Mask R-CNN

Kaiming He, Georgia Gkioxari, Piotr Dollar, Ross Girshick; Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2961-2969

Facebook AI Research(FAIR)

한지수

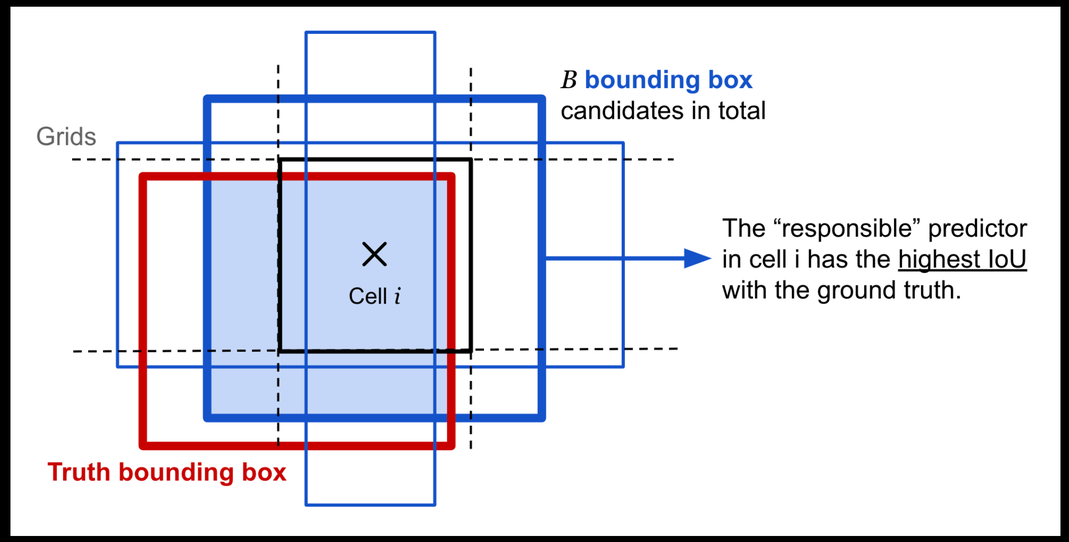
그림 출처: 특별히 명시되지 않은 한 논문 바탕 그림들

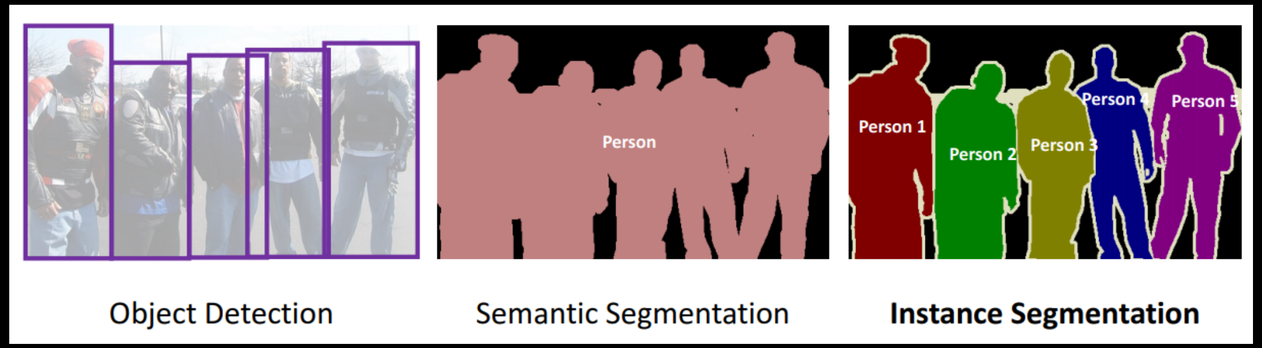
Contents

- 용어 정리
- 기존의 논문들
  - FCN
  - R-CNN, Fast R-CNN, Faster R-CNN
- Mask R-CNN

용어 정리

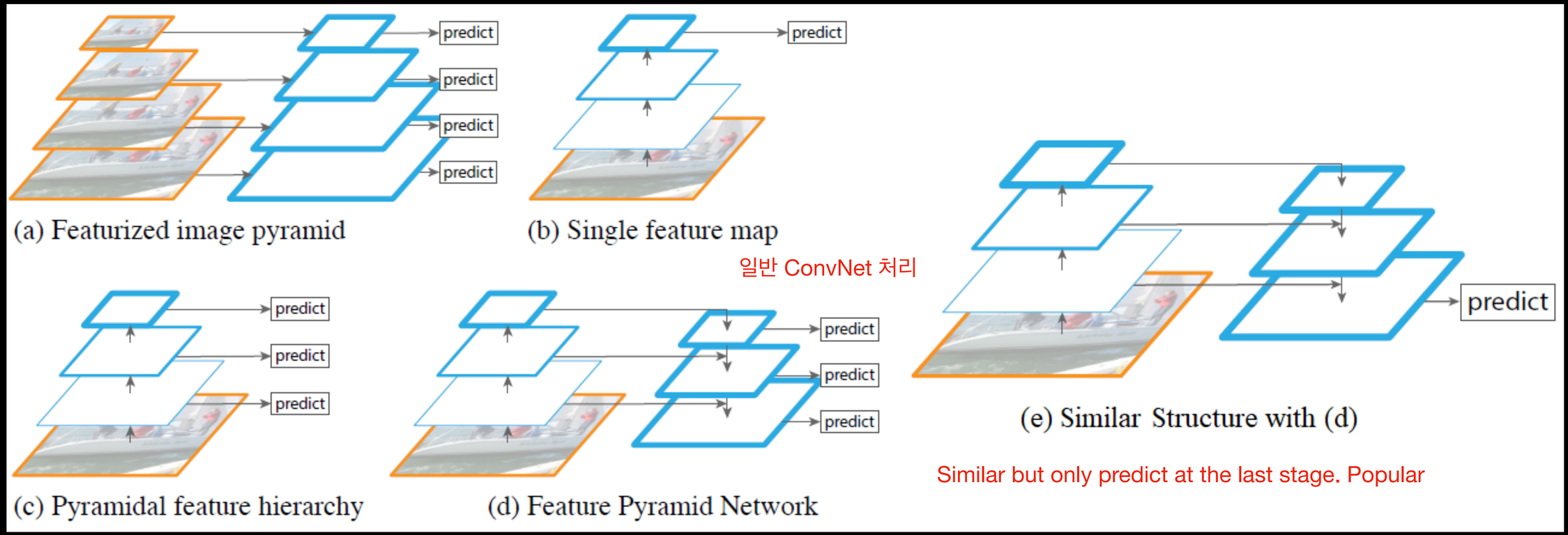
- RoI: Region of Interest < 후보 영역
- Bounding box offset: 각 cell(feature map 한 칸)을 기준으로 한 상대적 위치와 **박스**의 크기를 의미한다. 이를 위해 필요한 정보는 x, y, width, height 로 4개이다
