Object detection review

Objectives

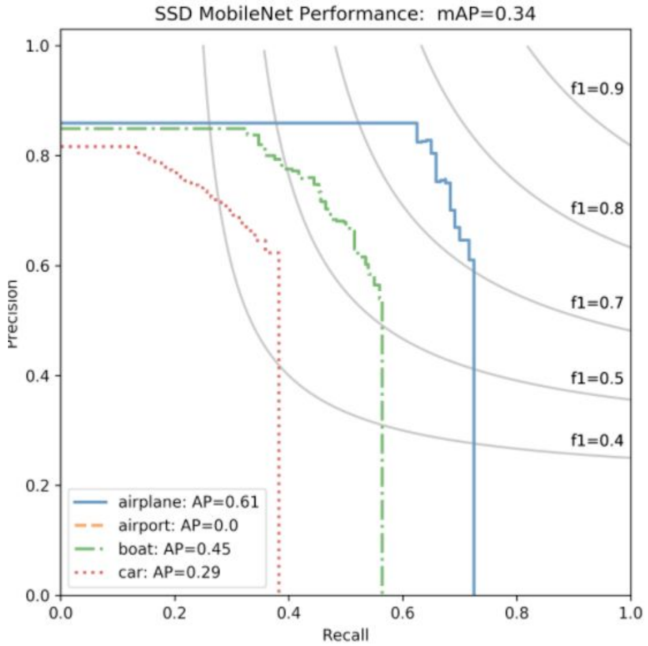
- Localize objects - bounding box (2D / 3D)
- Classification- multiple-classes
- Recognize objects at vastly different scales
- Multi-task: key-point detection.
- High mean average precision. Precision vs Recall.
- Computational efficiency.

Mean Average Presicion - mAP

- **Precision** measures how accurate is your predictions. The percentage of your predictions are correct.
- **Recall** measure how good you find all the positives.
- **mAP** is defined as area below the precision-recall curve.

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

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

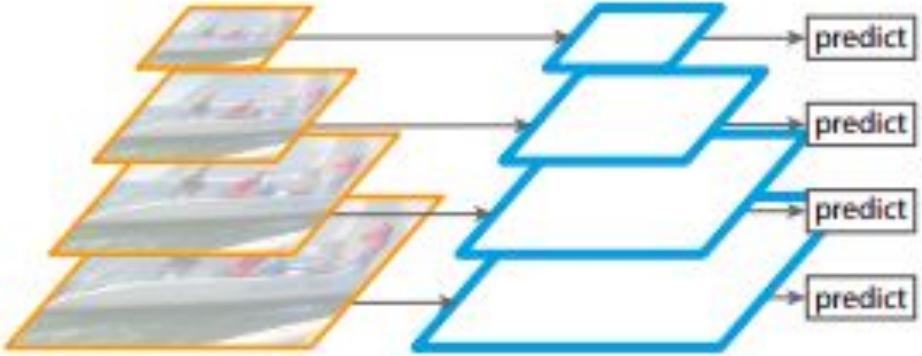


Overview

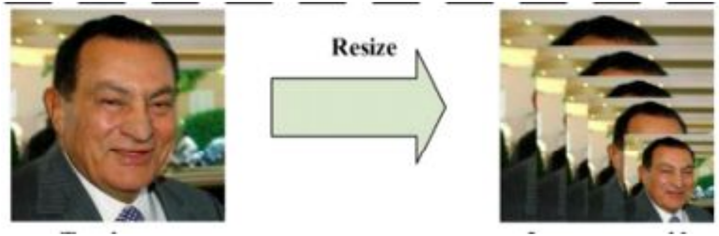
- Scale pyramid
- Two stage detectors:
    - R-CNN: Mask-R-CNN, Fast-R-CNN, **Faster-R-CNN**
- **One stage detectors:**
    - Detection heads: SSD, **FPN**, YOLO.
    - **Anchor based vs Anchor free.**

Scale pyramid

- Input image is scaled multiple times, to detect wide range of objects size.
- Nms between all predictions.
- Example, MTCNN for face detection.
- High inference time.



