

# Object-Detection Con Machine Learning

PhD(e). Jonnatan Arias Garcia – [jonnatan.arias@utp.edu.co](mailto:jonnatan.arias@utp.edu.co) –  
[jariasg@uniquindio.edu.co](mailto:jariasg@uniquindio.edu.co)

PhD. David Cardenas peña - [dcardenas@utp.edu.co](mailto:dcardenas@utp.edu.co)

PhD. Hernán Felipe Garcia - [hernanf.garcia@udea.edu.co](mailto:hernanf.garcia@udea.edu.co)

# Contenido

- **Introducción**

- Object-Detection
- Como funciona?
- Detección a partir de la clasificación
- Metrica: mAP

- **Modelos base**

- IoU
- NMS
- R-CNN
  - Fast
  - Faster
  - Mask

- **Modelos famoso**

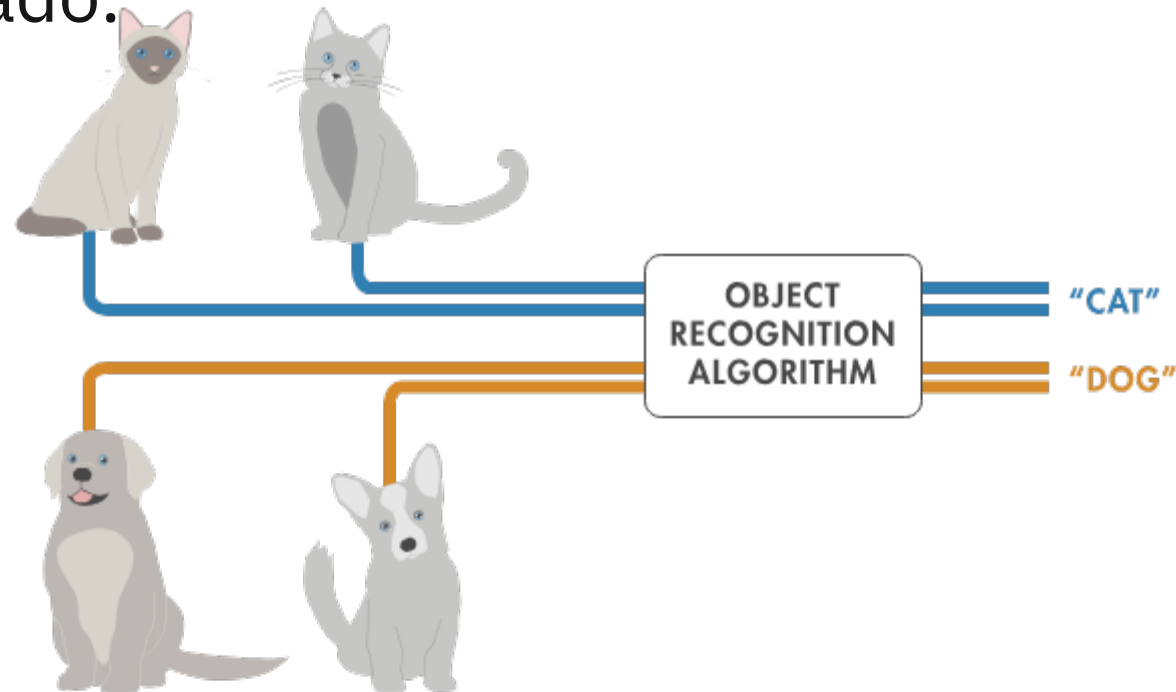
- Yolo
- SSD
- Retina
- Google
- Facebook

# Object-Detection

Mecanismo de Visión por computador capaz de reconocer objetos en imágenes o videos.

## Reconocimiento de objetos con Machine Learning

Reconocer patrones en una imagen o video y ofrece la posición del objeto identificado.

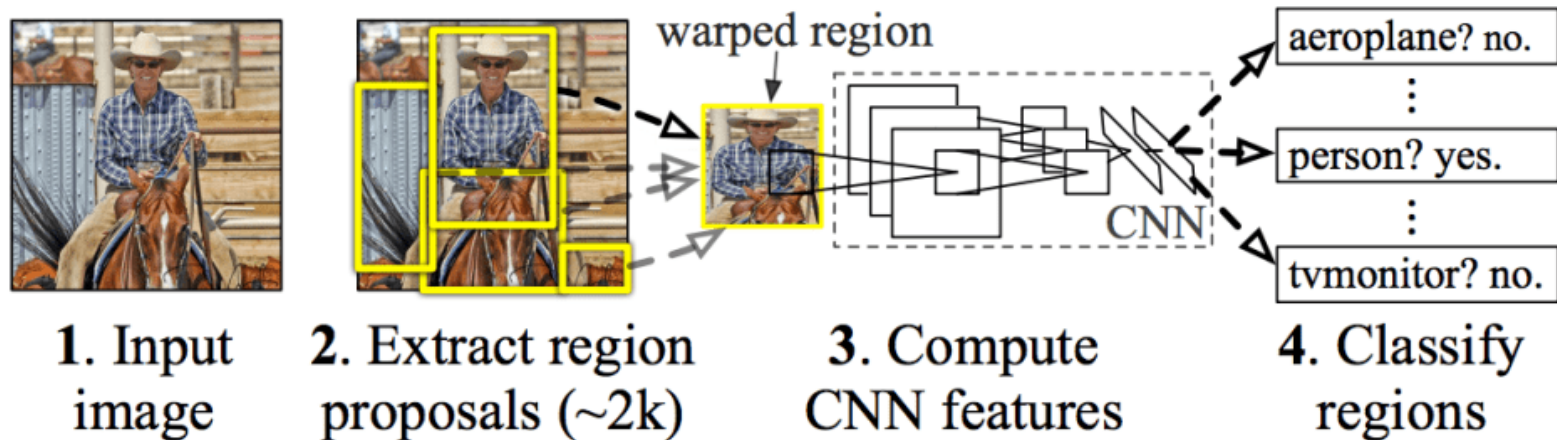


# ¿Cómo funciona el reconocimiento de objetos con ML?

Se basa en recopilación de imágenes donde se seleccionan las propiedades que destacan ca objeto.

Las características identificadas se añaden a un modelo de ML que las categoriza para, posteriormente, realizar análisis y clasificación.