  - (Ground-)truth bounding box: 실제 bounding box
- Instance Segmentation: Correct detection of all objects in an image while also precisely segmenting each instance.





Feature Pyramid Network(FPN)

- Top-down architecture with lateral connections



Misses the detection for small objects

Top-down + skip connection
: Combines low-resolution,
semantically strong features with high-resolution

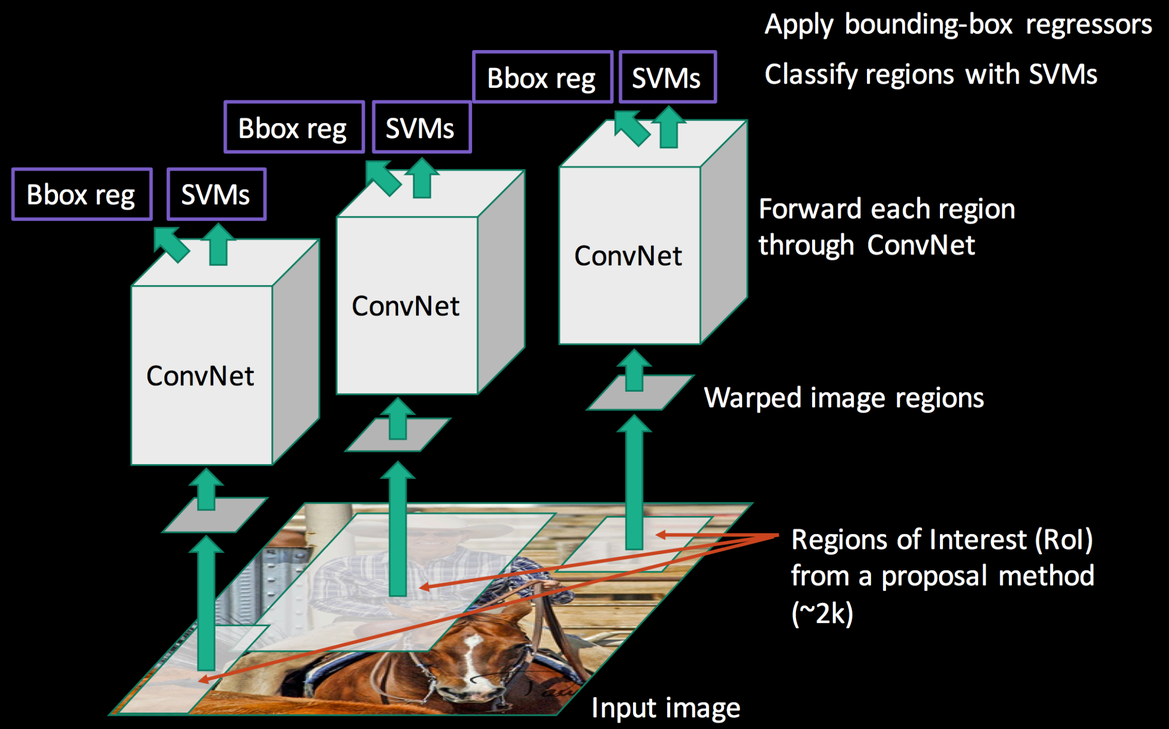
Similar but only predict at the last stage. Popular

