Understanding DNN-Based Object Detectors

Azhar Quddus, PhD.
Senior Computer Vision Engineer
Au-Zone Technologies Inc.

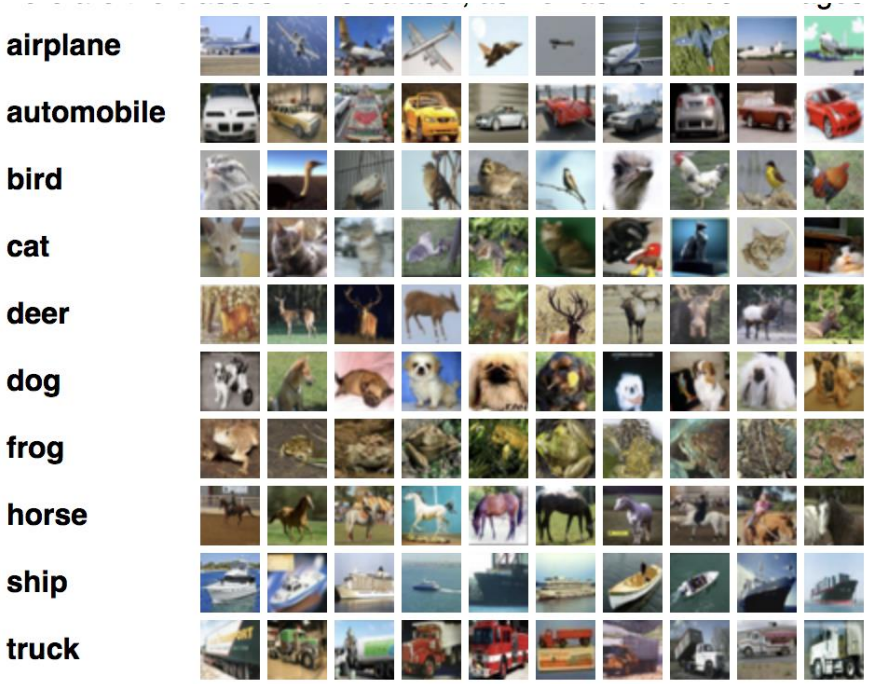
Presentation Overview

1. Problem Definition
2. Early Deep-Learning Based Approaches
   - R-CNN, Fast-RCNN, Faster-RCNN
3. Recent Advancements
   - Yolo, SSD, RetinaNet, CenterNet
4. Performance Metrics
   - Reported in papers
5. Performance Comparisons
6. Choosing an Object Detector

Problem Definition

Problem Definition

Image Classification (global- easier)



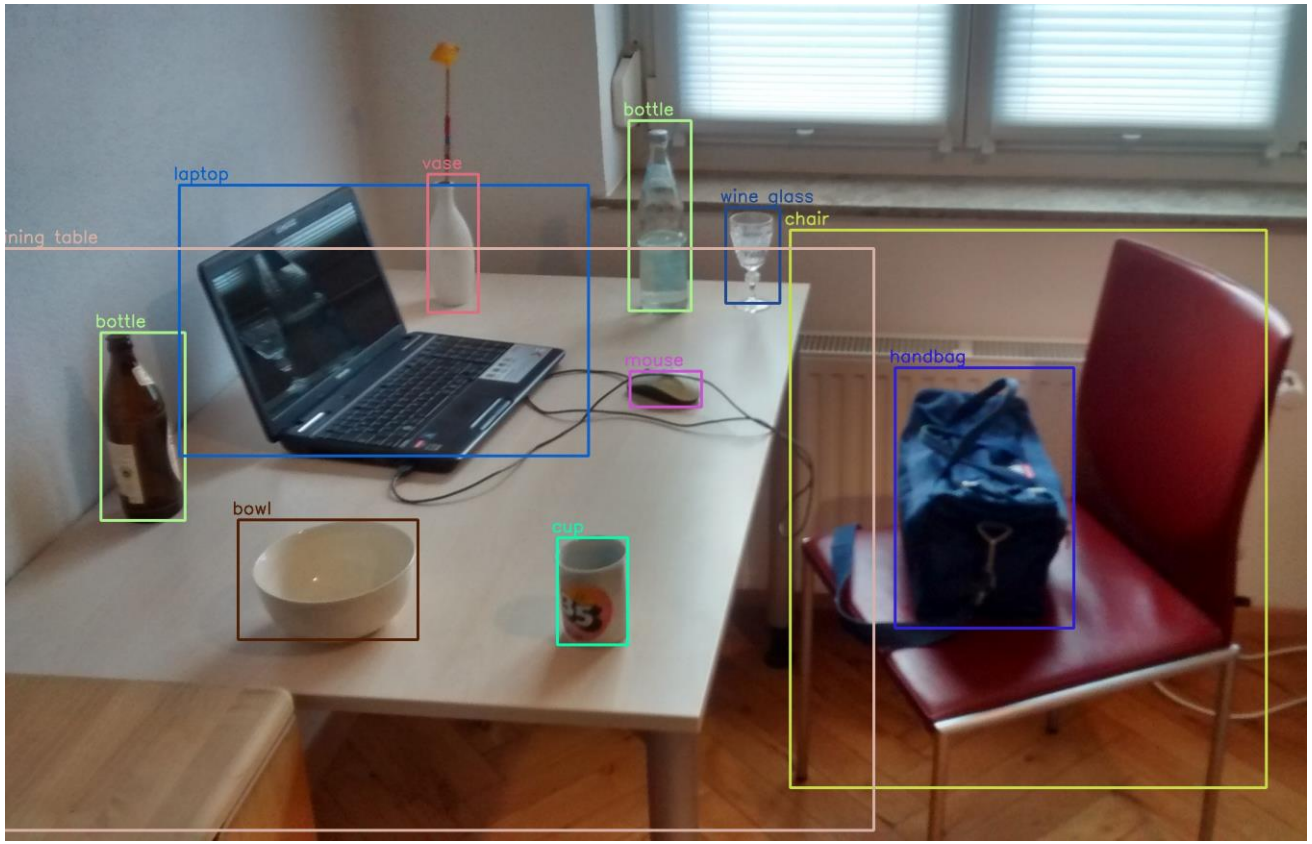
Ref: Example of Photographs of Objects From the CIFAR-10 Dataset

Object Segmentation (detailed- tougher)

Ref: https://medium.com/syncedreview/facebook-pointrend-rendering-image-segmentation-f3936d50e7f1

Problem Definition

Object Detection (Compromise):

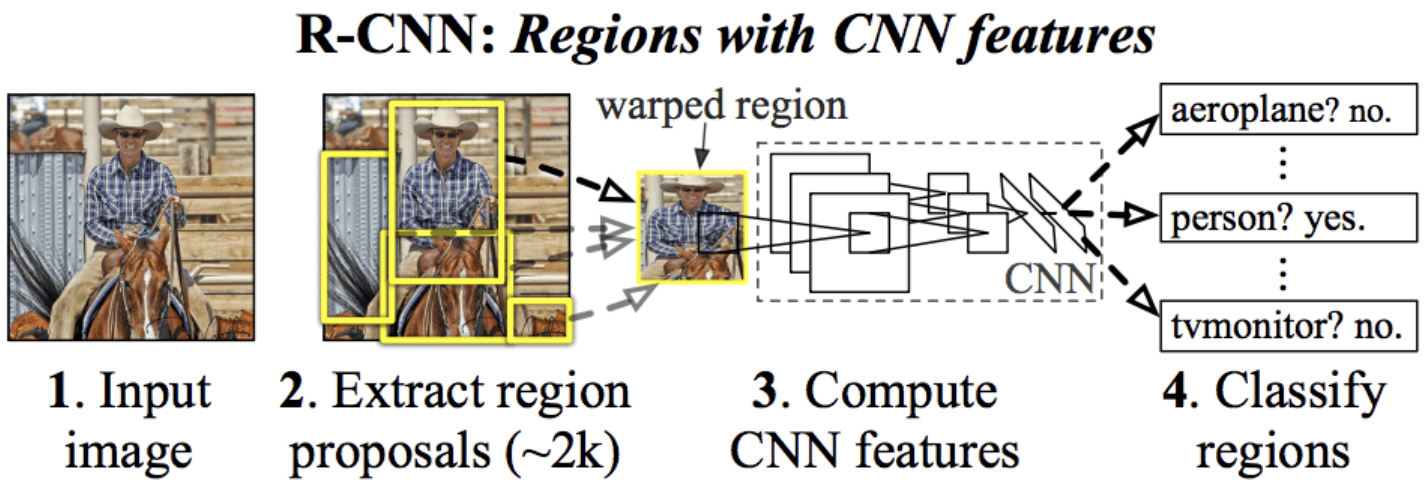


Ref: By (MTheiler) - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=75843378

