Ciencia y Tecnología

Secretaría de Ciencia, Humanidades, Tecnología e Innovación

CIMAT
UNIDAD MÉRIDA

DETECCIÓN DE OBJETOS USANDO DEEP LEARNING

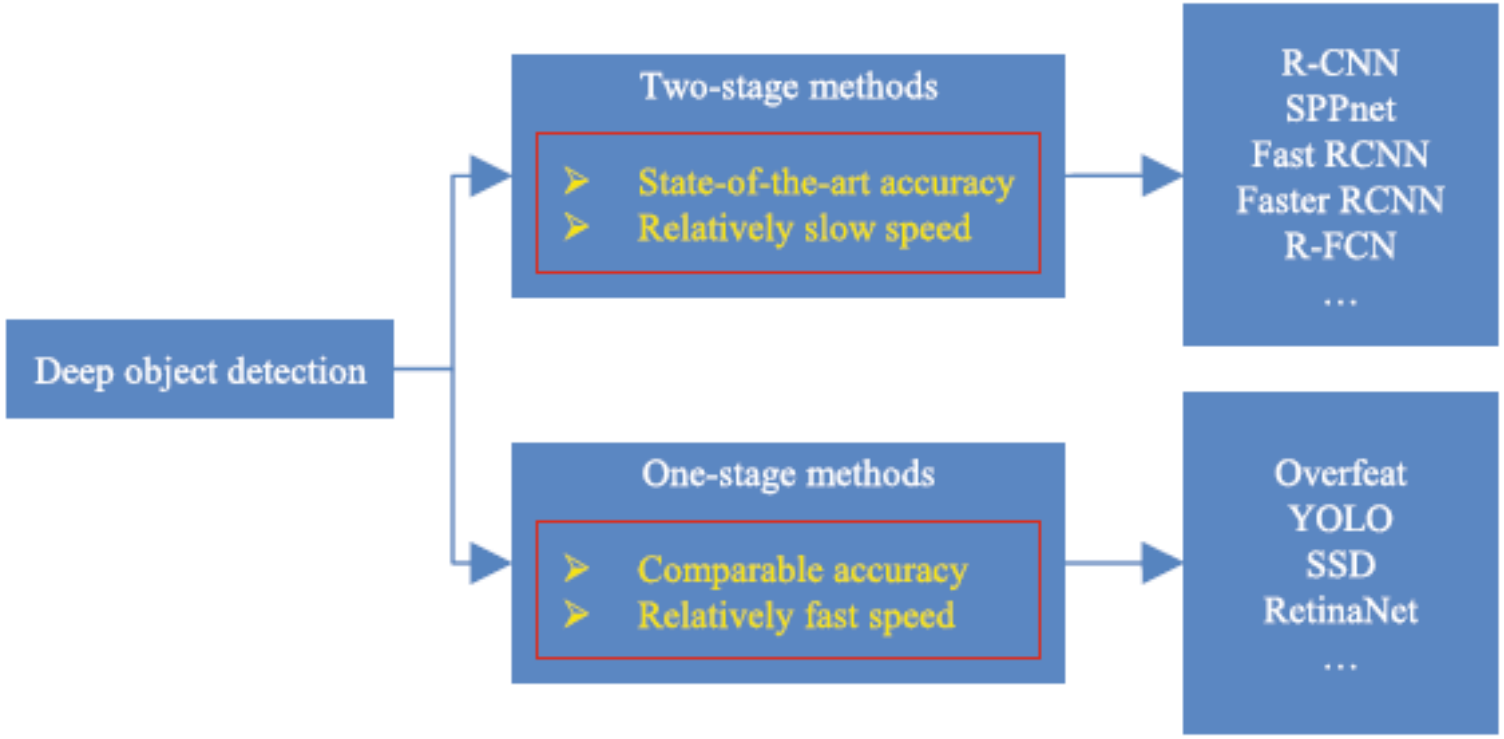
Dr. Francisco J. Hernández López

Investigador por México, CONAHCYT

CIMAT - Unidad Mérida

fcoj23@cimat.mx

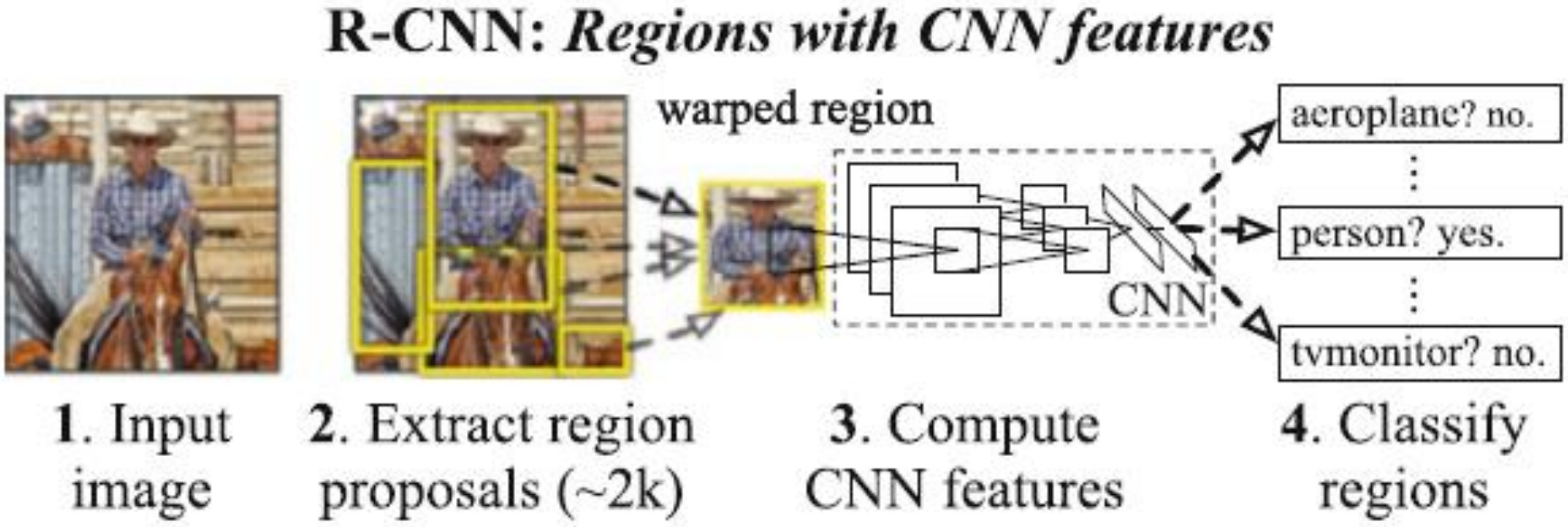
MÉTODOS BASADOS EN DEEP LEARNING



Jiang, Xiaoyue, et al., eds. *Deep Learning in object detection and recognition*. Singapore: Springer, 2019.

R-CNN

- Reg. con caract. CNN o red neur. conv. basada en regiones



BBoxes candidatas

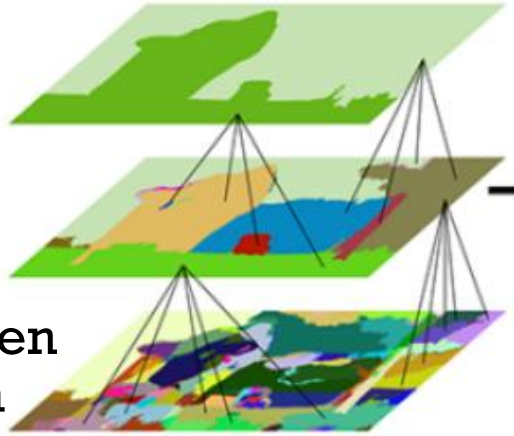
Extracción de caract. en cada BBox candid. con una CNN

Clasificación de la BBox en alguna de las clases

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580-587).