## **R-CNN: *Regions with CNN features***



# ML para detección de objetos

Un algoritmo de Machine Learning de detección, para considerarse como tal deberá:

- Detectar múltiples objetos.
- Dar la posición X e Y del objeto en la imagen (o su centro) y dibujar un rectángulo a su alrededor.

Entonces para entrenar nuestra máquina de manera supervisada deberemos indicar la clase del objeto (por ejemplo perro ó gato) y además la posición dentro de la imagen, X, Y el ancho y alto del objeto.

Y por si esto fuera poco, podrían ser múltiples objetos en la misma imagen

# Detección a partir de la clasificación

Tenemos una red CNN entrenada para detectar perros y gatos y supongamos que tiene una muy buena tasa de aciertos

Entre las redes CNN pre-entregadas más conocidas están Alexnet, Resnet, y VGG

Si a nuestra red pre-entrenada, le pasamos una imagen con 2 perros será incapaz de detectarlos

Si le pasamos una imagen con perros y gatos, tampoco los podrá identificar y mucho menos localizar.

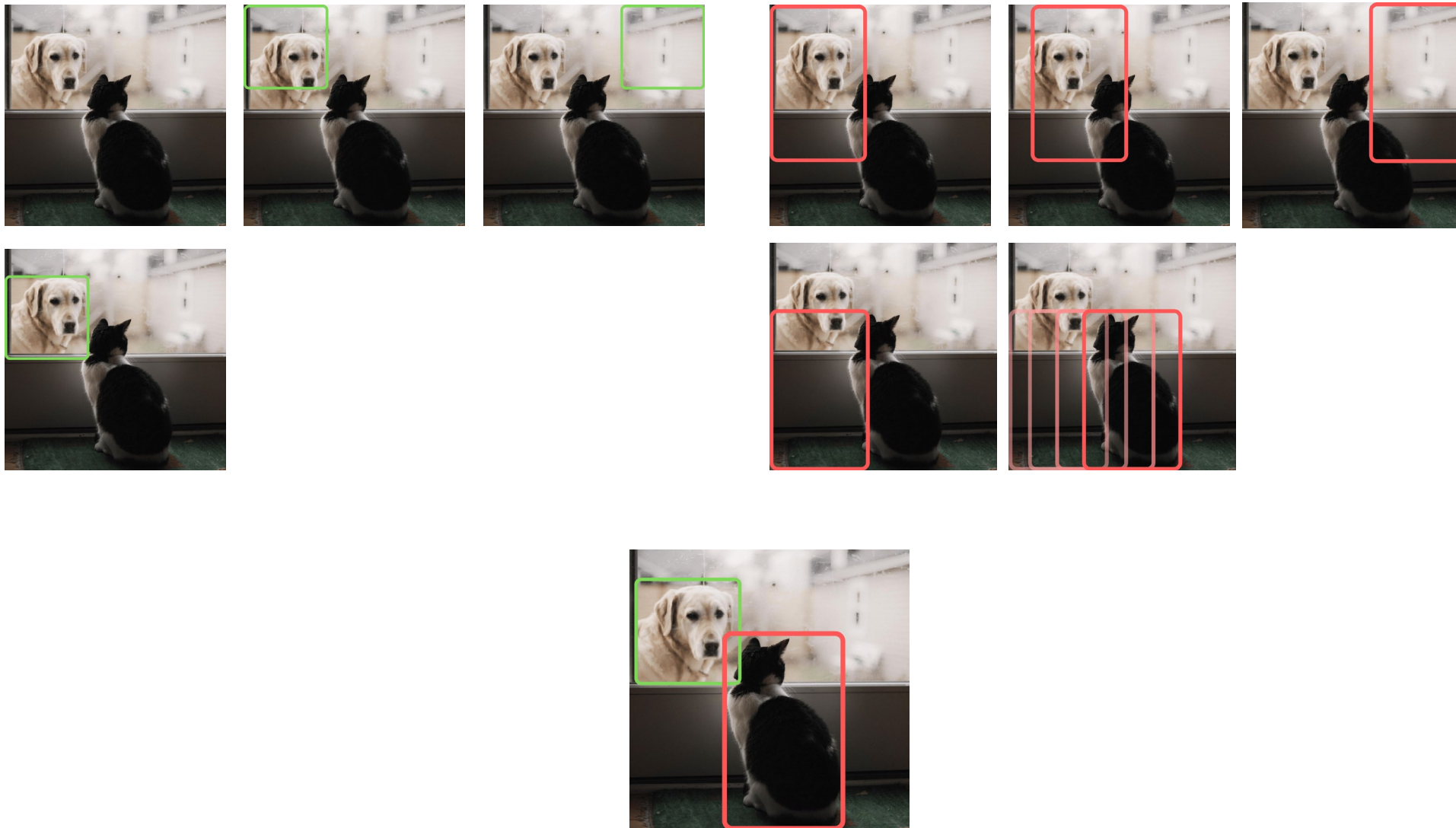
El “**sentido común de ingenieros**” nos dice es: “vamos a iterar”.

Es decir, iteremos un “área reducida” dentro de la foto de izquierda a derecha y de arriba abajo y le aplicamos la CNN pre-entrenada para ver si detecta algo.

- Al ir iterando, lograremos detectar los 2 animales de la foto.



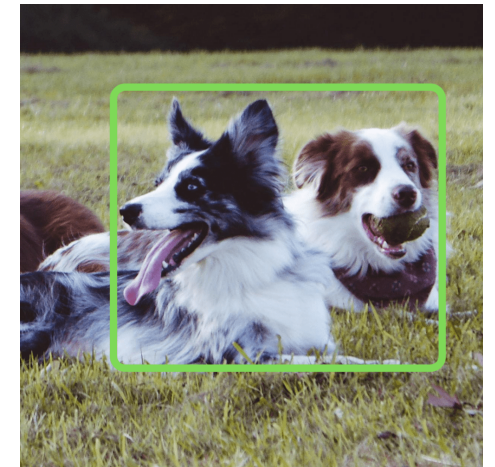
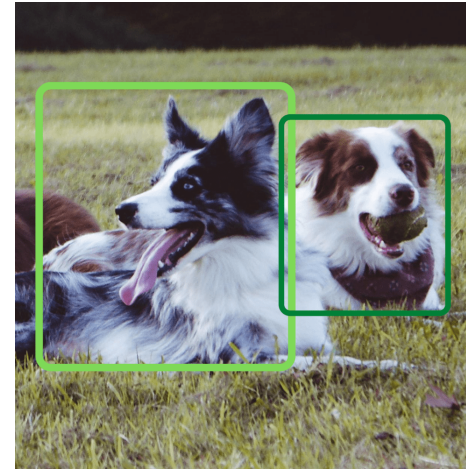
# Detección a partir de la clasificación



