

# YOLOv7

Trainable bag-of-freebies sets new  
state-of-the-art for real-time object  
detectors

**PRESENTATION - 1**

**Presented by :**

**RAKSHITH RAM C.A.**  
**SM22MTECH12003**

**Faculty Incharge :**

**Prof. C. KRISHNA MOHAN**

**Teaching Assistant :**

**AVEEN DAYAL**

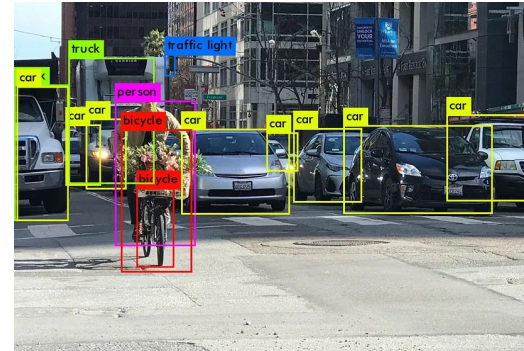


“ YOLOv7 : Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors.”

ArXiv (open access archive) , July 2022.



**Authors :** Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao



- ❑ Motivation
- ❑ Problem statement
- ❑ Introduction
- ❑ Existing methods
- ❑ Object detection procedure in YOLO
- ❑ Paper methodology
- ❑ Architecture
- ❑ Evaluation metrics
- ❑ Experiments and results
- ❑ Summary (based on paper)
- ❑ References



## Motivation

- The motivation for Yolov7 is to provide a fast and accurate object detection algorithm.
- This can be used in real-time applications such as autonomous driving, surveillance, and robotics.
- Yolov7 is a significant step forward in the development of efficient and accurate object detection algorithms.



## Problem statement

The problem statement for YOLOv7 (You Only Look Once version 7) is to further improve the speed and accuracy, while reducing inference cost of object detection in images, building on the success of the previous versions of YOLO specially YOLOv4.

1. Achieve state-of-the-art performance on object detection benchmarks, including high accuracy and low false positive rates.
2. Process images in real-time, making it suitable for applications such as autonomous driving and surveillance.
3. Be efficient in terms of computational resources, such as memory and processing power, making it accessible to a wider range of users and devices.

## Classification v/s Localization v/s Detection

Classification



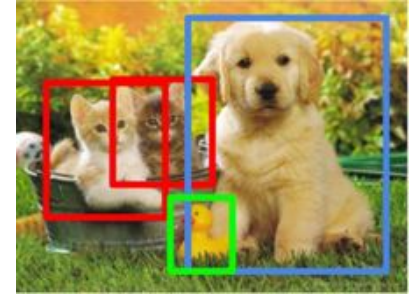
Cat

Localization



Cat

Detection

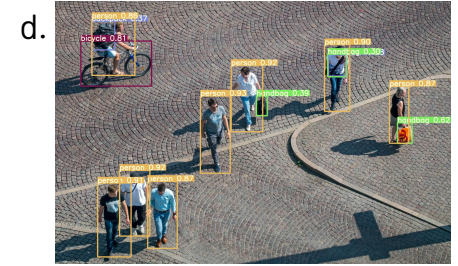
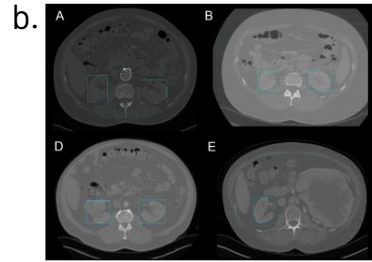
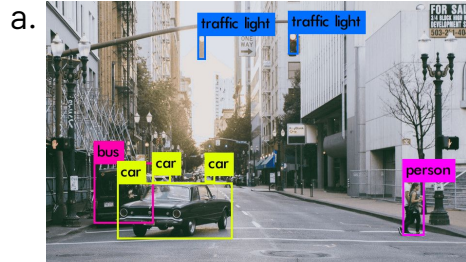


Cat, Dog, Duck



# Applications of object detection and YOLO

1. Self-driving cars
2. Traffic monitoring
3. Parking occupancy
4. Vehicle tracking
5. Healthcare
6. Agriculture
7. Animal detection in agriculture
8. PPE detection
9. Inventory management
10. Contactless checkout
11. Defect and anomaly detection
12. Video analytics
13. Security surveillance
14. Face detection



