

FoveaBox: Beyond Anchor Based Object Detector

Presented by: Tao Kong

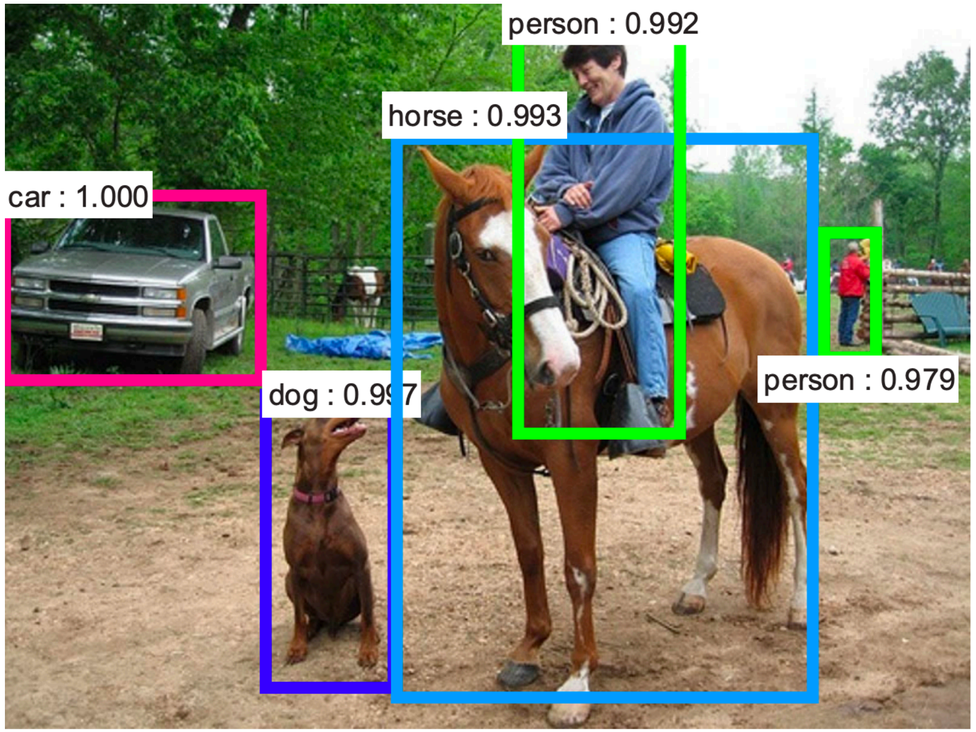
Kong, Tao, Fuchun Sun, Huaping Liu, Yuning Jiang, and Jianbo Shi. "FoveaBox: Beyond Anchor-based Object Detector." *arXiv preprint arXiv:1904.03797* (2019)

Highlights

- FoveaBox: a completely anchor-free object detector.
- More accurate and flexible compared with sliding-window mechanism.
- Great robustness and generalization ability to the box shape distributions

Object detection

- Two main tasks:
- **Recognition**: whether there are any instances of semantic objects from predefined categories
- **Localization**: the spatial location and extent (bounding box)



Object recognition

pixels → classifier → “bus”?

Pixels

→ edges → classifier → “bus”?

Edges

SIFT/HOG

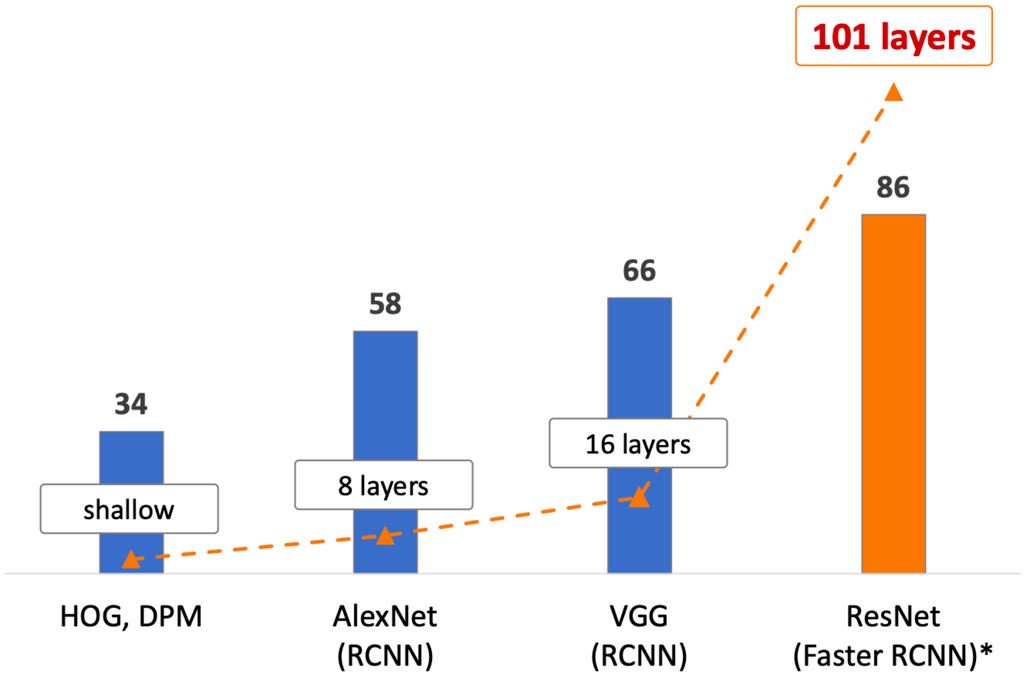
→ edges → histogram → classifier → “bus”?

→ edges → histogram → K-means sparse code FV/VLAD → classifier → “bus”?

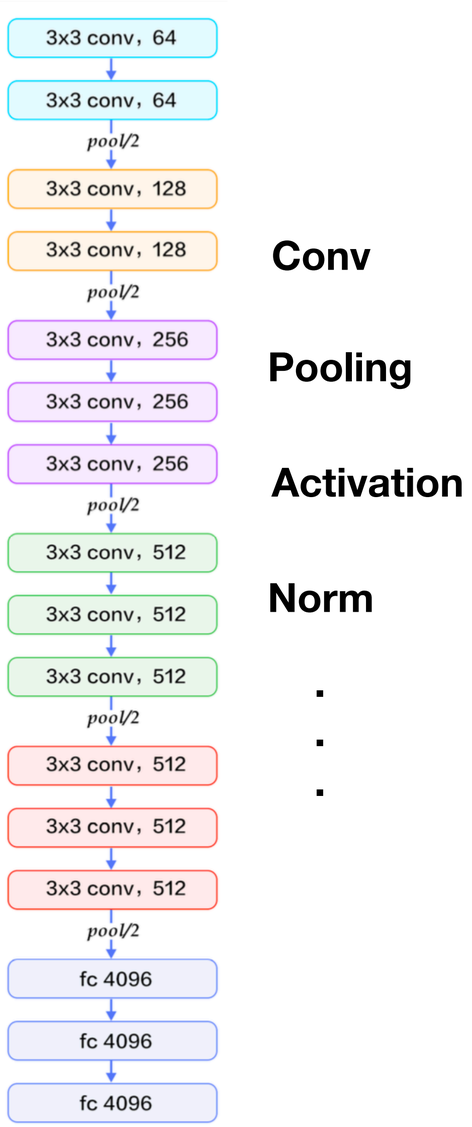
Components

Object recognition

ConvNet, Generic components/“layers”, deeper and deeper



PASCAL VOC 2007 **Object Detection** mAP (%)



