

The background image shows the Colosseum in Rome, Italy, under a clear sky. In the foreground, there is a paved walkway with a metal railing. Several people are walking along the path. Overlaid on the image are numerous bounding boxes for object detection. Red boxes are labeled 'OBJECT' and are placed around various architectural elements like statues, lampposts, and parts of the Colosseum. Yellow boxes are labeled 'PERSON' and are placed around the people walking. The text 'You Only Look Once' is prominently displayed in the center-left of the image.

You Only Look Once

Object Detection task

Nicolás Dominutti

Carlos Suárez Gurruchaga

Hernán Telechea

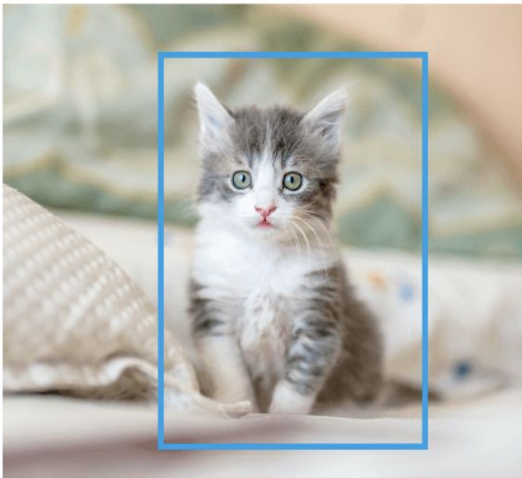
Detección de objetos

classification

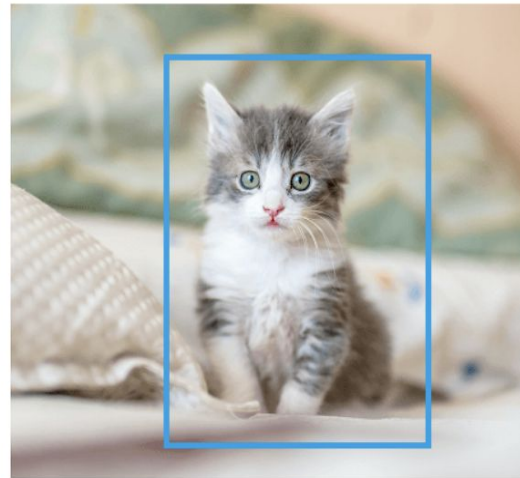


cat

localization



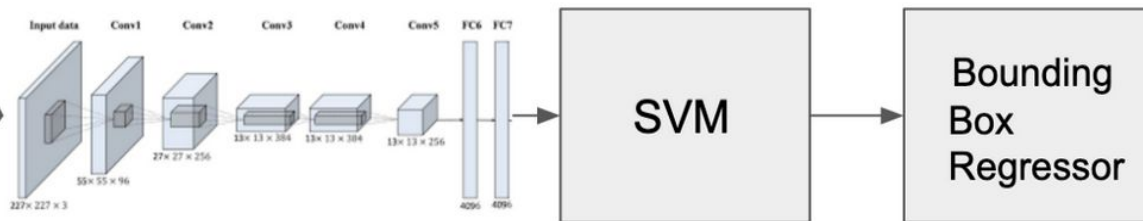
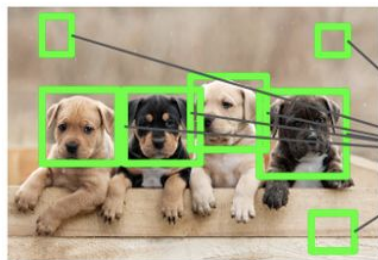
detection



cat

Estado del arte B.Y. (before YOLO)

- Deformable parts models (DPM)
- R-CNN
- Fast R-CNN



1) Region proposals with selective search

2) Feature extraction with CNN

3) Classify features with a SVM

4) Improve the bounding box

The R-CNN system

YOLO - Introducción

You Only Look Once: Unified, Real-Time Object Detection

Joseph Redmon*, Santosh Divvala*[†], Ross Girshick[¶], Ali Farhadi*[†]

University of Washington*, Allen Institute for AI[†], Facebook AI Research[¶]

<http://pjreddie.com/yolo/>

Abstract

We present YOLO, a new approach to object detection. Prior work on object detection repurposes classifiers to perform detection. Instead, we frame object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. A single neural network predicts bounding boxes and class probabilities directly from full images in one evaluation. Since the whole detection pipeline is a single network, it can be optimized end-to-end

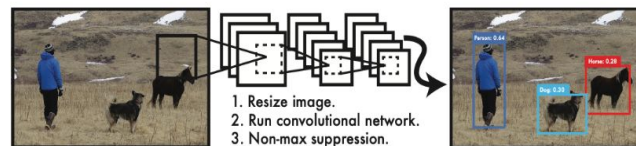


Figure 1: The YOLO Detection System. Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448×448 , (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

YOLO - Mejoras

- Muy buena performance
- Tiempo de ejecución muy rápido (aplicación en tiempo real)