R-CNN

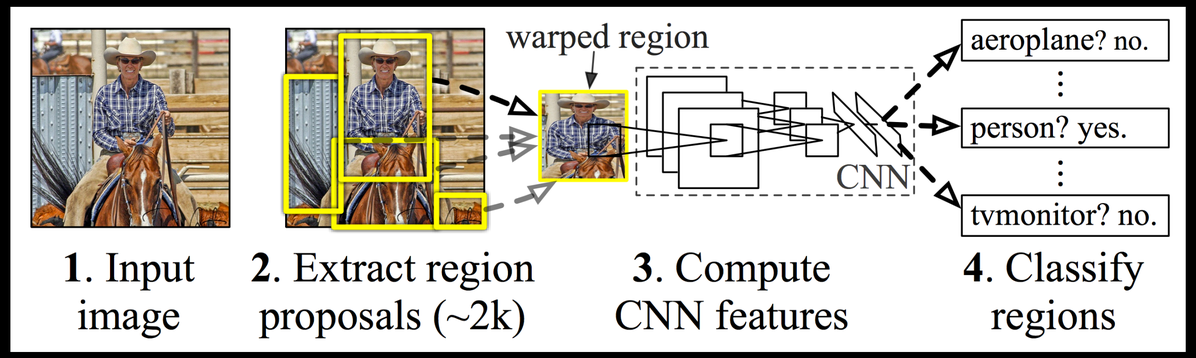
Object Detection

1. Input Image
2. Region Proposal
    1. RoI Projection: Selective Search
    2. RoIPool(RoI Pooling layer): max-pooling 으로 유효한 RoI 내부의 feature를 고정된 feature map으로 변환
3. ConvNet
4. Classify Region: (Category-specific linear) SVMs & Bbox Reg

$$L = L_{cls}, L_{box}$$



Girshick et al. CVPR14.



