TensorFlow Object Detection API에서 제공하는 다양한 Object Detection을 위한 최신 모델들

- TensorFlow Object Detection API는 다음과 같은 최신 Object Detection 모델의 다양한 backbone을 이용한 구현을 제공합니다.

① Faster R-CNN

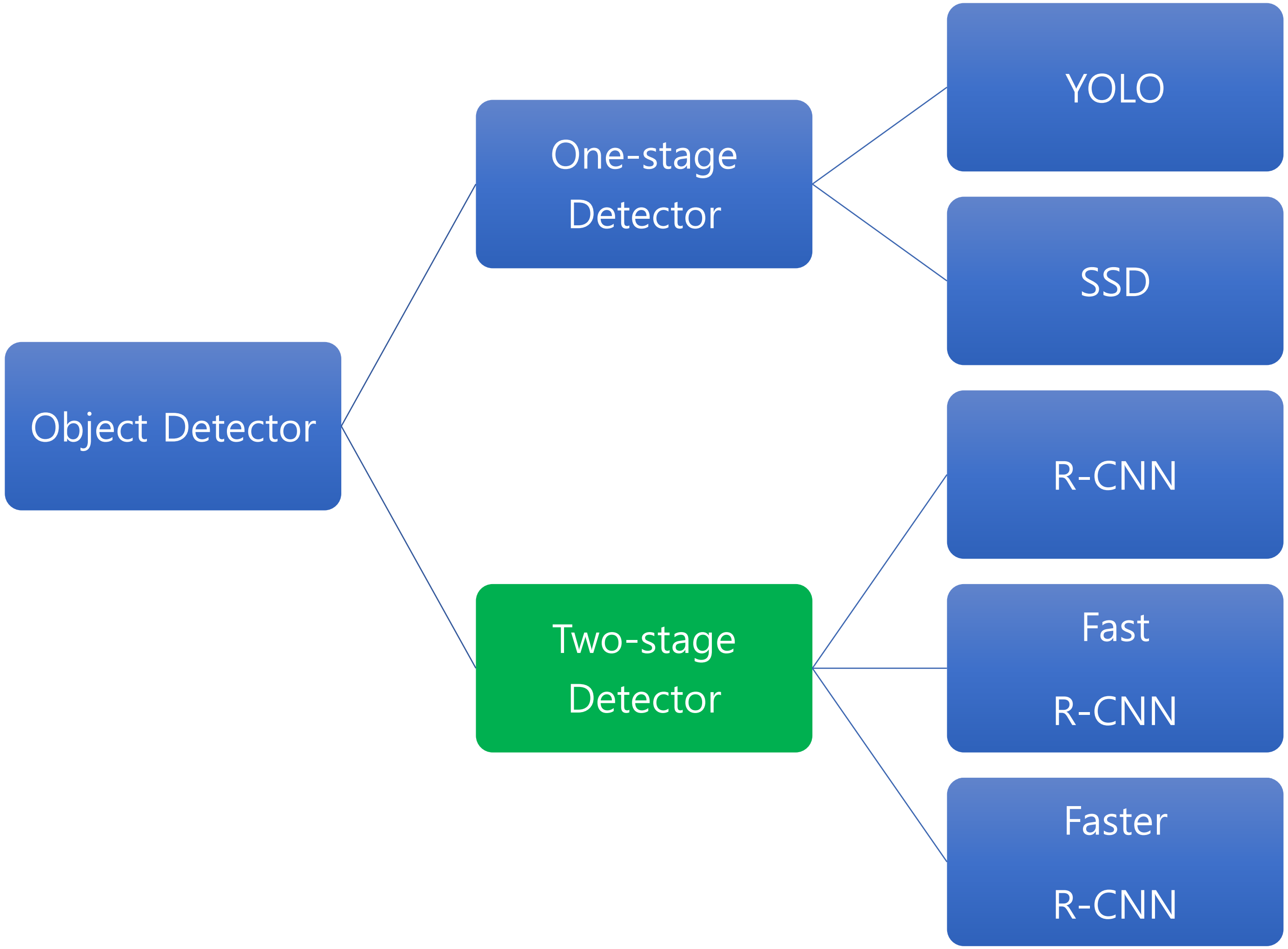
② SSD(Single Shot Multi-box Detector)