Applications in :-

a) Self-driving cars

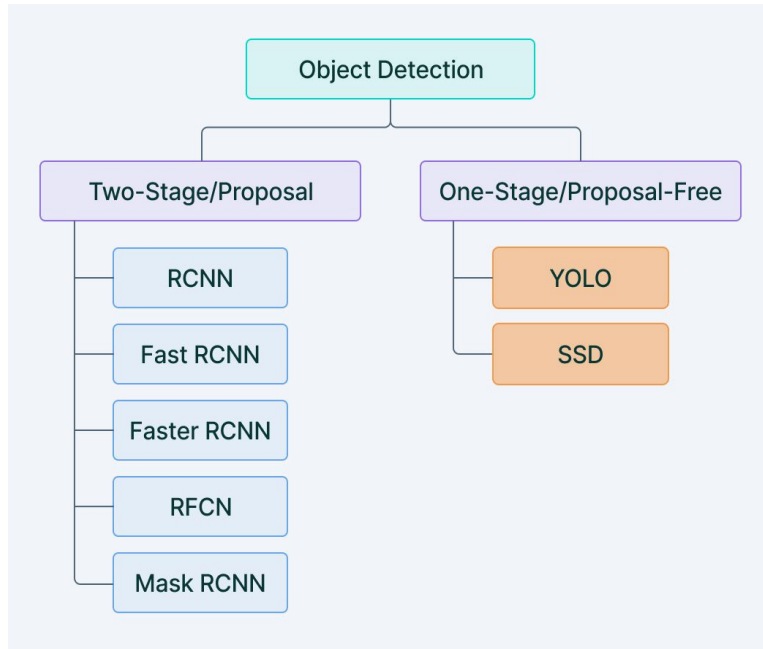
b) MRI scans

c) Agriculture

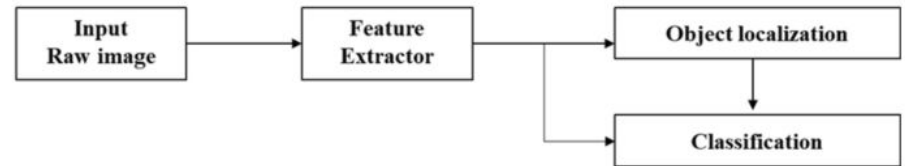
d) Security surveillance



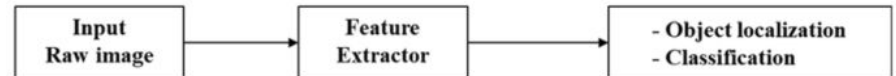
## One and Two stage object detection



Two-stage detectors :



One-stage detectors :





## What is YOLO ?

- You Only Look Once (YOLO) proposes using an end-to-end neural network that makes predictions of bounding boxes and class probabilities all at once.
- YOLOv7 was introduced to the YOLO family in July'22. According to the YOLOv7 paper, it is the fastest and most accurate real-time object detector to date.



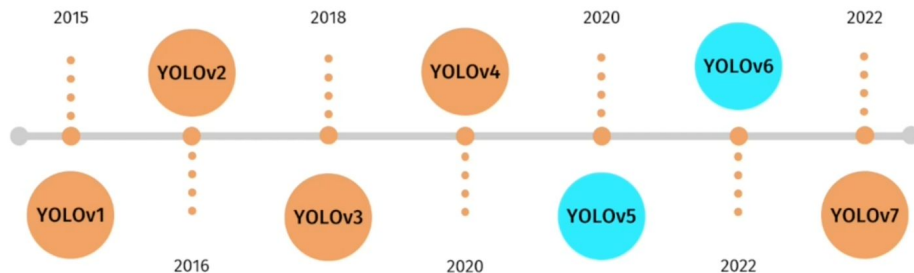
## Why is it called You Only Look Once ?

- It refers to the fact that YOLO processes an entire image in a single forward pass through a neural network, as opposed to the traditional object detection methods that require multiple passes through the network.
- Algorithms like Faster RCNN work by detecting possible regions of interest using the Region Proposal Network and then performing recognition on those regions separately.

## Why use YOLO ?

- YOLO models are used because they are small, nimble, and trainable on a single GPU. This is the opposite of the giant transformer architectures coming out of the leading labs in big tech which, while effective, are more difficult to run on consumer hardware.
- One of the main advantages of YOLO is its fast inference speed, which allows it to process images in real time. It's well-suited for applications such as video surveillance, self-driving cars, and augmented reality.
- YOLO is more generalized. It outperforms other methods when generalizing from natural images to other domains like artwork.

## Versions of YOLO ?





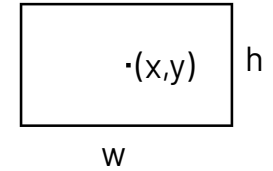
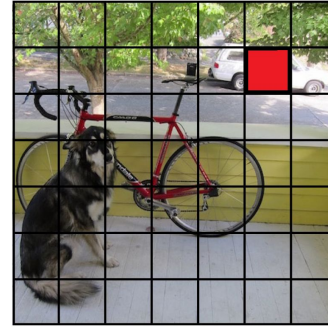
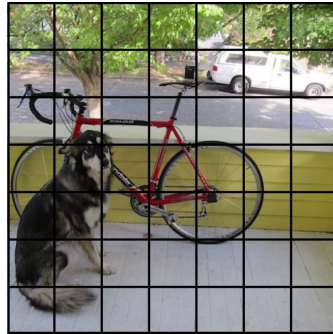
## Object detection procedure in YOLO



Input image

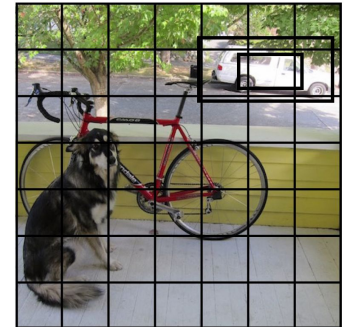
The image is split  
into  $\mathbf{S} * \mathbf{S}$  grid