R-CNN

Object Detection

1. Input Image
2. Region Proposal
    1. RoI Projection: Selective Search

region들 간의 (color, texture, size, fill)유사도를 바탕으로 조사 but CPU만. GPU 못 사용, 오래 걸림

    2. RoIPool(RoI Pooling layer): max-pooling으로 유효한 RoI 내부의 feature를 고정된 feature map으로 변환

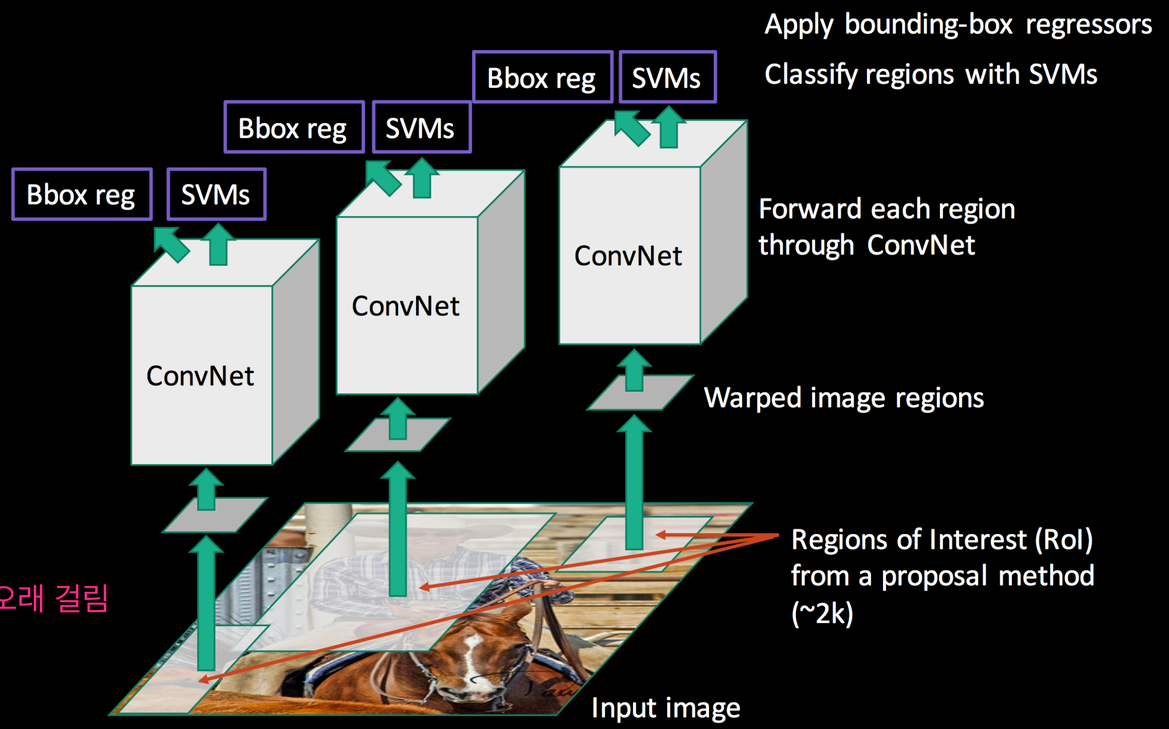
Feature 뽑는데 제약

3. ConvNet 하나하나 처리하기에는 너무 오래 걸림
4. Classify Region: (Category-specific linear) SVMs & Bbox Reg

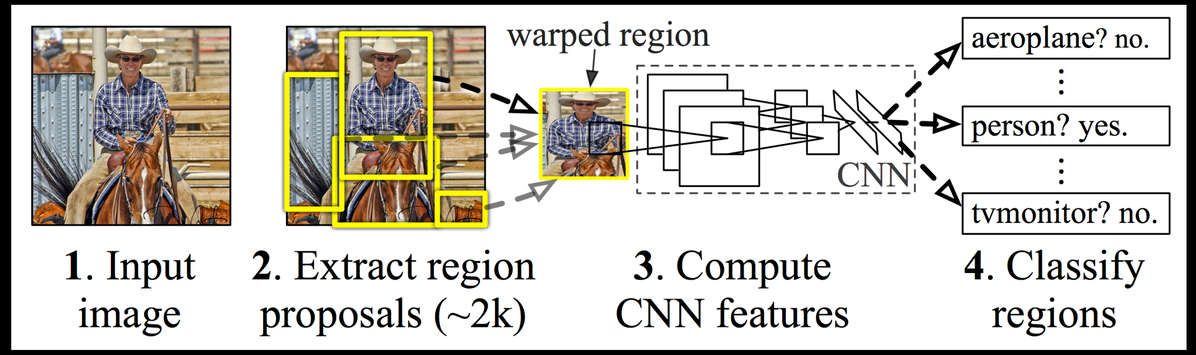
Bounding box가 Ground Truth에 맞추도록 transform <덕분에 3가지 모델 < 복잡

$$\{(P^i, G^i)\}_{i=1,\ldots,N}, \text{where } P^i = (P^i_x, P^i_y, P^i_w, P^i_h)$$

각각 계산



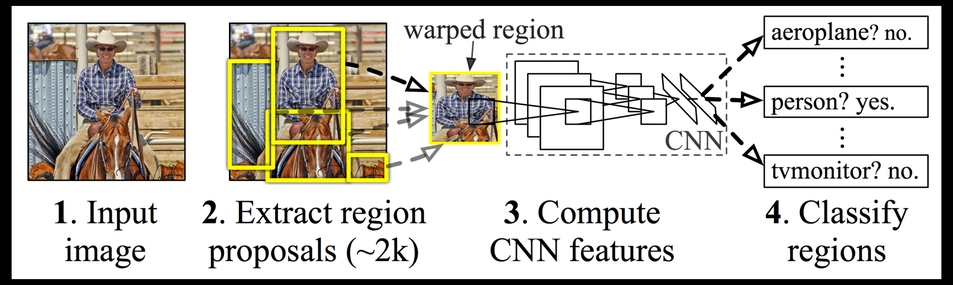
Girshick et al. CVPR14.



