Detección de objetos II

Visión por computador II

Juan Carlos Arbeláez.

Contenido

1. Single Shot Detector (SSD)
2. You Only Look Once (Yolo)
3. Mean Average Precision (mAP)
4. Ejercicio

Algoritmos

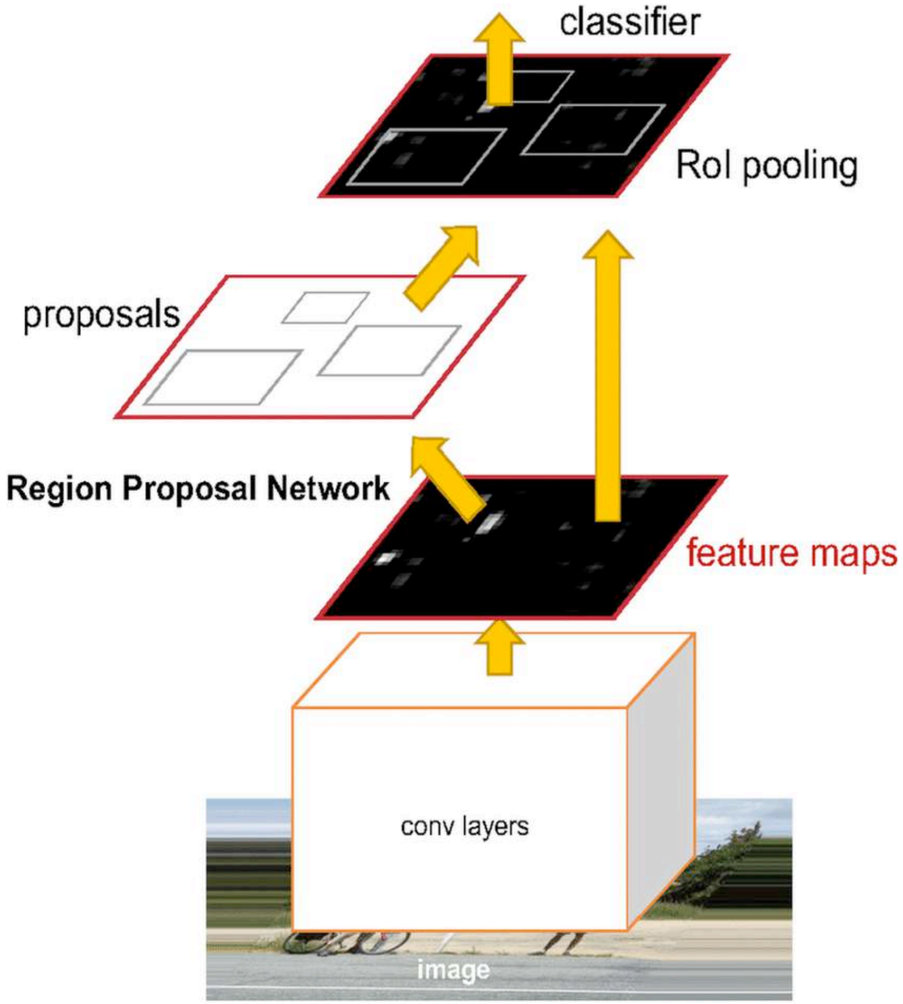
- Existen muchos más algoritmos (ya vimos RCNN) y cada uno tiene sus pros y sus contras

- 3 Algoritmos son los más conocidos en la industria (A nivel introductorio)

    1. Faster R-CNN

    2. SSD

    3. YOLO

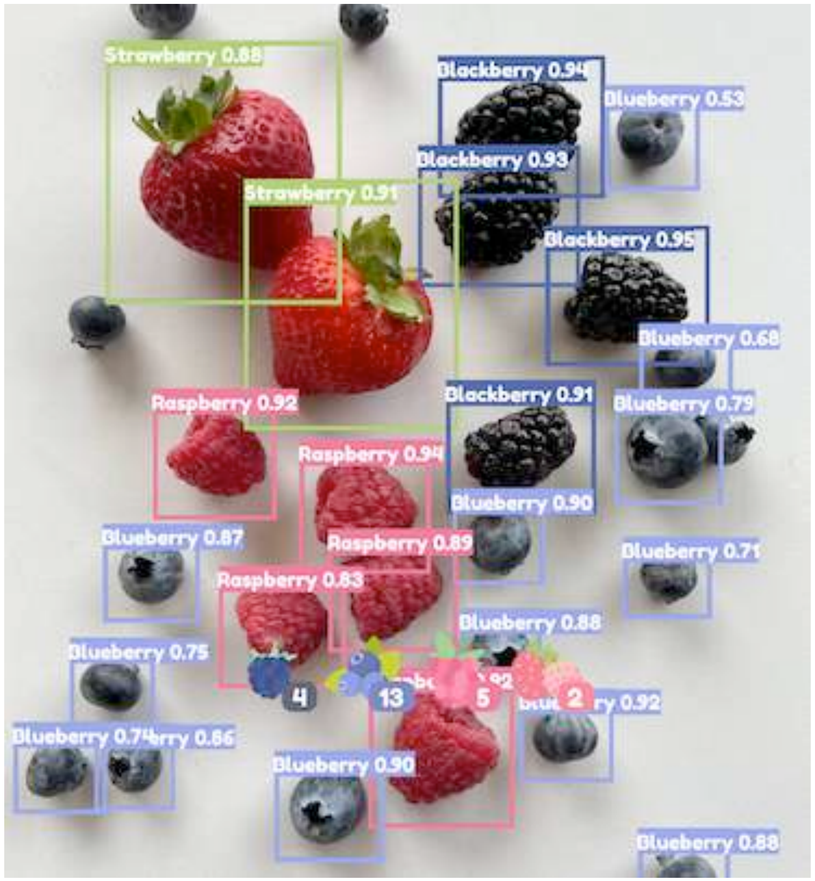


2015 - Fast RCNN

SSD

Single Shot Detector

- SSD está diseñado para la detección de objetos en tiempo real.
- Es un detector de una sola etapa. (Fast R-CNN utiliza un *“region proposal”* y un clasificador CNN).
- SSD acelera el proceso eliminando el *“region proposal”*



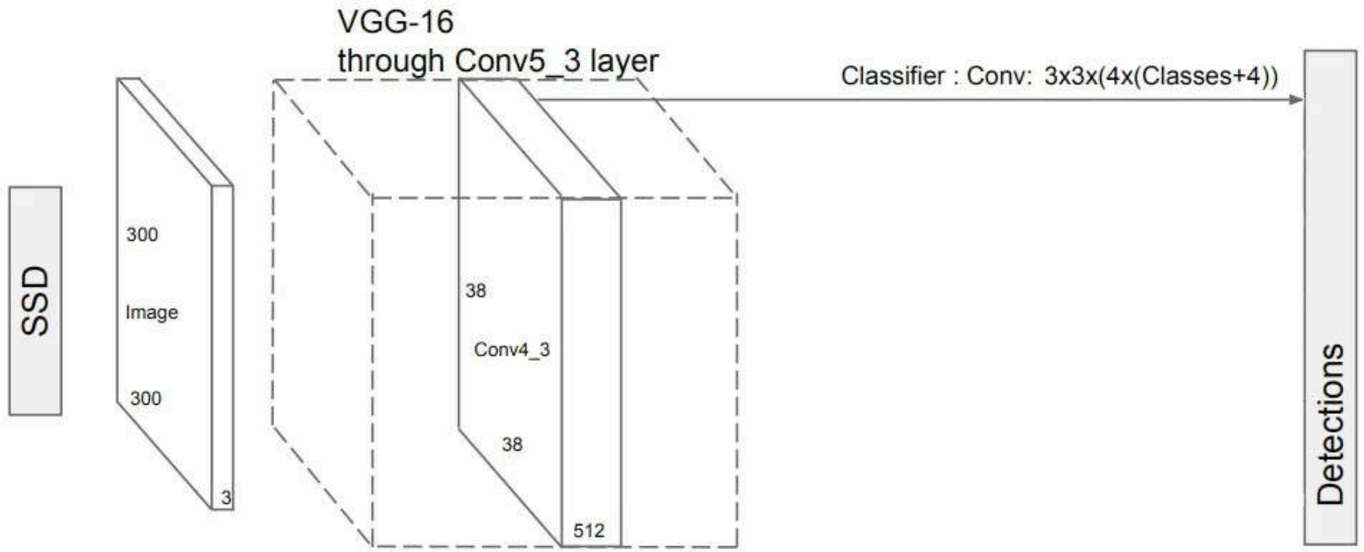
https://docs.snap.com/lens-studio/references/templates/ml/multi-object-detection

SSD

El proceso de detección se compone de dos partes:

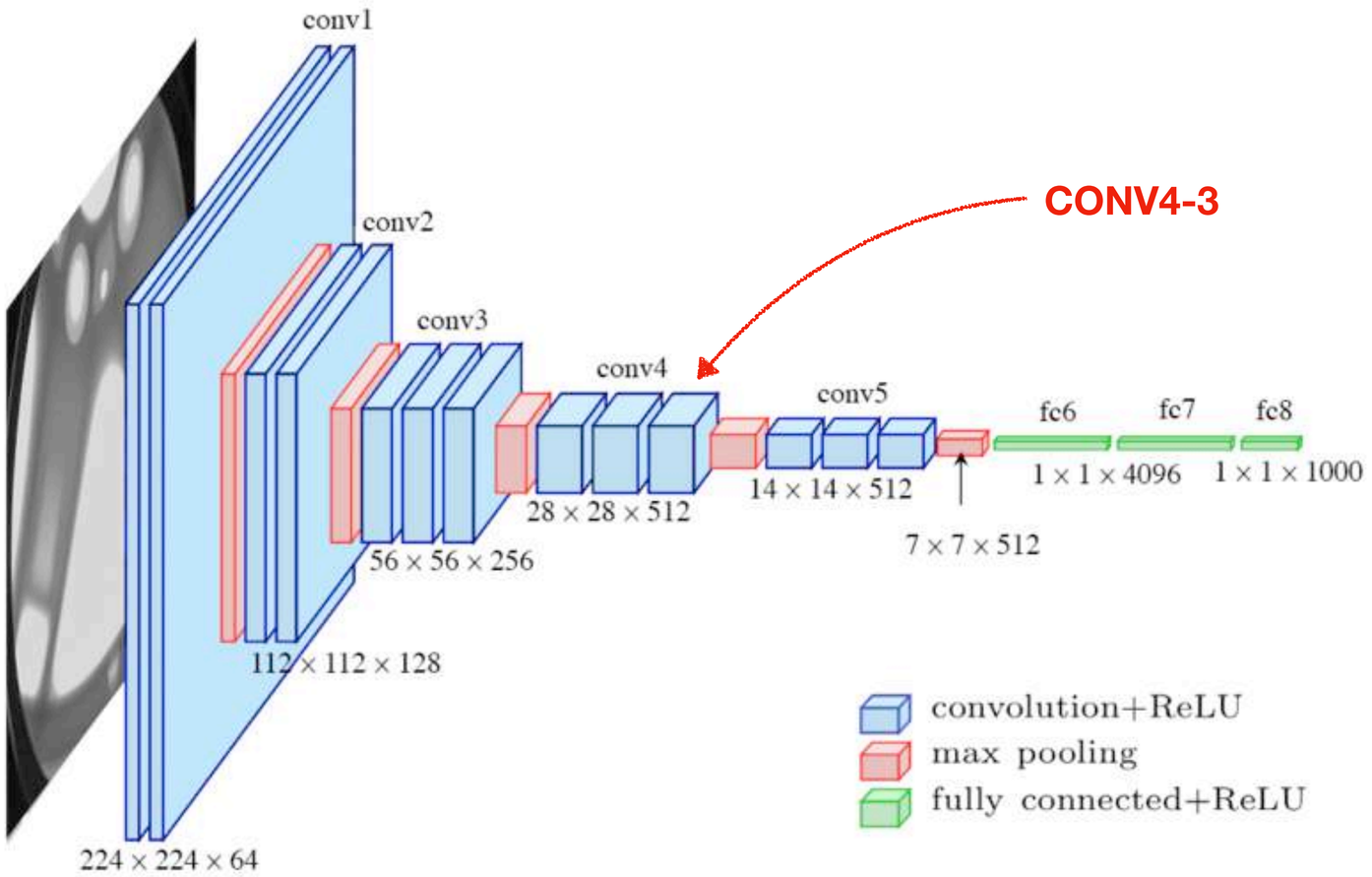
1. Extraer el mapa de características

2. Aplicar CNN para realizar la detección



2016 - SSD: Single Shot MultiBox Detector

Feature extractor

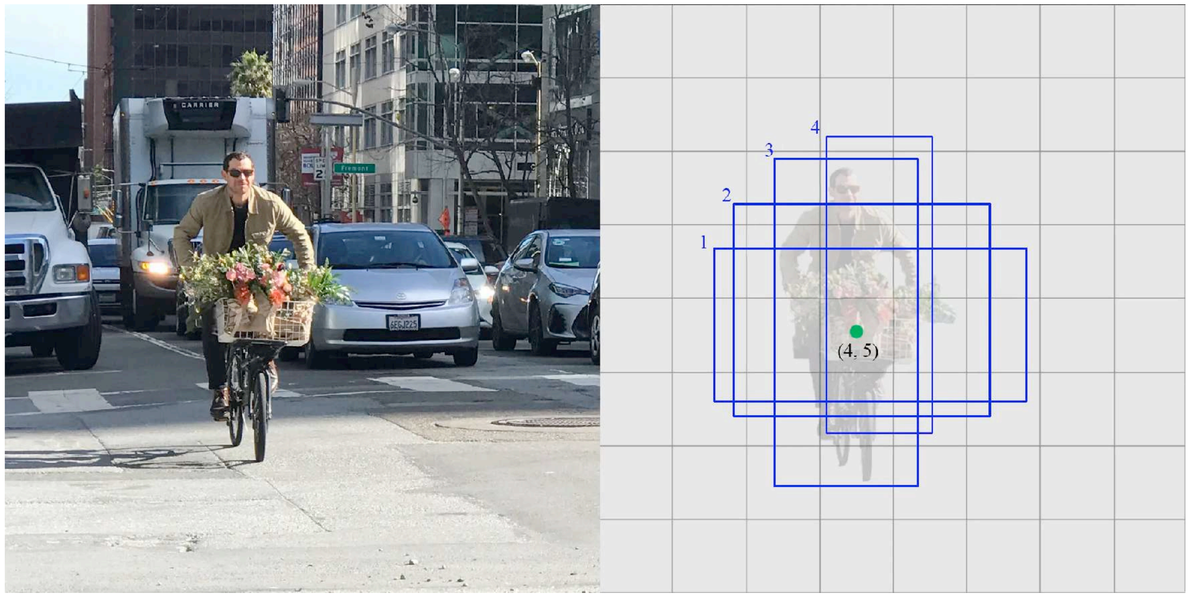


SSD Usa VGG16 para extraer el mapa de características. Luego identifica los objetos usando los mapas desde CONV4-3

2017 - Automatic localization of casting defects with convolutional neural networks

Predicción de detección

- Por cada celda (Ubicación) del mapa de características **predice *k* objetos**
- Cada predicción se compone de:
  A. Rectángulo (Bounding Box)
  B. El vector correspondiente a la clase del BB **+** una clase de no objeto
- Se escoge la clase con mayor puntaje

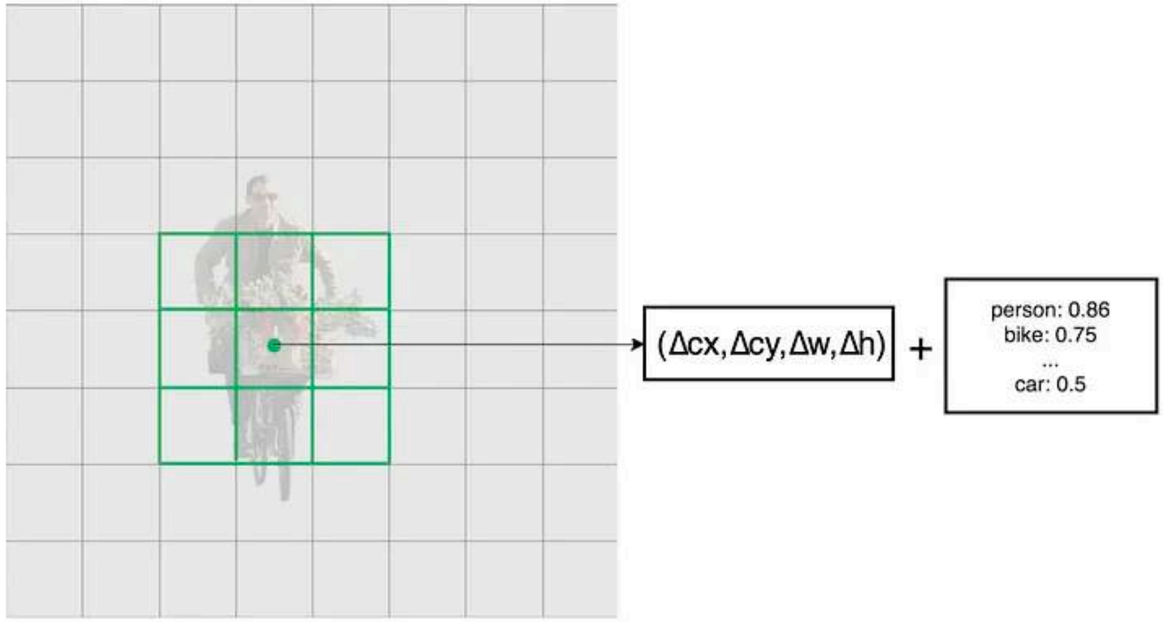


Predicción de detección por cada celda del mapa de características

https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06

Predicción de detección

- Calcula la ubicación y la clase usando pequeños filtros convolucionales
- Aplica una capa de filtros convolucionales de 3x3 con stride de 1 sobre el mapa de características
- **Por cada ubicación del mapa predice *k* rectángulos más la clase**
  - Cada elemento k corresponde a la predicción de rectángulos de distintos ratios y tamaños
  - Un rectángulo vertical es más adecuado para una persona y uno horizontal para un vehículo

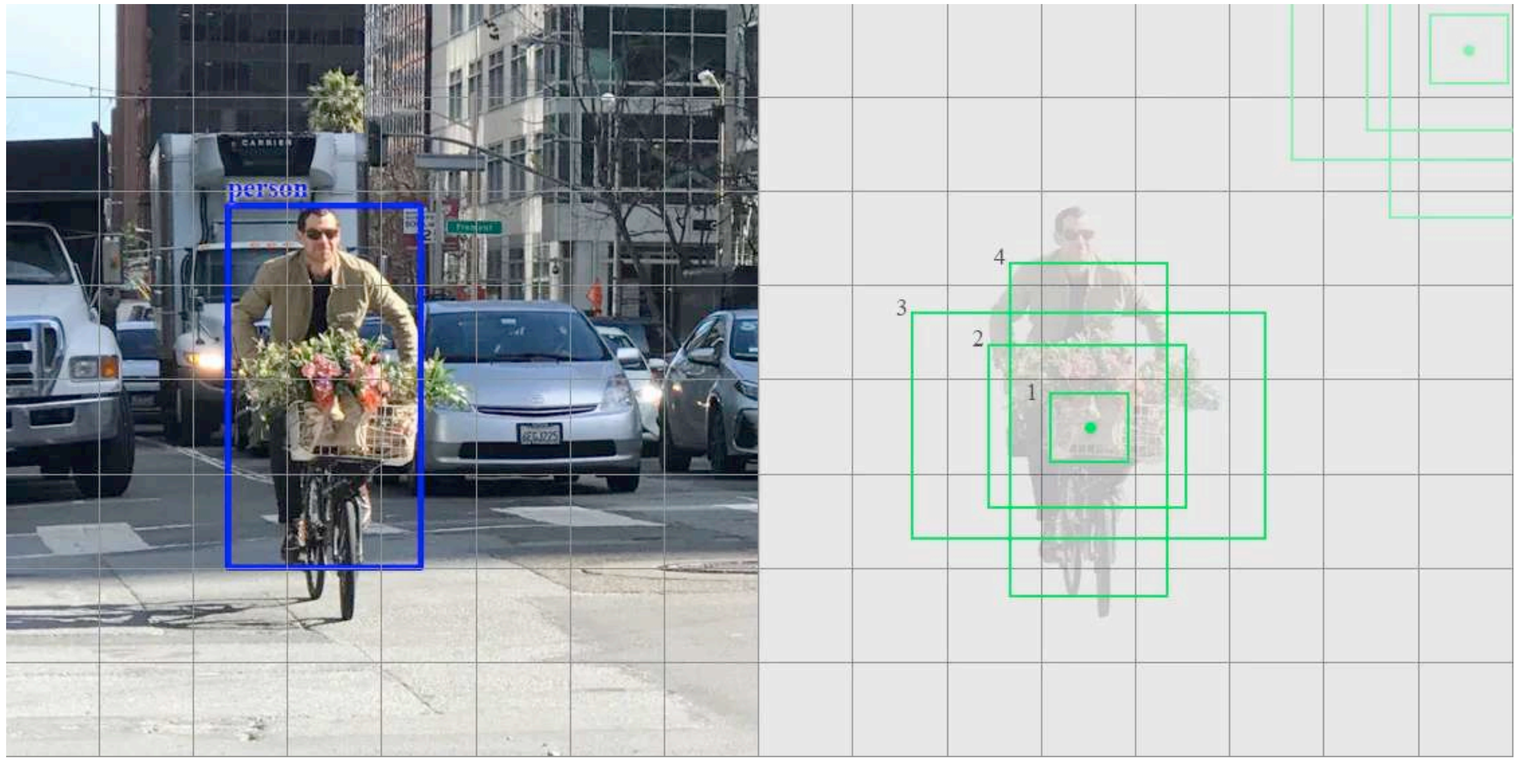


Predicción de detección por cada celda del mapa de características

https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06

Predicción de detección

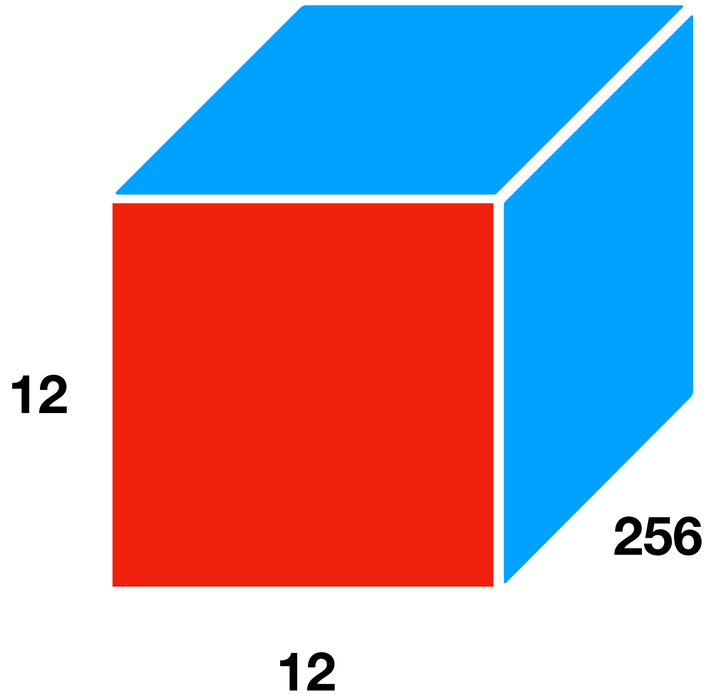
- El mapa de características es de $m \times n$
- Por cada celda predice k número de objetos
- Cada k está compuesto de:
  - c puntajes de la clase
  - 4 offsets relativos al rectángulo base



https://jonathan-hui.medium.com/ssd-object-detection-single-shot-multibox-detector-for-real-time-processing-9bd8deac0e06

Predicción de detección

- Si tenemos un mapa de características de 12x12
- Vamos a predecir 10 clases
- Y tenemos 2 rectángulos base (2 aspect ratio)
- **¿Cuántos filtros debe tener la capa convolucional de detección?**