# Detección a partir de la clasificación

Inconvenientes:

1. Tamaño de la ventana
2. Cuando es el desplazamiento
- 3. Tiempo de computo**
4. Si detectamos algo ya tengo todas las coords.?
5. Si nos movemos podemos seguir detectando el mismo perro varias veces
6. Animales cerca o lejos





# Detección a partir de la clasificación

De los puntos 5 y 6 anteriores nació la nueva métrica que evalúe la clase y la posición “Bounding Box” (X,Y, ancho y alto)

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)}$$

Mean Average Precision

TP: True positive

FP: False positive

Modelos

# R-CNN: Búsqueda selectiva

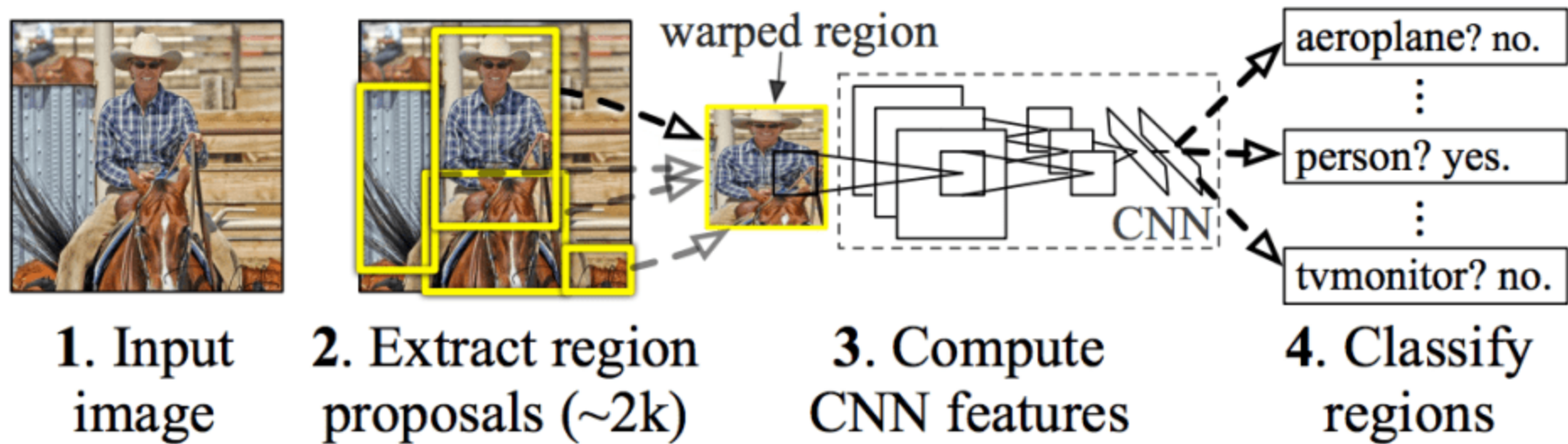
## Region Based Convolutional Neural Networks

1. Primero determine las regiones de interés (selective search)
2. Luego clasifique imágenes con una red pre-entrenada

Es necesario un algoritmo que **determine las áreas de interés**. Luego pasar las regiones por la CNN y un clasificador binario que valide clase correcta y elimine las de poca confianza. Finalmente un regresor ajusta correctamente la posición.

# R-CNN: Búsqueda selectiva

## **R-CNN: *Regions with CNN features***




Fuente: <https://arxiv.org/abs/1311.2524>

# R-CNN: Búsqueda selectiva

La selección podría darse por “Áreas contiguas con mismo color, o líneas delimitantes o cambios de contraste y brillo.

Para evitar solapamiento del mismo objeto se utilizar IoU (Intersection over Union)

**IoU:** nos da un porcentaje de acierto del área de predicción frente a la bounding-box real que queríamos detectar.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


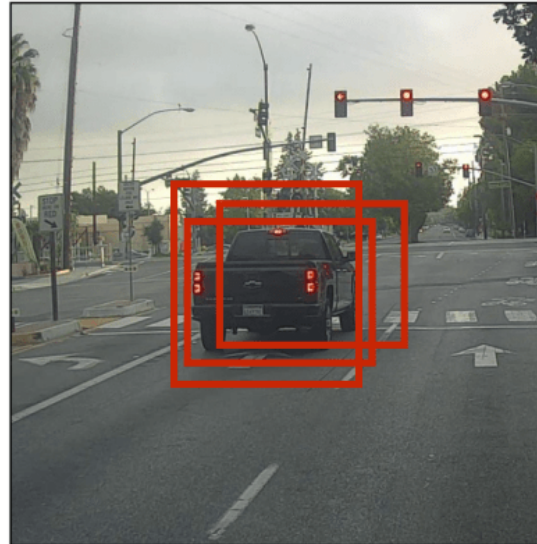
# R-CNN: Búsqueda selectiva

El IoU en un conjunto “Non-Máximo-Supression” ayuda a seleccionar objeto a localizar.