Fast R-CNN
Object Detection

1. Input Image
2. Region Proposal
    1. ConvNet 처리
    2. RoI Projection: Selective Search
    3. RoI Pooling Layer
3. Classify Region: SVMs & Bbox Reg

$$L = L_{cls} + L_{box}$$



R-CNN

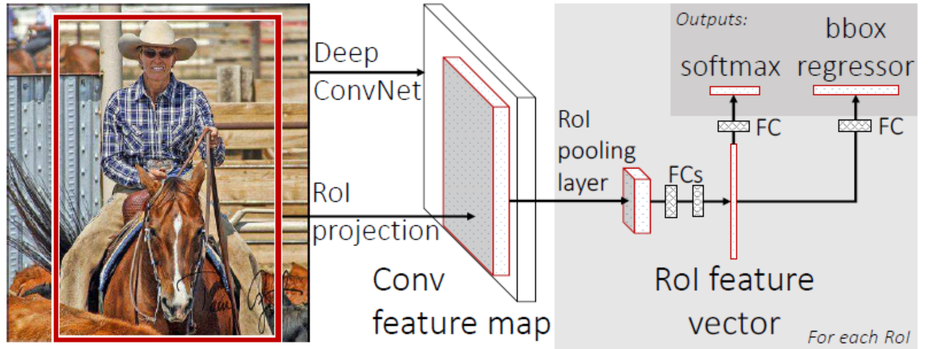


Figure 1. Fast R-CNN architecture. An input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers (FCs). The network has two output vectors per RoI: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.

Fast R-CNN

Fast R-CNN
Object Detection

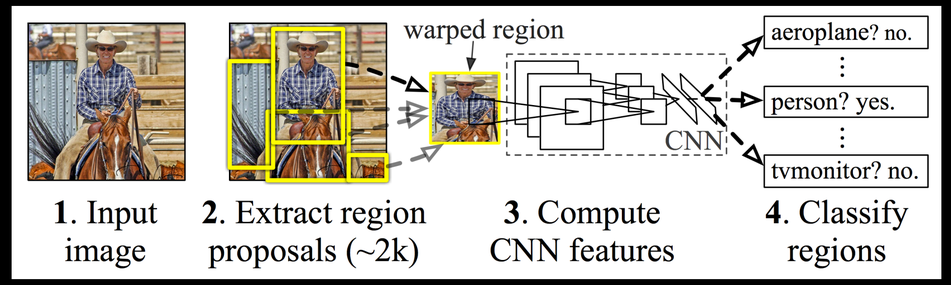
1. Input Image
2. Region Proposal
    1. ConvNet 처리 ConvNet을 먼저 처리하여 연산 줄임
    2. RoI Projection: Selective Search