③ RetinaNet

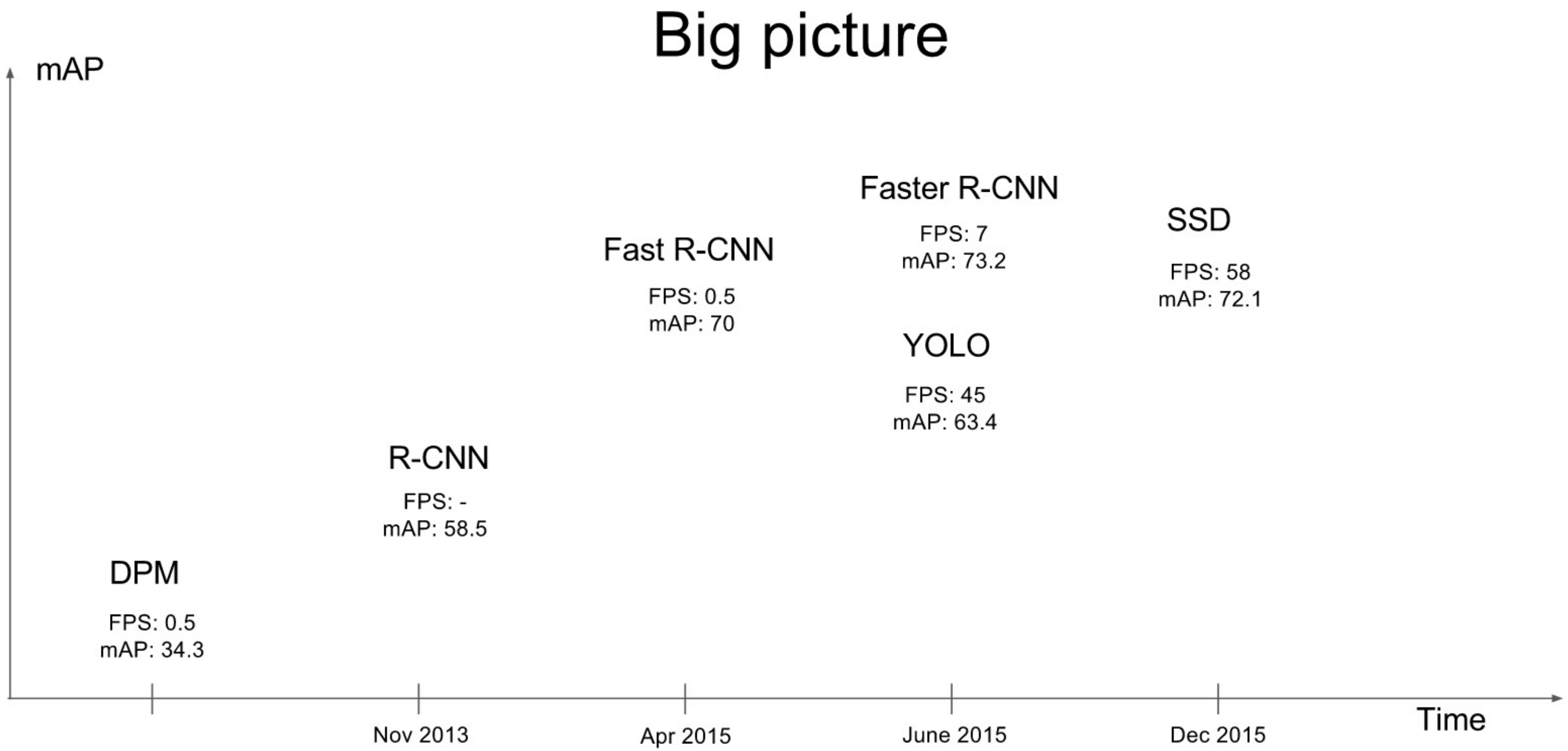
④ CenterNet

⑤ EfficientDet

One-stage Detector vs Two-Stage Detector



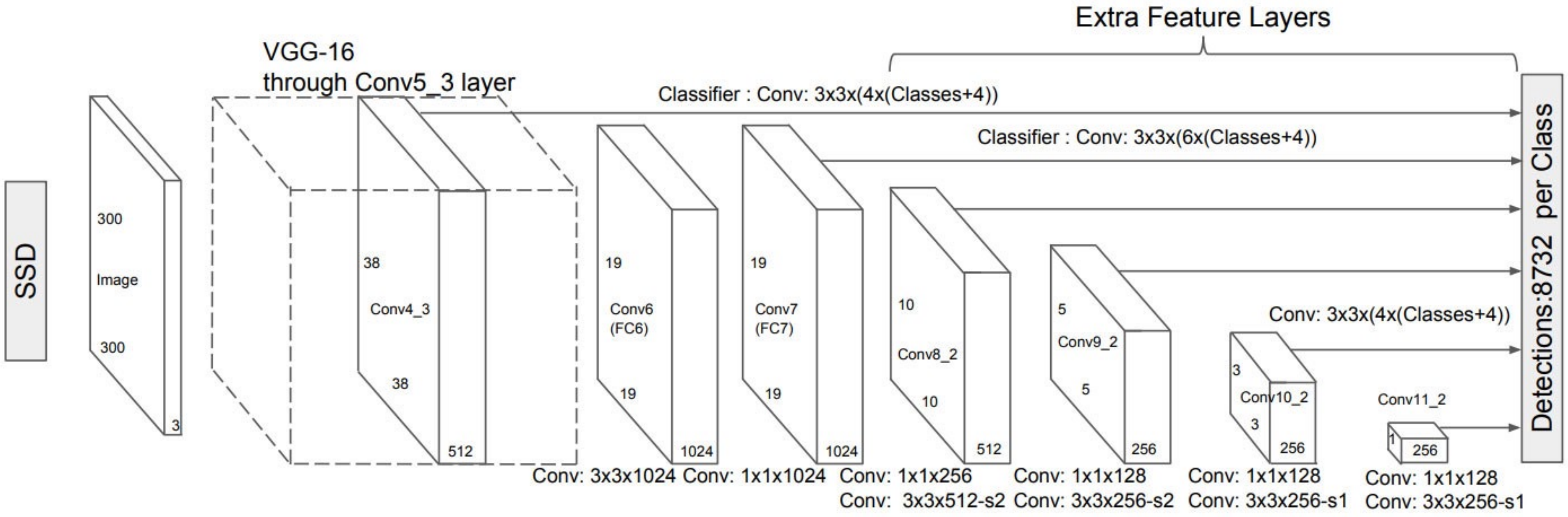
SSD



Slide credit: CS231n
(https://goo.gl/QSA36t)