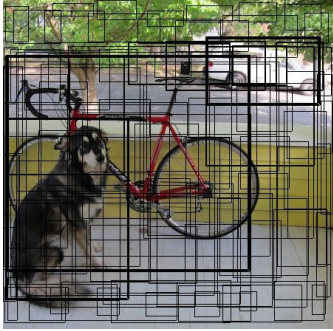
here,  $\mathbf{S} = 7$



here,  $\mathbf{B} = 2$

Each cell of the  
grid predicts  $\mathbf{B}$   
boxes  $(\mathbf{x}, \mathbf{y}, \mathbf{w}, \mathbf{h})$  and  
the confidences of  
each of the box  $\mathbf{P}$   
(object)

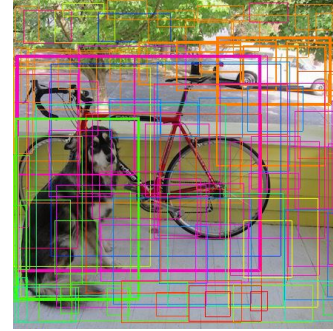
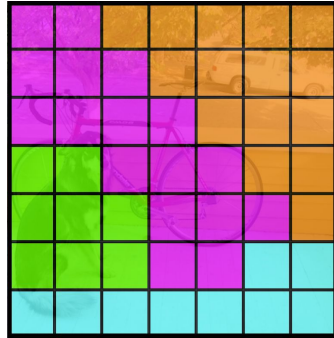




Each of the cell also predicts a class probability.

- Car
- Bicycle
- Dog

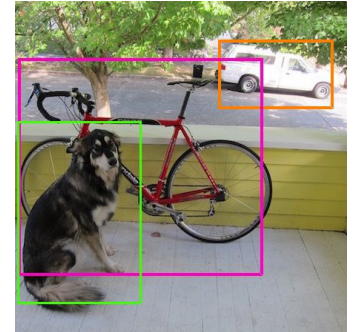
All of the cells predict boxes and confidences.



The output bounding boxes and the class predictions are combined.

$$\begin{aligned} P(\text{class} | \text{object}) \\ \times \\ P(\text{object}) \\ = \\ P(\text{class}) \end{aligned}$$

Finally, **NMS** is performed to obtain the best bounding boxes.



Each cell predicts :

For each bounding box :

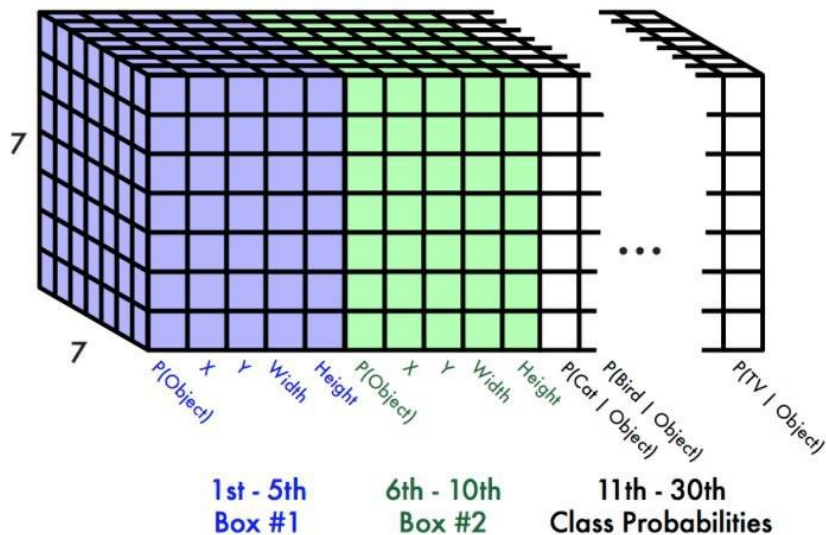
- 4 co-ordinates (x, y, w, h)
- 1 confidence value
- Some number of class probabilities

$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

Example :

- 7 \* 7 grid
- 2 bounding boxes
- 20 classes

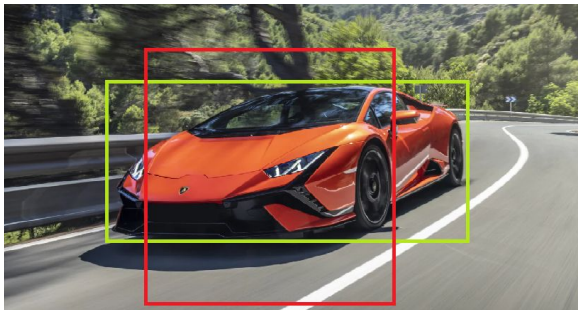
$$7 * 7 * (2 * 5 + 20) = 1470 \text{ outputs}$$



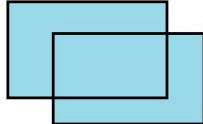
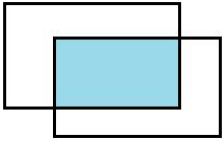


## Intersection over Union (IoU)

Intersection over Union is a popular metric to measure localization accuracy and calculate localization errors in object detection models.



- Predicted bounding box
- Ground truth

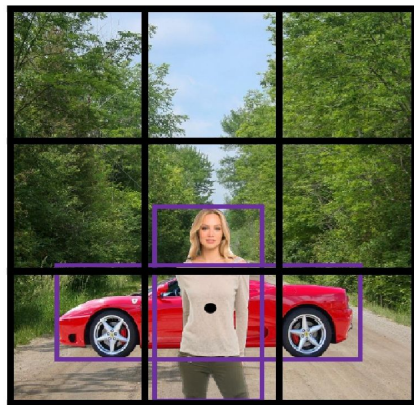
$$\text{IoU} = \frac{\text{Area of Intersection}}{\text{Area of Union}} = \frac{\text{Diagram 1}}{\text{Diagram 2}}$$


Higher the IoU value, closer the bounding box is to the ground truth.