**NMS:** nos permite quedarnos con la mejor caja, y elimina el resto

Costo Comp. Alto, una imagen podría demorar 25 segundos

Before non-max suppression



Non-Max  
Suppression



After non-max suppression



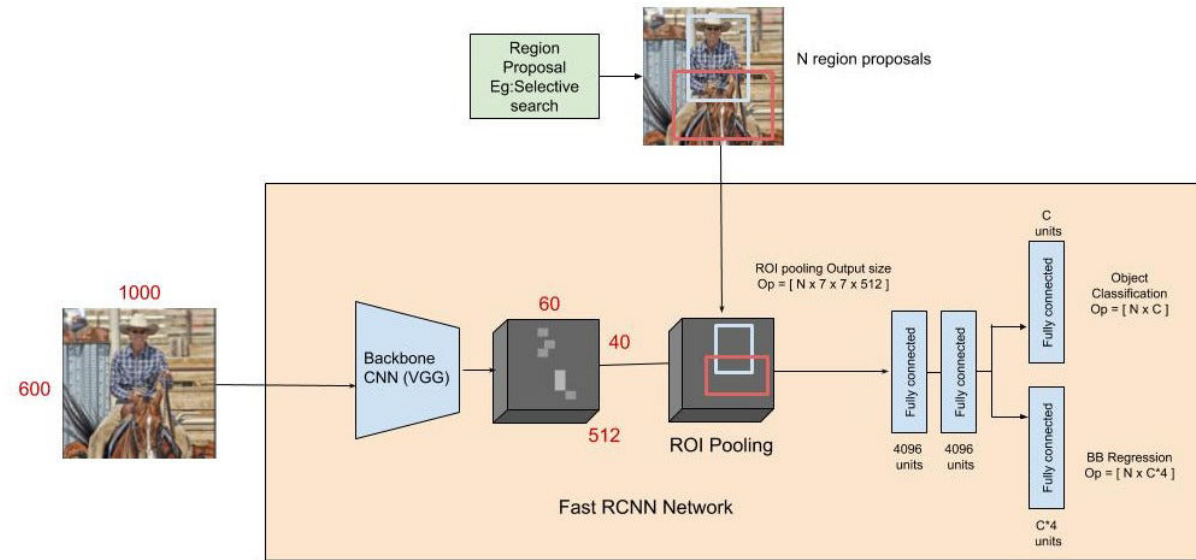


# Mejoras sobre R-CNN: fast y faster R-Cnn

- **Fast R-CNN**

Hace reusó de recursos como las features extraídas por la CNN agilizando el entreno y detección de las imágenes.

Esta nueva red tiene mejoras también en el IOU y en la función de Loss para mejorar el posicionamiento de la “caja delimitante”. Sin embargo no ofrece un aumento dramático de velocidad en el entrenamiento y detección.



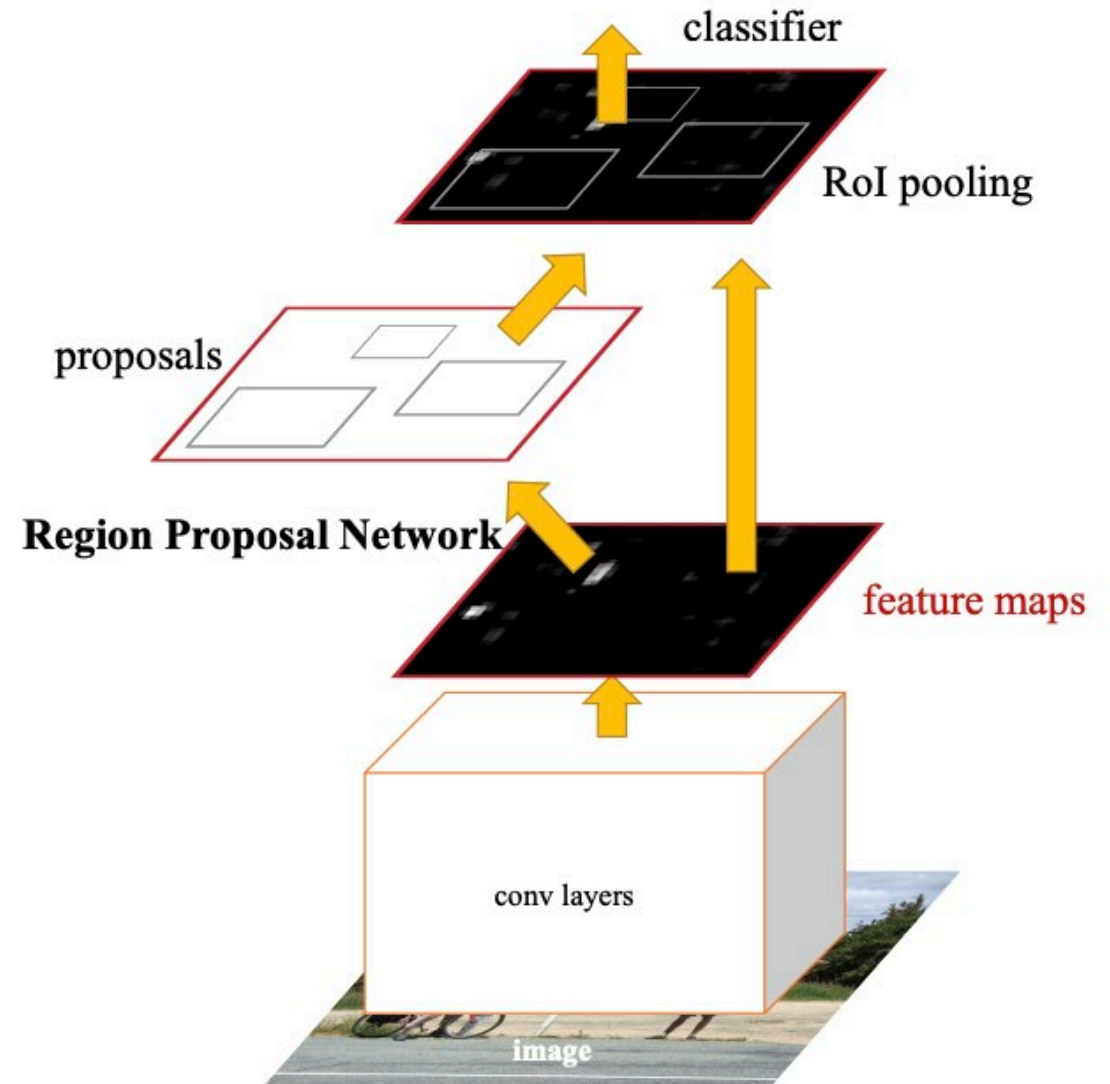
# Mejoras sobre R-CNN: fast y faster R-Cnn

- **Faster R-CNN**

Logra una mejora en velocidad al integrar el algoritmo de “región proposal” sobre la propia CNN.