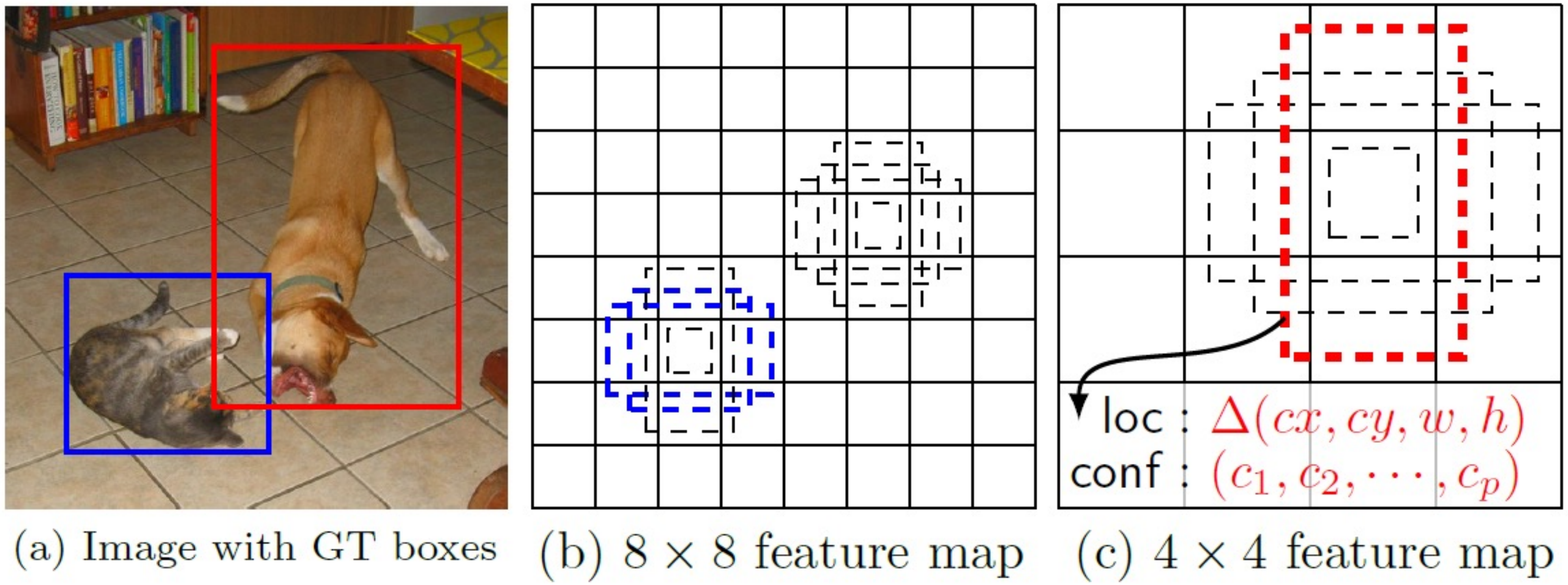
Ssd: Single shot multibox detector

- Liu, Wei, et al. "Ssd: Single shot multibox detector." European conference on computer vision. Springer, Cham, 2016.
- https://arxiv.org/pdf/1512.02325.pdf



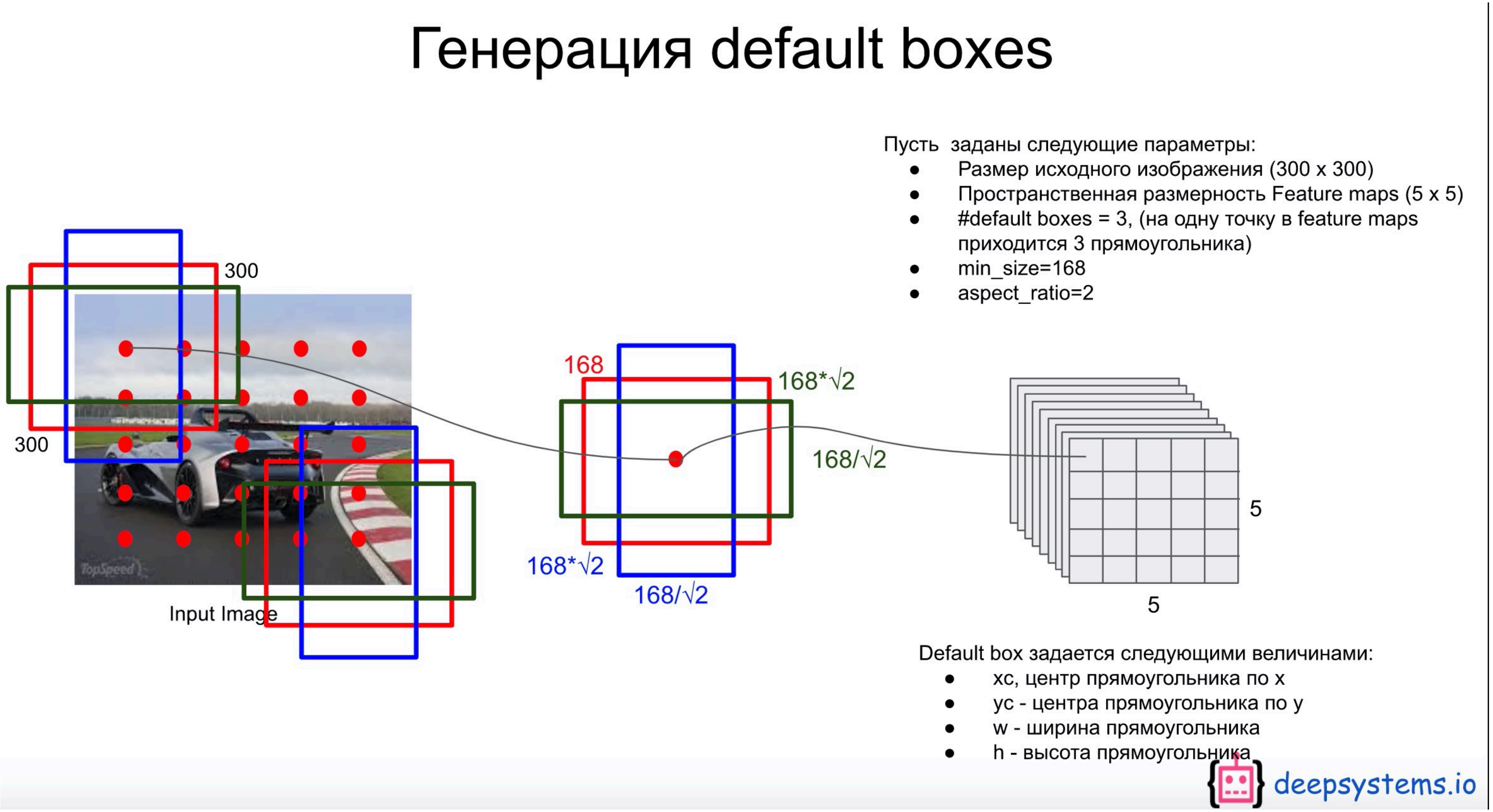
Evaluate with different scales

- we evaluate a small set (e.g. 4) of **default boxes** of **different aspect ratios** at **each location in several feature maps** with **different scales** (e.g. 8 x 8 and 4 x 4 in (b) and (c)).
- For example, in Fig. 1, the dog is **matched to a default box in the 4x4 feature map**, but not to any **default boxes in the 8x8 feature map**.


loc : $\Delta(cx, cy, w, h)$
conf : $(c_1, c_2, \cdots, c_p)$

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

Default Box



Генерация default boxes

Пусть заданы следующие параметры:

- Размер исходного изображения (300 x 300)
- Пространственная размерность Feature maps (5 x 5)
- #default boxes = 3, (на одну точку в feature maps приходится 3 прямоугольника)
- min_size=168
- aspect_ratio=2

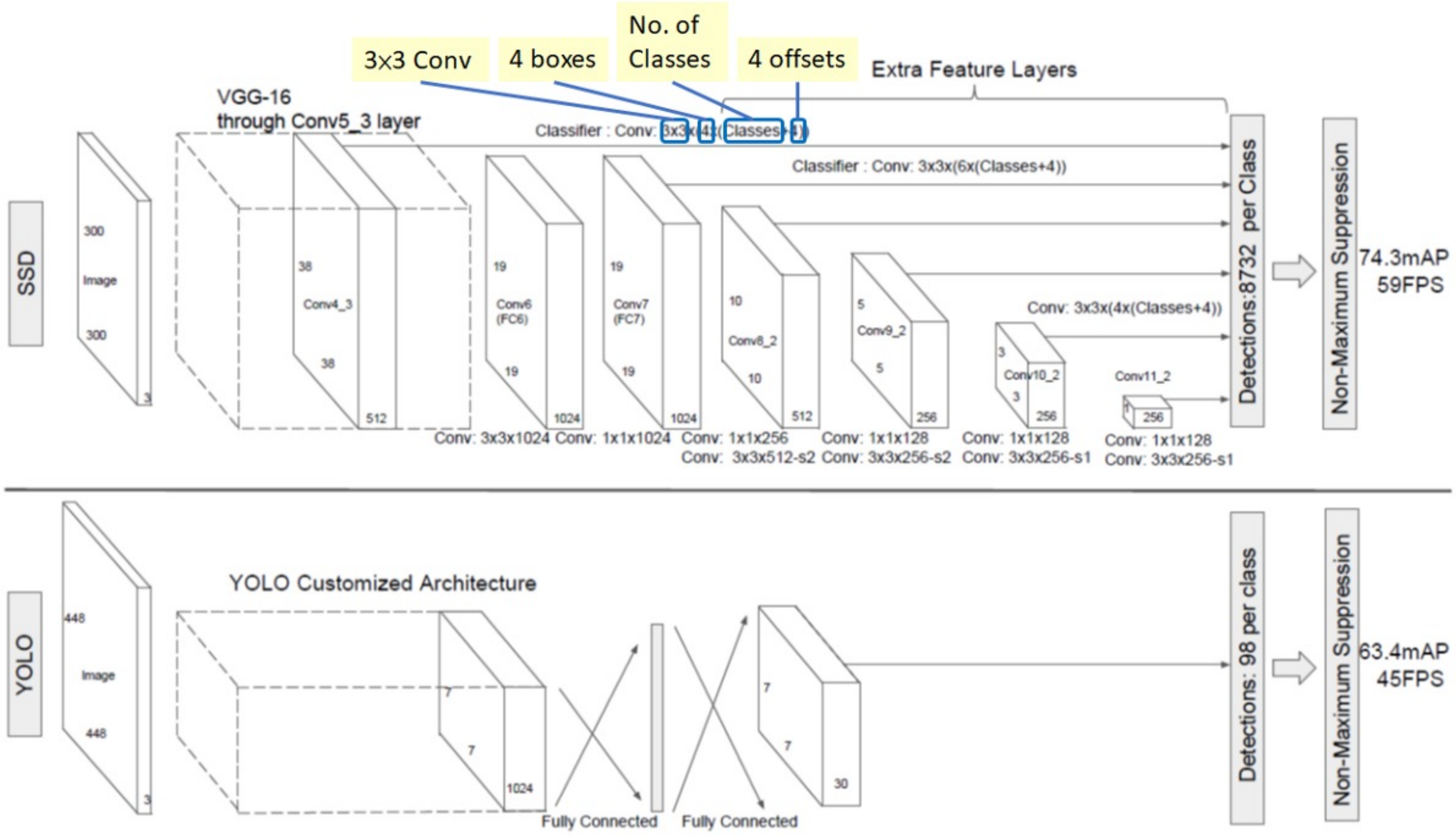
Default box задается следующими величинами:

- xc, центр прямоугольника по x
- yc - центра прямоугольника по y
- w - ширина прямоугольника
- h - высота прямоугольника

deepsystems.io

Slide credit: Deep Systems

SSD Architecture



Reference : https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11

SSD Architecture

- Say for example, at Conv4_3, it is of size **38×38×512**. 3×3 conv is applied. And there are **4 bounding boxes** and each bounding box will have (classes + 4) outputs. Thus, at Conv4_3, the output is **38×38×4×(c+4)**.
- Suppose there are **20 object classes plus one background class**, the output is 38×38×4×(21+4) = 144,400.
- In terms of number of bounding boxes, there are 38×38×4 = 5776 bounding boxes.
- Conv7: 19×19×6 = 2166 boxes (6 boxes for each location)
- Conv8_2: 10×10×6 = 600 boxes (6 boxes for each location)
- Conv9_2: 5×5×6 = 150 boxes (6 boxes for each location)
- Conv10_2: 3×3×4 = 36 boxes (4 boxes for each location)
- Conv11_2: 1×1×4 = 4 boxes (4 boxes for each location)
- If we sum them up, we got 5776 + 2166 + 600 + 150 + 36 +4 = **8732 boxes** in total. If we remember YOLO, there are 7×7 locations at the end with 2 bounding boxes for each location. YOLO only got 7×7×2 = **98 boxes**. Hence, SSD has 8732 bounding boxes which is more than that of YOLO.

