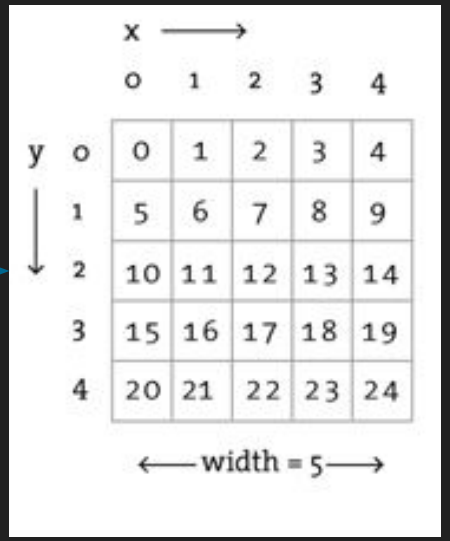
1º RASCore

Visão Computacional e Deep Learning

Ministrante:
Guilherme Christmann

Introdução

- O que é uma imagem?
- Resolução.
- Canais de cor.




```
cv2.imread() cv2.imshow()
cv2.VideoCapture()
cv2.waitKey()
```

Operações Básicas

- Dimensões de uma imagem. `img.shape`
- Acessando valor de um pixel. `img[y, x]`
- Separando e juntando canais. `cv2.split cv2.merge`
- ROI (Region of Interest). `img[y0:y1, x0:x1]`

Operações Básicas

- Dimensões de uma imagem. `img.shape`
- Acessando valor de um pixel. `img[y, x]`
- Separando e juntando canais. `cv2.split cv2.merge`
- ROI (Region of Interest). `img[y0:y1, x0:x1]`

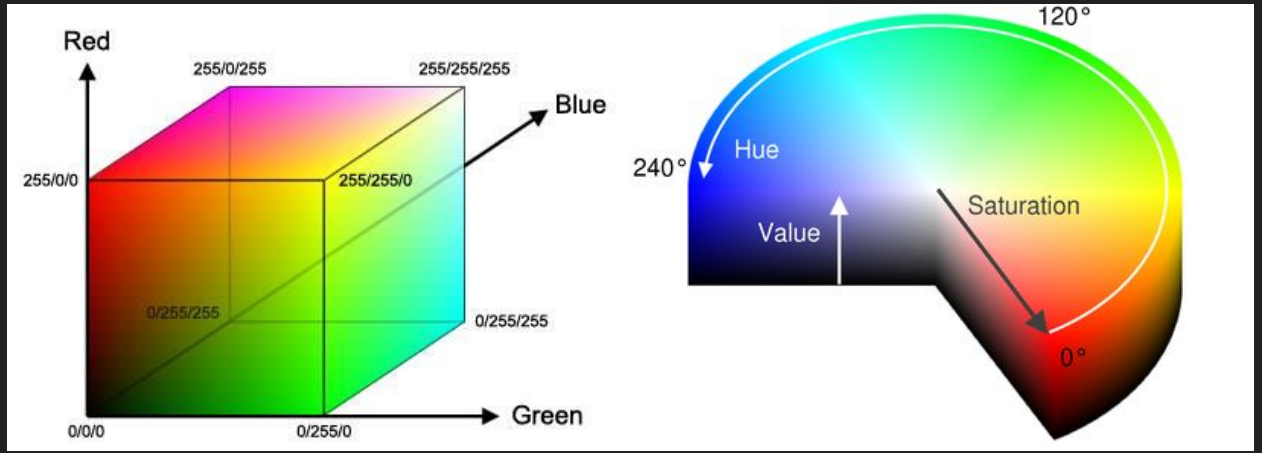
Exercício: Aumentar o brilho de um dos canais de uma imagem RGB.

Exercício: Aumentar o brilho de apenas uma região da imagem.

Espaços de Cor

- RGB.
- GRAY.
- HSV.

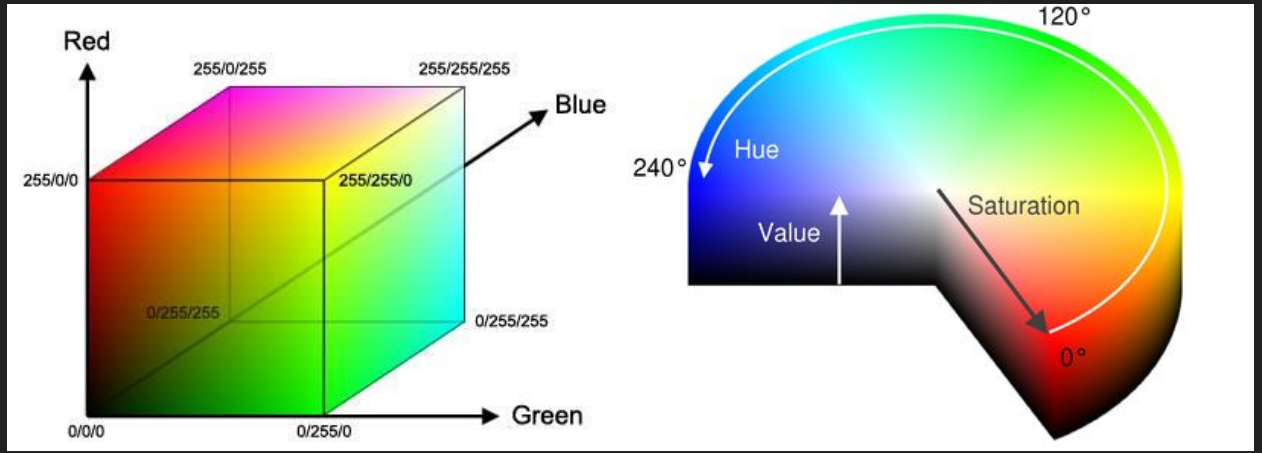
```
cv2.cvtColor(imagem, flag_de_conversao)
```



Espaços de Cor

- RGB.
- GRAY.
- HSV.