Generally a threshold (example :  $\text{IoU} > 0.5$ ) is set.



## Anchor boxes



Anchor box 1



Anchor box 2



$$y = \begin{bmatrix} p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \\ p_c \\ b_x \\ b_y \\ b_h \\ b_w \\ c_1 \\ c_2 \\ c_3 \end{bmatrix}$$

- The concept of the Anchor box is to pre-define multiple Anchor boxes of different shapes, and then combine the prediction results with multiple Anchor boxes.
- For several Anchor boxes, the vector of the output  $y$  will be stretched several times for correlation.

→ YOLOv5 and YOLOv7 use **9** anchor boxes by default.



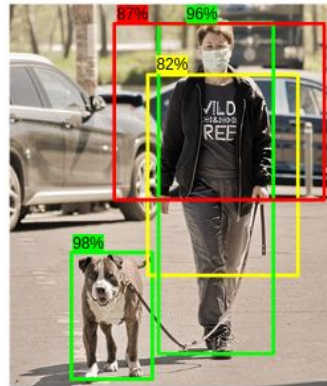
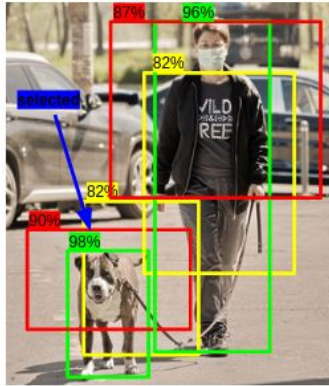


## Non-maximum suppression (NMS)

The purpose of non-max suppression is to select the best bounding box for an object and reject or “suppress” all other bounding boxes.

The NMS takes two things into account :

1. The objectiveness score is given by the model.
2. The overlap or IOU of the bounding boxes.



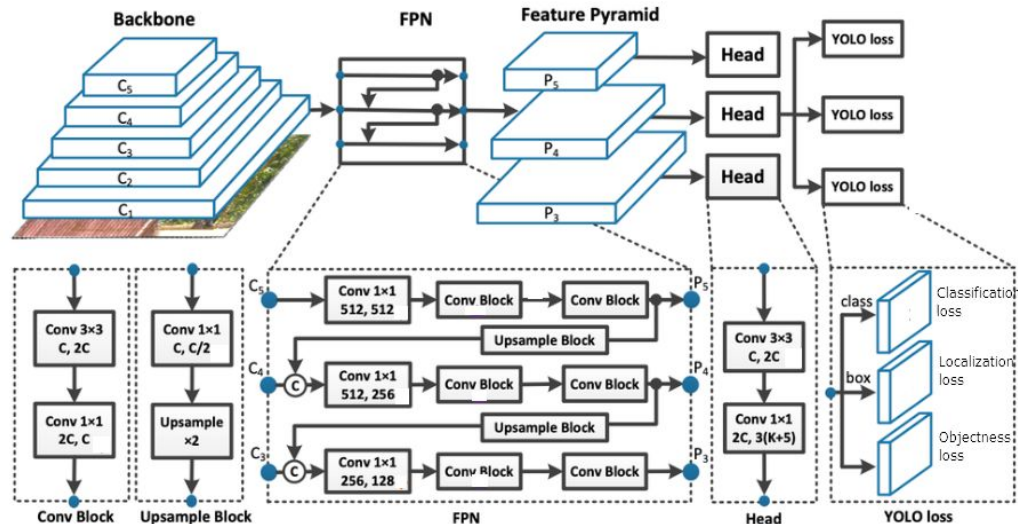




## YOLOv7 - Architecture

The YOLO framework has three main components :

- The **Backbone** mainly extracts essential features of an image and feeds them to the Neck.
- The **Neck** collects feature maps extracted by the Backbone and creates feature pyramids.
- Finally, the **Head** consists of output layers that have final detections.



## Loss functions in YOLOv7

The YOLOv7 model uses three main loss functions to optimize the model for object detection:

1. **Objectness loss:** This loss function is used to train the model to predict whether an object is present in a given bounding box. The objectness loss is computed using a binary cross-entropy loss between the predicted objectness score and the ground truth objectness score.
2. **Classification loss:** The classification loss is computed using a multi-class cross-entropy loss between the predicted class probabilities and the ground truth class labels.
3. **Localization loss:** The localization loss is computed using the mean squared error (MSE) between the predicted bounding box coordinates and the ground truth bounding box coordinates.



## YOLOv7 architecture, What's new ?

### **E-ELAN (Extended Efficient Layer Aggregation Network)**

- ELA involves aggregating feature maps from different convolutional layers to improve the quality of object detections. This is achieved by creating shortcuts between different convolutional layers, allowing features from earlier layers to be combined with features from later layers. This approach allows for a more comprehensive understanding of the scene and can lead to more accurate object detection predictions.
- EELA is an extension of ELA that is used in YOLOv7. EELA further improves on ELA by incorporating additional convolutional layers. This is achieved by using a multi-scale feature aggregation approach, where feature maps from multiple scales are combined to form a single, more comprehensive feature map.