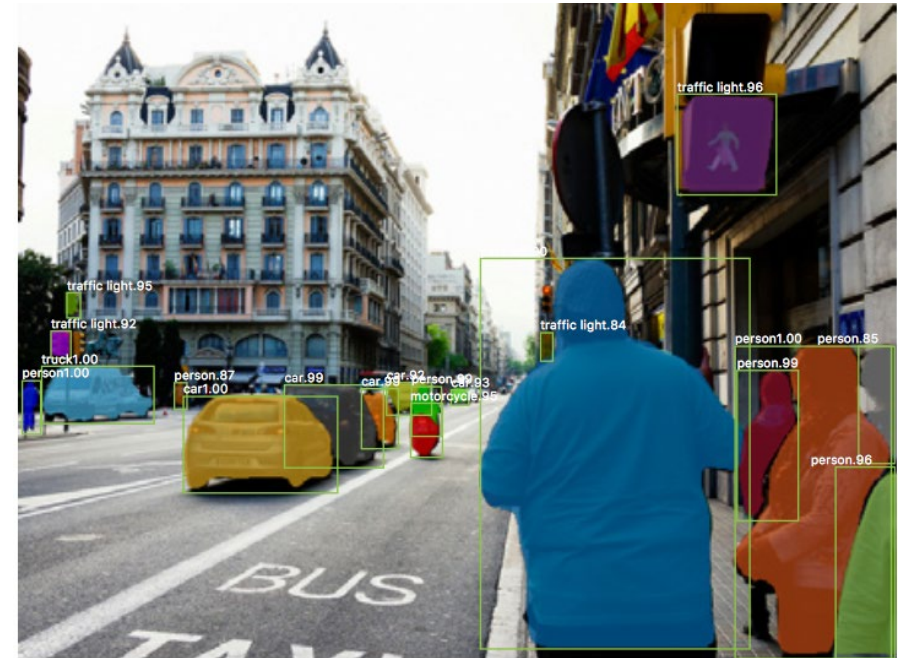
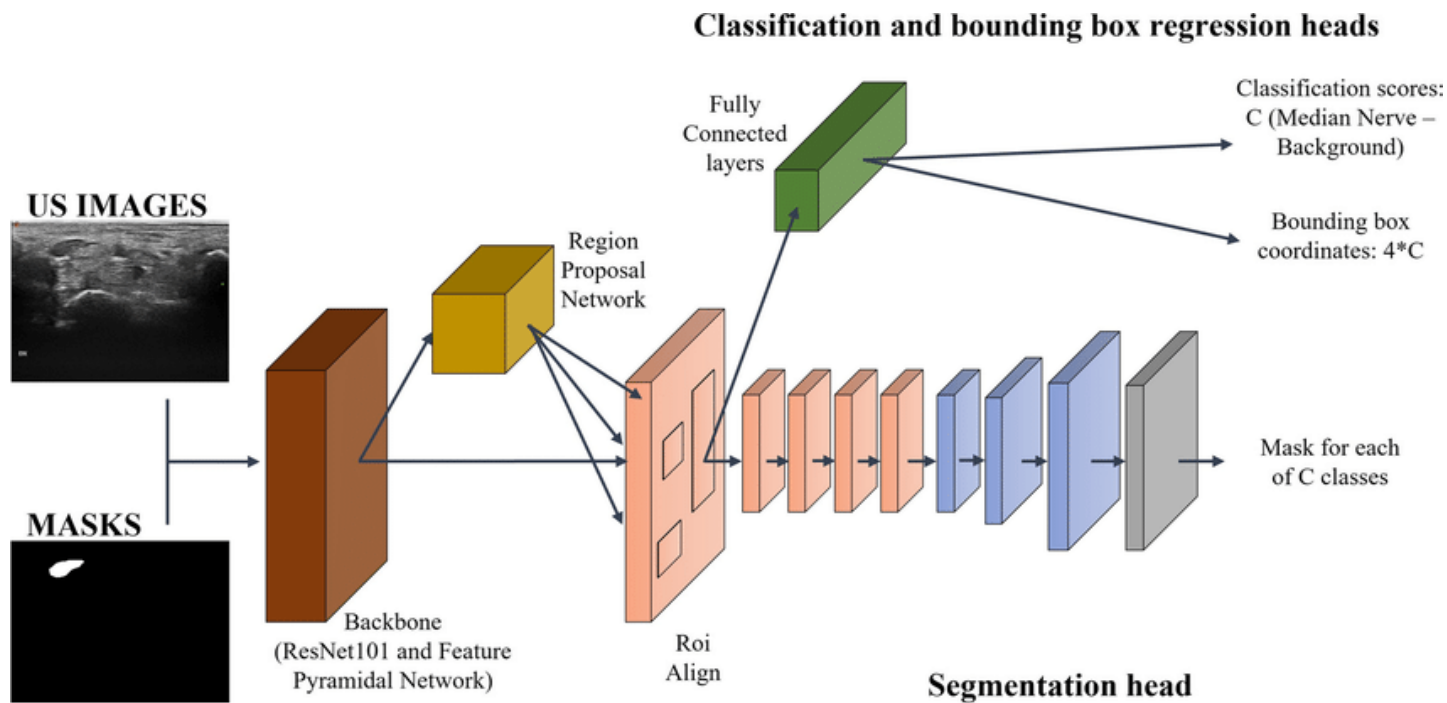
Además aparece el concepto de usar “anchor” fijos, es decir, ciertos tamaños pre calculados para la detección de objetos específicos de la red.

Por ejemplo, podemos definir 3 tamaños de ventana en 3 escalas distintas de tamaños, es decir un total de 9 anclas.



# Mask R-CNN

Esta red, intenta hacer uso de las R-CNN pero en vez de detectar el “bounding box” de cada objeto, intentará hacer segmentación de imagen, definiendo la superficie de cada objeto.



