

BE530 – Medical Deep Learning

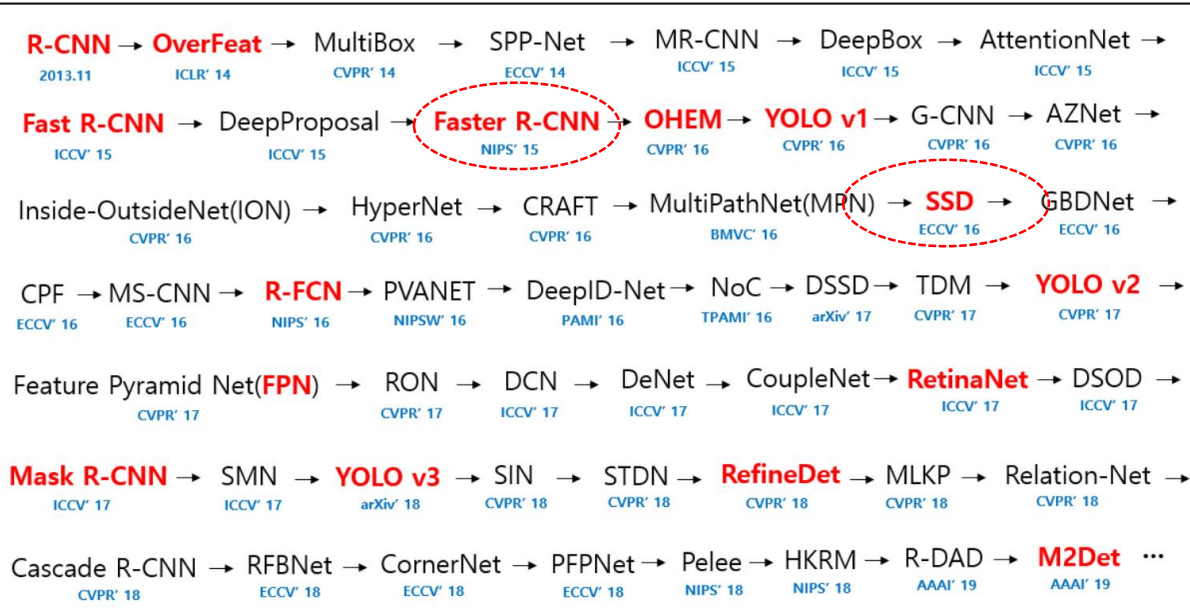
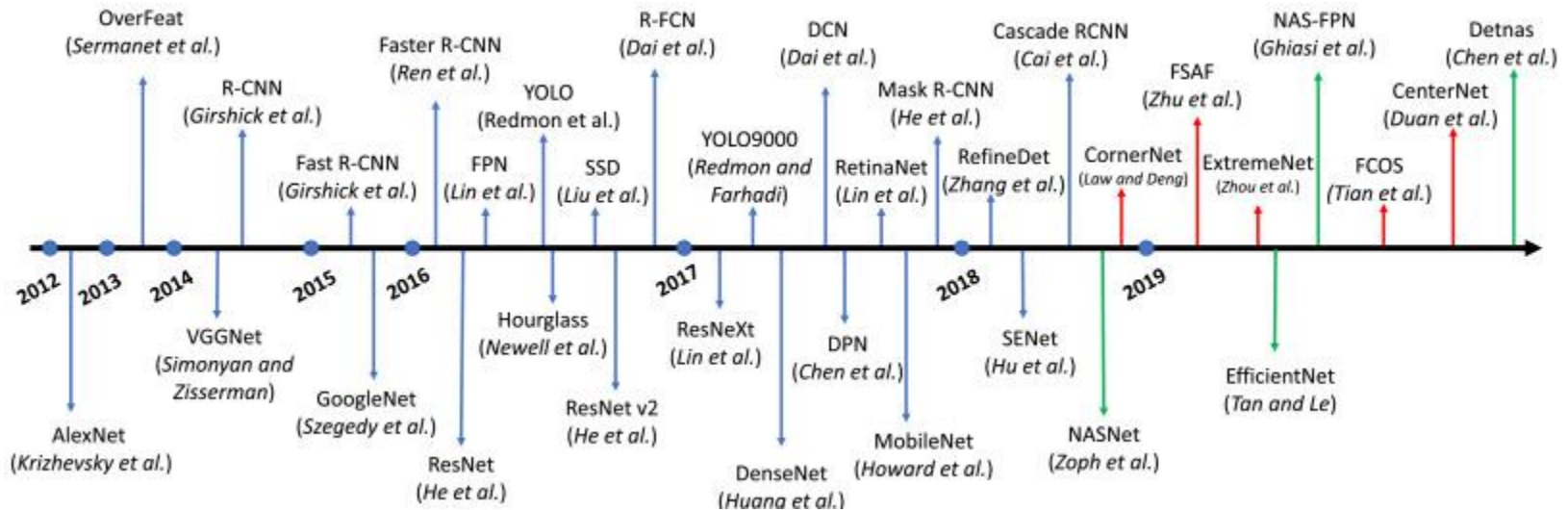
– Object Detection & Segmentation –

Byoung-Dai Lee

Division of AI Computer Science and Engineering

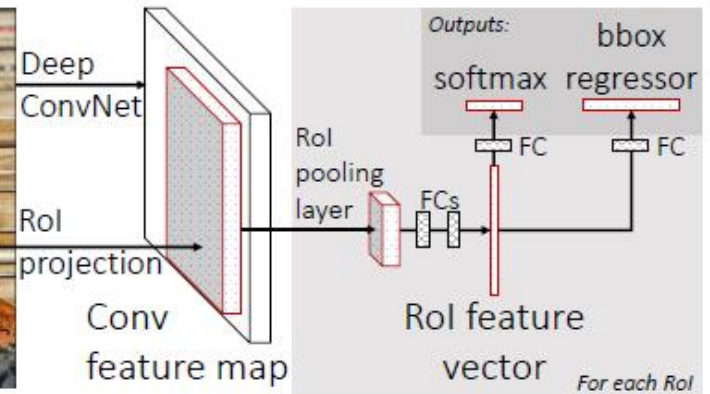
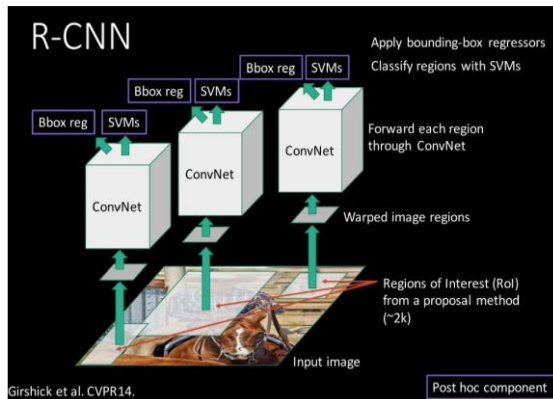
Kyonggi University