(a) Featurized image pyramid



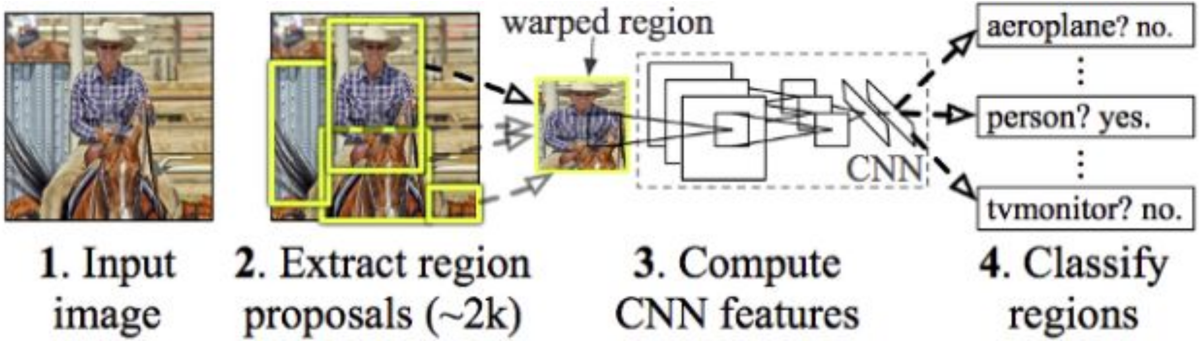
Two-Stage detectors

- Two stage detectors include basicly 3 main parts:
  - CNN backbone
  - Region-Of-Interest (ROI) proposals
  - Classification / Regression heads

Two-Stage detectors

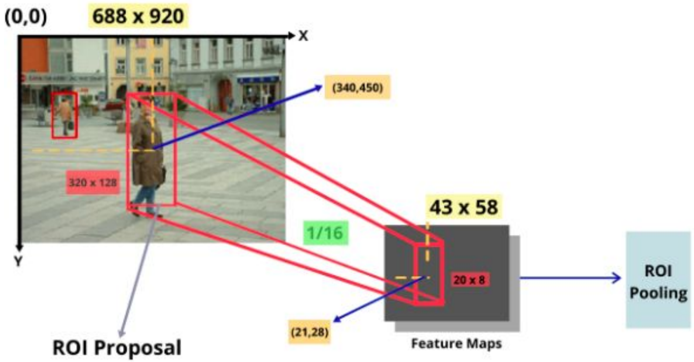
R-CNN

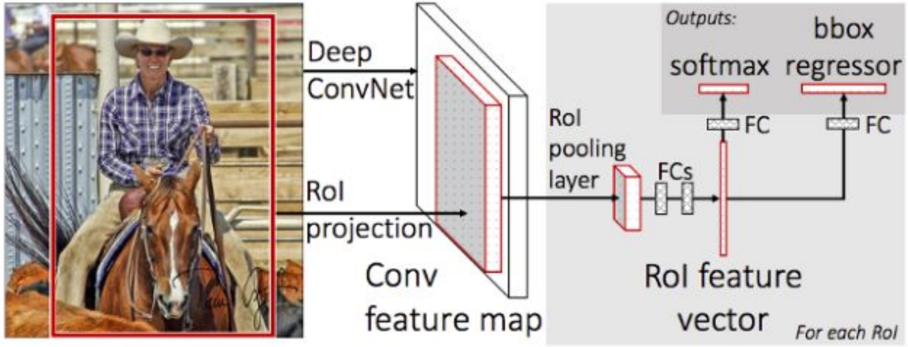
- Region proposal of 2000 regions using Selective search SS.
- SS is and iterative algorithm to segment the image to regions, very greedy!.
- Far from real-time around 47 seconds for each image!!.
- SS is a fixed algorithm, no learning is happening.



Two-Stage detectors
Fast R-CNN

- Instead of feeding 2000 regions, the input image is fed to CNN to generate one feature map.
- SS is running as usual, the region proposals generated are projected to the feature map.
- proposals are wrapped to fixed-size squares using ROI pooling layer.
- 25x faster than R-CNN