## Model scaling for concatenation-based models

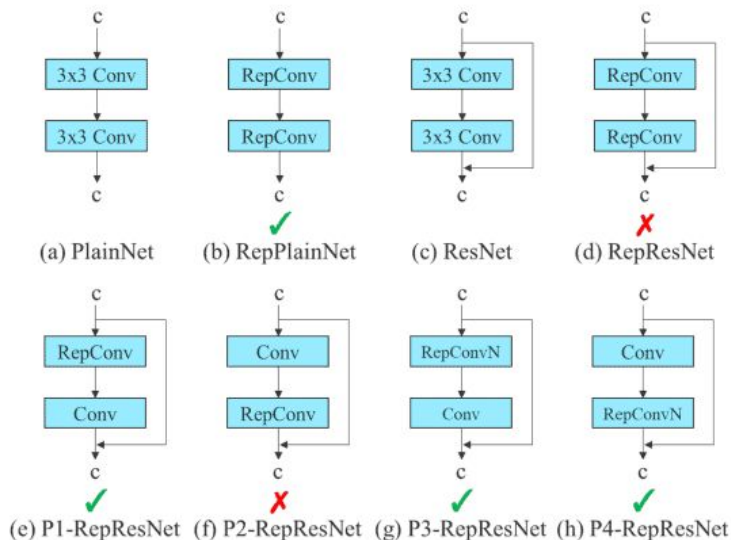
- Different applications require different models. While some need highly accurate models, some prioritize speed. Model scaling is performed to suit these requirements and make it fit in various computing devices.

## Coarse for Auxiliary and Fine for Lead Loss

- When it comes to loss functions, different parts of the network can have different losses depending on their function.
- A "coarse" loss function is used for the auxiliary loss, which focuses more on low-level features and spatial information.
- A "fine" loss function for the lead loss, which focuses more on high-level semantic information and object detection accuracy.

## Planned Re-parameterized Convolution

- Re-parameterization is a technique used after training to improve the model. It increases the training time but improves the inference results.
- The  $3 \times 3$  convolution layer of the E-ELAN computational block is replaced with the **RepConv** layer. Experiments we carried out by switching or replacing the positions of RepConv,  $3 \times 3$  Conv, and Identity connection. The residual bypass arrow shown is an identity connection. It is nothing but a  $1 \times 1$  convolutional layer. We can see the configurations that work and the ones that do not.



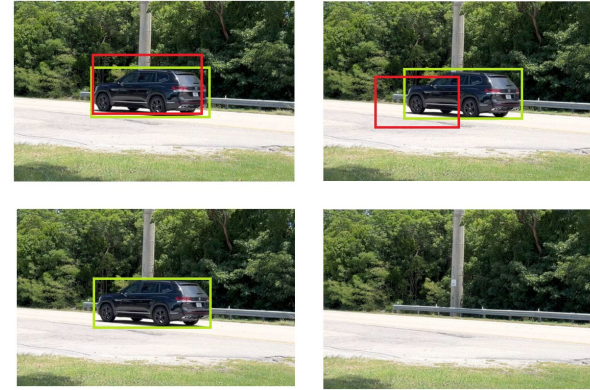


## Mean Average Precision (mAP)

Mean Average Precision (mAP) is the current benchmark metric used by the computer vision research community to evaluate the robustness of object detection models

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Confusion matrix



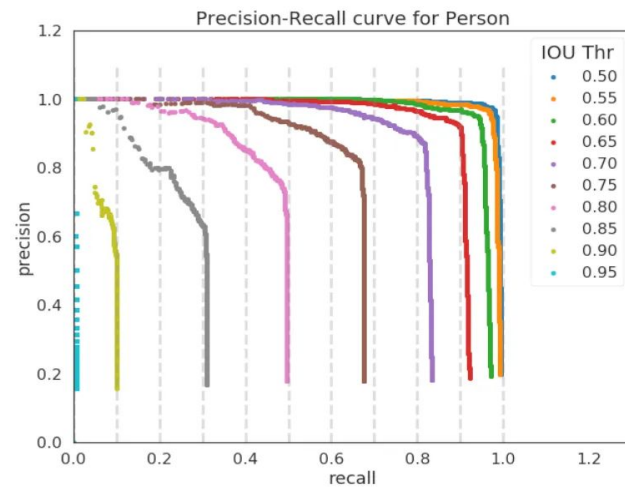
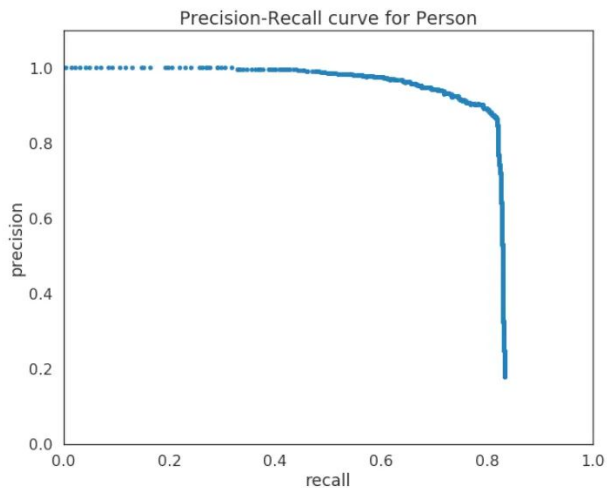
**recision** : It measures how well you can find true positives out of all positive predictions.  
(If your model predicts how often does it predicts correctly?)