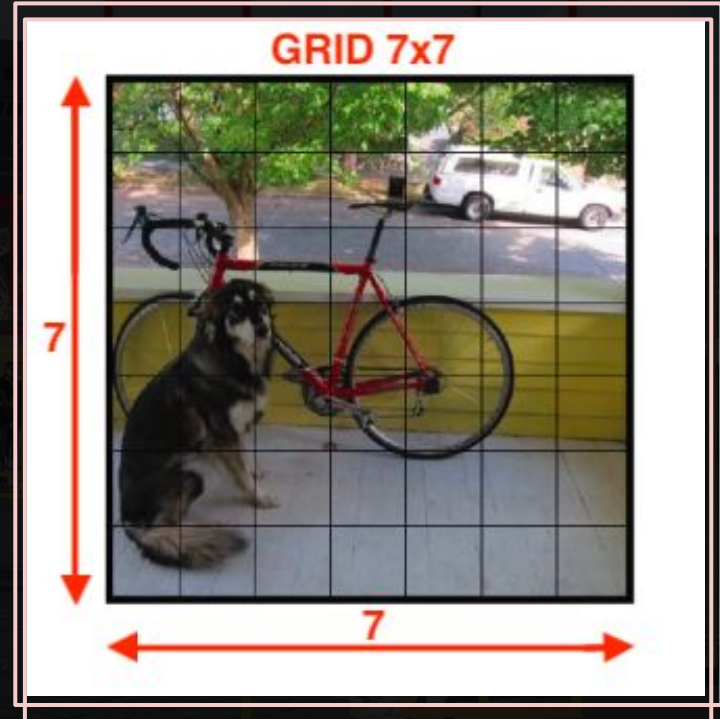
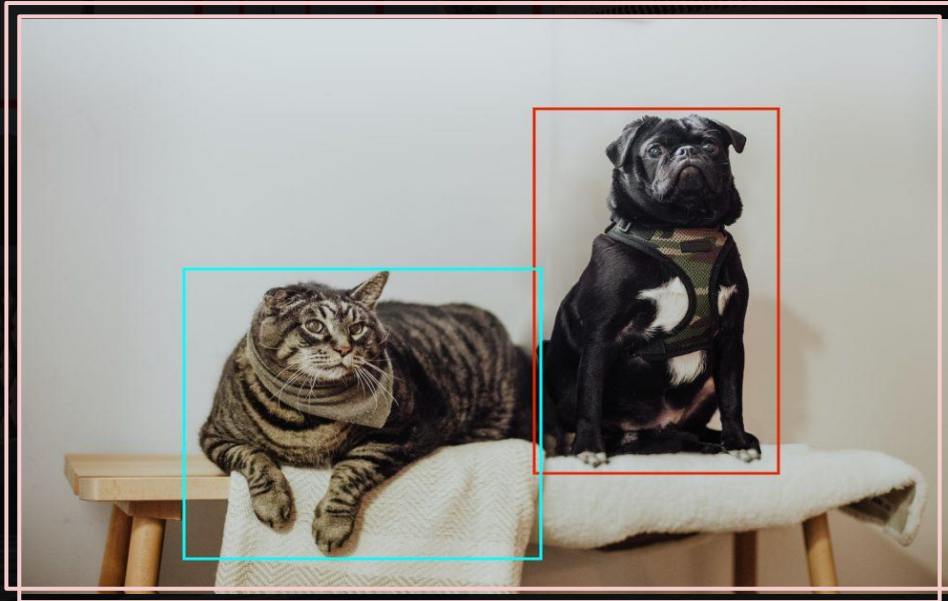
Real-Time Detectors	Train	mAP	FPS
100Hz DPM [31]	2007	16.0	100
30Hz DPM [31]	2007	26.1	30
Fast YOLO	2007+2012	52.7	155
YOLO	2007+2012	63.4	45

*on Pascal VOC 2007

Less Than Real-Time	Train	mAP	FPS
Fastest DPM [38]	2007	30.4	15
R-CNN Minus R [20]	2007	53.5	6
Fast R-CNN [14]	2007+2012	70.0	0.5
Faster R-CNN VGG-16[28]	2007+2012	73.2	7
Faster R-CNN ZF [28]	2007+2012	62.1	18
YOLO VGG-16	2007+2012	66.4	21

