Deep Sync

SSD

나보영

DeepSync, South Korea

Object Detection

Deep Sync

Classification + Localization

Classification : Multiple objects에 대해 어떤 물체인지 부류하는 문제

Object Detection

Deep Sync

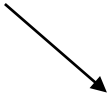
Classification + **Localization**

Localization : Bounding box를 통해 객체의 위치 정보를 나타내는 문제

Object Detection

1-stage Detector : Classification , Localization TASK를 동시에

2-stage Detector: **Classification , Localization** TASK를 순차적으로



SSD가 여기에 속함!

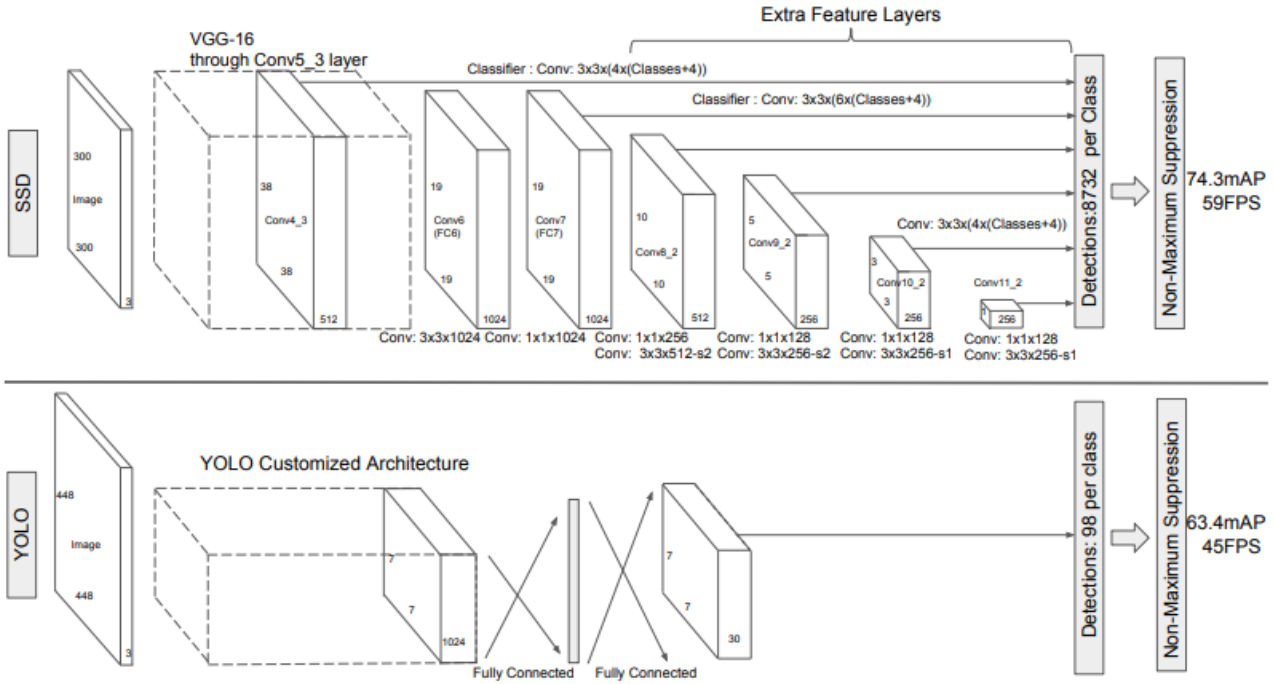
SSD 배경

Deep Sync

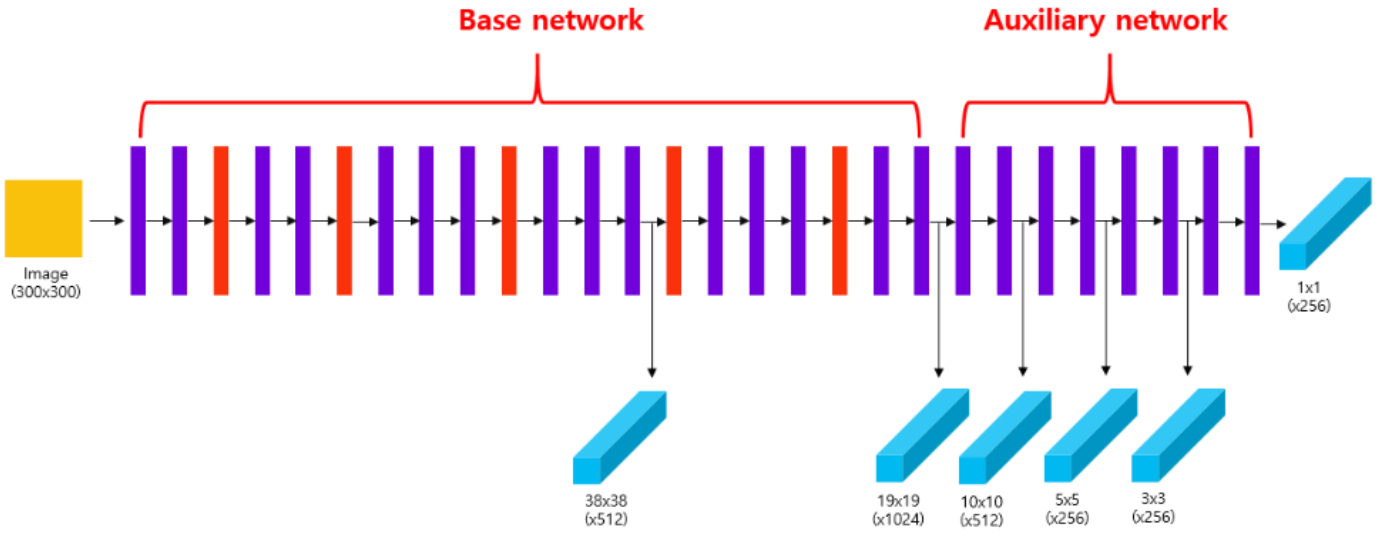
기존 모델들을 보면 YOLO의 경우 빠르지만 정확성이 낮았고 faster-r-cnn의 경우 정확성은 좋았지만 매우 느렸다