Fuente: <https://arxiv.org/abs/1703.06870>

Modelos reconocidos

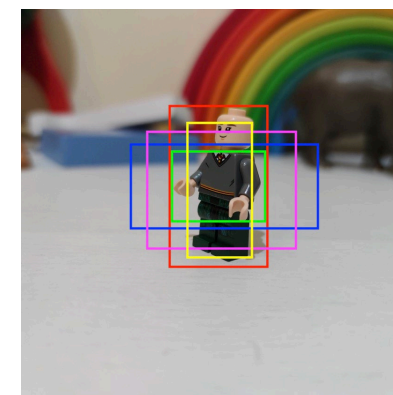
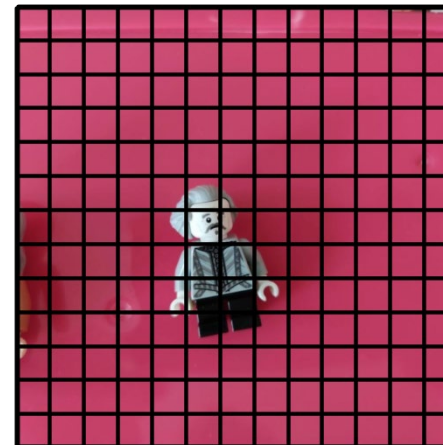
# YOLO: “You Only Look Once”

hace una única pasada a la red convolucional y detecta todos los objetos para los que ha sido entrenada para clasificar.

Al no tener **necesidad de iterar**, logra altas velocidades.  
Permitiendo detección sobre video en tiempo real de cientos de objetos en simultáneo y hasta su ejecución en dispositivos móviles.

# Como funciona YOLO

- Define una grilla de tamaño fijo.
- Sobre esas celdas intentará detectar objetos valiéndose de anchors fijos, por ejemplo de 3 anclas con 3 tamaños distintos (9 predicciones por cada celda).
- Hace uso de IoU y Non-Max-supression.
- También tiene asociada una red de regresión al final para las posiciones de los bounding-boxes.





# YOLO

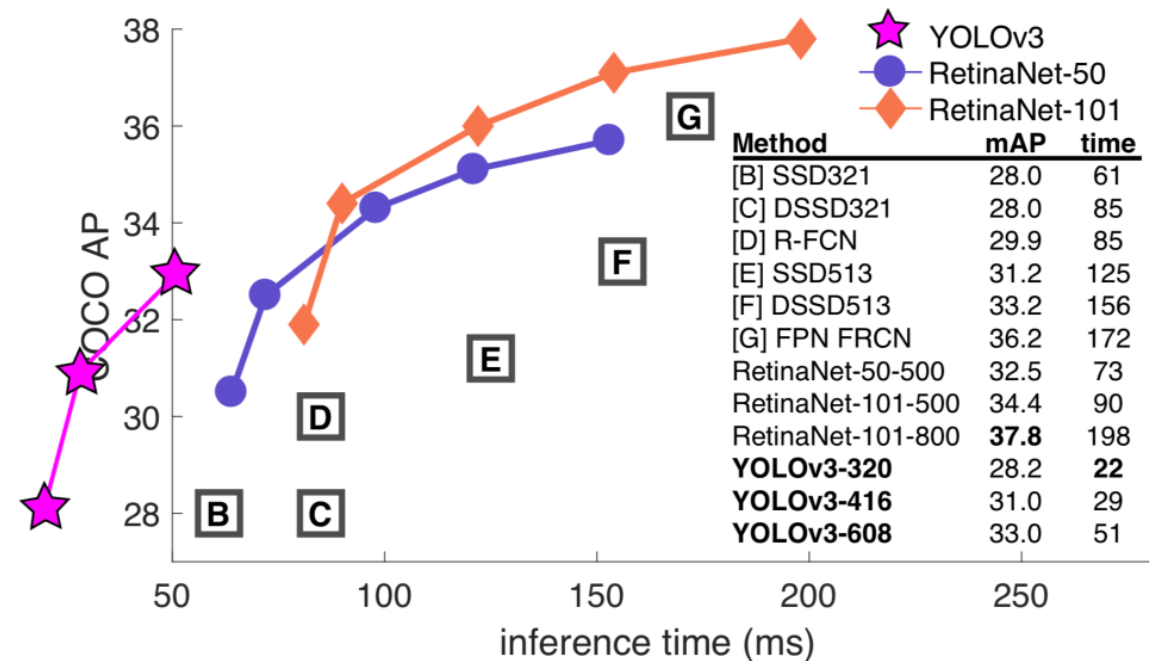
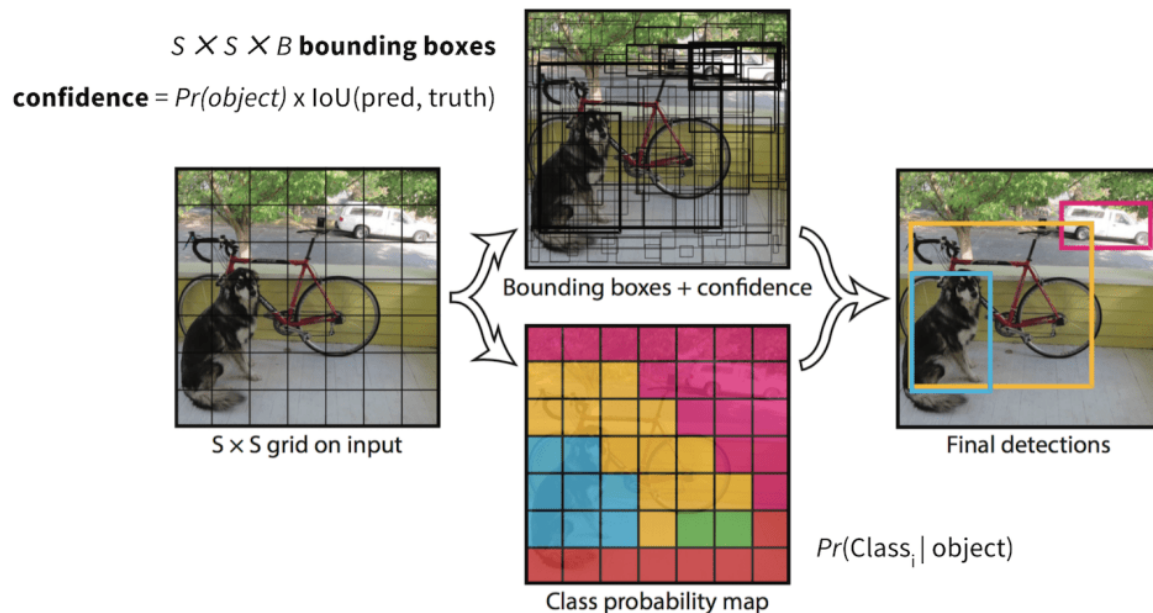
La “grandiosidad” de YOLO consiste en su red CNN. Antes vimos que R-CNN utilizaba algún algoritmo adicional para seleccionar las regiones de interés sobre las que realiza las predicciones. pero YOLO, utiliza la misma CNN de clasificación con un “truco” por el cual **no necesita iterar la grilla**, si no que la propia red se comporta como si hiciera un especie de “offset” que le permite hacer la detección en simultáneo de todas las casillas

YOLO utiliza una red CNN llamada Darknet. Al mismo tiempo de entrenarse se crea la red con este <<offset>> que comentaba.