1. Localize the object
   - Bounding box location
2. Estimate the size
   - Bounding box dimensions
3. Classify the object
   - Object label

Early Deep-Learning Based Approaches

Early Deep-Learning Based Approaches (R-CNN) 2014



"Rich feature hierarchies for accurate object detection and semantic segmentation"

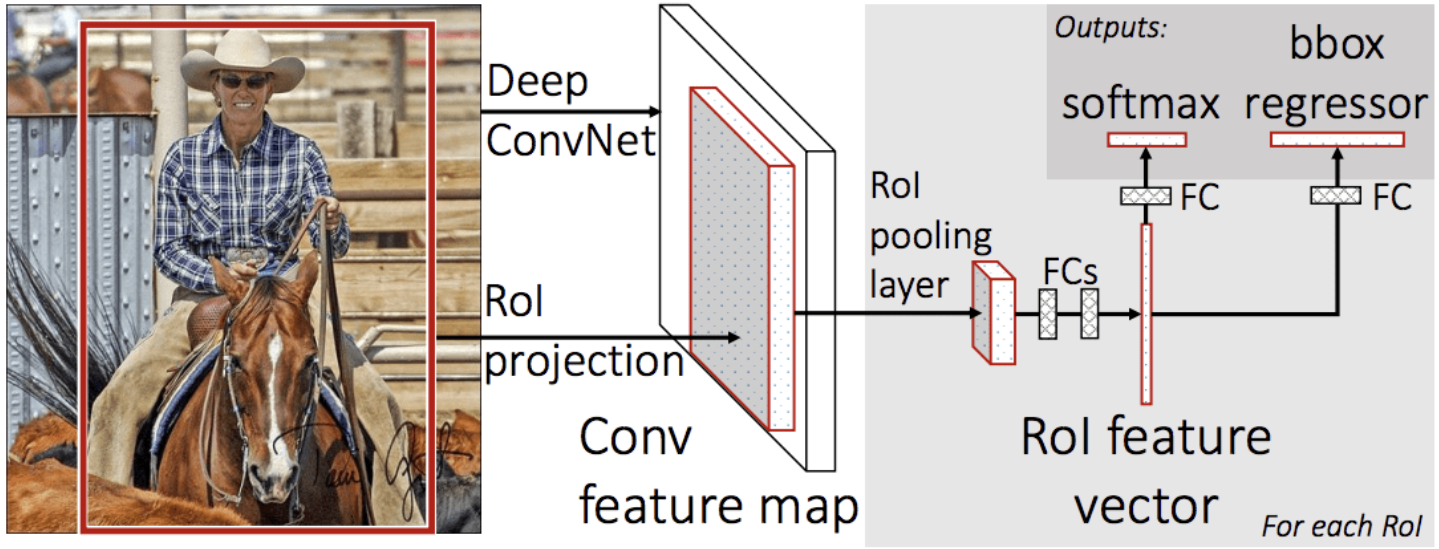
Issues:

1. Training is a multi-stage pipeline. Involves the preparation and operation of three separate models.
2. Training a deep CNN on so many region proposals per image is very slow.
3. Object detection is slow. Making predictions using a deep CNN on so many region proposals (~2000) is very slow.

Region Proposal: Generate and extract category independent region proposals, e.g. candidate bounding boxes. (Selective search)

Feature Extractor: Extract feature from each candidate region, e.g. using a deep CNN
Classifier. Classify features as one of the known class, e.g. linear SVM classifier model.

Early Deep-Learning Based Approaches (Fast R-CNN) 2015



Issue:
The model is significantly faster to train and to make predictions, yet still requires a set of candidate regions to be proposed along with each input image.

Fast R-CNN, ICCV 2015.

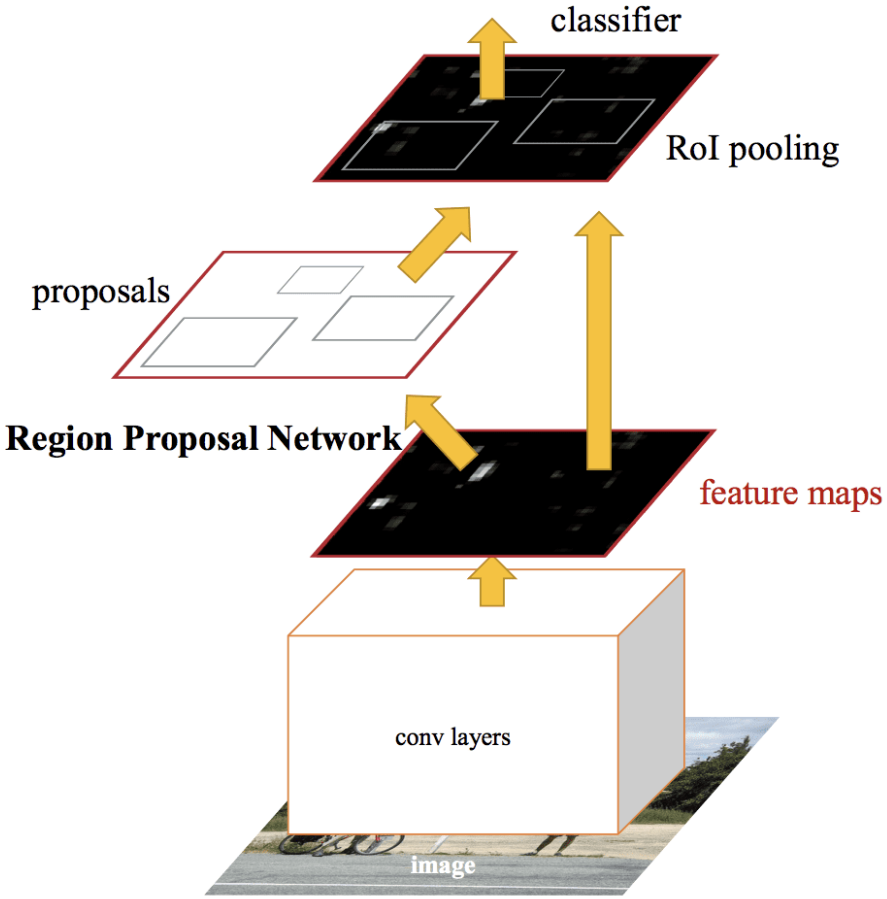
Region Proposal: Generate and extract category independent region proposals, e.g. candidate bounding boxes. (Selective search)

Classification and Bounding Box Combined: A single model instead of a pipeline to learn and output regions and classifications directly.

Early Deep-Learning Based Approaches (Faster R-CNN) 2016

Region Proposal Network: Convolutional neural network for proposing regions and the type of object to consider in the region.

Fast R-CNN: Convolutional neural network for extracting features from the proposed regions and outputting the bounding box and class labels.

