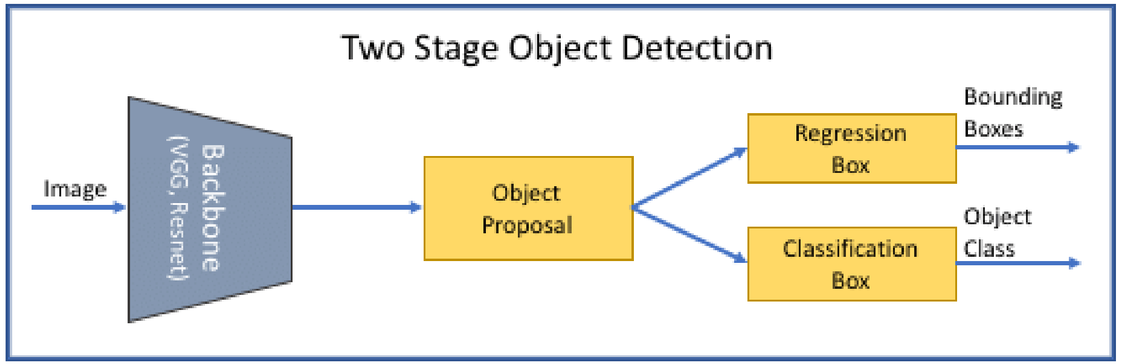
시각지능 A – Object Detection

2-stage Detector

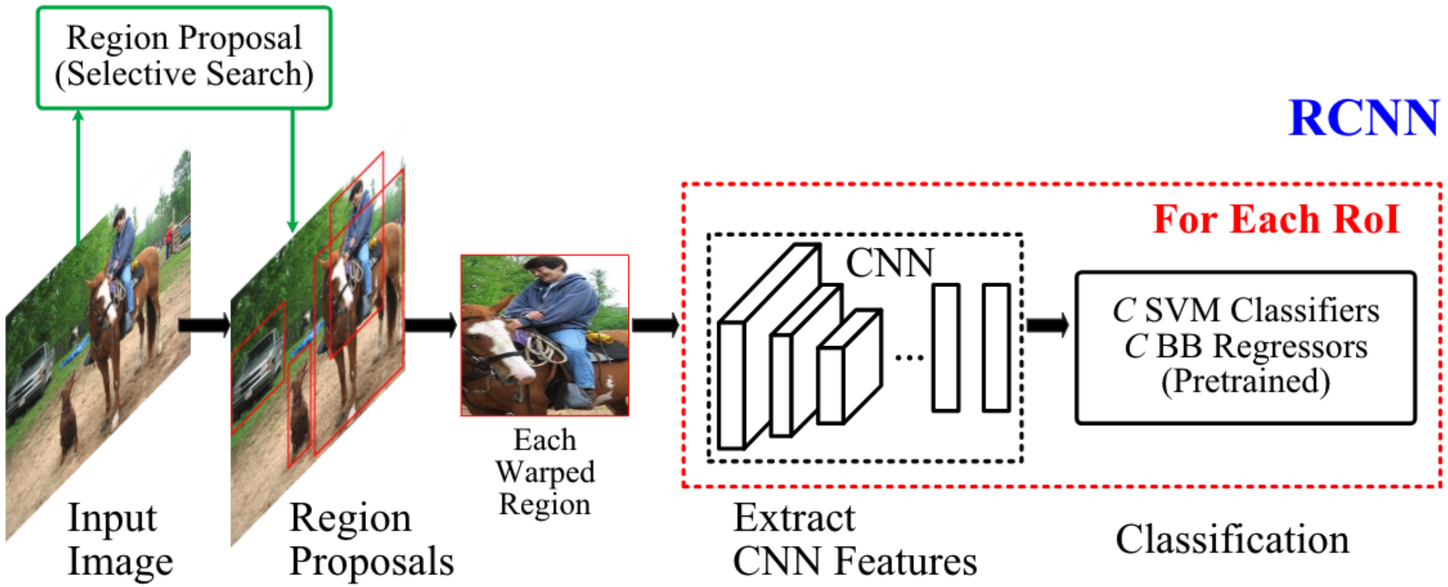
2-Stage Detector

- 2-Stage detector
  - 분류와 Region proposal 두 단계를 순차적으로 해결하는 알고리즘
  - 순서
    - Region proposal, Selective search 등의 방법을 통해 많은 수의 후보 영역을 생성
    - 이들을 분류하고, 박스를 선별하여 크기와 위치를 정확하게 조정
  - 이미지를 두 단계에 걸쳐 정교하게 처리하므로, 정확도와 박스의 위치 및 크기 예측 성능이 우수함