Conv

Pooling

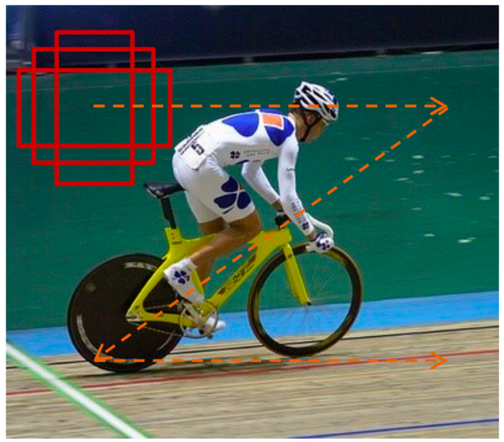
Activation

Norm

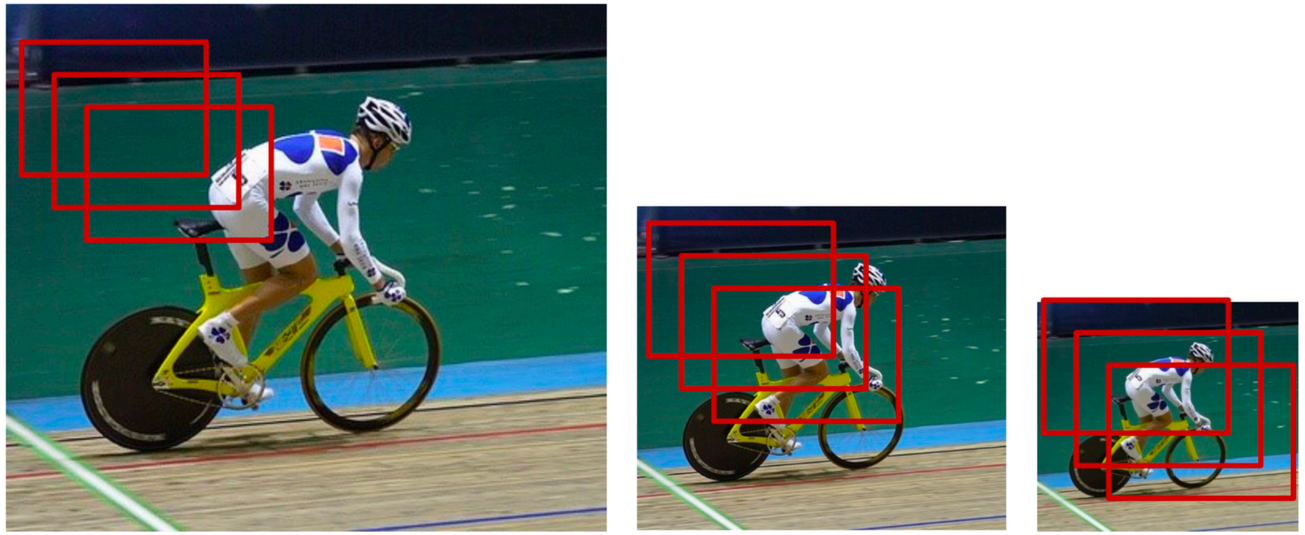
.
.
.

Object localization

- Sliding windows



on single scale image

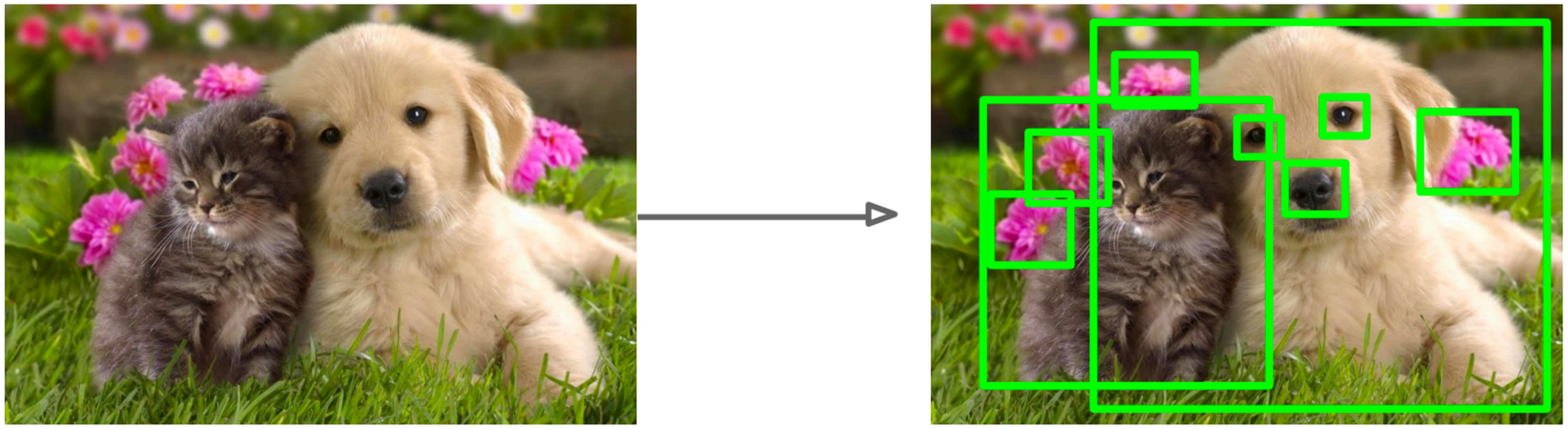


on image pyramid

Image size: W*H, window# n: cost = O(n*W*H)

Object localization

- Region proposals
  - Find image regions that are likely to contain objects
  - “class-agnostic” object detector
  - Color, edge, texture features



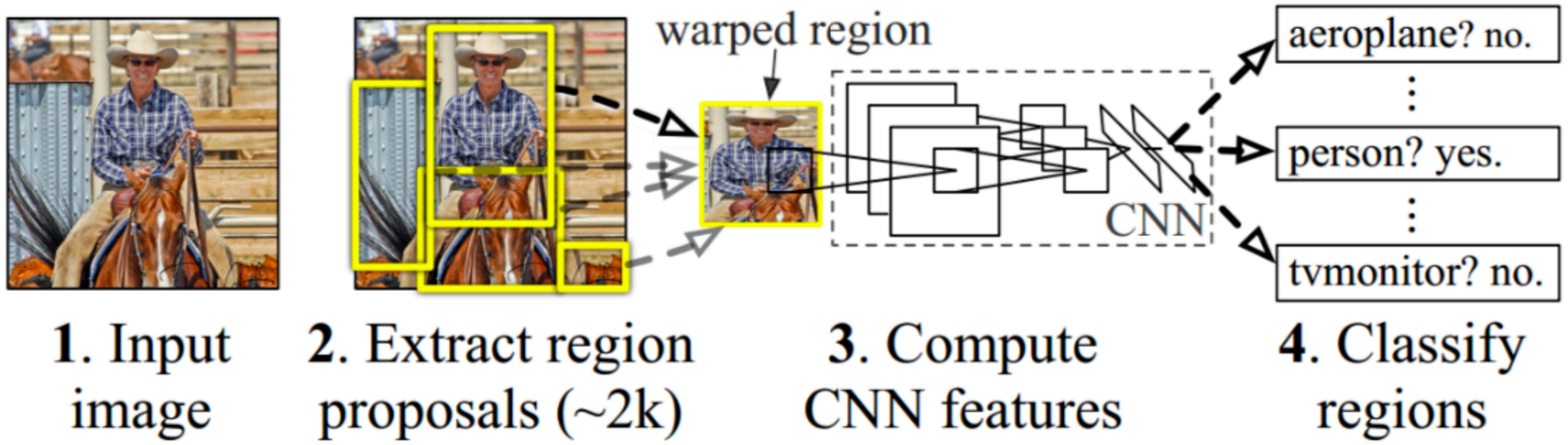
Object localization

- Region proposals

Method	Approach	Outputs Segments	Outputs Score	Control #proposals	Time (sec.)	Repea-tability	Recall Results	Detection Results
Bing [18]	Window scoring		✓	✓	0.2	⋆⋆⋆	⋆	·
CPMC [19]	Grouping	✓	✓	✓	250	-	⋆⋆	⋆
EdgeBoxes [20]	Window scoring		✓	✓	0.3	⋆⋆	⋆⋆⋆	⋆⋆⋆
Endres [21]	Grouping	✓	✓	✓	100	-	⋆⋆⋆	⋆⋆
Geodesic [22]	Grouping	✓		✓	1	⋆	⋆⋆⋆	⋆⋆
MCG [23]	Grouping	✓	✓	✓	30	⋆	⋆⋆⋆	⋆⋆⋆
Objectness [24]	Window scoring		✓	✓	3	·	⋆	·
Rahtu [25]	Window scoring		✓	✓	3	·	·	⋆
RandomizedPrim's [26]	Grouping	✓		✓	1	⋆	⋆	⋆⋆
Rantalankila [27]	Grouping	✓		✓	10	⋆⋆	·	⋆⋆
Rigor [28]	Grouping	✓		✓	10	⋆	⋆⋆	⋆⋆
SelectiveSearch [29]	Grouping	✓	✓	✓	10	⋆⋆	⋆⋆⋆	⋆⋆⋆
Gaussian				✓	0	·	·	⋆
SlidingWindow				✓	0	⋆⋆⋆	·	·
Superpixels		✓			1	⋆	·	·
Uniform				✓	0	·	·	·