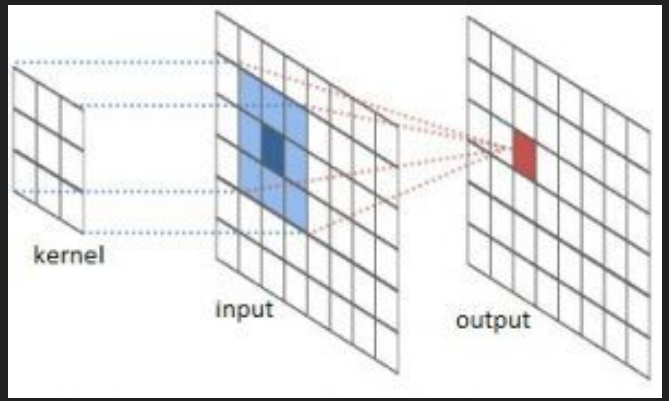
```
cv2.cvtColor(imagem, flag_de_conversao)
```



Exercício: Fazer uma conversão de RGB (BGR) para Gray na “mão”.

Convolução e Filtros

- Convolução 2D.



1	1	1	1	1
1	0	0	0	1
1	0	0	0	1
1	0	0	0	1
1	1	1	1	1

Blur

1	1	1	1	1
1	5	5	5	1
1	5	44	5	1
1	5	5	5	1
1	1	1	1	1

Smooth

1	1	2	1	1
1	2	4	2	1
2	4	8	4	2
1	2	4	2	1
1	1	2	1	1

Gaussian

Convolução e Filtros

- Convolução 2D.
- Filtro Box.
- Filtro Gaussiano.

```
cv2.filter2D(img, -1, kernel)
```

```
cv2.blur(img, kernel_size)
```

```
cv2.GaussianBlur(img, kernel_size, sigmaX)
```

Convolução e Filtros

- Convolução 2D.
- Filtro Box.
- Filtro Gaussiano.

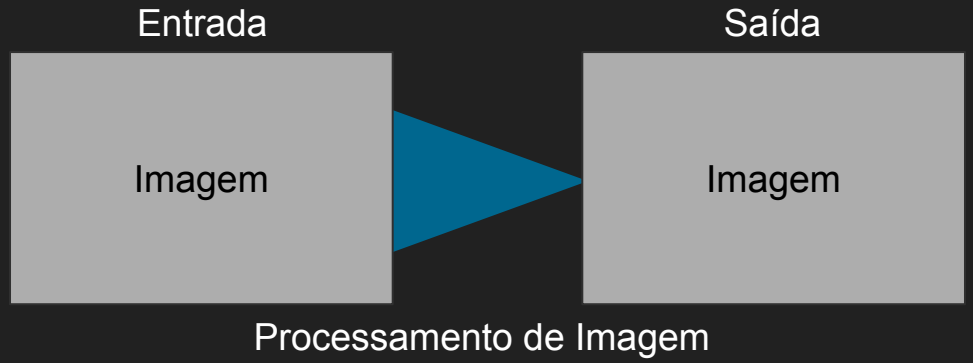
```
cv2.filter2D(img, -1, kernel)
```

```
cv2.blur(img, kernel_size)
```

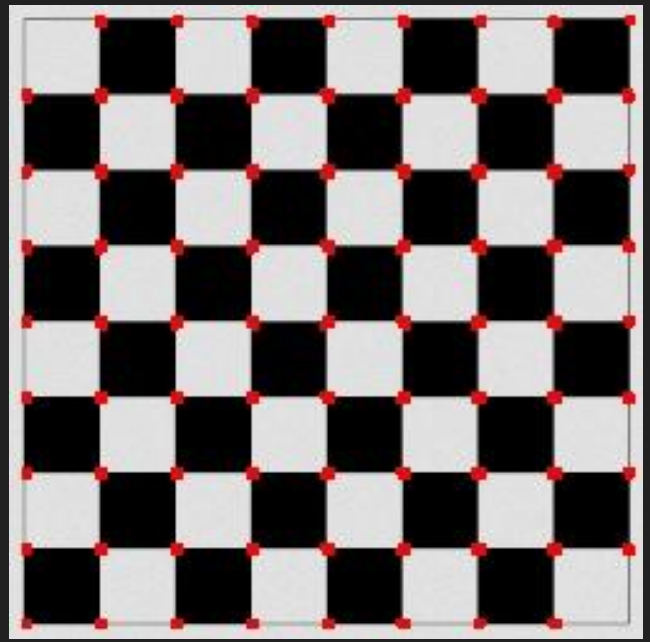
```
cv2.GaussianBlur(img, kernel_size, sigmaX)
```

Exercício: Experimente utilizar kernels diferentes com a operação Filter2D.

Processamento de Imagem x Visão Computacional



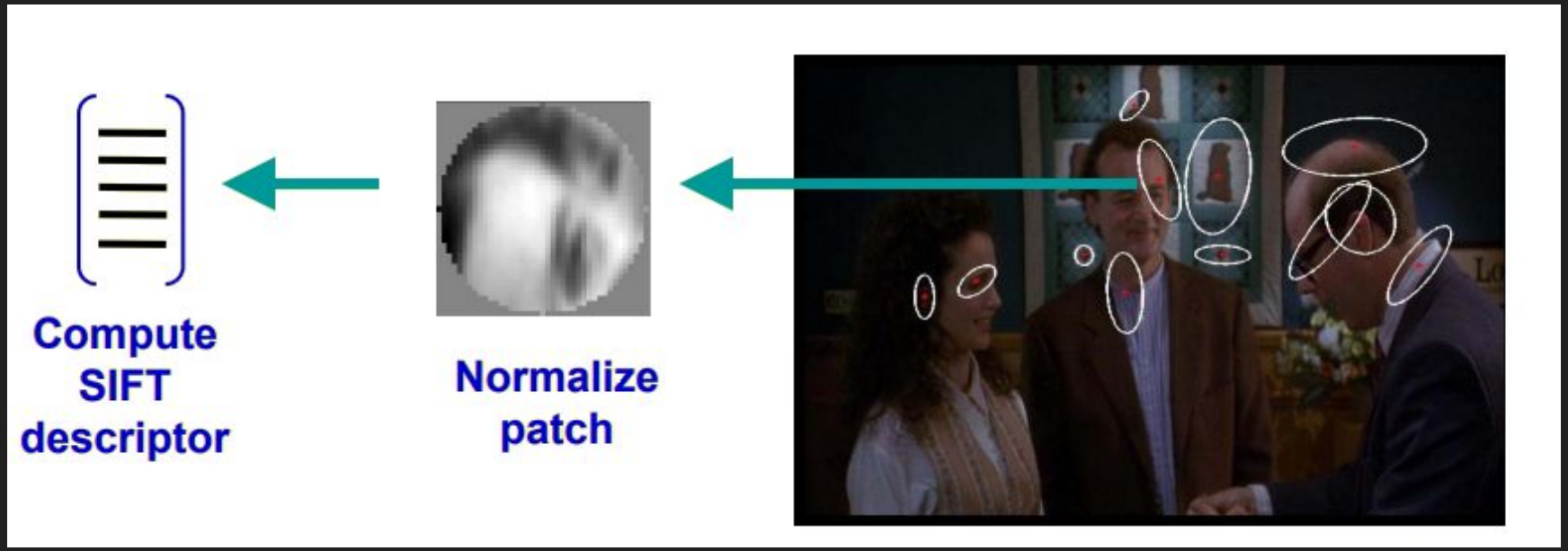
Algoritmos “Clássicos” - Deteccção de Features



