Reconocimiento de objetos en imágenes

Ariel Rossanigo

Quien soy?

- Ariel Rossanigo
- Profe de Inteligencia Artificial
- Developer, Data Scientist

Objetivos de la charla

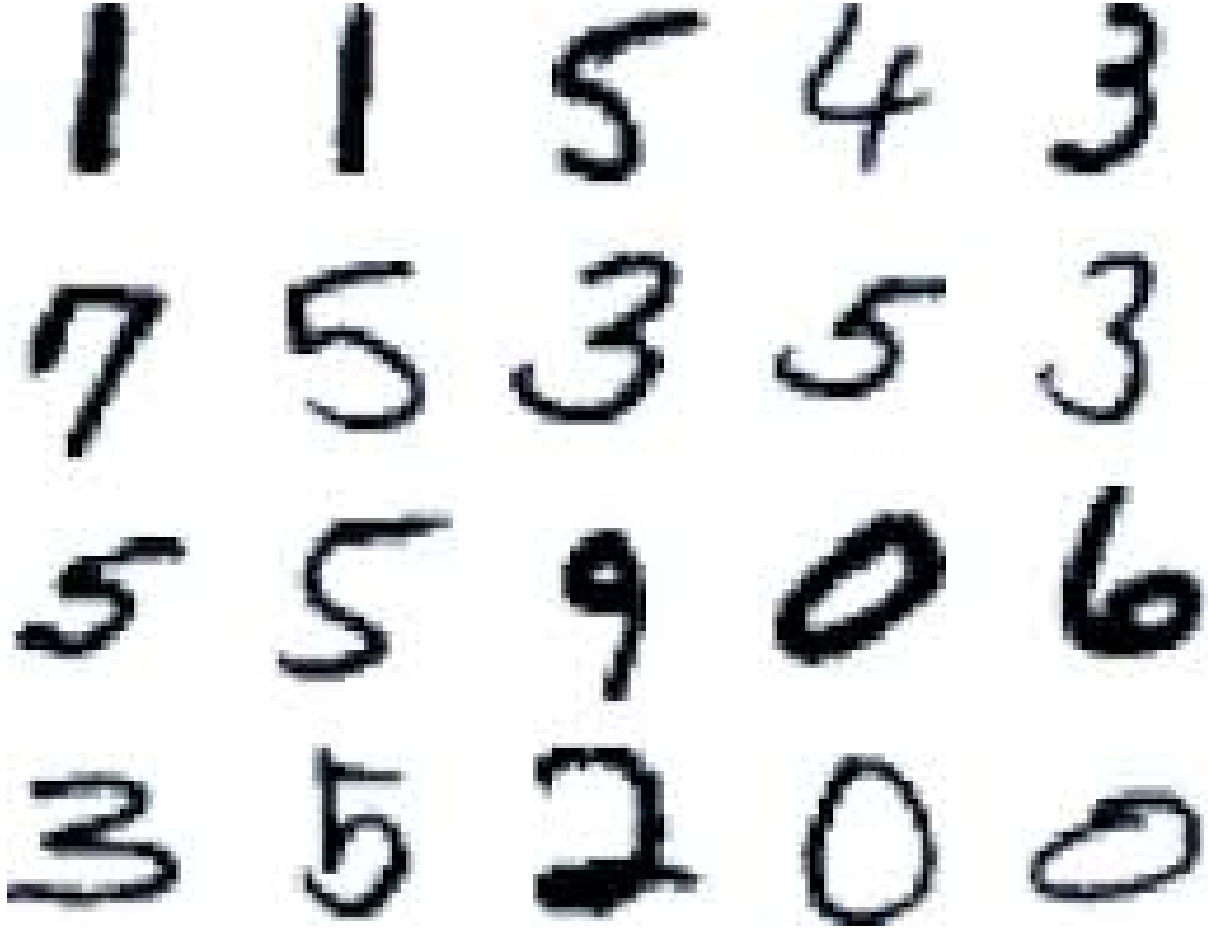
- Mostrar como podemos atacar este tipo de problemas
- Que todos se vayan con una idea del pipeline y que hace cada paso

Agenda

- Definición del problema
- Código
- Explicación de las distintas partes involucradas