-> 이러한 문제점들을 해결하고자 SSD 제안

SSD AND YOLO

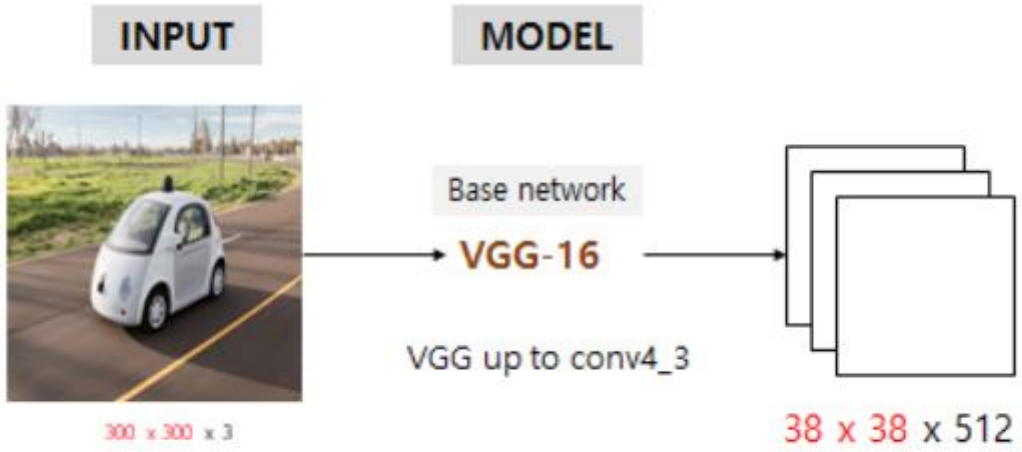

Extra Feature Layers
VGG-16
through Conv5_3 layer
Classifier : Conv: 3x3x(4x(Classes+4))
Classifier : Conv: 3x3x(6x(Classes+4))
Conv: 3x3x(4x(Classes+4))
SSD
300
Image
300
3
38
Conv4_3
38
512
19
Conv6
(FC6)
19
1024
19
Conv7
(FC7)
19
1024
10
Conv8_2
10
512
5
Conv9_2
5
256
3
Conv10_2
3
256
Conv11_2
1
256
Detections:8732 per Class
Non-Maximum Suppression
74.3mAP
59FPS
Conv: 3x3x1024
Conv: 1x1x1024
Conv: 1x1x256
Conv: 3x3x512-s2
Conv: 1x1x128
Conv: 3x3x256-s2
Conv: 1x1x128
Conv: 3x3x256-s1
Conv: 1x1x128
Conv: 3x3x256-s1
YOLO
448
Image
448
3
YOLO Customized Architecture
7
7
1024
Fully Connected
Fully Connected
7
7
30
Detections: 98 per class
Non-Maximum Suppression
63.4mAP
45FPS