  - 그러나 속도가 매우 느리며, 모델의 구조가 상대적으로 복합하기에 최적화가 어려움



Semantic Image Cropping, cs.CV(2021)

R-CNN

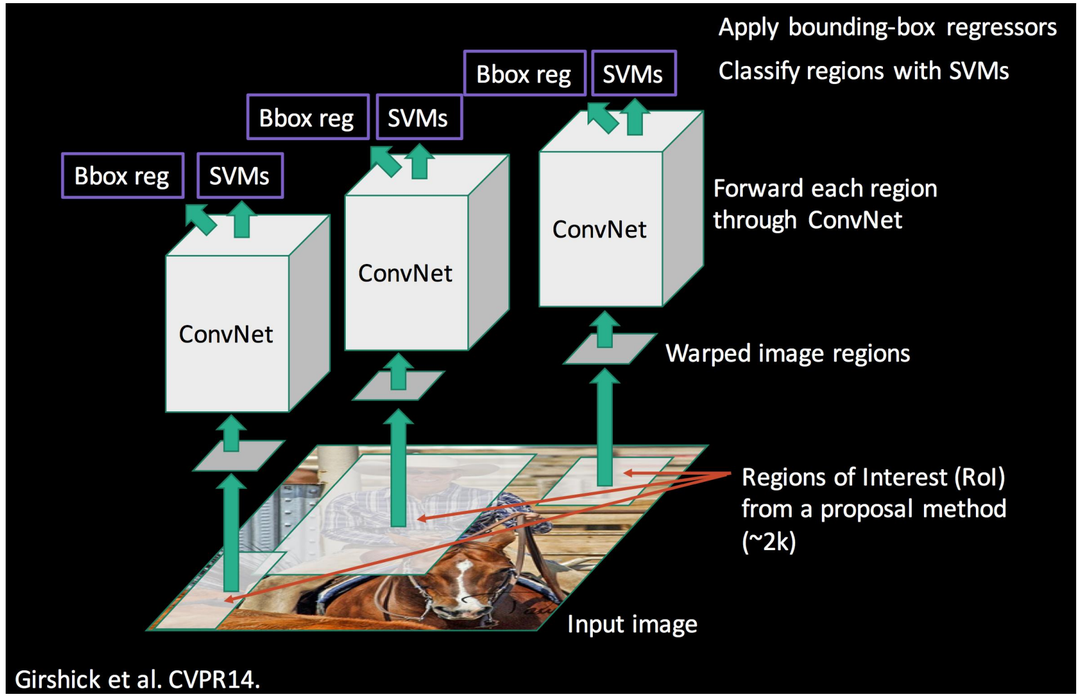


- R-CNN(2014)
  - Region-based CNN
  - 딥러닝 객체 탐지 모델의 포문을 연 2-stage detector
  - 이미지 분류 문제에서 뛰어난 성능을 보여준 AlexNet 활용
  - 이에 Selective search 방식을 첨가하여 후보 영역 검출

Deep Learning for Generic Object Detection: A Survey, cs.CV(2018)