# YOLO

Suele tener menos porcentaje de acierto que las R-CNN pero es mucho mas rápida.

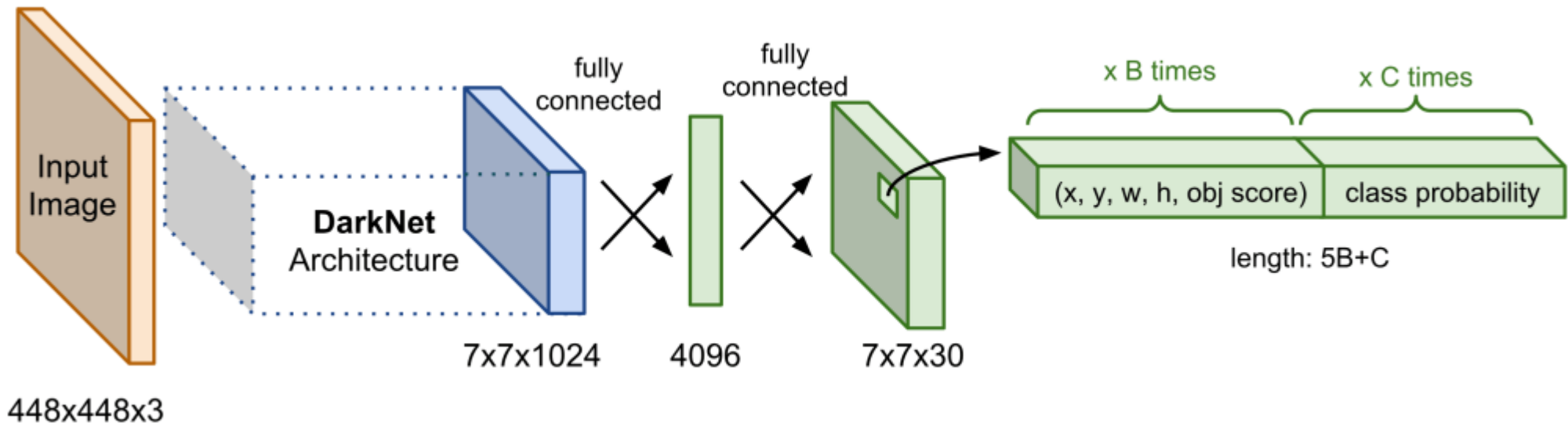
Hoy YOLO V4 mejora la precisión y es igual de rápida.



# Arquitectura YOLO

Se basa en una red convolucional - GoogleNet y consta de 24 capas convolucionales. “Darknet”

Embebe en su salida tanto la parte que clasifica las imágenes como la de posicionamiento y tamaño de los objetos.



# Arquitectura YOLO

Para el CocoDataset que debe detectar 80 objetos diferentes, tendremos como salida.

Tamaño de grilla	Cantidad Anclas	Cantidad de clases	Ccore, X, Y, Alto, Ancho
13 * 13	* 3 *	(80 +	* 5)

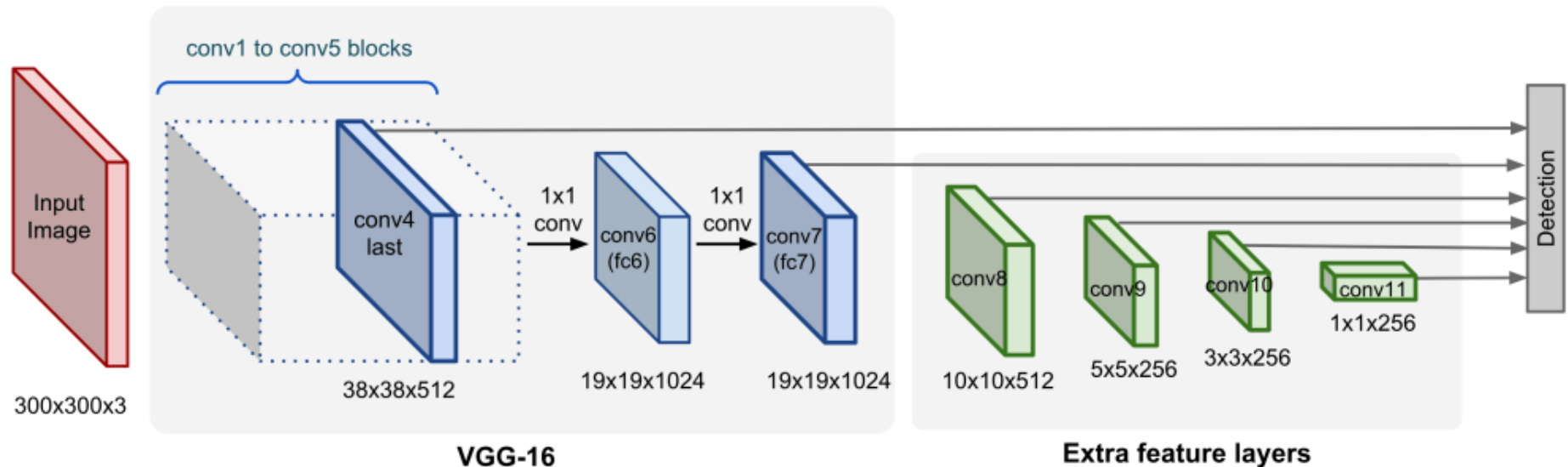
Este ejemplo nos dará un array de 43.095 datos siendo el máximo de objetos que puede detectar y localizar  $13 \times 13 \times 3 = 507$  objetos de 80 clases en la misma foto en una sola pasada. (Realmente hará  $13 \times 13 \times 3 \times 3$  tamaños = 1521 predicciones). Sorprendente!.

# SSD- Single Shot Detector

Tiene una estructura piramidal en su CNN.