Object Detection Milestones

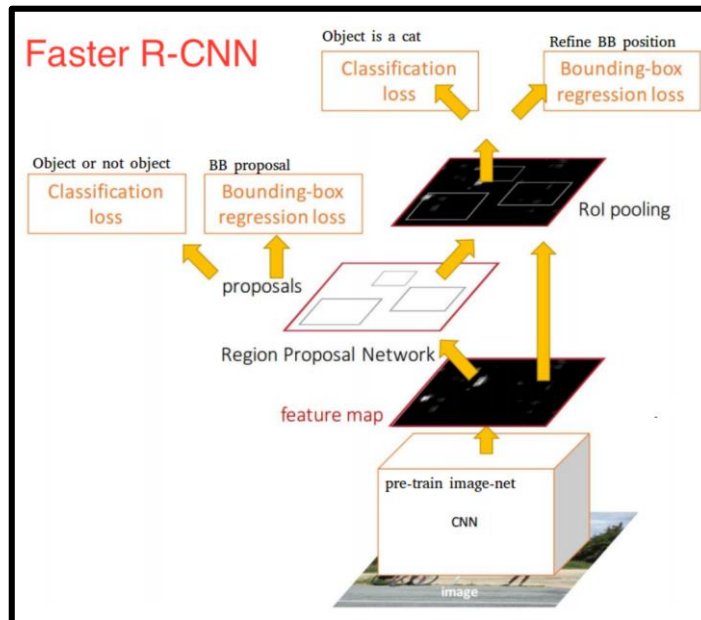


Faster R-CNN (1)

■ A Brief History

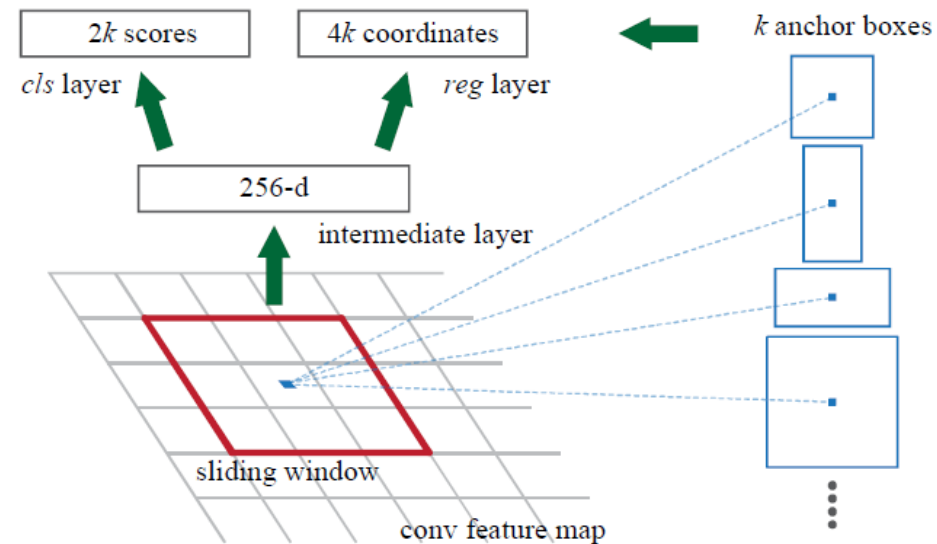
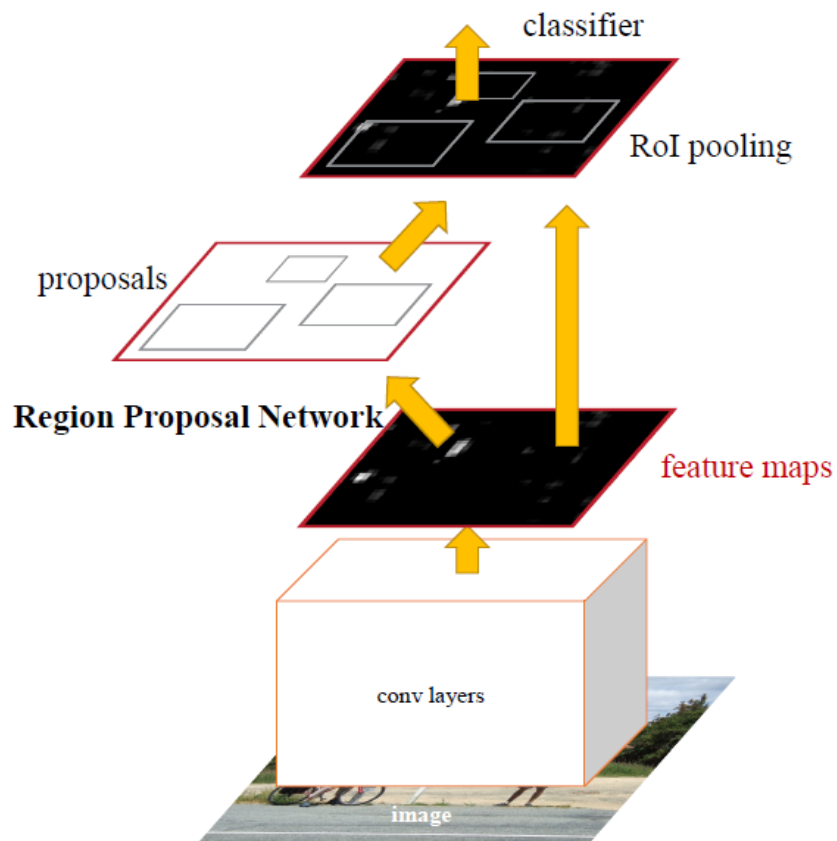


