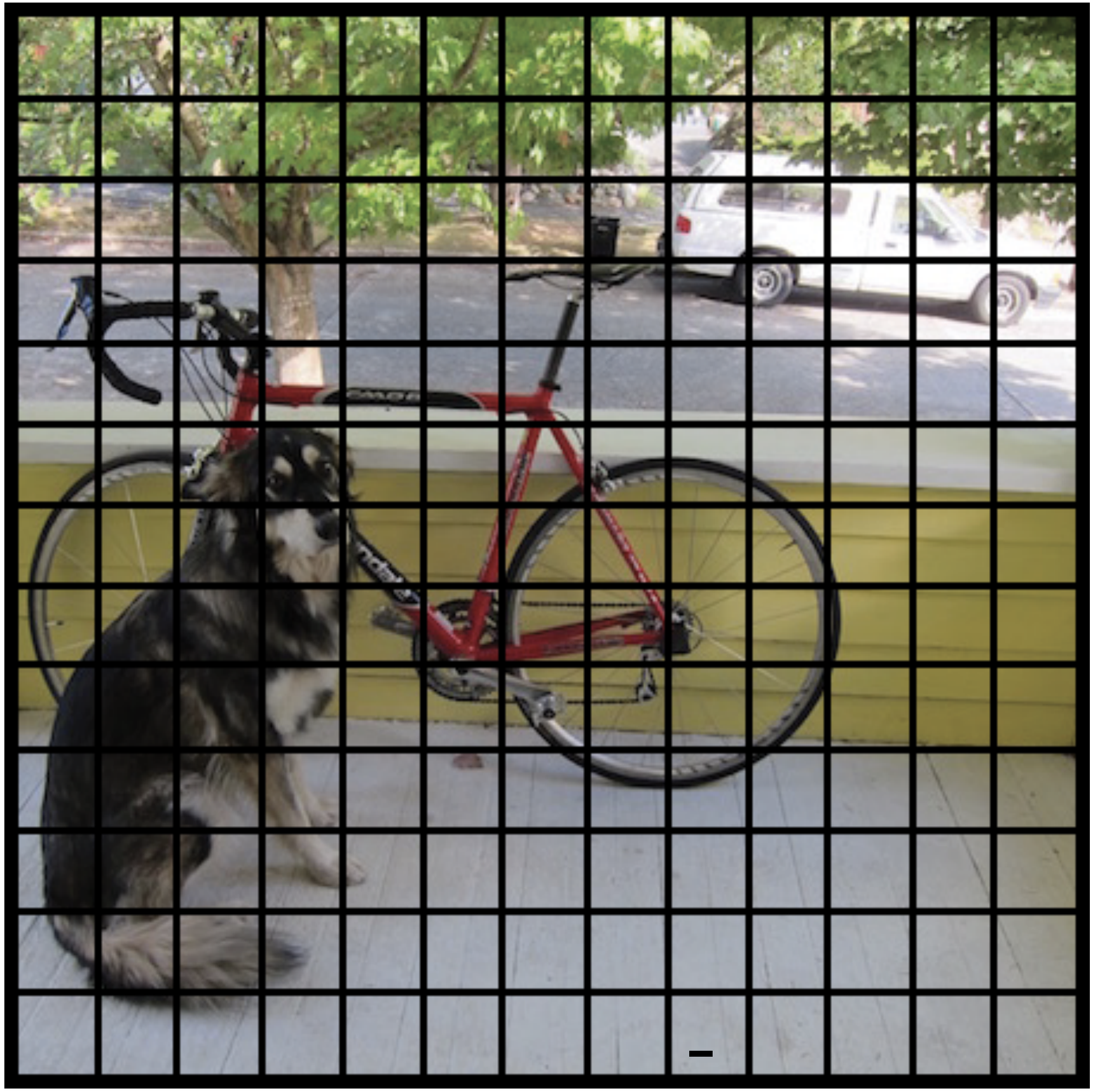
Deixa mais robusto a rotação e translação.

Max é utilizado pois as features
mais importantes devem ativar.

CLASSIF

Oq mais?

traffic light.96
traffic light.95
traffic light.92
truck1.00
person1.00
person.87
car1.00
car.99
car.98
car.92
person.90
car.93
motorcycle.95
traffic light.84
person1.00
person.85
person.99
person.96
BUS



YOLO

Grid

Output

B boxes:
Anchor (x,y),
w, h, (altura, prof)
confidence

Class:
1 por classe

YOLO

Grid

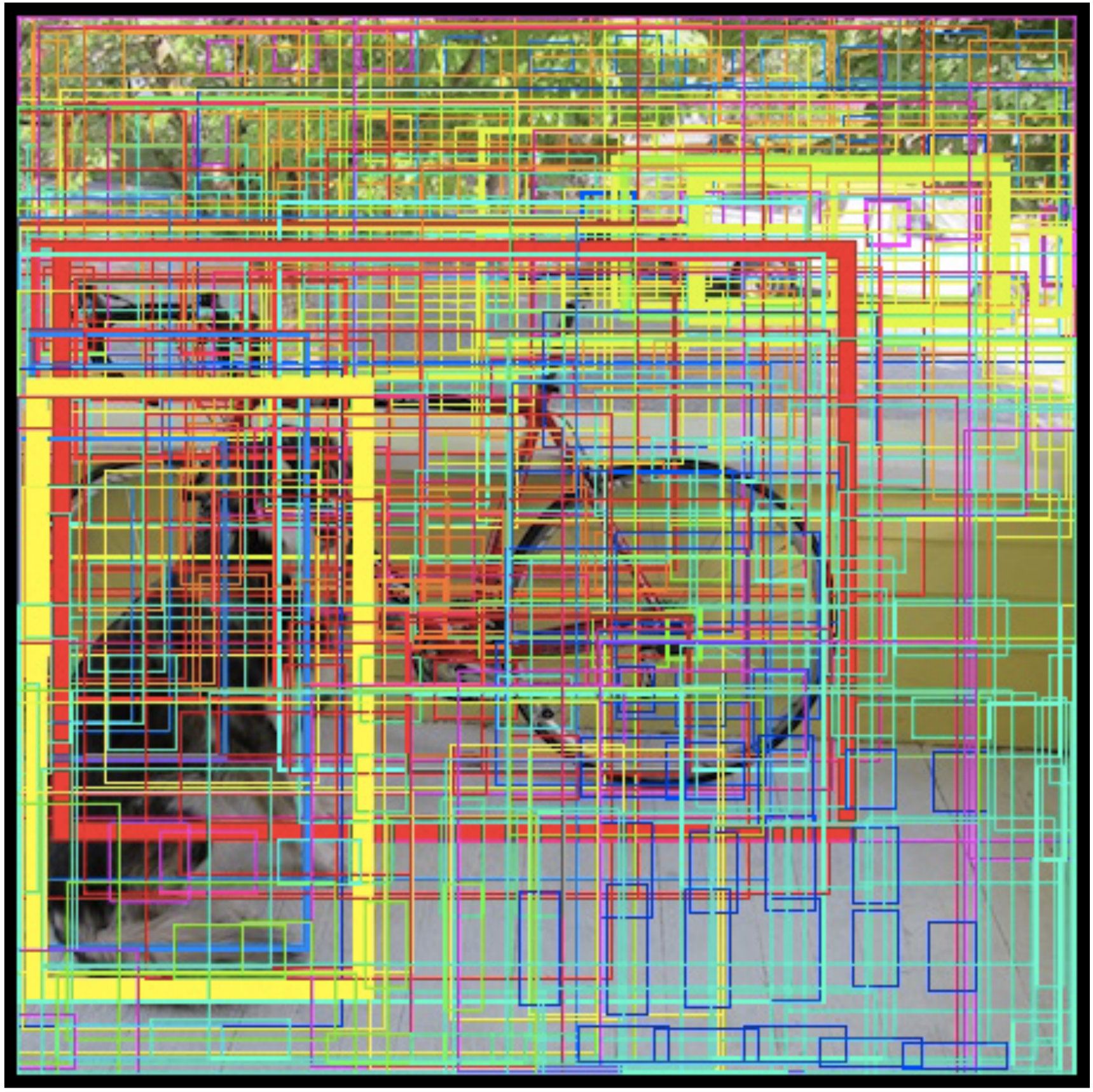
Output

B boxes:

Anchor (x,y),
w, h, (altura, prof)
confidence

Class:

1 por classe



YOLO

Grid

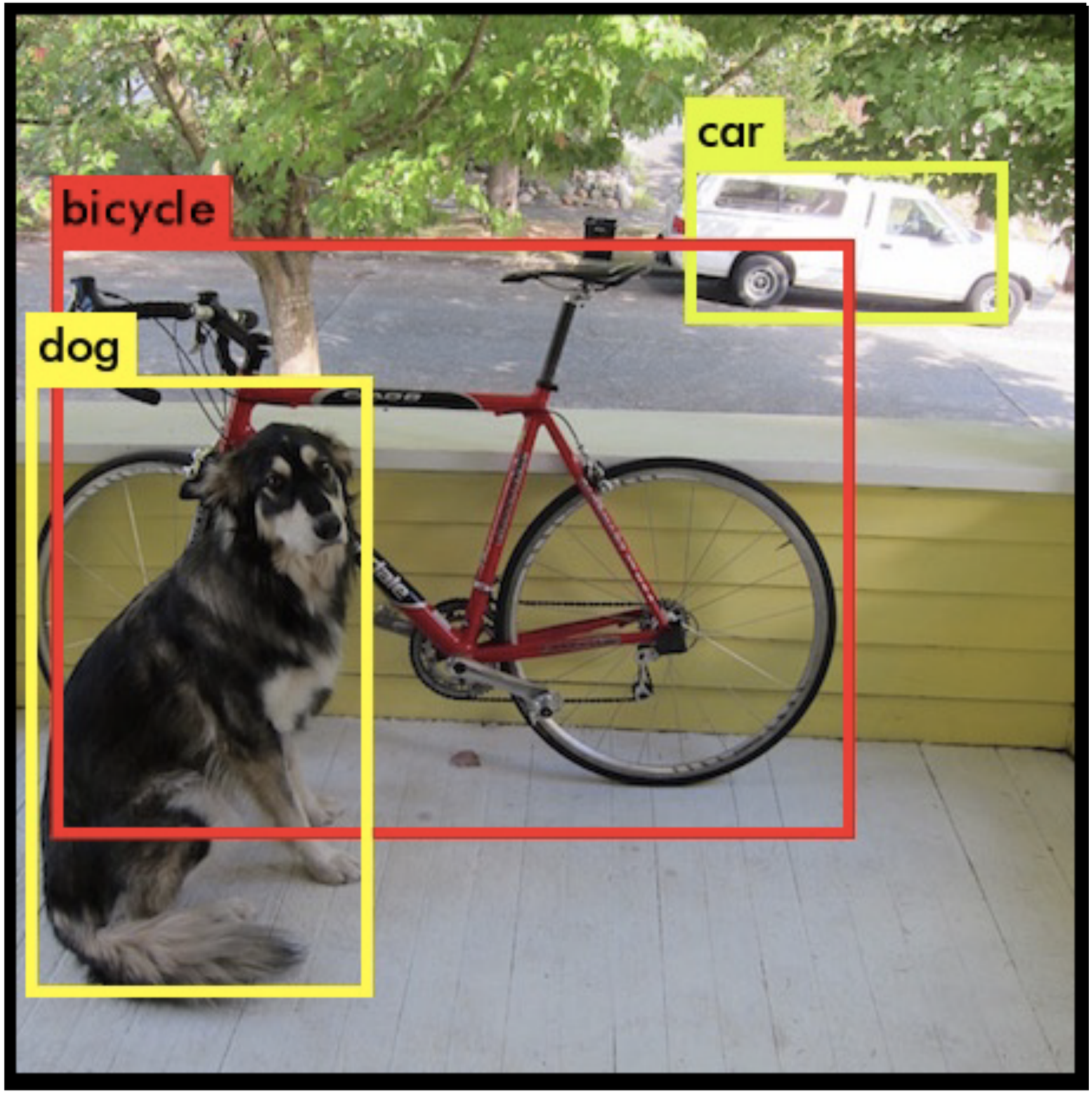
Output

B boxes:

Anchor (x,y),
w, h, (altura, prof)
confidence

Class:

1 por classe



YOLO

Grid

Output

B boxes:
Anchor (x,y),
w, h, (altura, prof)
confidence

Class:
1 por classe

Layer	kernel	stride	output shape
Input			(416, 416, 3)
Convolution	3×3	1	(416, 416, 16)
MaxPooling	2×2	2	(208, 208, 16)
Convolution	3×3	1	(208, 208, 32)
MaxPooling	2×2	2	(104, 104, 32)
Convolution	3×3	1	(104, 104, 64)
MaxPooling	2×2	2	(52, 52, 64)
Convolution	3×3	1	(52, 52, 128)
MaxPooling	2×2	2	(26, 26, 128)
Convolution	3×3	1	(26, 26, 256)
MaxPooling	2×2	2	(13, 13, 256)
Convolution	3×3	1	(13, 13, 512)
MaxPooling	2×2	1	(13, 13, 512)
Convolution	3×3	1	(13, 13, 1024)
Convolution	3×3	1	(13, 13, 1024)
Convolution	1×1	1	(13, 13, 125)

YOLO

Grid

Output

B boxes:
Anchor (x,y),
w, h, (altura, prof)
confidence

Class:
1 por classe

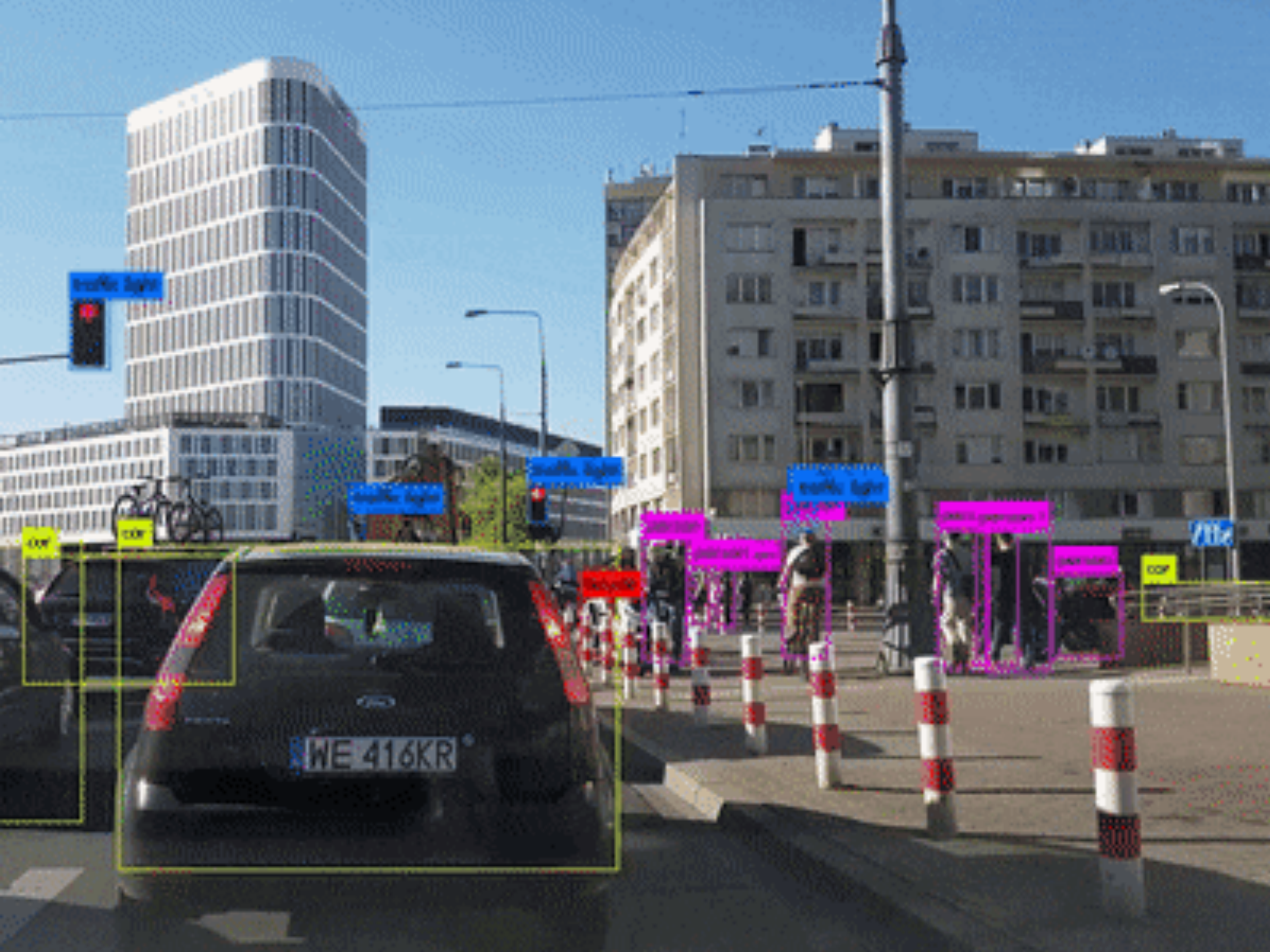
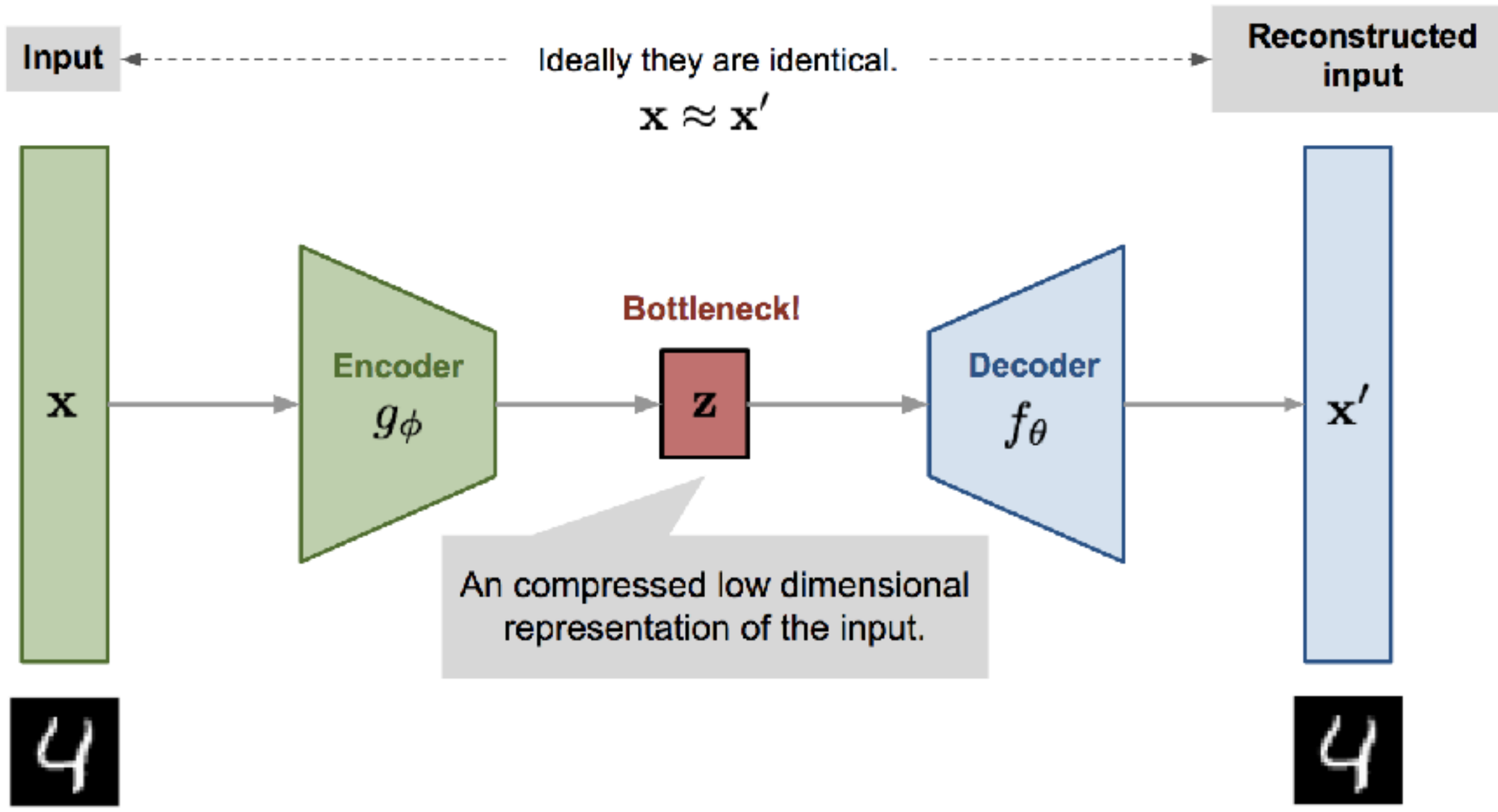
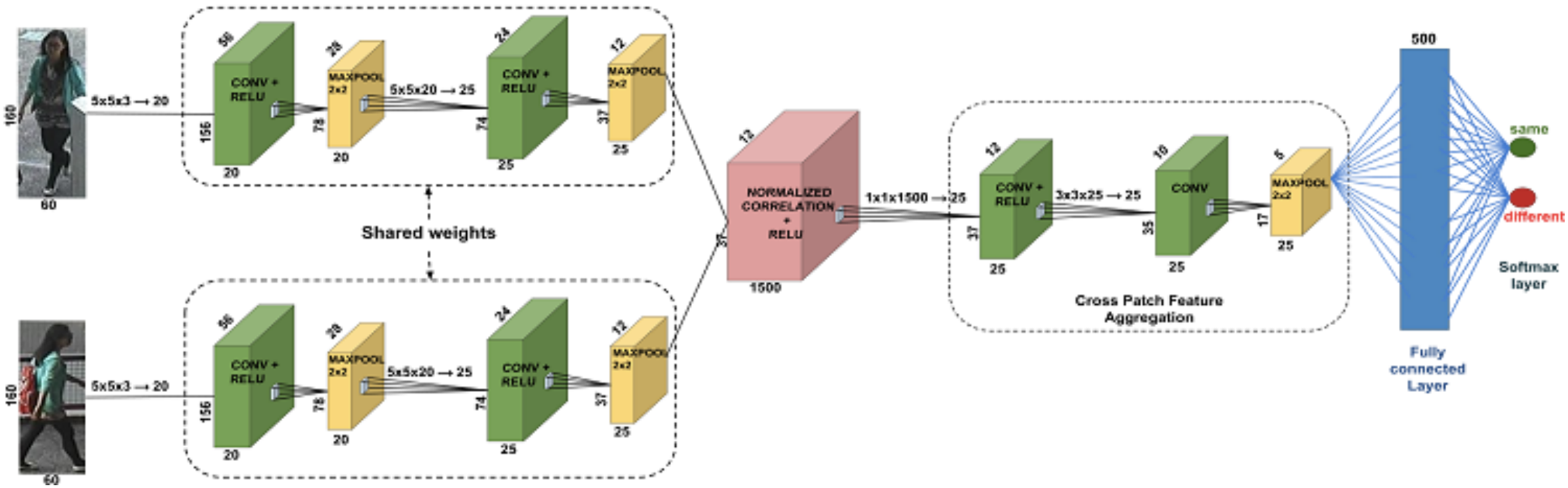

traffic light
traffic light
traffic light
traffic light
car
car
car
WE 416KR

Image-to-Image
(AutoEncoder)