Distintos tipos de problemas en CV

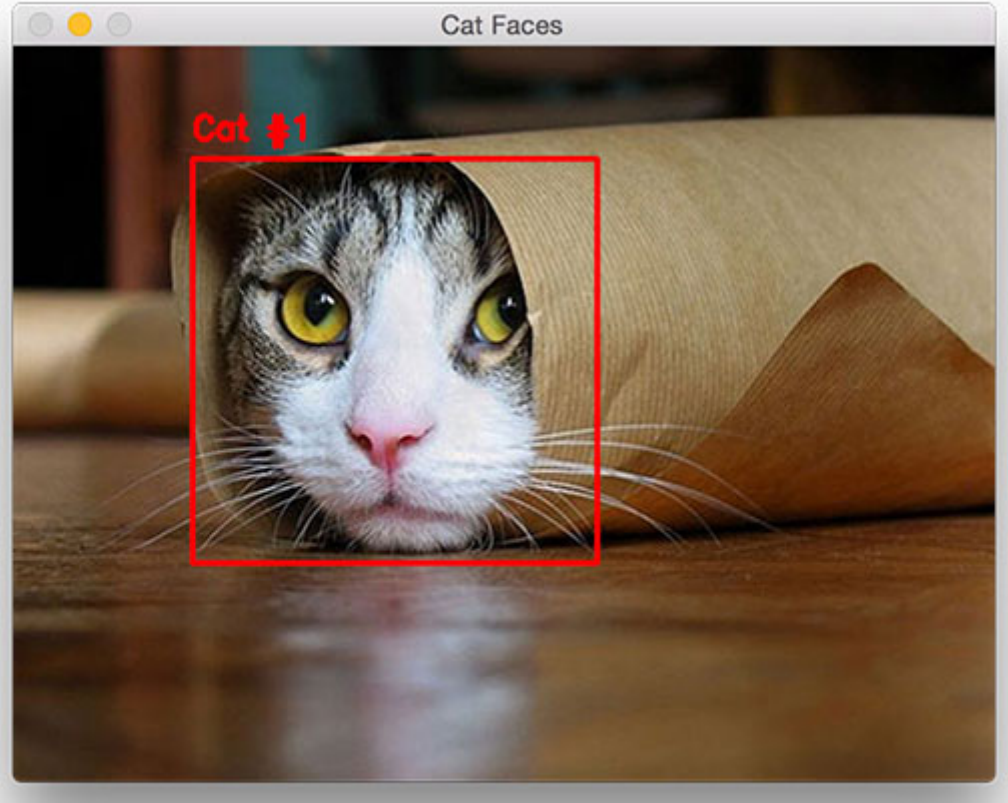
Clasificación



Distintos tipos de problemas en CV

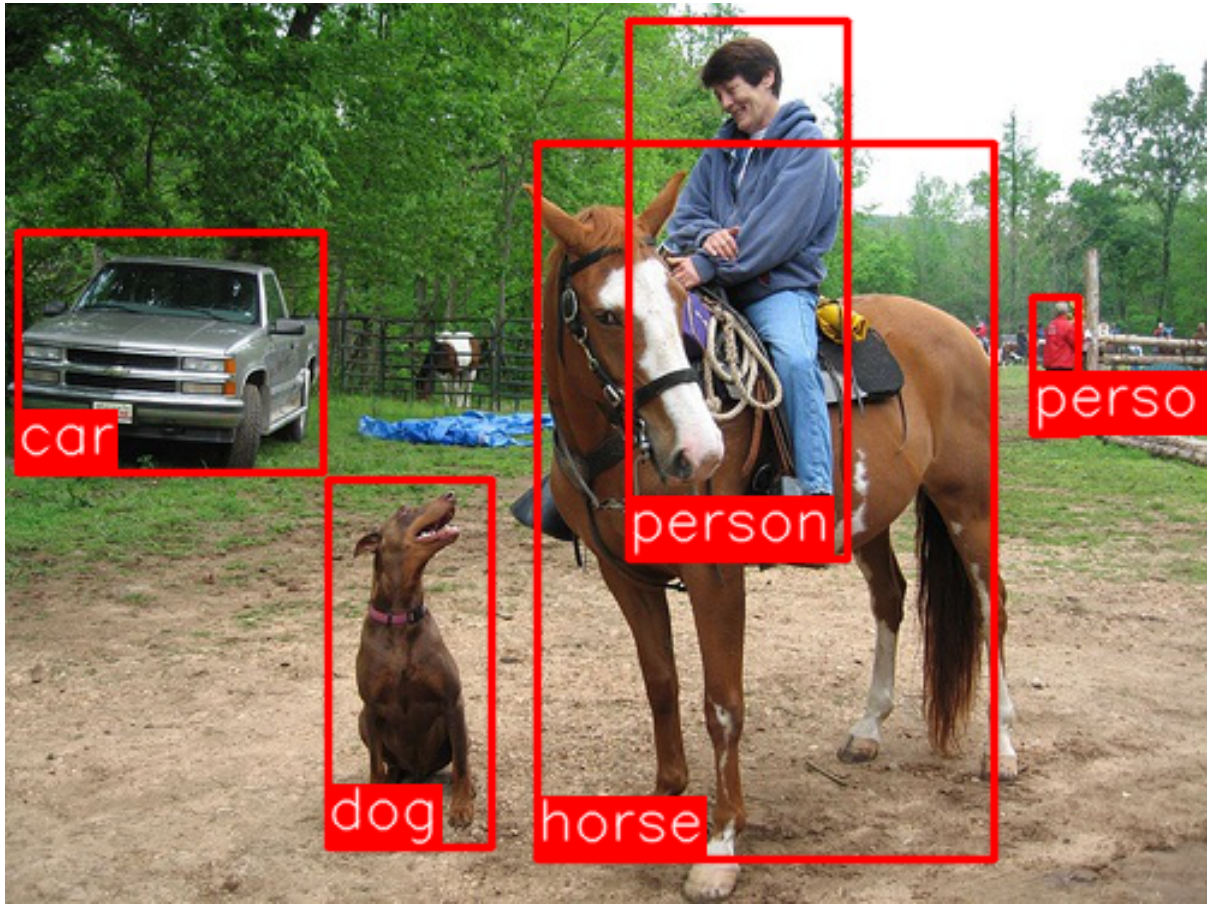
Clasificación: Algunas veces no es tan simple