Fast R-CNN

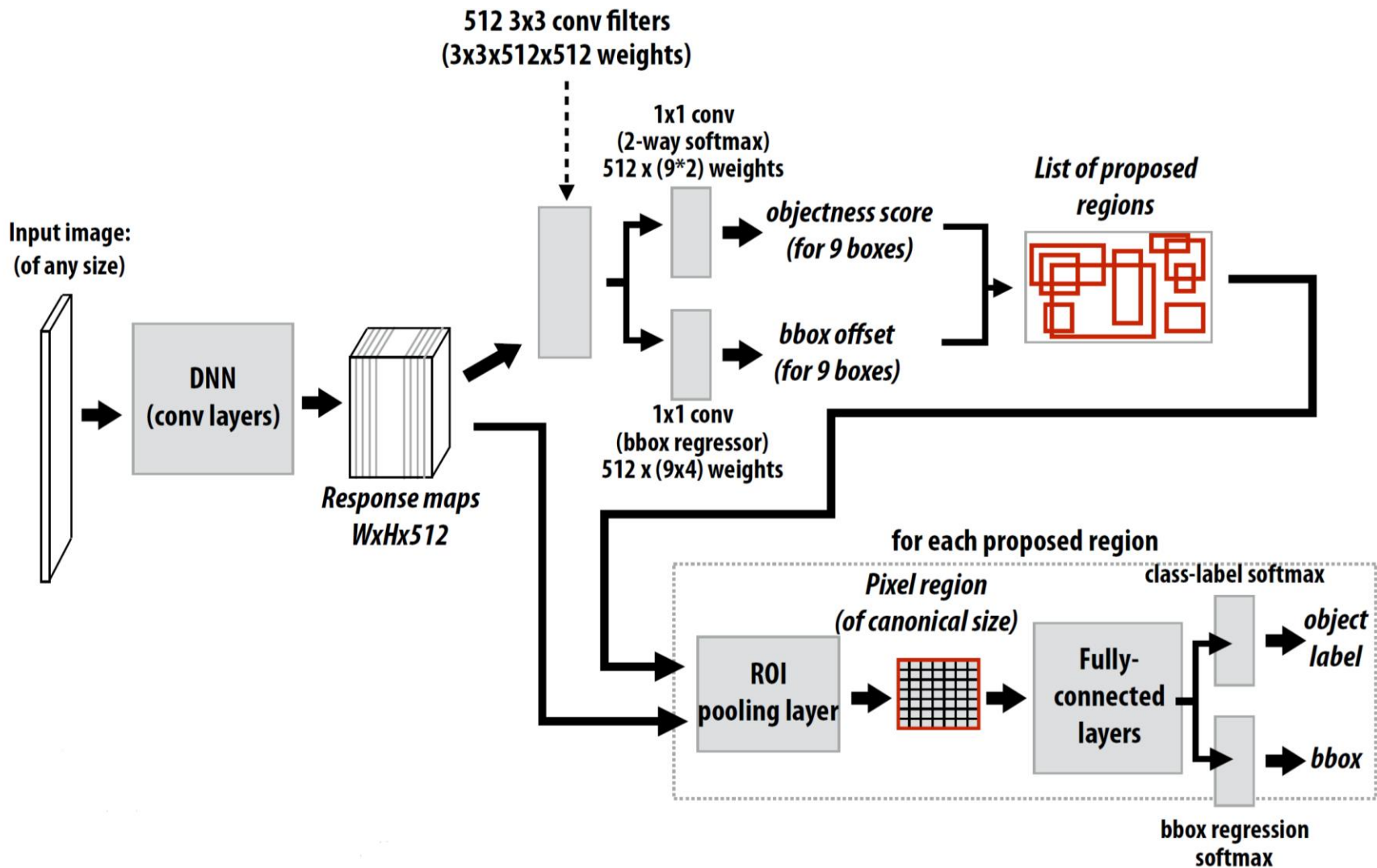


Faster R-CNN (2)

■ Architecture



Faster R-CNN (3)



Faster R-CNN (4)

■ Multi-task loss function

$$L(p, u, t^u, v) = L_{\text{cls}}(p, u) + \lambda[u \geq 1]L_{\text{loc}}(t^u, v)$$

- $L_{\text{cls}}(p, u) = -\log p_u$
 - Log loss for true class u
- $L_{\text{loc}}(t^u, v)$
 - bbox regression target for class $u \Rightarrow (v_x, v_y, v_w, v_h)$
 - bbox regression target for predicted tuple $\Rightarrow t^u = (t_x^u, t_y^u, t_w^u, t_h^u)$

$$L_{\text{loc}}(t^u, v) = \sum_{i \in \{x, y, w, h\}} \text{smooth}_{L_1}(t_i^u - v_i)$$

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

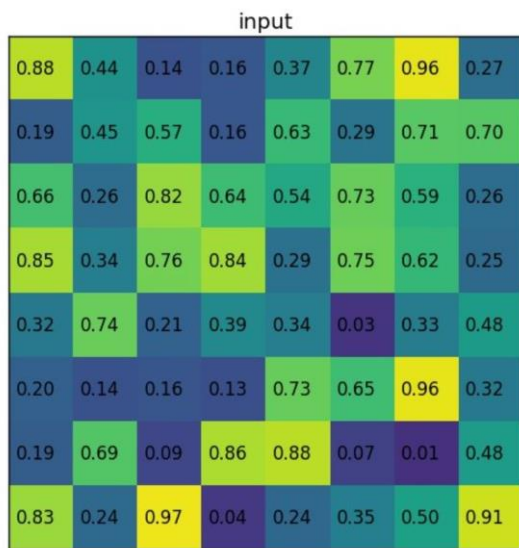
$$\begin{aligned} t_x &= (x - x_a)/w_a, & t_y &= (y - y_a)/h_a, \\ t_w &= \log(w/w_a), & t_h &= \log(h/h_a), \\ t_x^* &= (x^* - x_a)/w_a, & t_y^* &= (y^* - y_a)/h_a, \\ t_w^* &= \log(w^*/w_a), & t_h^* &= \log(h^*/h_a), \end{aligned}$$

where x, y, w , and h denote the box's center coordinates and its width and height. Variables x, x_a , and x^* are for the predicted box, anchor box, and ground-truth box respectively (likewise for y, w, h).

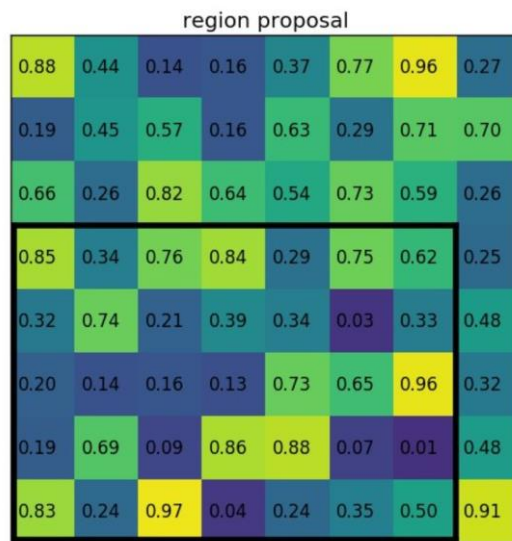
Faster R-CNN (5)

■ ROI Pooling

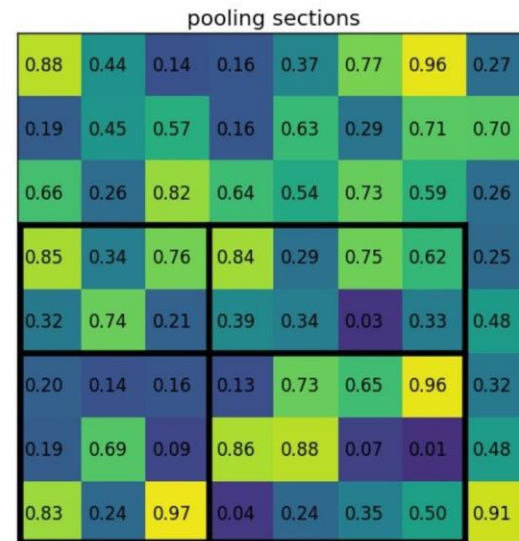
- 서로 다른 크기의 Region Proposal을 동일 크기의 feature map으로 변환
- FC Layer의 입력으로 사용되기 위해 반드시 필요



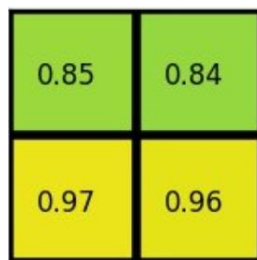
8×8 Feature map



RPN Output: [(0,3), (7,8)]



Section(2x2)으로 분할



Max pooling

Faster R-CNN (6)


■ Non-Maximum Suppression (NMS)

- 동일한 객체를 가리키는 다수의 중복된 Region Proposal 제거 시 사용

```
1 def nms(bboxes, probs, threshold):
2     """Non-Maximum suppression.
3     Args:
4         bboxes: array of [cx, cy, w, h] (center format)
5         probs: array of probabilities
6         threshold: two boxes are considered overlapping if their IOU is larger than
7                     this threshold
8         form: 'center' or 'diagonal'
9     Returns:
10        keep: array of True or False.
11        """
12
13     order = probs.argsort()[::-1]
14     keep = [True]*len(order)
15
16     for i in range(len(order)-1):
17         ovps = batch_iou(bboxes[order[i+1:]], bboxes[order[i]])
18         for j, ov in enumerate(ovps):
19             if ov > threshold:
20                 keep[order[j+i+1]] = False
21     return keep
```