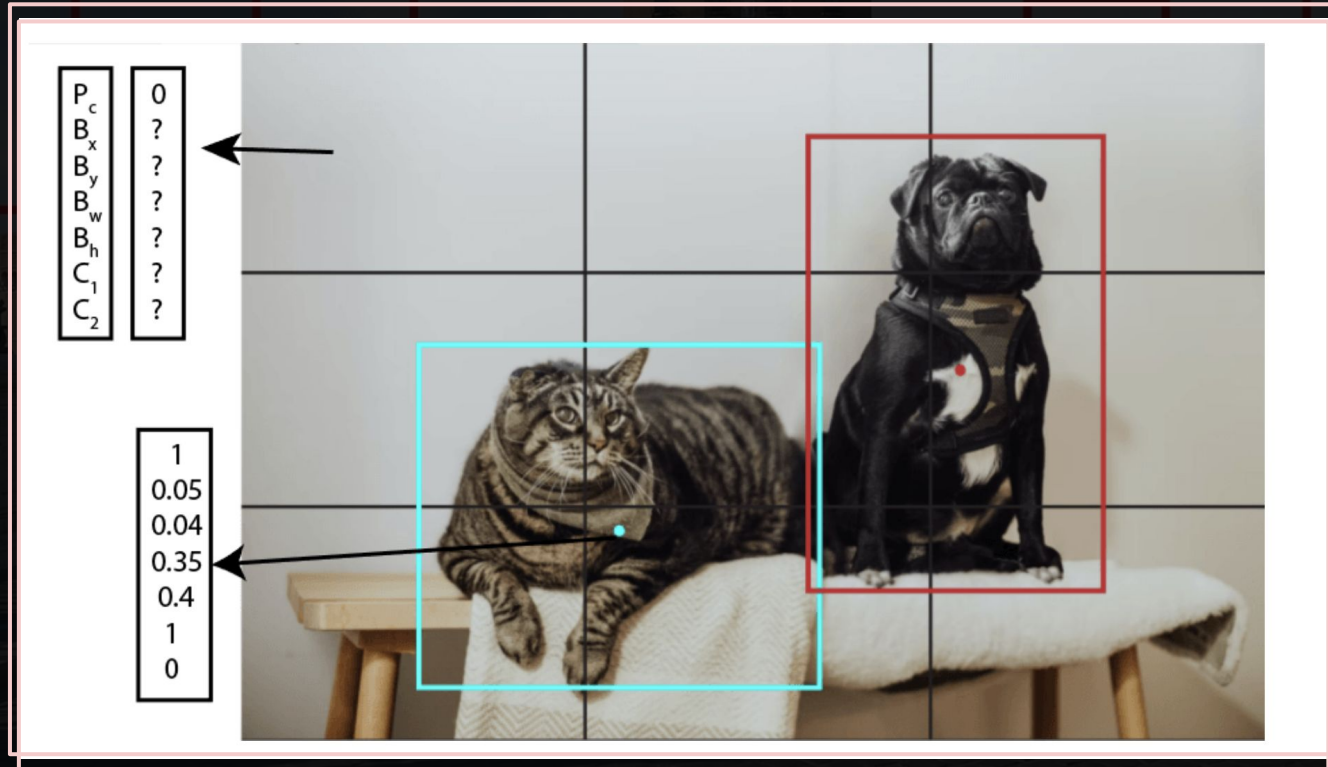
YOLO - Algunas definiciones previas

- Grid
- Bounding box



YOLO - Algunas definiciones previas

- Output (2 categorías)

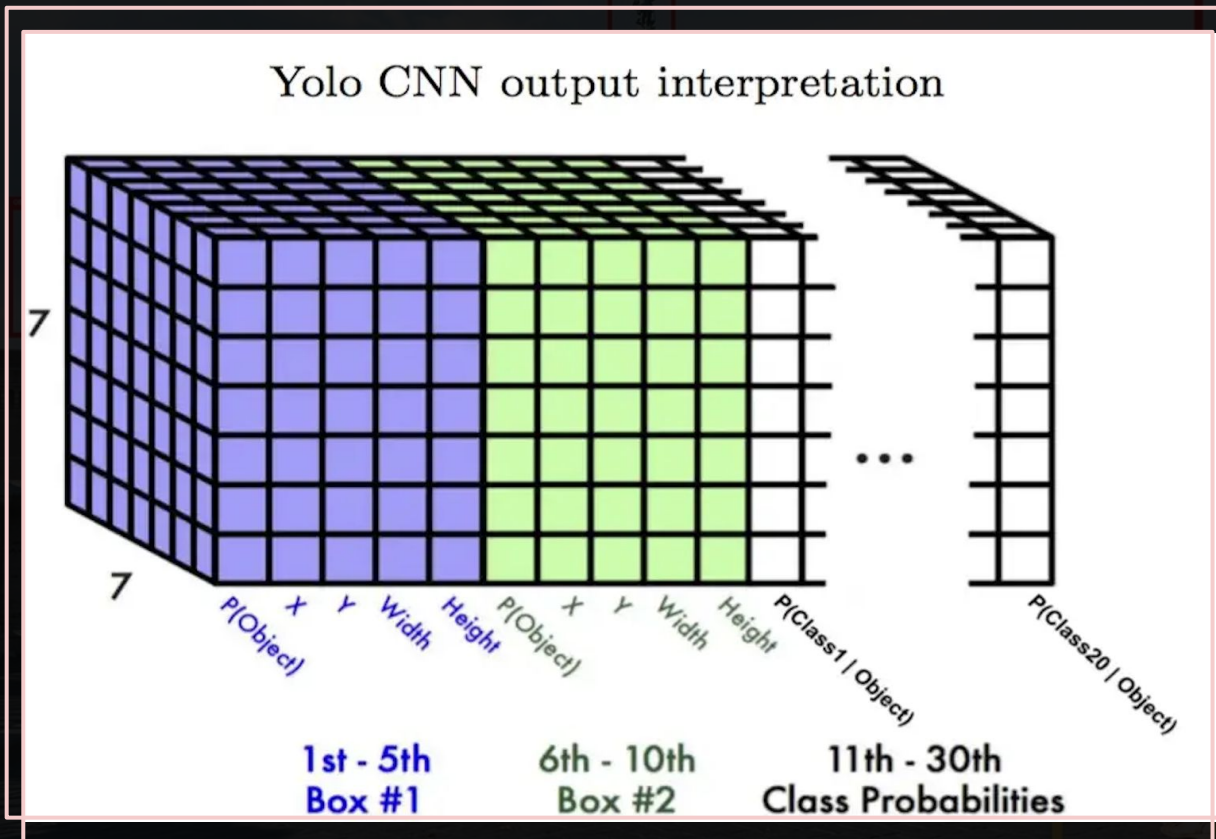


24 convolutional layers followed by 2 fully connected layers.