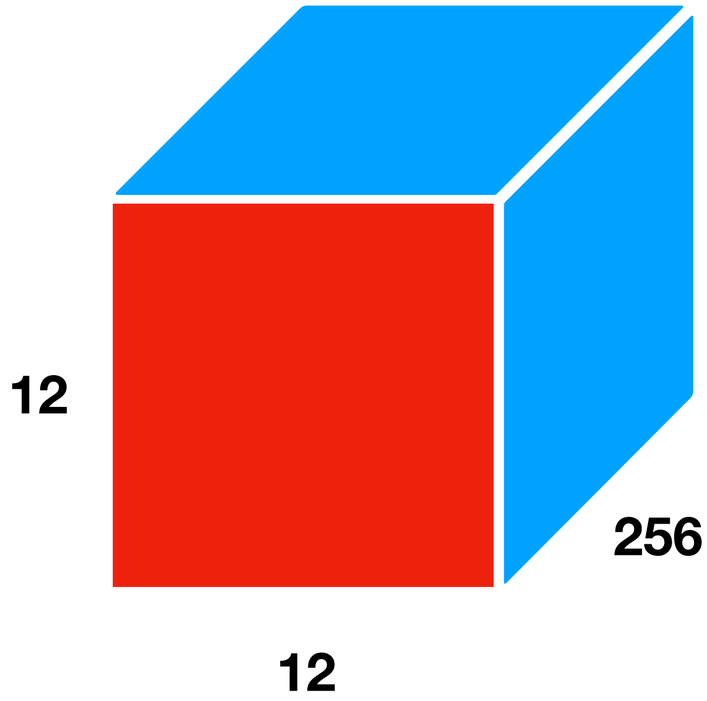


Predicción de detección

Número de filtros de la capa de predicción

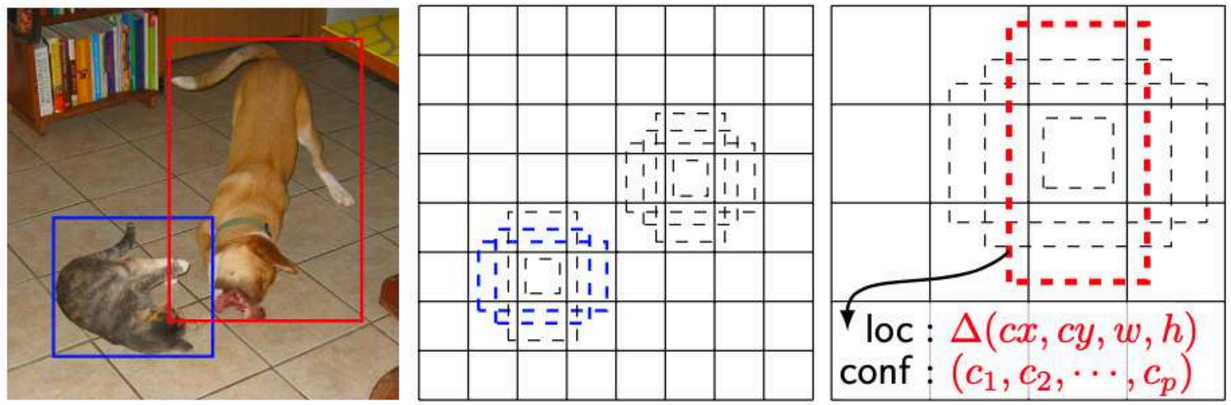
$$(c + 5) \times k$$

$$(10 + 5) \times 2 = 30$$



Múltiples escalas

- Describimos como el SSD detecta objetos a partir de una sola capa.
- En realidad usa varias capas de la red que extrae características para detectar objetos de forma independiente.
- La CNN reduce gradualmente la dimensión espacial, la resolución de los mapas de características también disminuye.
- SSD utiliza capas de menor resolución para detectar objetos a mayor escala.

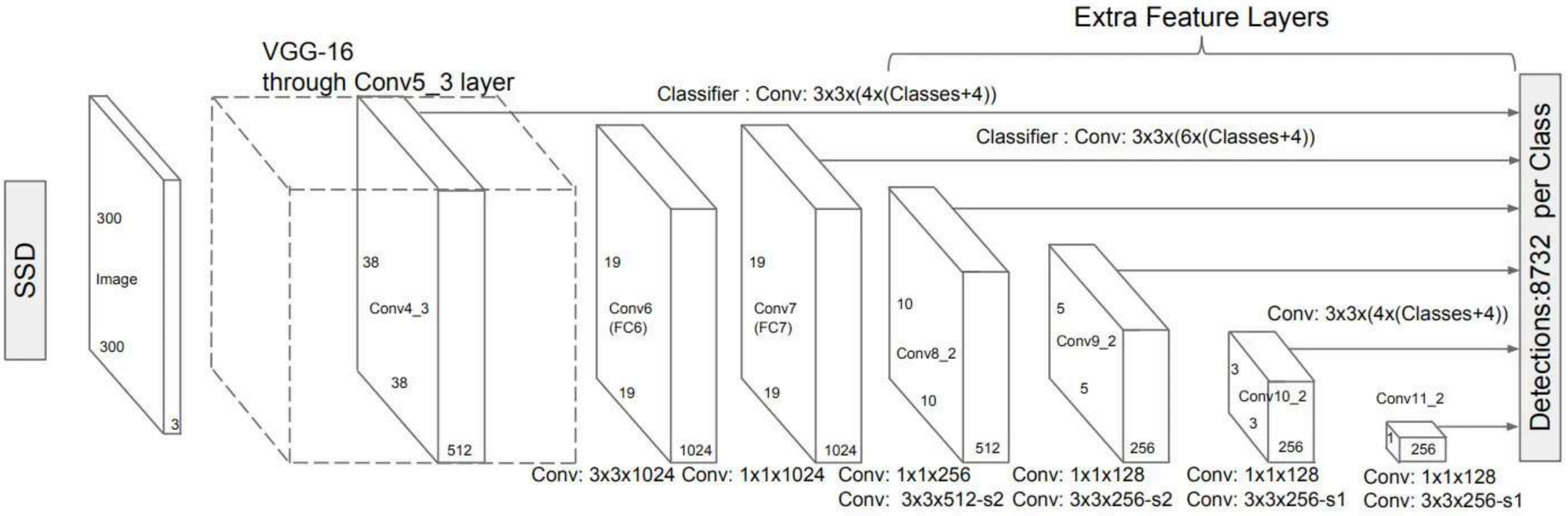


(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Los mapas de características de menor resolución (derecha) detectan objetos de mayor escala.

2016 - SSD- Single Shot MultiBox Detector

SSD



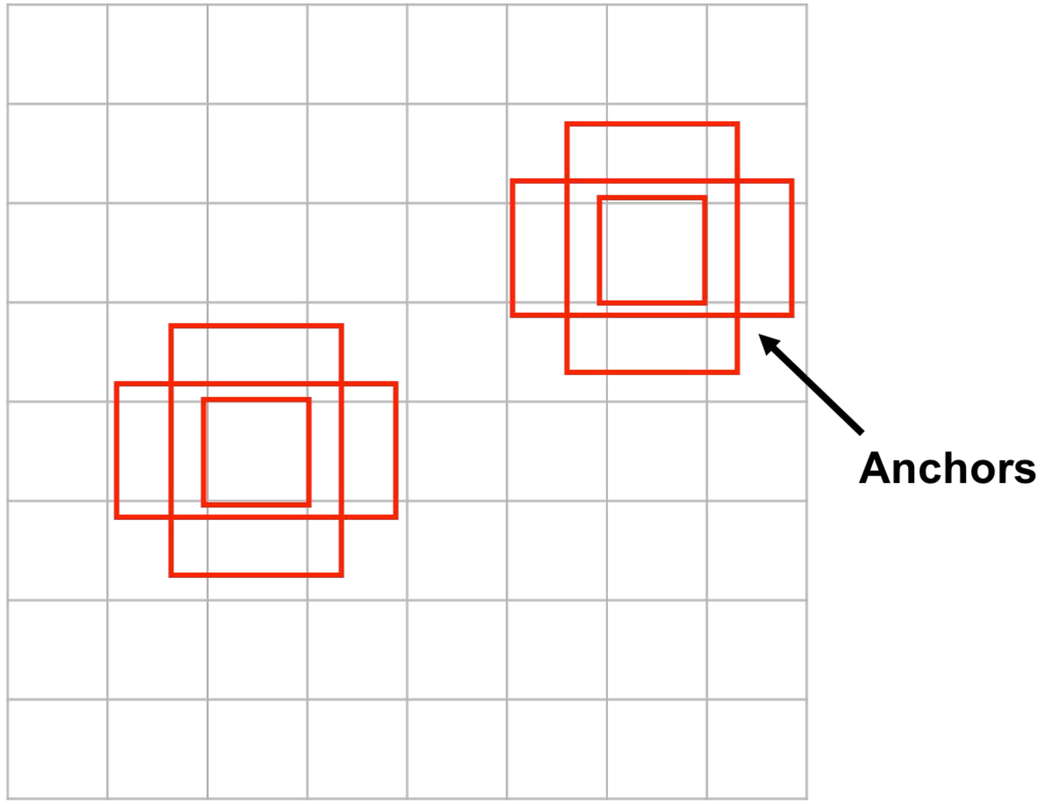
SSD añade **6 capas** más a la VGG16 que produce **k predicciones** de objetos por capa

Default boxes and aspect ratios

- Durante el entrenamiento el modelo puede luchar entre sí para determinar qué formas (personas o vehículos) deben optimizarse

- El entrenamiento inicial puede ser muy inestable. Las predicciones de los rectángulos que funcionan bien para una categoría para otras no

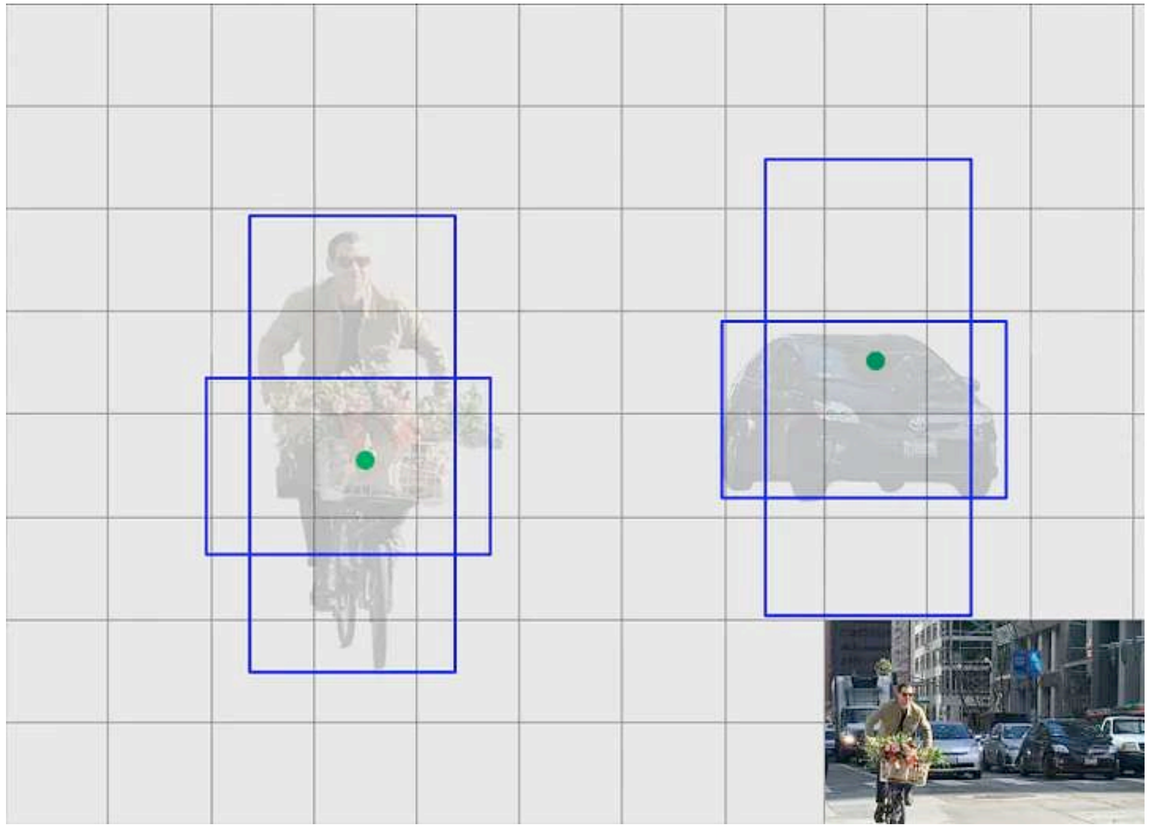
- Se quiere que las predicciones iniciales sean diversas y no se parezcan.



http://kimberlyfessel.com/algorithms/literature%20reviews/object-detection-challenges/

Default boxes and aspect ratios

- Para reducir la complejidad, las rectángulos predeterminados se pre-seleccionan manual para cubrir el espectro de los objetos a predecir
- SSD sugiere 4 o 6 rectángulos por defecto
- Cada capa del mapa de características, comparte el mismo rectángulos por defecto
- Las diferentes capas utilizan diferentes rectángulos por defecto para detecciones de diferentes tamaños



Default boxes and aspect ratios

- SSD define un valor de escala para cada capa del mapa de características
- Las escalas van desde 0.2 a 0.9 (última capa)
- Se calcula el tamaño del rectángulo usando la escala y los *aspect ratios* que se quieren usar
- SSD propone 5 aspect ratios: 1, 2, 3, 1/2, y 1/3.