Object localization

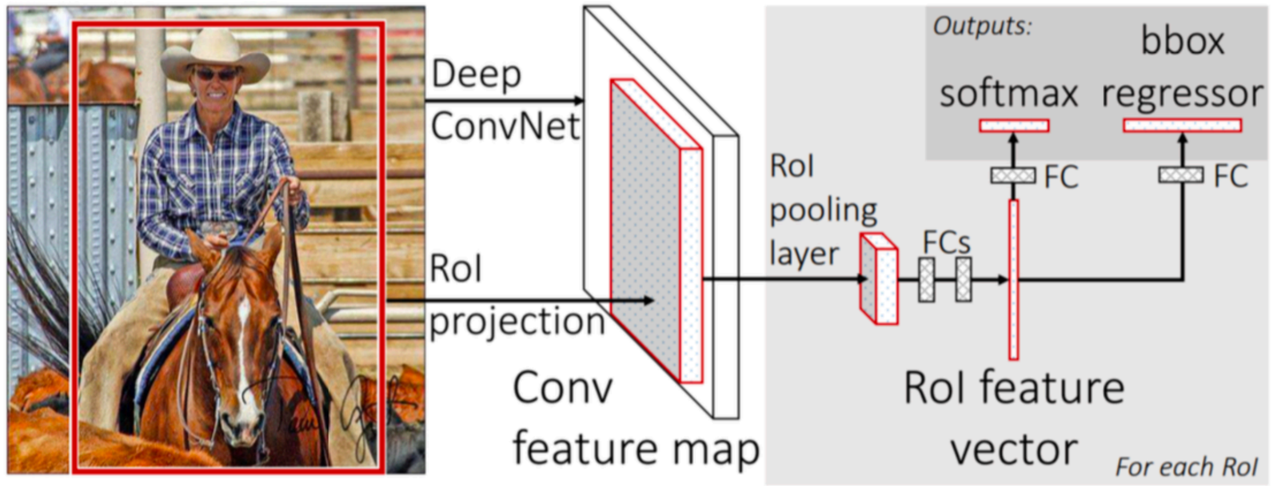
- R-CNN: Region based ConvNet



Siding windows on the input image

Object localization

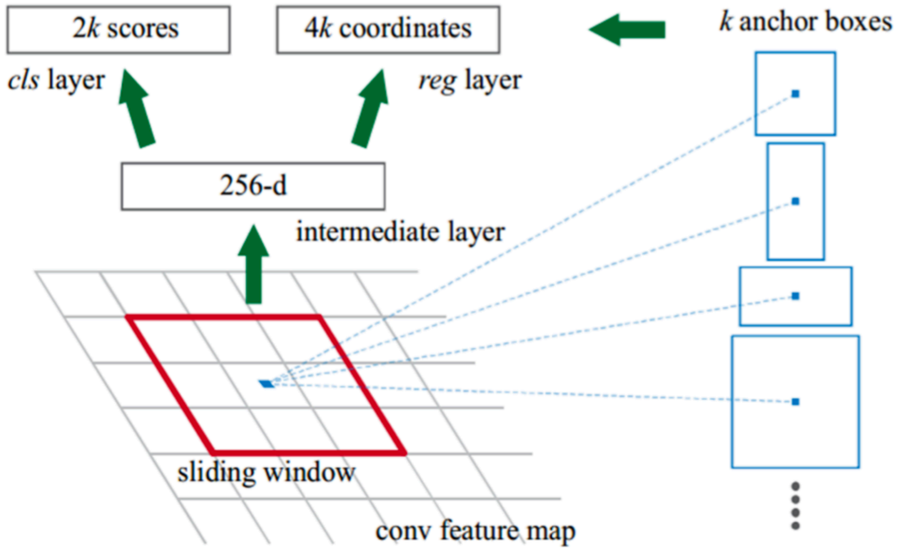
- Fast R-CNN

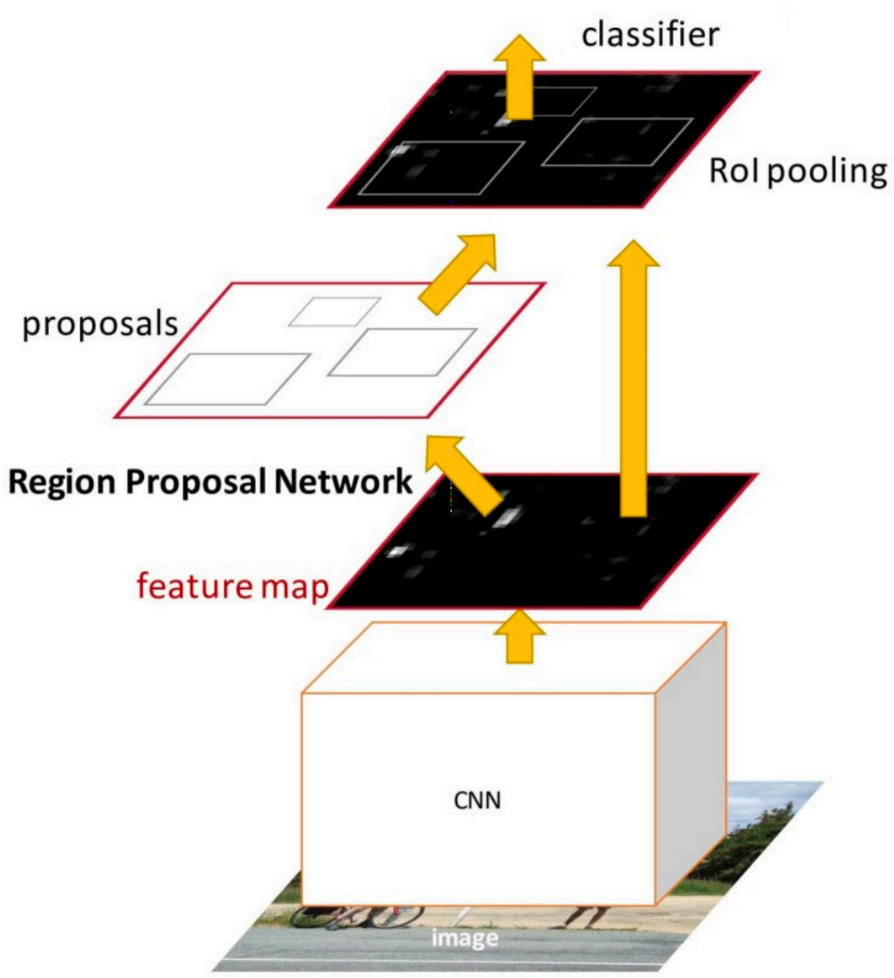


Siding windows on the feature maps

Object localization

- Faster R-CNN

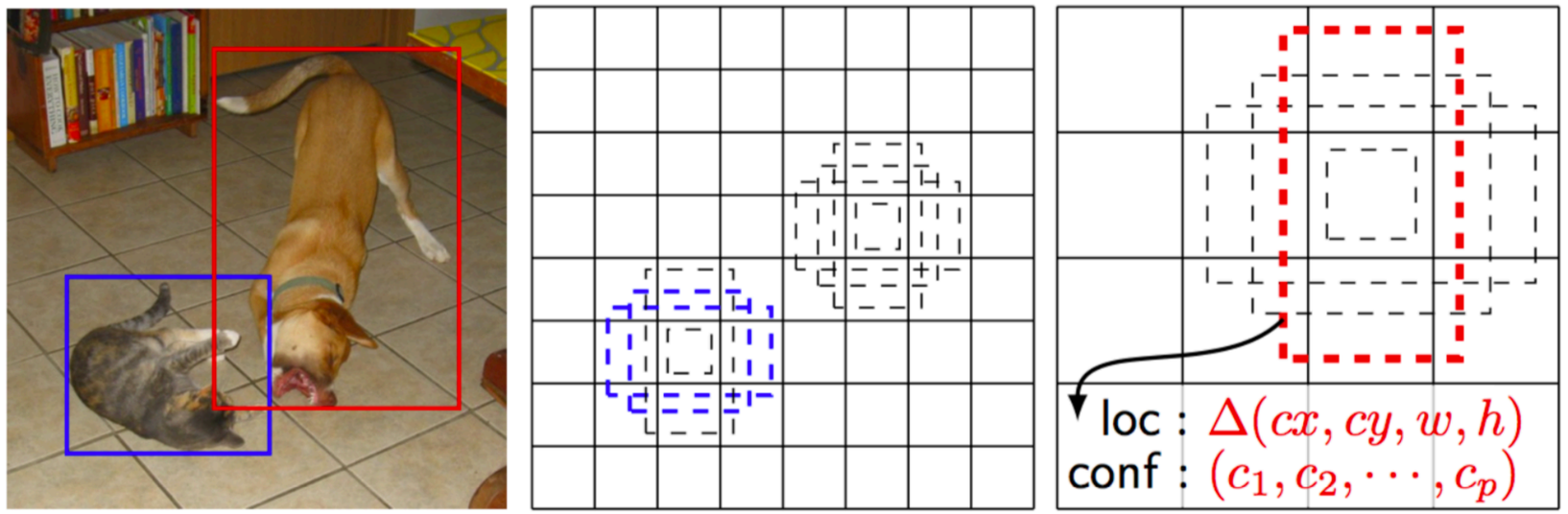




Siding anchors on the feature maps

Object localization

- SSD/RetinaNet/FPN/Mask R-CNN…

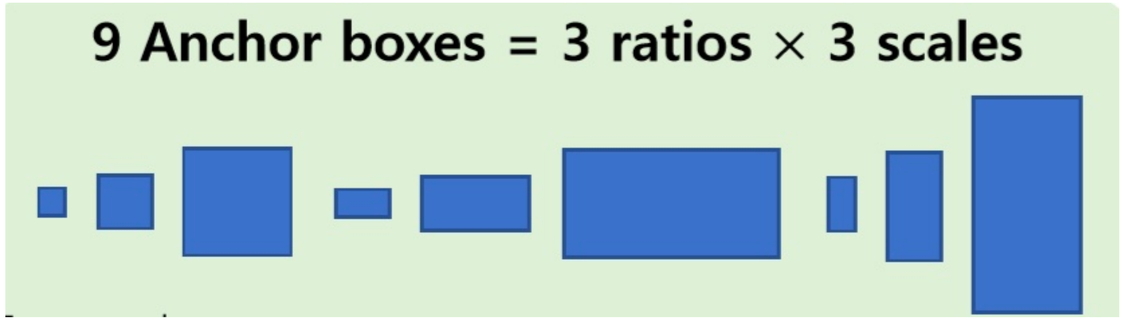


Siding anchors on the feature pyramid

Anchor-based detector

- Anchors are regression references and classification candidates to predict proposals for two-stage detectors (Faster/Cascade R-CNN/Mask R-CNN)