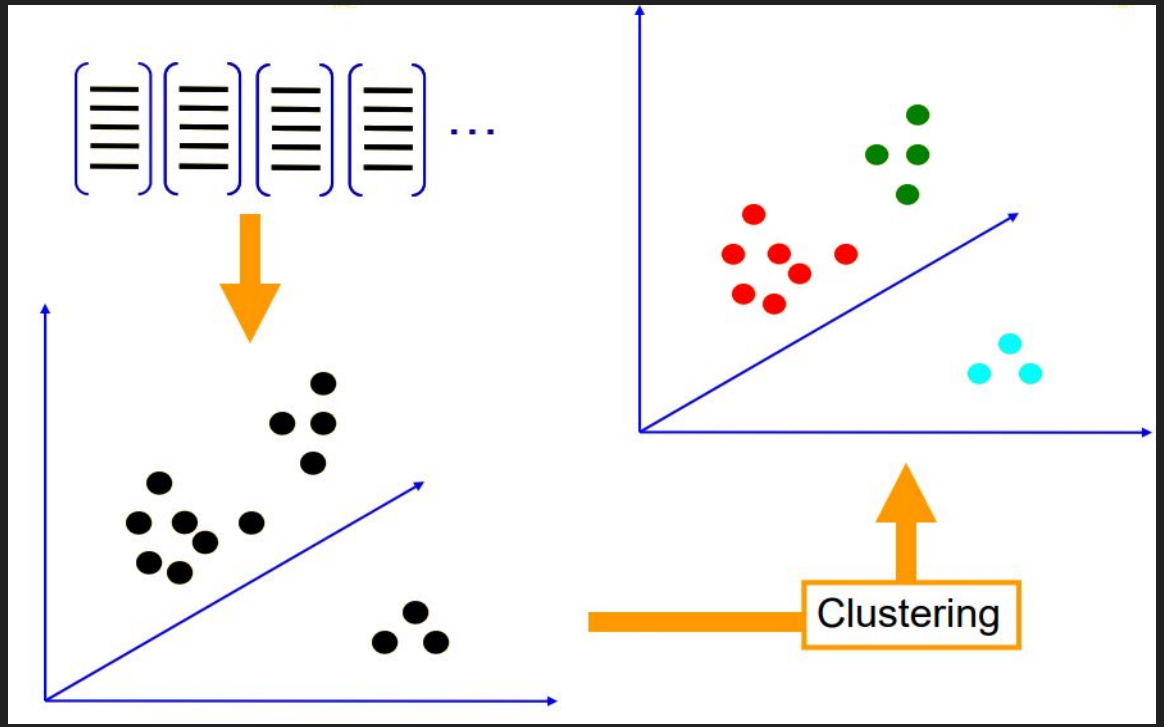
- Harris-Corner
- Shi-Tomasi
- SIFT (Scale Invariant Feature Transform)

- Detecção de Objetos - Homografia

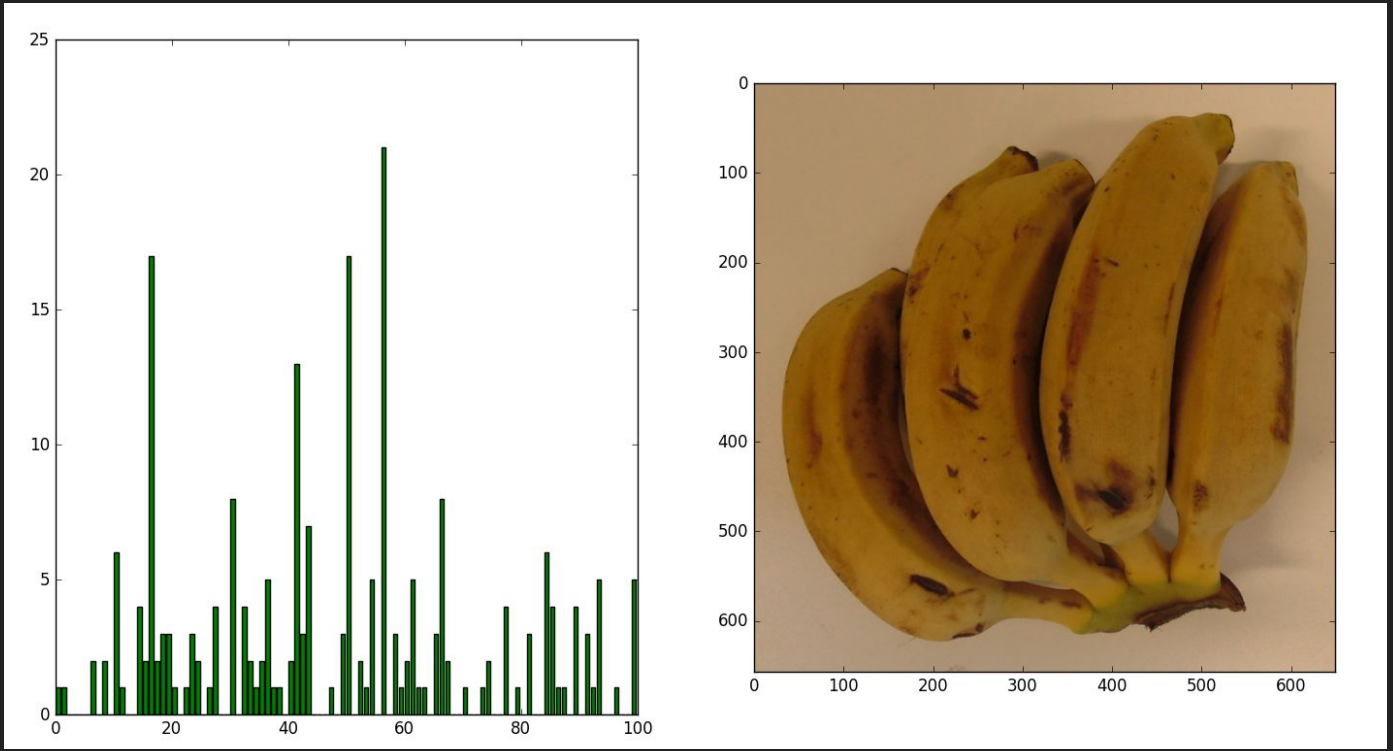
Classificação por Bag of Features - Extração