Cálculo del ancho

$$w = s\sqrt{a}$$

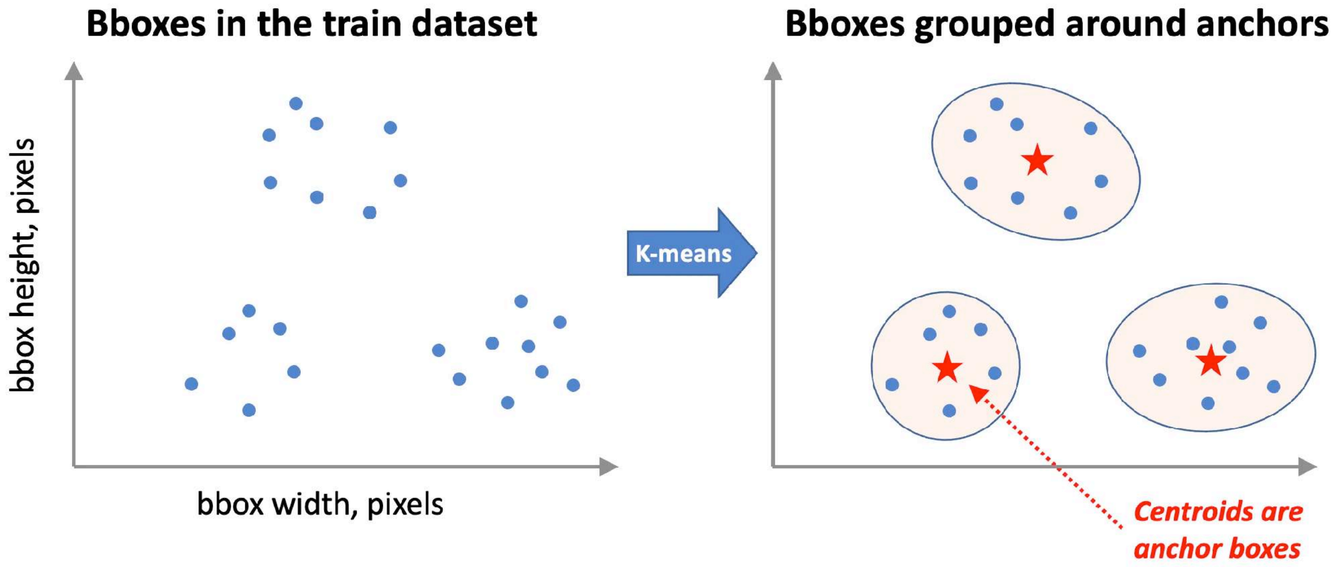
Calculo del alto

$$h = \frac{s}{\sqrt{a}}$$

En donde s es la escala y a el *“aspect ratio”*

Default boxes and aspect ratios

- Los *Bounding Box* también pueden ser definidos por agrupamiento (k-means)
- Se usan las etiquetas de entrenamiento para el calculo de los BB (Yolo)
- Como calcular los BB con K-means

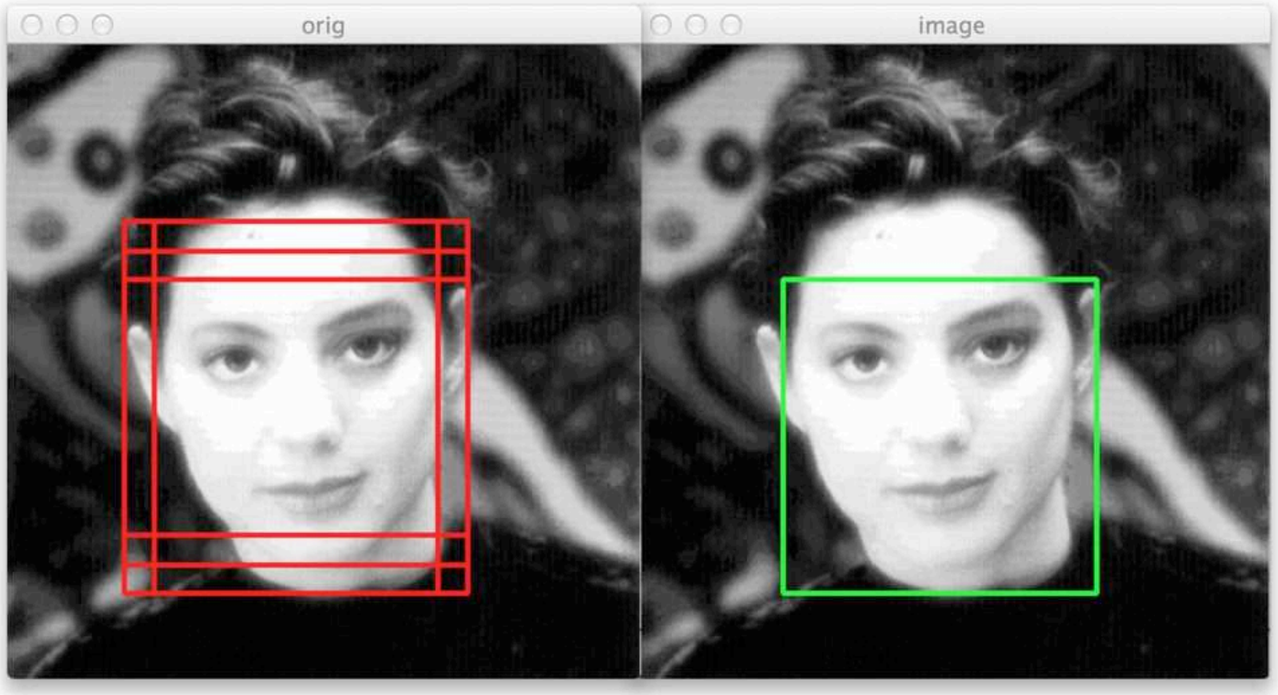


https://towardsdatascience.com/training-yolo-select-anchor-boxes-like-this-3226cb8d7f0b

Predicción final

- Luego de la predicción se tienen múltiples rectángulos que se solapan

- Se realiza un paso de **non-maximum suppression** para producir el resultado final

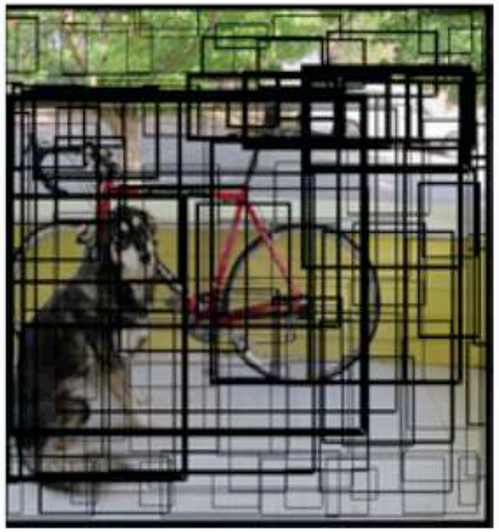


https://pyimagesearch.com/2015/02/16/faster-non-maximum-suppression-python/

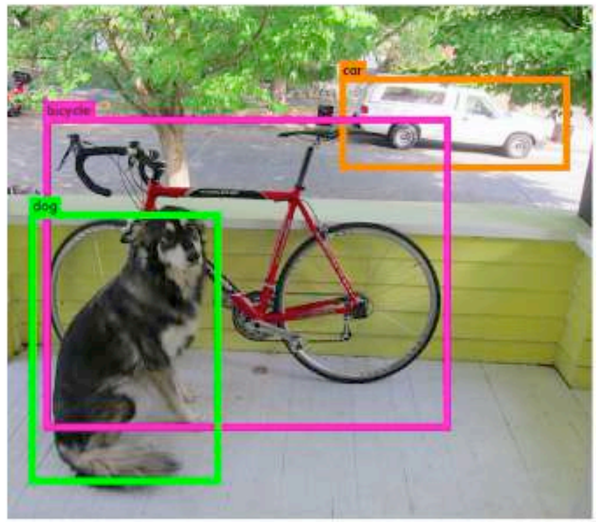
Non-maximum Suppression

- La primera etapa tiene una alta sensibilidad para predecir todas las posibles regiones, esto da paso a tener cientos de regiones similares

- Non-maximum Suppression es una técnica para “fusionar” rectángulos del mismo objeto



Multiple Bounding Boxes



Final Bounding Boxes

https://www.analyticsvidhya.com/blog/2020/08/selecting-the-right-bounding-box-using-non-max-suppression-with-implementation/

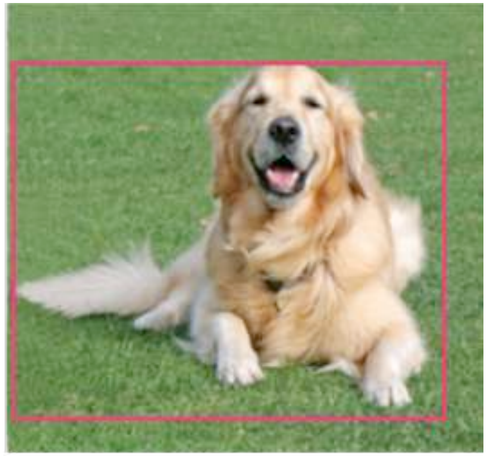
Non-maximum Suppression

Algoritmo

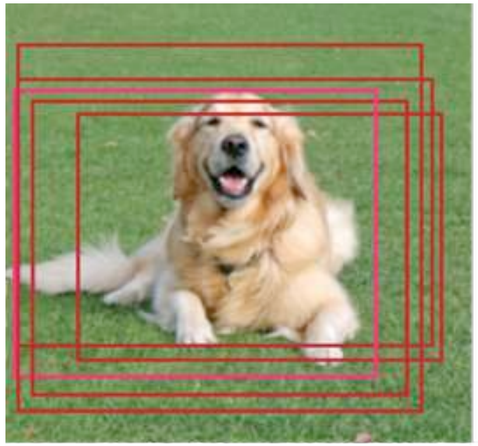
1. Seleccione la propuesta con mayor *score*, elimínela de *B* y añádala a la lista final de propuestas *D*. (Inicialmente *D* está vacía).

2. Calcula el IOU de la propuesta con los otros rectánculos. Si el IOU es mayor que el umbral *N* (*Y es la misma clase) se elimina