R-CNN

- R-CNN 작동 원리
  - Selective search를 통하여 객체가 포함될 가능성이 있는 영역 제안
    - 이러한 영역을 RoI(Region of Interest)라 함
    - 이미지 당 최대 2000개의 RoI 제안
  - 각 RoI의 크기를 균일하게 조정하여 AlexNet의 Feature extractor에 입력
  - 처리된 Feature map을 SVM에 통과시켜 이미지를 분류
  - 처리된 Feature map을 Bounding box regressor 알고리즘에 입력하여 바운딩 박스의 정보(xywh) 추출



Region Searching – Selective Search

- Selective search
  - 모든 이미지를 아날로그 방식으로 탐색하는 대신, 픽셀 값을 통해 위치를 추정
  - RoI(Region of Interest): 객체가 있을 것으로 추정되는 영역
  - CPU에서 수행됨
- 방법
  - 각 픽셀의 값을 기준으로 유사한 픽셀을 군집화
  - 작은 군집들을 다시 모아 대표적인 큰 군집 형성
  - 군집의 수가 충분히 줄어들 때까지 이를 반복
    - 2000여개(R-CNN)
  - 이 또한 지나치게 많은 Bbox가 생성됨

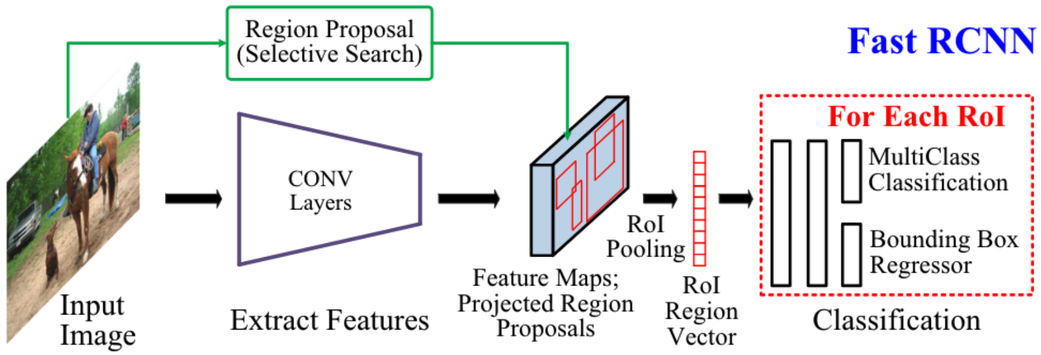
https://wikidocs.net/136492

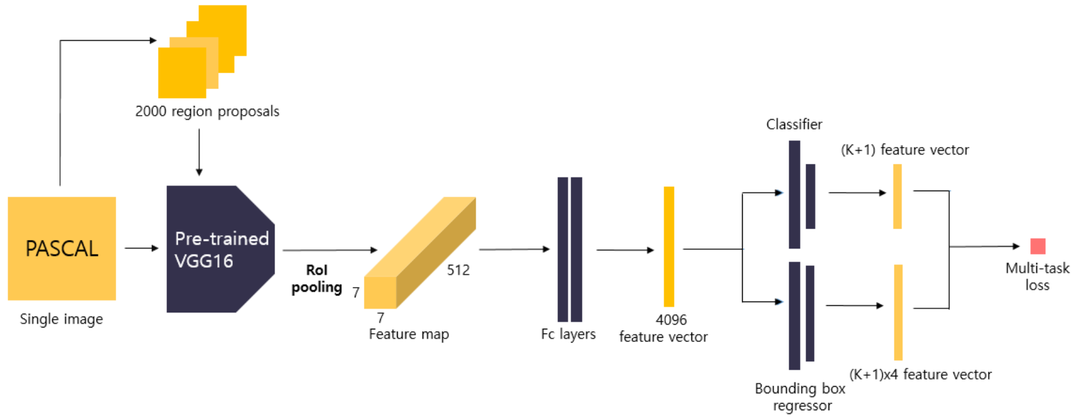
R-CNN

- R-CNN의 한계
  - 여러 단계의 복잡한 파이프라인을 통과하여 훈련이 진행되므로 최적화가 어려움
    - 일반적인 딥러닝 모델은 입력부에서 출력부까지 모든 범위가 한 번에 훈련