Redes Siamesas

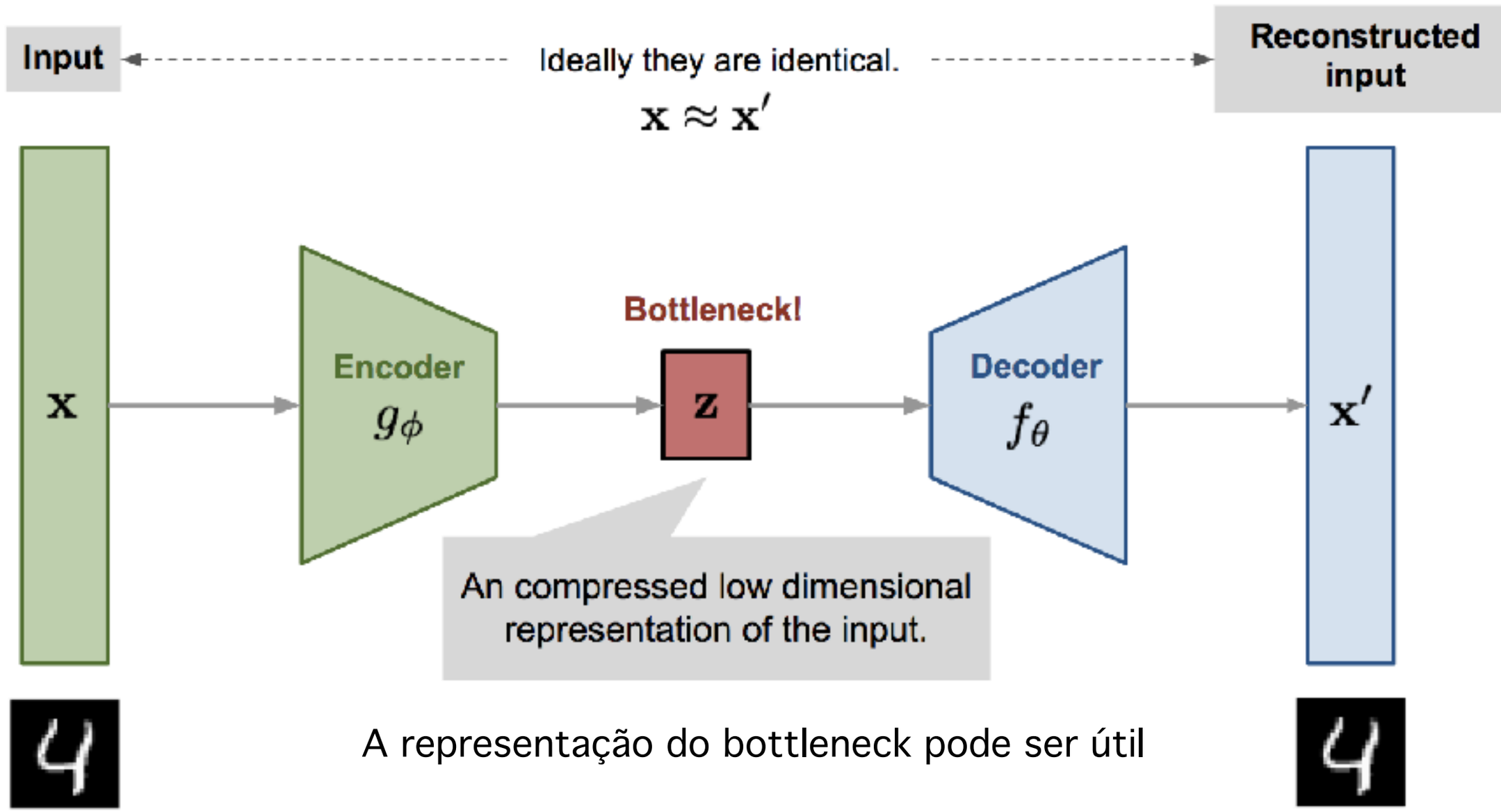


Possui shared weights

Utiliza normalized correlation para unir o output

Muito utilizado para comparação de imagens

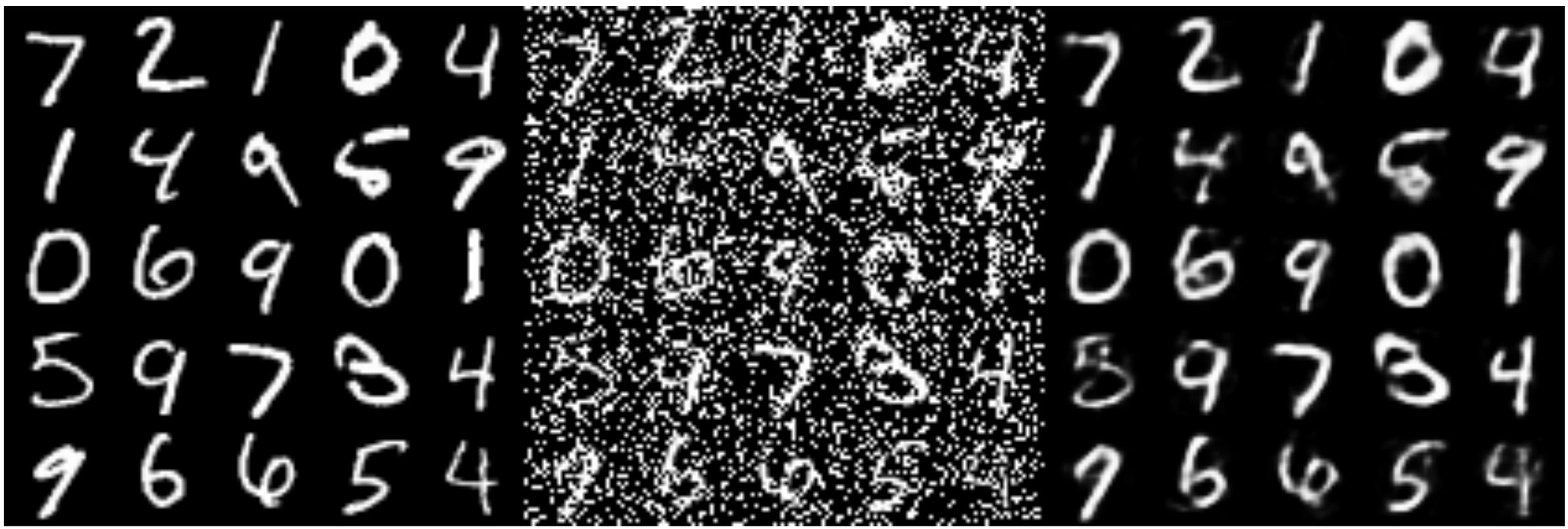
Image-to-Image (AutoEncoder)



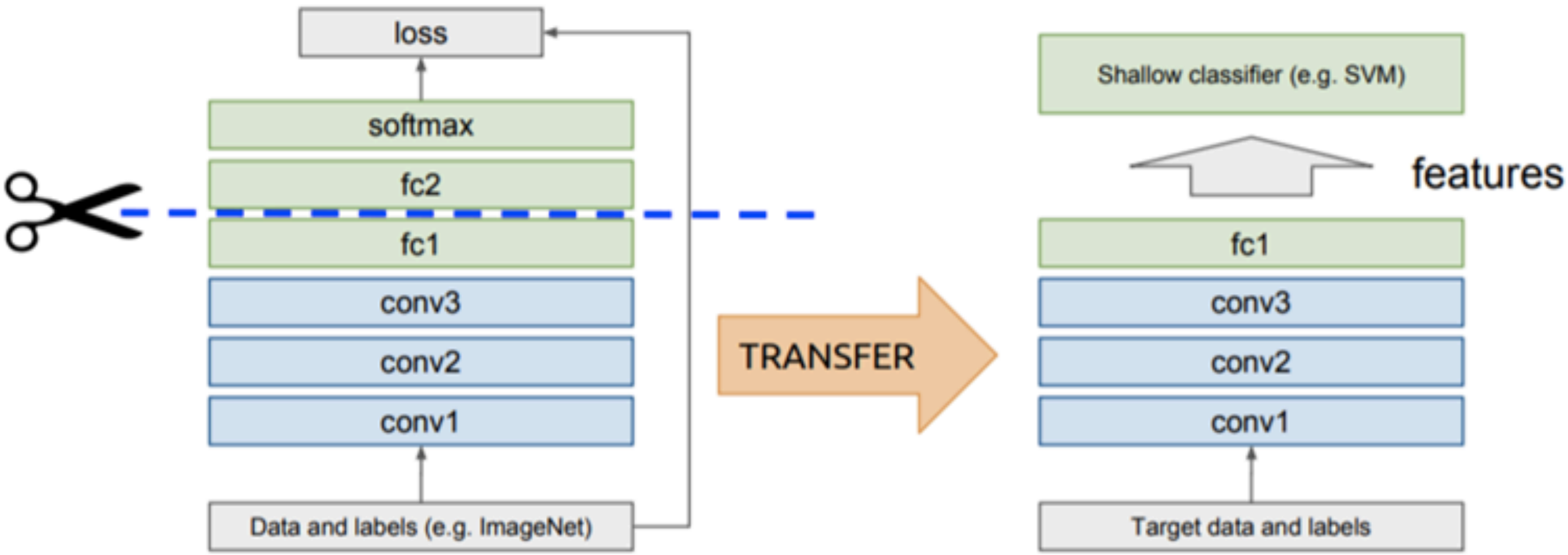
A representação do bottleneck pode ser útil

Podemos usar a mesma arquitetura para outros problemas, como denoising.

Image-to-Image (AutoEncoder)