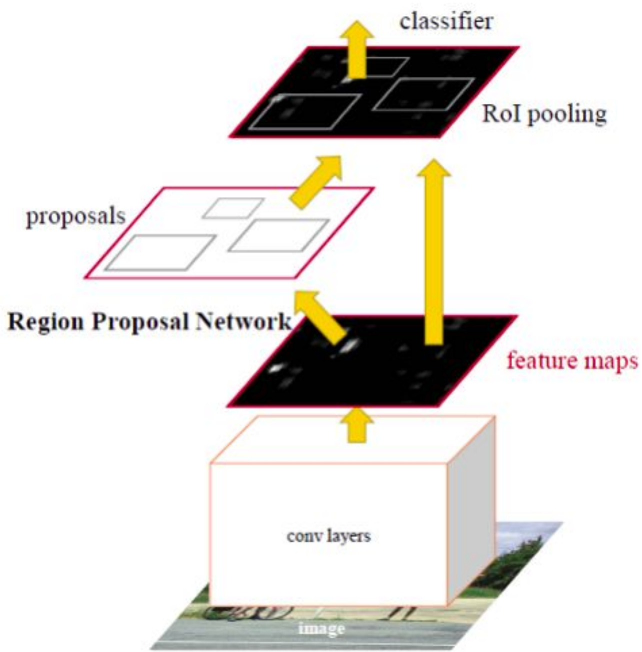



ROI pooling

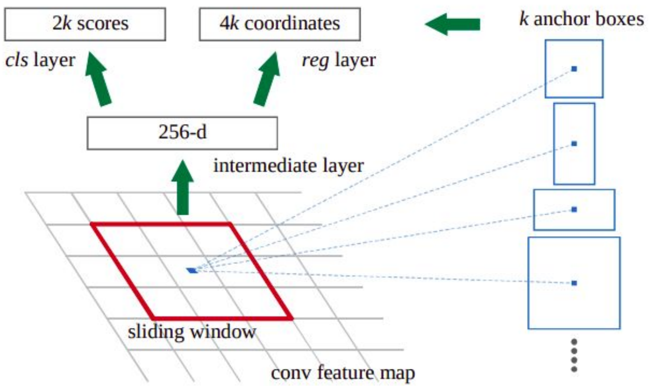
Two-Stage detectors

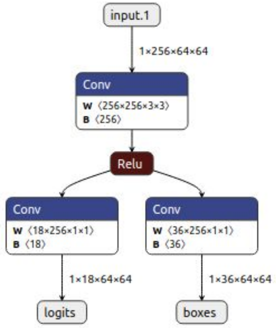
Faster R-CNN - RPN explained

- Region Proposal Network (**RPN**) instead of Selective Search.
- Slide over the feature map with n x n spatial window.
- Each sliding window regress k bounding boxes which are compared to k reference anchor boxes.
- Proposals are filtered based on their “objectness score” and nms.



Unified Network of Faster R-CNN. Image Source





k = 9

	RCNN	Fast RCNN	Faster RCNN
Test time per image with Proposals	50 seconds	2 seconds	0.2 seconds
(Speedup)	1x	25x	250x
mAP (PASCAL VOC 07)	66.0	66.9	66.9

SSD - Single Shot Detection
Architecture

- one-stage detector, bottom-up design.
- Use base network and SSD layers to regress location and confidence from different scaled feature maps.
- Feature maps from different levels are known to have different receptive field sizes.

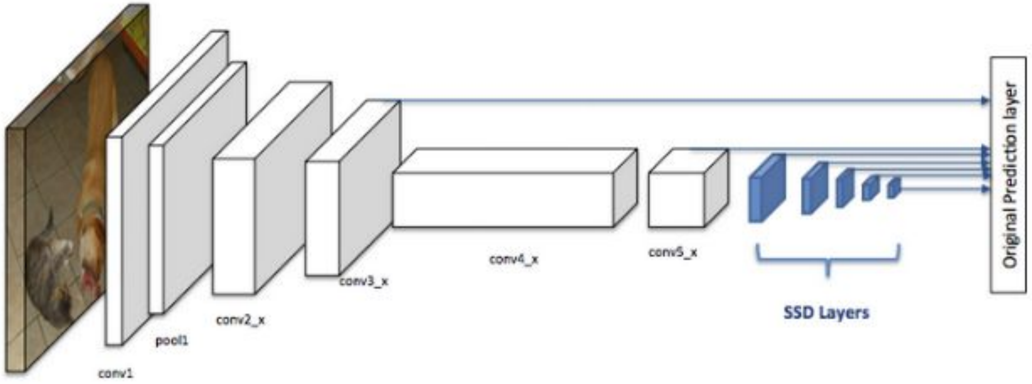
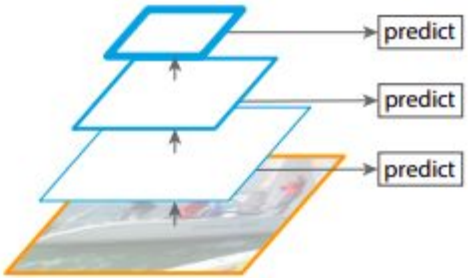


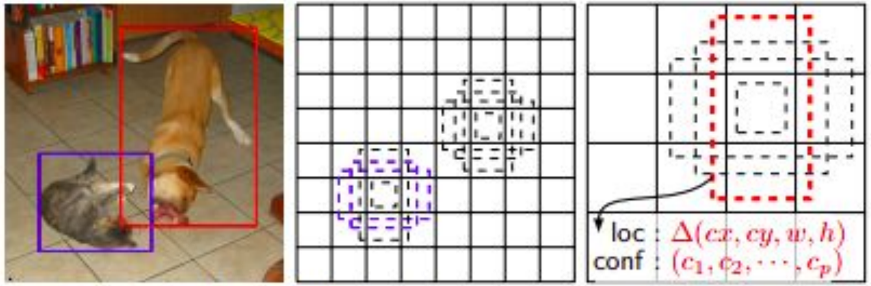
Figure 3. Architecture of a convolutional neural network with a SSD detector [2]



(c) Pyramidal feature hierarchy

SSD - Single Shot Detection
Regression & positive assignment

- SSD net output multiple feature maps, with different spatial sizes m x n.
- For each position in m x n we assign k anchors with different aspect ratios. For example {1, 2, 3, ½, ⅓}
- For each feature maps anchors regress:
    - cx, cy offset from anchor center position.
    - w, h size of box.
    - s_c classification score for the c_{th} class.
- In total for a single feature map:
    - num of anchors: k x m x n
    - regress for each anchor: c + 4
- An anchor is considered as positive if IOU > 0.5.