**Recall** : It measures how well you can find true positives out of all predictions.  
(Has your model predicted every time that it should have predicted?)

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

**Precision-Recall curve** is obtained by plotting the model's precision and recall values as a function of the model's confidence score threshold.



The mean Average Precision or **mAP** score is calculated by taking the mean **AP** over **all classes** and / or over **all IoU thresholds**, depending on the competition.

For example:

- PASCAL VOC2007 challenge only 1 IoU threshold was considered: 0.5 so the mAP was averaged over all 20 object classes.
- For the COCO 2017 challenge, the mAP was averaged over all 80 object categories and all 10 IoU thresholds.

**AP** = Area under the Precision-Recall curve

$$AP = \int_0^1 p(r) dr$$

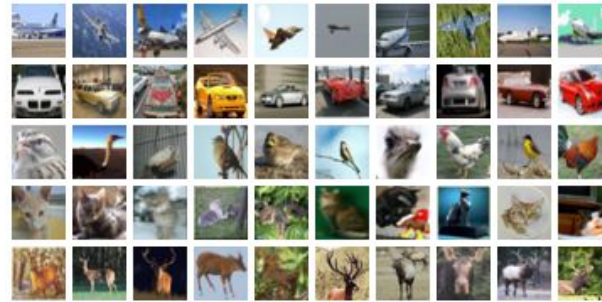
$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$$

$AP_k$  = the average precision of class k  
 $n$  = number of classes





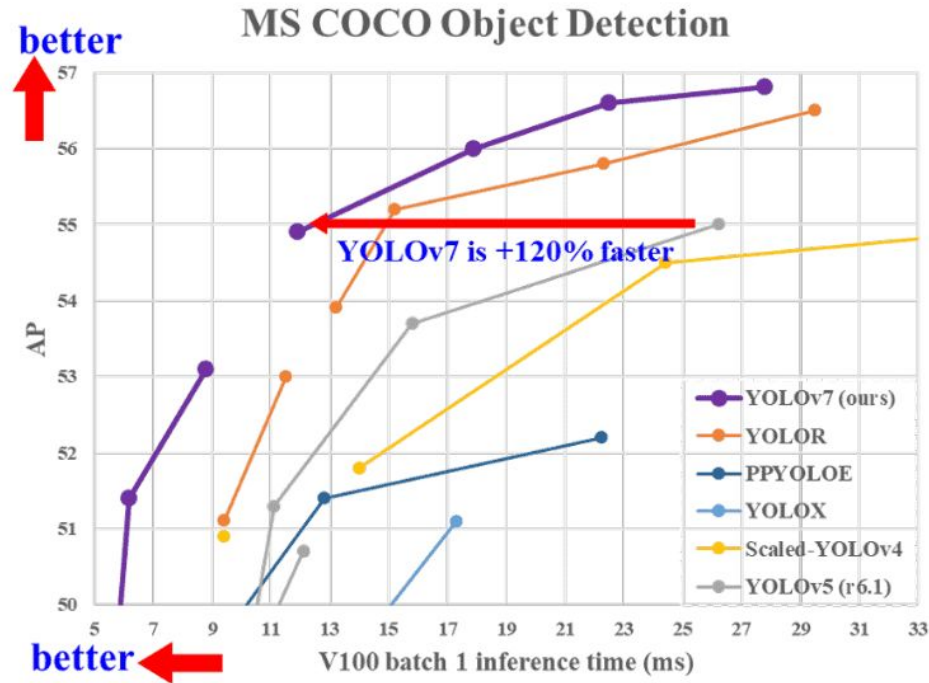
## MS-COCO dataset



- MS COCO (Microsoft Common Objects in Context) is a large-scale image recognition dataset that was introduced in 2014 by Microsoft Research.
- It contains 330,000 images with more than 2.5 million object instances labeled with 80 different object categories.
- The MS COCO dataset is notable for its high-quality annotations,
- The dataset being widely used for both academic research and industrial applications.



## Experimentation and Results



All the YOLOv7 models surpass the previous object detectors in speed and accuracy in the range of 5 FPS to 160 FPS. The following figure gives a pretty good idea about the Average Precision(AP) and speed of the YOLOv7 models compared to the others.