Diagram illustrating the architecture of a deep convolutional neural network (CNN) with 24 convolutional layers followed by 2 fully connected layers. The input is a 448x448x3 image. The network structure is as follows:

- Convolution Layer 1:** 7x7x64-s-2, Maxpool Layer 2x2-s-2. Output: 192x112x3.
- Convolution Layer 2:** 3x3x192, Maxpool Layer 2x2-s-2. Output: 256x56x3.
- Convolution Layers 3-6:** 1x1x128, 3x3x256, 1x1x256, 3x3x512, Maxpool Layer 2x2-s-2. Output: 512x28x3.
- Convolution Layers 7-9:** 1x1x256, 3x3x512, 1x1x512, 3x3x1024, Maxpool Layer 2x2-s-2. Output: 1024x14x3.
- Convolution Layers 10-12:** 1x1x512, 3x3x1024, 3x3x1024. Output: 1024x7x3.
- Convolution Layers 13-15:** 1x1x512, 3x3x1024, 3x3x1024. Output: 1024x7x3.
- Convolution Layers 16-18:** 3x3x1024, 3x3x1024. Output: 1024x7x3.
- Convolution Layer 19:** 3x3x1024. Output: 4096.
- Convolution Layer 20:** 3x3x1024. Output: 30.

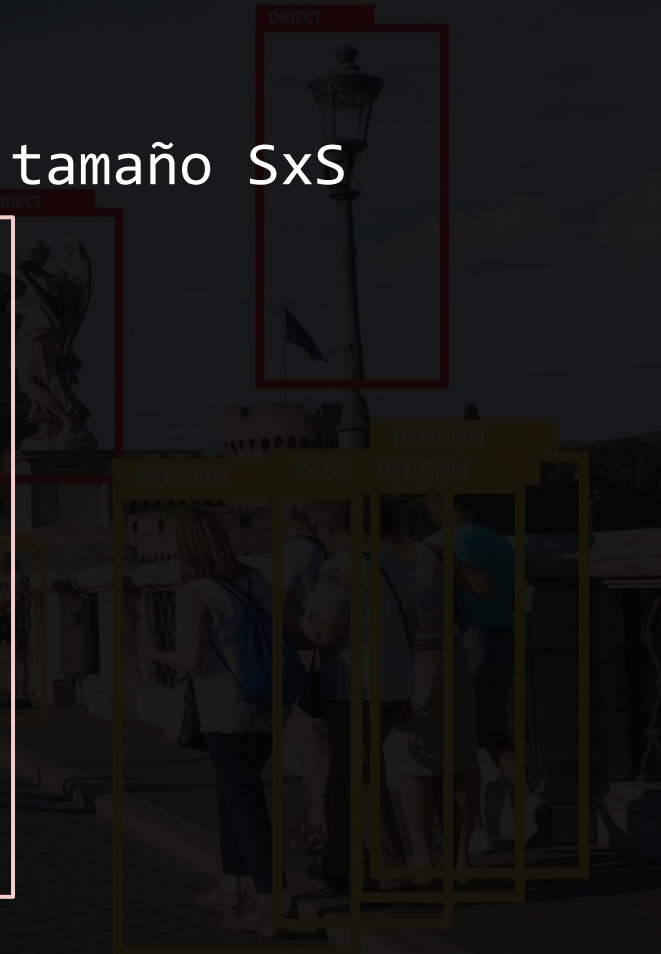
YOLO - Output



*Pascal VOC object detection DATASET

Algoritmo original de YOLO

1. Divide una imagen en un grid de tamaño $S \times S$



Algoritmo original de YOLO

1. Divide una imagen en un grid de tamaño $S \times S$



- Cada celda genera 2 bounding boxes
- Cada bounding box tiene su propia métrica de confianza
- En caso de haber un objeto en una celda, se predice la clase

Algoritmo original de YOLO

2. Evalúa en qué grid cae el centro del objeto



- Grid cell encargado de la predicción del objeto
- Tiene 2 bounding boxes, solo uno debe ser usado para entrenar

¿Cuál?

Algoritmo original de YOLO

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Intersection over Union / Índice de Jaccard

- Cada celda tiene más de una bounding box
- Solo uno puede ser utilizado para predecir la clase
- Elegimos el bounding box con mayor índice vs el ground truth box

Función de pérdida

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \\ & + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2 \\ & + \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \end{aligned}$$