Localización



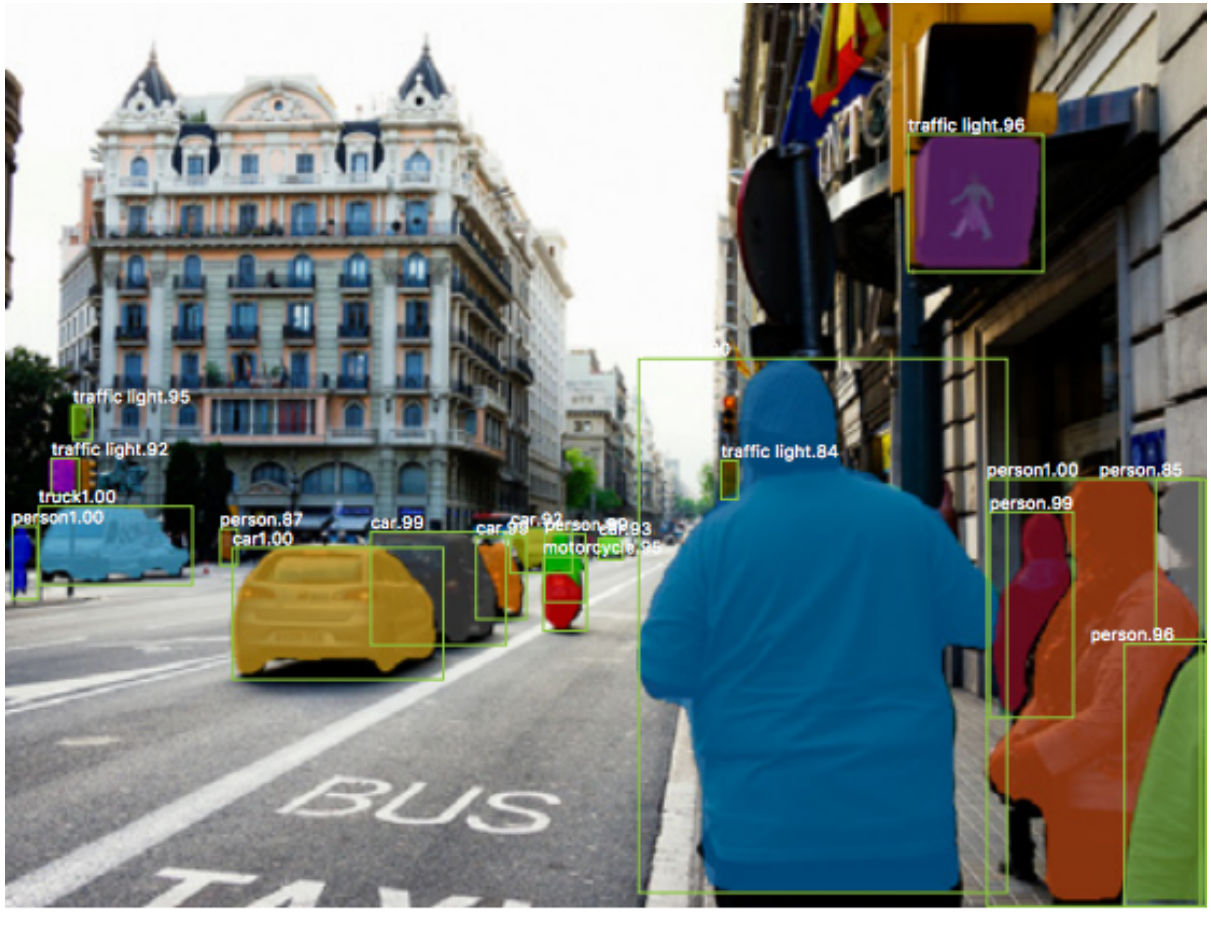
https://www.pyimagesearch.com/2016/06/20/detecting-cats-in-images-with-opencv/ (https://www.pyimagesearch.com/2016/06/20/detecting-cats-in-images-with-opencv/)

Detección de objetos



https://arxiv.org/abs/1311.2524 (https://arxiv.org/abs/1311.2524)

Segmentación



https://arxiv.org/abs/1703.06870 (https://arxiv.org/abs/1703.06870)

Una solución *"Out of the box"*

Google ofrece una serie de modelos implementados en `TensorFlow`, entre ellos uno que permite detectar objetos.

Se puede descargar desde acá:

https://github.com/tensorflow/models/tree/master/research/object_detection (https://github.com/tensorflow/models/tree/master/research/object_detection)

Tiene un tutorial para usarlo *"de fábrica"*, y otro para para entrenar nuestro propio reconocedor de objetos.

In [ ]:

```
%matplotlib inline
import numpy as np
import os
import tensorflow as tf
from matplotlib import pyplot as plt

from object_detection_helpers import open_image, load_labels, visualize_imag
e
```

Bajamos el modelo y configuramos algunos paths

El modelo se puede bajar desde

http://download.tensorflow.org/models/object_detection/ssd_mobilenet_v1_coco_11_0
(http://download.tensorflow.org/models/object_detection/ssd_mobilenet_v1_coco_11_0

In [ ]:
```
base_path = 'object_detection'
MODEL_NAME = 'ssd_mobilenet_v1_coco_11_06_2017'
PATH_TO_CKPT = os.path.join('object_detection', MODEL_NAME, 'frozen_inferenc
e_graph.pb')

# Leemos el modelo que bajamos de internet
detection_graph = tf.Graph()
with detection_graph.as_default():
    od_graph_def = tf.GraphDef()
    with tf.gfile.GFile(PATH_TO_CKPT, 'rb') as fid:
        serialized_graph = fid.read()
        od_graph_def.ParseFromString(serialized_graph)
        tf.import_graph_def(od_graph_def, name='')

# Leemos las clases con las que fue entrenado el modelo
category_index = load_labels(os.path.join(base_path, 'data', 'mscoco_label_m
ap.pbtxt'), 90)
```

Leemos algunas imágenes...