BÚSQUEDA SELECTIVA

1. Superpixeles
2. Agrupar regiones (medir similitud entre regiones)
3. Hasta que la imagen es una sola región



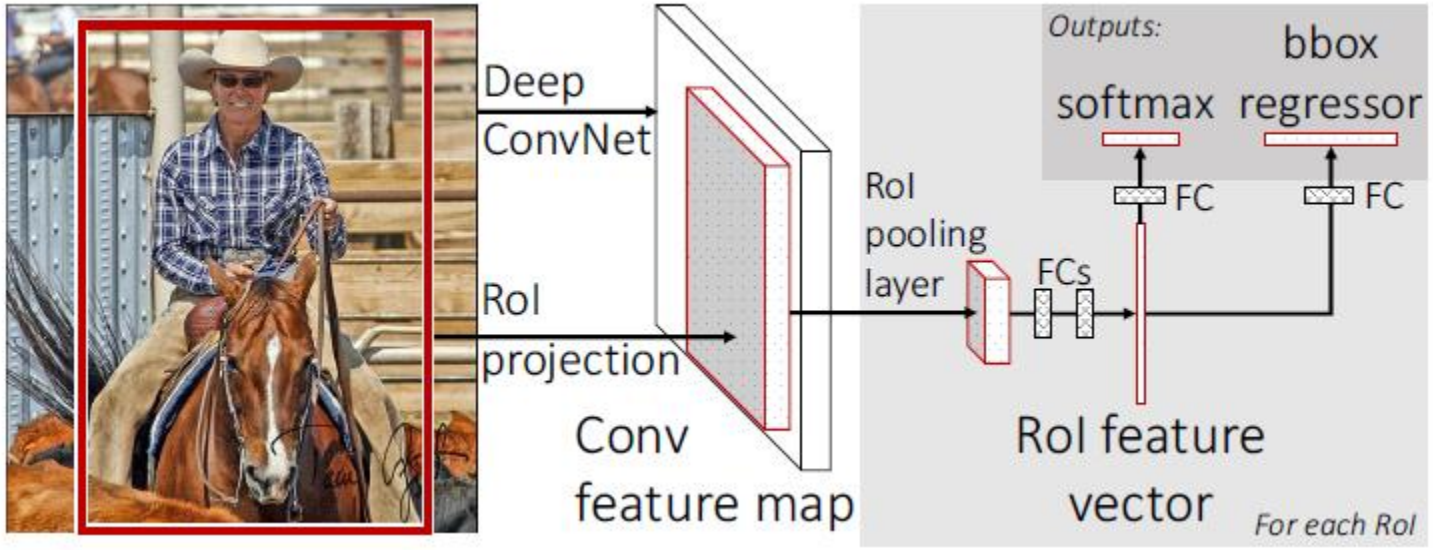
https://pyimagesearch.com/2020/06/29/opencv-selective-search-for-object-detection/

Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smeulders, A. W. (2013). Selective search for object recognition. *International journal of computer vision*, *104*, 154-171.

FAST R-CNN

https://github.com/rbgirshick/fast-rcnn

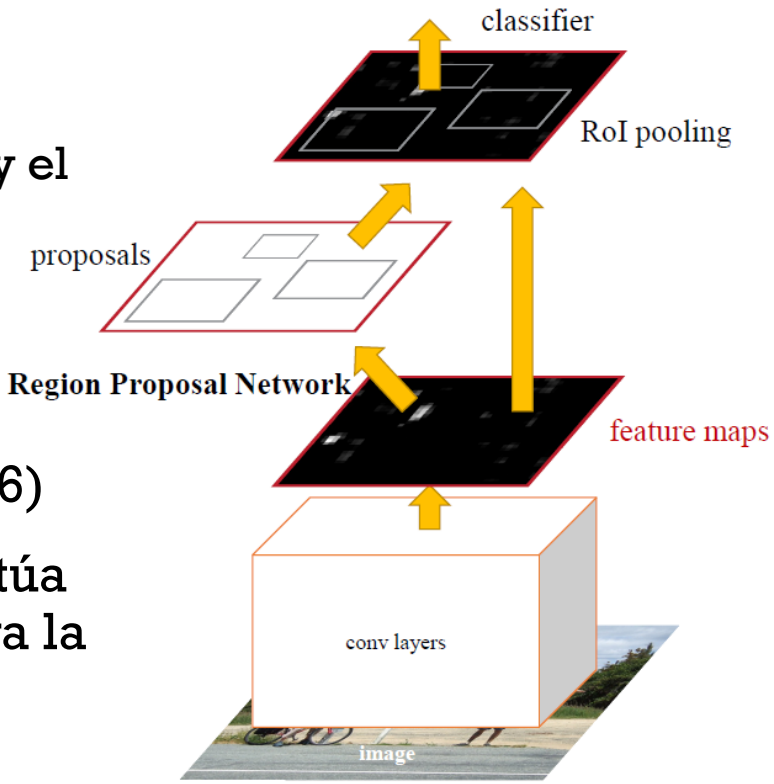
- Dada una imagen y sus BBoxes candidatas:
  - Imagen completa → Red conv. para obtener un mapa de caract.
  - Para cada BBox → Capa pooling RoI extrae vector de caract. de tamaño fijo a partir del mapa de caract.
  - Vector de caract. → capas complet. conec. (FCs)
  - FCs → se ramifican en dos capas FC (clasificación, localización)



R. Girshick, "Fast R-CNN," *2015 IEEE International Conference on Computer Vision (ICCV)*, Santiago, Chile, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.

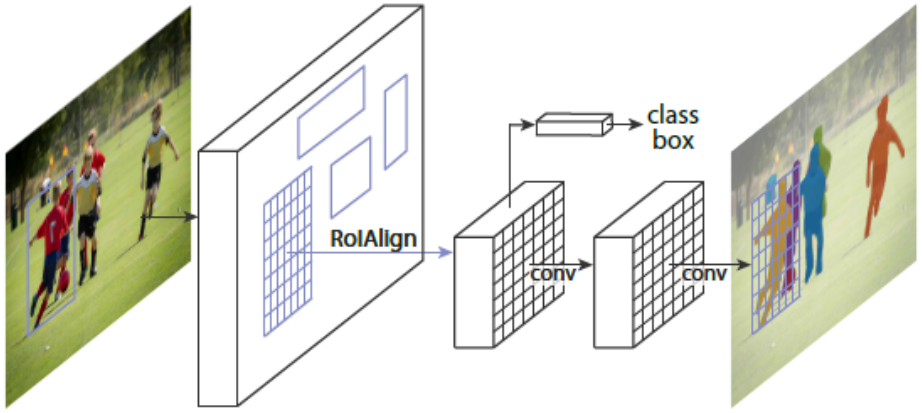
FASTER R-CNN

- Compuesto por dos módulos:
  - RPN. Una CNN que propone regiones y el tipo de obj. en la región
  - Fast R-CNN. Una CNN para extraer las caract., localizar y clasificar a los objs.
- Ambos módulos operan en la misma salida de una CNN profunda (VGG-16)
- La red de propuestas de regiones actúa como un mecanismo de atención para la red Fast R-CNN

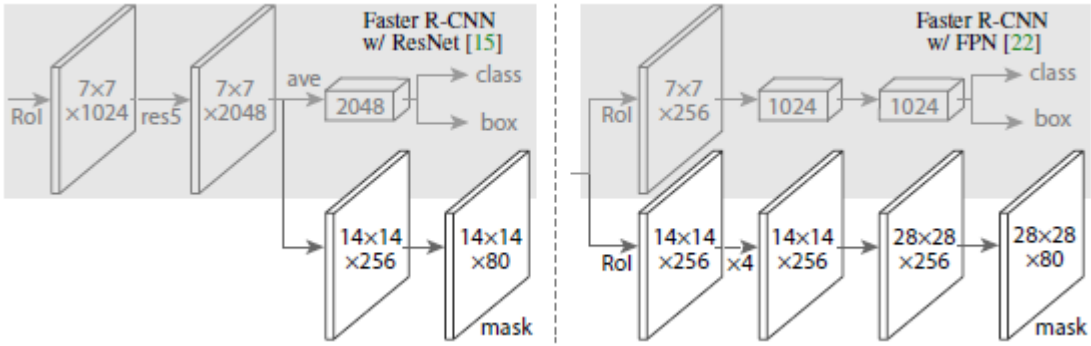


Ren, S., He, K., Girshick, R., & Sun, J. (2016). Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE transactions on pattern analysis and machine intelligence*, *39*(6), 1137-1149.

MASK R-CNN

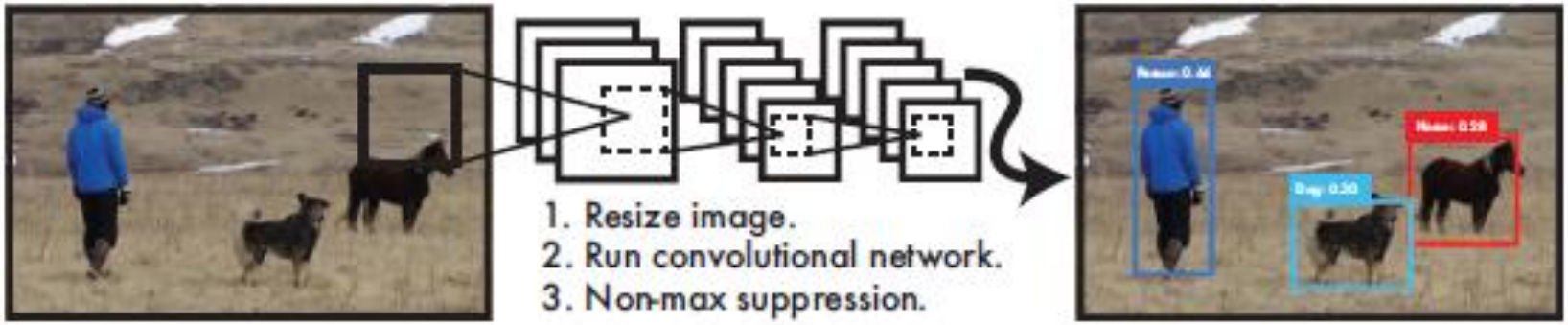


- Se agrega una tercera rama al Faster R-CNN
  - Devuelve una máscara binaria para cada RoI
- Función de error:
  $L = L_{cls} + L_{box} + L_{mask}$
- Permite estimar poses humanas



He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).