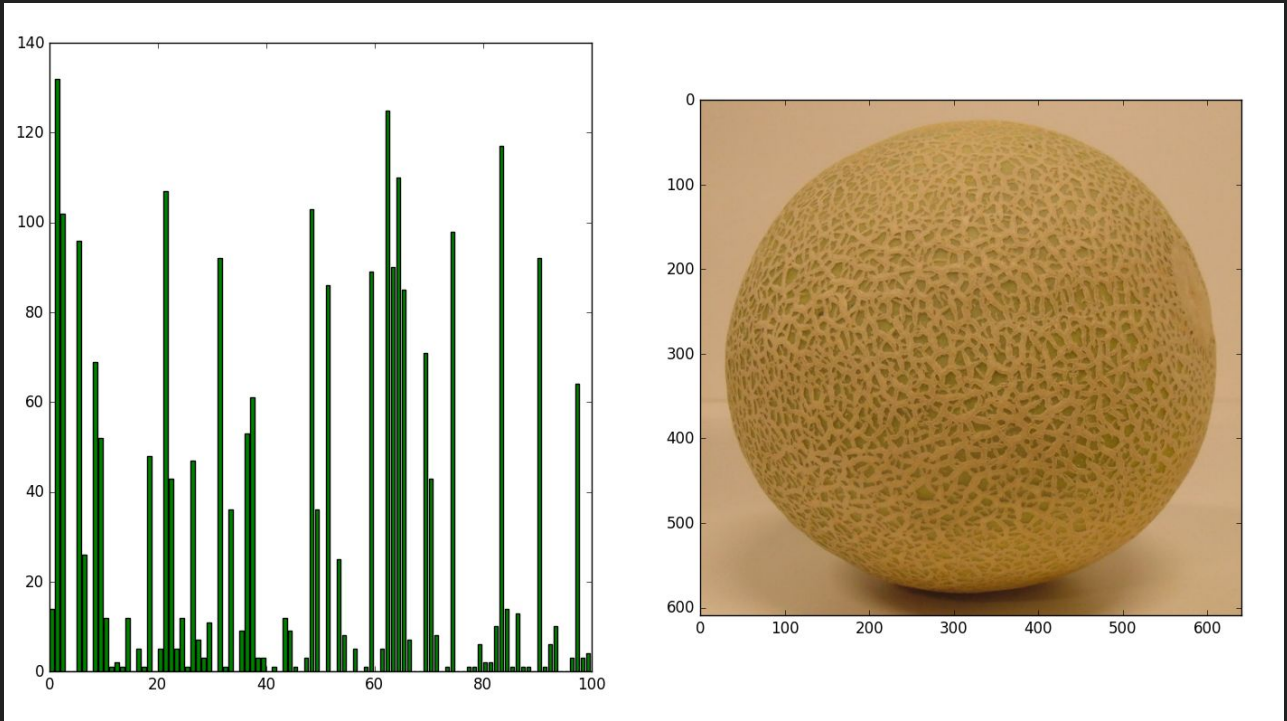
Classificação por Bag of Features - Clustering