① 동일 클래스에 대해 Confidence score 기준 내림차순으로 정렬 (line 13)

② 가장 confidence score가 높은 bbox와의 IOU가 설정된 임계값보다 높은 bbox는 동일 물체를 검출한 것으로 판단하여 제거 (line 16-20)

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


Single Shot Multibox Detector (1)

- Final feature map만을 사용하는 것이 아니라 Convolution 과정에 존재하는 각각의 layer를 대상으로 detection prediction 수행
 - 각 layer에서 사용되는 receptive field 크기가 다르기 때문에 다양한 resolution의 object에 대한 학습이 가능해짐

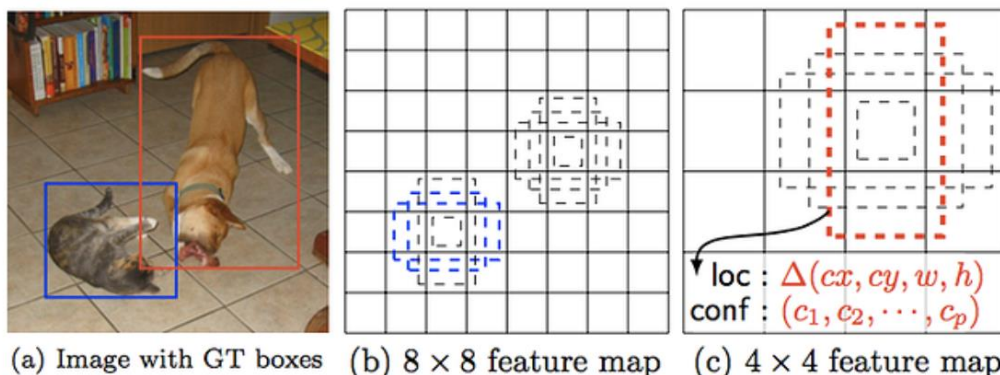
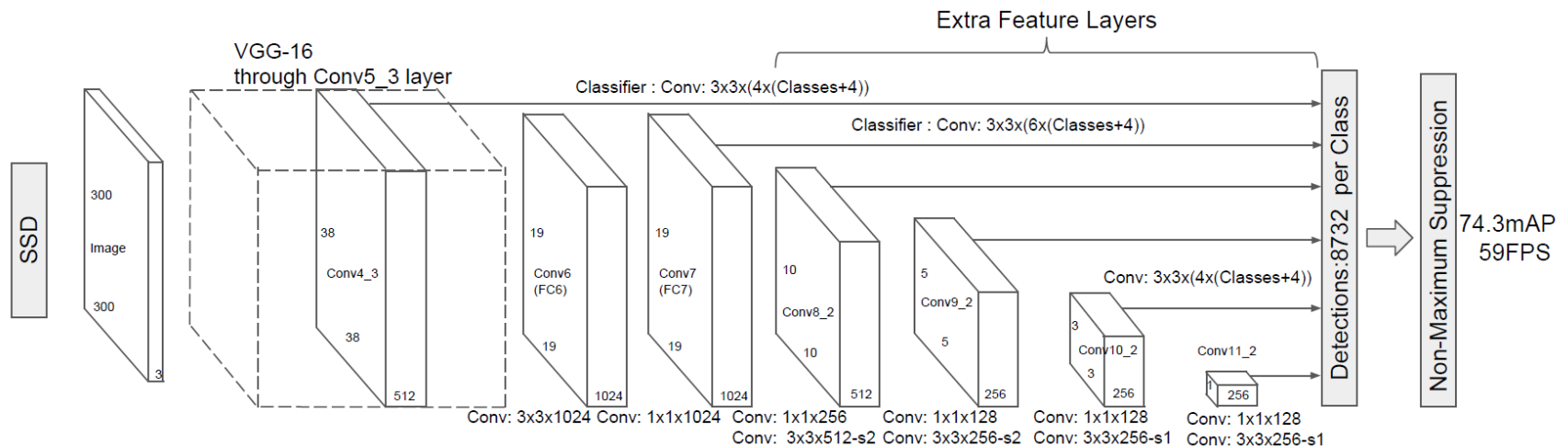


Fig. 1: **SSD framework.** (a) SSD only needs an input image and ground truth boxes for each object during training. In a convolutional fashion, we evaluate a small set (e.g. 4) of default boxes of different aspect ratios at each location in several feature maps with different scales (e.g. 8×8 and 4×4 in (b) and (c)). For each default box, we predict both the shape offsets and the confidences for all object categories $((c_1, c_2, \dots, c_p))$. At training time, we first match these default boxes to the ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss (e.g. Smooth L1 [6]) and confidence loss (e.g. Softmax).

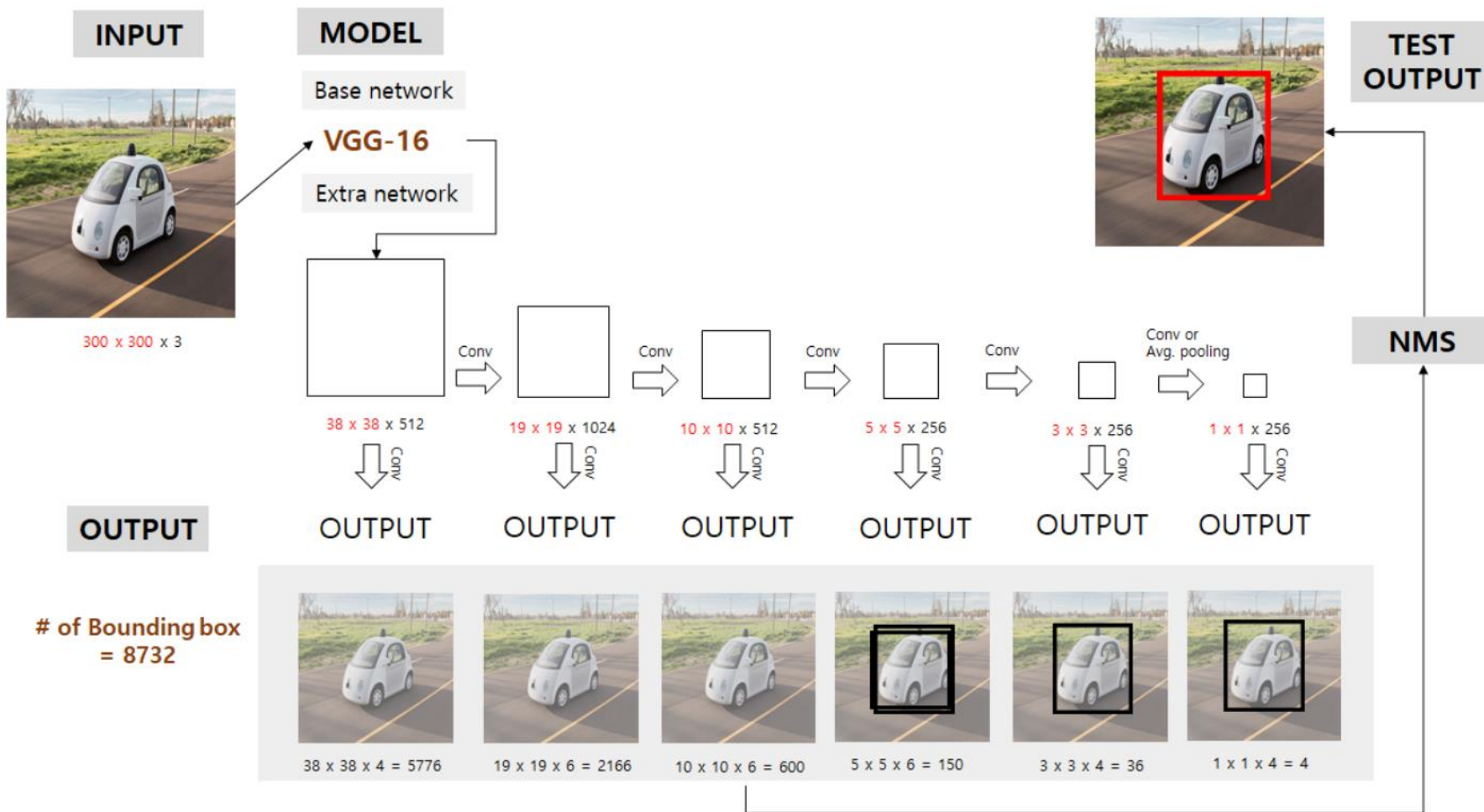
Single Shot Multibox Detector (2)