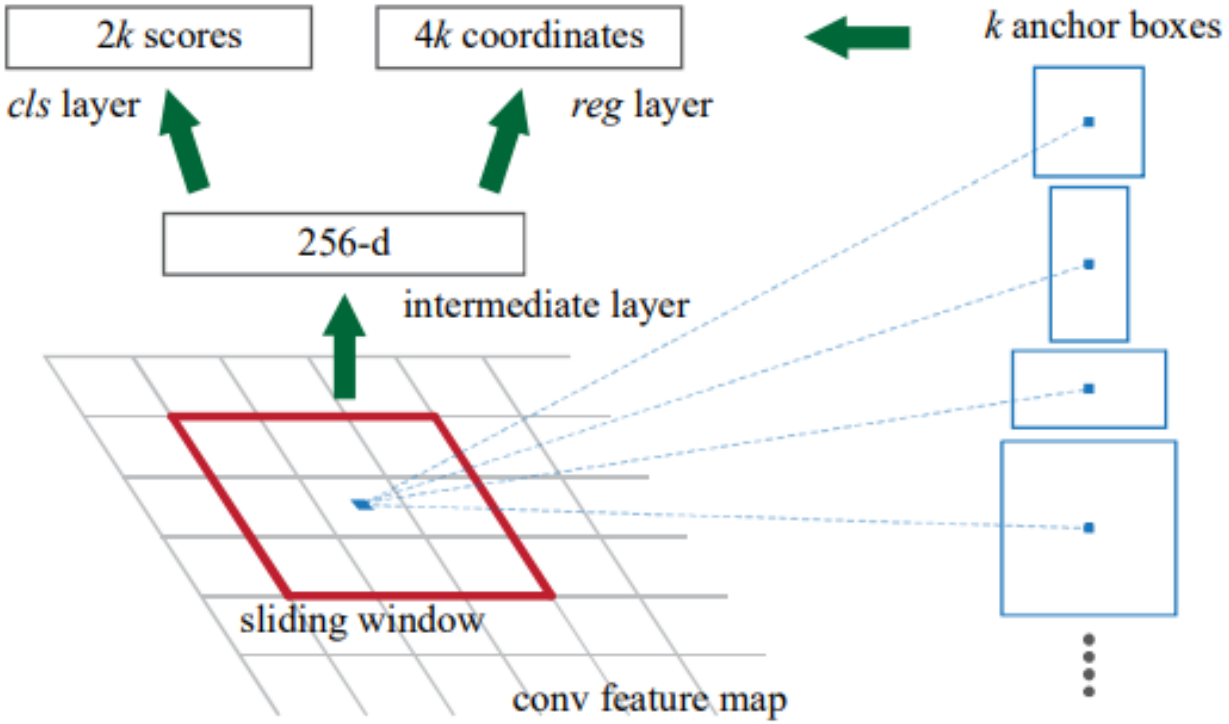


"Faster R-CNN: Towards Real-Time Object Detection Proposal Networks"

Early Deep-Learning Based Approaches (Faster R-CNN)

Region Proposal Network

- A procedure of alternating training is used where both sub-networks are trained at the same time. This allows the parameters in the feature detector deep CNN to be fine-tuned for both tasks simultaneously.
- For VGG-16 model, it has a frame rate of 5 fps on a K-40 GPU (**200 msec per image**).

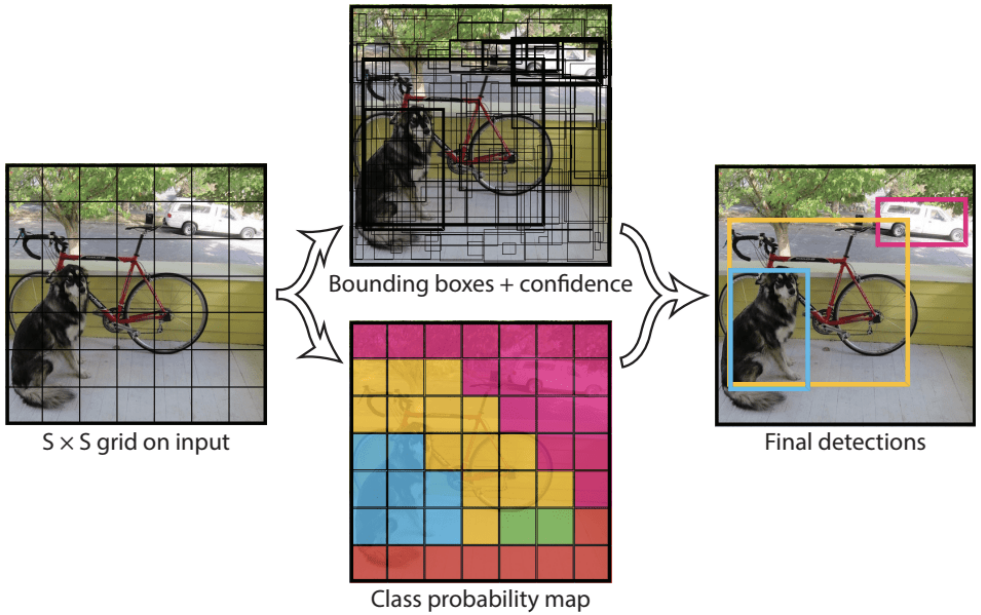


"Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks"

Recent Advancements

Au-Zone Technologies

Recent Advancements (YOLO-V1) 2016

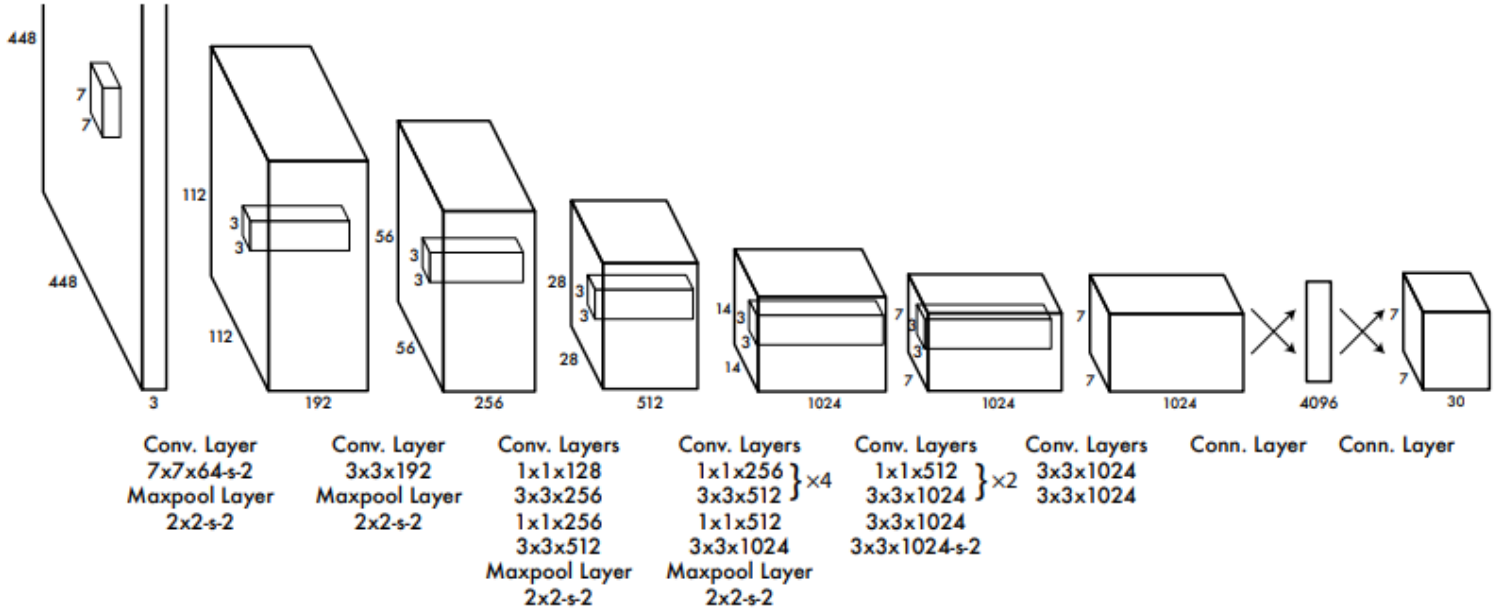


- Doesn't run network sequentially on regions.
- It divides the input image into an S × S grid. If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.
- Each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities.
- These predictions are encoded as an S × S × (B ∗ 5 + C) tensor.
- On PASCAL VOC, S = 7, B = 2. PASCAL VOC has 20 labelled classes so C = 20. The final prediction is a 7 × 7 × 30 tensor

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

- **A single neural network predicts bounding boxes and class probabilities** directly from full images in one evaluation.
- Base YOLO model processes images on Titan X GPU in real-time at 45 fps, and up to 155 fps for a speed-optimized version of the model.
- YOLO makes more localization errors but is far less likely to predict false detections where nothing exists.