YOU ONLY LOOK ONCE (YOLO)



Sistema de detección YOLO. Tomado de [Redmon et al., 2016]

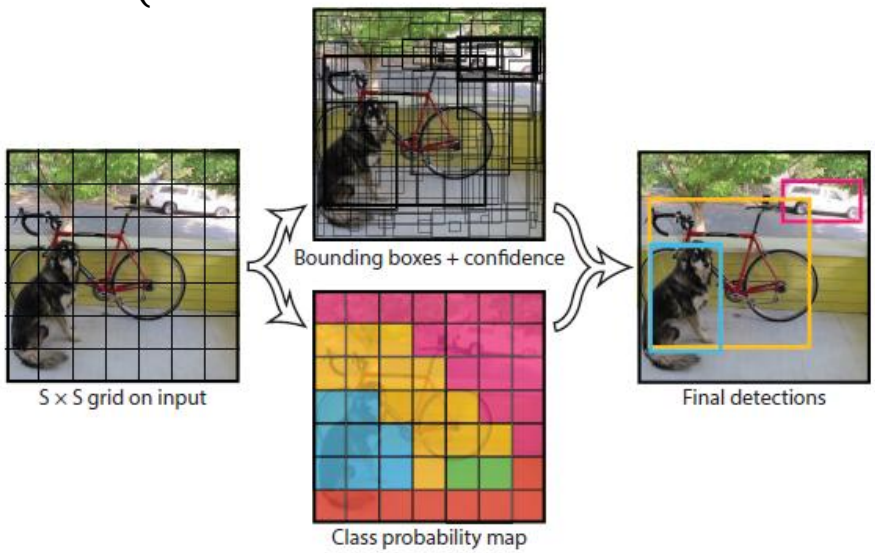
- Se trata de una CNN que predice las BBoxes y las probabilidades de las clases en una sola pasada a la imagen
- Codifica de forma implícita la información de la apariencia del objeto, así como la del contexto (fondo)

[Redmon et al., 2016] Redmon, J., Divvala, S., **Girshick**, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779-788).

YOLO V1

- Divide la imagen de entrada en una malla de $S \times S$ celdas
- Si el centro de un objeto cae en una de las celdas, entonces la celda es responsable de detectar al objeto
- Cada celda, predice B BBoxes y una confianza o prob. definida como:

$$\Pr(Obj) * IOU_{pred}^{truth} = \begin{cases} 0 & si\ no\ hay\ Obj \\ IOU_{pred}^{truth} & otro\ caso \end{cases}$$



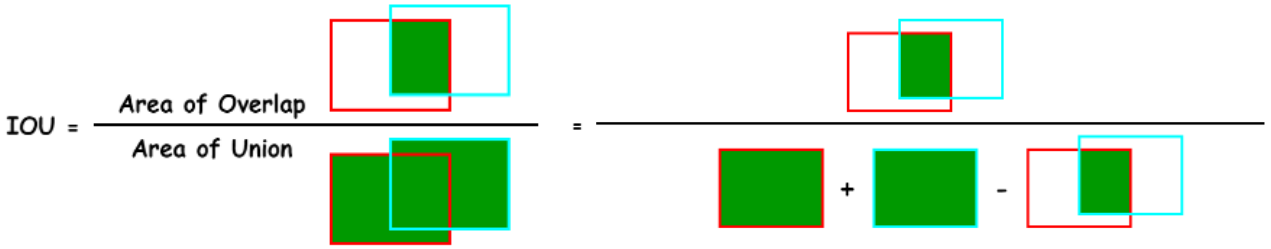
El modelo de YOLO. Tomado de [Redmon et al., 2016]

YOLO V1 [BBOX Y PROBABILIDADES]

- Cada BBox tiene 5 predicciones: x, y, w, h y la confianza o prob.
  - (x, y): Es el centro del BBox relativo a los límites de la celda
  - (w, h): Ancho y alto relativos a la imagen completa
  - Confianza: Representa el IOU entre el BBox predicho y el BBox del ground truth (GT)
- Cada celda también predice C probabilidades condicionales:

  $\Pr(\text{Class}_i|\text{Obj})$ condicionada a que la celda contenga un obj.
- En la fase de prueba, se mutiplican las prob. cond. de la clase y las predicciones de la confianza de las BBox individuales:

$$\Pr(\text{Class}_i|\text{Obj}) * \Pr(\text{Obj}) * \text{IOU}_{\text{pred}}^{\text{truth}} = \Pr(\text{Class}_i) * \text{IOU}_{\text{pred}}^{\text{truth}}$$

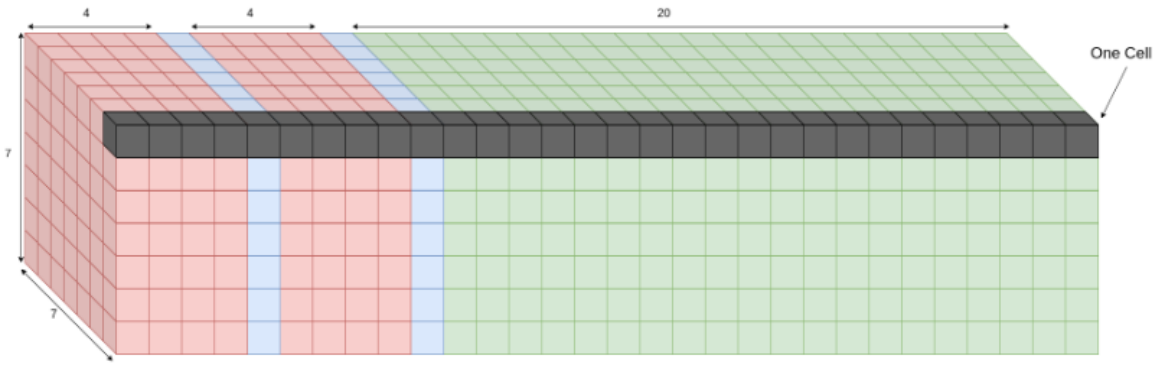


DLB, https://wikidocs.net/167699

YOLO V1 [PREDICCIÓN]

- Las predicciones quedan codificadas en un tensor de $S \times S \times (B * 5 + C)$
- $S \times S$: tamaño de la malla
- B: Número de BBoxes para cada celda
- 5: $(x, y, w, h, confianza)$
- C: Probabilidad condicional de la clase por celda

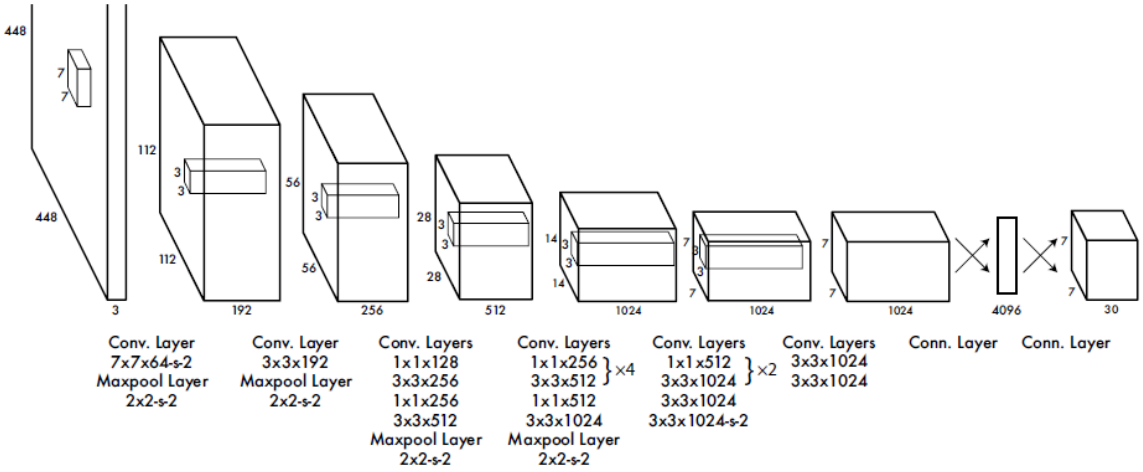
Por ejemplo, para evaluar YOLO en el PASCAL VOC, el tensor es de $7 \times 7 \times (2 * 5 + 20)$