Multiscale feature maps

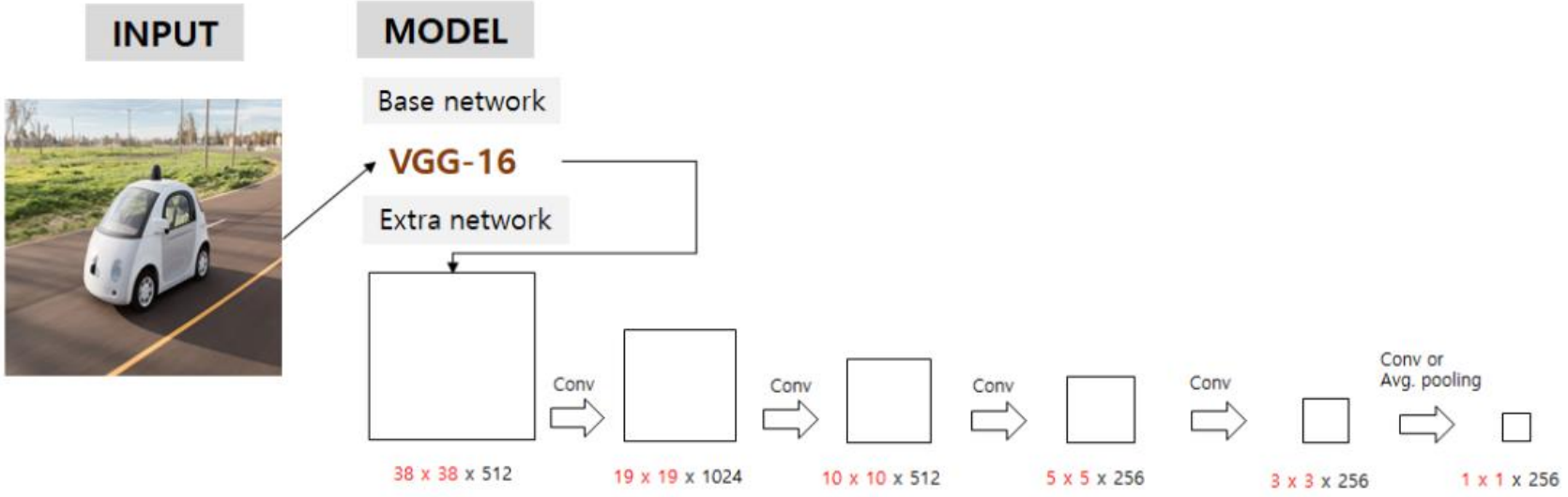


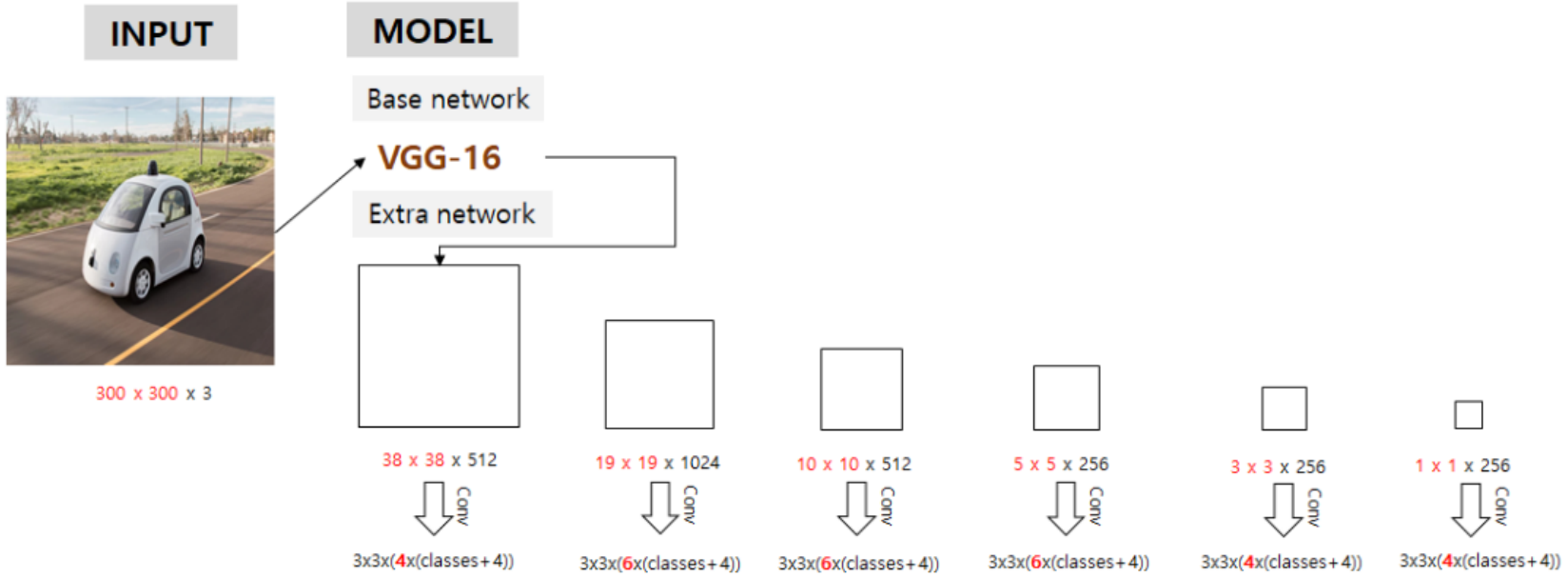
SSD architecture

Model


INPUT
MODEL
Base network
VGG-16
VGG up to conv4_3
300 x 300 x 3
38 x 38 x 512

Model


INPUT
MODEL
Base network
VGG-16
Extra network
Conv
Conv
Conv
Conv
Conv or
Avg. pooling
38 x 38 x 512
19 x 19 x 1024
10 x 10 x 512
5 x 5 x 256
3 x 3 x 256
1 x 1 x 256



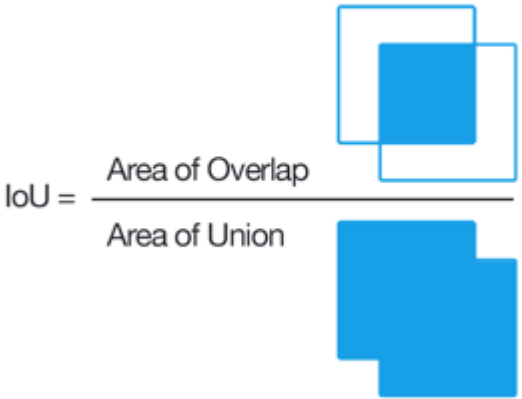
경계 박스 좌표와 객체 클래스 신뢰도 점수 구하기

3 x 3
x (# bounding box x (classes + offset))
(s = 1, p = 1)