- Anchors are final box reference for one-stage detectors (SSD/YOLOv2/ YOLOv3/RetinaNet)



Anchor-based detector

MS COCO object detection leaderboard

	AP	AP^{50}	AP^{75}	AP^{S}	AP^{M}	AP^{L}	AR^{1}	AR^{10}	AR^{100}	AR^{S}	AR^{M}	AR^{L}	date
Megvii (Face++)	0.525	0.729	0.587	0.345	0.556	0.649	0.393	0.645	0.690	0.513	0.727	0.824	2017-10-05
UCenter	0.510	0.706	0.557	0.326	0.540	0.640	0.395	0.640	0.679	0.490	0.721	0.827	2017-10-05
MSRA	0.504	0.715	0.563	0.337	0.529	0.623	0.379	0.637	0.690	0.518	0.723	0.826	2017-10-05
FAIR Mask R-CNN	0.503	0.719	0.558	0.326	0.537	0.624	0.382	0.622	0.661	0.477	0.708	0.799	2017-10-05
bharat_umd	0.481	0.695	0.535	0.311	0.515	0.601	0.366	0.604	0.648	0.457	0.697	0.790	2017-10-05

All top object detection entries are using anchor-based detectors!

Anchor drawbacks

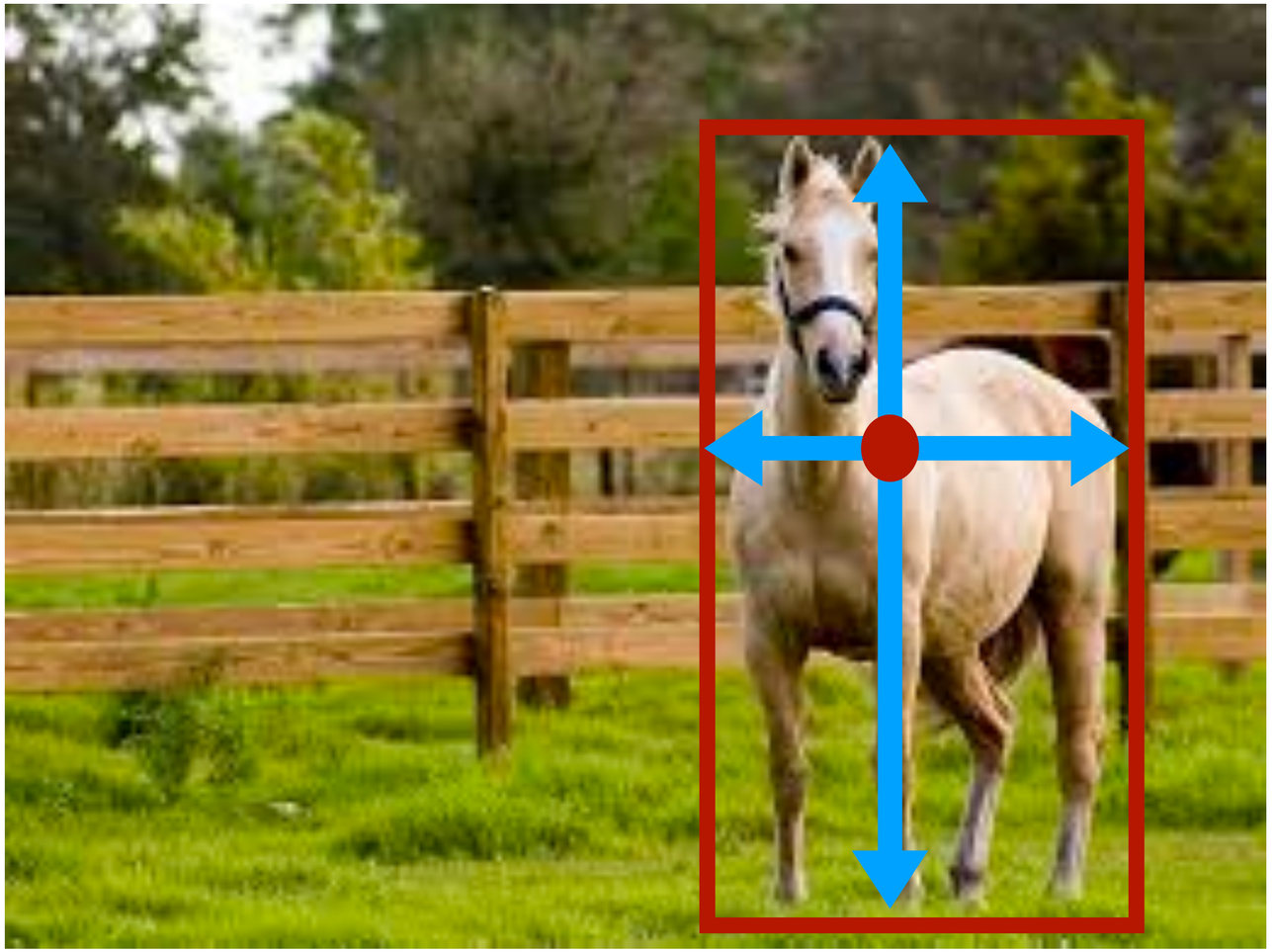
- Anchor boxes introduce additional hyper-parameters of design choices.—
  - Scales, aspect ratios, density, not flexible
- To achieve a high recall rate, the anchors are carefully designed based on the statistics computed from the training/validation set.
- One design choice based on a particular dataset is not always applicable to other applications, which harms the generality.
- Dense object detectors usually rely on effective techniques to deal with the foreground-background class imbalance challenge.

How we human do such task?

How we human do such task?

Coarse position

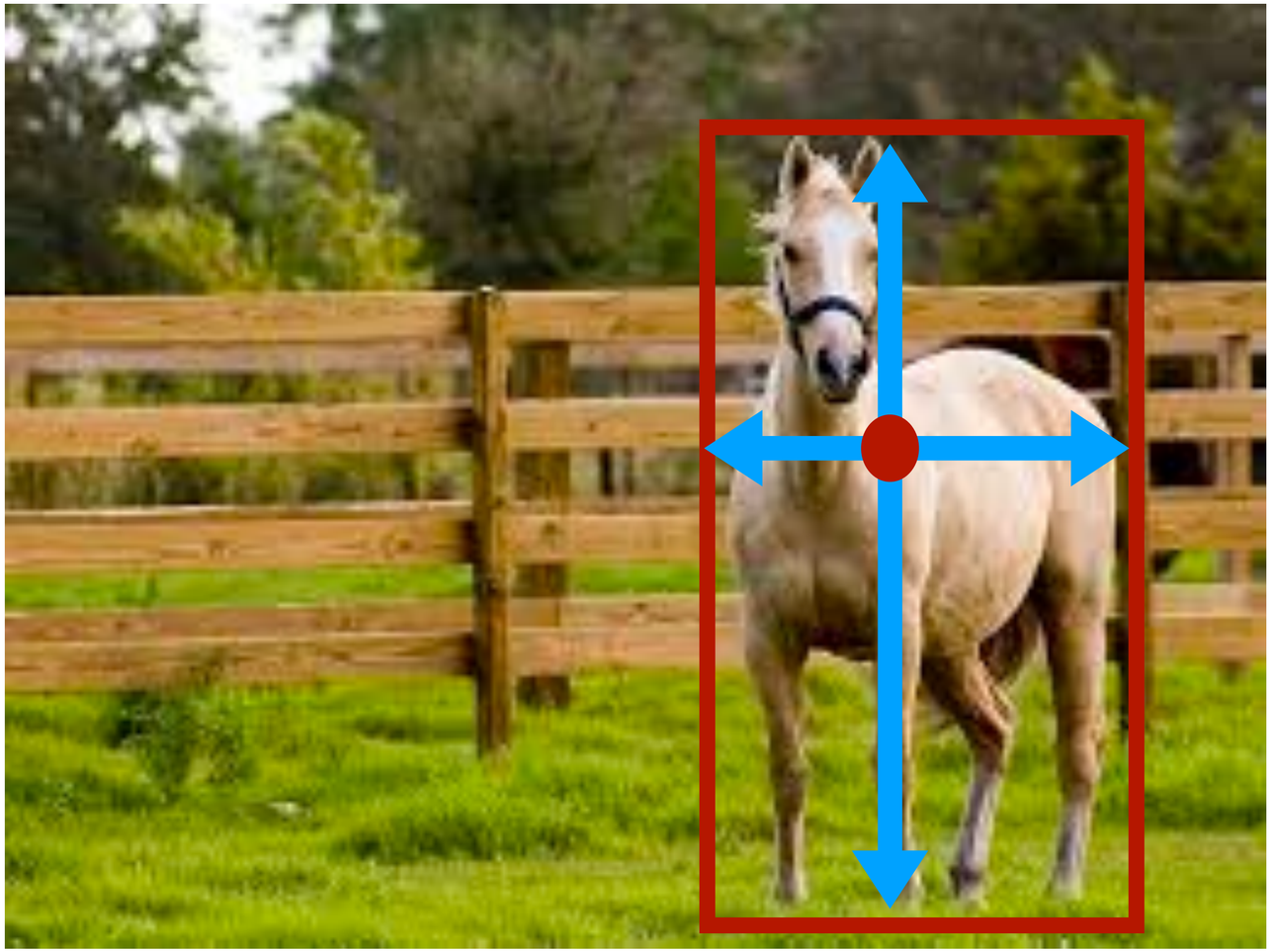
How we human do such task?