■ Architecture



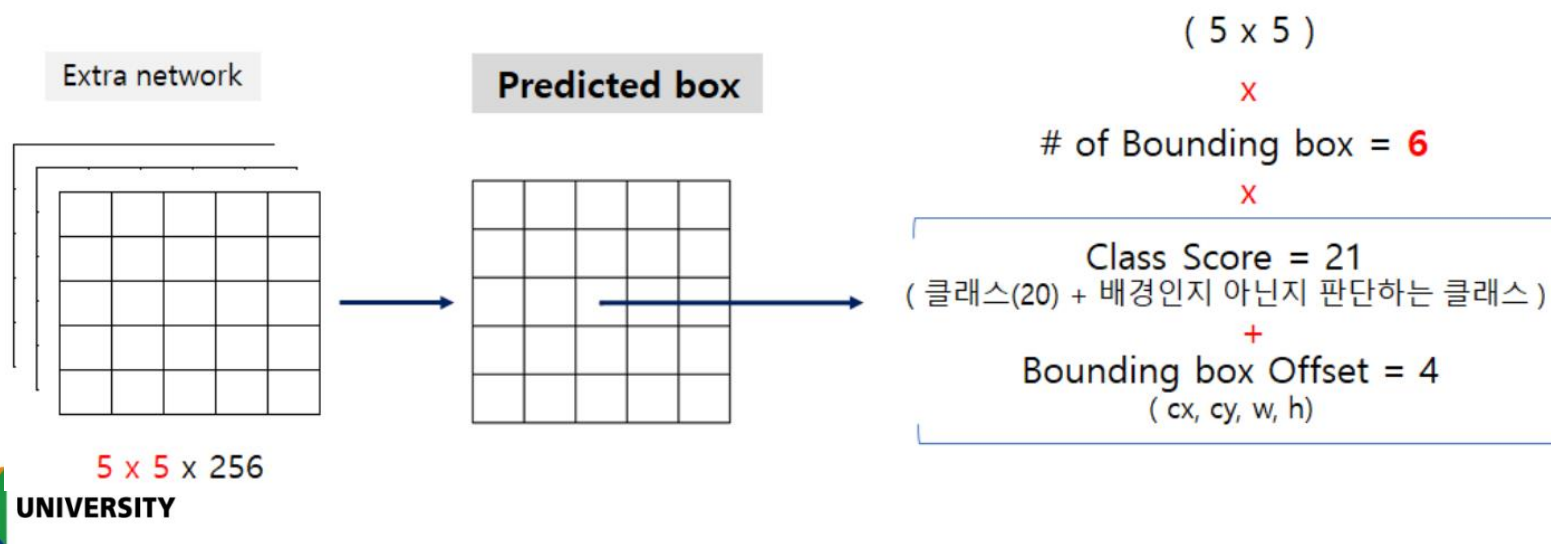
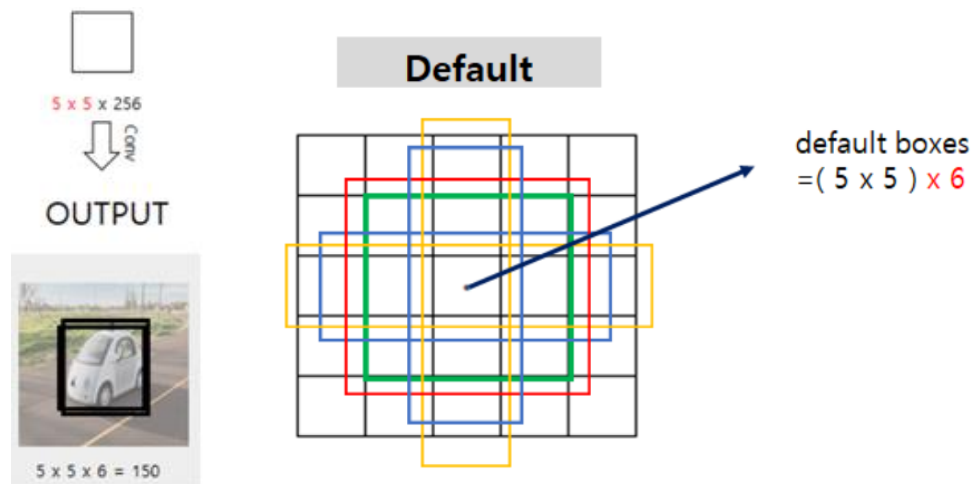
- 각각의 conv. layer에서 사용되는 default box의 개수는 서로 다를 수 있음. 예를 들어, Conv4_3에서는 4개의 default box를 사용하지만 Conv7에서는 6개의 default box를 사용함

Single Shot Multibox Detector (3)



Single Shot Multibox Detector (4)

■ Training process



Single Shot Multibox Detector (4)