OUTPUT

# of Bounding box
= 8732

Model

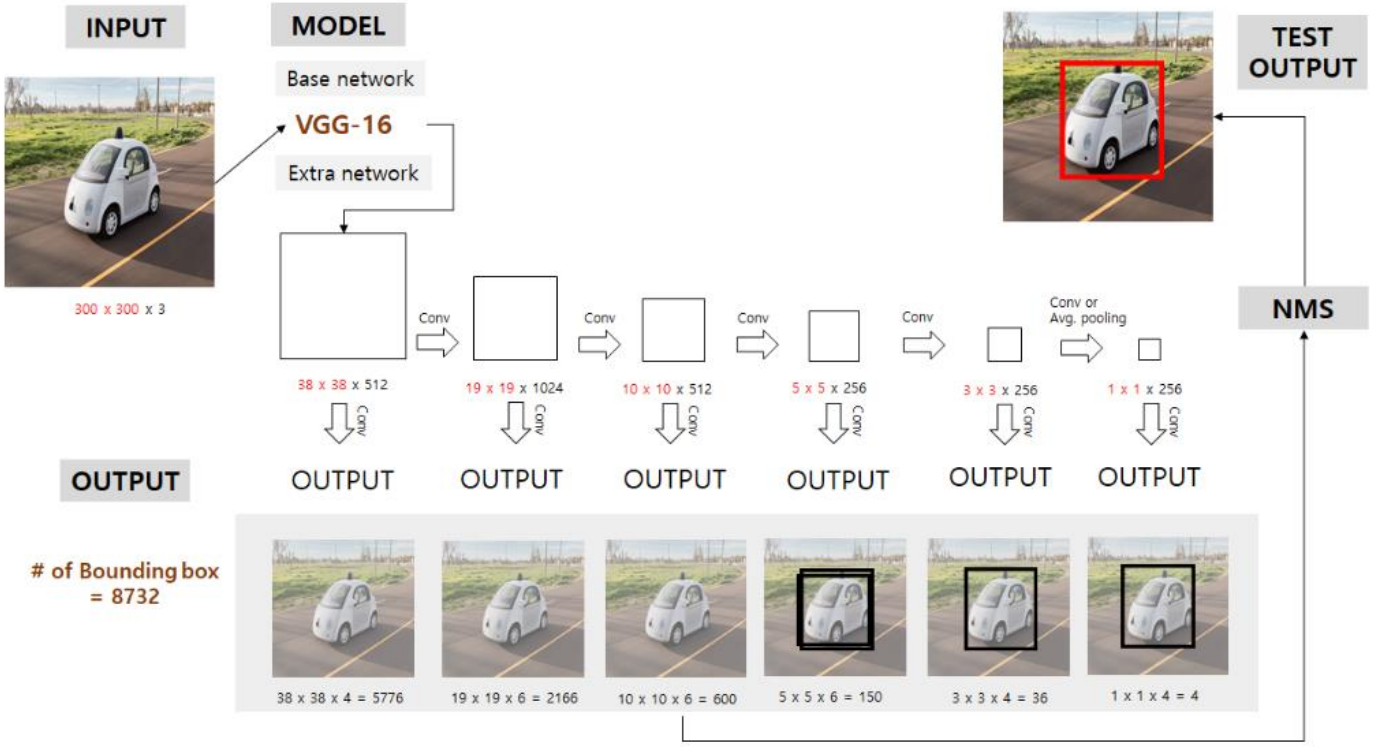


ground truth boxes and Default boxes

IOU >= 0.5 -> 1

IOU < 0.5 -> 0

Model


INPUT
MODEL
Base network
VGG-16
Extra network
300 x 300 x 3
Conv
38 x 38 x 512
19 x 19 x 1024
10 x 10 x 512
5 x 5 x 256
3 x 3 x 256
Conv or Avg. pooling
1 x 1 x 256
OUTPUT
OUTPUT
OUTPUT
OUTPUT
OUTPUT
OUTPUT
OUTPUT
# of Bounding box = 8732
38 x 38 x 4 = 5776
19 x 19 x 6 = 2166
10 x 10 x 6 = 600
5 x 5 x 6 = 150
3 x 3 x 4 = 36
1 x 1 x 4 = 4
NMS
TEST OUTPUT

Default boxes

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

s_min = 0.2, s_max = 0.9이며 m값에 따라 구간을 나누어줌.
m=6일때 [0.2, 0.34, 0.48, 0.62, 0.76, 0.9]
각각의 피쳐맵에서 default box의 크기를 계산시
입력 이미지의 너비, 높이에 대해서 얼만큼 큰 지를 나타내는 값
즉 첫번째 피처맵에서는 0.2비율의 Default boxes를 , 2번째 피처맵에서는 0.9비율의 Default boxes을 사용