Bag of Features - Histograma

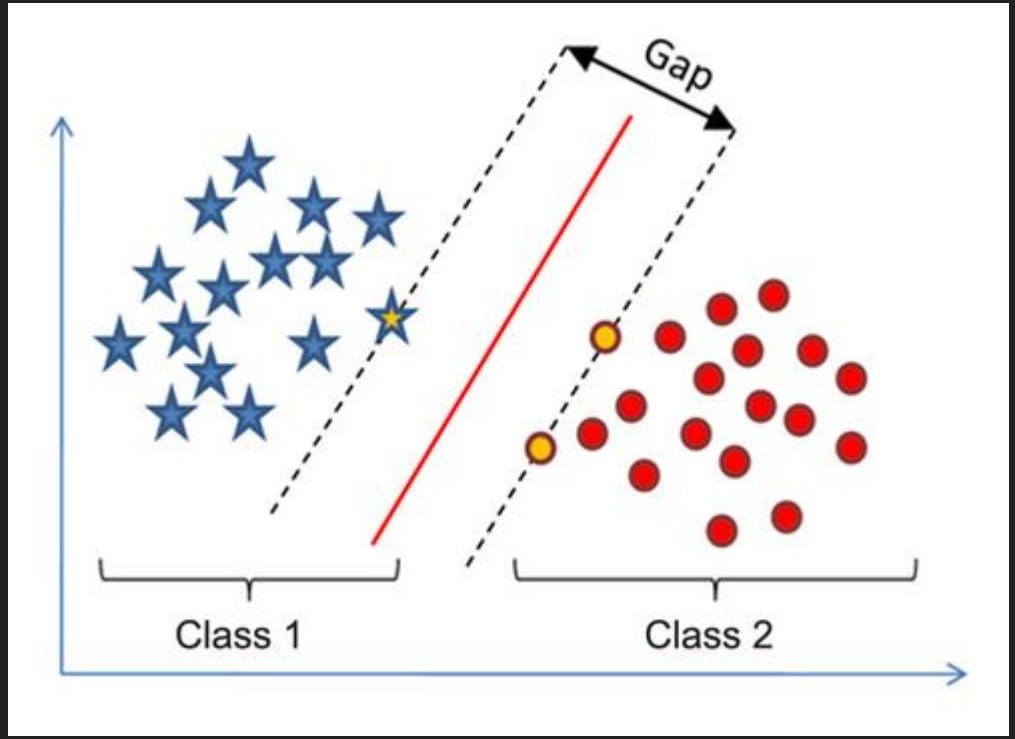


Bag of Features - Histograma



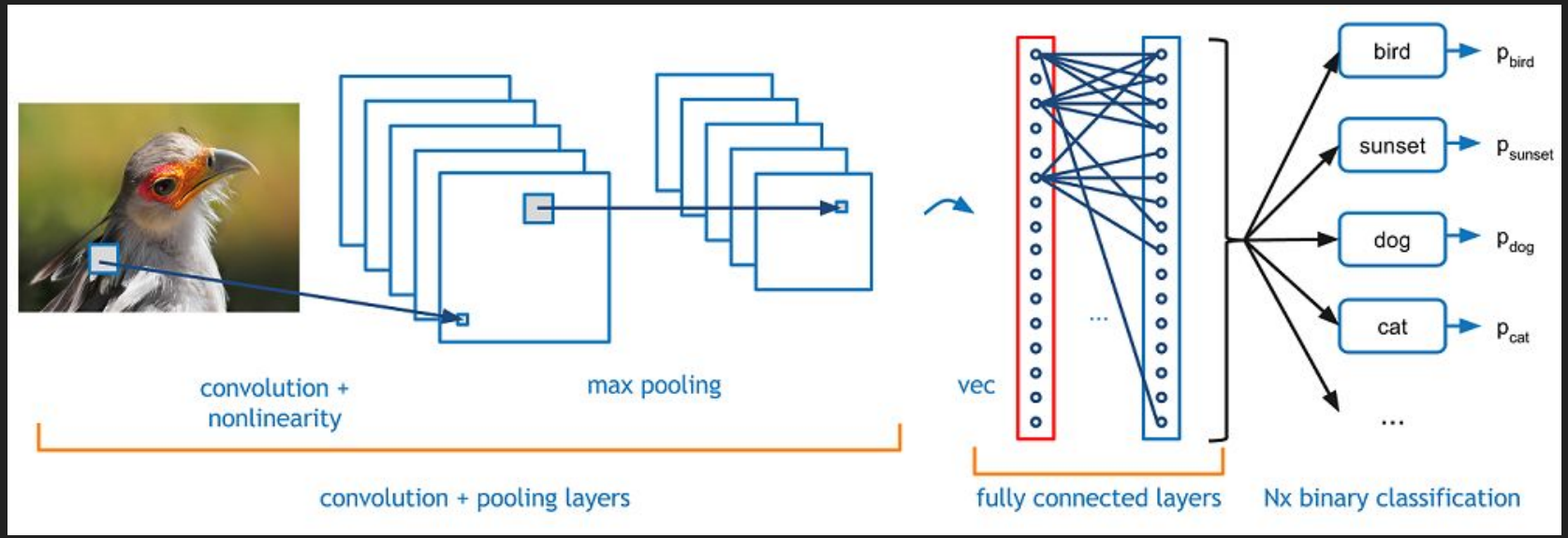
Classificação a partir dos Histogramas

- Regressão Linear
- SVMs
- Redes Neurais

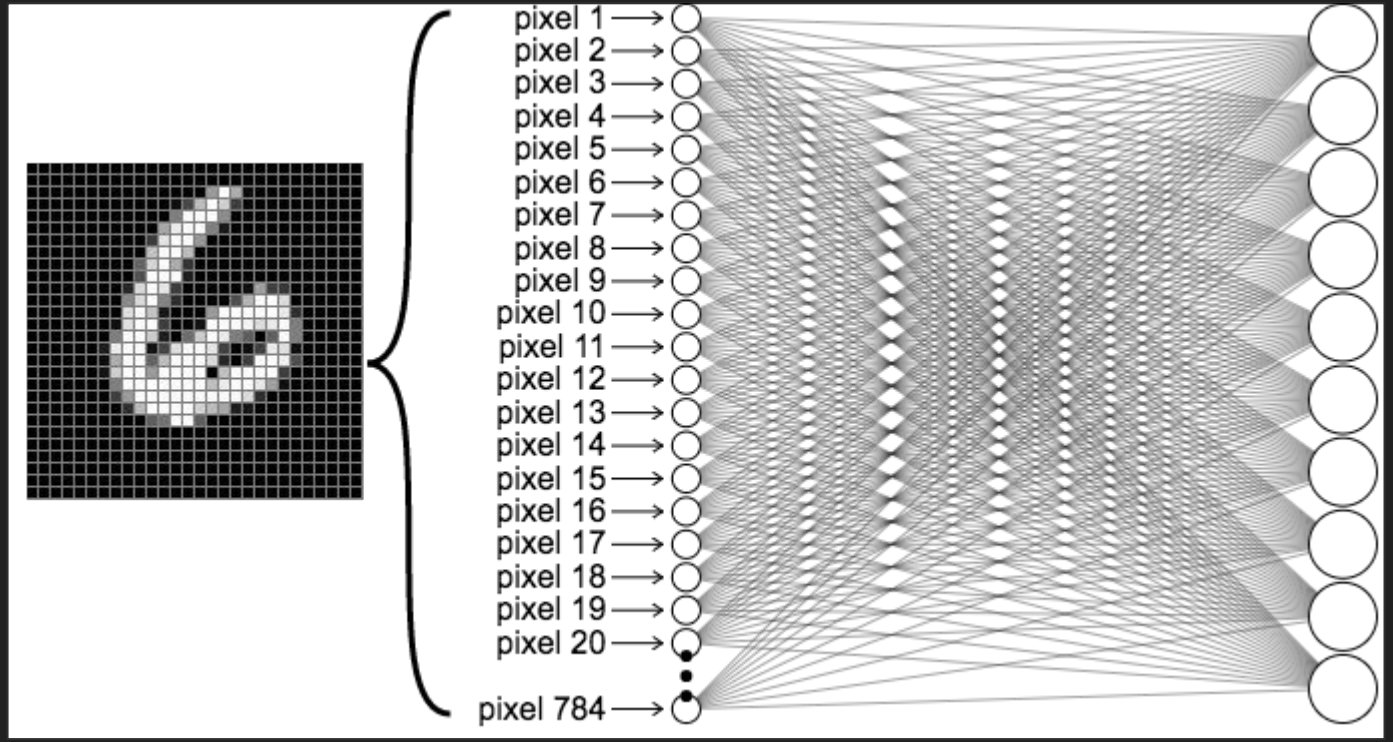


Deep Learning para Visão Computacional

- Classificação End-To-End



Camadas Fully-Connected (Dense)