Función de pérdida

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

Función de pérdida

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

Función de pérdida

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2 \end{aligned}$$

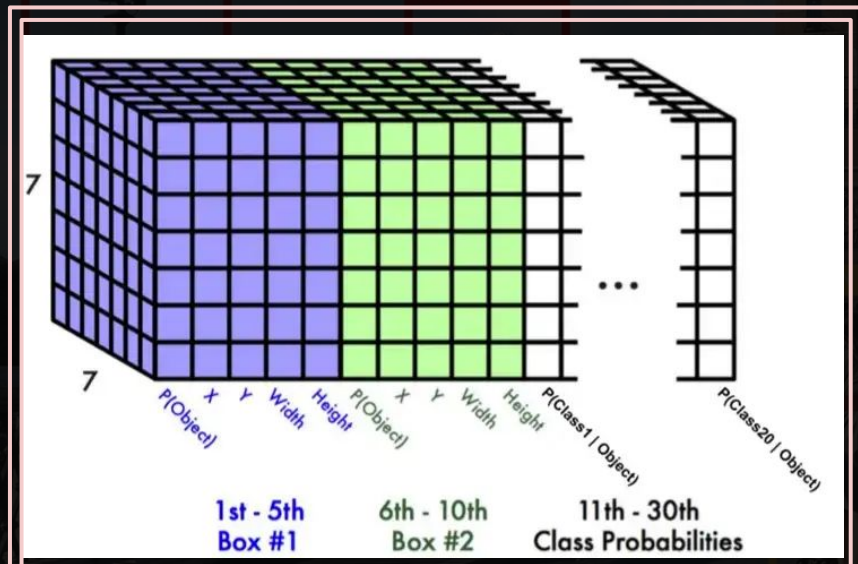
Función de pérdida

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \\ & + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2 \end{aligned}$$

Función de pérdida

$$\begin{aligned} & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{obj}} (C_i - \hat{C}_i)^2 \\ & + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B \mathbb{1}_{ij}^{\text{noobj}} (C_i - \hat{C}_i)^2 \\ & + \sum_{i=0}^{S^2} \mathbb{1}_i^{\text{obj}} \sum_{c \in \text{classes}} (p_i(c) - \hat{p}_i(c))^2 \end{aligned}$$

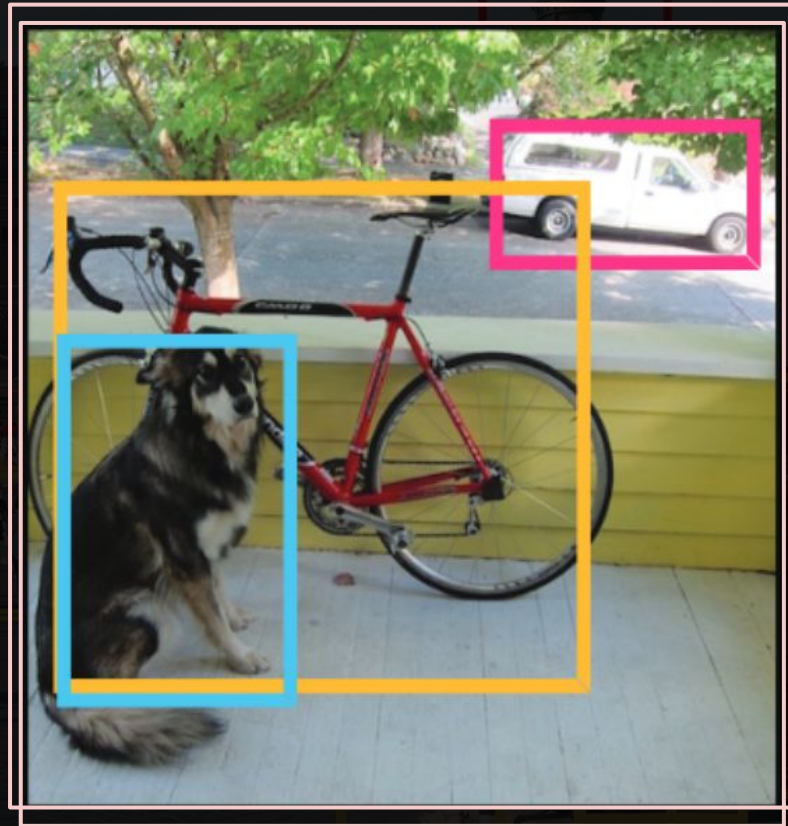
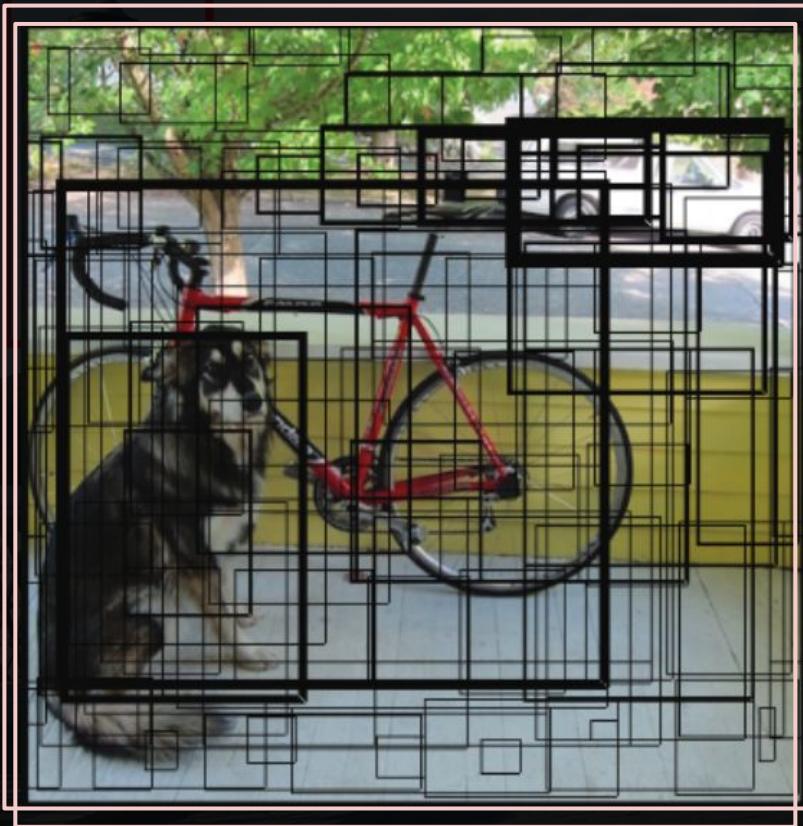
Predicción