Recent Advancements (YOLO-v1: Network Architecture)



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

Limitations:

- YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes.
- It can only have one class in a cell. This model struggles with small objects that appear in groups, such as flocks of birds.
- This model uses relatively coarse features for predicting bounding boxes (it has multiple down-sampling layers).
- Finally, they train on a loss function that approximates detection performance.

- The network has 24 convolutional layers followed by 2 fully connected layers.
- Alternating 1 × 1 convolutional layers reduce the features space from preceding layers.
- The convolutional layers were pretrained on the ImageNet classification task.

Recent Advancements (YOLO-v3 and YOLO-v4)

V3 – 2018:

- The input image is divided into S x S grids at three different resolution scales (8x8, 16x16, 32x32) (large, medium and small)
- Each resolution scale is attended by each model output (3) and the output is conditioned by the input dimension H x W ([H/8, W/8, 3 * (5 + C)], [H/16, W/16, 3 * (5 + C)], [H/32, W/32, 3 * (5 + C)])
- Objects are learned based on the resolution and anchors. Each resolution has at least 3 anchors to define objects ratios at each cell.

V4 – 2020:

- Backbone: CSPDarknet53
- Head: YOLOv3

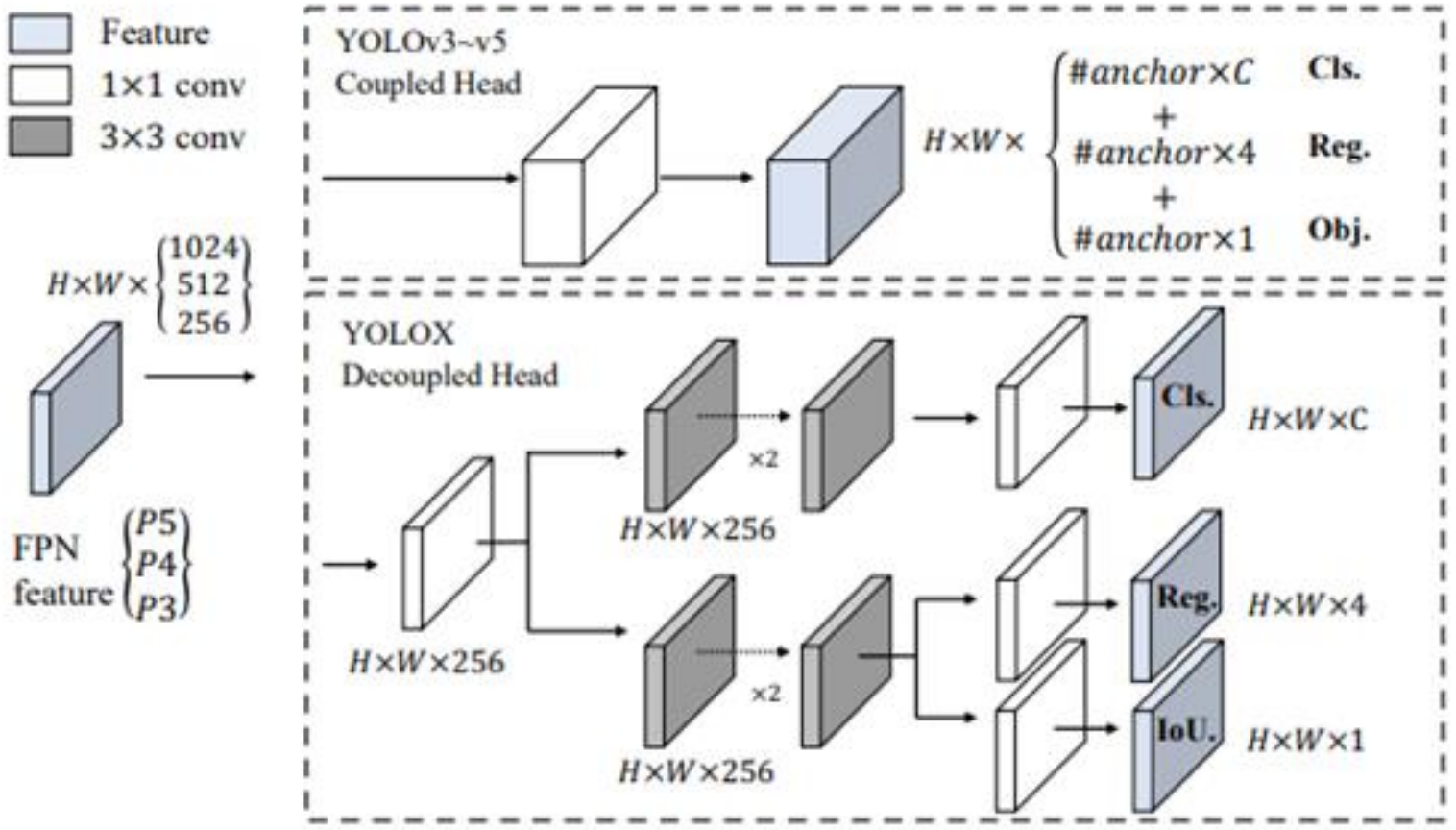
YOLO v4 utilizes:

- Bag of Freebies (BoF) for backbone: CutMix, Mosaic data augmentation, DropBlock regularization, Class label smoothing
- Bag of Specials (BoS) for backbone: Mish activation, Cross-stage partial connections (CSP), Multi-input weighted residual connections
- Bag of Freebies (BoF) for detector: CIoU- loss, CmBN, DropBlock regularization, Mosaic data augmentation etc.

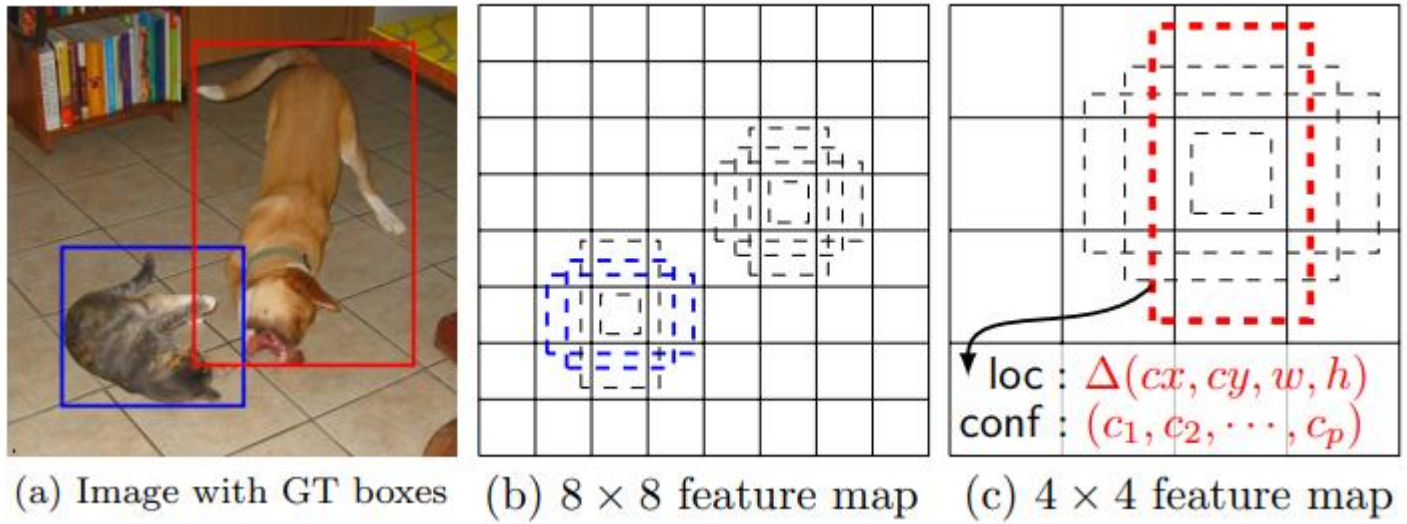
Recent Advancements (YOLOX) 2021

Anchor free, decoupled heads

- Feature channel is reduced to 256 using 1x1 conv.
- Add two parallel branches with two 3x3 conv layers (classification and regression)
- IoU branch is added to the regression branch.