3. Este proceso se repite hasta que no queden más propuestas en *B*.



After applying non-maximum suppression



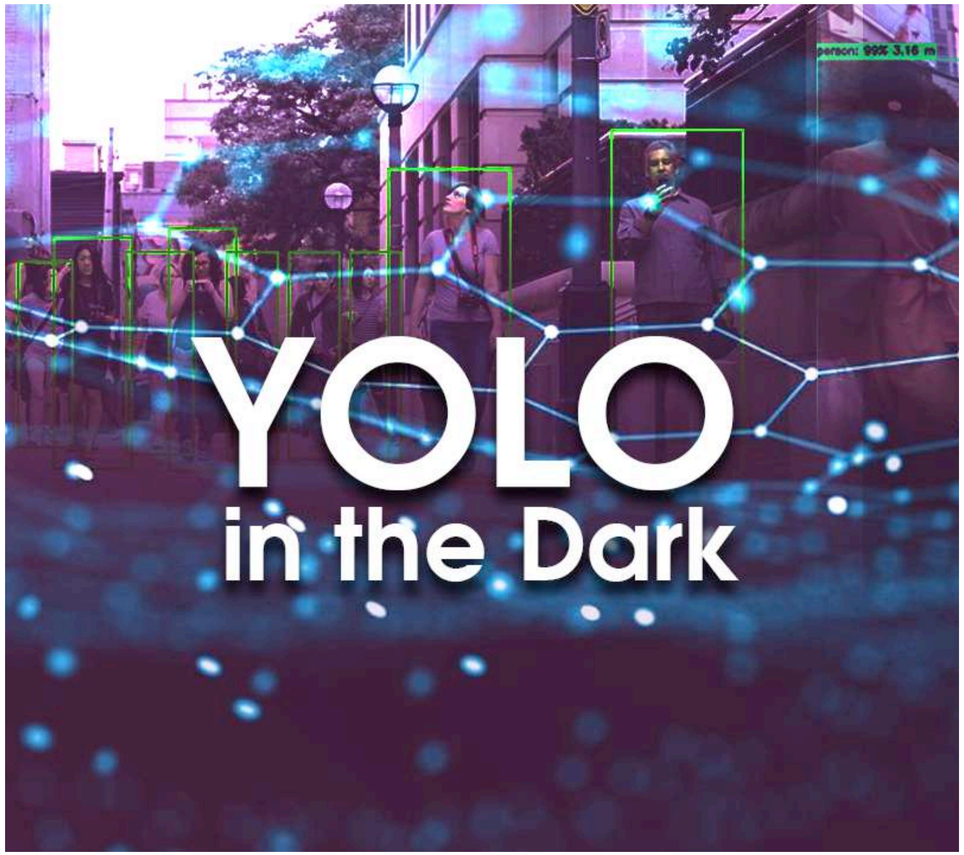
Predictions before NMS

https://livebook.manning.com/concept/deep-learning/object

Yolo

You Only Look Once

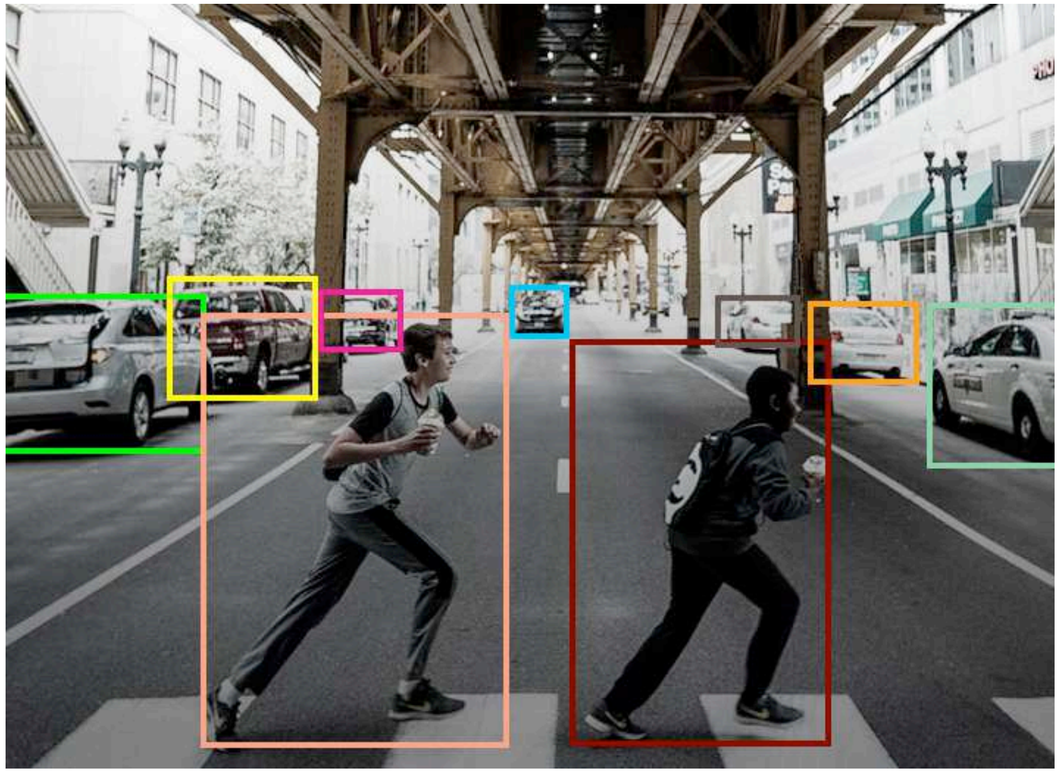
- Es un algoritmo muy conocido de detección
- Es un detector de una sola etapa
- Reutiliza los clasificadores de imágenes tradicionales para la tarea de regresión de identificar los rectángulos



https://socionextus.com/pressreleases/deep-learning-method-for-object-detection-in-low-light-conditions/

Introducción

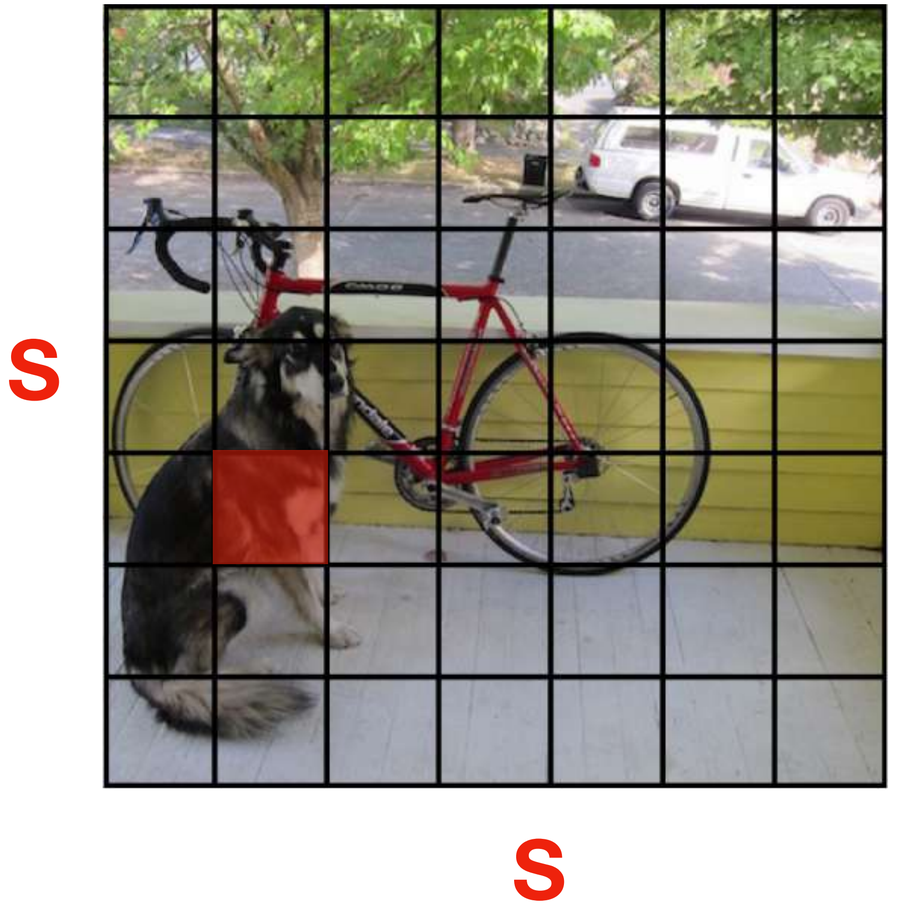
- YOLOv1, la primera de las muchas iteraciones por las que ha pasado esta arquitectura.
- La idea básica de la arquitectura sigue siendo la misma. YOLOv1 se denomina simplemente YOLO (Actual v8)
- **La clave es velocidad, predice a 45 frames por segundo**
- **Es una buena opción para aplicaciones que requieren detección en tiempo real**



https://www.analyticsvidhya.com/blog/2019/08/introduction-slimyolov3-real-time-object-detection/

Introducción

- YOLO se basa en la idea de segmentar una imagen en imágenes más pequeñas.
- La imagen se divide en una cuadrícula de dimensiones **S×S**
- La celda en la que se encuentra el centro de un objeto, por ejemplo, es la responsable de detectar ese objeto (centro del perro)



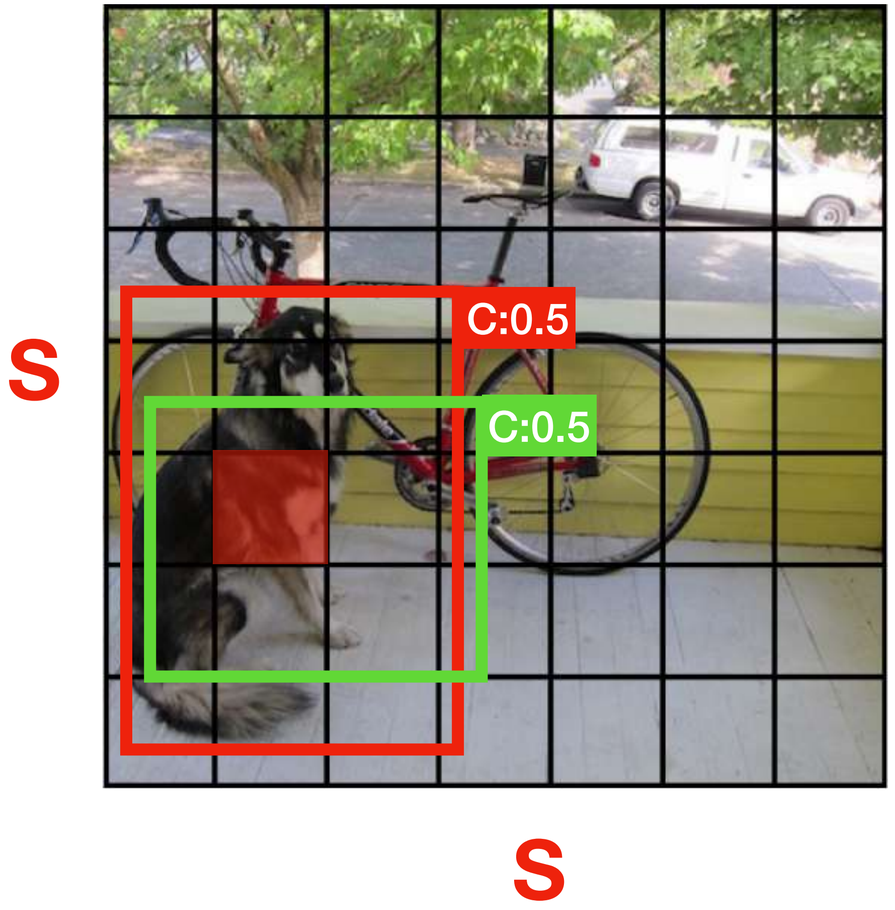
https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31

Predicciones

- Cada celda predecirá ***K*** rectángulos y una puntuación de confianza para cada uno

- Estos niveles de confianza reflejan la certeza del modelo de que existe un objeto en esa celda

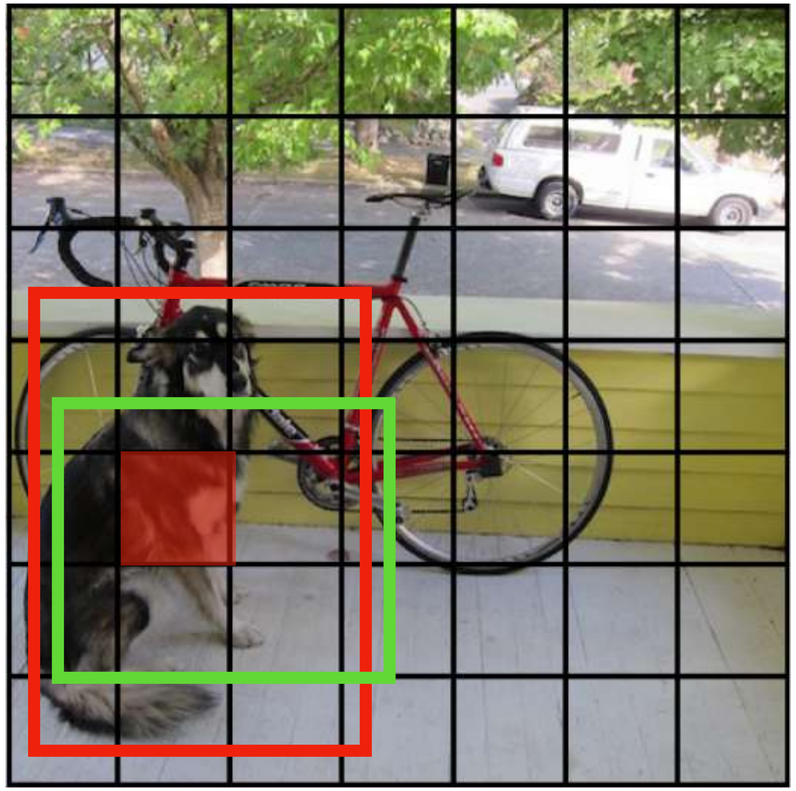
- La confianza representa la diferencia entre el IOU de la predicción y el rectángulo verdadero



Predicciones

Además de rectángulos y la puntuación de confianza, cada celda predice la clase del objeto (on-hot)

Cuántas predicciones hace un modelo si tiene 5 clases y una cuadrícula de 7x7 y predice 2 rectángulos por cuadro?