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

SSD - Single Shot Detection

Losses

- x_{ij} {0, 1} match between i box to j gt box..
- N num of positive boxes.
- l predicted box.
- g ground-truth box
- d default-box (anchor)

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,l,g)) \qquad (1)$$

$$L_{loc}(x,l,g) = \sum_{i \in Pos}^{N} \sum_{m \in \{cx,cy,w,h\}} x_{ij}^{k} \text{smooth}_{\text{L1}}(l_i^m - \hat{g}_j^m)$$

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h \qquad (2)$$

$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right)$$

$$L_{conf}(x,c) = -\sum_{i \in Pos}^{N} x_{ij}^p log(\hat{c}_i^p) - \sum_{i \in Neg} log(\hat{c}_i^0) \quad \text{where} \quad \hat{c}_i^p = \frac{\exp(c_i^p)}{\sum_p \exp(c_i^p)} \qquad (3)$$

Hard Mining of Top negative samples, 3:1 ratio negatives:positives.

SSD - Single Shot Detection
Experimental results

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

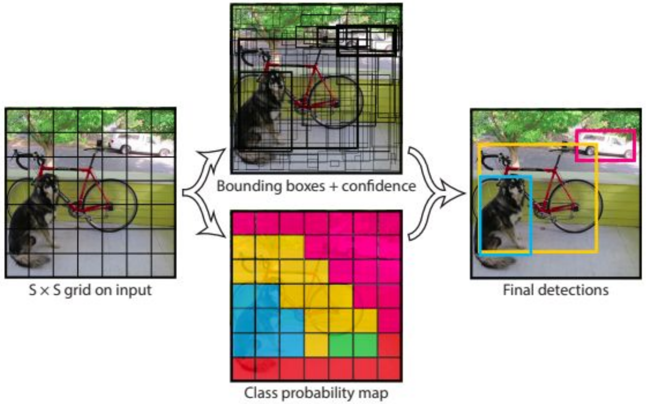
Table 7: **Results on Pascal VOC2007 `test`.** SSD300 is the only real-time detection method that can achieve above 70% mAP. By using a larger input image, SSD512 outperforms all methods on accuracy while maintaining a close to real-time speed.

Pro's:
- Fast

Con's:
- Low accuracy on small objects, due to low resolution feature maps.
- Hyper-parameters configurations: aspect ratios.
- Change in input size lead to change in pre-defined anchors.

YOLO

- Divide the input image to S x S grid, grid cell is responsible to the detect an object if it center falls in it.
- Each cell predicts B bounding boxes, regressing x,y,w,h and confidence which represent the IOU between predicted box and ground truth box.
- Each cell also predicts C conditional class probabilities
- In total YOLO predicts:

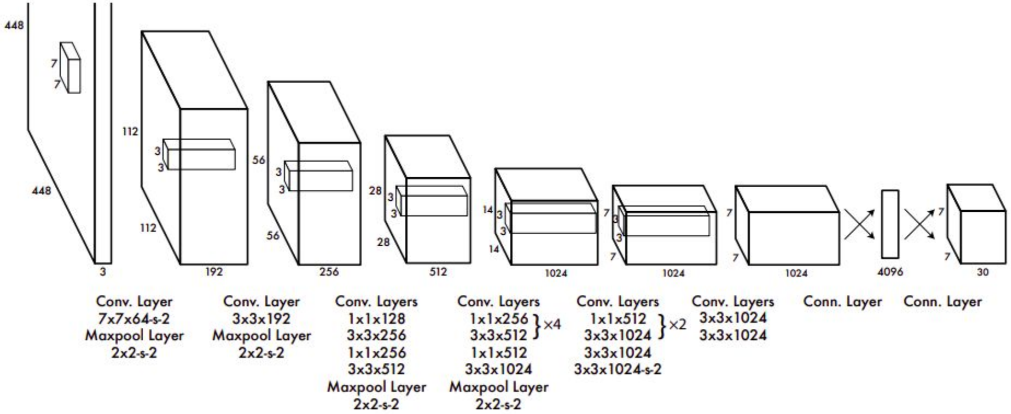


$$S \times S \times (B * 5 + C)$$

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

YOLO

- GoogleNet backbone, with 24 conv layers, followed by 2 fully connected layers



YOLO

- Partially address contribution of large vs. small boxes predicting square root of sizes.
- Background / foreground imbalance.
- $\mathbb{1}_i^{\text{obj}}$ - object appears in cell i.
- $\mathbb{1}_{ij}^{\text{obj}}$ - predictor is “responsible” for that prediction, meaning have highest IOU with g-t.