Recent Advancements
SSD (Single Shot Detectors) 2016



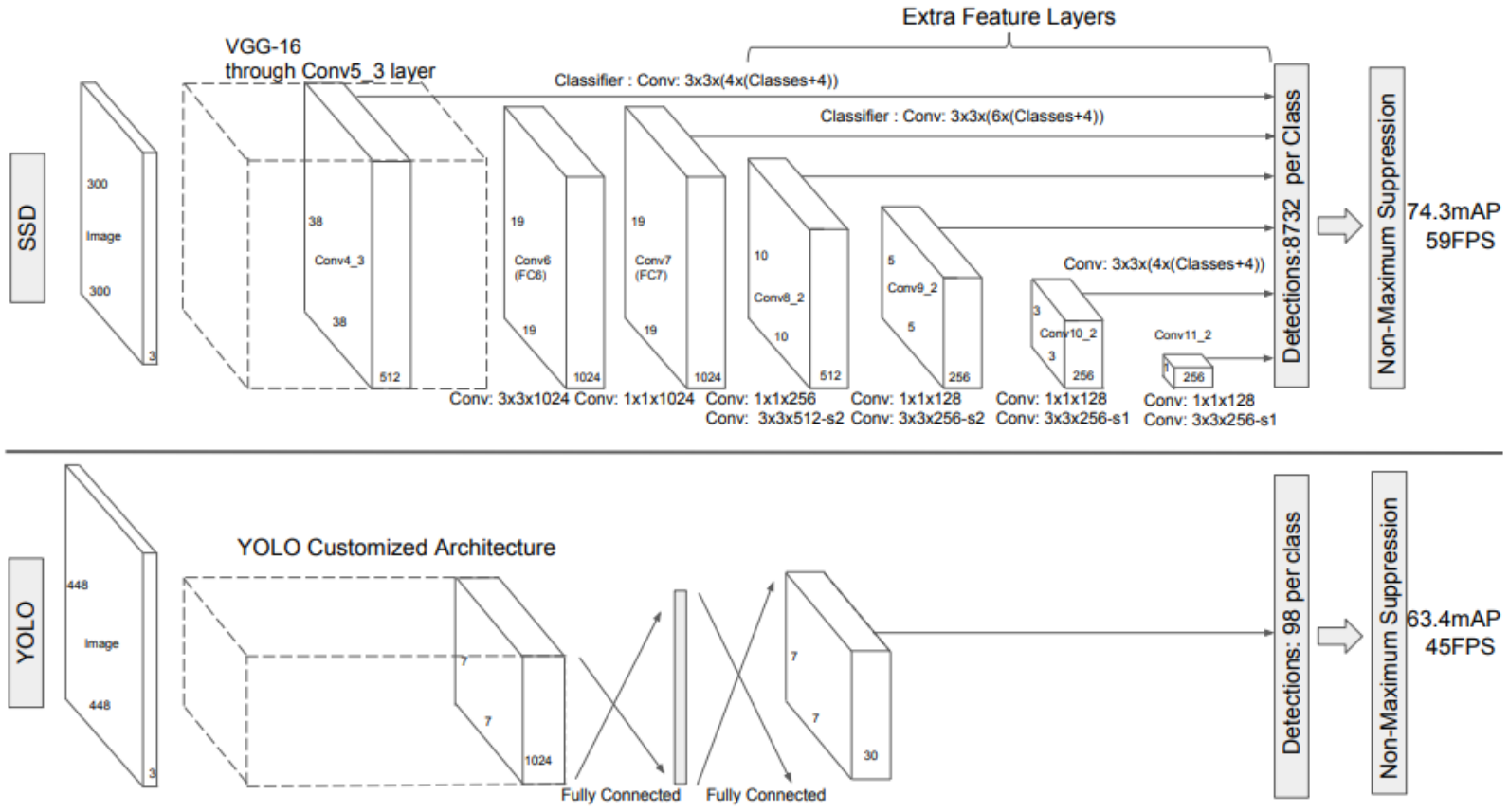
"SSD: Single Shot MultiBox Detector"

- In a convolutional fashion, it evaluates a small set of default boxes of different aspect ratios (anchor boxes) at each location in several feature maps with different scales (e.g. 8 × 8 and 4 × 4 in (b) and (c)).

- For each default box, it predicts both the shape offsets and the confidences for all object categories.

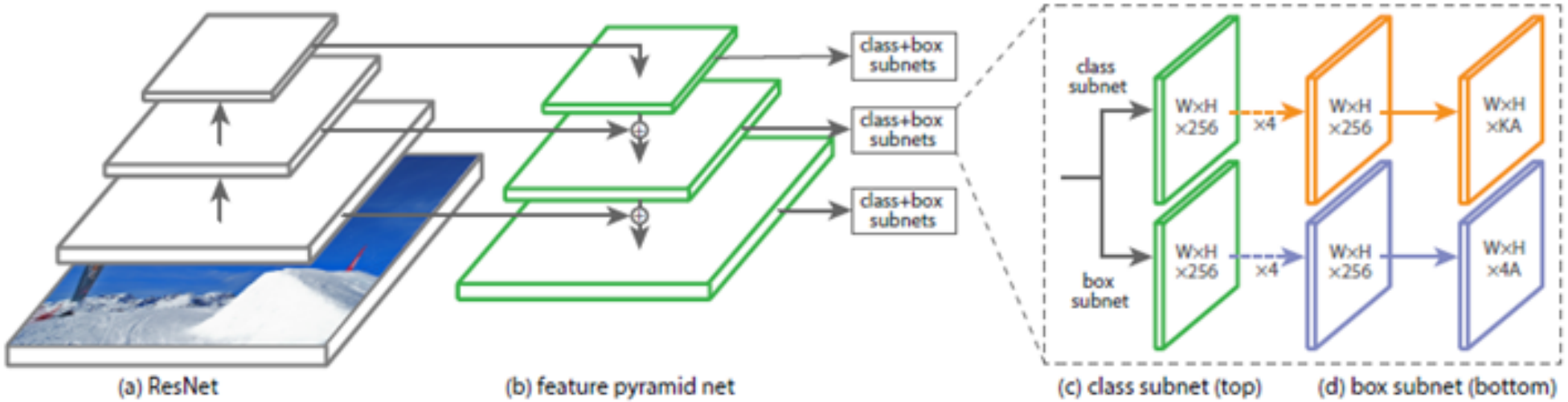
Recent Advancements (SSD – Network Architecture)

- SSD model adds several "feature scaler layers" to the end of a backbone network, which predict the offsets to default boxes of different scales and aspect ratios and their associated confidences.
- Inference frame rate (speed with batch size 8 using Titan X and cuDNN v4 with Intel Xeon E5-2667v3 @ 3.20 GHz) is 46 fps for SSD of 300x300 pixel images with mAP = 74.3%
- Whereas for YOLO (VGG-16) frame rate is 21 fps of 448x448 pixel images with mAP = 66.4%
- For SSD, candidate bounding boxes can be very large (~8732); NMS post-processing needs to be optimized.