■ Loss function

- Faster R-CNN에서 사용된 multi-objective function과 유사

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

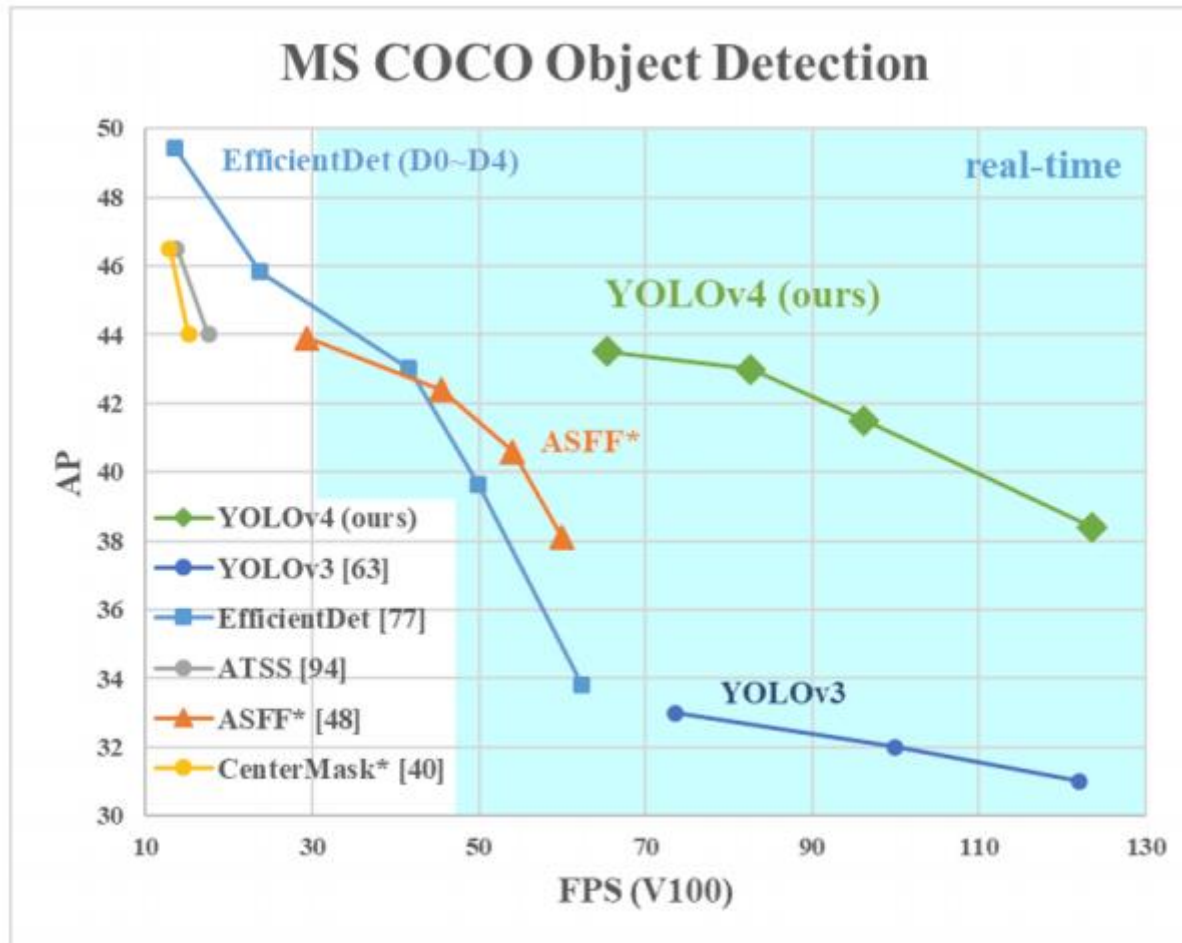
$$L_{loc}(x, l, g) = \sum_{i \in Pos}^N \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k \text{smooth}_{L1}(l_i^m - \hat{g}_j^m)$$

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

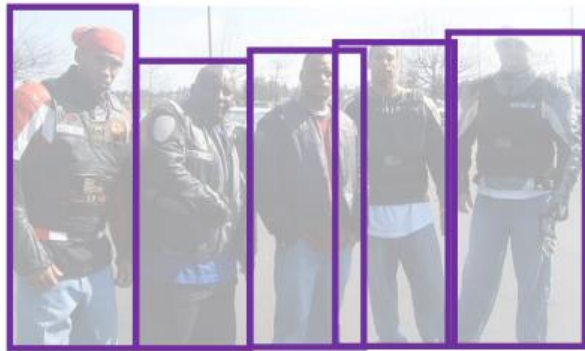
$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \quad \hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right)$$

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

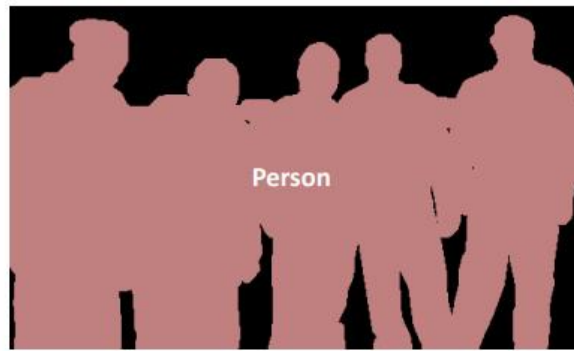
Object Detection Performance



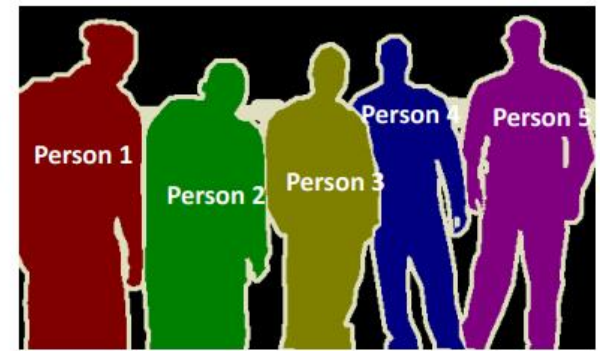
Object Segmentation



Object Detection



Semantic Segmentation

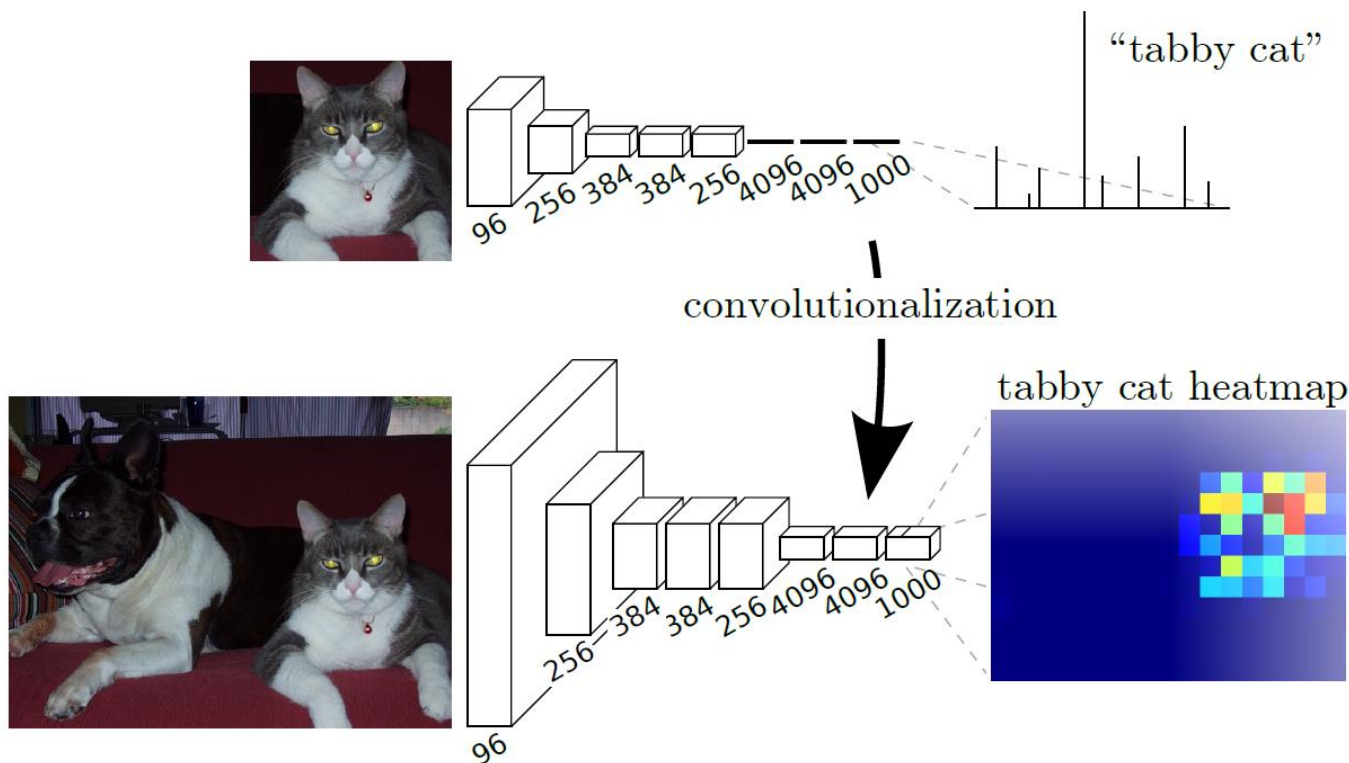


Instance Segmentation

Fully Convolutional Network (1)