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

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

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

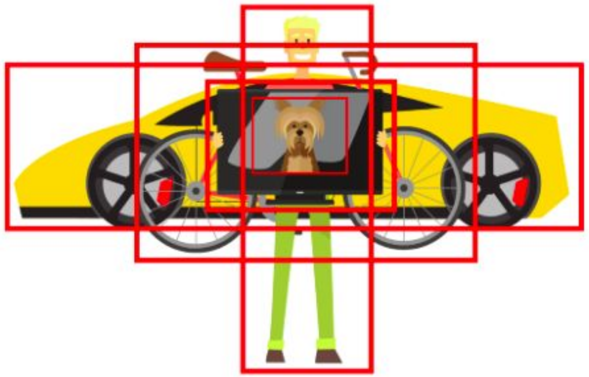
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

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

$\lambda_{\text{coord}} = 5$ and $\lambda_{\text{noobj}} = .5$.

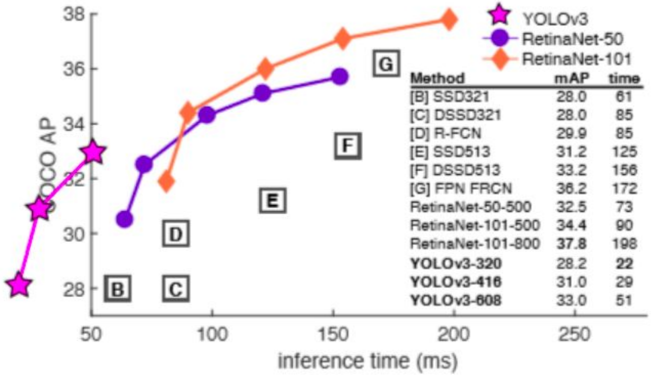
YOLOv2

- Yolo v1 has several limitations:
    - Detection of nearby objects.
    - Detection of small objects.
    - Bad localization in small boxes.
    - Low recall compared to Two-stage detectors.
- Yolo v2 - focus on improving recall and localization:
    - Batch normalization: 2% improvement in mAP.
    - Multi-scale training {320 to 608}, predicts objects in different resolutions.
    - Using anchor boxes. Better recall but drops in precision.
    - Darknet-19 as backbone.



YOLOv3

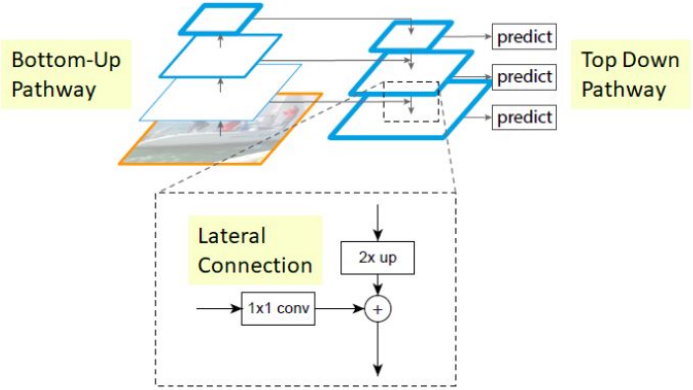
- Bigger and more accurate:
    - Multi labels prediction: object have multiple labels (woman:person), using binary cross-entropy instead of softmax.
    - Short-cut connections: more finer-grained information from earlier feature maps. improve detection of small objects.
    - perform less than previous version with medium and large-size objects.
    - Darknet-53 as backbone, with residual connections.
    - Predict boxes at 3 different scales. (one scale for YOLOv2).
    - Still perform poorly to regress localization well (AP75).



	backbone	AP	AP_{50}	AP_{75}
Two-stage methods				
Faster R-CNN+++ [5]	ResNet-101-C4	34.9	55.7	37.4
Faster R-CNN w FPN [8]	ResNet-101-FPN	36.2	59.1	39.0
Faster R-CNN by G-RMI [6]	Inception-ResNet-v2 [21]	34.7	55.5	36.7
Faster R-CNN w TDM [20]	Inception-ResNet-v2-TDM	36.8	57.7	39.2
One-stage methods				
YOLOv2 [15]	DarkNet-19 [15]	21.6	44.0	19.2
SSD513 [11, 3]	ResNet-101-SSD	31.2	50.4	33.3
DSSD513 [3]	ResNet-101-DSSD	33.2	53.3	35.2
RetinaNet [9]	ResNet-101-FPN	39.1	59.1	42.3
RetinaNet [9]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4