Reference : https://towardsdatascience.com/review-ssd-single-shot-detector-object-detection-851a94607d11

Default Box Scales & Aspect Ratios

- Suppose we have m feature maps for prediction, we can calculate Sk for the k-th feature map. Smin is **0.2**, Smax is **0.9**. That means the scale at the lowest layer is 0.2 and the scale at the highest layer is 0.9. All layers in between is regularly spaced.

$$s_k = s_{\text{min}} + \frac{s_{\text{max}} - s_{\text{min}}}{m-1}(k-1), \quad k \in [1,m]$$

- For each scale, sk, we have 5 non-square aspect ratios:

$$a_r \in \{1, 2, 3, \tfrac{1}{2}, \tfrac{1}{3}\} \quad (w_k^a = s_k\sqrt{a_r}) \quad (h_k^a = s_k/\sqrt{a_r})$$

- For aspect ratio of 1:1, we got s'_k:

$$s'_k = \sqrt{s_k s_{k+1}}$$

- Therefore, we can have at most **6 bounding boxes** in total with different aspect ratios. For layers with only **4 bounding boxes**, a_r = 1/3 and 3 are omitted.

Matching Strategy

- During training we need to determine which default boxes correspond to a ground truth detection and train the network accordingly.
- For each ground truth box we are selecting from default boxes that vary over location, aspect ratio, and scale.
- We begin by matching each ground truth box to the default box with the best jaccard overlap (as in MultiBox [7]).
- Unlike MultiBox, we then match default boxes to any ground truth with jaccard overlap higher than a threshold (0.5).
- This simplifies the learning problem, allowing the network to predict high scores for multiple overlapping default boxes rather than requiring it to pick only the one with maximum overlap.

Loss Function

- Classification + Bounding-box Regression(Localization Loss)
- Let $x_{ij}^{p} = \{0,1\}$ be an indicator for matching the i-th default box to the j-th ground truth box of category p.
- In the matching strategy above, we can have $\sum_i x_{ij}^{p} \geq 1$.

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

- where N is the **number of matched default boxes**. If N = 0, wet set the loss to 0.
- the weight term α is set to 1 by cross validation.

Loss Function – Localization Loss

- The localization loss is a Smooth L1 loss between the predicted box (l) and the ground truth box (g) parameters.
- Similar to Faster R-CNN [2], we regress to offsets for the center (cx, cy) of the default bounding box (d) and for its width (w) and height (h).

$$\begin{gathered}
L_{loc}(x,l,g) = \sum_{i\in Pos}^{N} \sum_{m\in\{cx,cy,w,h\}} x_{ij}^{k}\,\mathrm{smooth}_{\mathrm{L1}}(l_i^m - \hat{g}_j^m) \\
\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \qquad \hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h \\
\hat{g}_j^{w} = \log\left(\frac{g_j^w}{d_i^w}\right) \qquad \hat{g}_j^{h} = \log\left(\frac{g_j^h}{d_i^h}\right)
\end{gathered} \tag{2}$$

Loss Function – Confidence Loss

- The confidence loss is the softmax loss over multiple classes confidences (c).

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

Training Tricks

- **Hard Negative Mining** : Instead of using all the negative examples, we sort them using the **highest confidence loss for each default box** and pick the top ones so that the ratio between the negatives and positives is at most 3:1. This can lead to faster optimization and a more stable training.
- **Data Augmentation** : Each training image is randomly sampled by:

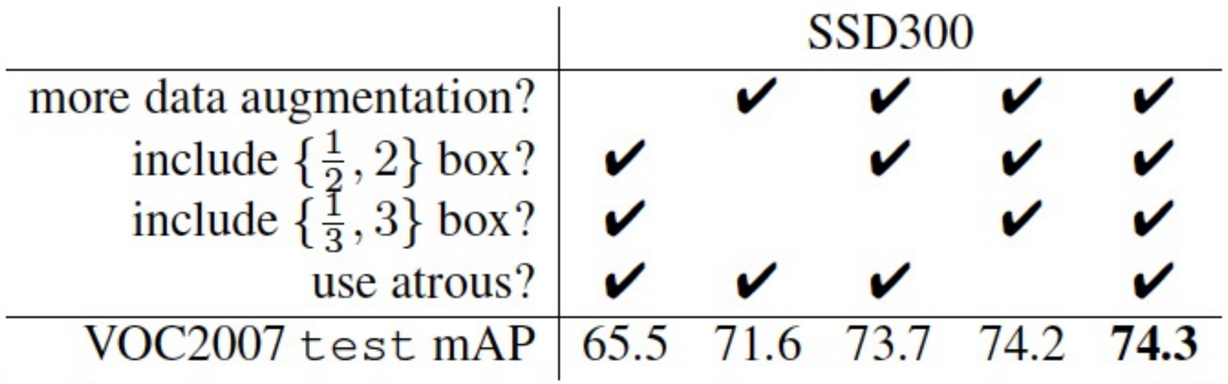
① entire original input image

② Sample a patch so that the overlap with objects is 0.1, 0.3, 0.5, 0.7 or 0.9.

③ Randomly sample a patch

- The size of each sampled patch is [0.1, 1] or original image size, and aspect ratio from 1/2 to 2. After the above steps, each sampled patch will be resized to fixed size and maybe horizontally flipped with probability of 0.5, in addition to some photo-metric distortions [14].