■ DNN for Classification

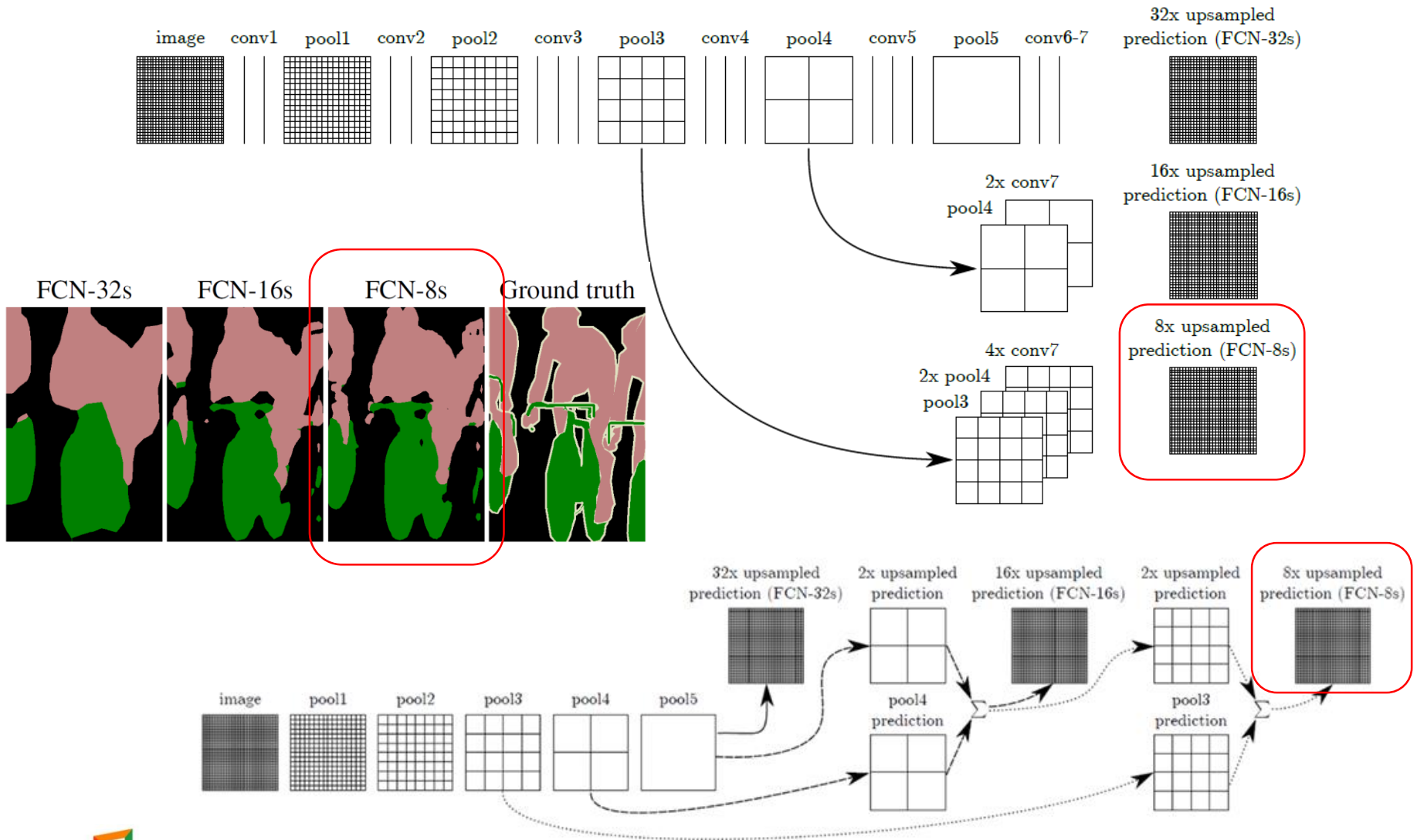
- Fully connected layer가 올 경우 문제점
 - 고정된 크기의 입력만을 받음
 - 위치 정보가 사라짐



Fully Convolutional Network (2)

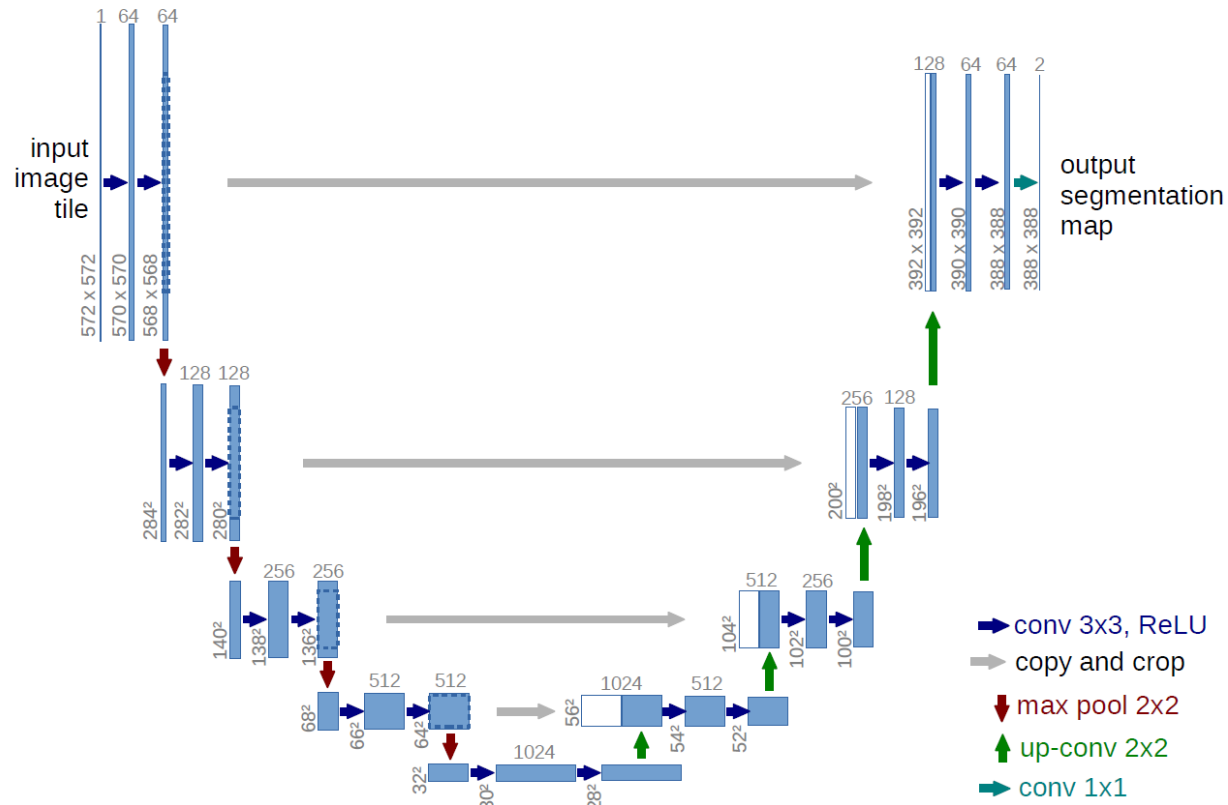
- Fully connected layer를 1×1 convolution으로 대체
 - 위치 정보를 유지하게 됨
 - 모든 layer가 convolutional layer로 구성되기 때문에 입력 영상의 크기를 고정할 필요가 없음
- 1×1 convolution을 거친 최종 feature map을 (Deconvolution + Skip Layer)을 사용하여 원본 영상의 크기로 확대
 - Deconvolution에 사용되는 계수는 학습을 통해 결정
- Skip Layer
 - (convolution+pooling) 과정을 여러 단계 거치면서 feature map의 크기가 작아지면 detail한 부분이 사라지는 효과 발생
 - 최종 과정보다 앞선 과정에서 생성된 결과를 사용하여 detail을 보강하자는 의미로 사용