In [ ]:
```
TEST_IMAGE_PATHS = [os.path.join('test_images', x) for x in os.listdir('test
_images')]
```

Datos en crudo

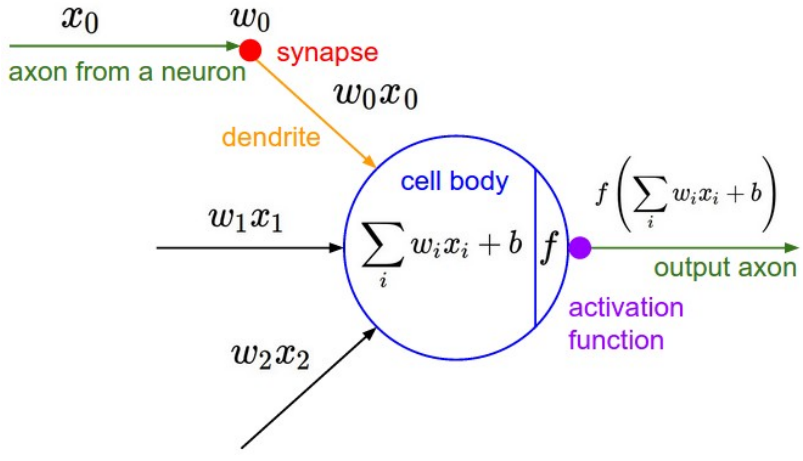
In [ ]:
```
scores
# visualize_image(image_np.copy(), boxes, classes, scores, category_index, m
in_score_thresh=0.01)
```

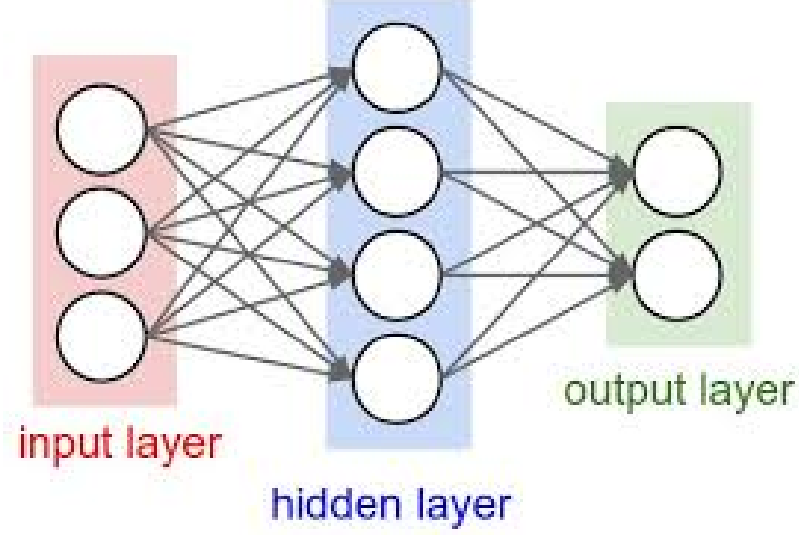
Los bloques

Clasificador de imágenes

Detector / proponedor de regiones

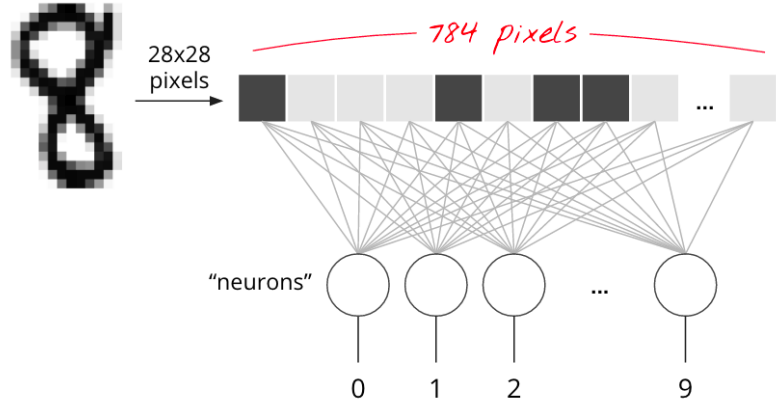
Redes neuronales





http://cs231n.github.io/neural-networks-1/ (http://cs231n.github.io/neural-networks-1/)

Redes neuronales: Noción básica de funcionamiento



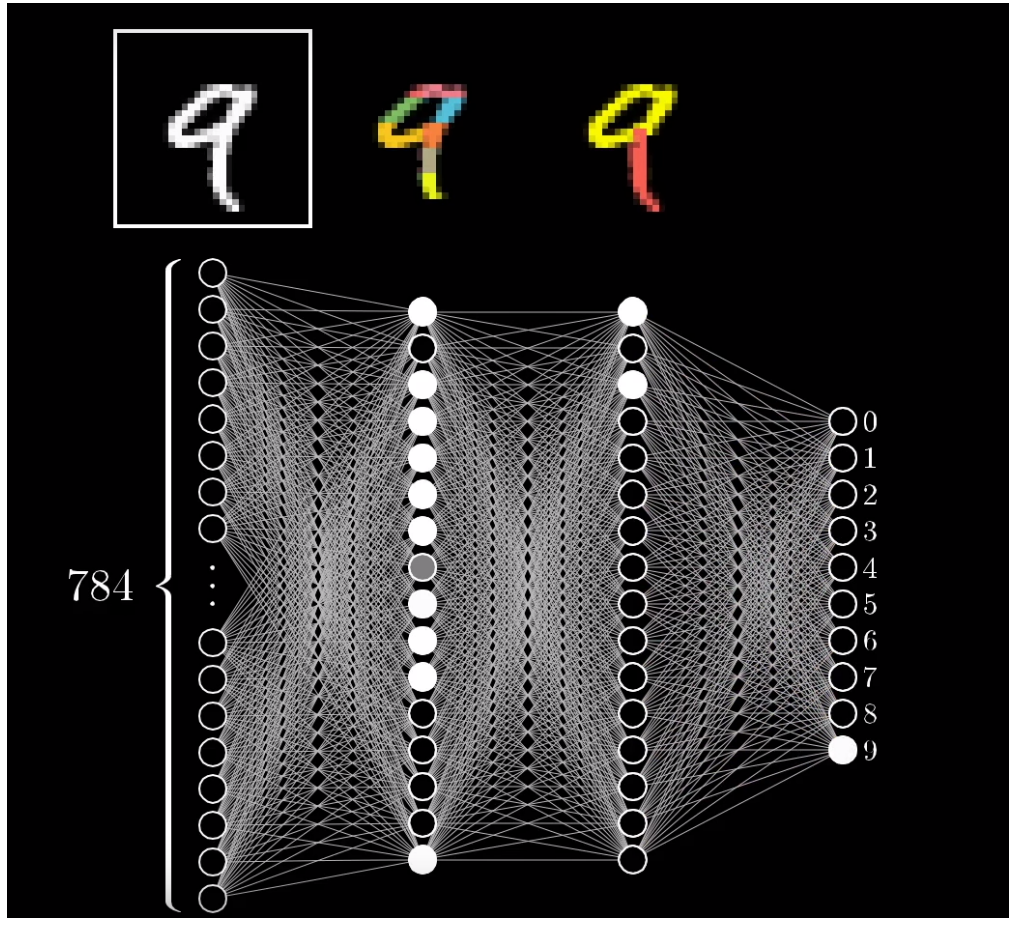
https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0
(https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist/#0)