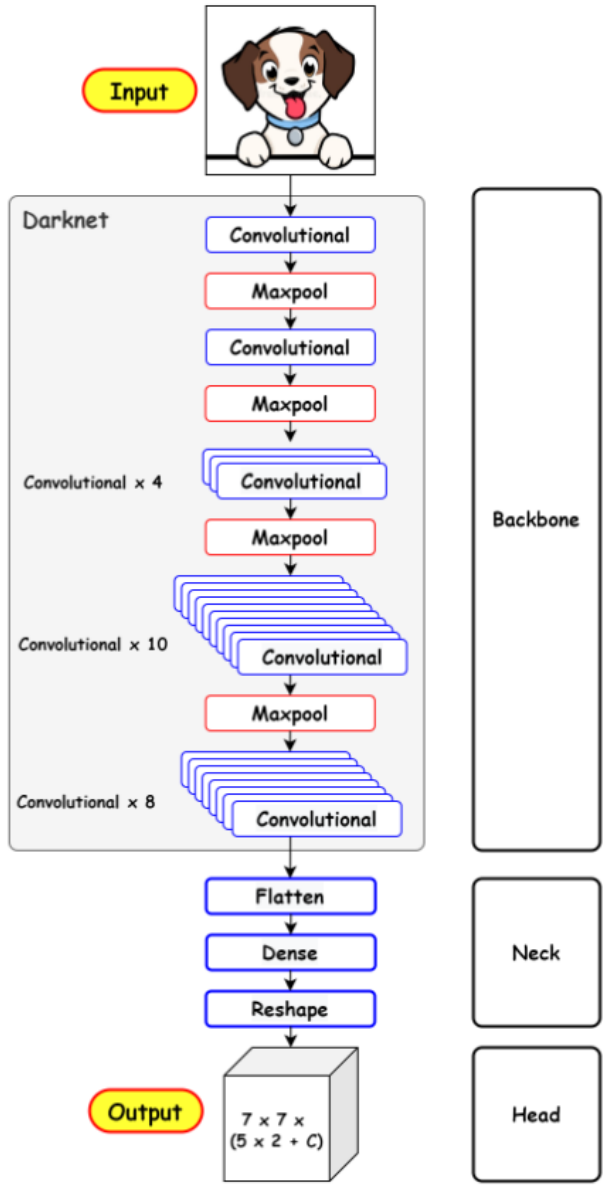
Deep Learning Bible, https://wikidocs.net/167699

ARQUITECTURA DE YOLO V1

- Inspirado en el GoogLeNet
- Sus primeras capas de conv. extraen las caract. a partir de la imagen de entrada
- Las capas complet. conec. predicen las coord. y probabilidades
- Tiene 24 capas de conv. y 2 capas complet. conect.



Arquitectura YOLO v1. Tomado de [Redmon et al., 2016]



DLB, https://wikidocs.net/167699

YOLO V1 [PRE-ENTRENAMIENTO]

- Pre-entrenan las capas de conv. en el conjunto de datos del ImageNet de 1000 clases
- Para el pre-entren. usan las primeras 20 capas de conv. seguidas por una capa de average-pooling y una capa complet. conect.
- Realizan este pre-entren. por aprox. una semana y alcanzan una precisión del 88% en el conjunto de validación del ImageNet 2012

YOLO V1 [ENTRENAMIENTO]

- Luego, convierten el modelo para que realice la detección, agregando 4 capas de conv. Y 2 capas complet. conect. con los pesos inicializados de forma aleatoria
- Se incrementa la resolución de 224×224 a 448×448
- El ancho y alto del BBox se normaliza c. r. al ancho y alto de la imagen quedando entre $[0,1]$
- Las coordenadas (x, y) son parametrizadas como compensaciones (offsets) de una celda en particular y también están acotados entre $[0,1]$
- Se utiliza una función de activación lineal en la última capa
- En las demás capas usan la función de activación:

$$\varphi(x) = \begin{cases} x, & \text{si } x > 0 \\ 0.1x, & \text{otro caso} \end{cases}$$

YOLO V1 (ESTRATEGIAS - OPTIMIZACIÓN)

- Se le da mayor peso a las predicciones de las coord. del BBox que a las predic. de la confianza: $\lambda_{\text{coord}} = 5, \lambda_{\text{noobj}} = 0.5$
- Se predicen de forma parcial la raíz cuadrada del ancho y el alto del BBox
- YOLO predice multiples BBoxes por celda, pero solo se necesita que un predictor de la BBox sea responsable para cada Obj., entonces se asigna un predictor "responsable" para predecir un Obj. con base en que predicción tiene el $\text{IOU}_{\text{pred}}^{\text{truth}}$ más alto

YOLO V1 [FUNCIÓN DE COSTO]

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

$\mathbb{1}_{ij}^{\text{obj}}$ denota que el j-ésimo BBox en la celda i es responsable para detector el Obj.

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

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

$\mathbb{1}_{ij}^{\text{noobj}}$ es el complemento de $\mathbb{1}_{ij}^{\text{obj}}$

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

$\mathbb{1}_{i}^{\text{obj}}$ denota que el Obj. aparace en la celda i

YOLO V1 [PRUEBA O INFERENCIA]

- Solo requiere de una evaluación de la CNN
- El diseño de la malla fuerza la diversidad espacial en las predicciones del BBox
- Es claro cuando un obj. cae en una celda y la red predice solamente una caja para cada obj., sin embargo, obj. grandes u obj. cerca del borde de multiples celdas pueden ser localizados por múltiples celdas
- Se aplica un Non-maximal suppression, con esto el mAP llega a subir un 2% o hasta un 3%

YOLO V1 [LIMITANTES]

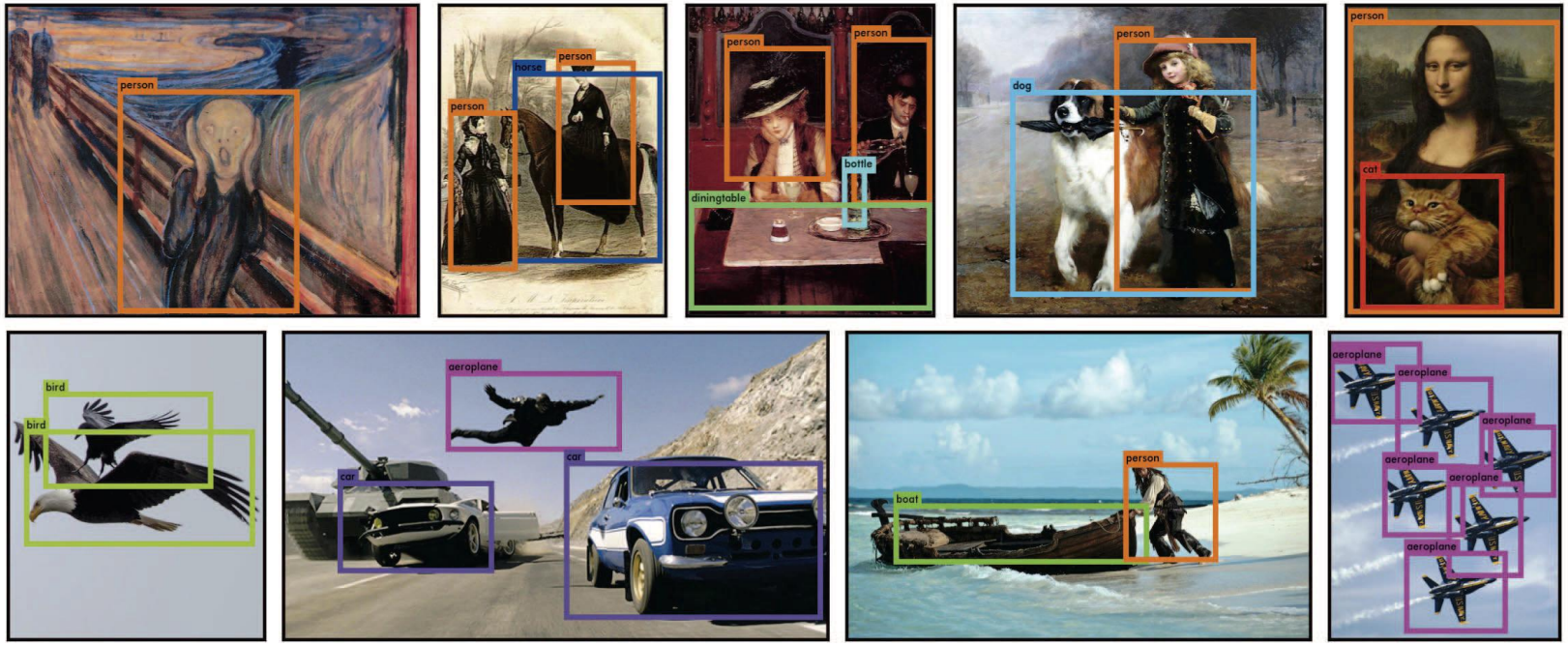
- El modelo sufre con obj. pequeños que aparecen en grupos como por ejemplo una bandada de pájaros
- Ya que el modelo aprende a predecir desde los datos, éste batalla para generalizar obj. en nuevas o inusuales razones de aspecto o configuraciones
- El modelo utiliza características burdas para predecir los BBox, ya que la arquitectura tiene múltiples capas de submuestreos (downsampling) a partir de la imagen de entrada
- La función de pérdida trata los errores en obj. pequeños y grandes de la misma forma