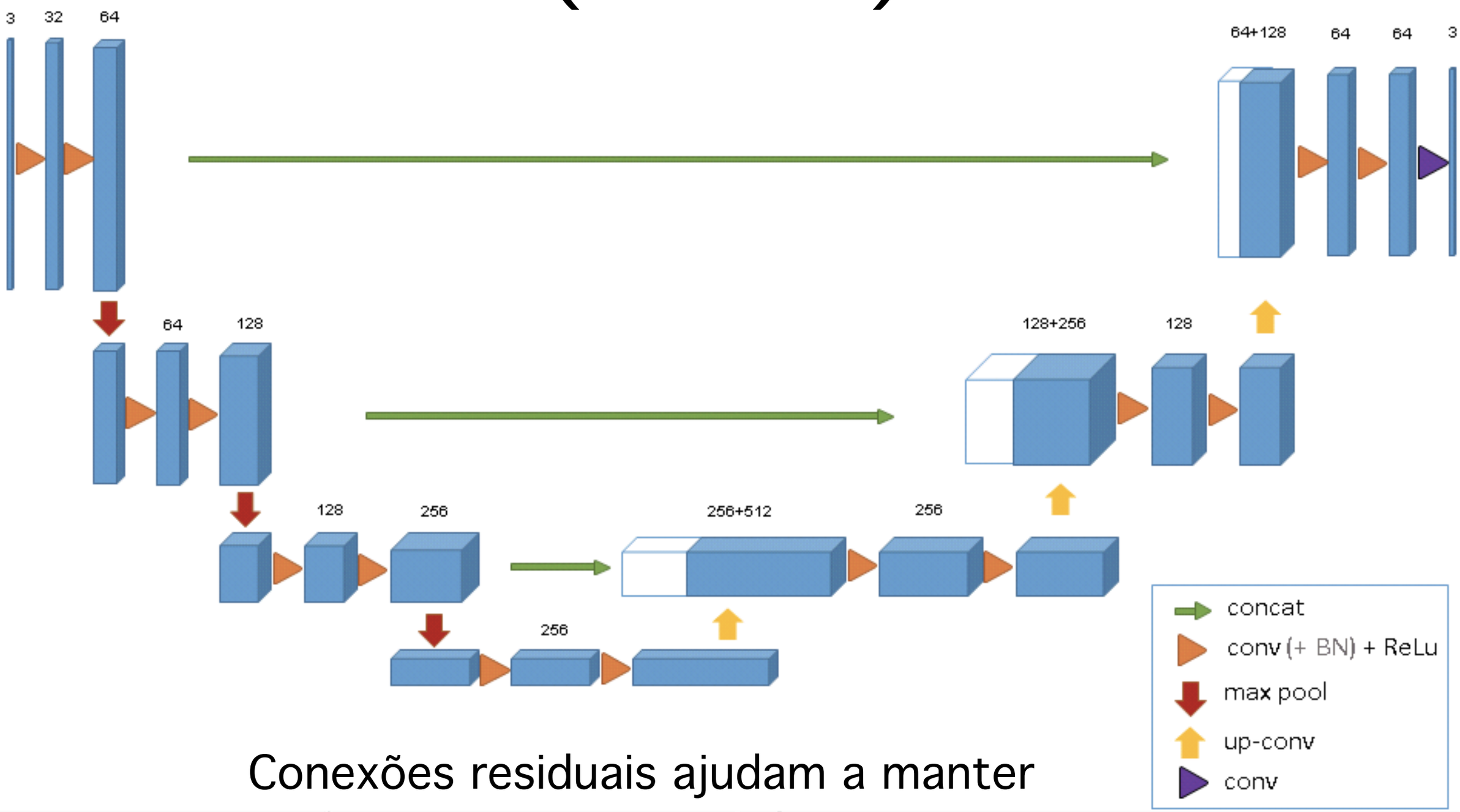
Transfer Learning



Reaproveitar os pesos de outras redes

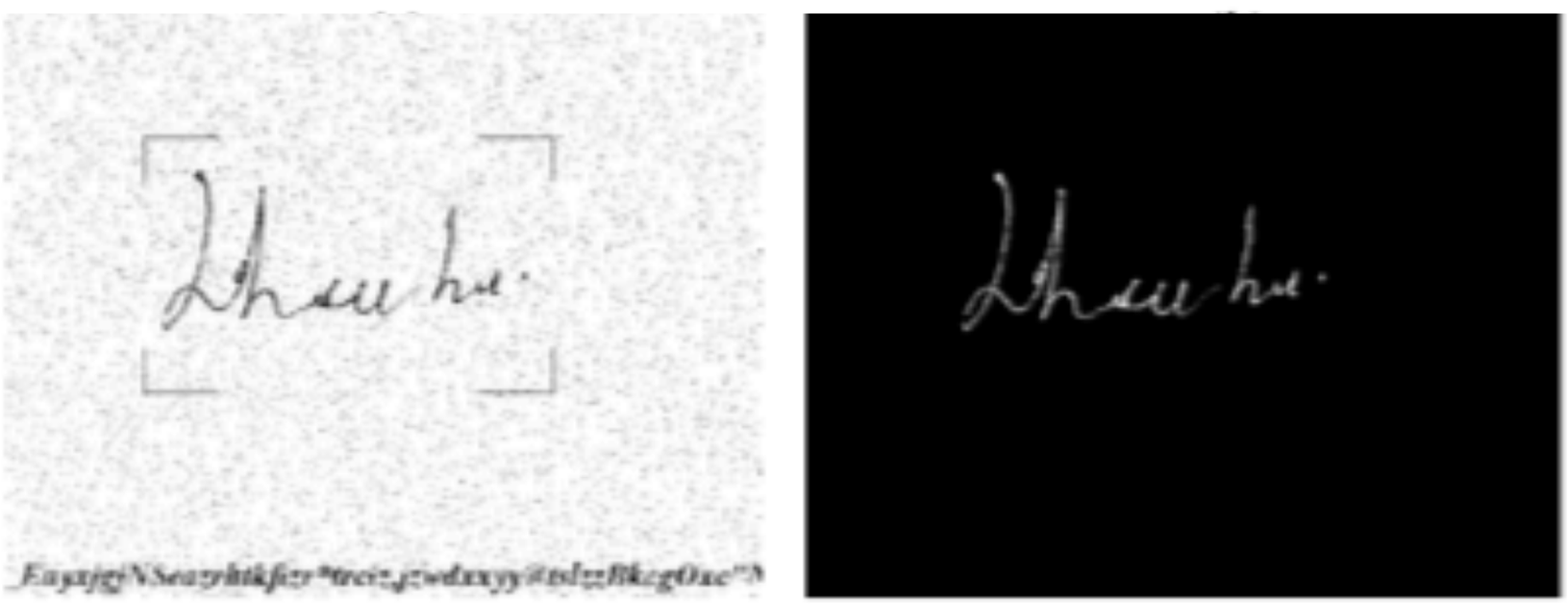
Cortar na parte convolucional permite certo
grande invariância a escala

Image-to-Image (U-Net)



Conexões residuais ajudam a manter informação estrutural da imagem.

Image-to-Image (U-Net)

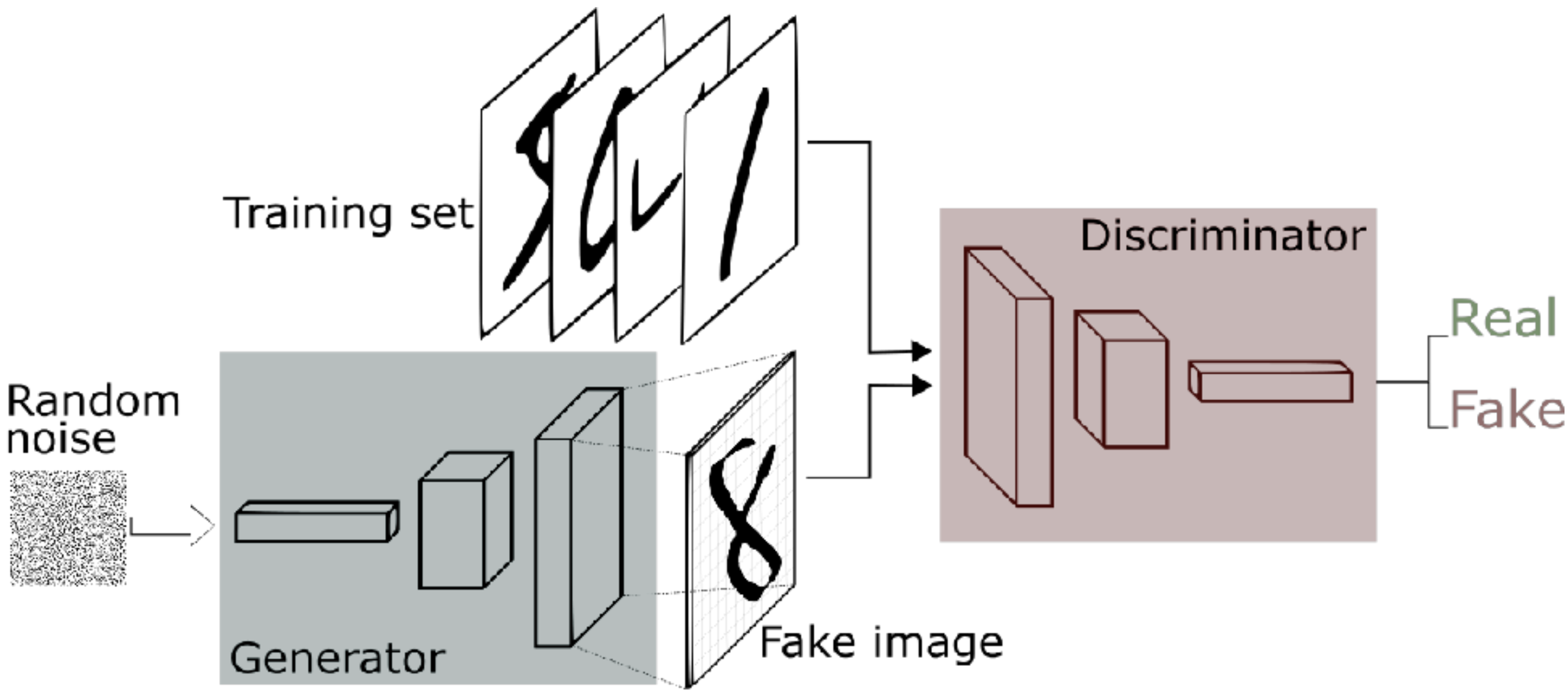


Aplicação no banco:
Separar texto de máquina de texto a mão e vice-versa

Image-to-Image (U-Net)

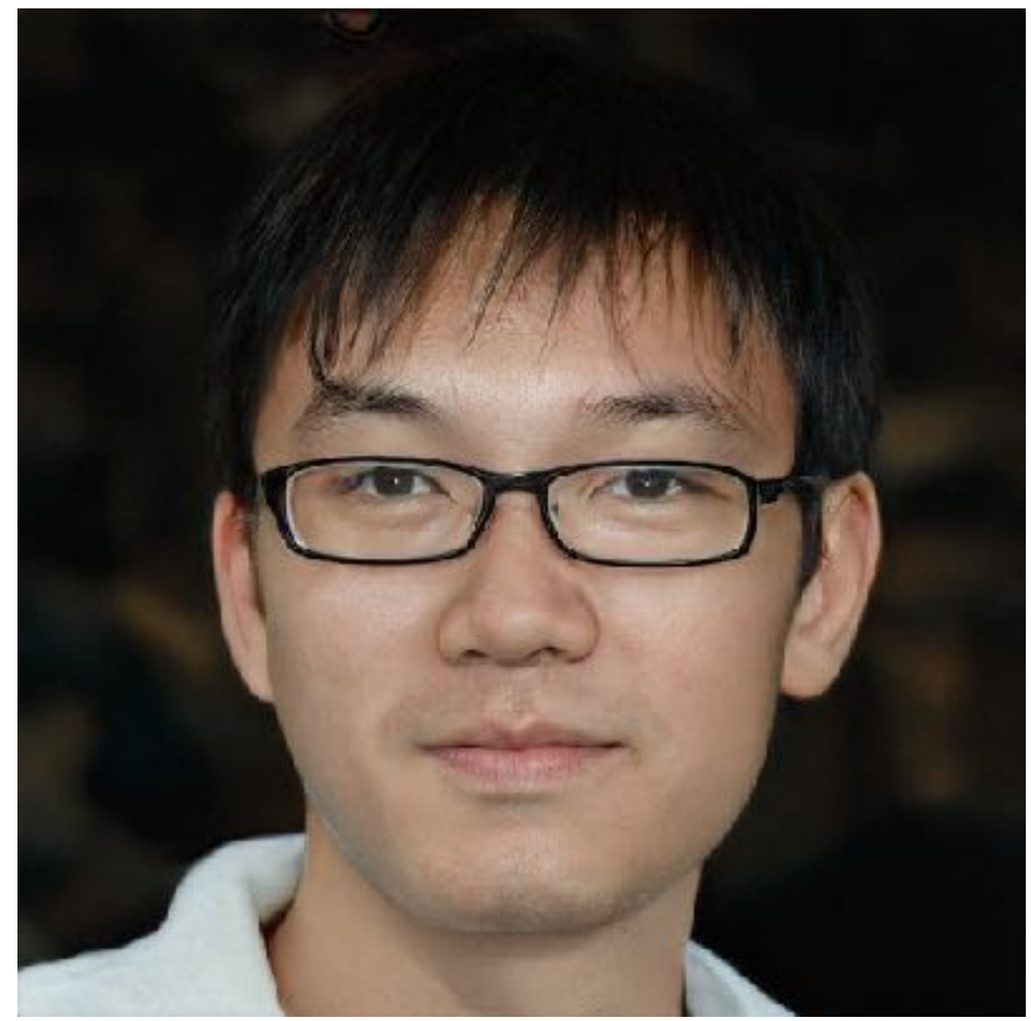
Segmentação semântica

Image-to-Image (GANs)



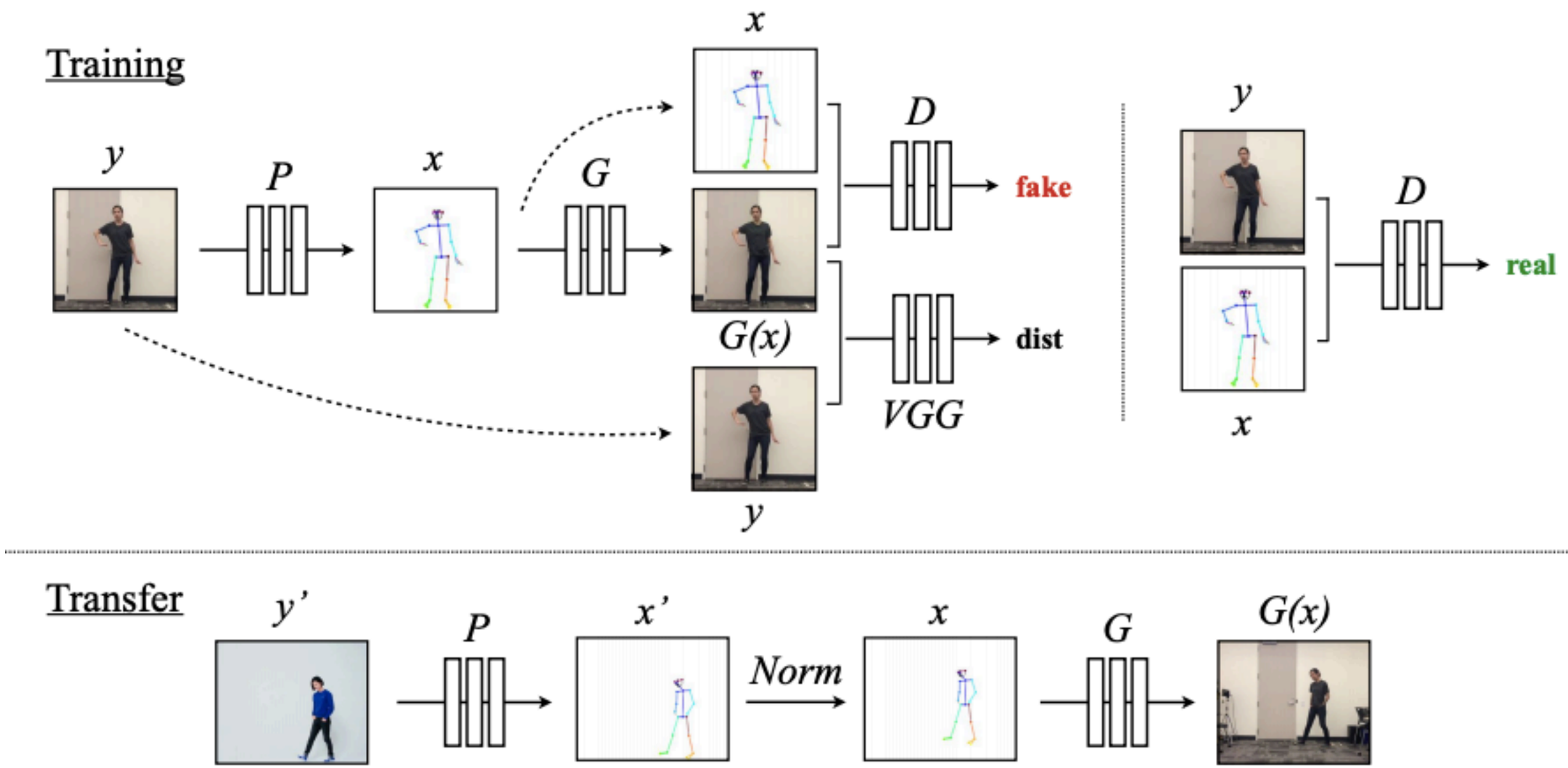
Duas redes competem (uma gera, a partir e ruído, a outra tenta ver se é real ou não)
É uma arquitetura que tende a ser instável, mas temos tricks para treinar.