In [ ]:
```
%matplotlib inline
from object_detection_helpers import get_mnist_demo, show_weights
model, X, y = get_mnist_demo()
```

In [ ]:
```
show_weights(model)
```

In [ ]:
```
for i in range(6):
    model.fit(X, y, epochs=1)
    show_weights(model)
```

Redes neuronales: Noción básica de funcionamiento



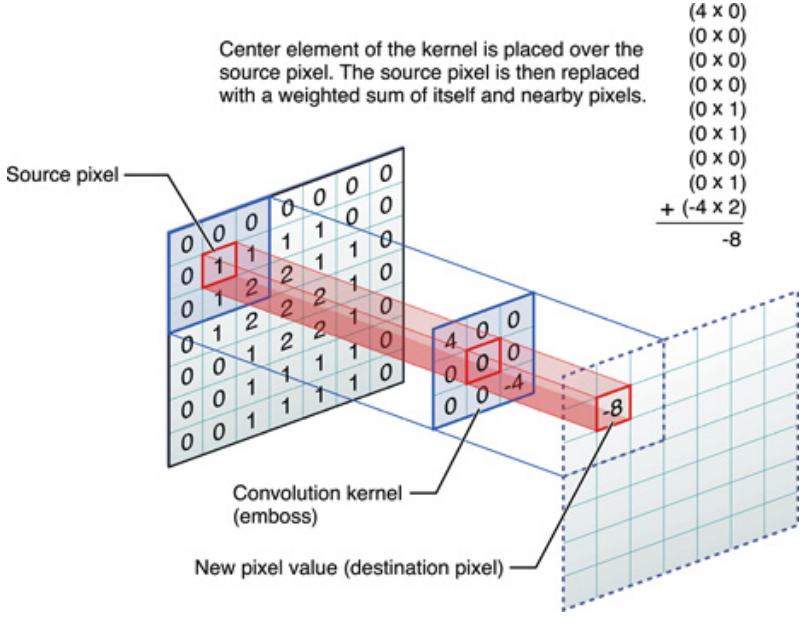
https://www.youtube.com/watch?v=aircAruvnKk (https://www.youtube.com/watch?v=aircAruvnKk)

Deep Learning

The hierarchy of concepts enables the computer to learn complicated concepts bybuilding them out of simpler ones. If we draw a graph showing how these concepts are built on top of each other, the graph is deep, with many layers. For this reason,we call this approach to AI deep learning.