    - 그러나 R-CNN의 경우 Selective search, SVM, Bounding box regressor 등의 알고리즘을 따로 훈련
    - Selevtive search는 신경망 학습과 별도로 CPU 상에서 진행됨
  - SVM, Bounding box regressor은 기존 AlexNet에서 사용되지 않은 알고리즘으로, AlexNet을 통과한 특징이 이들에 의해 잘 처리될 것이라는 증명이 불가능
    - 또한 이들은 학습하는 과정에 있어서 디스크 공간과 시간이 많이 소요됨
  - 학습 후 추론 과정에 있어서도 지나치게 긴 시간이 소요됨
    - 현업에서 사용하기 열악한 조건

Fast R-CNN

- Fast R-CNN(2015)
  - R-CNN을 여러 측면에서 개선
    - Feature extractor에 2000여개의 RoI가 아닌 이미지 한 장을 입력
    - Feature map에 Region proposal 정보를 합하여 RoI pooling으로 결과물을 통합하여 정확도 상승
    - 각 RoI 별 클래스 분류와 Bbox 예측을 단일 손실 함수를 통해 진행하므로 학습의 간편화
  - 속도 개선
    - 훈련의 경우 3배, 테스트의 경우 10배
  - 그러나 Region proposal 과정이 신경망 외부에서 진행되기에, 빠른 연산을 방해하는 병목 작용

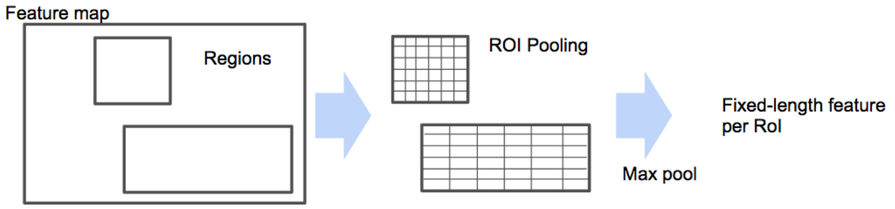




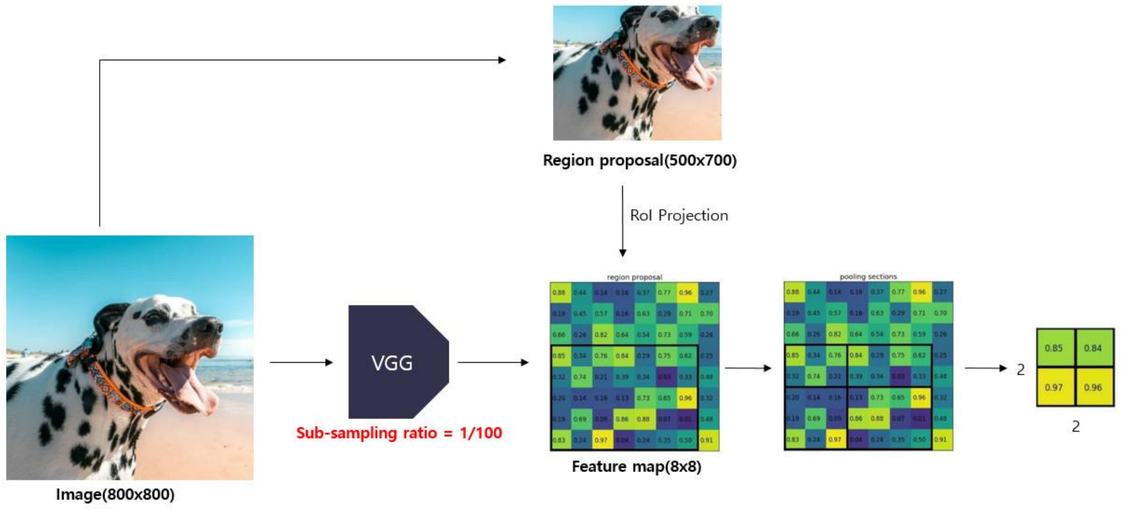
Deep Learning for Generic Object Detection: A Survey, cs.CV(2018),
https://herbwood.tistory.com/8