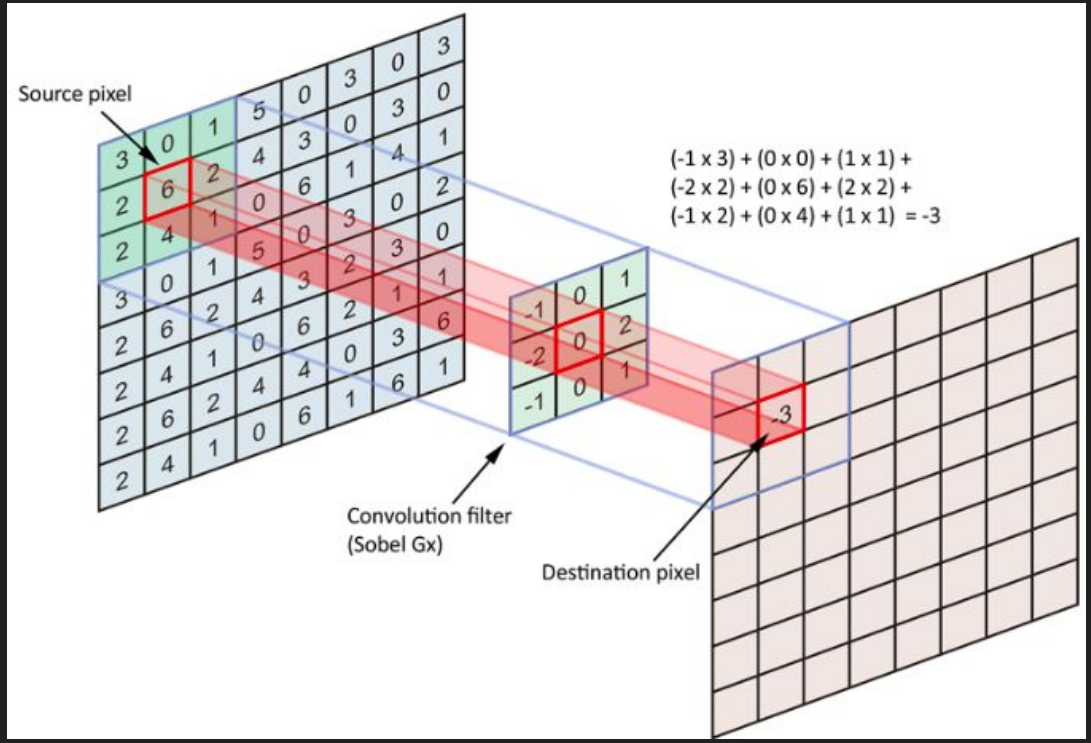
Camadas Fully-Connected (Dense)

- Número de parâmetros

```
Nº Parâmetros FC = DIM_INPUT * N_NEURONIOS + N_NEURONIOS
```