Default boxes

$$(h_k^a = s_k/\sqrt{a_r})$$
$$(w_k^a = s_k\sqrt{a_r})$$

(1, 2, 3, 1/2, 1/3) 비율 값

K = 3일때

```
k: 3 scale: 0.48
widht: 0.48 height: 0.48
widht: 0.68 height: 0.34
widht: 0.83 height: 0.28
widht: 0.34 height: 0.68
widht: 0.28 height: 0.84
width: 0.55 height: 0.55
```

Default boxes 중심점

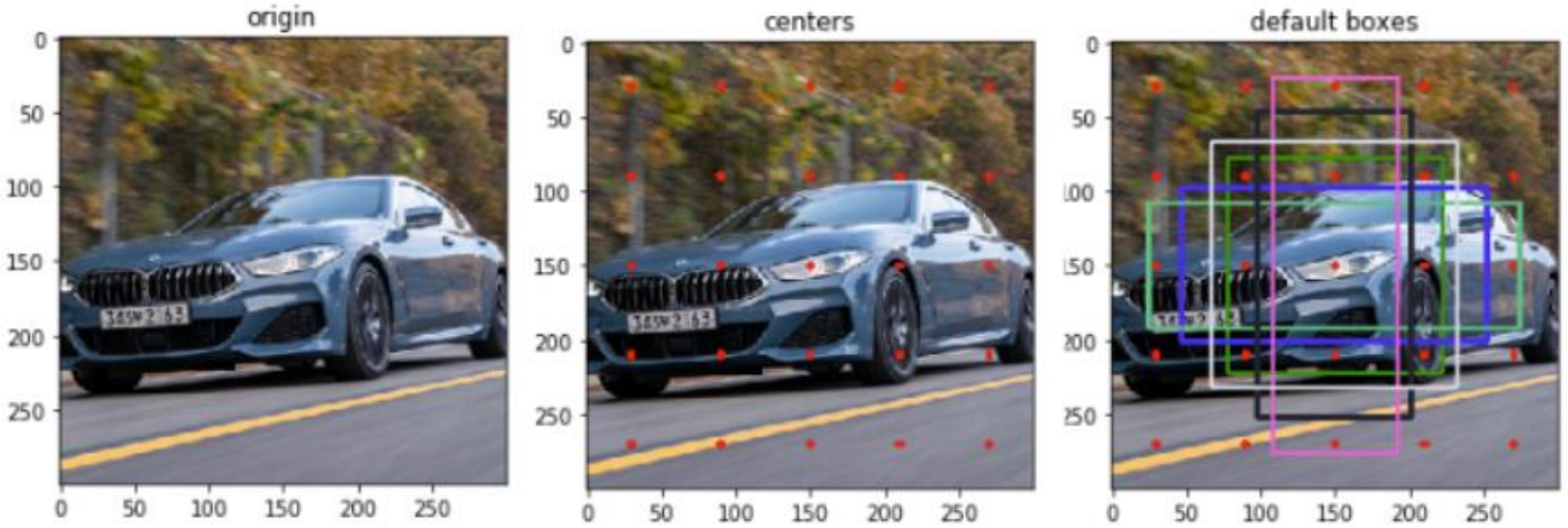
Deep Sync

$$\left(\frac{i+0.5}{|f_k|}, \frac{j+0.5}{|f_k|}\right), \quad i, j \in [0, |f_k|)$$