Fast R-CNN

- RoI Pooling
  - Classification/Regression은 Linear layer에서 연산됨



  - 이 과정을 수행하기 위해 일반적으로 최종 출력 Layer 전에 Flatten과 1~2개의 추가 Linear layer를 거침
    - 이는 입력 벡터의 크기를 고정시키기 위함
  - 그러나 객체의 예상 위치를 알려주는 Region proposal은 다양한 크기의 지역들을 제안함
  - 정이



0.54	0.77	0.81	0.73	0.25	0.56	0.66	0.71
0.13	0.45	0.84	0.25	0.62	0.73	0.45	0.56
0.16	0.88	0.32	0.22	0.31	0.67	0.65	0.91
0.83	0.55	0.65	0.43	0.17	0.55	0.30	0.72
0.17	0.51	0.08	0.37	0.55	0.62	0.54	0.43
0.32	0.23	0.43	0.59	0.19	0.93	0.71	0.75
0.55	0.69	0.94	0.72	0.68	0.42	0.81	0.49
0.68	0.37	0.51	0.14	0.52	0.73	0.68	0.35

Max Pooling

0.88	0.67
0.94	0.93

https://herbwood.tistory.com/8

Fast R-CNN

- RoI Pooling

```
[ ]  import torch
     import torchvision
     from torchvision.ops import roi_pool

     # 입력 특징 맵 (예시로 1장의 이미지에서 나온 특징 맵, 크기 [1, 512, 50, 50])
     features = torch.randn(1, 512, 50, 50)

     # RoI: [batch_idx, x1, y1, x2, y2]로 표현된 2개의 Region of Interest (RoI)
     rois = torch.tensor([[0, 10, 20, 30, 40], [0, 20, 30, 40, 50]], dtype=torch.float32)

     # RoI Pooling을 적용하여 출력 크기를 7x7로 설정
     output = roi_pool(features, rois, output_size=(7, 7))

     print(output.shape)  # 출력 형태는 [2, 512, 7, 7]로 RoI의 수만큼 나옴
```

Fast R-CNN

- Multi-task Loss

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

  - $p = p(p_0, \ldots, p_k)$: Class score(K+1개) Prediction
  - u: Class Label
  - $t^u = (t^u_{\mathrm{x}}, t^u_{\mathrm{y}}, t^u_{\mathrm{w}}, t^u_{\mathrm{h}})$: Bbox 좌표 Prediction
  - $v^u = (v_{\mathrm{x}}, v_{\mathrm{y}}, v_{\mathrm{w}}, v_{\mathrm{h}})$: Bbox 좌표 Label
  - $L_{cls}(p, u) = -\log p_u$: Classification loss
  - $L_{loc}(t^u, v) = \sum_{i \in \{x,y,w,h\}} smooth_{L1}(t^u_i - v_i)$: Smooth regression L1 loss
  - λ: 손실 함수 간 균형 조정을 위한 가중치