region들 간의 (color, texture, size, fill)유사도를 바탕으로 조사 but CPU만. GPU 못 사용, 오래 걸림

    3. RoI Pooling Layer
3. Classify Region: SVMs & Bbox Reg

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

따로따로 학습 >> 두 Loss를 더하여 학습



R-CNN

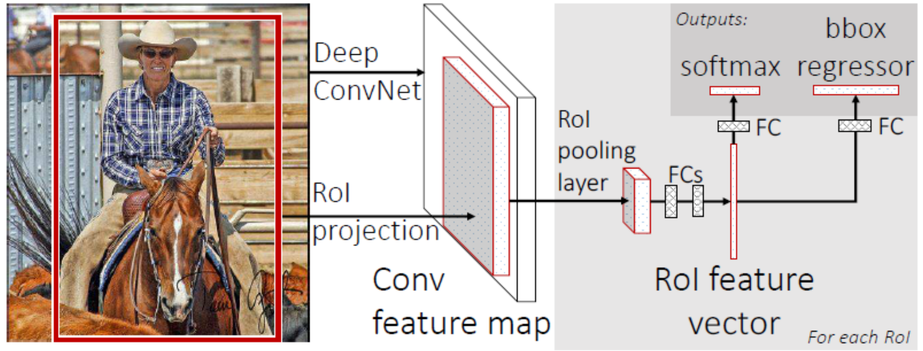


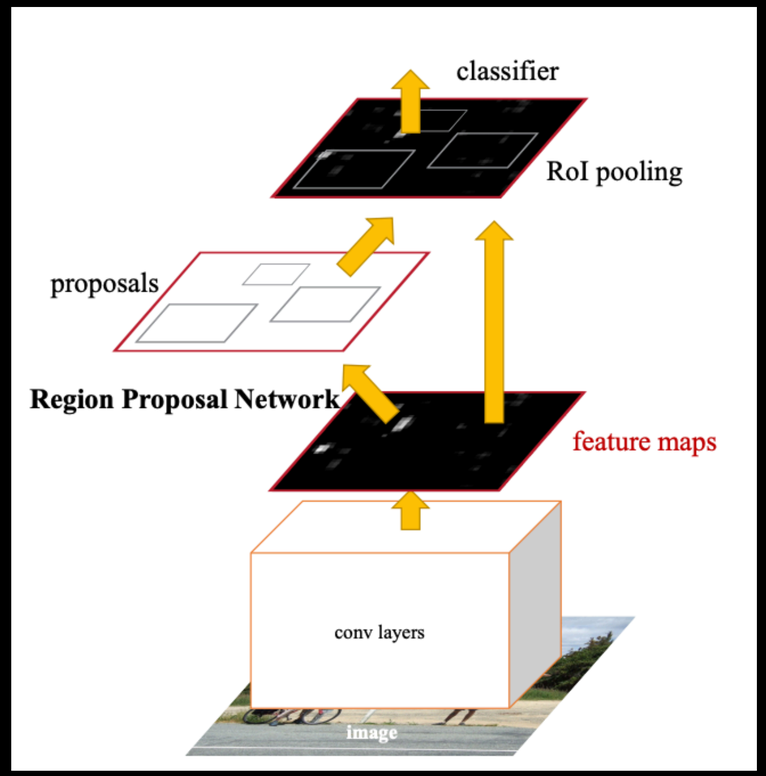
Figure 1. Fast R-CNN architecture. An input image and multiple regions of interest (RoIs) are input into a fully convolutional network. Each RoI is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers (FCs). The network has two output vectors per RoI: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.

Fast R-CNN

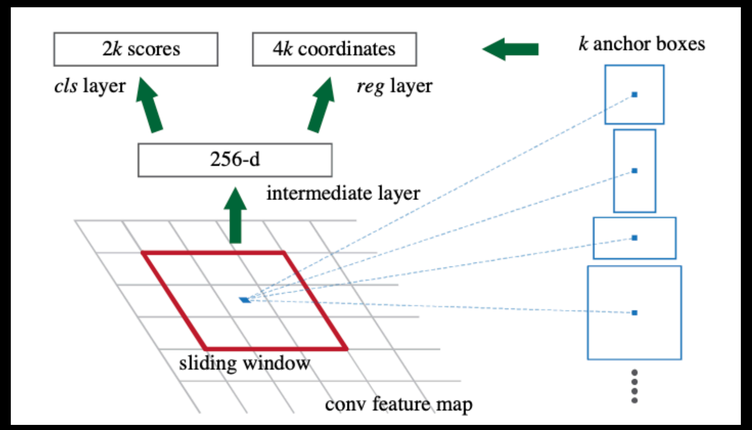
Faster R-CNN
Object Detection

1. Input Image
2. Conv Feature Map
    1. ConvNet 처리
    2. RoI Proposal Network (RPN)
    3. RoI Pooling Layer
3. Classify Region: SVMs & Bbox Reg