"SSD: Single Shot MultiBox Detector"

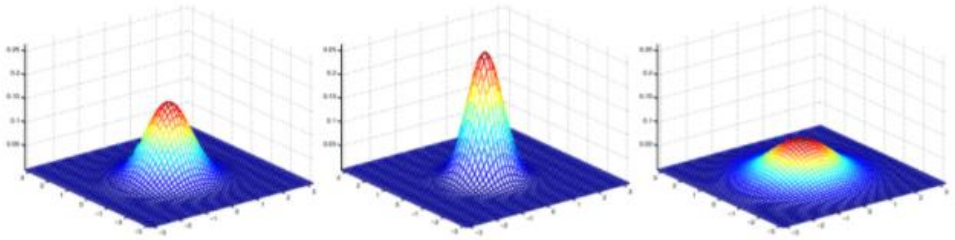
Recent Advancements (RetinaNet) 2017



'Focal Loss for Dense Object Detection〞

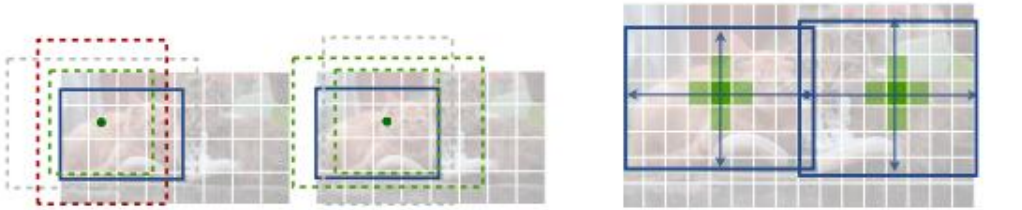
- Utilizes Feature Pyramid Network (FPN) backbone on top of feedforward ResNet architecture.
- Introduces Focal Loss to handle class imbalance in object detection dataset.
- RetinaNet with ResNet-101-FPN matches the accuracy of ResNet- 101-FPN Faster R-CNN [20],
- 172 msec on an Nvidia M40 GPU.
- At most 1k top-scoring predictions per FPN level, after thresholding detector confidence at 0.05.
- The top predictions from all levels are merged and non-maximum suppression (NMS) with a threshold of 0.5 is applied to yield the final detections.

Recent Advancements (CenterNet) 2019



https://sanjivgautamofficial.medium.com/generative-learning-algorithms-8a306976b9b1

- **Addresses the need for anchors and NMS post-processing**
- The backbone network generates heatmaps from the input image using CNNs.
- Object are localized by detecting the local maxima from the heatmaps.
- Single network predicts the keypoints (local maxima of heatmaps), offset, and size.
- The network predicts a total of C + 4 outputs at each location.



(a) Standard anchor based detection. Anchors count as positive with an overlap $IoU > 0.7$ to any object, negative with an overlap $IoU < 0.3$, or are ignored otherwise.

(b) Center point based detection. The center pixel is assigned to the object. Nearby points have a reduced negative loss. Object size is regressed.

"Objects as Points"

- Not best for accuracy but provides excellent accuracy-speed tradeoffs.
- Can easily be extended to 3D object detection, multi-object tracking and pose estimation etc.

Performance Metrics

Performance Metrics
How well the detector is doing?

Types of Errors:

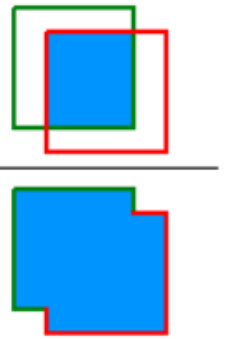
- True Positive (TP) — Correct detection made by the model.
- False Positive (FP) — Incorrect detection made by the detector.
- False Negative (FN) — A Ground-truth missed (not detected) by the object detector.
- True Negative (TN) — This is background region not detected by the model. This metric is not used in object detection because such regions are not explicitly annotated.
- Closeness or overlap of the detected bounding box with the ground truth (IoU).

Ref: https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e#:~:text=Average%20Precision%20(AP)%20and%20mean,COCO%20and%20PASCAL%20VOC%20challenges.

Intersection over Union (IoU):

IoU metric in object detection evaluates the degree of overlap between the ground (*gt*) truth and predicted (*pd*) bounding boxes. It is calculated as follows:

$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\quad}{\quad}$$

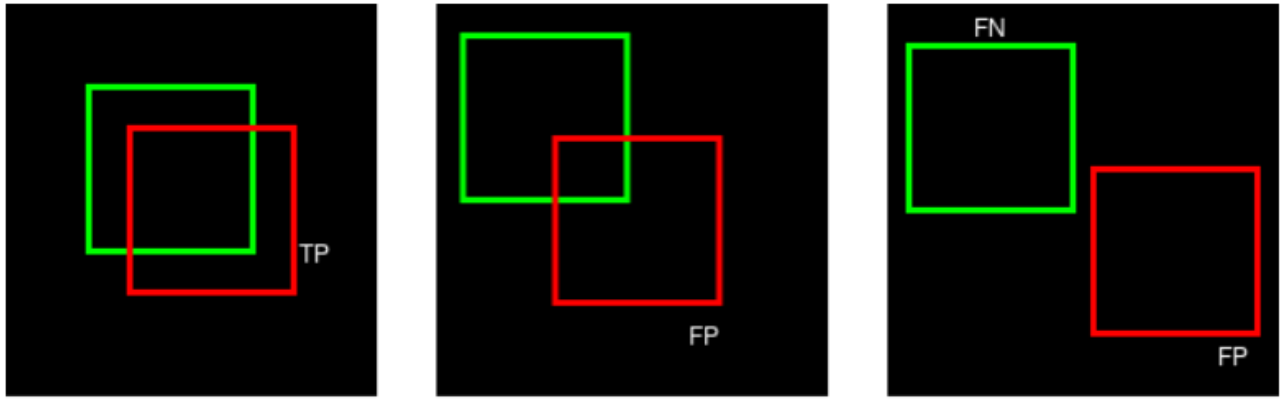


- IoU ranges between 0 and 1, where 0 shows no overlap and 1 means perfect overlap between *gt* and *pd*.
- IoU is used as a threshold (say, α), and using this threshold we can decide if a detection is correct or not.

Performance Metrics (IoU)

- True Positive (TP) is a detection for which IoU(*gt,pd*) ≥ α and
- False Positive (FP) is a detection for which IoU(*gt,pd*) < α.
- False Negative is a ground-truth (gt) missed together with gt for which IoU(*gt,pd*) < α.

Ref: https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e#:~:text=Average%20Precision%20(AP)%20and%20mean,COCO%20and%20PASCAL%20VOC%20challenges.



IoU = 0.86 IoU = 0.24 IoU = 0

Consider IoU threshold, α = 0.5, then TP, FP and FNs can be identified as shown in the figure above.

Note: If we raise the IoU threshold above 0.86, the first instance will be a FP, and if we lower the IoU threshold below 0.24, the second instance becomes TP.

Performance Metrics (Precision and Recall)

- **Precision** is the degree of exactness of the model in identifying only relevant objects. It is the ratio of TPs over all detections made by the model.
- **Recall** measures the ability of the model to detect all ground truths—TPs among all ground truths.
- A model is said to be a good model if it has high precision and high recall.

Ref: https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e#:~:text=Average%20Precision%20(AP)%20and%20mean,COCO%20and%20PASCAL%20VOC%20challenges.