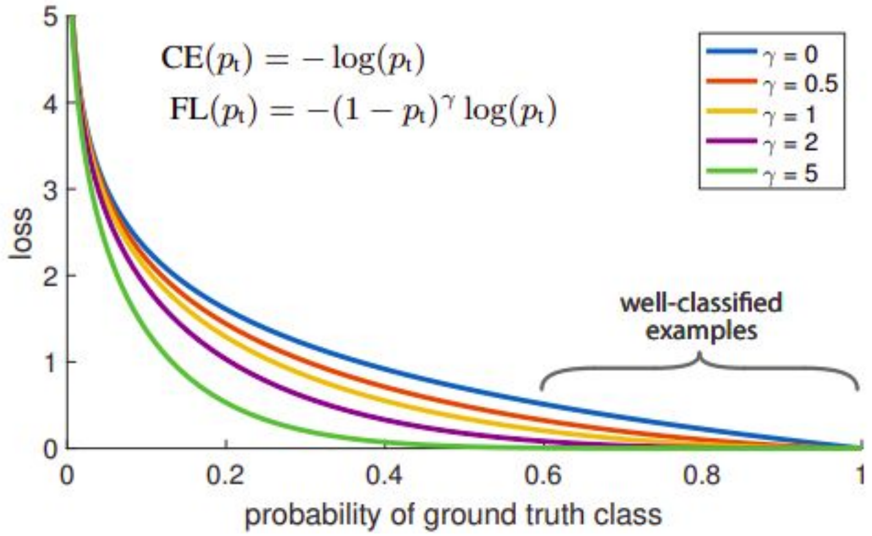
FPN

- Combines low-resolution, semantically strong features with high-resolution semantically weak features via a top-down pathway and lateral connections.
- This process is independent of the backbone, and can used with RPN detector (Faster R-CNN), anchor based detectors (RetinaNet) and widely used for almost all anchor free detectors (FCOS, CornerNet, CenterNet).
- This method achieve the best results compared to other multi-scale feature maps heads, such as, SSD and YOLOv3.



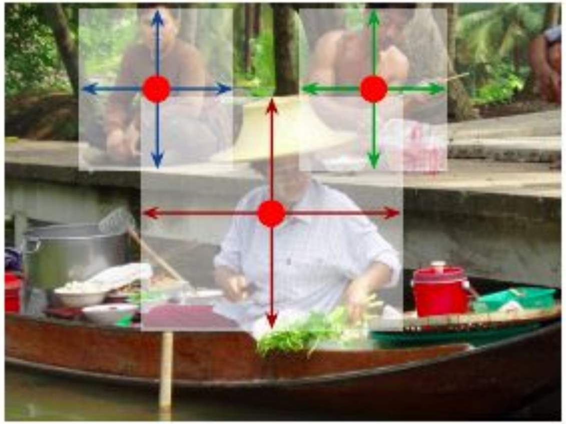
RetinaNet - Focal loss

- reduce contribution of well-classified (easy) samples



CenterNet - anchor free

- Anchor free center-based approach.
- CenterNet consider center of a box as an object, and then uses this predicted center to find the the scales / offsets of the box.
- Paper presents that their method can be extended easily to other tasks:
    - 3D detection
    - Human pose estimation



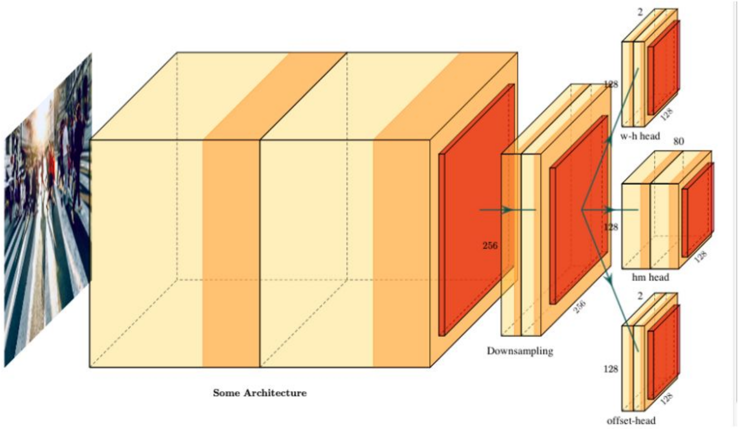
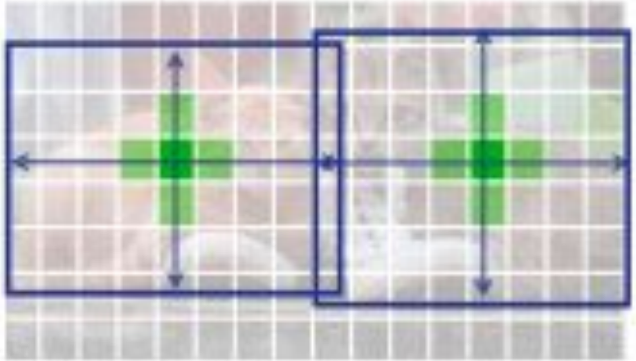


Fig 1. Three heads are predicted after one forward pass from the network architecture. 1)Offset Head. 2) Heatmap Head. 3) Dimension Head. Here Some Architecture(FCN) referees to any of the feature extractors which we want to use(Specified heads are for object detection).