$$7*7*2 = 98$$

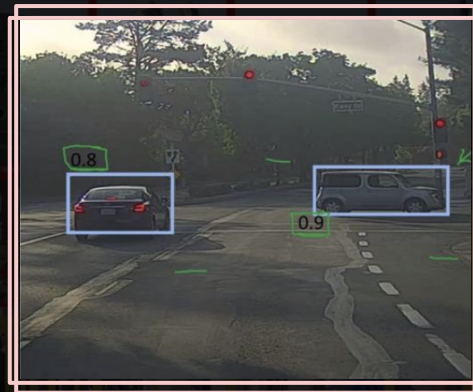
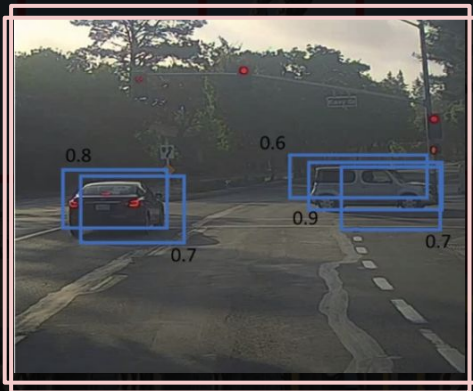


Predicción



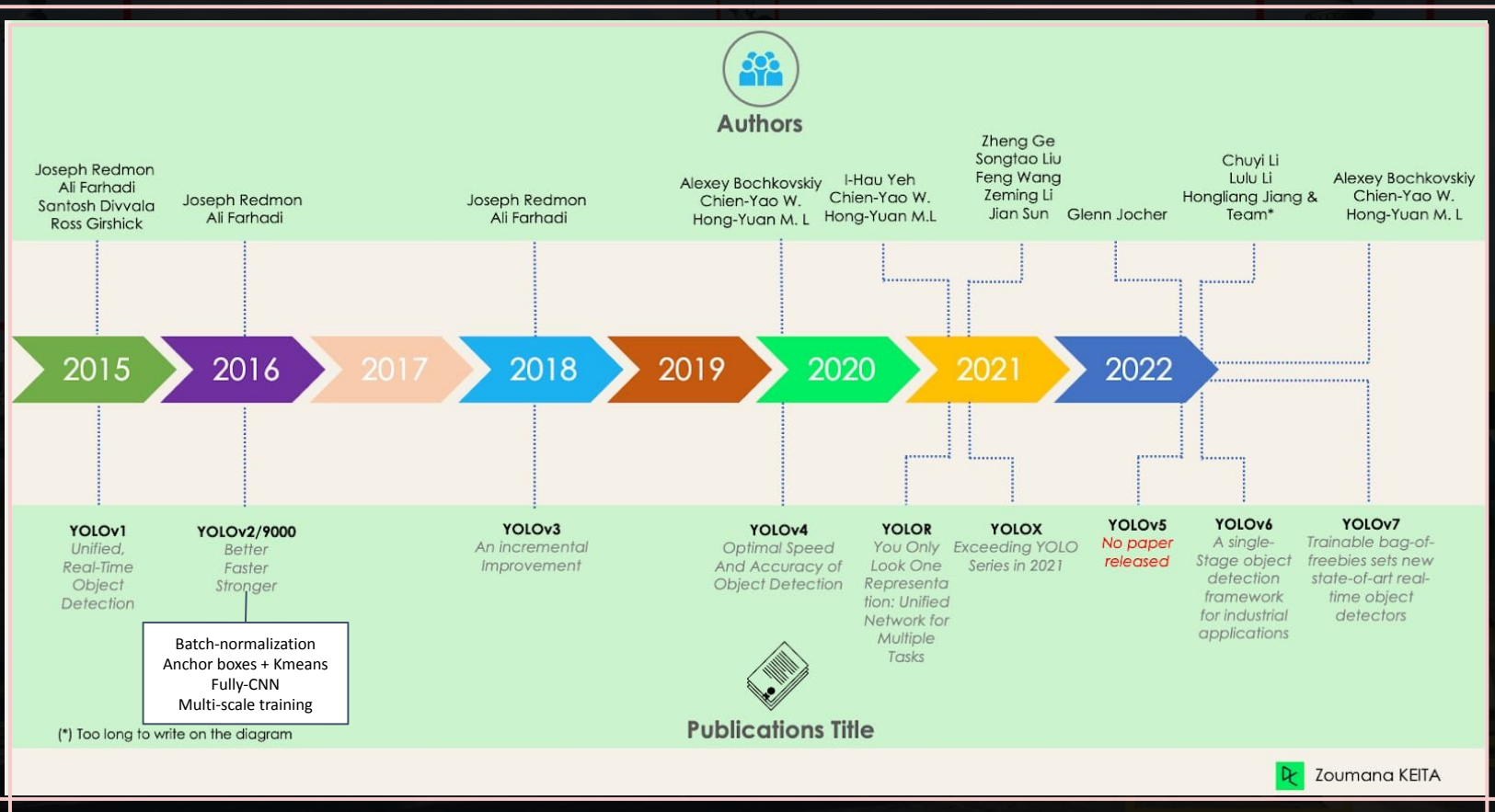
Predicción | Non-max suppression

- 1) Hacemos un pase forward de la imagen en nuestra red
- 2) Seteamos un threshold de confianza