R-CNN to Fast R-CNN

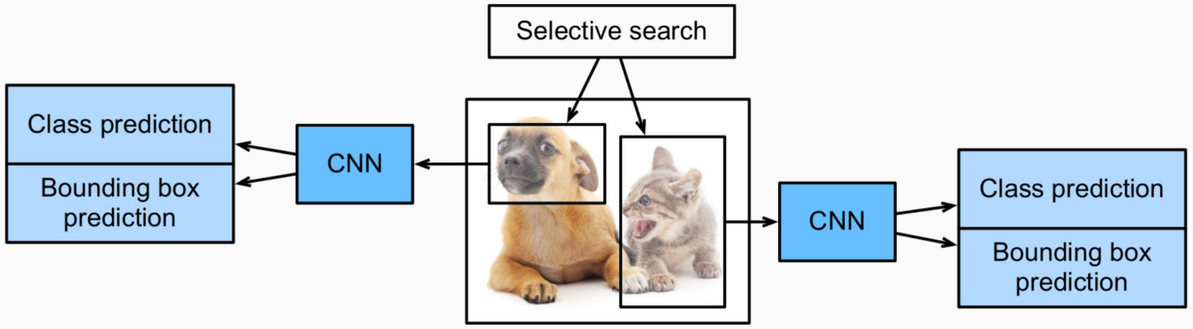


Fig. 14.8.1 The R-CNN model.

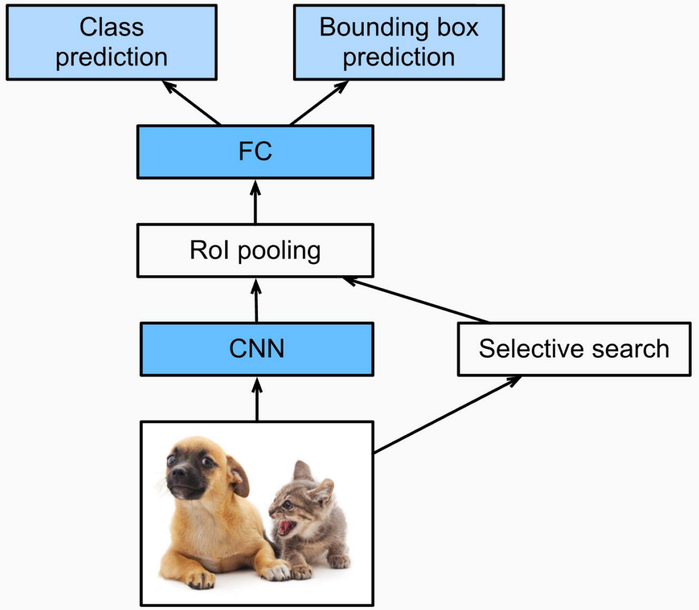
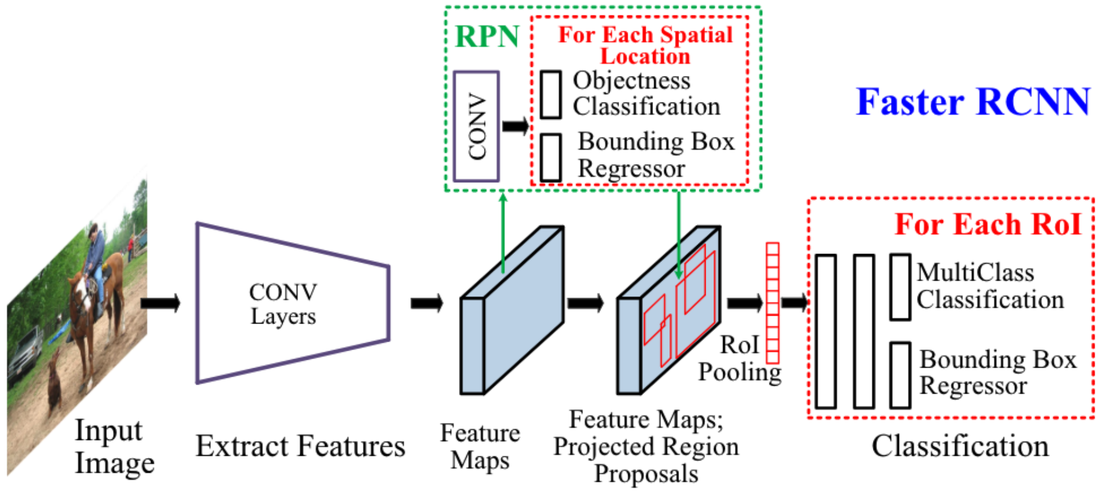


Fig. 14.8.2 The fast R-CNN model.