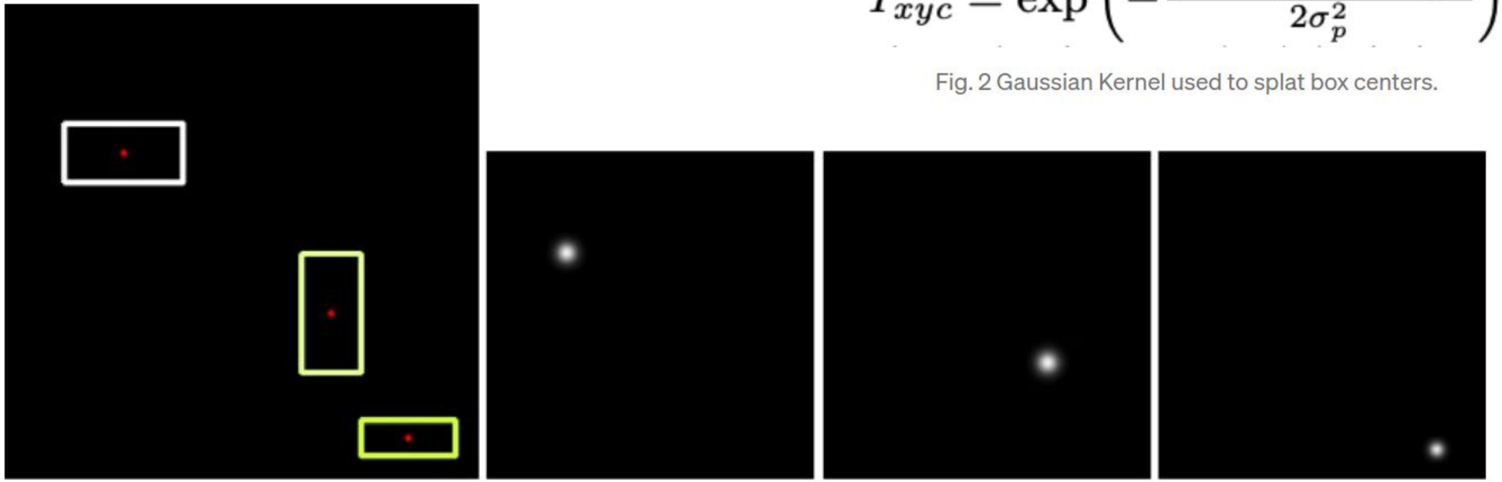
Positives assignment

- Each ground truth box center is interpreted to a center in the strided heatmap, surrounded by a gaussian gradient.
- Size of gradient depends on the ground-truth box size.
- Different heatmaps for each class.



$$Y_{xyc} = \exp\left(-\frac{(x-\tilde{p}_x)^2+(y-\tilde{p}_y)^2}{2\sigma_p^2}\right)$$

Fig. 2 Gaussian Kernel used to splat box centers.