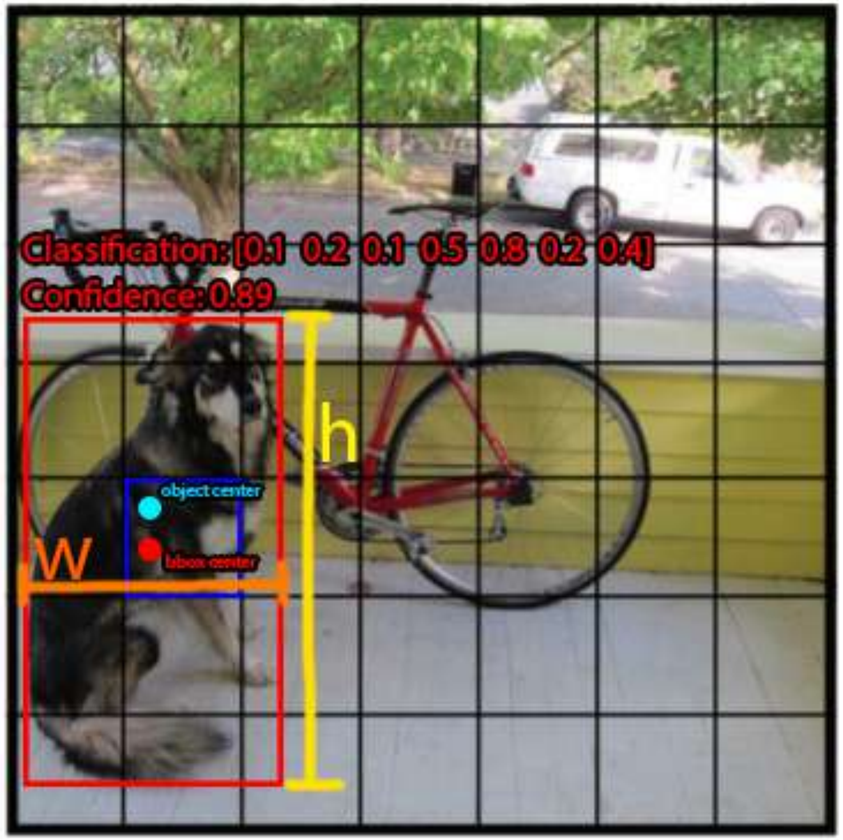
https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31

Predicciones

7x7 x (1+4+5) x 2

Cuadrícula | Bounding box + confiabilidad + clases | Predicciones

= 980



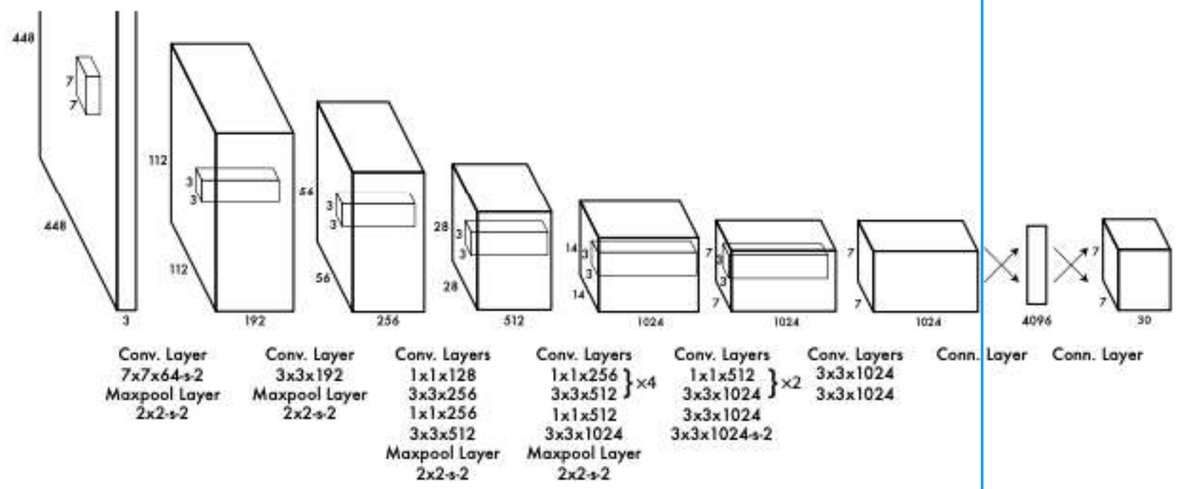
S x S grid on input

**** El modelo predice el centro del recuadro con el ancho y alto en lugar de las posiciones de las esquinas superior izquierda e inferior derecha.**

Arquitectura

- YOLO está formado por tres componentes: *head*, *neck*, y *backbone*
- El **backbone** encarga de la extracción de características
- The backbone se entrena primero en un conjunto de datos de clasificación (ImageNet)
- Se pre-entrena con una resolución inferior a la del modelo de detección, ya que la detección requiere más detalles

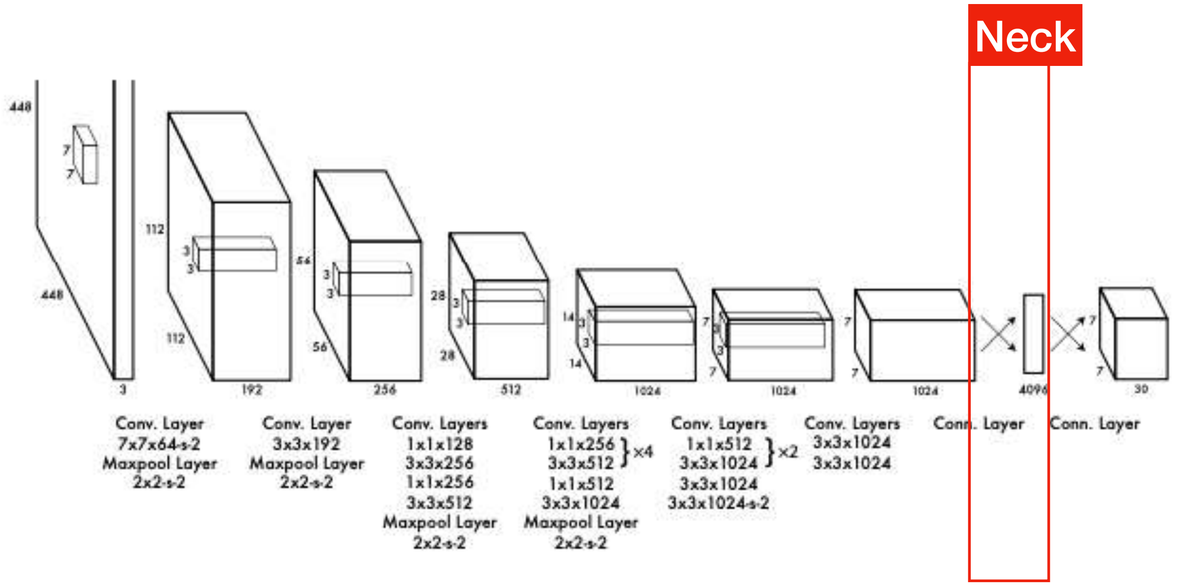
Backbone



2016 - You Only Look Once Unified Real-Time Object Detection

Arquitectura

- **El cuello** utiliza las características de las capas de convolución del *backbone* con capas FC
- Produce un vector que posteriormente es utilizado por la cabeza
- Esto tiene el objetivo que la cabeza del modelo sea intercambiable

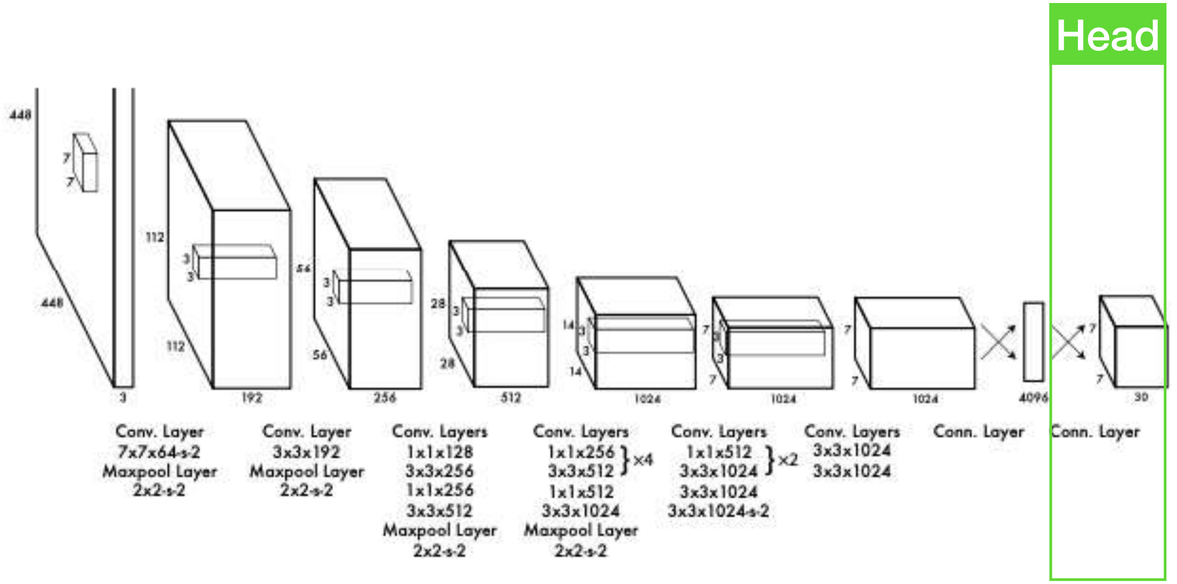


2016 - You Only Look Once Unified Real-Time Object Detection

Arquitectura

- **La cabeza** es la capa de salida final de la red que puede intercambiarse con otras capas con la misma forma de entrada para la transferencia de aprendizaje

- Realizar las predicciones finales de los rectángulos y sus clases



2016 - You Only Look Once Unified Real-Time Object Detection

Función de costo

- La función de pérdida sólo penaliza el error de clasificación si un objeto está presente en esa celda

- Sólo penaliza el error de coordenadas de la caja delimitadora si esa celda es la “responsable"

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

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

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

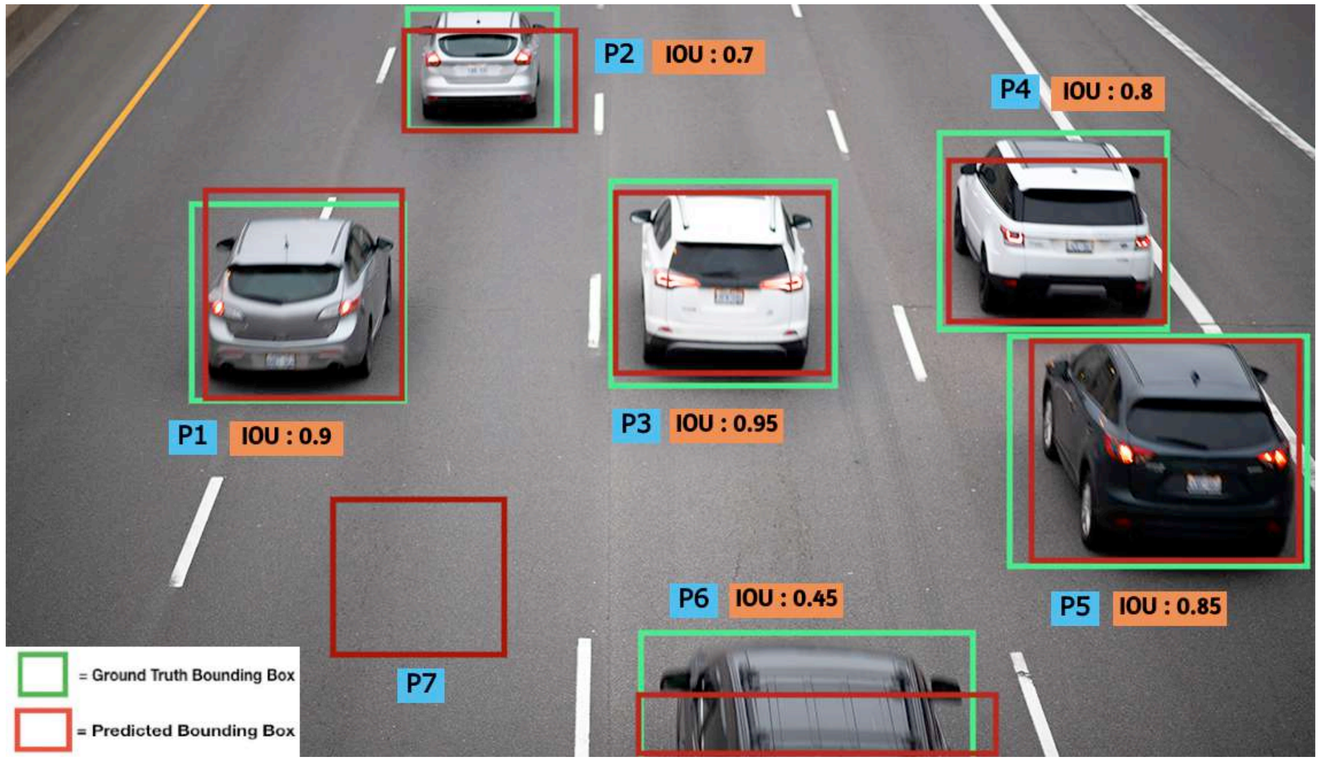
where $\mathbb{1}_i^{\text{obj}}$ denotes if object appears in cell i and $\mathbb{1}_{ij}^{\text{obj}}$ denotes that the jth bounding box predictor in cell i is “responsible” for that prediction.