RESULTADOS DE YOLO V1 EN EL VOC 2012

VOC 2012 test	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
MR_CNN_MORE_DATA [11]	**73.9**	**85.5**	**82.9**	**76.6**	**57.8**	**62.7**	**79.4**	77.2	86.6	**55.0**	**79.1**	**62.2**	87.0	**83.4**	**84.7**	78.9	45.3	73.4	65.8	80.3	74.0
HyperNet_VGG	71.4	84.2	78.5	73.6	55.6	53.7	78.7	**79.8**	87.7	49.6	74.9	52.1	86.0	81.7	83.3	**81.8**	**48.6**	**73.5**	59.4	79.9	65.7
HyperNet_SP	71.3	84.1	78.3	73.3	55.5	53.6	78.6	79.6	87.5	49.5	74.9	52.1	85.6	81.6	83.2	81.6	48.4	73.2	59.3	79.7	65.6
Fast R-CNN + YOLO	70.7	83.4	78.5	73.5	55.8	43.4	79.1	73.1	**89.4**	49.4	75.5	57.0	**87.5**	80.9	81.0	74.7	41.8	71.5	68.5	**82.1**	67.2
MR_CNN_S_CNN [11]	70.7	85.0	79.6	71.5	55.3	57.7	76.0	73.9	84.6	50.5	74.3	61.7	85.5	79.9	81.7	76.4	41.0	69.0	61.2	77.7	72.1
Faster R-CNN [27]	70.4	84.9	79.8	74.3	53.9	49.8	77.5	75.9	88.5	45.6	77.1	55.3	86.9	81.7	80.9	79.6	40.1	72.6	60.9	81.2	61.5
DEEP_ENS_COCO	70.1	84.0	79.4	71.6	51.9	51.1	74.1	72.1	88.6	48.3	73.4	57.8	86.1	80.0	80.7	70.4	46.6	69.6	**68.8**	75.9	71.4
NoC [28]	68.8	82.8	79.0	71.6	52.3	53.7	74.1	69.0	84.9	46.9	74.3	53.1	85.0	81.3	79.5	72.2	38.9	72.4	59.5	76.7	68.1
Fast R-CNN [14]	68.4	82.3	78.4	70.8	52.3	38.7	77.8	71.6	89.3	44.2	73.0	55.0	**87.5**	80.5	80.8	72.0	35.1	68.3	65.7	80.4	64.2
UMICH_FGS_STRUCT	66.4	82.9	76.1	64.1	44.6	49.4	70.3	71.2	84.6	42.7	68.6	55.8	82.7	77.1	79.9	68.7	41.4	69.0	60.0	72.0	66.2
NUS_NIN_C2000 [7]	63.8	80.2	73.8	61.9	43.7	43.0	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	38.6	68.3	58.0	68.7	63.3
BabyLearning [7]	63.2	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6
NUS_NIN	62.4	77.9	73.1	62.6	39.5	43.3	69.1	66.4	78.9	39.1	68.1	50.0	77.2	71.3	76.1	64.7	38.4	66.9	56.2	66.9	62.7
R-CNN VGG BB [13]	62.4	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3
R-CNN VGG [13]	59.2	76.8	70.9	56.6	37.5	36.9	62.9	63.6	81.1	35.7	64.3	43.9	80.4	71.6	74.0	60.0	30.8	63.4	52.0	63.5	58.7
YOLO	57.9	77.0	67.2	57.7	38.3	22.7	68.3	55.9	81.4	36.2	60.8	48.5	77.2	72.3	71.3	63.5	28.9	52.2	54.8	73.9	50.8
Feature Edit [32]	56.3	74.6	69.1	54.4	39.1	33.1	65.2	62.7	69.7	30.8	56.0	44.6	70.0	64.4	71.1	60.2	33.3	61.3	46.4	61.7	57.8
R-CNN BB [13]	53.3	71.8	65.8	52.0	34.1	32.6	59.6	60.0	69.8	27.6	52.0	41.7	69.6	61.3	68.3	57.8	29.6	57.8	40.9	59.3	54.1
SDS [16]	50.7	69.7	58.4	48.5	28.3	28.8	61.3	57.5	70.8	24.1	50.7	35.9	64.9	59.1	65.8	57.1	26.0	58.8	38.6	58.9	50.7
R-CNN [13]	49.6	68.1	63.8	46.1	29.4	27.9	56.6	57.0	65.9	26.5	48.7	39.5	66.2	57.3	65.4	53.2	26.2	54.5	38.1	50.6	51.6

YOLO no es el más preciso, pero es el único que detecta objetos en tiempo real (45 fps). Tomado de [Redmon et al., 2016]

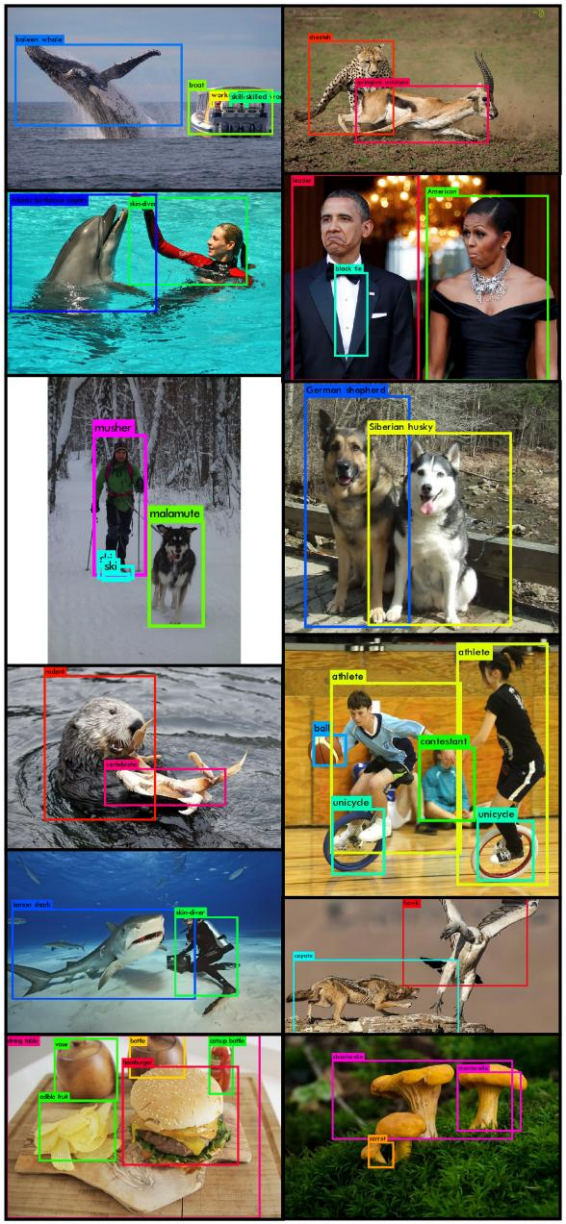
RESULTADOS DE YOLO V1



Probando YOLO v1 en imágenes de obras de arte y tomadas de internet.
Tomado de [Redmon et al., 2016]

YOLO V2 - YOLO 9000

- Puede detectar 9000 categorías de Obj.
- Obtiene 67 fps con 76.8% de mAP en el VOC 2007
- Se entrena YOLO 9000 de forma simultánea en los conjuntos de datos de COCO y el ImageNet
- Obtiene un 19.7% de mAP en el conjunto de validación del ImageNet para 44 de las 200 clases