Camadas de Convolução

- Filtros
- Tamanho de Kernel
- Strides
- Padding



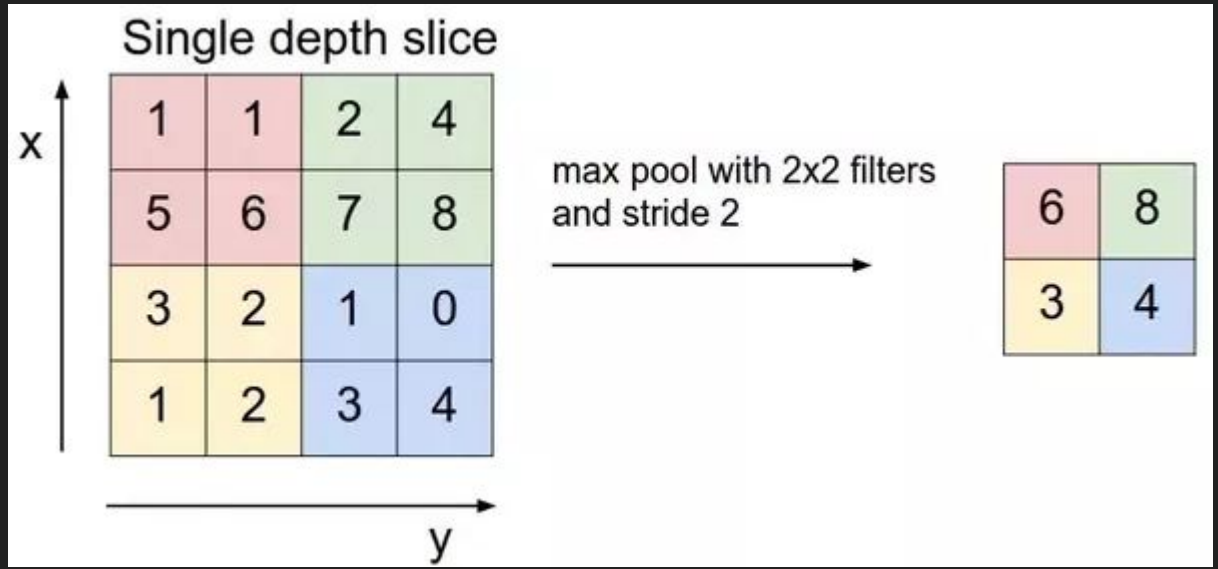
Camadas de Convolução

- Número de parâmetros

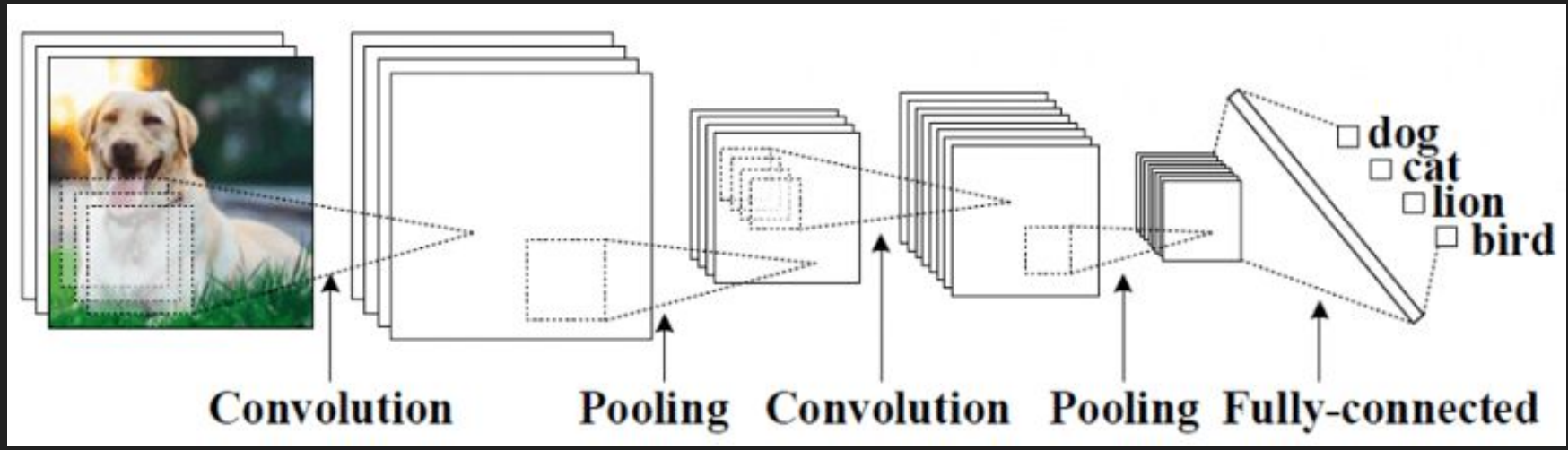
```
Nº Parâmetros CNN = (INPUT_CHANNELS * KERNEL_SIZE *
OUT_CHANNELS + OUT_CHANNELS)
```

Camada de Pooling

- Pool Size
- Max Pooling
- Average Pooling



Topologia CNN -> Pooling -> FC



- Exemplo com MNIST

Exercício

Treinar uma rede com camadas de convolução seguidas de max pooling e classificação a partir de camada Fully-Connected.

YOLO V2

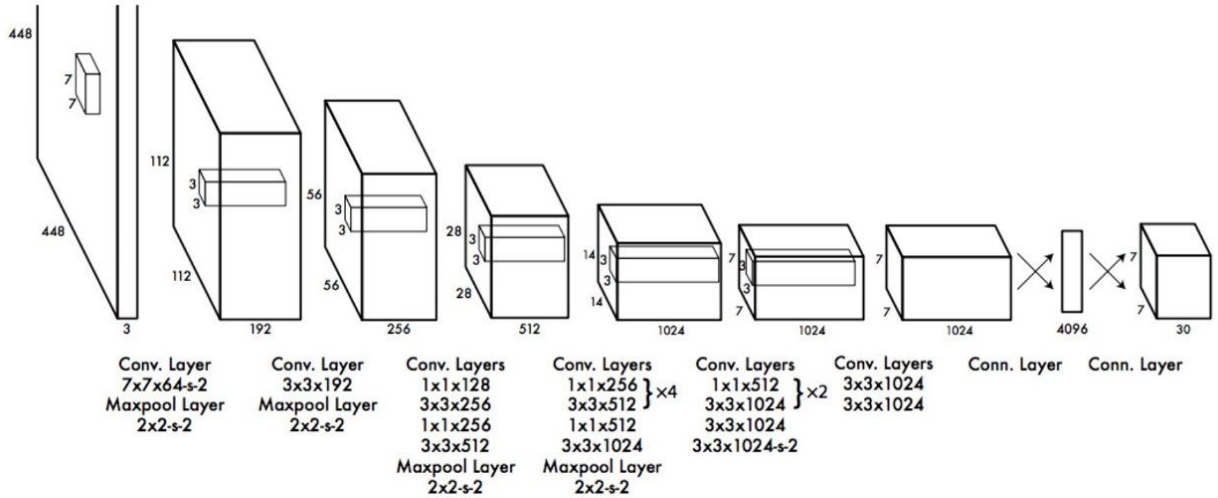
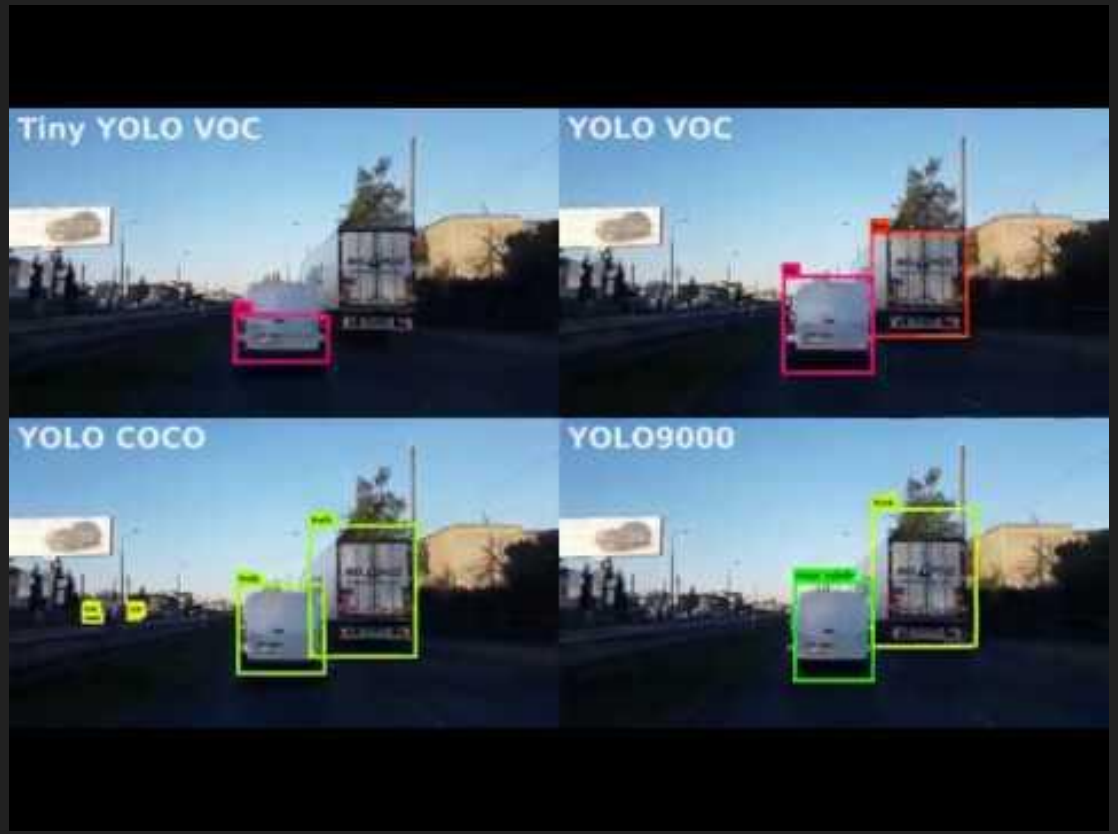