Image-to-Image (GANs)



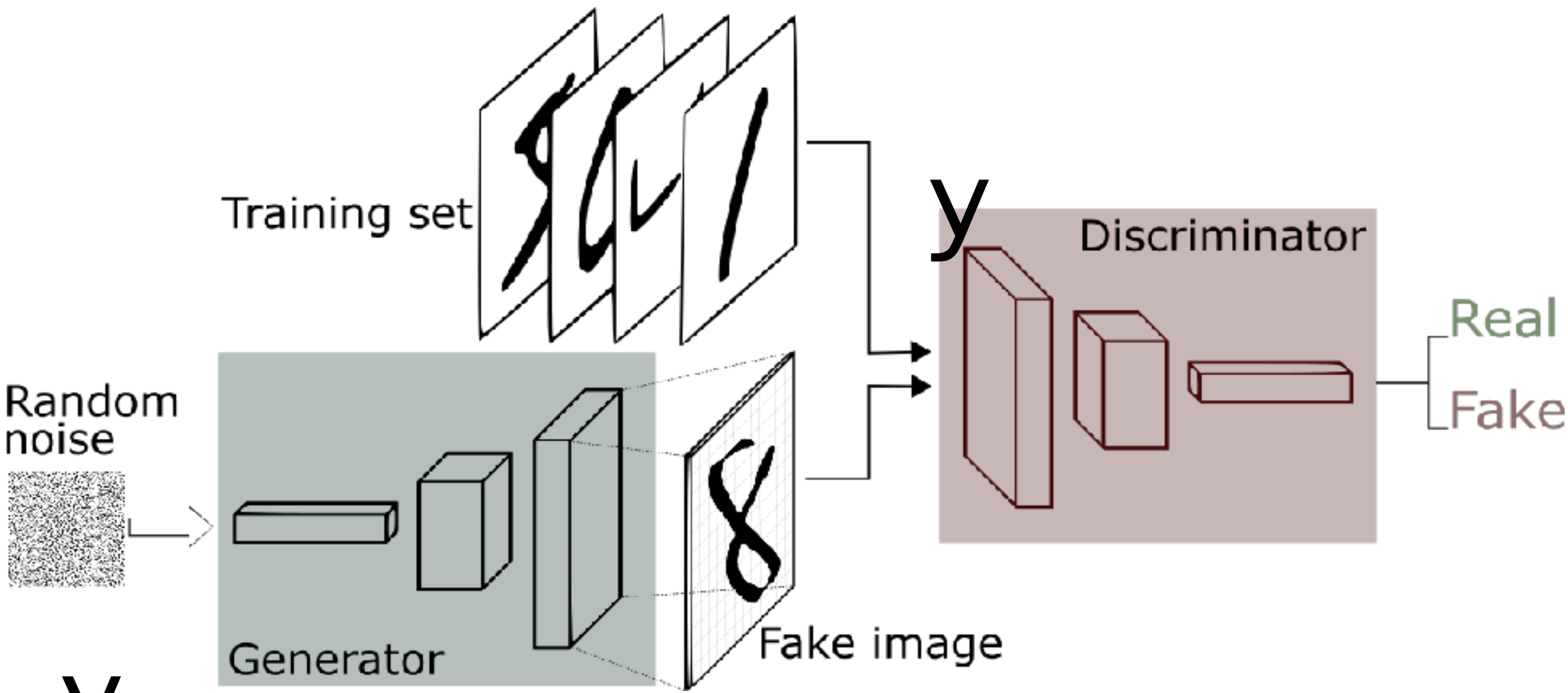
https://thispersondoesnotexist.com

Everybody Dance Now



1) Utiliza um pose detector para detectar as poses
2) Utiliza GAN para aprender a gerar uma única pessoa a partir da pose
3) Pega as poses de alguém que sabe dançar e utiliza para se gerar dançando :P

Image-to-Image (Conditional GANs)

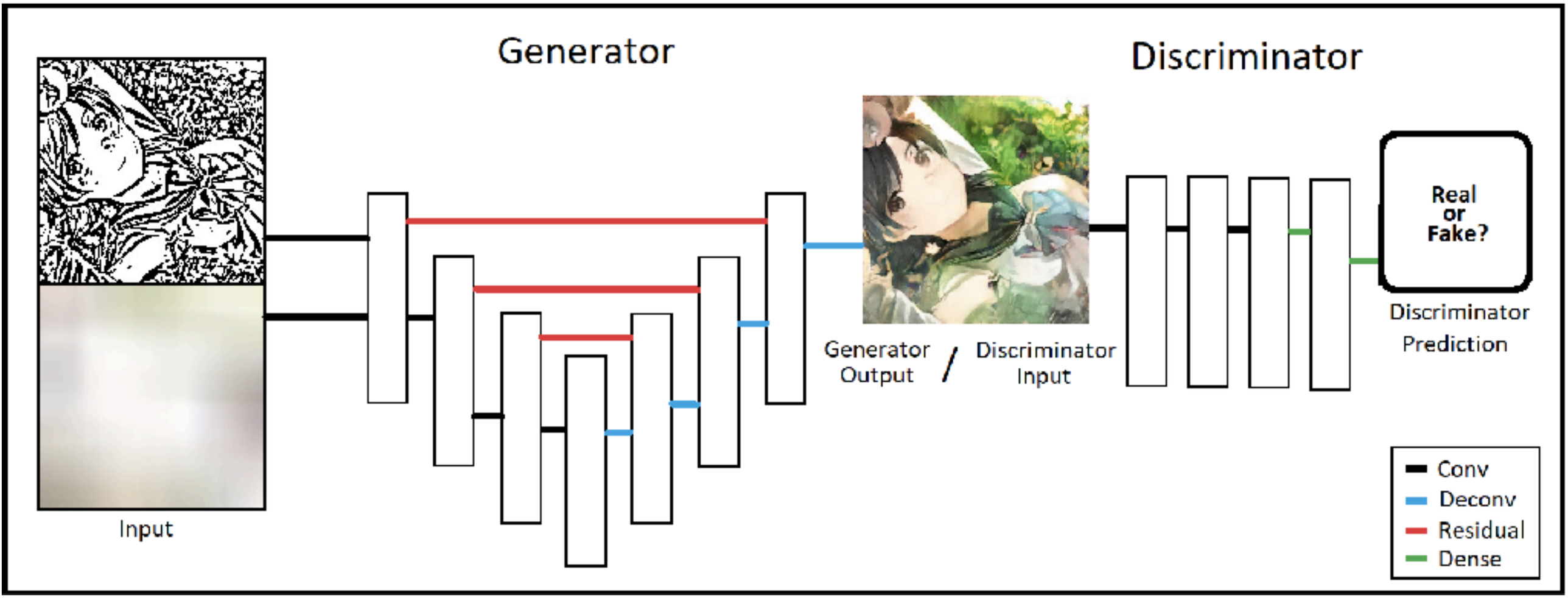


y

GANs são originalmente incondicionais (geram em cima de ruído apenas)

Podemos adicionar uma condição (y) que pode ser desde 0 para rostos femininos e 1 para rostos masculinos, até mesmo uma imagem de entrada.

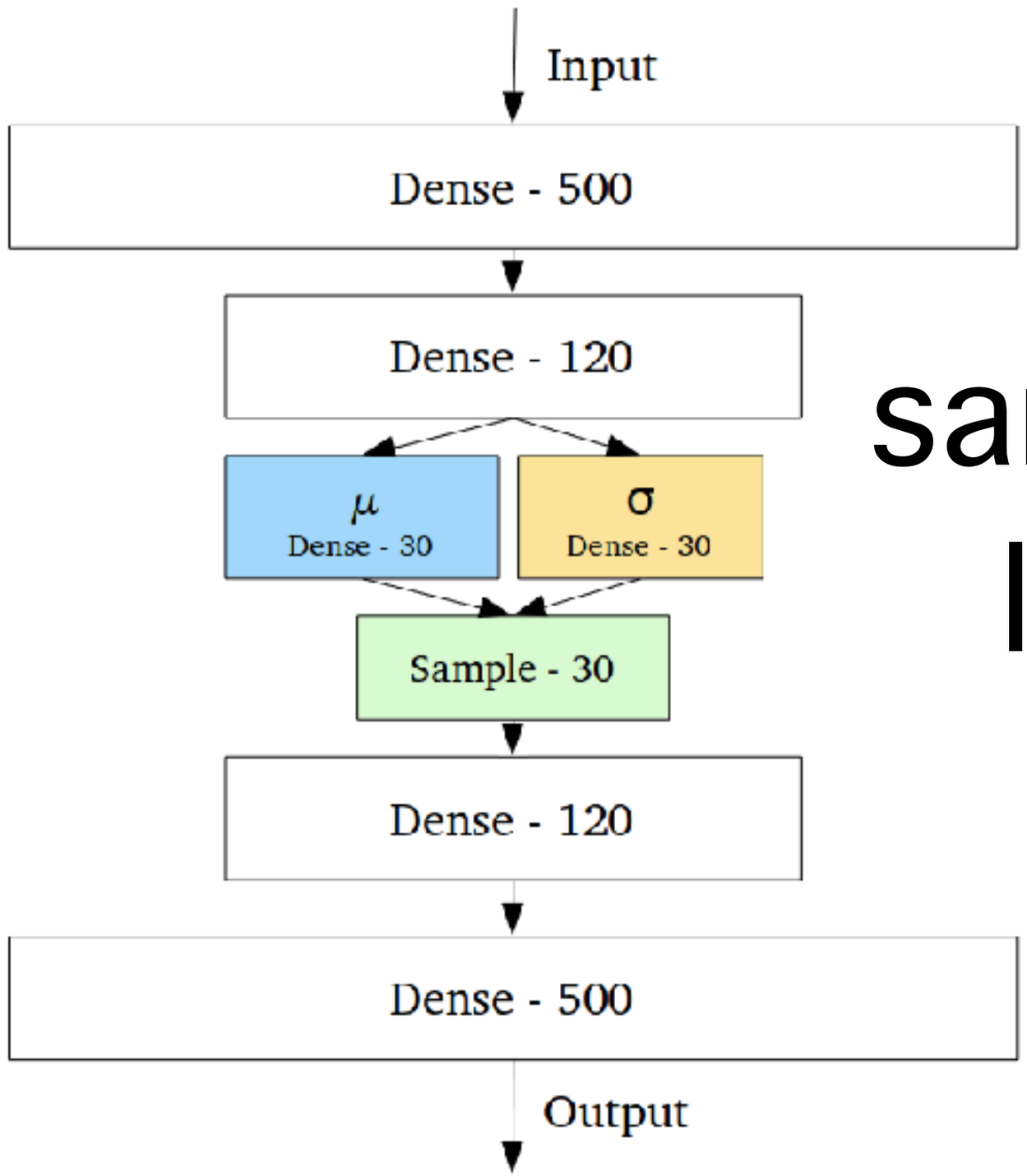
Image-to-Image (Conditional GANs)