- 이미지 분류 알고리즘 변화
- RoI 당 CNN 적용 -> 전체 이미지 1장에 대한 CNN 적용
- Feature map 수준에서의 정보 추출

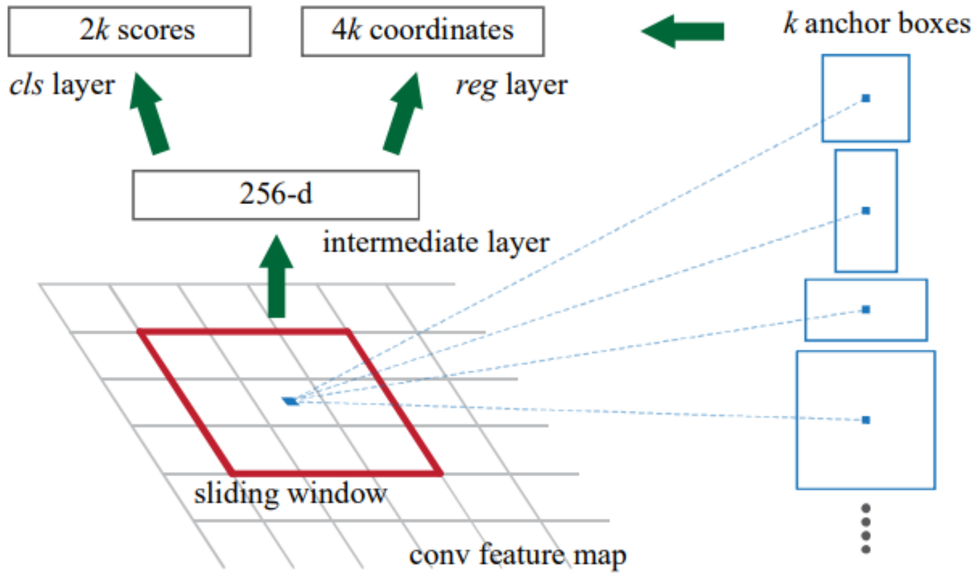
Faster R-CNN

- Faster R-CNN(2016)
  - Fast R-CNN의 구조를 개선하여 속도 향상
  - 신경망 외부에서 진행되는 Region proposal 작업을 신경망 내부로 통합
  - RPN(Region Proposal Network) 형성
    - 빠른 연산을 위해 Bbox를 제한된 수와 크기 만큼만 생성(Anchor box)
    - Bbox마다 객체 포함 여부와 크기 조정을 거침
    - Bbox 중 의미없는 것들을 제거함(NMS)
- 모델의 정확도와 학습 속도 개선
  - 그러나 5 FPS 처리 속도로 Real-time 분석에는 부적