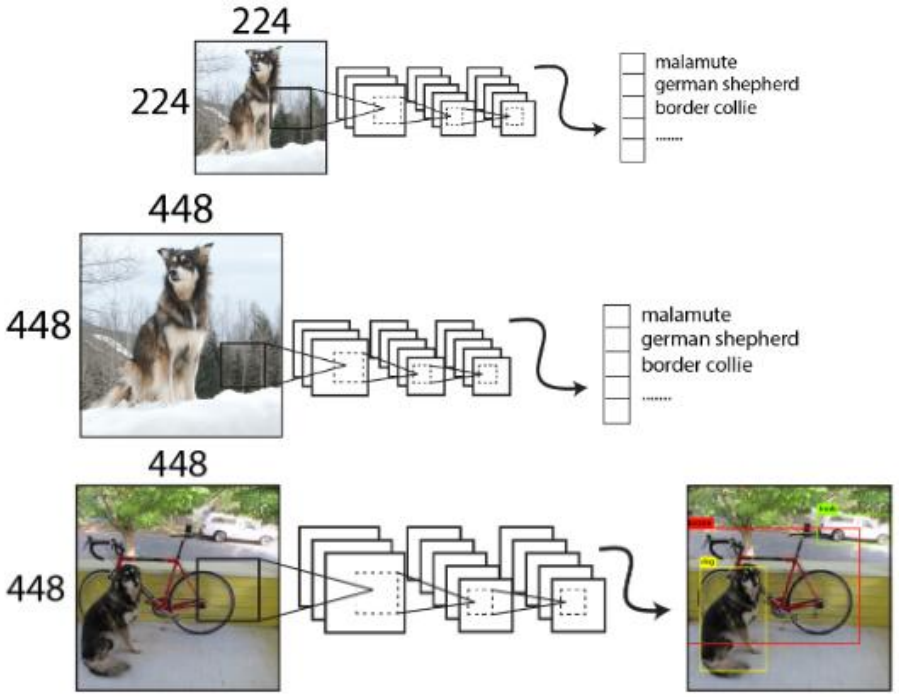
[Redmon y Farhadi, 2017] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 7263-7271).

YOLO V2 [BETTER]

- YOLO v1 tiene una exhaustividad (*recall)* bajo comparado con métodos basados en regiones. Entonces se enfocan en mejorar el *recall* manteniendo la exactitud en la clasificación
- En lugar de escalar la red, simplifican la red y hacen la representación más fácil de aprender
  - Agregando Normalización por Lotes (*Batch Normalization*) en todas las capas de conv. se obtiene más del 2% de mejora en el mAP, con esto se puede remover el *dropout (*remover de forma aleatoria ciertas neuronas en la red neuronal*)*

YOLO V2 [CLASIFICADOR DE ALTA RESOLUCIÓN]

- Primero, se entrena el clasif. en el ImageNet con una resol. de 224×224 por 160 épocas
- Luego se afina el modelo entren. con una resol. de 448×448 por 10 épocas
- Y se entrena toda la red para detección con la resol. de 448×448
- Esto da un incremento de aprox. 4% en el mAP



https://docs.google.com/presentation/d/14qBAiyhMOFl_wZW4dA1CkixgXwf0zKGbpw_0oHK8yEM/edit#slide=id.g23eff386b7_0_24

YOLO V2 [CAJAS DE ANCLAJE (ANCHOR BOXES)]

- YOLO v1 predice las coord. de los BBox directamente usando capas complet. conect. en la parte superior del extractor de caract. conv.
- El método RPN (Region Proposal Network) en el Faster R-CNN predice compensaciones (offsets) y confianzas para cajas de anclaje
- Predecir compens. en lugar de coord. simplifica el problema y hace que la red aprenda más fácil
- Entonces, se remueven las capas complet. conect. de YOLO v1 y se usan cajas de anclaje para predecir los BBox
- Eliminan una capa pooling para tener la salida de la capa de conv. de alta resolución

YOLO V2 [AGRUPAMIENTO DE LOS BBOXES]

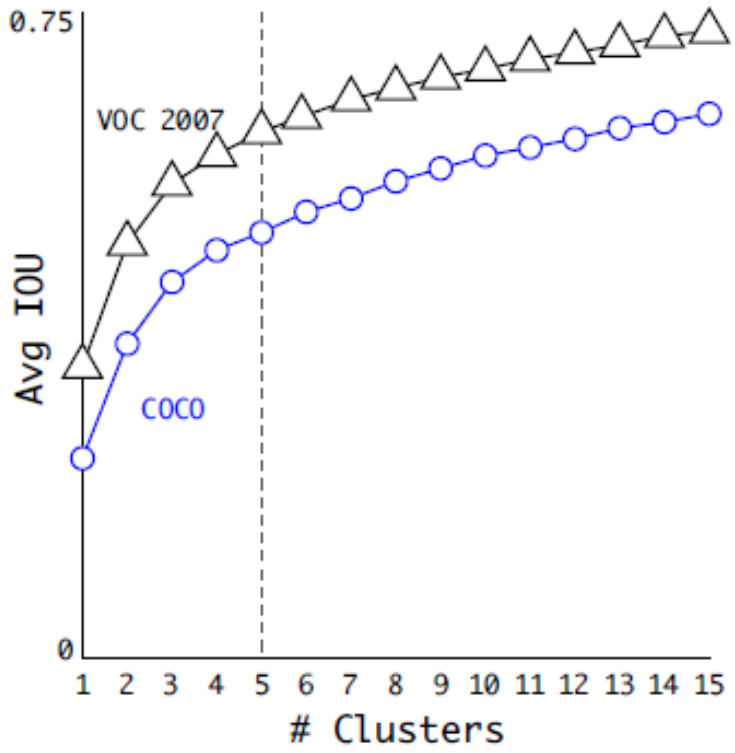
- Un detalle con usar cajas de anclaje es que estas se eligen a mano
- En lugar de elegirlas a mano, se ejecuta un K-means en el conjunto de entrenamiento para estimar las dimensiones de forma automática
- Si se usa K-means con la dist. Euclidiana, entonces cajas grandes generan más error que las cajas pequeñas, por lo que ellos usan:

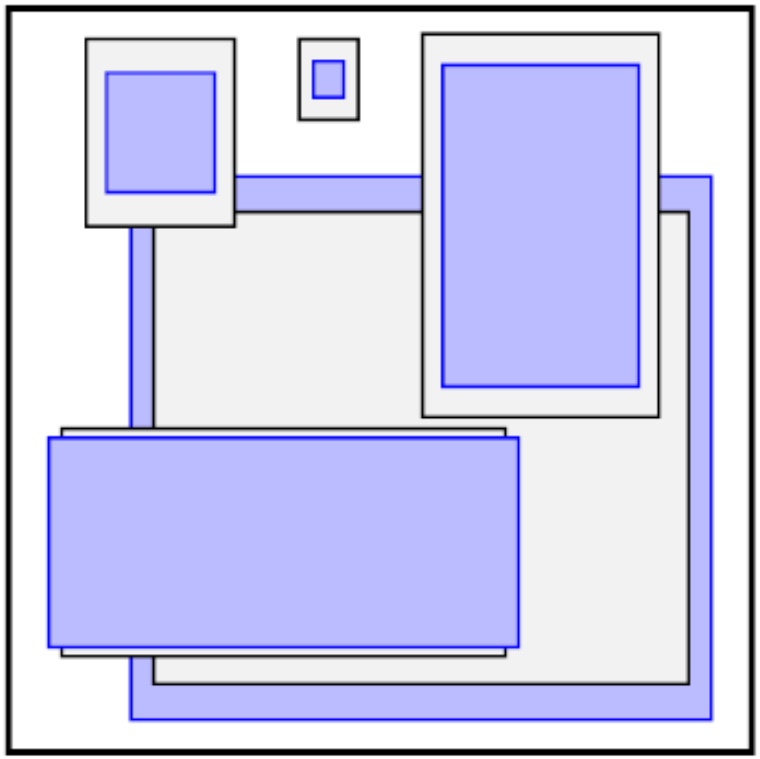
$$d(\text{box}, \text{centroid}) = 1 - \text{IOU}(\text{box}, \text{centroid})$$

- Ejecutan K-means para diferentes valores de K y grafican el promedio del IOU con el centroide más cercano

YOLO V2 [AGRUPAMIENTO EN VOC Y COCO]

Aquí vemos los diferentes tamaños elegidos por agrupamiento para las cajas de anclaje





Agrupamiento en VOC y COCO. Tomado de [Redmon y Farhadi, 2017]

YOLO V2 [PREDICCIÓN DE LA LOCALIZACIÓN]