Model	#Param.	FLOPs	Size	AP <sup>val</sup>	AP <sup>val</sup> <sub>50</sub>	AP <sup>val</sup> <sub>75</sub>	AP <sup>val</sup> <sub>S</sub>	AP <sup>val</sup> <sub>M</sub>	AP <sup>val</sup> <sub>L</sub>
YOLOv4 [3]	64.4M	142.8G	640	49.7%	68.2%	54.3%	32.9%	54.8%	63.7%
YOLOR-u5 (r6.1) [81]	46.5M	109.1G	640	50.2%	68.7%	54.6%	33.2%	55.5%	63.7%
YOLOv4-CSP [79]	52.9M	120.4G	640	50.3%	68.6%	54.9%	34.2%	55.6%	65.1%
YOLOR-CSP [81]	52.9M	120.4G	640	50.8%	69.5%	55.3%	33.7%	56.0%	65.4%
YOLOv7	36.9M	104.7G	640	<b>51.2%</b>	<b>69.7%</b>	<b>55.5%</b>	<b>35.2%</b>	<b>56.0%</b>	<b>66.7%</b>
improvement	-43%	-15%	-	+0.4	+0.2	+0.2	+1.5	=	+1.3
YOLOR-CSP-X [81]	96.9M	226.8G	640	52.7%	<b>71.3%</b>	57.4%	36.3%	57.5%	68.3%
YOLOv7-X	71.3M	189.9G	640	<b>52.9%</b>	71.1%	<b>57.5%</b>	<b>36.9%</b>	<b>57.7%</b>	<b>68.6%</b>
improvement	-36%	-19%	-	+0.2	-0.2	+0.1	+0.6	+0.2	+0.3
YOLOv4-tiny [79]	6.1	6.9	416	24.9%	42.1%	25.7%	8.7%	28.4%	39.2%
YOLOv7-tiny	6.2	5.8	416	<b>35.2%</b>	<b>52.8%</b>	<b>37.3%</b>	<b>15.7%</b>	<b>38.0%</b>	<b>53.4%</b>
improvement	+2%	-19%	-	+10.3	+10.7	+11.6	+7.0	+9.6	+14.2
YOLOv4-tiny-3l [79]	8.7	5.2	320	30.8%	47.3%	32.2%	<b>10.9%</b>	31.9%	51.5%
YOLOv7-tiny	6.2	3.5	320	<b>30.8%</b>	<b>47.3%</b>	<b>32.2%</b>	10.0%	<b>31.9%</b>	<b>52.2%</b>
improvement	-39%	-49%	-	=	=	=	-0.9	=	+0.7
YOLOR-E6 [81]	115.8M	683.2G	1280	55.7%	73.2%	60.7%	40.1%	<b>60.4%</b>	69.2%
YOLOv7-E6	97.2M	515.2G	1280	<b>55.9%</b>	<b>73.5%</b>	<b>61.1%</b>	<b>40.6%</b>	60.3%	<b>70.0%</b>
improvement	-19%	-33%	-	+0.2	+0.3	+0.4	+0.5	-0.1	+0.8
YOLOR-D6 [81]	151.7M	935.6G	1280	56.1%	73.9%	61.2%	<b>42.4%</b>	60.5%	69.9%
YOLOv7-D6	154.7M	806.8G	1280	56.3%	73.8%	61.4%	41.3%	60.6%	70.1%
YOLOv7-E6E	151.7M	843.2G	1280	<b>56.8%</b>	<b>74.4%</b>	<b>62.1%</b>	40.8%	<b>62.1%</b>	<b>70.6%</b>
improvement	=	-11%	-	+0.7	+0.5	+0.9	-1.6	+1.6	+0.7

Comparison of  
baseline object  
detectors.

Comparison of  
state-of-the-art  
real-time object  
detectors.

Model	#Param.	FLOPs	Size	FPS	$AP^{test} / AP^{val}$	$AP_{50}^{test}$	$AP_{75}^{test}$	$AP_S^{test}$	$AP_M^{test}$	$AP_L^{test}$
YOLOX-S [21]	9.0M	26.8G	640	102	40.5% / 40.5%	-	-	-	-	-
YOLOX-M [21]	25.3M	73.8G	640	81	47.2% / 46.9%	-	-	-	-	-
YOLOX-L [21]	54.2M	155.6G	640	69	50.1% / 49.7%	-	-	-	-	-
YOLOX-X [21]	99.1M	281.9G	640	58	51.5% / 51.1%	-	-	-	-	-
PPYOLOE-S [85]	7.9M	17.4G	640	208	43.1% / 42.7%	60.5%	46.6%	23.2%	46.4%	56.9%
PPYOLOE-M [85]	23.4M	49.9G	640	123	48.9% / 48.6%	66.5%	53.0%	28.6%	52.9%	63.8%
PPYOLOE-L [85]	52.2M	110.1G	640	78	51.4% / 50.9%	68.9%	55.6%	31.4%	55.3%	66.1%
PPYOLOE-X [85]	98.4M	206.6G	640	45	52.2% / 51.9%	69.9%	56.5%	33.3%	56.3%	66.4%
YOLOv5-N (r6.1) [23]	1.9M	4.5G	640	159	- / 28.0%	-	-	-	-	-
YOLOv5-S (r6.1) [23]	7.2M	16.5G	640	156	- / 37.4%	-	-	-	-	-
YOLOv5-M (r6.1) [23]	21.2M	49.0G	640	122	- / 45.4%	-	-	-	-	-
YOLOv5-L (r6.1) [23]	46.5M	109.1G	640	99	- / 49.0%	-	-	-	-	-
YOLOv5-X (r6.1) [23]	86.7M	205.7G	640	83	- / 50.7%	-	-	-	-	-
YOLOR-CSP [81]	52.9M	120.4G	640	106	51.1% / 50.8%	69.6%	55.7%	31.7%	55.3%	64.7%
YOLOR-CSP-X [81]	96.9M	226.8G	640	87	53.0% / 52.7%	71.4%	57.9%	33.7%	57.1%	66.8%
YOLOv7-tiny-SiLU	6.2M	13.8G	640	286	38.7% / 38.7%	56.7%	41.7%	18.8%	42.4%	51.9%
YOLOv7	36.9M	104.7G	640	161	51.4% / 51.2%	69.7%	55.9%	31.8%	55.5%	65.0%
YOLOv7-X	71.3M	189.9G	640	114	53.1% / 52.9%	71.2%	57.8%	33.8%	57.1%	67.4%
YOLOv5-N6 (r6.1) [23]	3.2M	18.4G	1280	123	- / 36.0%	-	-	-	-	-
YOLOv5-S6 (r6.1) [23]	12.6M	67.2G	1280	122	- / 44.8%	-	-	-	-	-
YOLOv5-M6 (r6.1) [23]	35.7M	200.0G	1280	90	- / 51.3%	-	-	-	-	-
YOLOv5-L6 (r6.1) [23]	76.8M	445.6G	1280	63	- / 53.7%	-	-	-	-	-
YOLOv5-X6 (r6.1) [23]	140.7M	839.2G	1280	38	- / 55.0%	-	-	-	-	-
YOLOR-P6 [81]	37.2M	325.6G	1280	76	53.9% / 53.5%	71.4%	58.9%	36.1%	57.7%	65.6%
YOLOR-W6 [81]	79.8G	453.2G	1280	66	55.2% / 54.8%	72.7%	60.5%	37.7%	59.1%	67.1%
YOLOR-E6 [81]	115.8M	683.2G	1280	45	55.8% / 55.7%	73.4%	61.1%	38.4%	59.7%	67.7%
YOLOR-D6 [81]	151.7M	935.6G	1280	34	56.5% / 56.1%	74.1%	61.9%	38.9%	60.4%	68.7%
YOLOv7-W6	70.4M	360.0G	1280	84	54.9% / 54.6%	72.6%	60.1%	37.3%	58.7%	67.1%
YOLOv7-E6	97.2M	515.2G	1280	56	56.0% / 55.9%	73.5%	61.2%	38.0%	59.9%	68.4%
YOLOv7-D6	154.7M	806.8G	1280	44	56.6% / 56.3%	74.0%	61.8%	38.8%	60.1%	69.5%
YOLOv7-E6E	151.7M	843.2G	1280	36	56.8% / 56.8%	74.4%	62.1%	39.3%	60.5%	69.0%