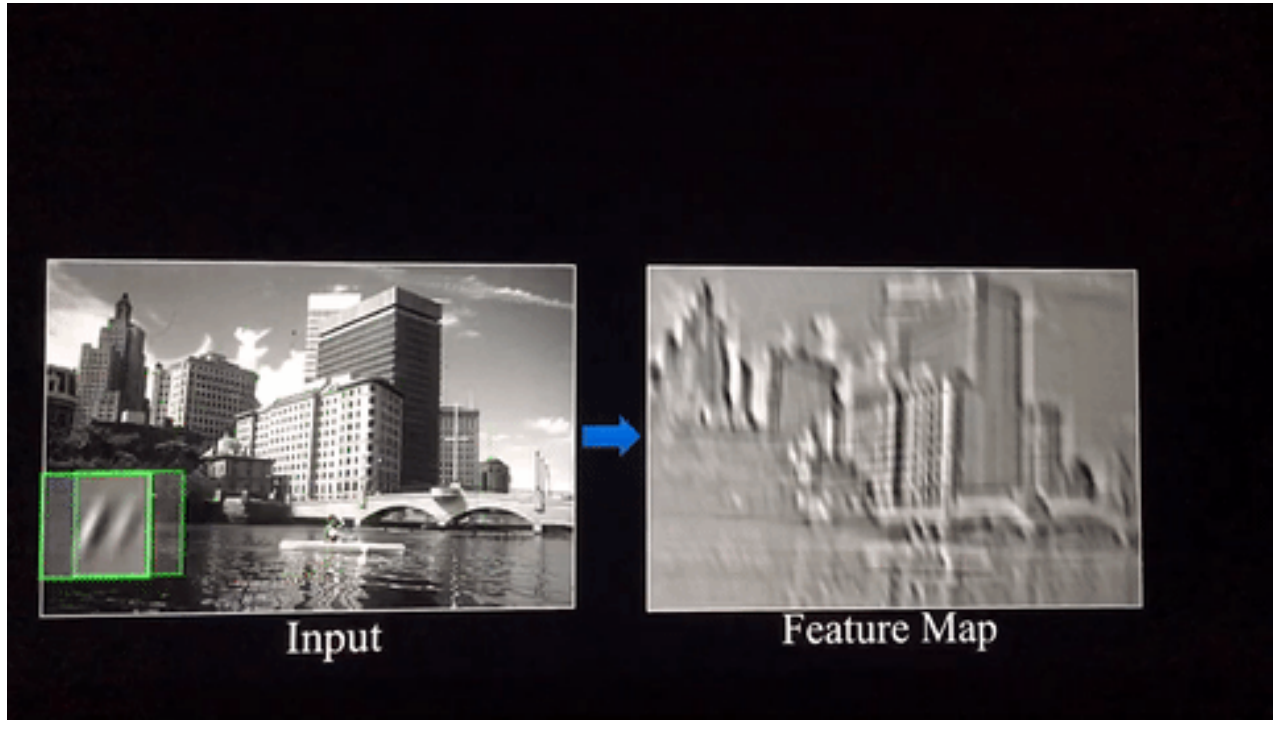
Goodfellow et al 2016

Convoluciones



https://developer.apple.com/library/content/documentation/Performance/Conceptual/v
(https://developer.apple.com/library/content/documentation/Performance/Conceptual/

Convoluciones



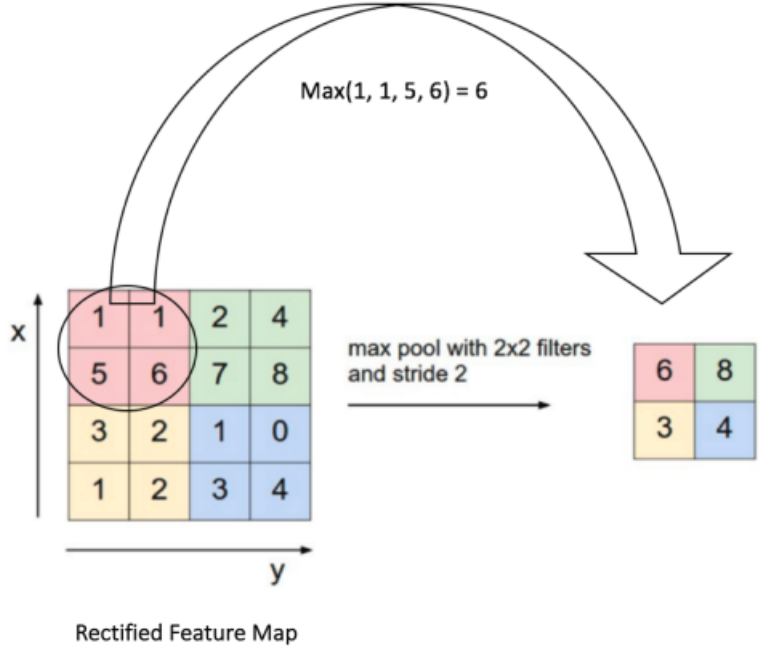
https://www.kdnuggets.com/2016/11/intuitive-explanation-convolutional-neural-networks.html/2 (https://www.kdnuggets.com/2016/11/intuitive-explanation-convolutional-neural-networks.html/2)

Convoluciones

Parámetros más comunes

- **filters:** cantidad de filtros
- **kernel_size:** tamaño del kernel
- **strides:** la cantidad de pasos que muevo el kernel
- **padding:** agrega ceros en los bordes para mantener el tamaño original
- **activation:** función de activación aplicada `pixel a pixel`

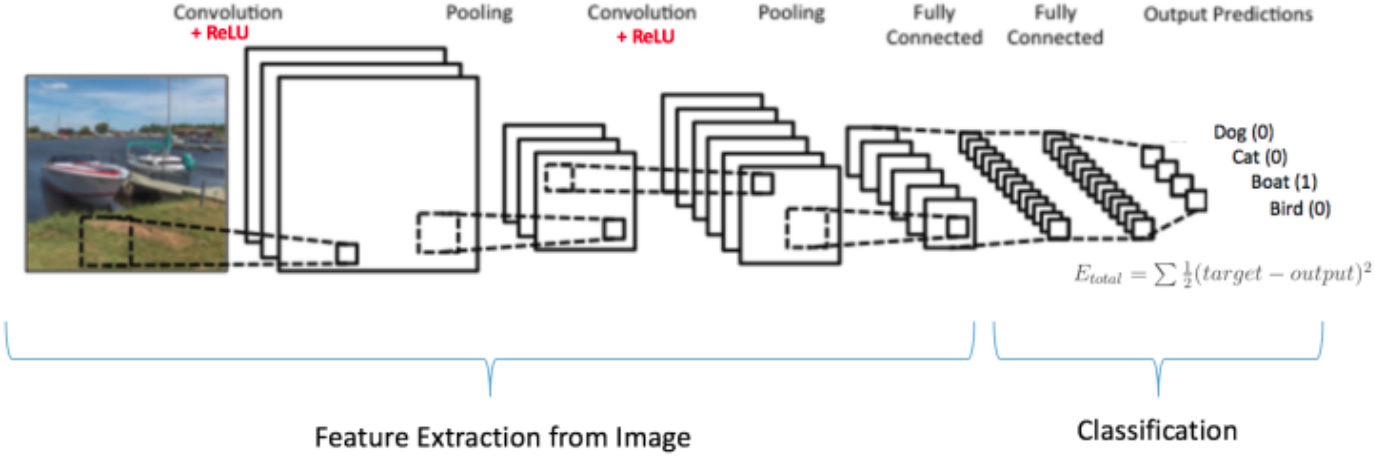
Pooling



- reduce el tamaño de los filtros
- brinda robustez
- features *equivariantes*

http://textminingonline.com/dive-into-tensorflow-part-v-deep-mnist (http://textminingonline.com/dive-into-tensorflow-part-v-deep-mnist)