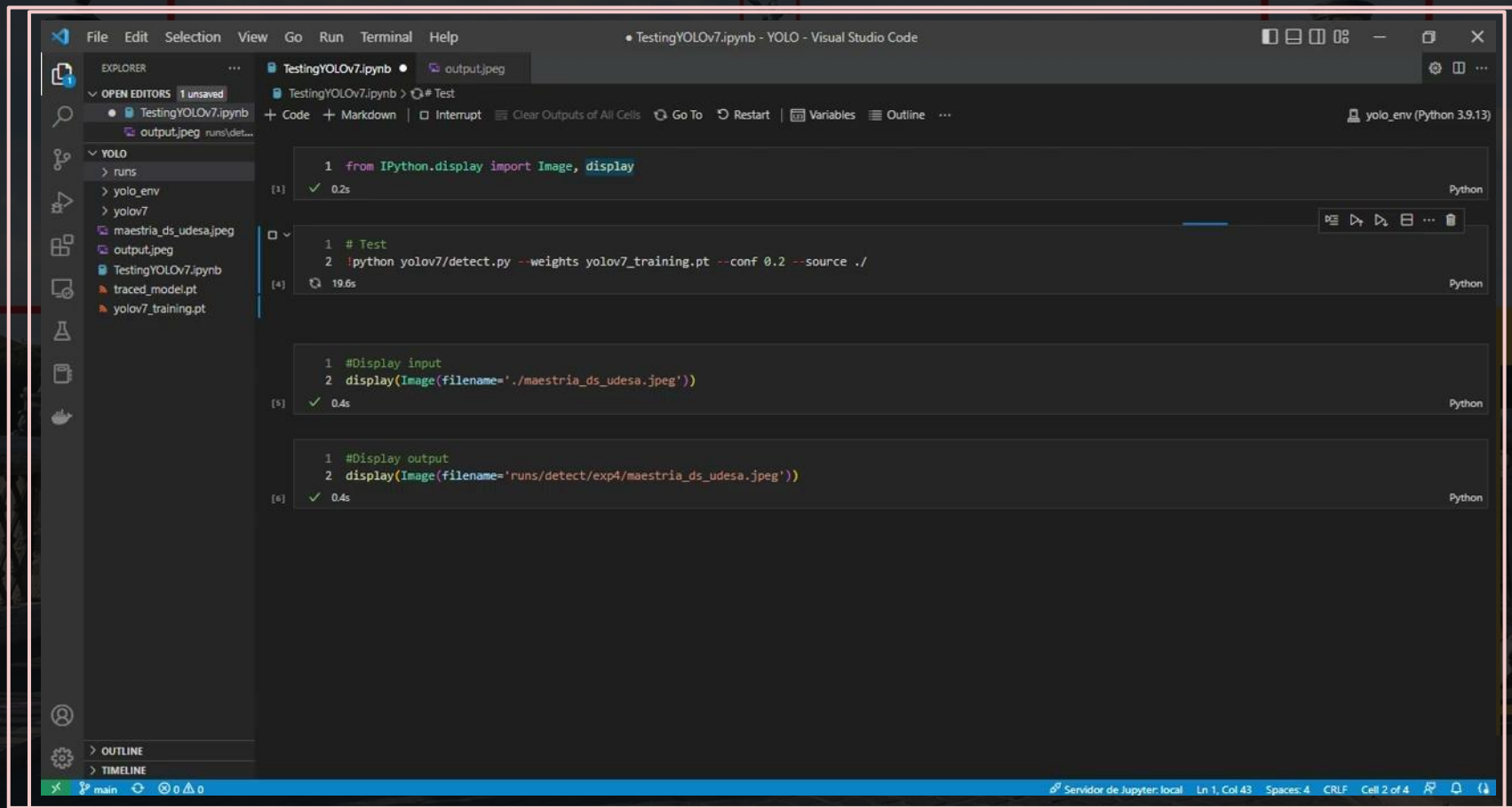


- 3) Non-Max Supression:
 - a) Encontramos el bounding-box con mayor confianza
 - b) Eliminamos todos los bounding-box que tengan alto IOU con este
 - c) Volvemos a encontrar el siguiente bounding-box con mayor confianza (dejando de lado el primero) y repetimos hasta converger