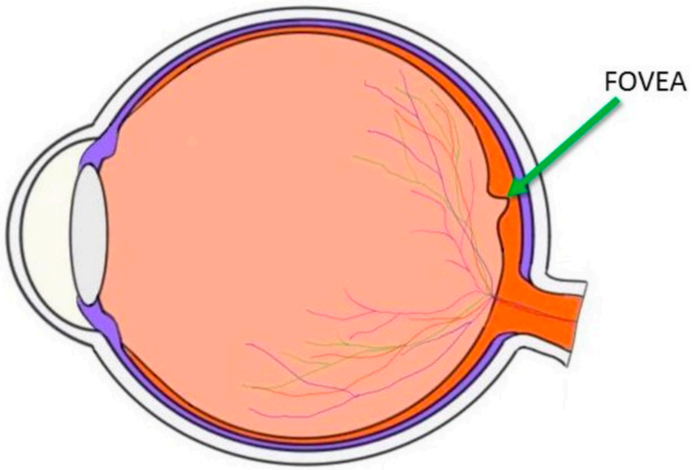


Coarse position

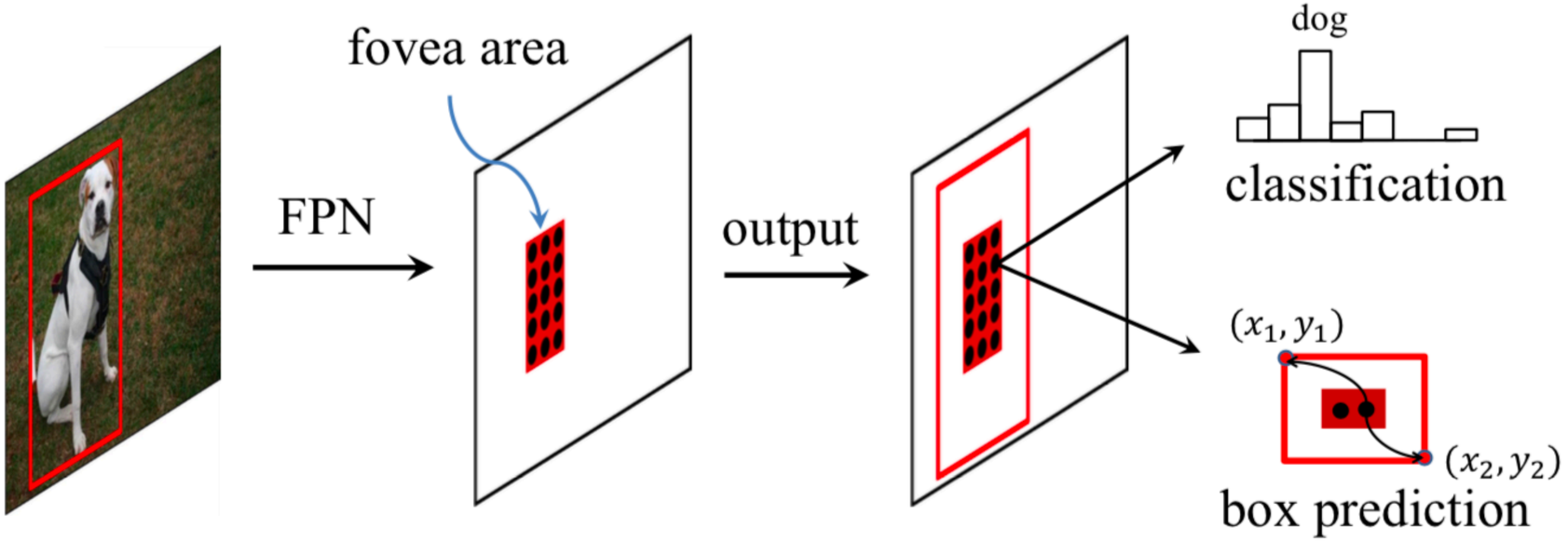
Fine boundary

FoveaBox





FoveaBox

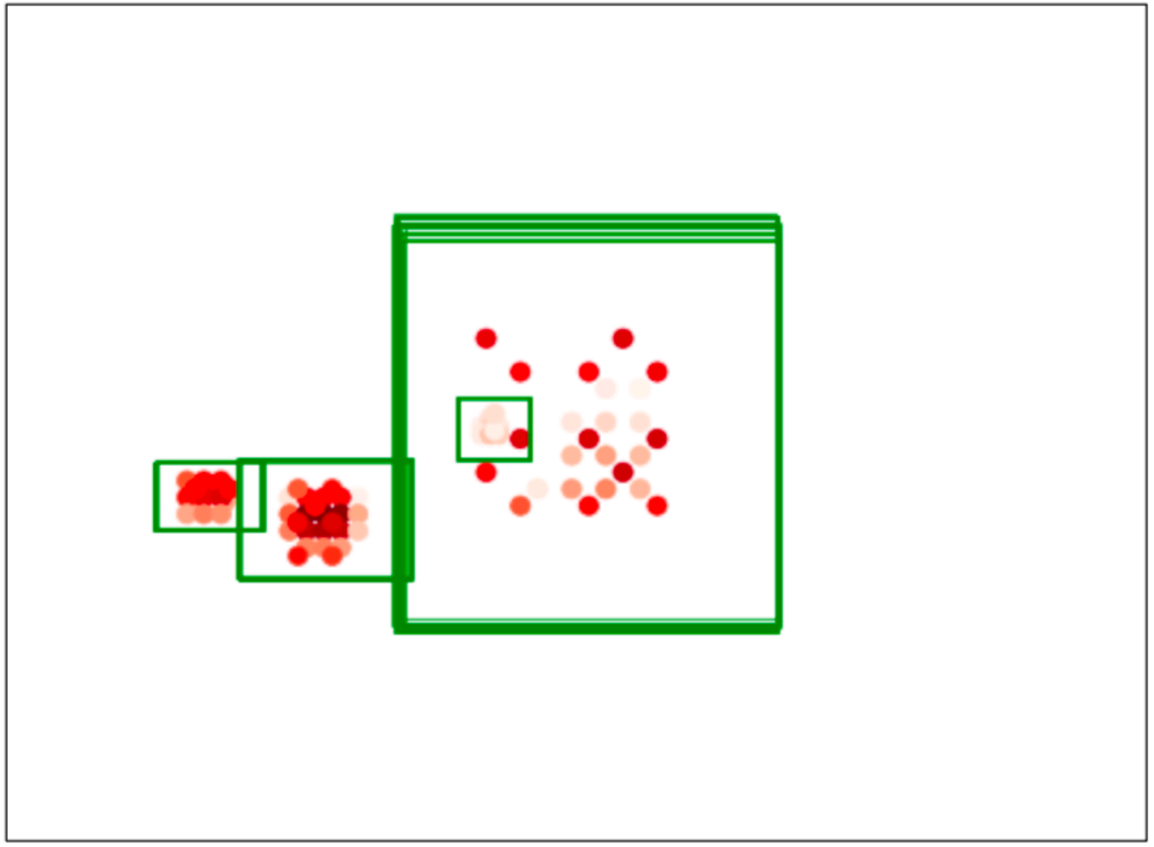


Coarse position

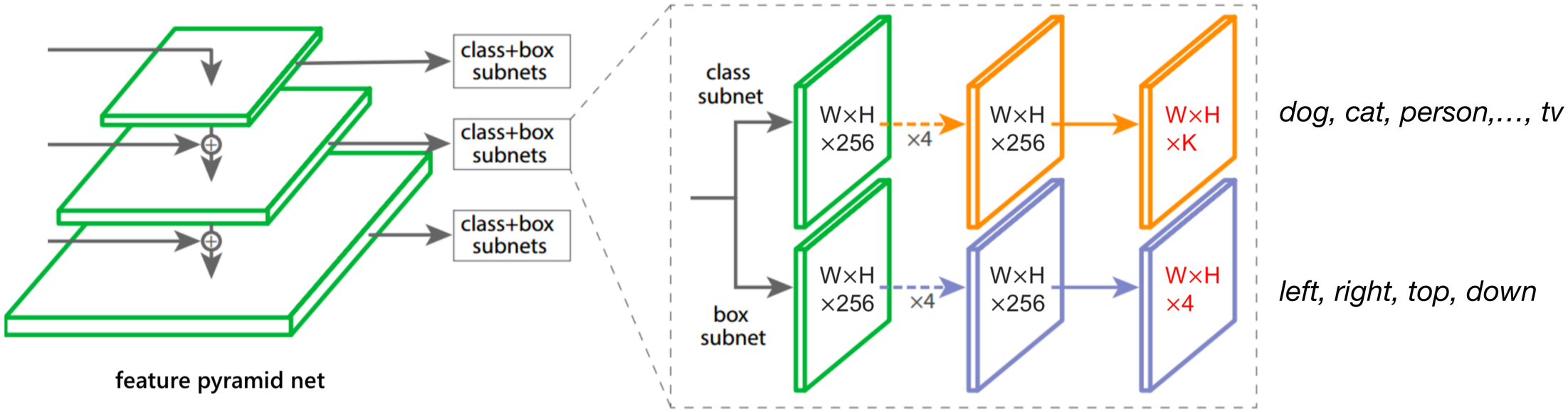
Fine boundary

FoveaBox


bus 0.80
person 0.57
car 0.78
car 0.85
GEYLANG LOR 1 via BENDEMEER ROAD
985
MAN



FoveaBox: FPN backbone



- Top-down architecture with lateral connections
- Each level is used for detecting objects at a different scale.