Mean Average Precision (mAP)

- **AP** (Average precision) es una métrica popular para medir la precisión de detectores de objetos como Faster R-CNN, SSD, etc.

- **AP** es el AUC de la curva precisión-sensibilidad.

- El **mAP** es el primed de **AP** de cada clase **.



https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

Average Precision (AP)

- **AP** No es el promedio de la precisión de las distintas clases.
- **AP** se calcula con la ayuda de otras métricas como IoU, matriz de confusión (TP, FP, FN), precisión y la sensibilidad.

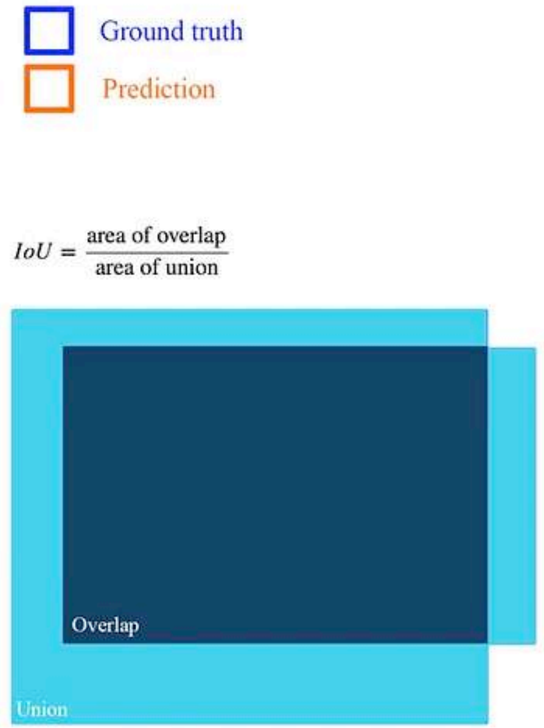
$$\textbf{mAP} = \frac{1}{n} \sum_{h=1}^{n} \textbf{AP}_k$$

AP$_k$: Average Precision de la clase k

n**:** EL número de clases

Interception Over Union (IOU)

- IoU mide el solapamiento entre dos rectángulos.
- Lo utilizamos para medir cuánto se solapa el rectángulo predicho con el real
- **Predefinimos un umbral de IoU** (por ejemplo, 0,5) para clasificar si la predicción es un verdadero positivo o un falso positivo.



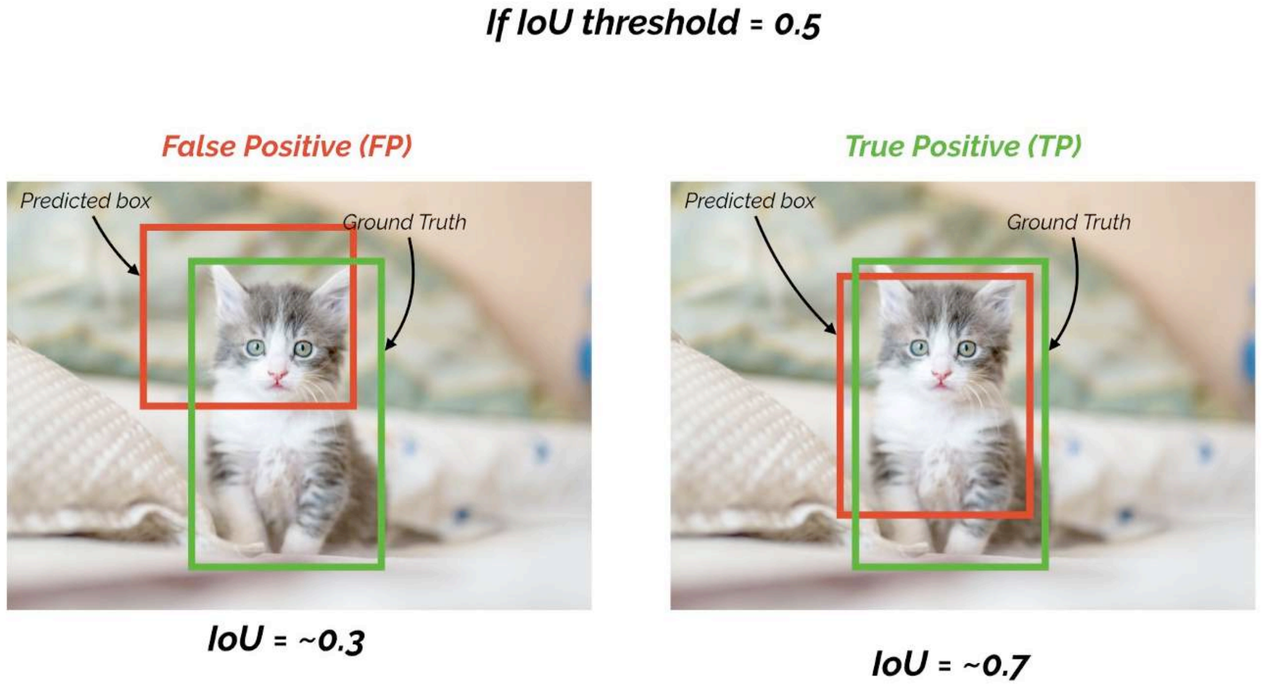
https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173

Average Precision (AP)

A partir del IoU calculamos:

Verdadero positivo: El modelo predijo que existía un rectángulo en una posición y acertó.

Falso positivo: El modelo predijo que existía un rectángulo en una posición determinada pero se equivocó

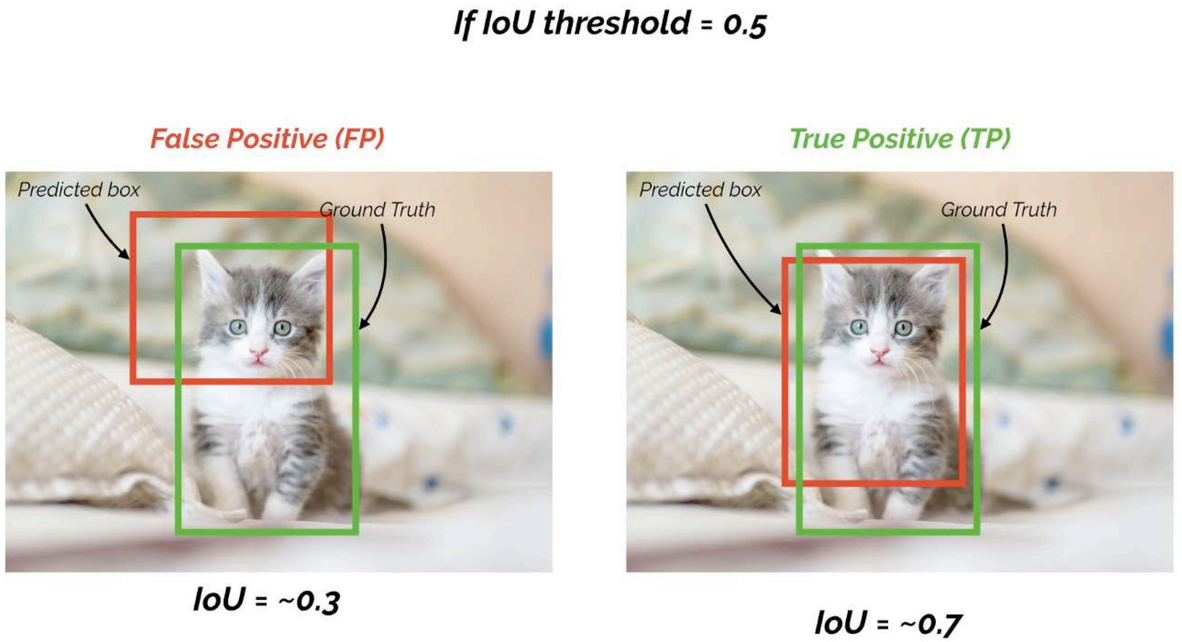


https://www.v7labs.com/blog/mean-average-precision

Average Precision (AP)

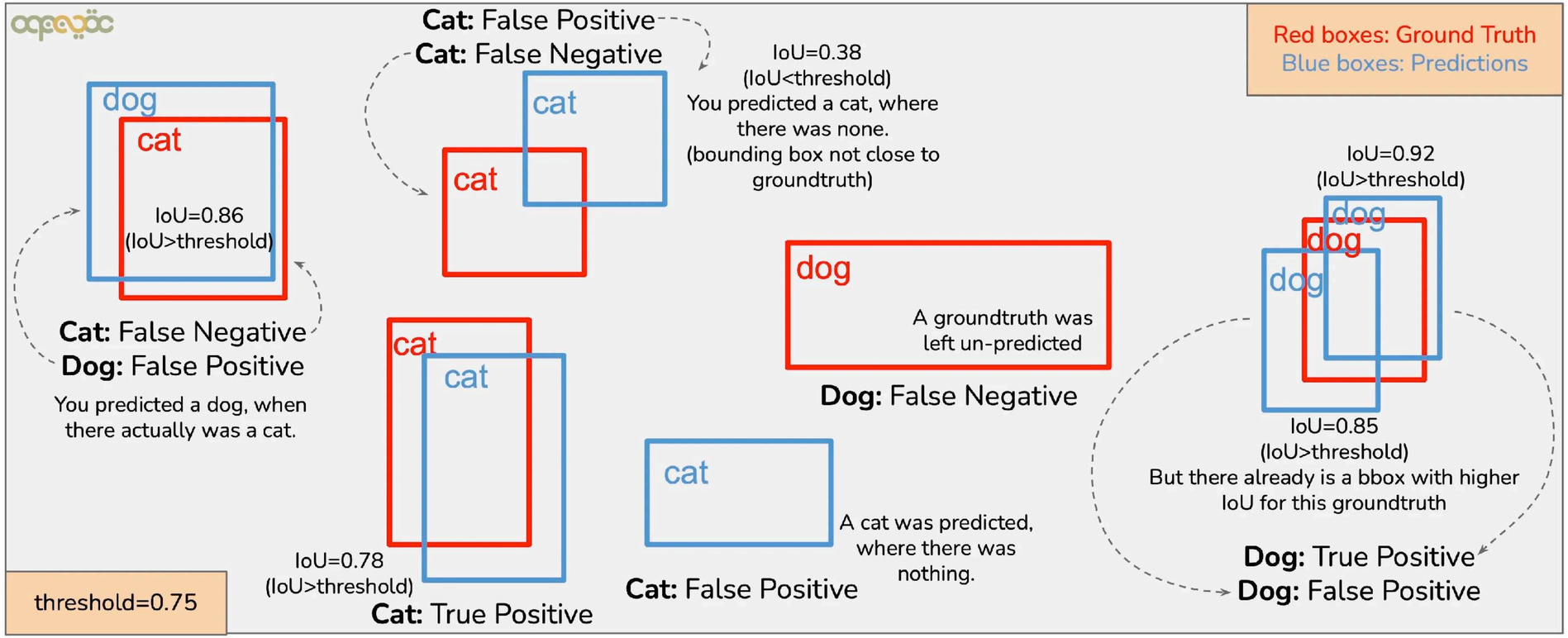
Falso negativo: El modelo no predijo un rectángulo en una posición determinada y se equivocó

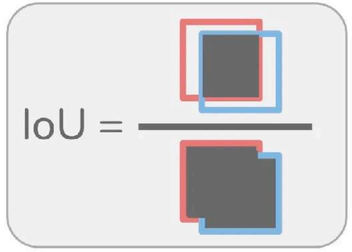
Verdadero negativo: Corresponde al fondo, la zona sin rectángulos, y no se utiliza para calcular las métricas finales



https://www.v7labs.com/blog/mean-average-precision

Object Detection and Localization - IoU, True Positive, False Positive, False Negative