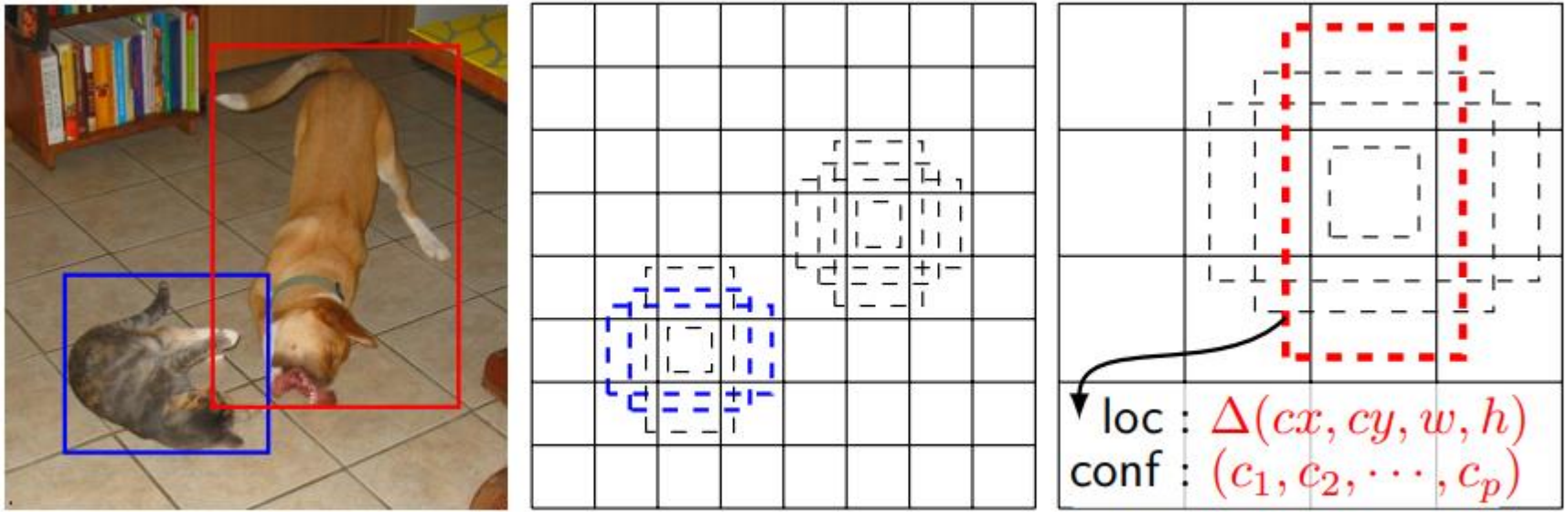
fk는 피쳐맵의 가로 세로 크기
K = 3일때

```
feature map size: 5x5
total indexes: 25
[(0.1, 0.1), (0.1, 0.3), (0.1, 0.5), (0.1, 0.7), (0.1, 0.9), (0.3, 0.1), (0.3, 0.3), (0.3, 0.5), (0.3, 0.7), (0.3, 0.
9), (0.5, 0.1), (0.5, 0.3), (0.5, 0.5), (0.5, 0.7), (0.5, 0.9), (0.7, 0.1), (0.7, 0.3), (0.7, 0.5), (0.7, 0.7), (0.7,
0.9), (0.9, 0.1), (0.9, 0.3), (0.9, 0.5), (0.9, 0.7), (0.9, 0.9)]
```