- Si la celda está desplazada desde la esquina sup. izq. de la imagen por (c_x, c_y) y el BBox apriori tiene ancho p_w y alto p_h, entonces las predicciones se calculan de la sig. manera:

$$b_x = \sigma(t_x) + c_x$$

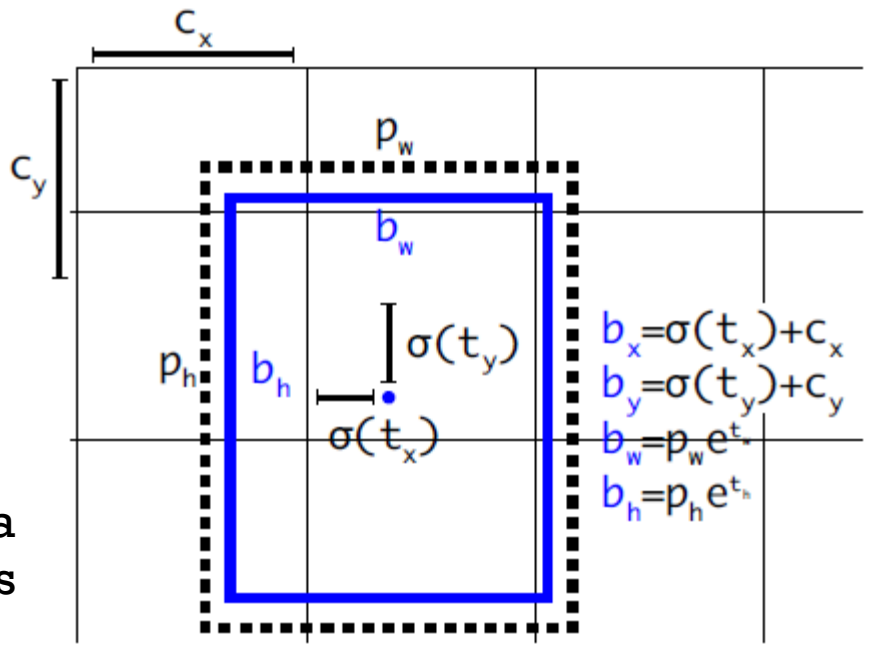
$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w e^{t_w}$$

$$b_h = p_h e^{t_h}$$

$$\Pr(\text{Obj}) * \text{IOU}(b, \text{Obj}) = \sigma(t_o)$$

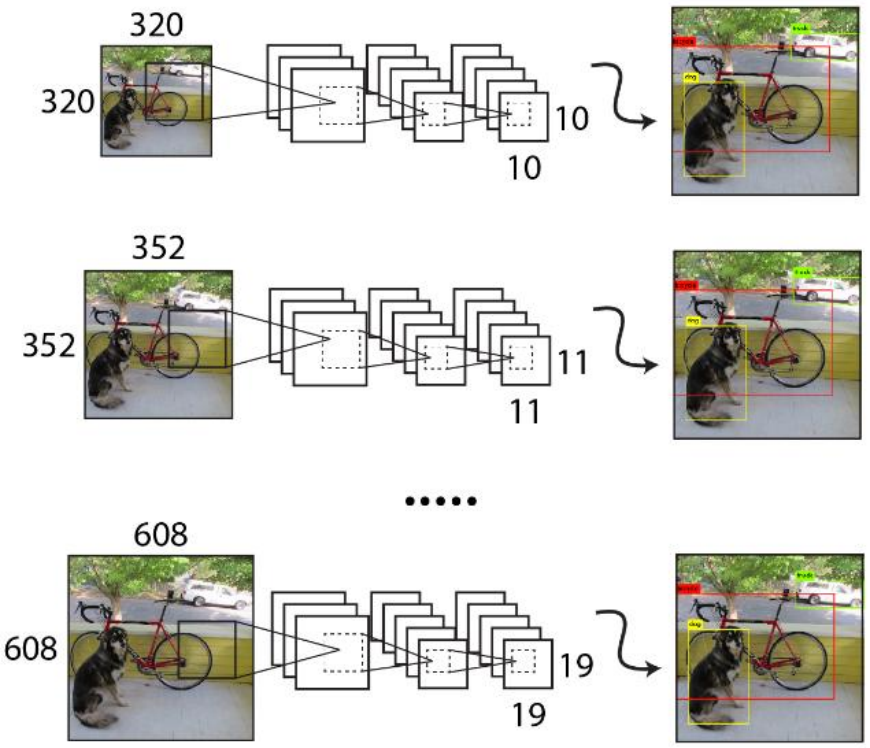
Esto mejora en un 5% sobre la versión en donde se predicen las compensaciones



Tomado de [Redmon y Farhadi, 2017]

YOLO V2 [ENTRENAMIENTO MULTI-ESCALA]

- Con las cajas de anclaje, se cambia la resolución a 416×416, sin embargo, ya que el modelo usa capas conv. y pooling, este puede cambiar su tamaño
- En lugar de fijar el tamaño, se cambia la red cada vez que se cumpla un número de iter. pequeño. Cada 10 batches, la red elige de forma aleatoria nuevas dimensiones de la imagen
- Ya que el modelo submuestrea por un factor de 32, se eligen los siguientes tam. multiplos de 32: $\{320, 352, \ldots, 608\}$. Entonces la red se redimensiona y se continua el entrenamiento
- Es por esto que la red puede predecir detecciones a diferentes resoluciones

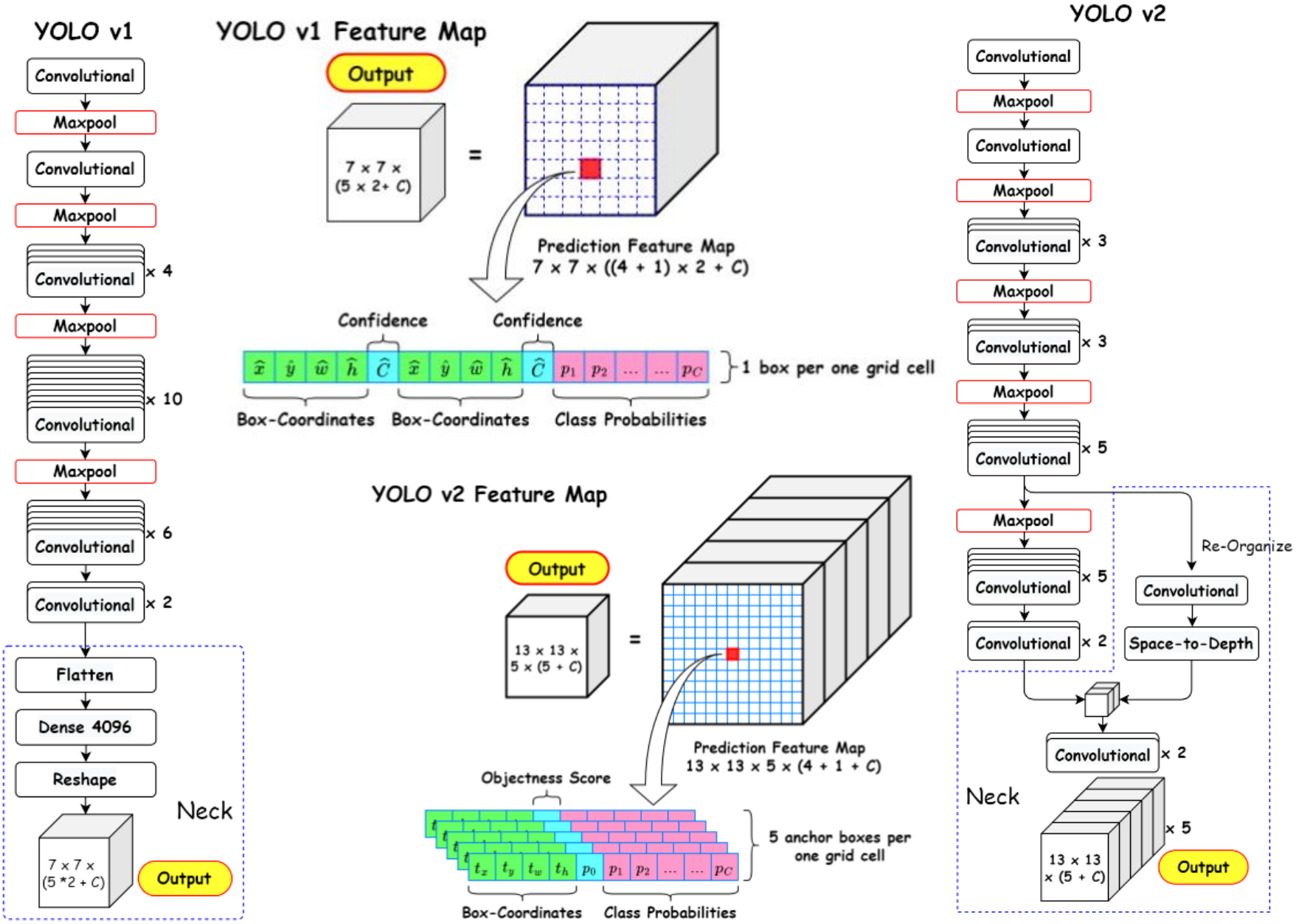


https://docs.google.com/presentation/d/14qBAiyhMOFl_wZW4dA1CkixgXwf0zKGbpw_0oHK8yEM/edit#slide=id.g23eff386b7_0_24

YOLO V2 [FASTER]