Exemplo: Colorização de Mangás (condição são os traços e uma “dica de cor”)

Image-to-Image (Variational AutoEncoders)

Outro modelo generativo

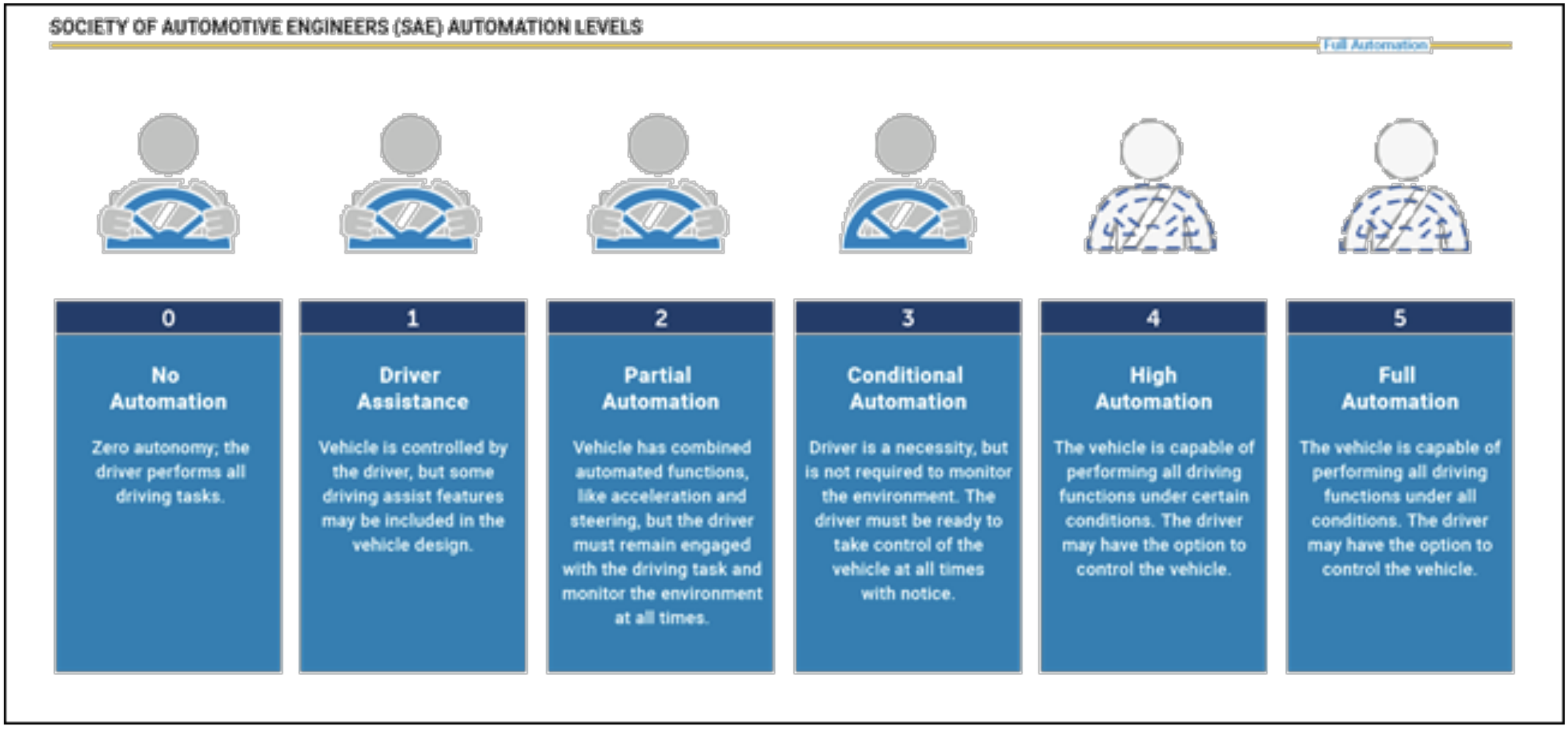


sampling layer

Temos varias abordagens em self-driving cars para cada empresa que está na luta.

Algumas acreditam que é importante criar um state-vector claro e interpretáveis (waymo)

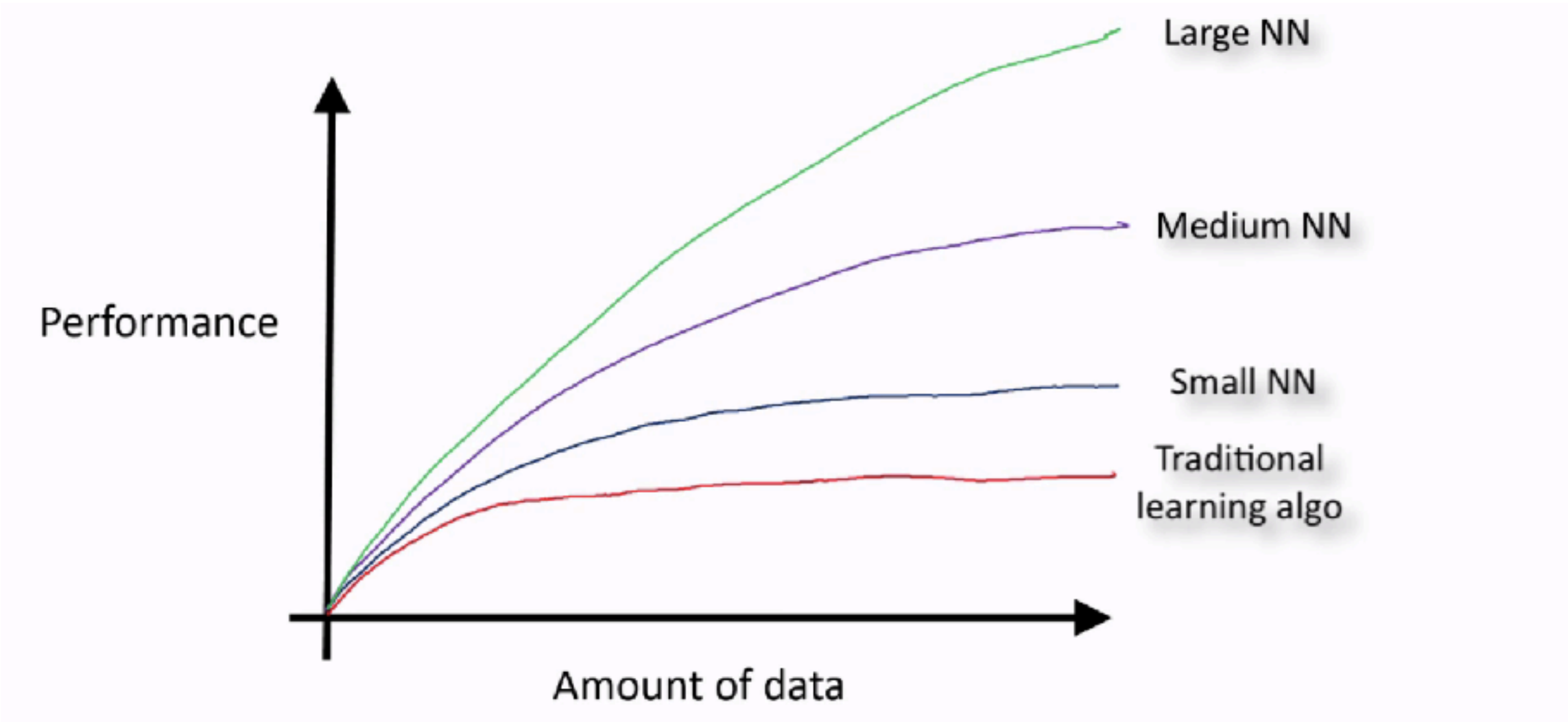
Outras já querem realizar o mapeamento direto dos sensores para a ação.



Hoje a briga ainda está no Level 2

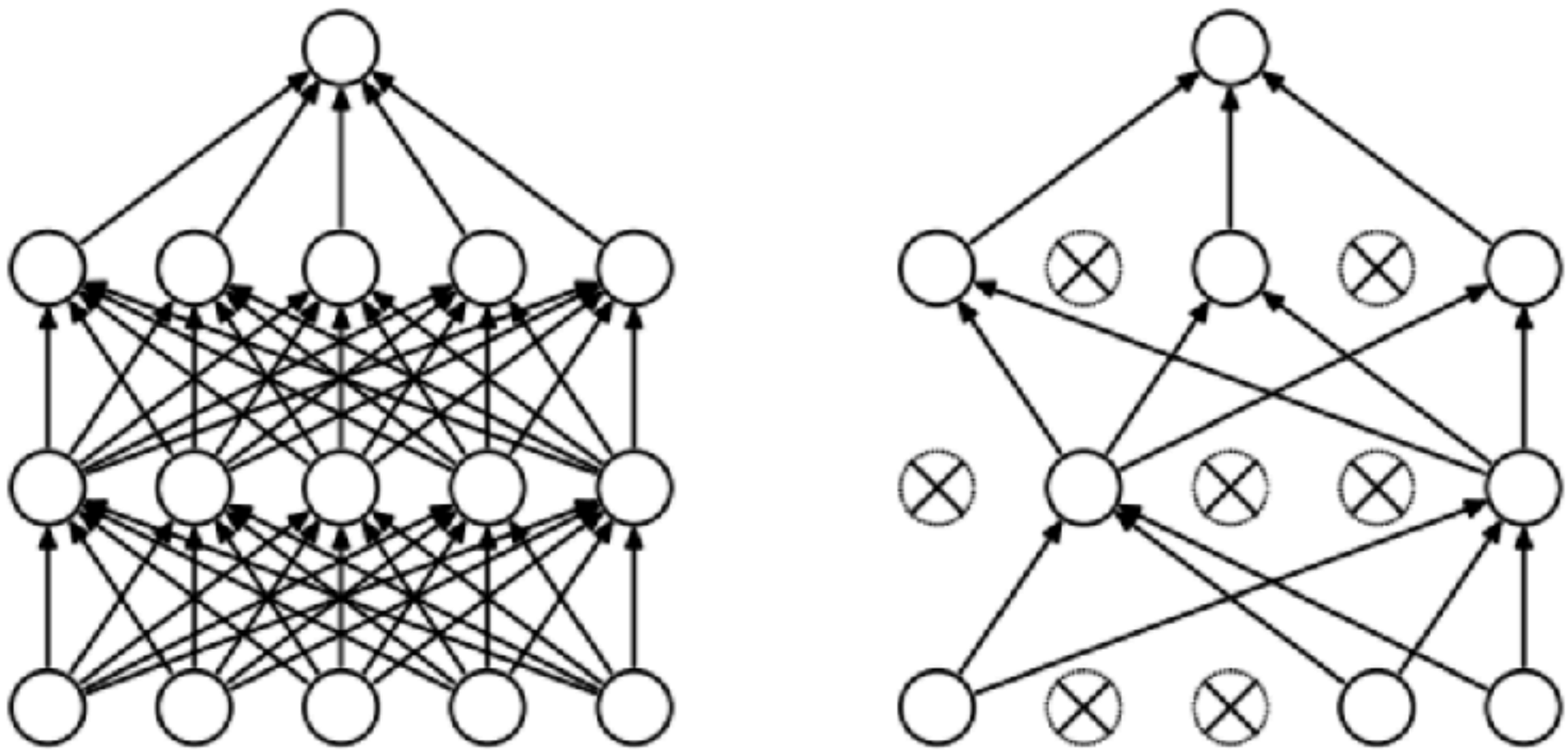
MAS...