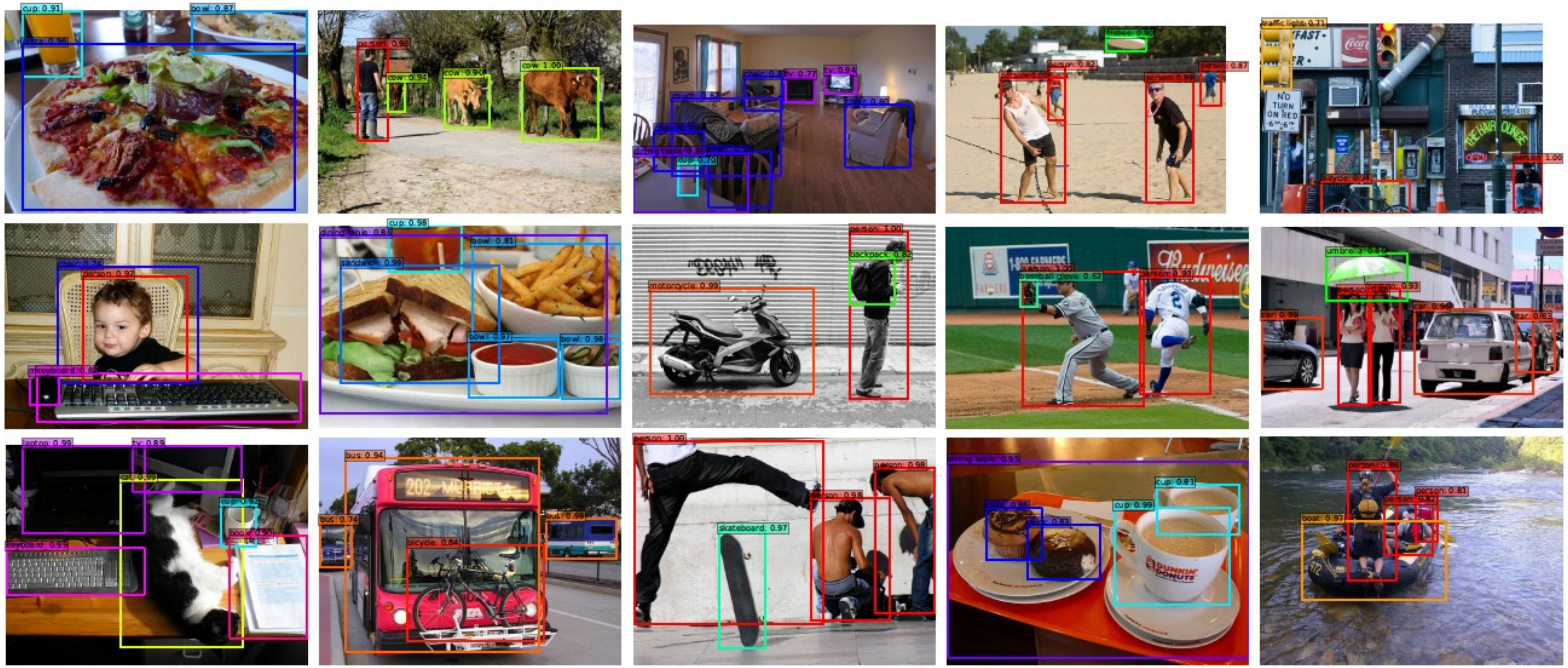
Model Analysis – Ablation Study



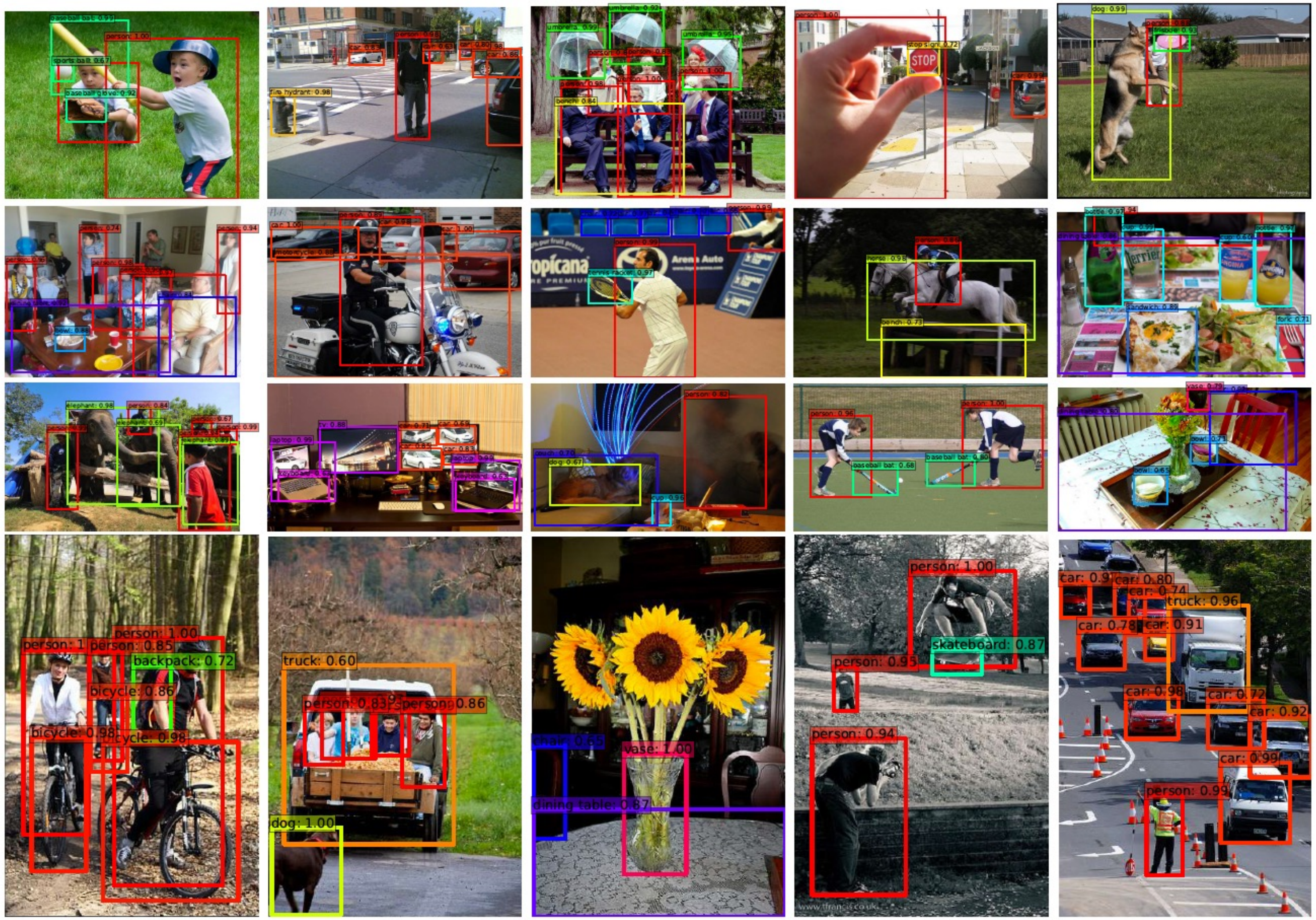
	SSD300				
more data augmentation?		✓	✓	✓	✓
include $\{\frac{1}{2}, 2\}$ box?	✓		✓	✓	✓
include $\{\frac{1}{3}, 3\}$ box?	✓			✓	✓
use atrous?	✓	✓	✓		✓
VOC2007 `test` mAP	65.5	71.6	73.7	74.2	**74.3**