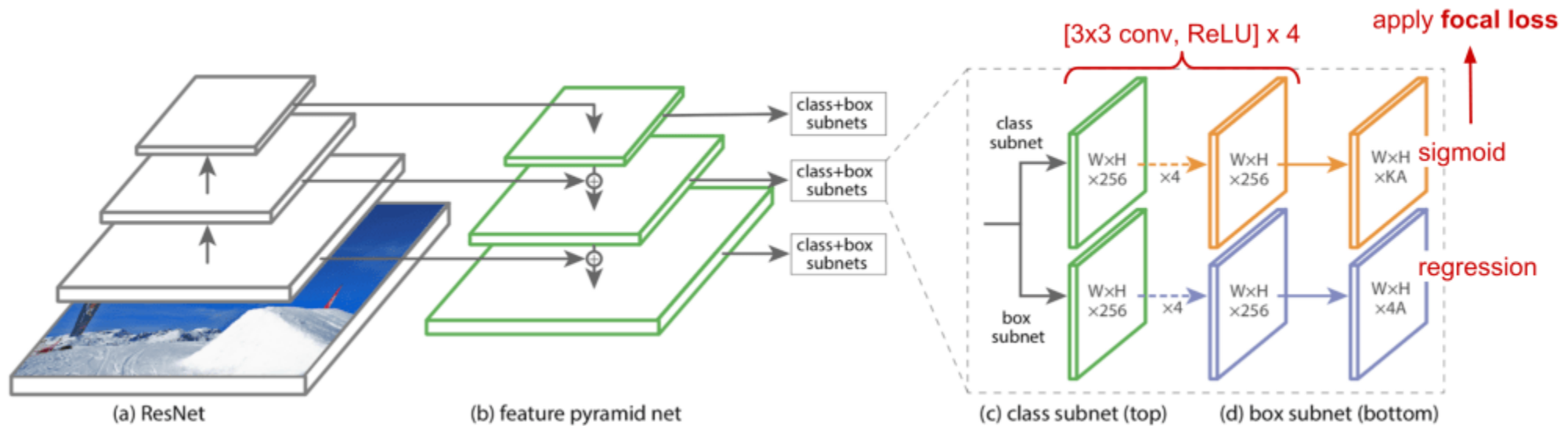
Esto le permite poder detectar objetos grandes y pequeños.

No utiliza una grilla predefinida, pero cuenta con “anclas” de distintas proporciones que se van escalando a medida que descendemos por la pirámide (mapa de features más pequeños, con anclas proporcionalmente más grandes).



# RetinaNet (2018)

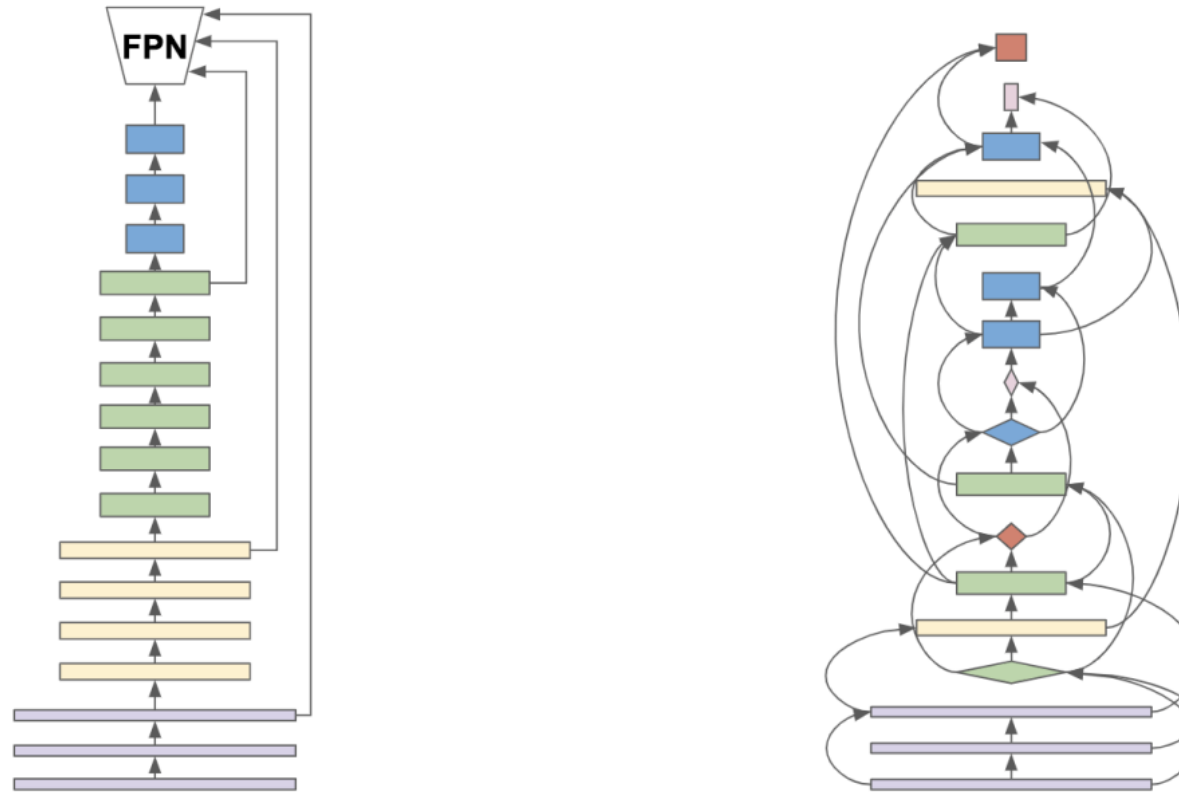
CNN piramidal mejorada para reconocer objetos de diversos tamaños en una sola pasada. Innova con una nueva función de pérdida llamada <<Focal Loss>>.





# Google: Spinet (2019-20)

Rompe con la estructura piramidal y propone una arquitectura novedosa llamada “scale-permuted” en la que se alternan diversos tamaños en las convoluciones.



# Facebook: DETR (2020)

Facebook propone una “[End to End object detection with Transformers](#)“. Es decir, utilizar la más novedosa y efectiva técnica de redes neuronales utilizada en NLP pero aplicada a la detección de imágenes! Muy ingenioso!