$$L = L_{cls} + L_{box}$$

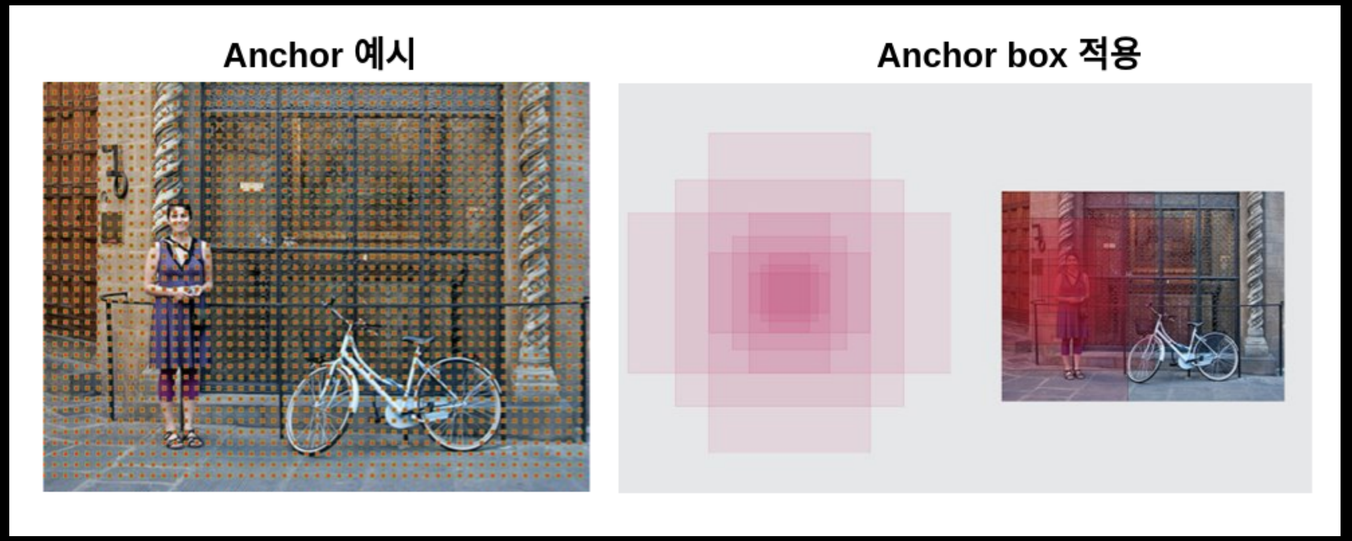


Faster R-CNN

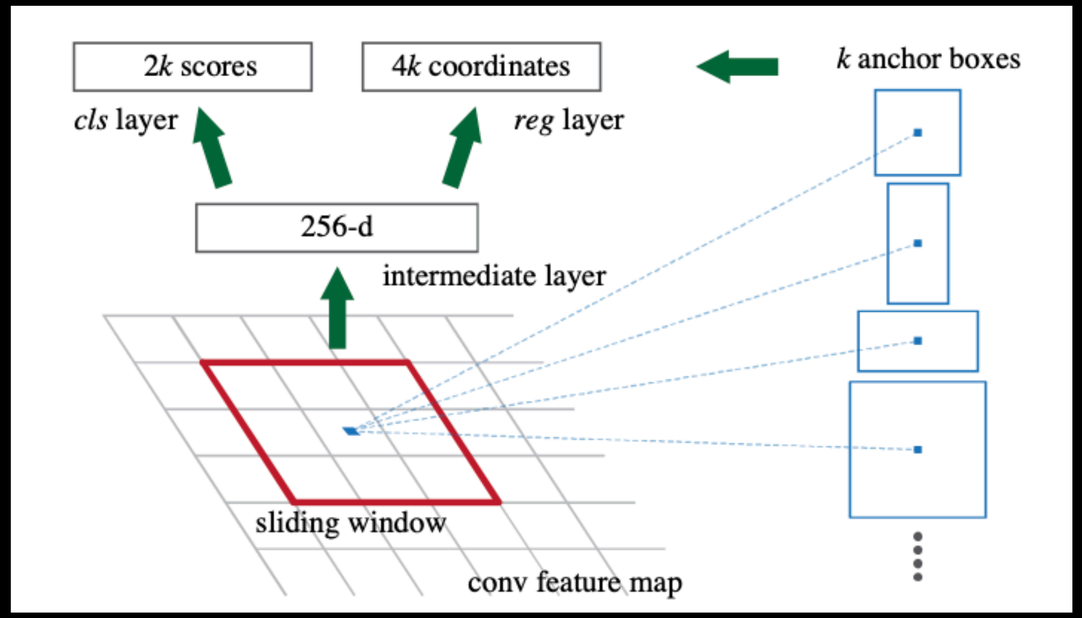


RPN

Faster R-CNN - RPN



출처: https://chacha95.github.io/2020-02-14-Object-Detection2/



Input: Feature Maps —> cls Layer, reg layer

Positive Anchor box 찾기

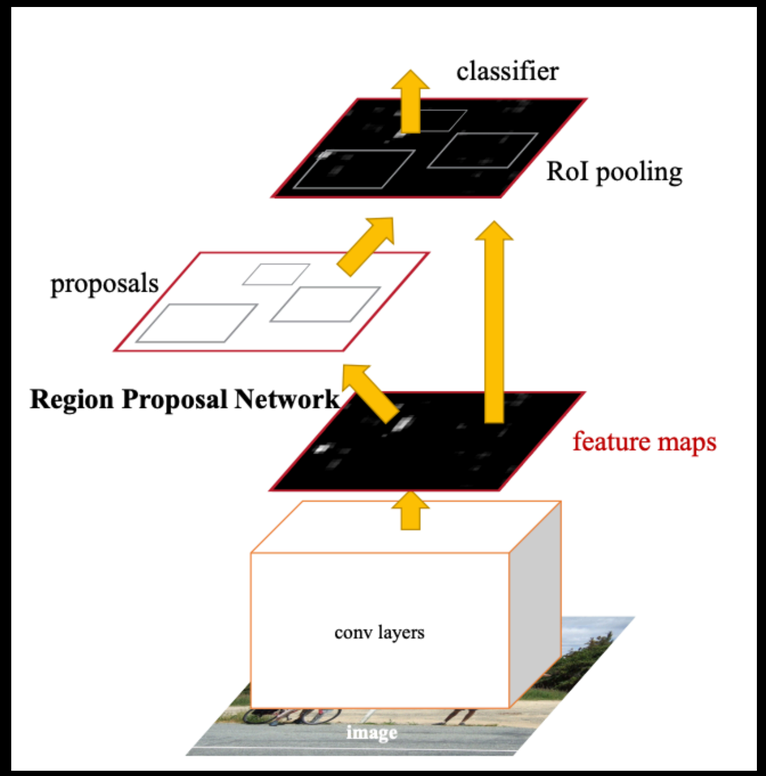
- Positive anchor box: GT box와 IoU가 0.7이상
- Negative anchor box: GT box와 IoU가 0.3이하
- 그 외는 사용 x

Multi-scale 학습 가능

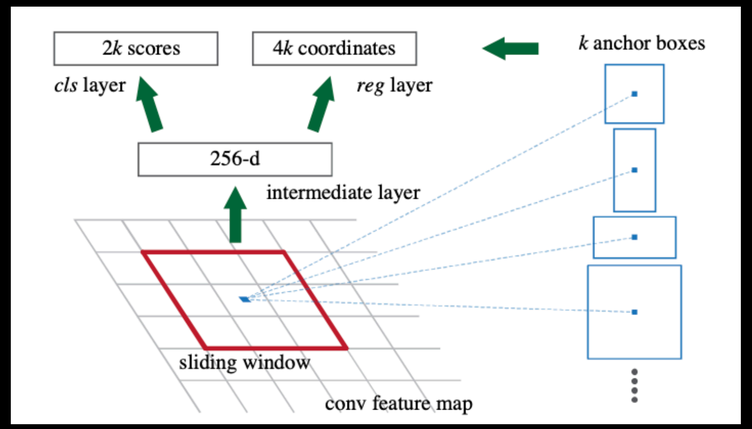
Faster R-CNN
Object Detection

1. Input Image
2. Conv Feature Map
    1. ConvNet 처리
    2. RoI Proposal Network (RPN)
    3. RoI Pooling Layer
3. Classify Region: SVMs & Bbox Reg