Fully Convolutional Network (3)



U-Net (1)

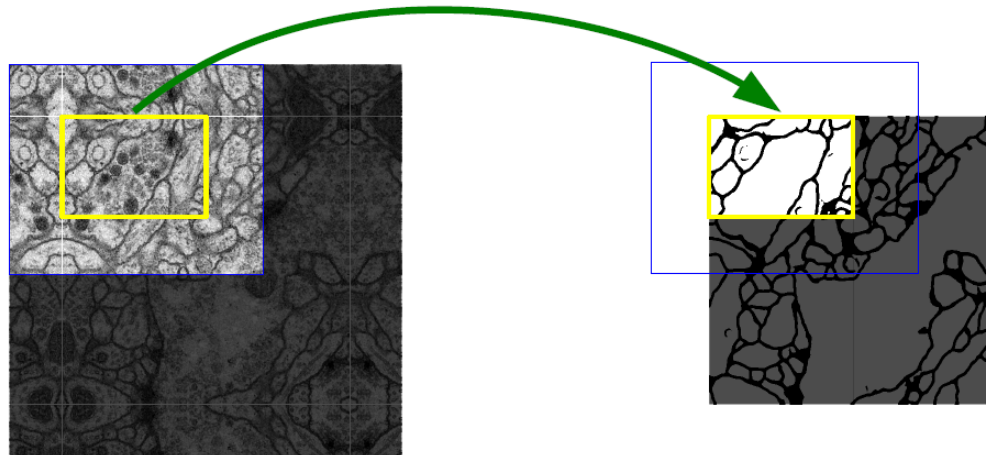
■ Architecture



U-Net (2)

■ Overlay-tile strategy

- 전체 이미지를 tile로 구성한 후 각 tile에 대한 segmentation 수행
 - 실제 segmentation 예측 부분보다 더 큰 영역을 Input을 사용
 - 입력 이미지의 경계 부분은 Mirror padding 방법 적용 → 그림 경계의 바깥 부분을 0으로 채우는 것이 아니라 원본을 그대로 복사하여 사용



U-Net (3)

■ Loss function

- 일반적인 cross entropy loss를 대상으로 weight를 추가함

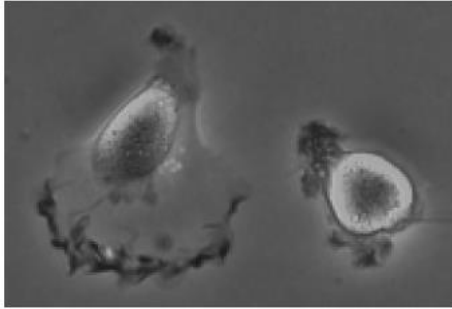
$$E = \sum_{\mathbf{x} \in \Omega} w(\mathbf{x}) \log(p_{\ell(\mathbf{x})}(\mathbf{x}))$$

$$w(\mathbf{x}) = w_c(\mathbf{x}) + w_0 \cdot \exp\left(-\frac{(d_1(\mathbf{x}) + d_2(\mathbf{x}))^2}{2\sigma^2}\right)$$

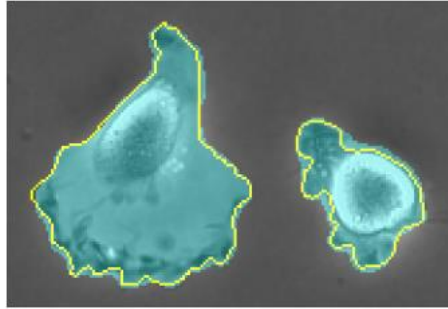
- Weight의 역할은 경계 영역에 존재하는 픽셀에 대해 좀 더 높은 weight값을 줌으로써 경계 영역에 대한 학습을 강화시킴
 - $W(\mathbf{x})$ 는 각각의 ground truth 이미지를 대상으로 pre-computed된 값임

U-Net (4)

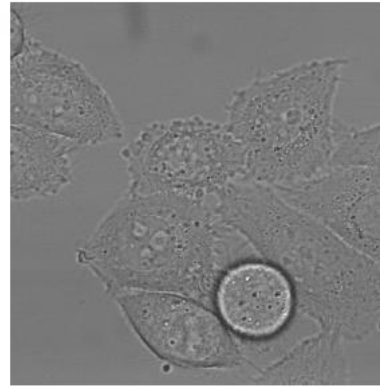
a



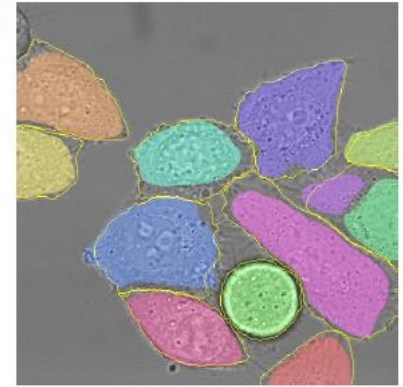
b



c



d



References

- Deep Learning for Generic Object Detection: A Survey, <https://link.springer.com/article/10.1007/s11263-019-01247-4>
- Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, <https://arxiv.org/pdf/1506.01497.pdf>
- Faster R-CNN, <http://incredible.ai/deep-learning/2018/03/17/Faster-R-CNN/>
- SSD: Single Shot Multibox Detector, <https://arxiv.org/pdf/1512.02325.pdf>
- SSD: Single Shot Multibox Detector 분석, <https://taeu.github.io/paper/deeplearning-paper-ssd/>
- NMS (Non-Maximum Suppression), <https://dyndy.tistory.com/275>
- Assisted Excitation of Activations: A Learning Technique to Improve Object Detectors, <https://arxiv.org/pdf/1906.05388.pdf>
- Fully Convolutional Networks for Semantic Segmentation, https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf
- CNN – Transposed, Dilated, Casual, <https://dataplay.tistory.com/29>
- U-Net: Convolutional Networks for Biomedical Image Segmentation, <https://arxiv.org/pdf/1505.04597.pdf>

**ANY
QUESTIONS?**