Table 2: **Effects of various design choices and components on SSD performance.**

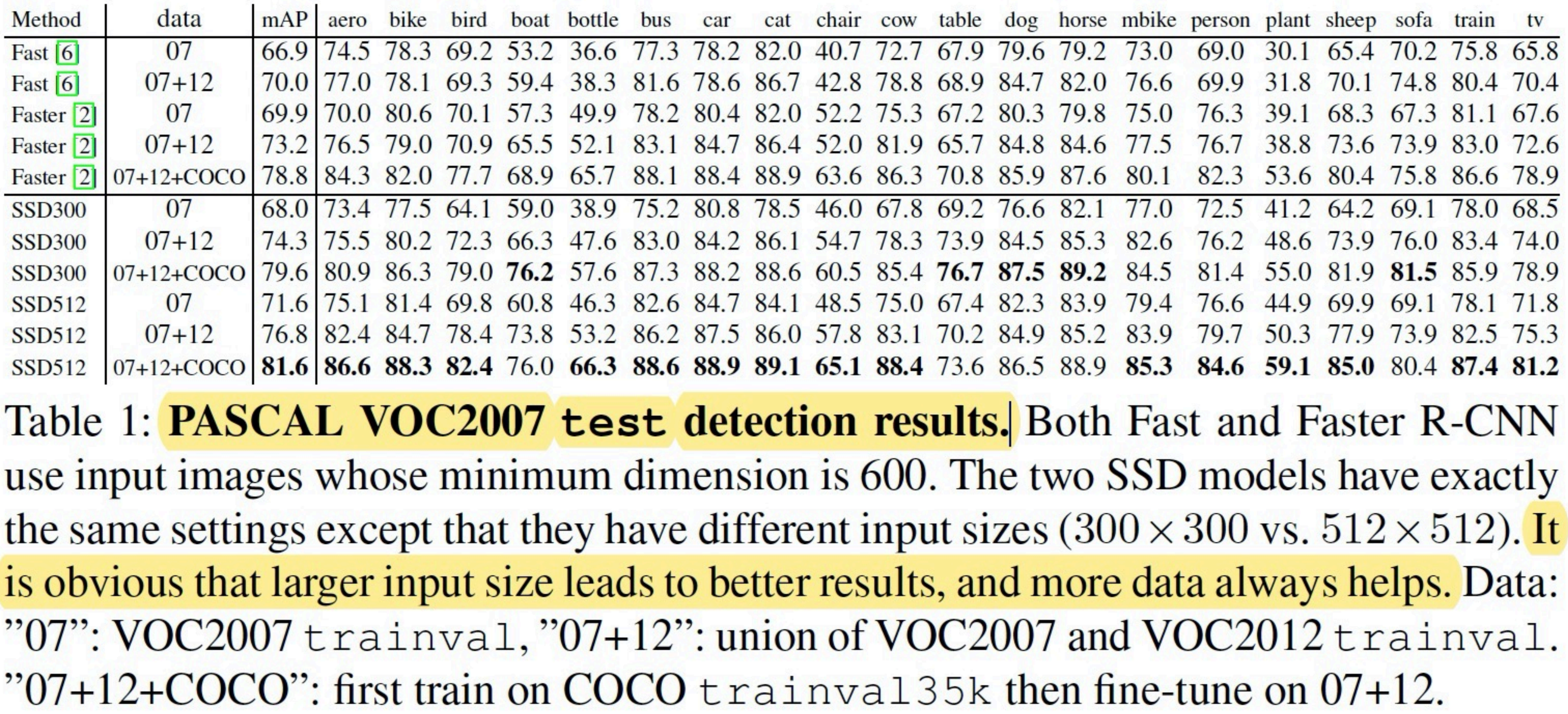
COCO test-dev with SSD512 model



COCO test-dev with SSD512 model



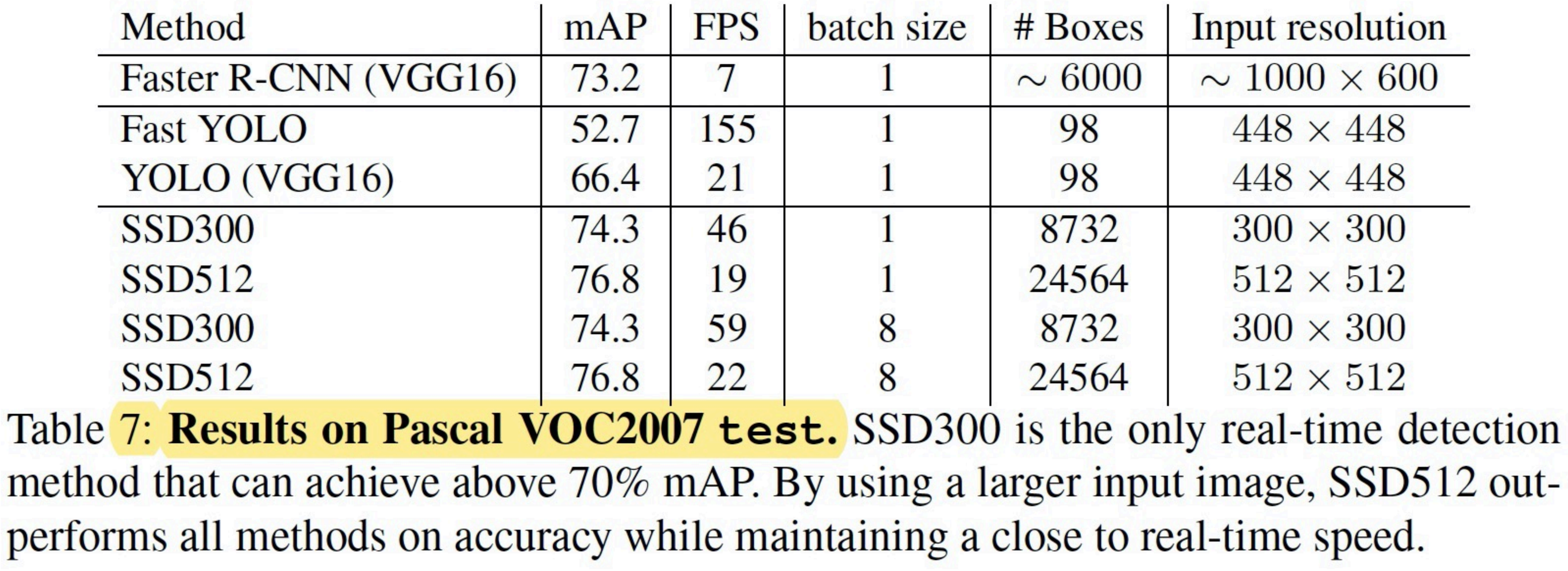
SSD 성능 (정량적 분석)