- For each output position: (a) object existing possibility and (b) bounding box

FoveaBox: Scale assignment

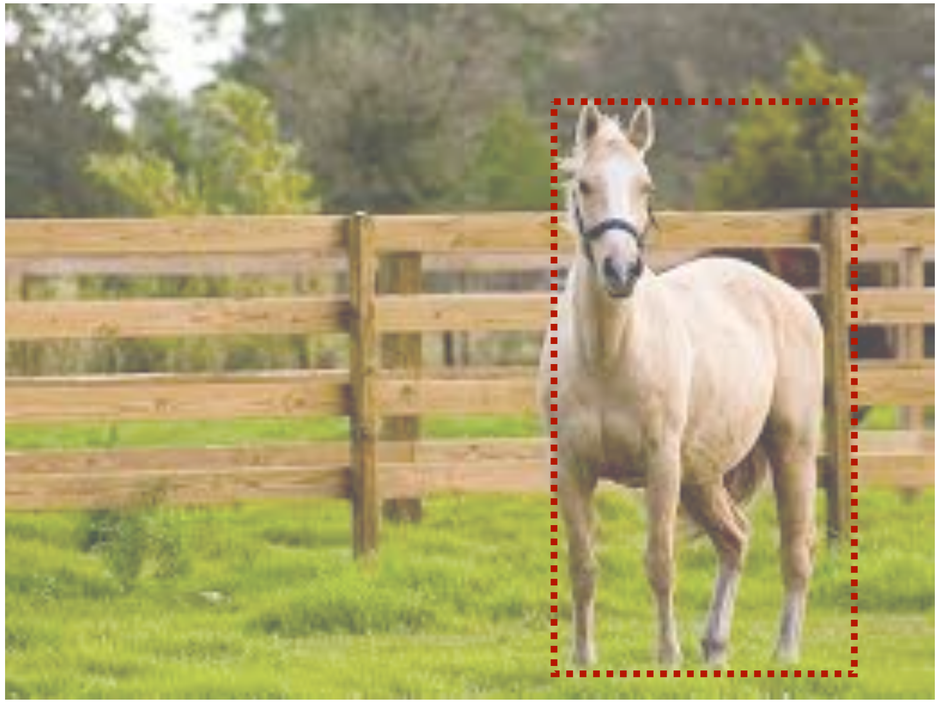
- Divide the scales of objects into several bins
- Each pyramid level is used for detecting the objects at a different scale.
- Basic area for level l:

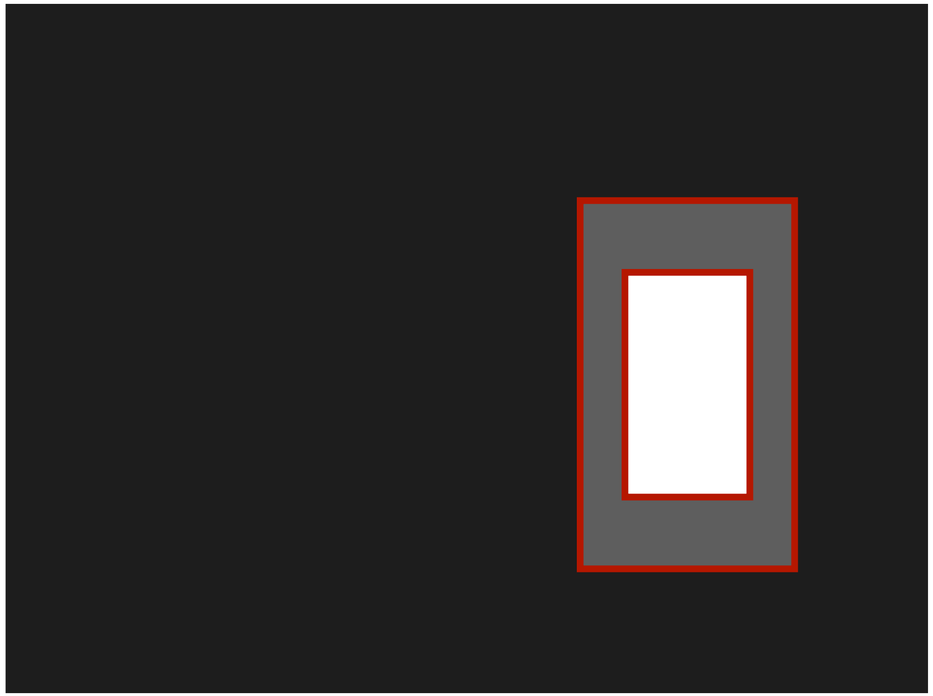
$$S_l = 4^l \cdot S_0$$

- And the the valid scale range of the target boxes for pyramid level l is computed as