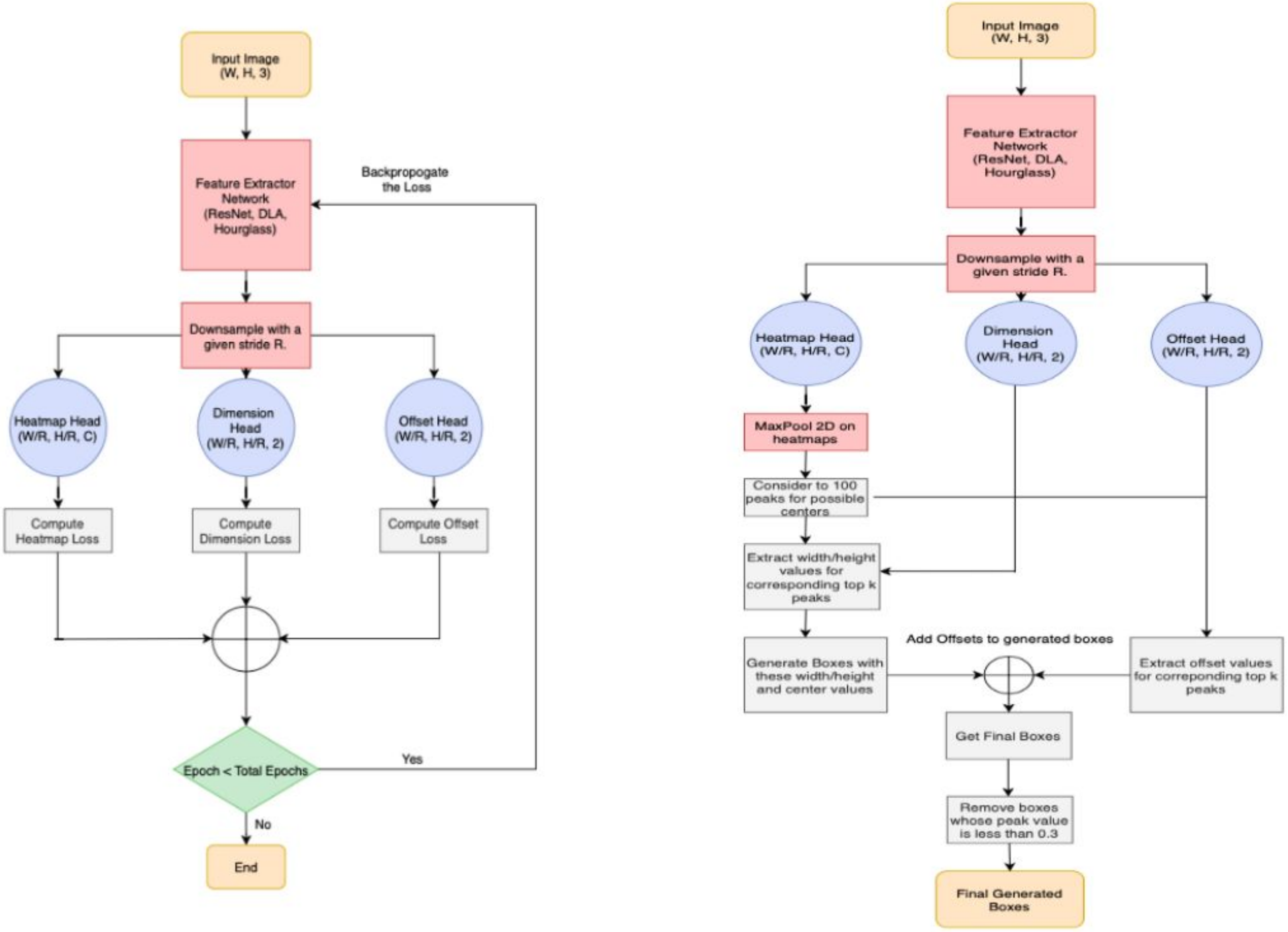


Fig. 4 (Left) Training, (Right) Inference flowchart of the proposed Object Detection algorithm

Losses

1. Heatmap variant Focal loss:
   - Add negative weights.
2. Offset localization loss:
   - Positive centers
3. Object sizes loss:

$$L_k = \frac{-1}{N} \sum_{xyc} \begin{cases} (1 - \hat{Y}_{xyc})^{\alpha} \log(\hat{Y}_{xyc}) & \text{if } Y_{xyc} = 1 \\ \boxed{(1 - Y_{xyc})^{\beta}} (\hat{Y}_{xyc})^{\alpha} & \text{otherwise} \\ \log(1 - \hat{Y}_{xyc}) & \end{cases}$$

$$L_{off} = \frac{1}{N} \sum_{p} \left| \hat{O}_{\tilde{p}} - \left(\frac{p}{R} - \tilde{p} \right) \right|. \qquad \hat{O} \in \mathcal{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$$

$$L_{size} = \frac{1}{N} \sum_{k=1}^{N} \left| \hat{S}_{p_k} - s_k \right|. \qquad \hat{S} \in \mathcal{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$$

$$L_{det} = L_k + \lambda_{size} L_{size} + \lambda_{off} L_{off}.$$

$$\lambda_{size} = 0.1 \text{ and } \lambda_{off} = 1$$

Human Pose estimation - keypoints

- These strided maps with sizes {W/R, H/R} are regressed additionally for k points:
  - k x 2 for x,y offsets from object center point.
  - k heatmaps for keypoints centers.
  - k x 2 x,y offsets from k heatmaps centers.

Fig. 11 **(Left) Keypoint Center Offsets. (Middle) Keypoint Heatmaps. (Right) Offsets of keypoints[1]**

Why Anchor free?

- Avoid all hyper-parameters related to anchor boxes, which are very sensitive to the final performance. (sizes, aspect ratios and number of anchors)
- Anchor based method has excessive number of samples (k times anchor-free centers), most of them are background samples which aggravates the imbalance between positive and negative samples.
- Eliminating pre-defined anchor boxes, adjust well to different input sizes.