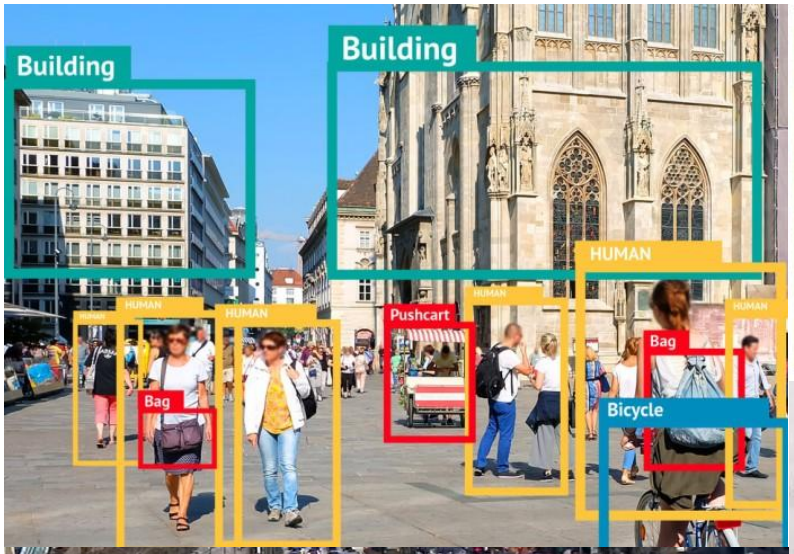
- Muchos *frameworks* de detección usan la VGG-16 como extractor de características base
- Aunque VGG-16 es una red de clasif. precisa, es muy compleja, ya que las capas convol. requieren 30.69 billones de operaciones en punto flotante para una simple pasada sobre una imagen de 224×224
- YOLO V2 usa una red personalizada con base en la arquit. de Googlenet. Esta red necesita 8.52 billones de operaciones para una pasada, sin embargo, su precisión es un poco peor que la VGG-16

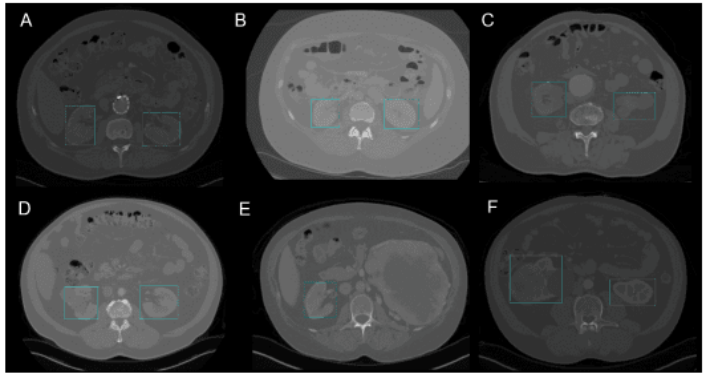
$$\text{ImageNet} \begin{cases} 88\% \ YOLO \\ 90\% \ VGG-16 \end{cases}$$

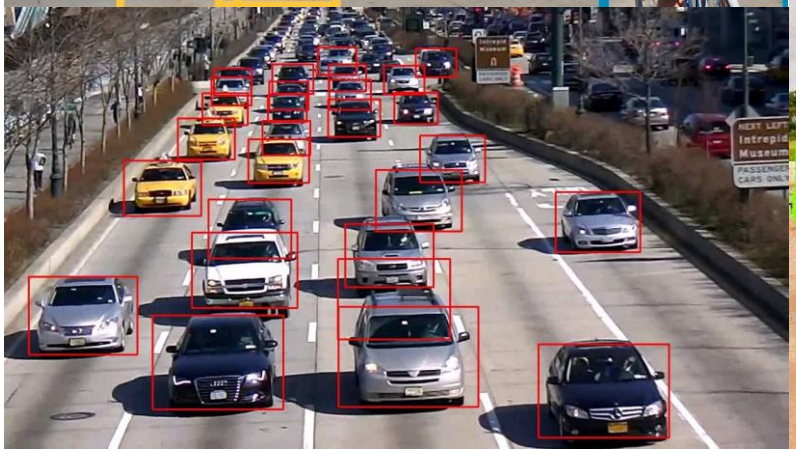


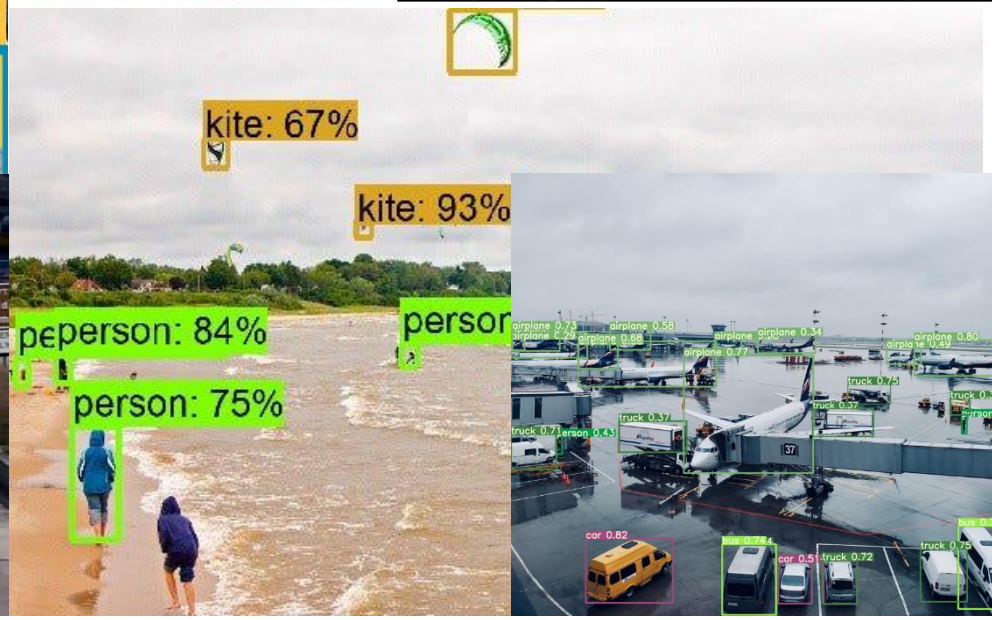
DLB, https://wikidocs.net/167699

APLICACIONES DE YOLO

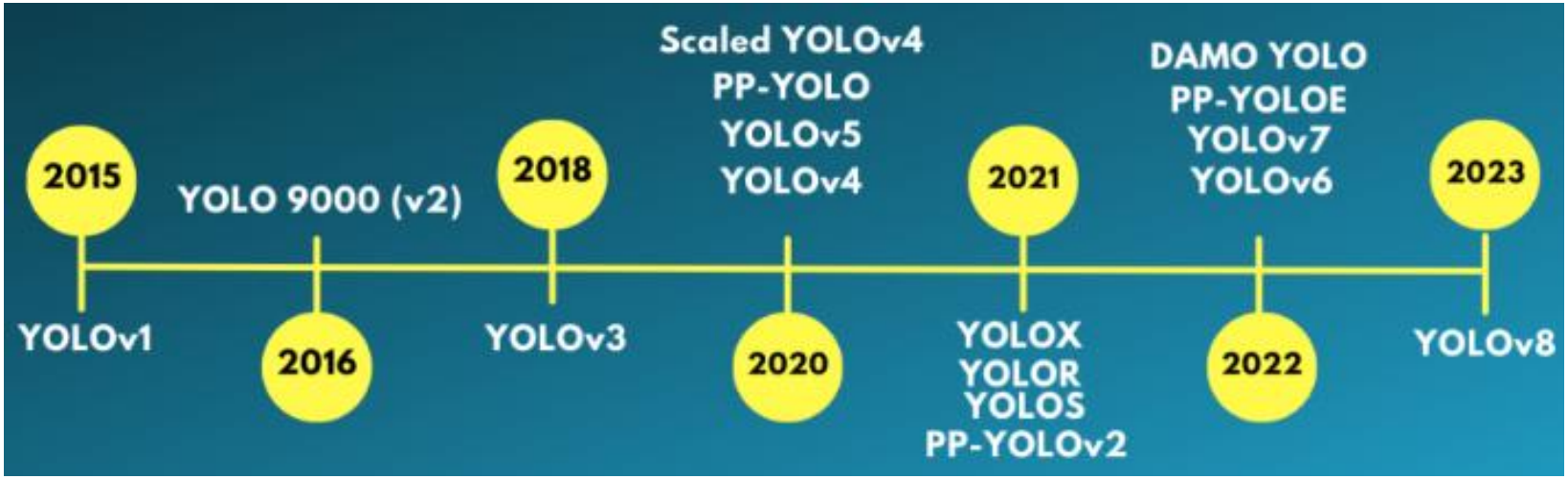








VERSIONES DE YOLO



https://learnopencv.com/ultralytics-yolov8/

Joseph Redmon

roboflow

Joseph Nelson

ultralytics

Glenn Jocher

GRACIAS POR SU ATENCIÓN

Francisco J. Hernández López

fcoj23@cimat.mx

WebPage:

www.cimat.mx/~fcoj23