Data Hungry



Conseguimos utilizar alguns truques para superar isso, mas muitas vezes as abordagens clássicas ainda valem a pena (custo computacional baixo), quando os problemas não são tão “difíceis”

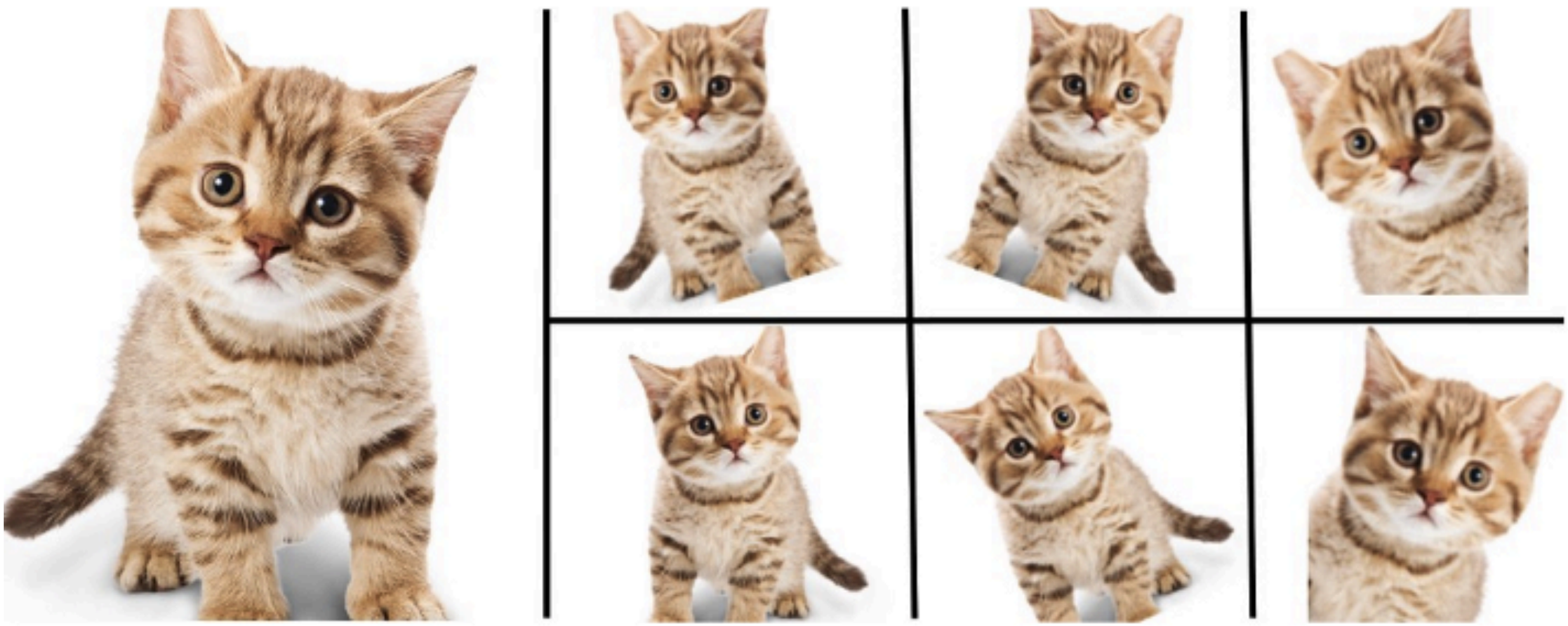
Dropout



(a) Standard Neural Net

(b) After applying dropout.

Data Augmentation



Enlarge your Dataset

Batch Normalization

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1\ldots m}\}$;
Parameters to be learned: γ, β

Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_\mathcal{B} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_\mathcal{B}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m} (x_i - \mu_\mathcal{B})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_\mathcal{B}}{\sqrt{\sigma_\mathcal{B}^2 + \epsilon}} \qquad \text{// normalize}$$

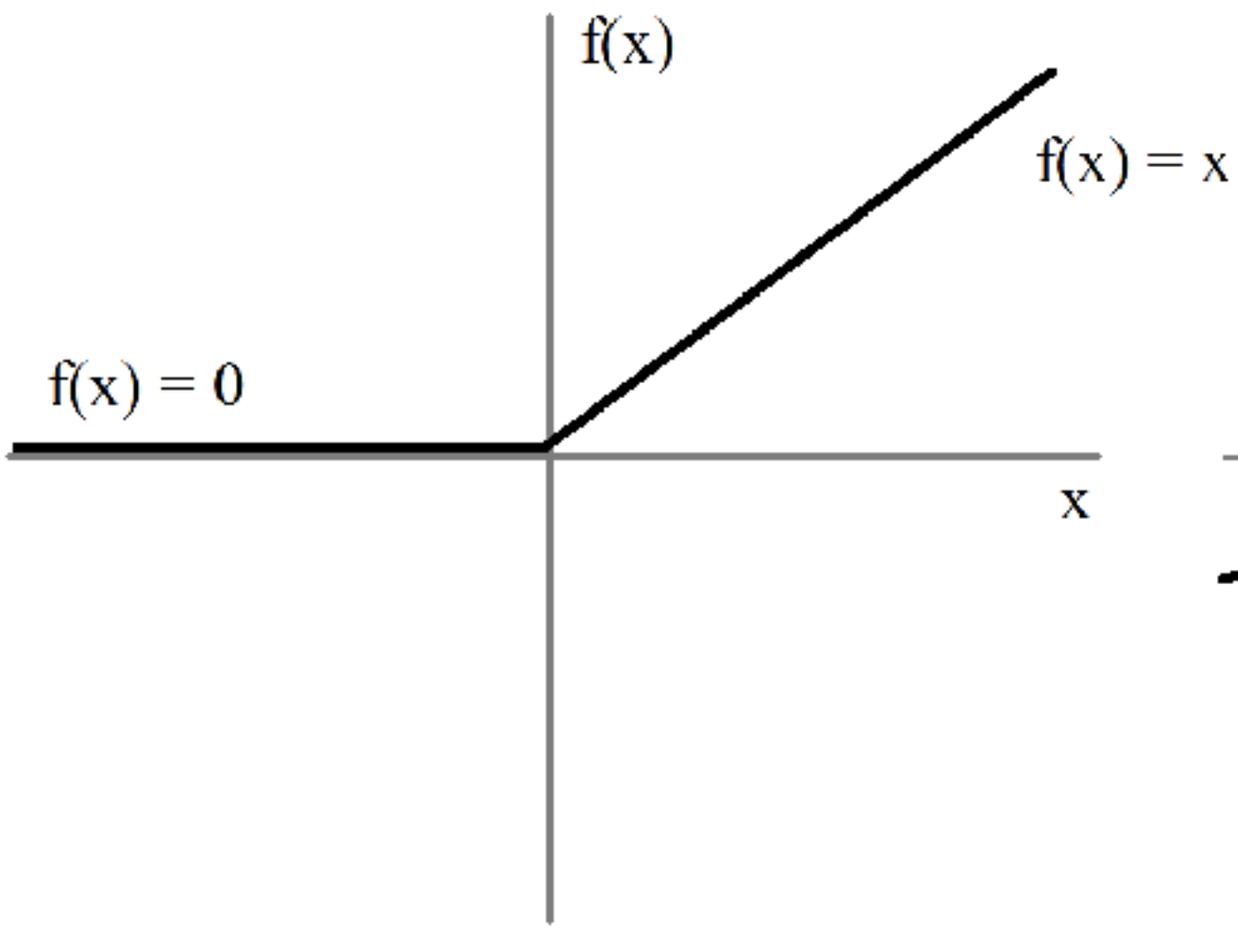
$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

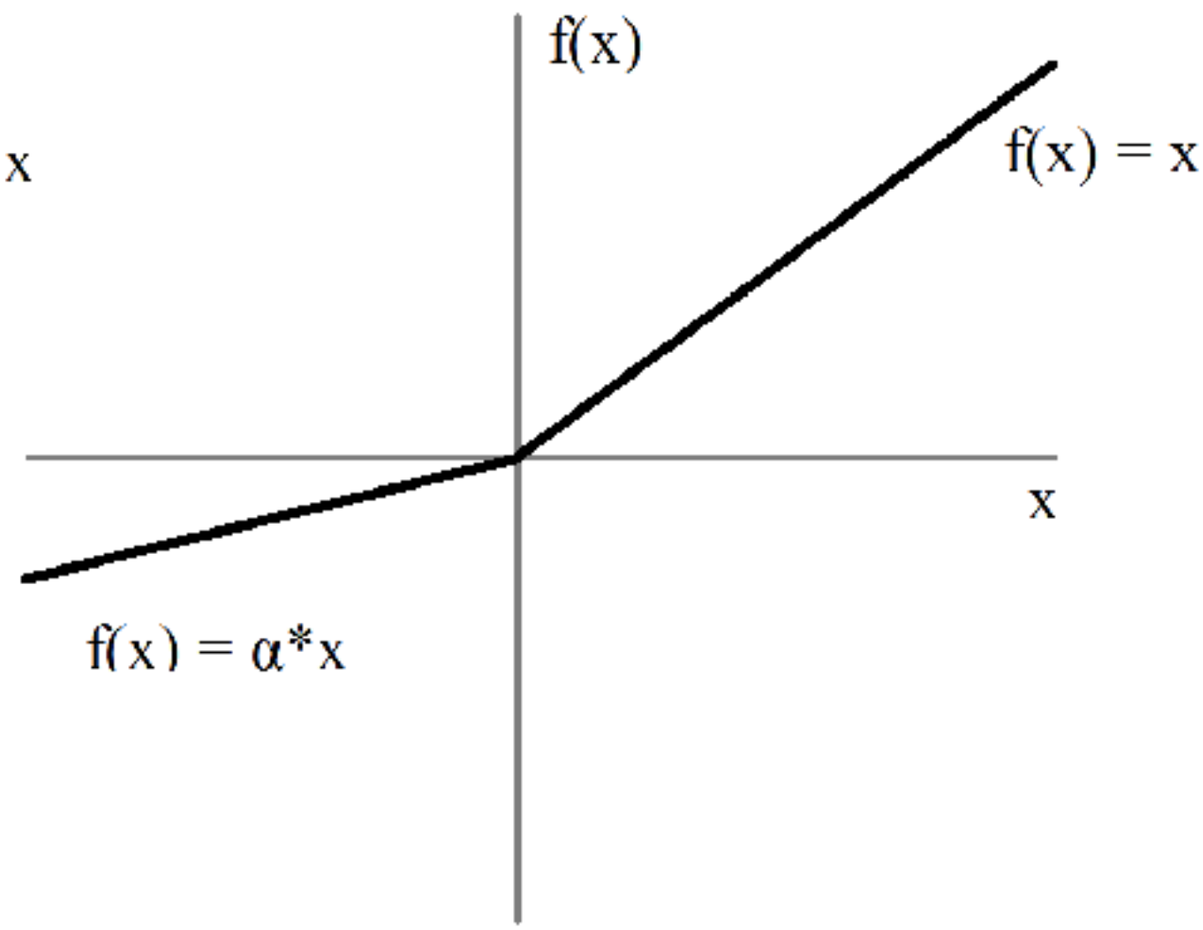
Algorithm 1: Batch Normalizing Transform, applied to activation x over a mini-batch.