$$L = L_{cls} + L_{box}$$



객체가 있는 영역인 positive anchor box 찾기 / GPU 사용 > 성능 증가

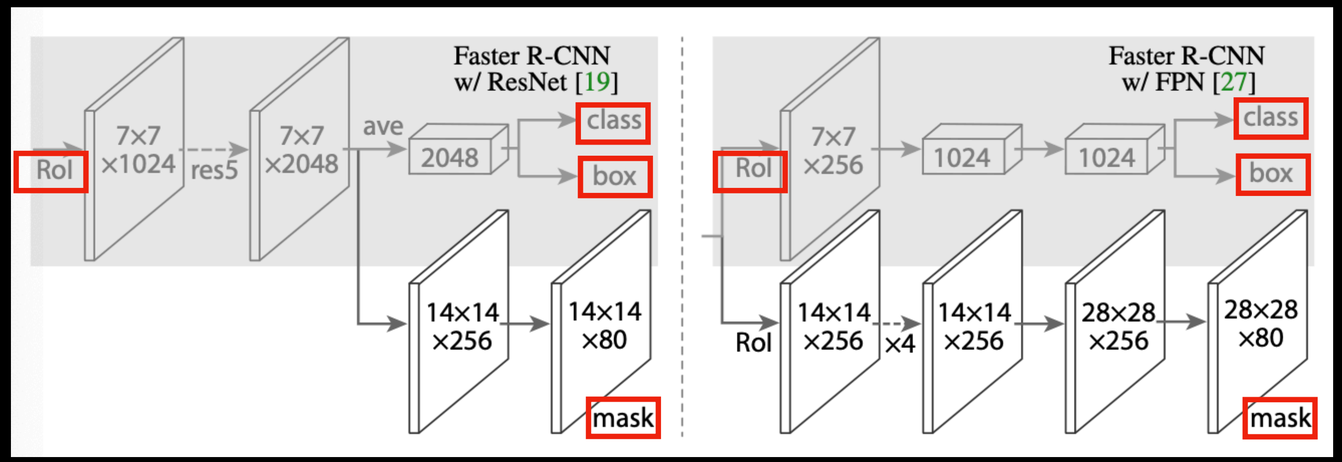


Mask R-CNN
Instance Segmentation

Faster R-CNN + Mask Branch

1. Input Image
2. Conv Feature Map
    1. ConvNet 처리
    2. RoI Proposal Network (RPN)
    3. RoI Pooling < RoIAlign
3. Classify Region: SVMs & Bbox Reg & Mask

$$L = L_{cls} + L_{box} + L_{mask}$$

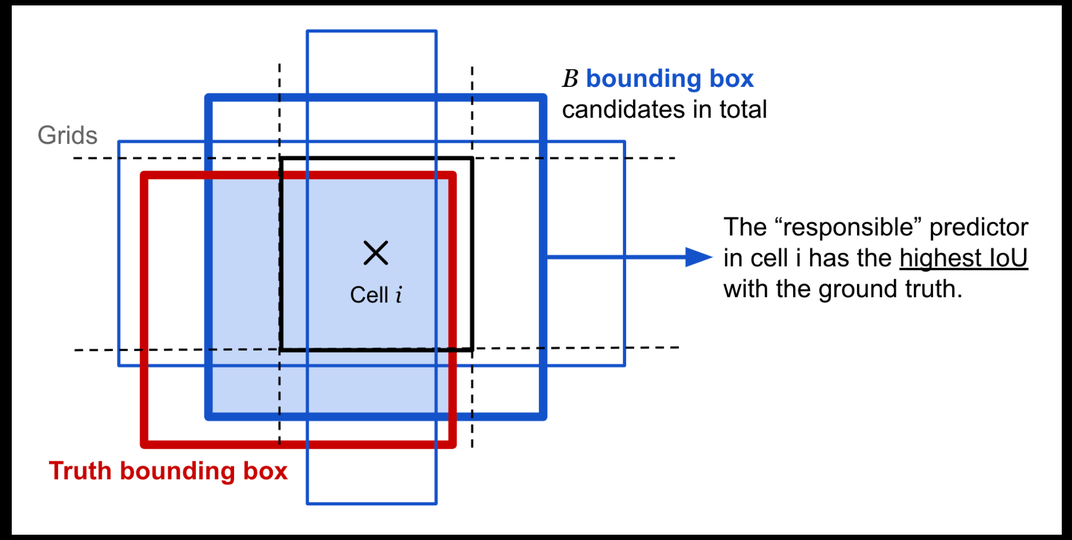


Mask R-CNN 변화

- Extends Faster R-CNN by adding a branch for predicting segmentation masks on each RoI
  - ***Mask branch***
    - Small FCN applied to each RoI
    - Binary mask for each class independently
  - ***RoIPool***: preserves exact spatial location
    - Stricter localization metrics: Mask accuracy by relative 10% to 50%

Mask R-CNN — Mask branch

- Encodes an input object's spatial layout
- L_{mask} (Km^2): K binary masks of resolution $m \times m$, one for each of the K classes
  - Binary mask for each class independently: 각 픽셀이 오브젝트에 해당하는 건지 아닌지를 Masking하는 네트워크
  - Mask target: intersection btw an RoI and its associated ground-truth mask
- 변화: Into Short output vectors(fc layers) —> $m \times m$ mask from each RoI: without collapse of spatial dimensions



Mask R-CNN — RoI Align

- 기존의 RoI Pooling Layer: max-pooling or average pooling
- 새로나온 RoIAlign: for extracting a small feature map($m \times m$ mask) from each RoI
  - Avoid quantization of the RoI boundaries or bins

    —> bilinear interpolations (using max or average)
  - RoIWrap

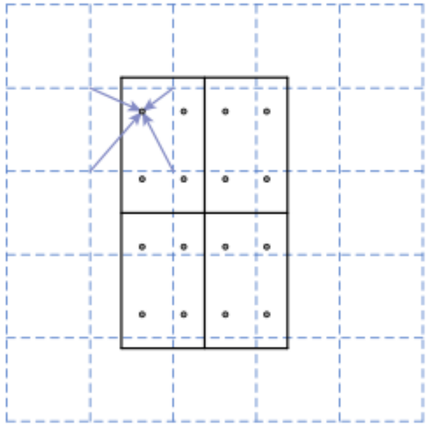


Figure 3. **RoIAlign:** The dashed grid represents a feature map, the solid lines an RoI (with 2×2 bins in this example), and the dots the 4 sampling points in each bin. RoIAlign computes the value of each sampling point by bilinear interpolation from the nearby grid points on the feature map. No quantization is performed on any coordinates involved in the RoI, its bins, or the sampling points.

감사합니다