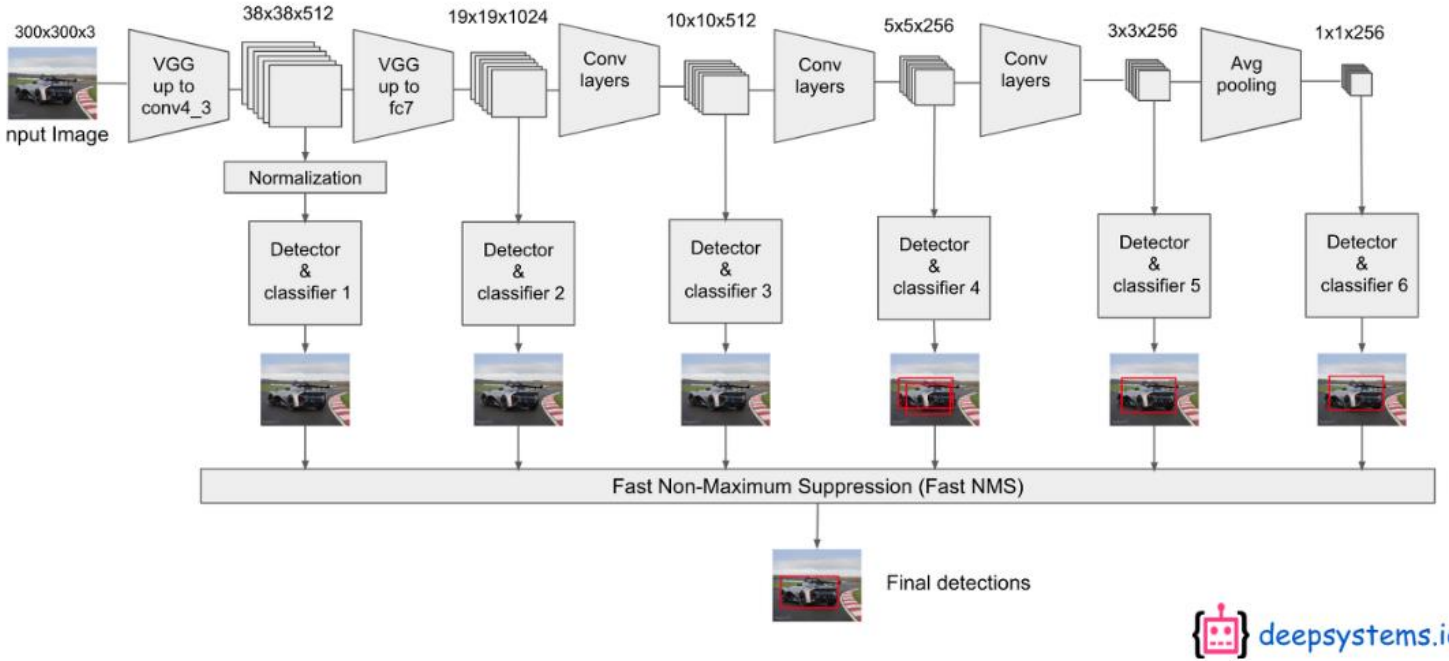
Default boxes



GPT-2



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



Training objective

Deep Sync

$$x_{ij}^{p} = \{1, 0\}$$

(클래스 p에 대해) i 번째 디폴트 박스와
j 번째 참 경계 박스가 매칭되는지 여부

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

신뢰도 손실(confidence loss) 가중치 Localization 손실

손실 함수

Deep Sync

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

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

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

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

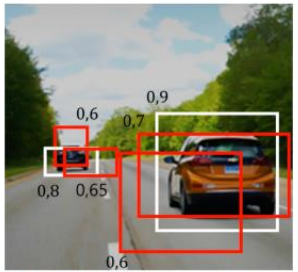
NMS

Figure 5

1. 모든 bounding box에 대하여 threshold 이하의 confidence score를 가지는 Bounding Box는 제거

2. 모두 내림차순 정렬
3. 다른 bounding box와 IoU 값을 구하고 IoU가 threshold 이상인 Bounding box들은 제거
4. 해당 과정을 순차적으로 시행하여 모든 Bounding box를 비교하고 제거
5. Confidense threshold가 높을수록, IoU threshold가 낮을수록 더 많은 bounding box가 제거

NMS



2. Bounding box를 Confidence score 기준 내림차순 정렬합니다.
   그 결과 [0.9 박스, 0.8 박스, 0.7 박스, 0.65 박스, 0.6 박스(왼쪽), 0.6 박스(오른쪽)] 이렇게 됩니다.
3. Confidence score 0.9인 bounding box를 기준으로 잡고 뒤의 모든 박스를 비교합니다.
   - 0.8 박스와는 겹치지 않으므로 남겨둠.
   - 0.7 박스와 IoU가 threshold 이상이므로 이 박스는 0.9 박스와 같은 것을 가리킨다고 간주하고 제거함.
   - 0.65 박스, 0.6 박스(왼쪽)과는 겹치지 않으므로 남겨둠.
   - 0.6 박스(오른쪽)와 IoU가 또 threshold 이상이므로 제거함.
4. 이제 0.8 박스를 기준으로 뒤의 모든 박스와 비교합니다.

Hard negative mining

Deep Sync

- 디폴트 박스를 매칭하고 나면, 대부분 디폴트 박스는 negative
- positive 훈련 샘플과 negative 훈련 샘플 간 불균형 완화를 위해
- 신뢰도 손실 점수를 기반으로 negative 샘플과 positive 샘플 비율이 3:1 이 되도록 negative 샘플을 제외

Experimental Results

Deep Sync

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**

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