$$[S_l/\eta^2, S_l \cdot \eta^2]$$

FoveaBox: Object Fovea





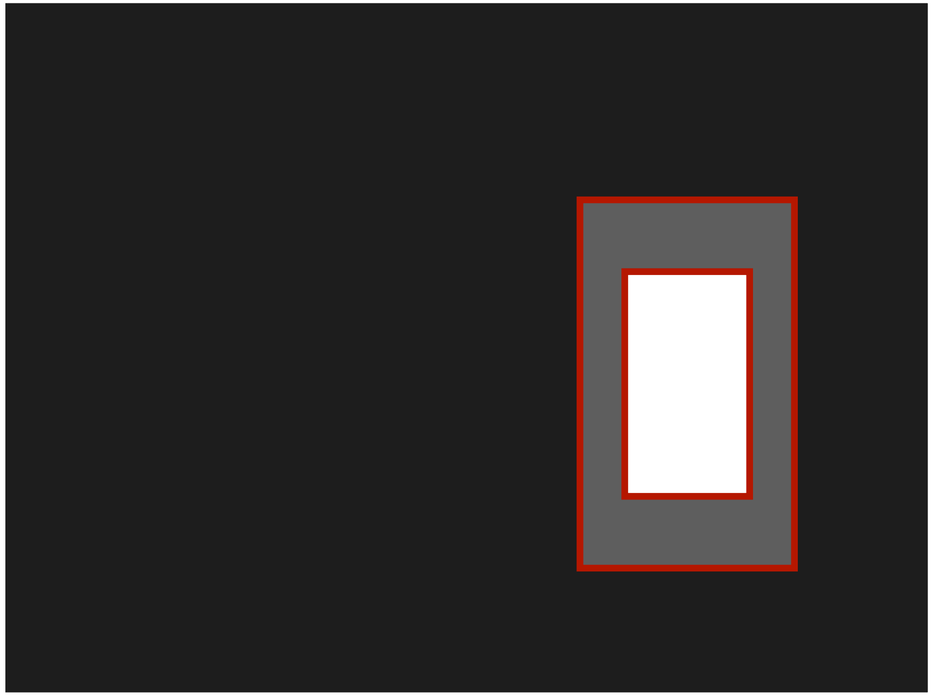
GT

Negative

Ignore

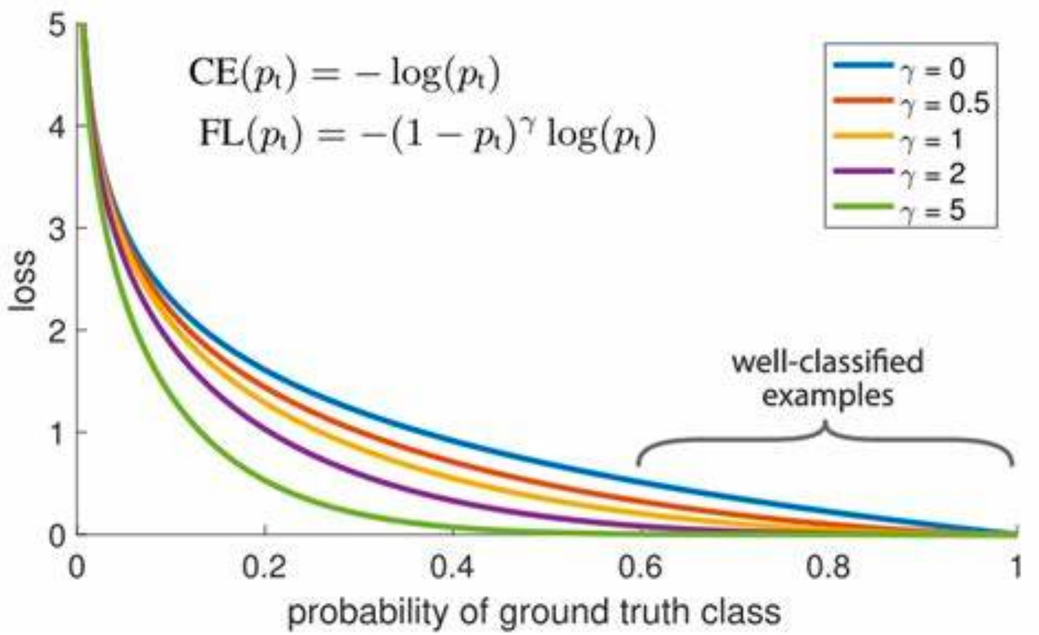
Positive

FoveaBox: Object Fovea

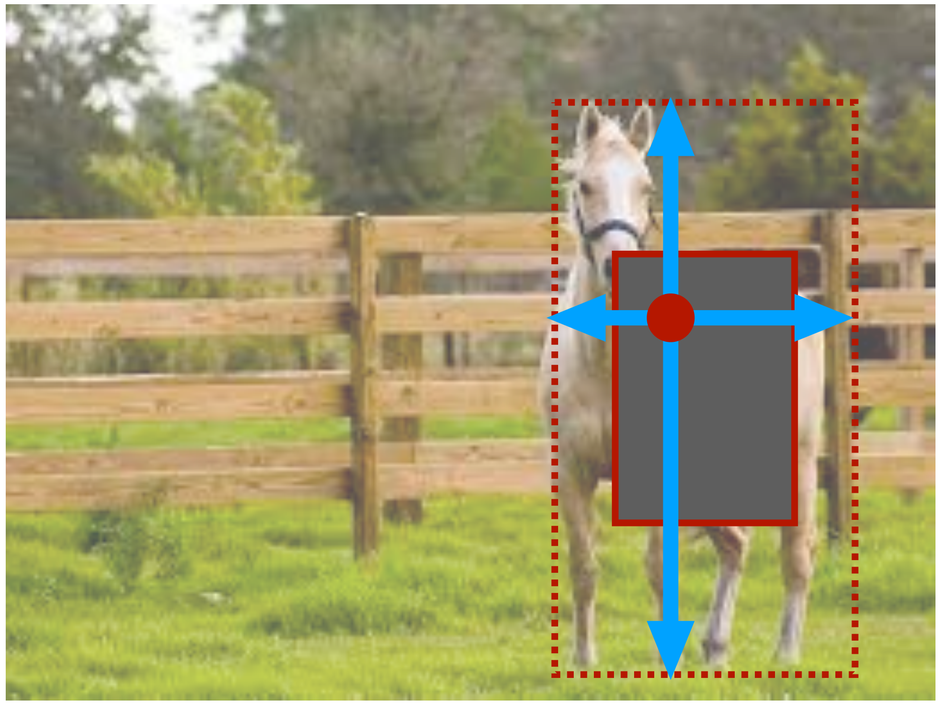


Negative Ignore Positive

Focal Loss



FoveaBox: Box Prediction



GT: (x_1, y_1, x_2, y_2)

Targets:

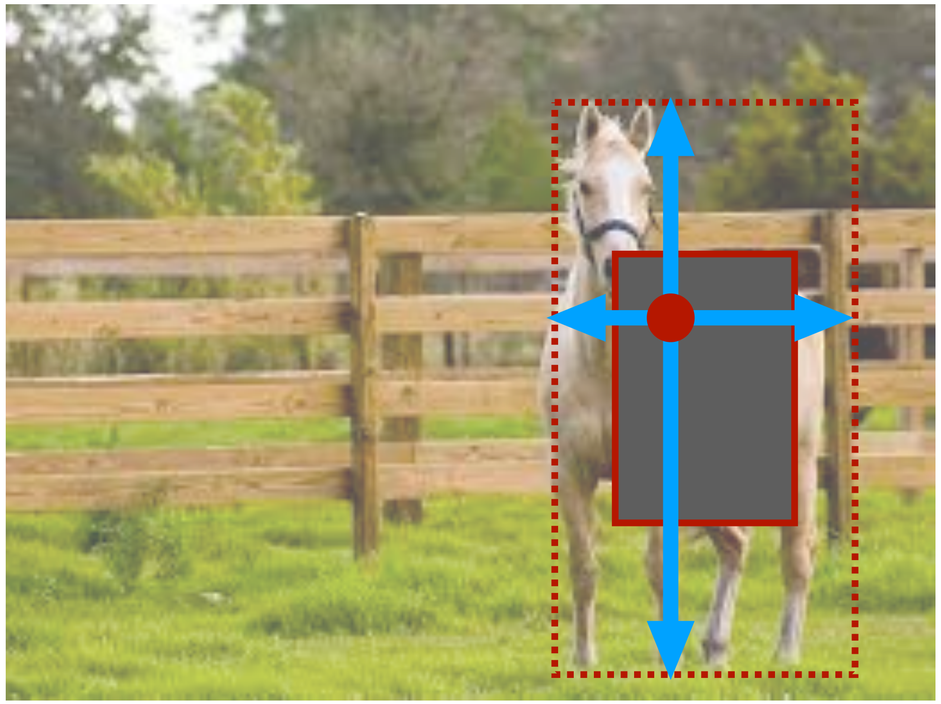
$$t_{x_1} = \log \frac{2^l(x + 0.5) - x_1}{z}, \quad \text{left}$$

$$t_{y_1} = \log \frac{2^l(y + 0.5) - y_1}{z}, \quad \text{top}$$

$$t_{x_2} = \log \frac{x_2 - 2^l(x + 0.5)}{z}, \quad \text{right}$$

$$t_{y_2} = \log \frac{y_2 - 2^l(y + 0.5)}{z}, \quad \text{bottom}$$

FoveaBox: Box Prediction



Smooth L1 Loss:

$$L_{\text{loc}}(t^u, v) = \sum_{i \in (x_1, y_1, x_2, y_2)} \text{smooth}_{L_1}(t_i^u - v_i),$$

in which

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise,} \end{cases}$$

Other loc loss: IoU loss, GIoU loss

Train-inference

- MS COCO dataset,
- Training: on *trainval35k* subset (115k)
  - Synchronized SGD over 4 GPUs, 2 images per GPU.
  - Standard training techniques following *Detection*
- Inference: on *minival* (5k) or *test-dev subset* (20k)
  - Hard-NMS
  - No testing-time augmentation.

Experiments: anchor density

- 600 input scale
- FoveaBox vs RetinaNet
- #sc: scale number
- #ar: aspect ratio number

method	#sc	#ar	AP	AP_{50}	AP_{75}
RetinaNet	1	1	30.2	49.0	31.7
RetinaNet	2	1	31.9	50.0	34.1
RetinaNet	3	1	31.9	49.4	33.8
RetinaNet	1	3	32.4	52.4	33.9
RetinaNet	2	3	34.2	53.1	36.5
RetinaNet	3	3	34.2	53.2	36.9
RetinaNet	4	3	33.9	52.1	36.2
FoveaBox	n/a	n/a	**36.0**	**55.2**	**37.9**

Varying anchor density of RetinaNet [28] and FoveaBox

Experiments: robustness to box

- 600 input scale
- FoveaBox vs RetinaNet
- * aspect ratio jittering

method	AP	$AP_{u<3}$	$AP_{3\leq u<5}$	$AP_{u\geq 5}$
RetinaNet	34.2	36.5	24.5	10.2
RetinaNet*	35.3	**37.1**	25.3	16.3
FoveaBox	**36.0**	**37.1**	**27.4**	**18.4**