Method	data	mAP	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person	plant	sheep	sofa	train	tv
Fast [6]	07	66.9	74.5	78.3	69.2	53.2	36.6	77.3	78.2	82.0	40.7	72.7	67.9	79.6	79.2	73.0	69.0	30.1	65.4	70.2	75.8	65.8
Fast [6]	07+12	70.0	77.0	78.1	69.3	59.4	38.3	81.6	78.6	86.7	42.8	78.8	68.9	84.7	82.0	76.6	69.9	31.8	70.1	74.8	80.4	70.4
Faster [2]	07	69.9	70.0	80.6	70.1	57.3	49.9	78.2	80.4	82.0	52.2	75.3	67.2	80.3	79.8	75.0	76.3	39.1	68.3	67.3	81.1	67.6
Faster [2]	07+12	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7	38.8	73.6	73.9	83.0	72.6
Faster [2]	07+12+COCO	78.8	84.3	82.0	77.7	68.9	65.7	88.1	88.4	88.9	63.6	86.3	70.8	85.9	87.6	80.1	82.3	53.6	80.4	75.8	86.6	78.9
SSD300	07	68.0	73.4	77.5	64.1	59.0	38.9	75.2	80.8	78.5	46.0	67.8	69.2	76.6	82.1	77.0	72.5	41.2	64.2	69.1	78.0	68.5
SSD300	07+12	74.3	75.5	80.2	72.3	66.3	47.6	83.0	84.2	86.1	54.7	78.3	73.9	84.5	85.3	82.6	76.2	48.6	73.9	76.0	83.4	74.0
SSD300	07+12+COCO	79.6	80.9	86.3	79.0	**76.2**	57.6	87.3	88.2	88.6	60.5	85.4	**76.7**	**87.5**	**89.2**	84.5	81.4	55.0	81.9	**81.5**	85.9	78.9
SSD512	07	71.6	75.1	81.4	69.8	60.8	46.3	82.6	84.7	84.1	48.5	75.0	67.4	82.3	83.9	79.4	76.6	44.9	69.9	69.1	78.1	71.8
SSD512	07+12	76.8	82.4	84.7	78.4	73.8	53.2	86.2	87.5	86.0	57.8	83.1	70.2	84.9	85.2	83.9	79.7	50.3	77.9	73.9	82.5	75.3
SSD512	07+12+COCO	**81.6**	**86.6**	**88.3**	**82.4**	76.0	**66.3**	**88.6**	**88.9**	**89.1**	**65.1**	**88.4**	73.6	86.5	88.9	**85.3**	**84.6**	**59.1**	**85.0**	80.4	**87.4**	**81.2**

Table 1: **PASCAL VOC2007 `test` detection results.** Both Fast and Faster R-CNN use input images whose minimum dimension is 600. The two SSD models have exactly the same settings except that they have different input sizes (300 × 300 vs. 512 × 512). It is obvious that larger input size leads to better results, and more data always helps. Data: ”07”: VOC2007 `trainval`, ”07+12”: union of VOC2007 and VOC2012 `trainval`. ”07+12+COCO”: first train on COCO `trainval35k` then fine-tune on 07+12.

SSD 속도 분석

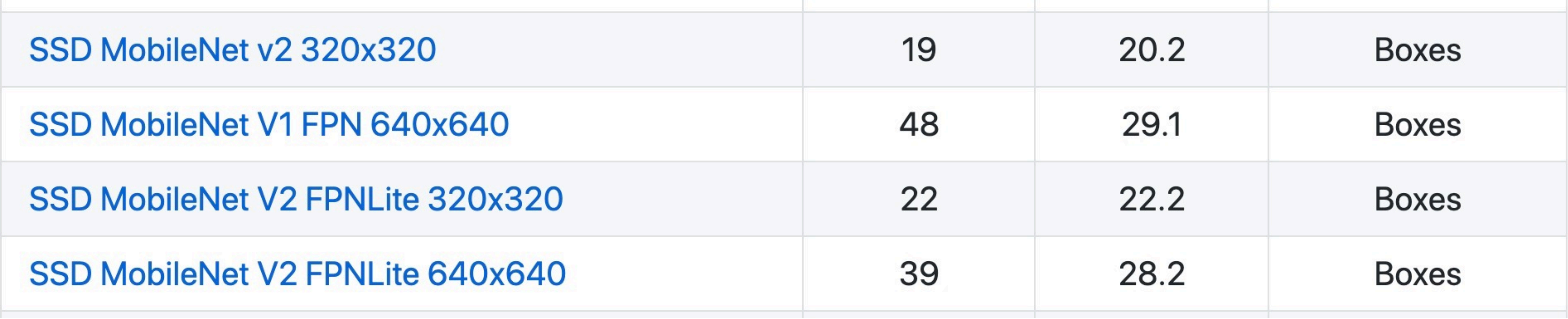


Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

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.

TensorFlow Detection Model ZOO - SSD

- https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md



SSD MobileNet v2 320x320	19	20.2	Boxes
SSD MobileNet V1 FPN 640x640	48	29.1	Boxes
SSD MobileNet V2 FPNLite 320x320	22	22.2	Boxes
SSD MobileNet V2 FPNLite 640x640	39	28.2	Boxes

SSD의 장점과 단점

- **장점** :

① Faster R-CNN과 YOLO v1에 비해 높은 성능과 빠른 속도를 가짐

- **단점** :

① 최신 Object Detection 모델에 비해 성능이 떨어짐

Thank you!