Redes neuronales



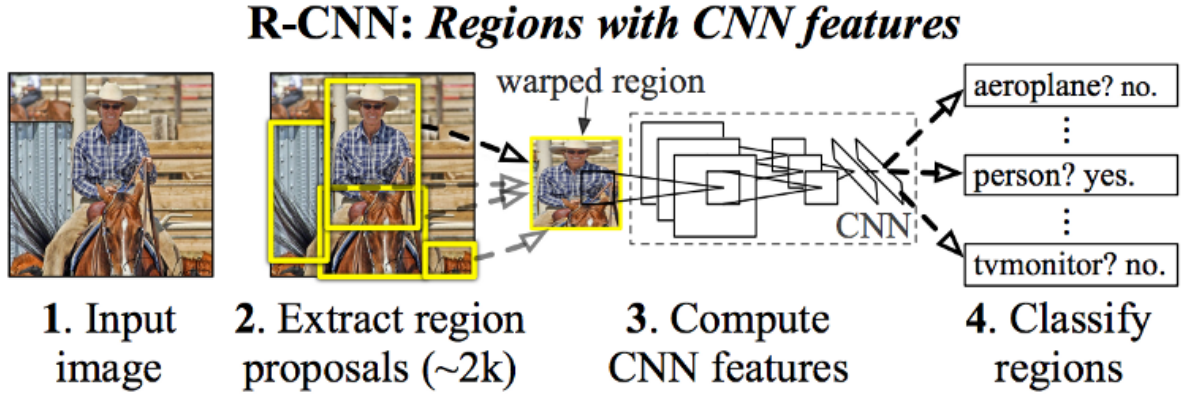
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/comment-page-2/ (https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/comment-page-2/)

Tenemos un clasificador, ¿y ahora qué?

Sliding windows

https://matthewearl.github.io/2016/05/06/cnn-anpr/
(https://matthewearl.github.io/2016/05/06/cnn-anpr/)

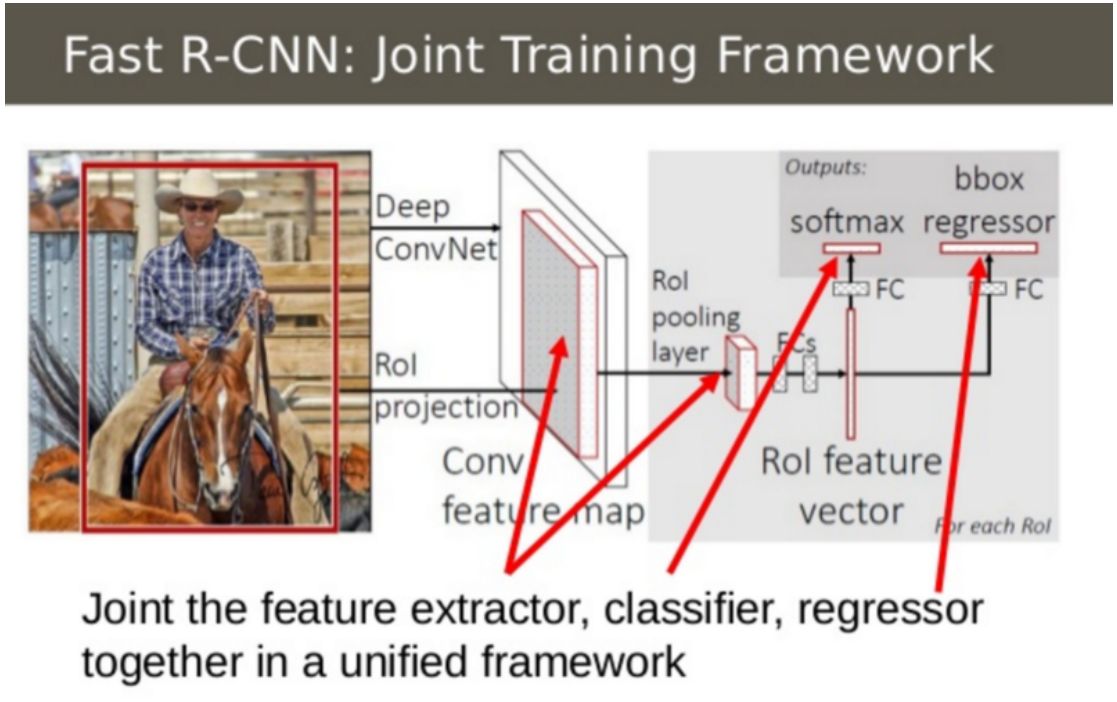
2014: R-CNN (Region Convolutional Neural Network)



https://arxiv.org/abs/1311.2524 (https://arxiv.org/abs/1311.2524)