Figure 3: The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

YOLO (You Only Look Once)



Retina Net

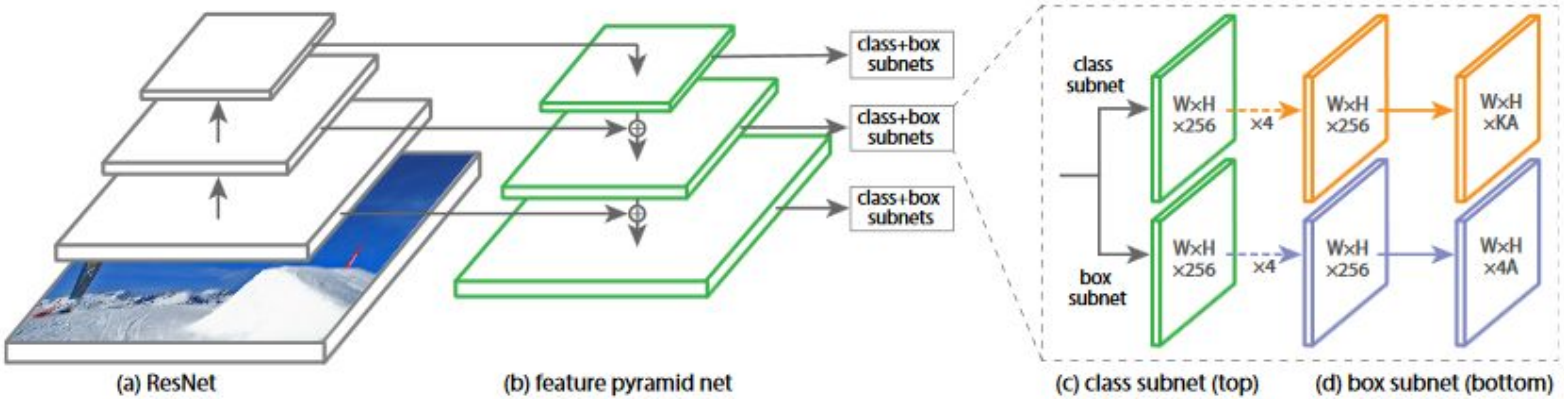
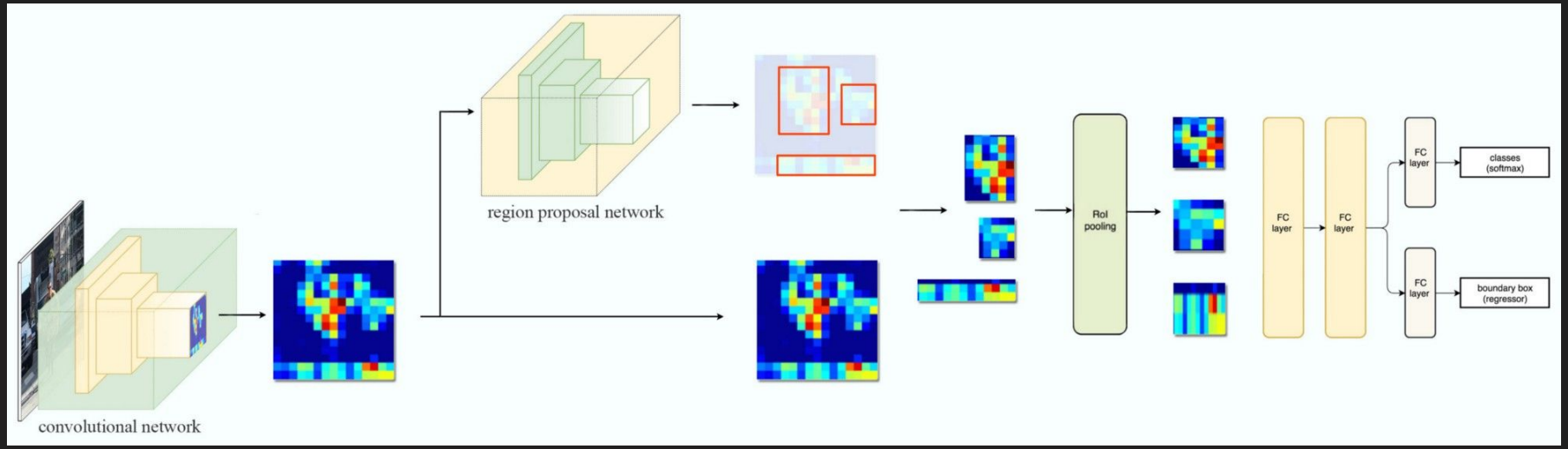


Figure 3. The one-stage **RetinaNet** network architecture uses a Feature Pyramid Network (FPN) [19] backbone on top of a feedforward ResNet architecture [15] (a) to generate a rich, multi-scale convolutional feature pyramid (b). To this backbone RetinaNet attaches two subnetworks, one for classifying anchor boxes (c) and one for regressing from anchor boxes to ground-truth object boxes (d). The network design is intentionally simple, which enables this work to focus on a novel focal loss function that eliminates the accuracy gap between our one-stage detector and state-of-the-art two-stage detectors like Faster R-CNN with FPN [19] while running at faster speeds.

Retina Net

Mask R-CNN


region proposal network
convolutional network
RoI pooling
FC layer
FC layer
FC layer
FC layer
classes (softmax)
boundary box (regressor)

Mask R-CNN


Mask R-CNN
- Object Detection
- Segmentation
backpack
7-23
person 99%
person 92%
person 99%