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{TP}{\text{all detections}}$$

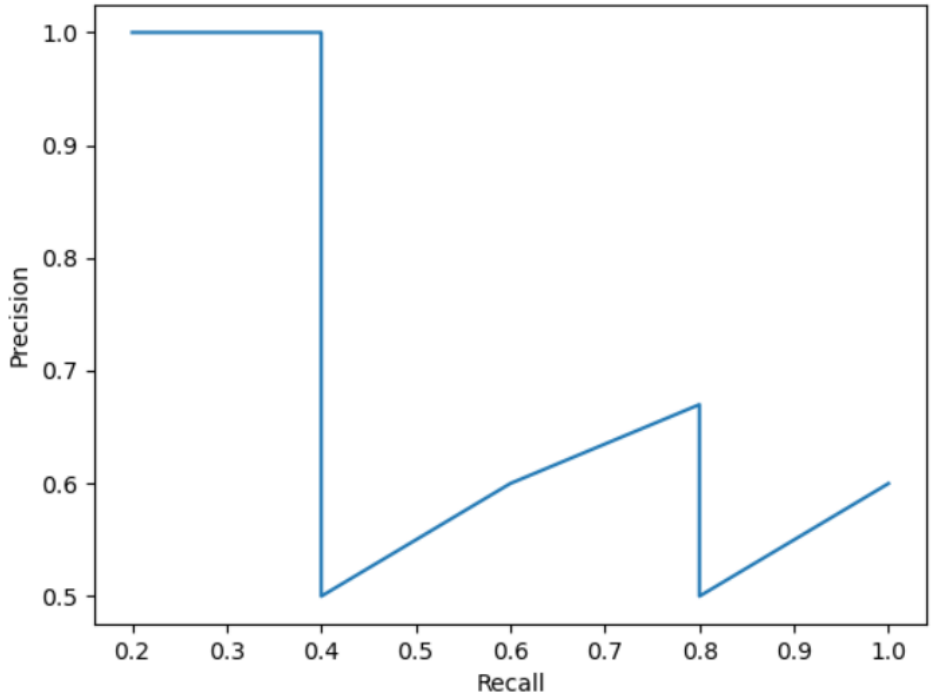
$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{TP}{\text{all ground-truths}}$$

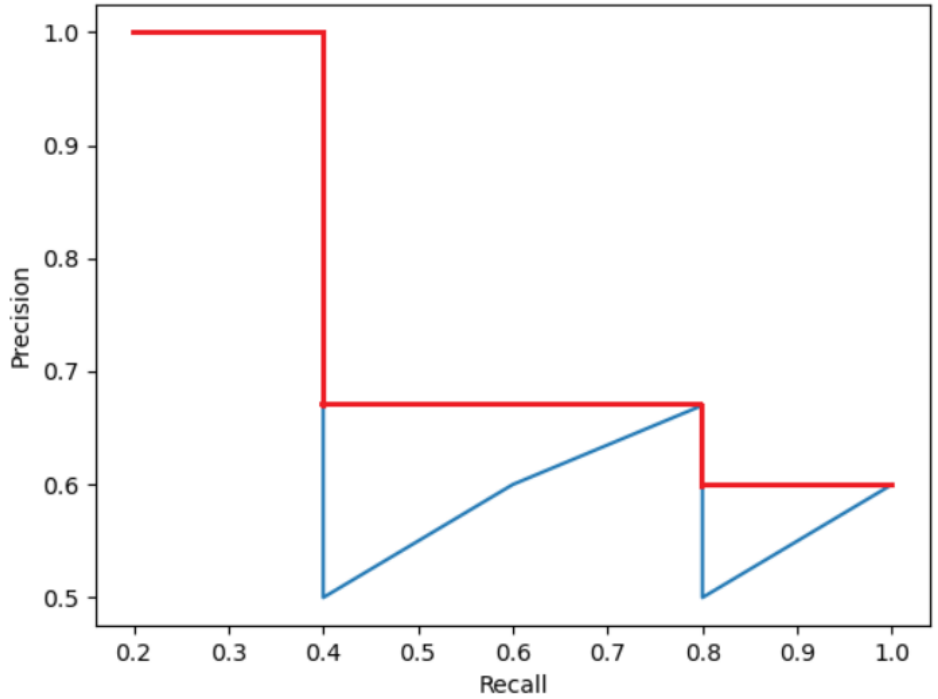
- Raising IoU threshold means that more objects will be missed by the model (more FNs and therefore low recall and high precision).
- Whereas a low IoU threshold will mean that the model gets more FPs (hence low precision and high recall).
- **For a good model, precision and recall stays high even when confidence score is varied.**

Performance Metrics (Precision-Recall Curve)

- The precision-recall (PR) curve is a plot of precision and recall at varying values of IoU thresholds.
- P-R curve is not monotonic, therefore smoothing is applied.

Ref: https://debuggercafe.com/evaluation-metrics-for-object-detection





Performance Metrics (mean Average Precision : mAP)

Average Precision (AP):

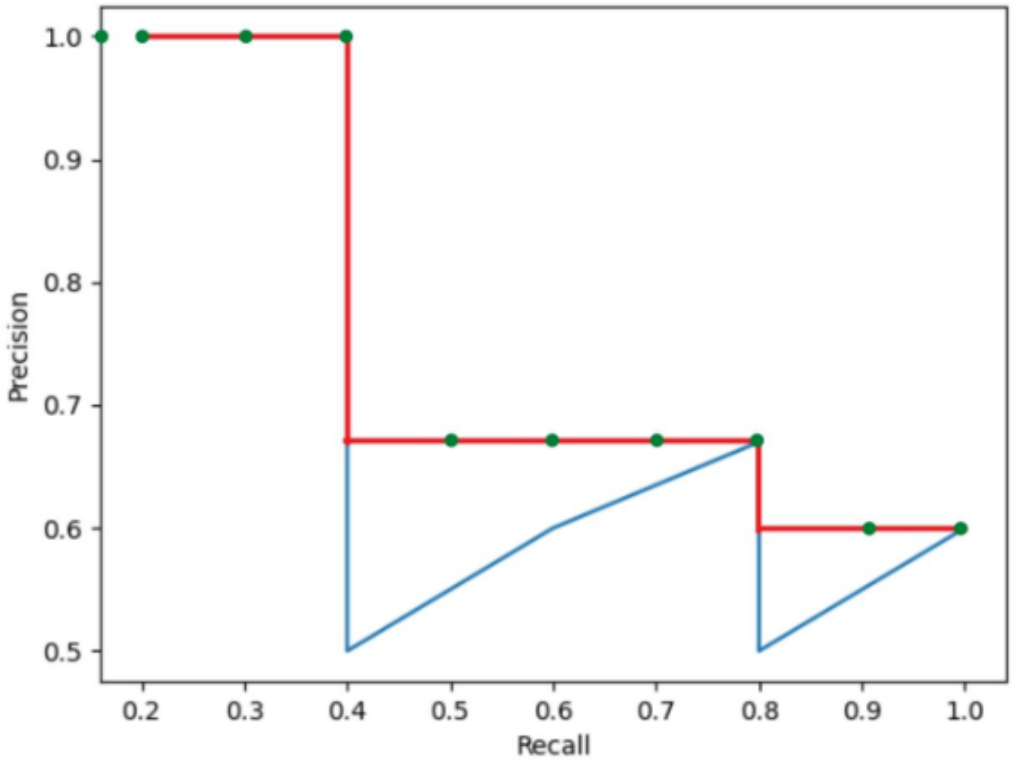
- AP@α is Area Under the Precision-Recall Curve (AUC-PR) evaluated at α IoU threshold.
- **AP50** and **AP75** mean AP calculated at IoU=0.5 and IoU=0.75, respectively.

Ref: https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e#:~:text=Average%20Precision%20(AP)%20and%20mean,COCO%20and%20PASCAL%20VOC%20challenges.

Ref: https://debuggercafe.com/evaluation-metrics-for-object-detection

Mean Average Precision (mAP):

- There are as many AP values as the number of classes.
- These AP values are averaged to obtain the metric - **mean Average Precision (mAP)**.
- **Pascal VOC benchmark** requires computing AP at 11 points - AP@[0,0.1,..., 1.0]



Performance Metrics (MS COCO Benchmark)

- MS COCO describes 12 evaluation metrics for submitting the results and determining the winners for the competition,
- The main evaluation metric is still the **mAP, or simply called AP**.