Model	Parameters (million)	FPS	AP test (%)
YOLOv7-Tiny	6.2	286	38.7
YOLOv7	36.9	161	51.4
YOLOv7-X	71.3	114	53.1
YOLOv7-W6	70.04	84	54.9
<b>YOLOv7-E6</b>	<b>97.2</b>	<b>56</b>	<b>56.0</b>
YOLOv7-D6	154.7	44	56.6
YOLOv7-E6E	151.7	36	56.8

- YOLOv7-Tiny, YOLOv7, and YOLOv7-W6 are meant for edge GPU, normal (consumer) GPU, and cloud GPU, respectively.
- YOLOv7-E6 and YOLOv7-D6, and YOLOv7-E6E are also meant for high-end cloud GPUs only.

Nonetheless, all of the YOLOv7 models run at more than 30 FPS on the Tesla V100 GPU, which is more than real-time FPS.



## Summary

- YOLOv7 is the new state-of-the-art real-time object detection model which can be used for different industrial applications.
- YOLOv7 surpasses all known object detectors in both speed and accuracy in the range from 5 FPS to 160 FPS and has the highest accuracy 56.8% AP among all known real-time object detectors with 30 FPS or higher.
- In short, the new reforms presented in this paper namely,
  - Extended Efficient Layer Aggregation Network (E-ELAN)
  - Model Scaling for Concatenation based Models
  - Planned re-parameterized convolution
  - Coarse for auxiliary and fine for lead loss



## Future work

- Implementing the YOLOv7 algorithm on an autonomous vehicle after training it on a custom data set to detect features like potholes, speedbumps, pedestrians and other vehicles.
- Augmenting the training data i.e. adding more training data with various lighting conditions, viewpoints, and object sizes can improve the accuracy of YOLOv7.
- Fine-tuning the hyperparameters i.e. the learning rate, batch size, and others which can improve the performance of YOLOv7, specifically for autonomous vehicle applications.





- [1] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2016
- [2] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2017
- [3] Joseph Redmon and Ali Farhadi. YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018.
- [4] Alexey Bochkovskiy, Chien-Yao Wang, and Hong- Yuan Mark Liao. YOLOv4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:2004.10934, 2020.
- [5] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc V Le. NAS-FPN: Learning scalable feature pyramid architecture for object detection. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019.
- [6] Jocher Glenn. YOLOv5 release v6.1. <https://github.com/ultralytics/yolov5/releases/tag/v6.1>, 2022.
- [7] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019.



- [8] Chien-Yao Wang, Alexey Bochkovskiy, and Hong- Yuan Mark Liao. Scaled-YOLOv4: Scaling cross stage partial network. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2021.
- [9] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, et al. PPYOLOE: An evolved version of YOLO. arXiv preprint arXiv:2203.16250, 2022.
- [10] Xianzhi Du, Barret Zoph, Wei-Chih Hung, and Tsung-Yi Lin. Simple training strategies and model scaling for object detection. arXiv preprint arXiv:2107.00057, 2021



# Thank you!