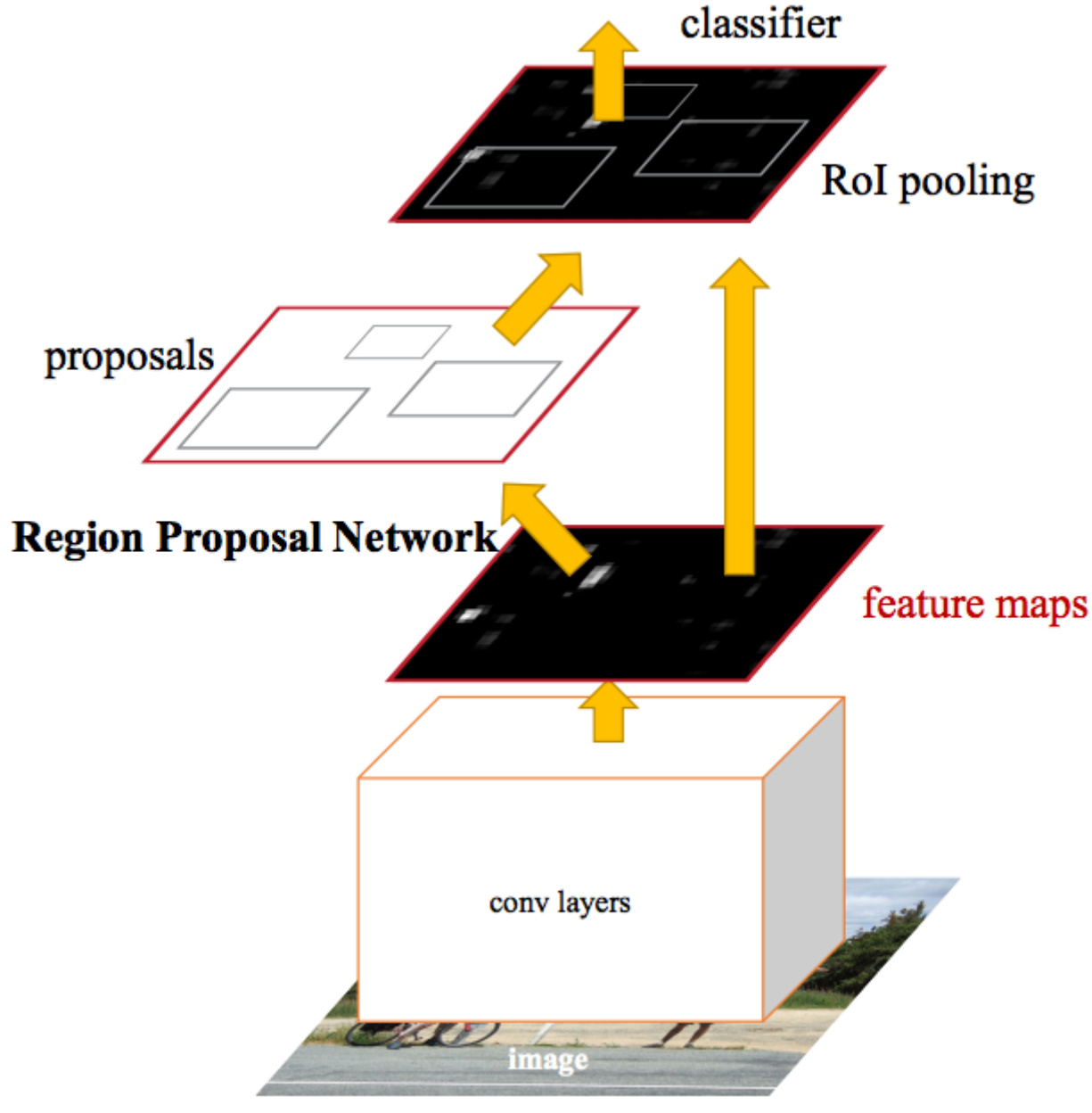
1. Selective search para proponer regiones (~2000 por imagen)
2. AlexNet -> SVM para clasificar
3. Linear regression para mejorar el box

2015: Fast R-CNN



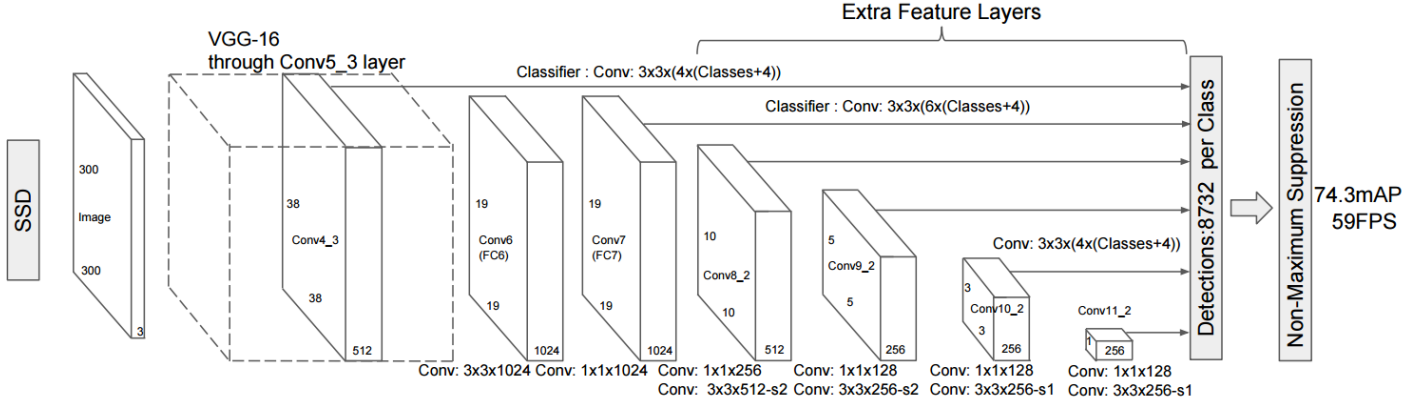
https://arxiv.org/abs/1504.08083 (https://arxiv.org/abs/1504.08083)

2016: Faster R-CNN



https://arxiv.org/abs/1506.01497 (https://arxiv.org/abs/1506.01497)

2016: SSD (Single Shot Detector)



https://arxiv.org/abs/1512.02325 (https://arxiv.org/abs/1512.02325)

Resumen para atacar un problema

- Probar solución *Out of the box*
- Re-entrenar el modelo para que se ajuste a las clases del problema
- Elegir el modelo en base a las necesidades de tiempo y precisión que tengamos

Preguntas?

Gracias!

Mis datos de contacto:


arielrossanigo@gmail.com


@arielrossanigo


https://github.com/arielrossanigo