Ref: https://debuggercafe.com/evaluation-metrics-for-object-detection

Average Precision (AP):

AP	% AP at IoU=.50:.05:.95 **(primary challenge metric)**
$AP^{IoU=.50}$	% AP at IoU=.50 (PASCAL VOC metric)
$AP^{IoU=.75}$	% AP at IoU=.75 (strict metric)

AP Across Scales:

AP^{small}	% AP for small objects: area < 32^2
AP^{medium}	% AP for medium objects: 32^2 < area < 96^2
AP^{large}	% AP for large objects: area > 96^2

Average Recall (AR):

$AR^{max=1}$	% AR given 1 detection per image
$AR^{max=10}$	% AR given 10 detections per image
$AR^{max=100}$	% AR given 100 detections per image

AR Across Scales:

AR^{small}	% AR for small objects: area < 32^2
AR^{medium}	% AR for medium objects: 32^2 < area < 96^2
AR^{large}	% AR for large objects: area > 96^2

Comparisons

Comparisons (SSD, YOLO and Others)

Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

Table 7: **Results on Pascal VOC2007 `test`.** SSD300 is the only real-time detection method that can achieve above 70% mAP. By using a larger input image, SSD512 outperforms all methods on accuracy while maintaining a close to real-time speed.

"SSD: Single Shot MultiBox Detector"

(With batch size 8 on Titan X and cuDNN v4 with Intel Xeon E5-2667v3 @ 3.20GHz.)

Comparisons (RetinaNet with Others)

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [16]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [20]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [17]	Inception-ResNet-v2 [34]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [32]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [27]	DarkNet-19 [27]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [22, 9]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [9]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet (ours)	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet (ours)	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2

Table 2. **Object detection** *single-model* results (bounding box AP), *vs.* state-of-the-art on COCO `test-dev`. We show results for our RetinaNet-101-800 model, trained with scale jitter and for 1.5× longer than the same model from Table 1e. Our model achieves top results, outperforming both one-stage and two-stage models. For a detailed breakdown of speed versus accuracy see Table 1e and Figure 2.

"Focal Loss with Dense Object Detection"

(RetinaNet with ResNet-101-FPN and a 600x600 pixel image size (*on nVidia M40 GPU) runs at 122 msec per image.)*

Comparisons (CenterNet and Others)

	Backbone	FPS	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
MaskRCNN [21]	ResNeXt-101	**11**	39.8	62.3	43.4	22.1	43.2	51.2
Deform-v2 [63]	ResNet-101	-	46.0	67.9	50.8	27.8	49.1	59.5
SNIPER [48]	DPN-98	*2.5*	46.1	67.0	51.6	29.6	48.9	58.1
PANet [35]	ResNeXt-101	-	47.4	67.2	51.8	30.1	**51.7**	60.0
TridentNet [31]	ResNet-101-DCN	0.7	**48.4**	**69.7**	**53.5**	**31.8**	51.3	**60.3**
YOLOv3 [45]	DarkNet-53	20	33.0	57.9	34.4	18.3	25.4	41.9
RetinaNet [33]	ResNeXt-101-FPN	5.4	40.8	61.1	44.1	24.1	44.2	51.2
RefineDet [59]	ResNet-101	-	36.4 / 41.8	57.5 / 62.9	39.5 / 45.7	16.6 / 25.6	39.9 / 45.1	51.4 / 54.1
CornerNet [30]	Hourglass-104	4.1	40.5 / 42.1	56.5 / 57.8	43.1 / 45.3	19.4 / 20.8	42.7 / 44.8	**53.9** / 56.7
ExtremeNet [61]	Hourglass-104	3.1	40.2 / 43.7	55.5 / 60.5	43.2 / 47.0	20.4 / 24.1	43.2 / 46.9	53.1 / 57.6
FSAF [62]	ResNeXt-101	*2.7*	**42.9** / 44.6	**63.8** / **65.2**	**46.3** / 48.6	**26.6** / **29.7**	**46.2** / **47.1**	52.7 / 54.6
CenterNet-DLA	DLA-34	**28**	39.2 / 41.6	57.1 / 60.3	42.8 / 45.1	19.9 / 21.5	43.0 / 43.9	51.4 / 56.0
CenterNet-HG	Hourglass-104	7.8	42.1 / **45.1**	61.1 / 63.9	45.9 / **49.3**	24.1 / 26.6	45.5 / **47.1**	52.8 / **57.7**

Table 2: State-of-the-art comparison on COCO test-dev. Top: two-stage detectors; bottom: one-stage detectors. We show single-scale / multi-scale testing for most one-stage detectors. Frame-per-second (FPS) were measured on the same machine whenever possible. Italic FPS highlight the cases, where the performance measure was copied from the original publication. A dash indicates methods for which neither code and models, nor public timings were available.

"Points as Objects"

(With Intel Core i7-8086K CPU, Titan Xp GPU, Pytorch 0.4.1, CUDA 9.0, and CUDNN 7.1.)

Choosing the Object Detector (High-Level, General Guidelines)

- High accuracy → More complex, larger models → lower speed
- High speed → Smaller, less complex models → lower accuracy
- Quantization → specific data types (fp16, int16, uint8) → lower accuracy
- Specific compute unit (GPU, NPU) → Highest speed → More effort

Model type	Accuracy	Speed	Model Size	Portability
Floating-point-32 bit (natively trained)	**Highest**	Lowest	Largest	**High**
Floating-point-16 bit	Low	Low	Small	Medium
Quantized (int16, uint8)	Lower	Fast	**Smaller**	Low (special math libraries)
Compute Unit (Not GPUs) (quantized models)	Lower	**Fastest**	Small	Lower

Conclusions

- Early DNN-based object detectors (R-CNN, Fast R-CNN, Faster R-CNN)
- Recent object detectors (YOLO, YOLOX, SSD, RetinaNet, CenterNet)
- Introduced metrics used for object detections
- Discussed performance and speed comparison
- General guidelines to choose object detectors (no free lunch!)

Au-Zone Technologies Inc.

- Smarter devices. Valuable Insights. Enhanced Security & Privacy. Novel customer experiences.
- By taking advantage of our field proven Starter Kits, Optimized Vision Pipelines and Advanced AI Models, our customers can significantly improve runtime performance while reducing technology risk, need for specialized skills and program development time.
- Well over a decade of experience helping clients get their embedded AI and Computer Vision concepts off the drawing board and into production, the engineering design team can help you quickly and confidently explore, develop, and deploy practical embedded vision and AI solutions at scale.

References (Early Approaches)

Method	Bibliography	Link
R-CNN	Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", Tech report (v5), UC Berkeley, 2014	https://arxiv.org/pdf/1311.2524.pdf
Fast R-CNN	Ross Girshick, "Fast R-CNN", Microsoft Research, 2015.	https://arxiv.org/pdf/1504.08083.pdf
Faster R-CNN	Shaoqing Ren, Kaiming He, Ross Girshick and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", Advances in Neural Information Processing Systems, 2015.	https://arxiv.org/pdf/1506.01497.pdf

References (Recent Advancements)

Method	Bibliography	Link
YOLO-v1	Joseph Redmon, Santosh Divvalay, Ross Girshick, Ali Farhadiy, "You Only Look Once: Unified, Real-Time Object Detection", 2015.	https://arxiv.org/pdf/1506.02640.pdf
YOLO-v3	Redmon, Joseph, and Ali Farhadi. "Yolov3: An incremental improvement.", 2018.	https://arxiv.org/pdf/1804.02767.pdf
YOLO-v4	Bochkovskiy A, Wang CY, Liao HY. Yolov4: Optimal speed and accuracy of object detection, 2020.	https://arxiv.org/pdf/2004.10934.pdf
YOLOX	Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, Jian Sun. YOLOX: Exceeding YOLO Series in 2021.	https://arxiv.org/pdf/2107.08430.pdf
SSD	Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, "SSD: Single Shot MultiBox Detector", 2015	https://arxiv.org/pdf/1512.02325.pdf

References (Recent Advancements)

Method	Bibliography	Link
RetinaNet	Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár, "Focal Loss for Dense Object Detection", 2017.	https://arxiv.org/pdf/1708.02002.pdf
CenterNet	Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, Qi Tian, "CenterNet: Keypoint Triplets for Object Detection", 2019.	https://arxiv.org/pdf/1904.08189.pdf
	Object Detection Metrics With Worked Example	https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e#:~:text=Average%20Precision%20(AP)%20and%20mean,COCO%20and%20PASCAL%20VOC%20challenges

Questions?

Au-Zone Technologies