@_aqeelanwar

aqeelanwarmalik

Threshold	Class	# GroundTruth	# predictions	TP	FP	FN	Precision	Recall
0.75	Cat	3	3	1	2	2	1/3	1/3
	Dog	2	3	1	2	1	1/3	1/2
0.35	Cat	3	3	2	1	1	2/3	2/3
	Dog	2	3	1	2	1	1/3	1/2

https://towardsdatascience.com/what-is-average-precision-in-object-detection-localization-algorithms-and-how-to-calculate-it-3f330efe697b

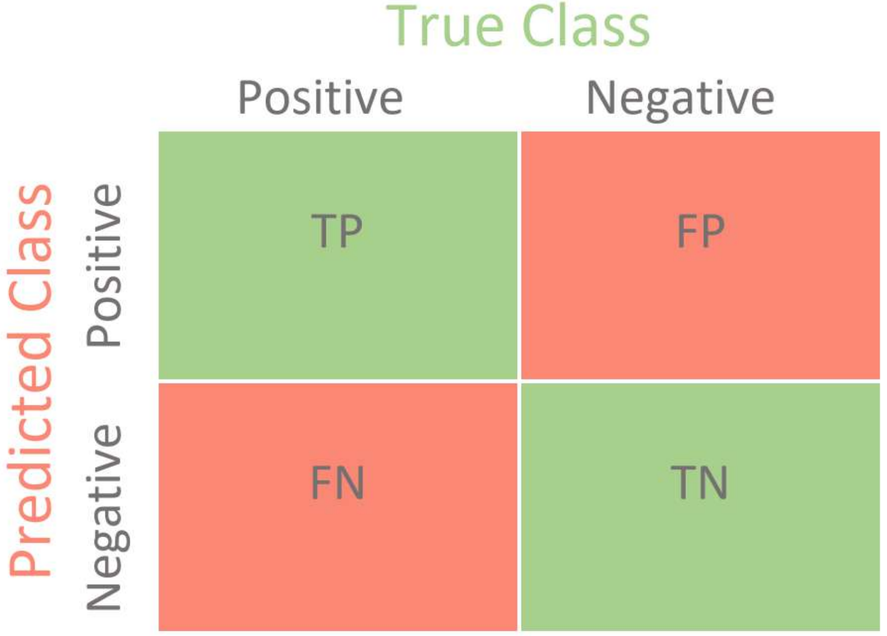
Sensibilidad y precisión

Sensibilidad: ¿Qué proporción de positivos reales se identificó correctamente?

$$\frac{TP}{TP + FN}$$

Precisión: ¿Qué proporción de predicciones positivas fue realmente correcta?

$$\frac{TP}{TP + FP}$$

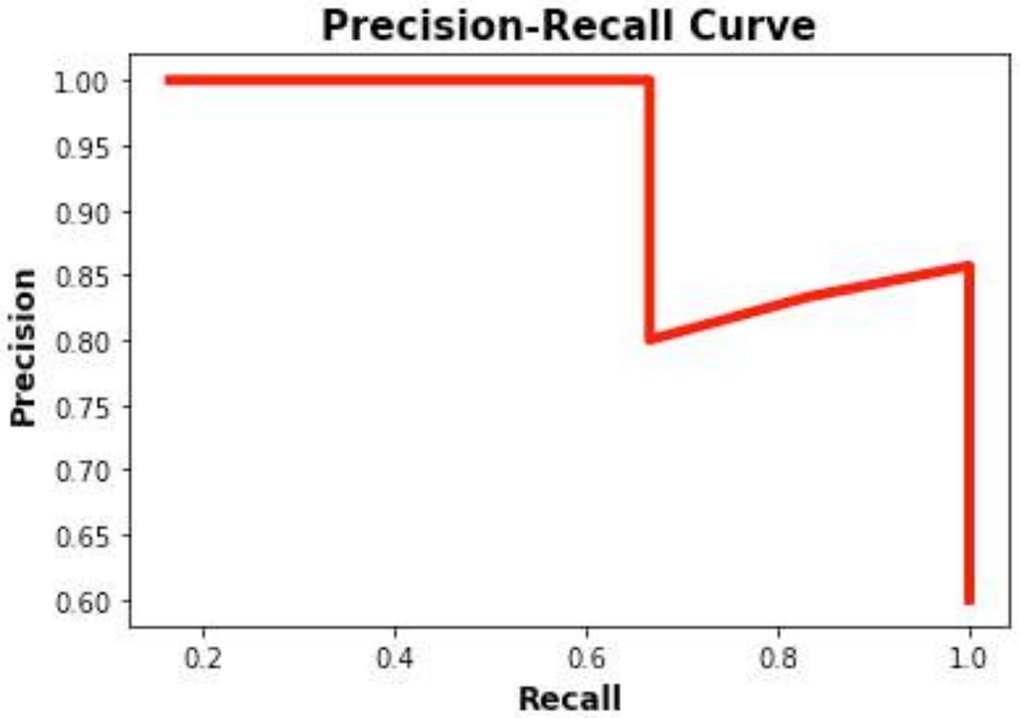


Matriz de confusión para la clasificación binaria.

https://www.sciencedirect.com/topics/engineering/confusion-matrix

Precision-Recall Curve

- Para evaluar un modelo, hay que examinar tanto la precisión como la sensibilidad
- Desafortunadamente, la precisión y la recuperación suelen estar en “tensión”
- **Una curva de precisión-sensibilidad representa el valor de la precisión frente a la sensibilidad para distintos umbrales de confianza.**



https://blog.paperspace.com/mean-average-precision/

Average Precision

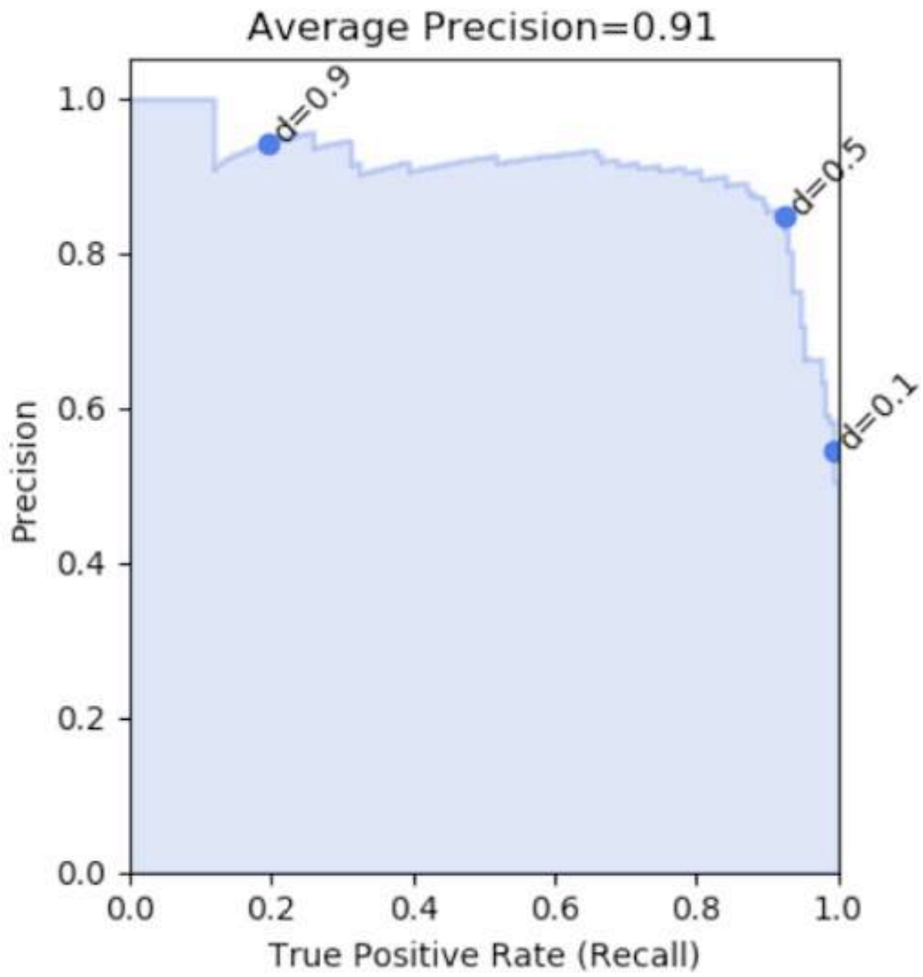
AP

Average precision es el área bajo la curva PR de una clase

$$\textbf{AP} = \int_{r=0}^{1} p(r)$$

mAP

$$\textbf{mAP} = \frac{1}{n} \sum_{h=1}^{n} \textbf{AP}_k$$



Ejercicio

- Realizar un modelo para la localización de objetos (Regresión de las coordenadas de los Bounding Box)

- **Pasos:**

    1. Descargar dataset y etiquetas en imágenes

    2. Crear matriz de etiquetas (y)

    3. Modelo: Extractor de características (Convoluciones) + Cabeza (regresión de 4 variables)

    4. Entrenar con función de pérdida MSE (Mean Squared Error)

Ejercicio

1. Descargar dataset de Teams

2. Graficar etiquetas en imágenes: Las etiquetas están dadas por dos puntos

    - Inicial (x_top, y_top)

    - Final (x_bottom, y_bottom)

Li, F.-F., Andreeto, M., Ranzato, M., & Perona, P. (2022). Caltech 101 (1.0) [Data set]. CaltechDATA. https://doi.org/10.22002/D1.20086

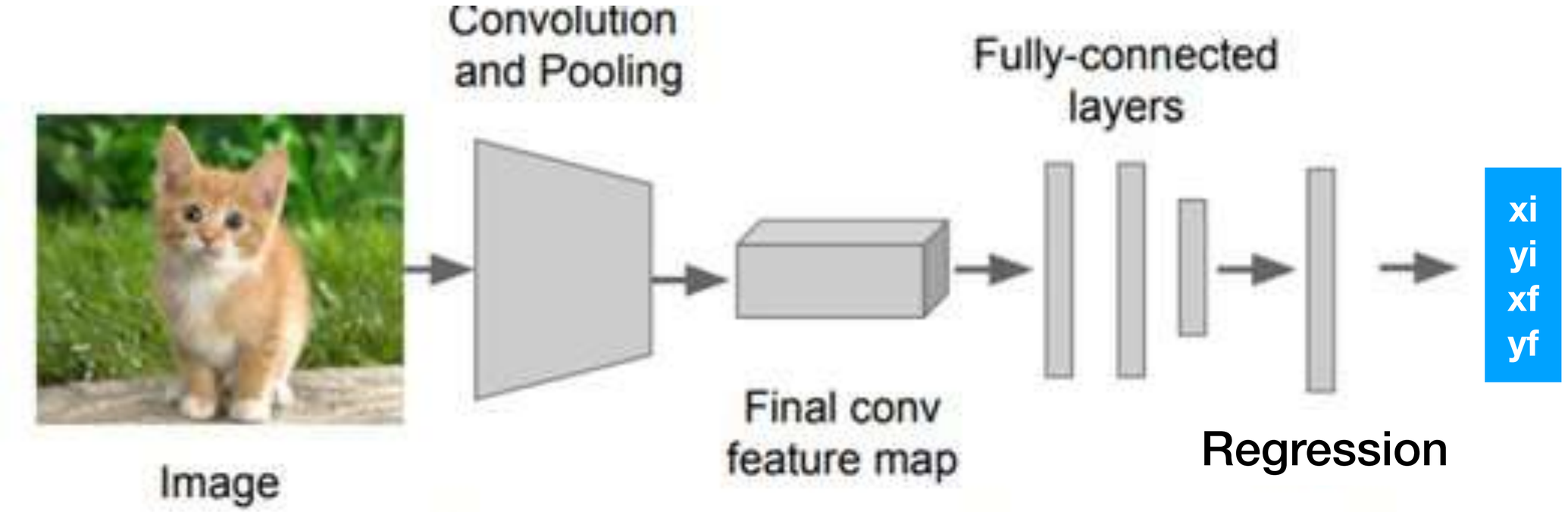
Ejercicio

Generar Etiquetas:

Normalizar por el tamaño de la imagen, de esta forma el modelo predice valores entre 0-1 por cada coordenada

$$\left[\frac{x_{...}}{w}, \frac{y_{...}}{h}\right]$$

Ejercicio


Convolution
and Pooling
Fully-connected
layers
xi
yi
xf
yf
Image
Final conv
feature map
Regression