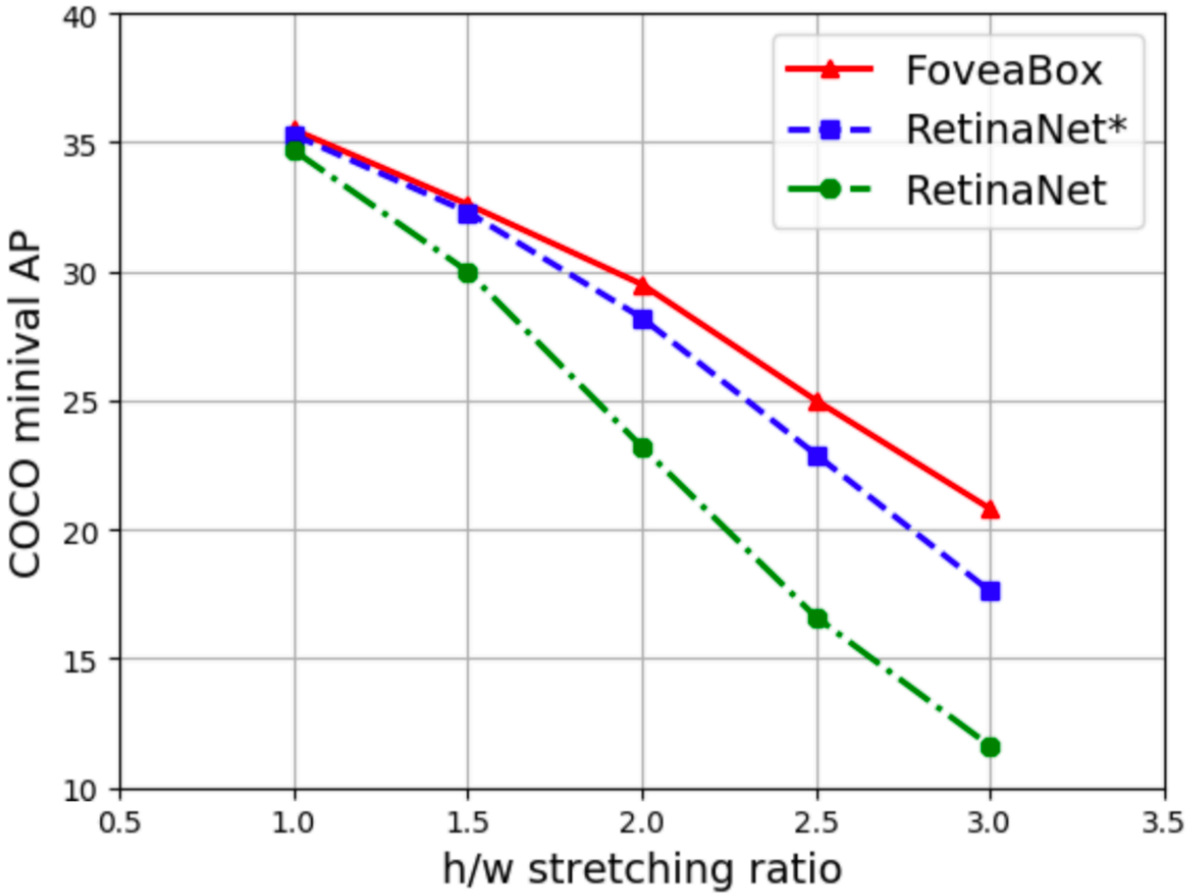
(b) **Detection performance with different aspect ratios**

Experiments: robustness to box

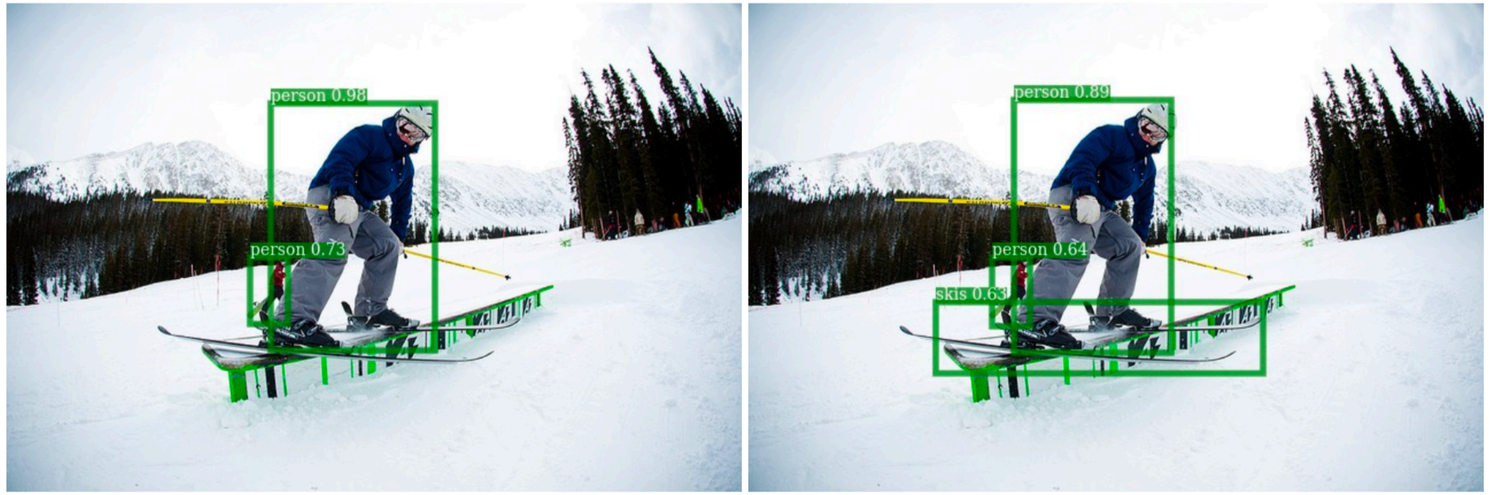
- 600 input scale
- FoveaBox vs RetinaNet
- Stretching the image

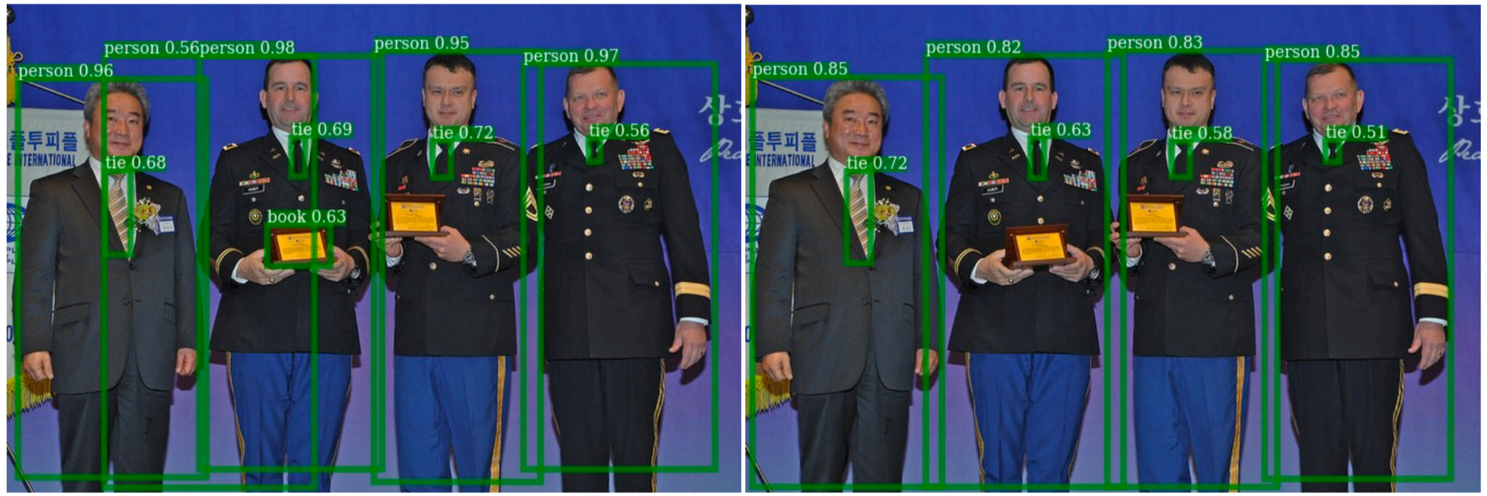
Experiments: robustness to box

- 600 input scale
- FoveaBox vs RetinaNet
- * aspect ratio jittering
- Stretching the image



Experiments: Qualitative result





Experiments: Across model depth and scale

	depth	scale	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
RetinaNet	50	400	30.5	47.8	32.7	11.2	33.8	46.1
FoveaBox	50	400	31.9	49.3	33.8	12.7	36.1	48.7
RetinaNet	50	600	34.2	53.2	36.9	16.2	37.4	47.4
FoveaBox	50	600	36.0	55.2	37.9	18.6	39.4	50.5
RetinaNet	50	800	35.7	55.0	38.5	18.9	38.9	46.3
FoveaBox	50	800	37.1	57.2	39.5	21.6	41.4	49.1
RetinaNet	101	400	31.9	49.5	34.1	11.6	35.8	48.5
FoveaBox	101	400	33.3	51.0	35.0	12.9	38.0	51.3
RetinaNet	101	600	36.0	55.2	38.7	17.4	39.6	49.7
FoveaBox	101	600	38.0	57.8	40.2	19.5	42.2	52.7
RetinaNet	101	800	37.8	57.5	40.8	20.2	41.1	49.2
FoveaBox	101	800	38.9	58.4	41.5	22.3	43.5	51.7

QA