independency

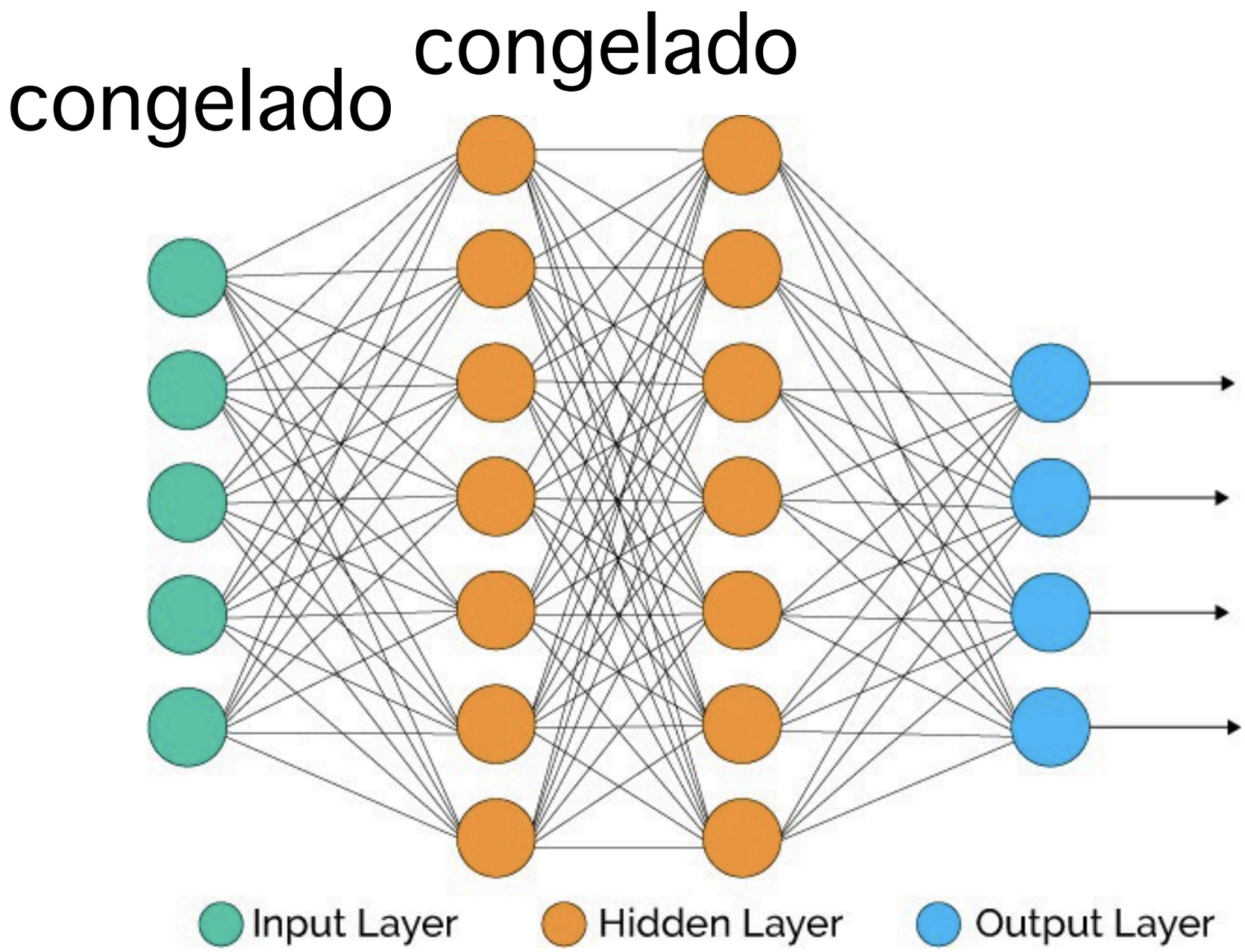
gradient explosion

ReLU vs Leaky ReLU





Extreme Learning Machine (ELM)



Referências

Youtube

Lex Friedman

Two Minute Papers

ArXiv Insights

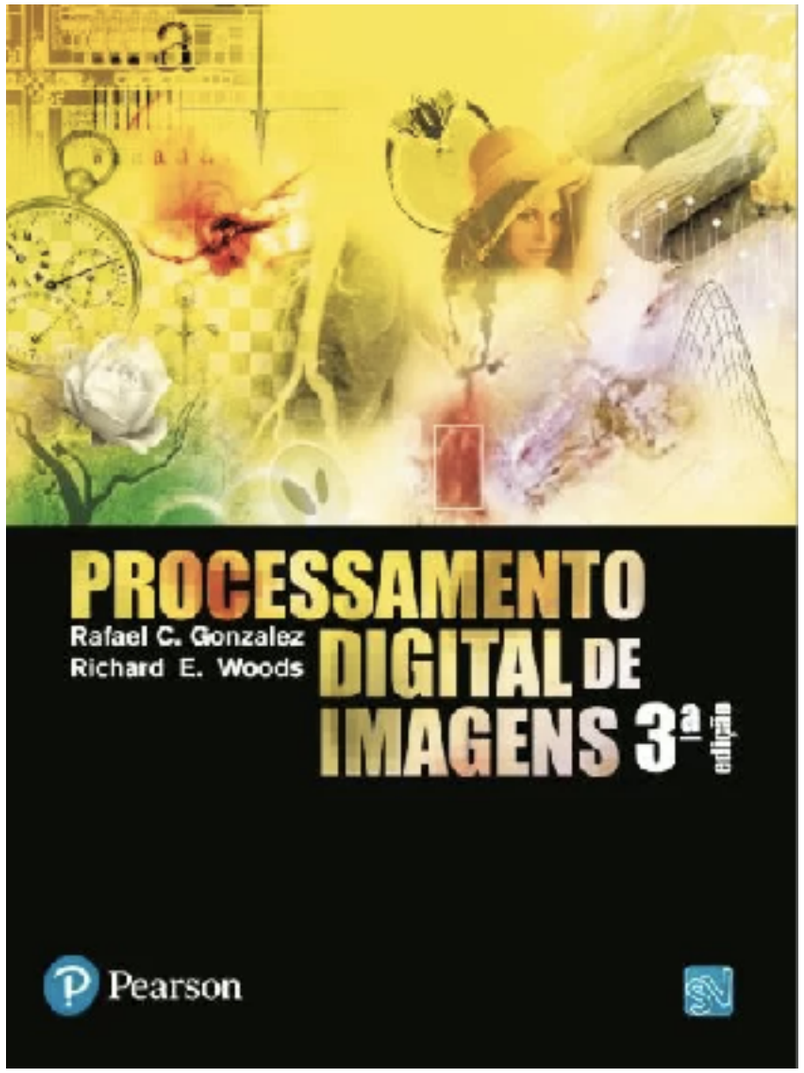
Courses

cs231n (Stanford)

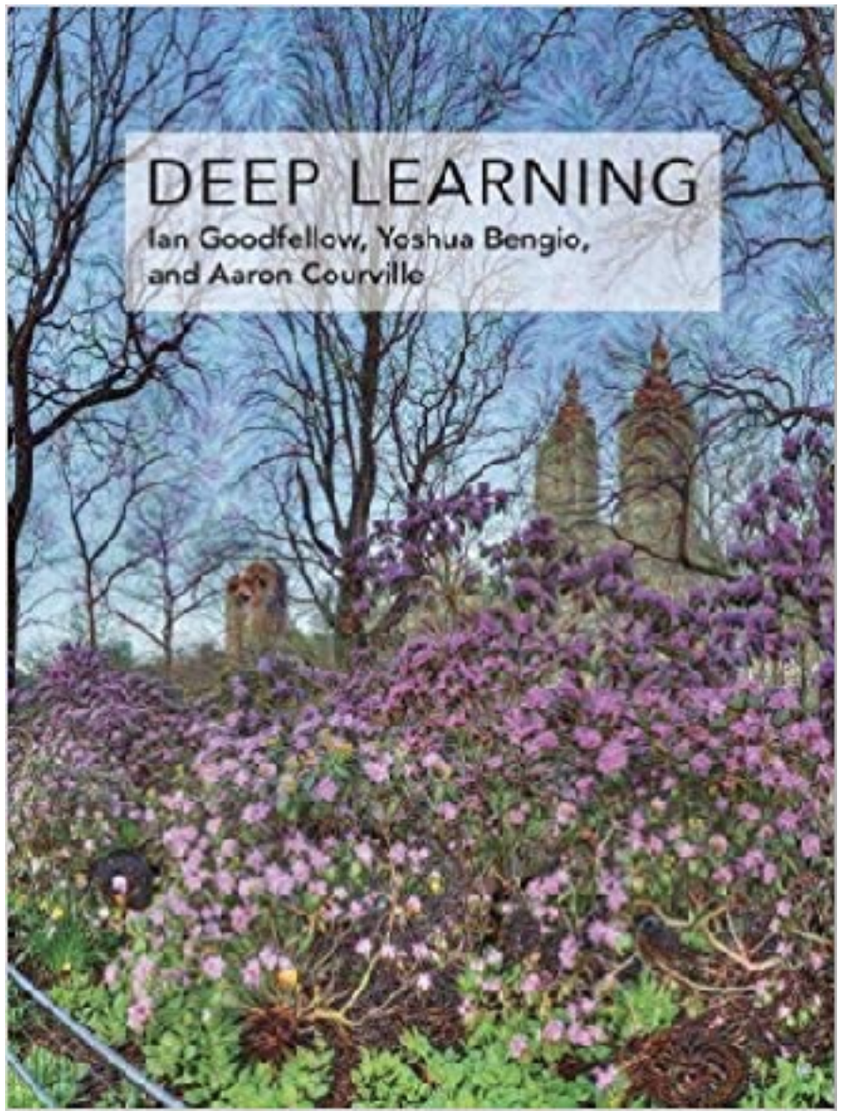
fast.ai (implementação)

Referências

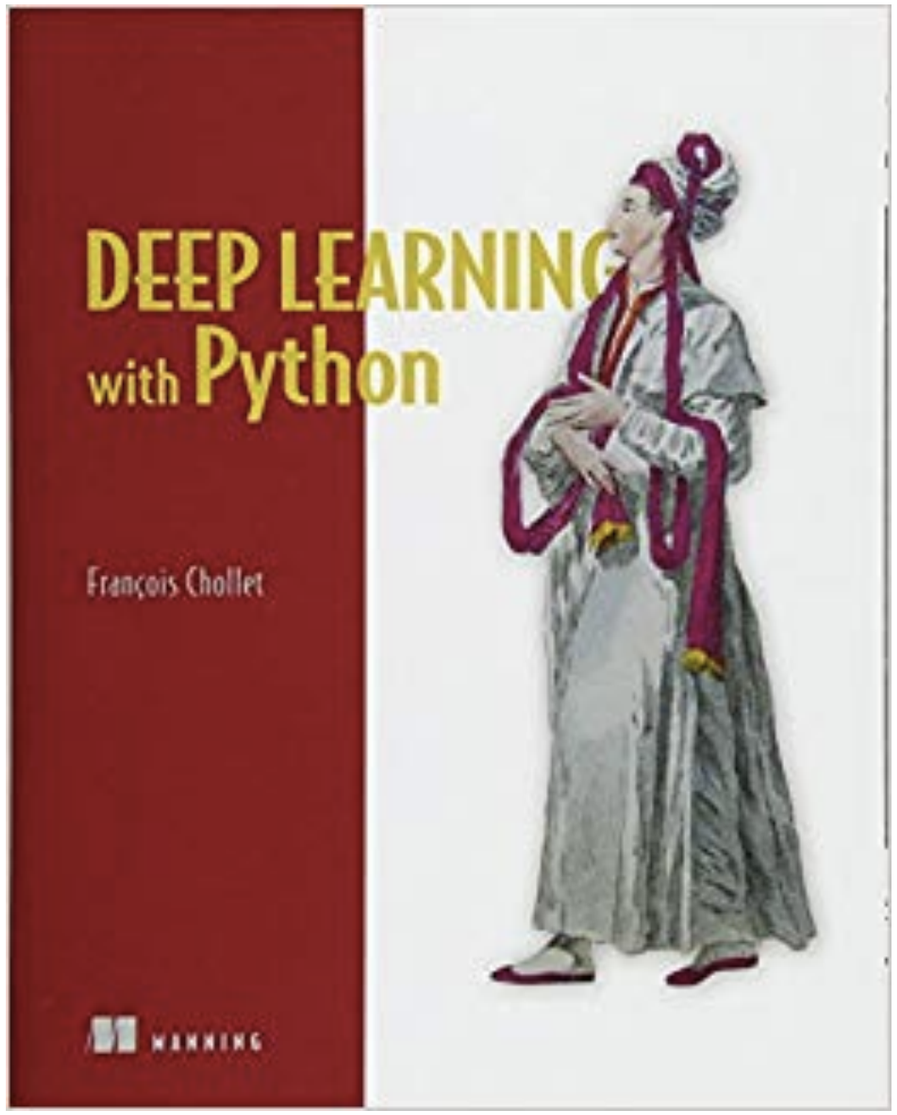
Teoria Clássica



Teoria Deep



Prática Deep (keras)



Ferramentas (Classico)

- **OpenCV:** Low-level e Utilitários
- **skimage:** Filtros e Transformadas
- **OCR:** Tesseract e Attention-OCR

Ferramentas (Deep)

- **TensorFlow:** Low-level
- **PyTorch:** Alternativa Low-level
- **Keras:** High-level
- **TensorFlow.js:** Browser
- **TensorFlow Lite:** Phone
- **TensorBoard:** Visualização

Outros tópicos

Pré-processamento:
- Transformadas (Hough, Fourrier, Wavelets)
- Técnicas de Realce

Extração de Features
- SIFT, SURF
- Features de Cores

Outros tópicos

Deep:

- Interpretação (LRP, Sensibilidade)
- Redes Siamesas
- Style Transfer
- Image Captioning
- Image Synthesis
- Weakly Supervision e Self-superv.

THANK YOU