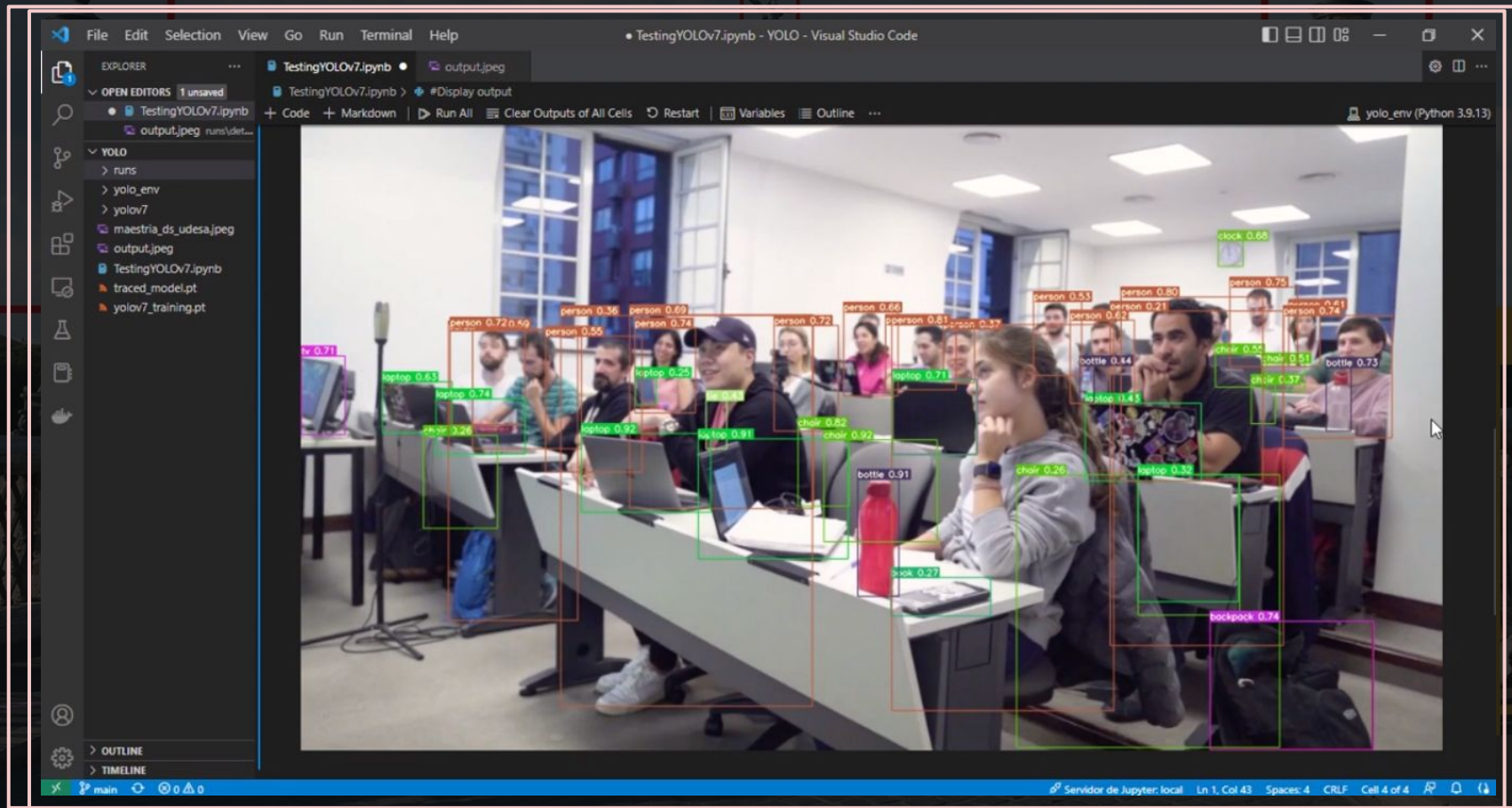
YOLO en el tiempo












YOTO (You Only Try Once)



YOTO (You Only Try Once)



Referencias

-  You Only Look Once: Unified, Real-Time Object Detection | Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi
-  YOLO9000: Better, Faster, Stronger | Joseph Redmon, Ali Farhadi
-  YOLOv3: An Incremental Improvement | Joseph Redmon, Ali Farhadi
-  YOLOv4: Optimal Speed and Accuracy of Object Detection | Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao
-  YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors | Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao
-  Deep Learning specialization (4th course, CNN) – Deeplearning.ai - Coursera
-  <https://www.datacamp.com/blog/yolo-object-detection-explained>
-  <https://medium.com/oracledevs/final-layers-and-loss-functions-of-single-stage-detectors-part-1-4abbfa9aa71c>
-  <https://naokishibuya.medium.com/yolo-v1-f9312337cf71>



OBJECT

OBJECT

OBJECT

OBJECT

¡Muchas gracias!

PERSON
PERS PERSON

The background image shows the Colosseum in Rome, Italy, under a clear sky. In the foreground, there is a paved walkway with a metal railing. Several people are walking along the path. Overlaid on the image are numerous bounding boxes for object detection. Red boxes are labeled 'OBJECT' and include a street lamp on the left, a statue on a pedestal in the middle ground, and another statue on the right. Yellow boxes are labeled 'PERSON' and include many individuals walking on the path. The text 'You Only Look Once' is prominently displayed in the upper left, with 'Object Detection task' below it.

You Only Look Once

Object Detection task