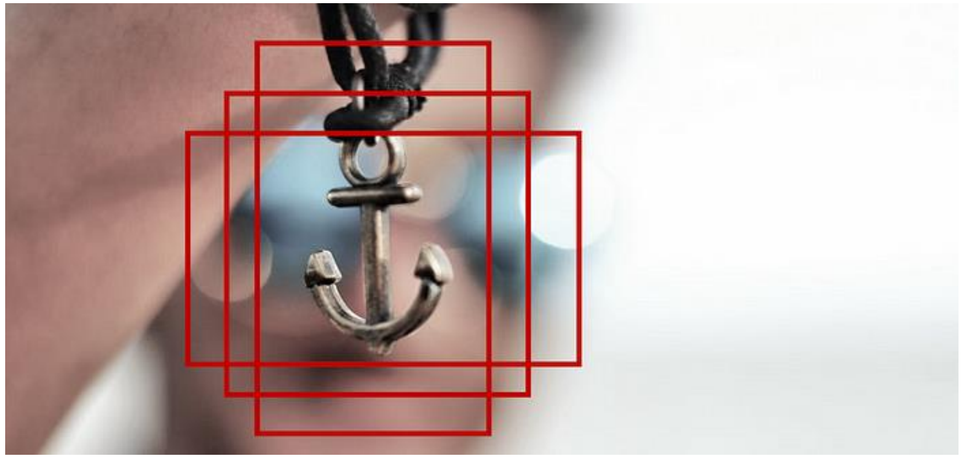
Faster R-CNN

- RPN(Region Proposal Network)
  - 이미지 내에서 객체가 있을 가능성이 높은 영역(후보 영역 또는 proposals)을 제안하는 역할
  - 기존의 Selective search를 대체하는, 신경망을 이용한 RoI 탐색과정
  - RoI 추출 과정을 GPU에 포함시키는 과정
    - End-to-end learning
  - 이미지를 격자로 나누고, 이마다 RoI를 탐색
    - CNN의 마지막 Feature map을 주로 사용
    - ResNet50: (3, 224, 224) → (2048, 14, 14)
    - Feature map의 가로*세로 1 칸마다 Anchor box 생성



Faster R-CNN

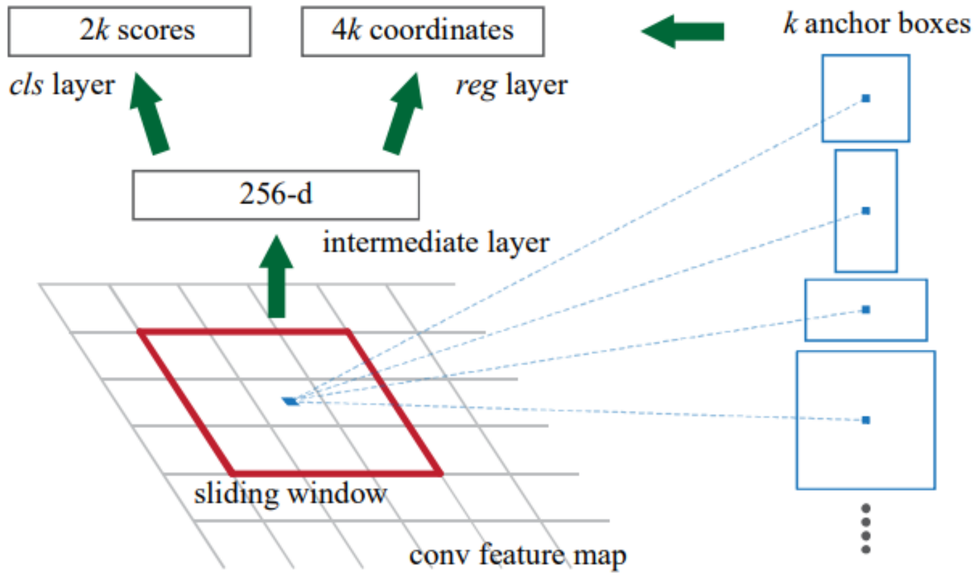
- Anchor box
  - 기존 Bbox 제안 방식의 문제점 중 다른 하나는 지나치게 많은 박스가 생성되는 것
  - 신경망이 이미지에서 객체를 탐지할 때에는 대상의 갯수를 고려하지 못하고 여러 박스를 생성
    - 고양이를 수염, 눈, 발톱으로 인식할 경우 객체 하나에서 여러 개의 Bbox가 제안됨
  - 또한 서로 다른 객체가 겹칠 때, 중심점이 가까울 경우 다른 대상의 Bbox를 제거할 수도 있음
    - 객체가 작거나 무리를 이룰 때(비둘기, 식물 등)자주 발생
  - 특정 구역에서 생성될 수 있는 Bbox의 수와 형태를 사전에 지정하는 것
    - Bbox의 생성 수가 줄어들기에 속도 개선
    - 박스의 좌표와 크기 변화가 작아지므로 학습 쉬워짐



https://towardsdatascience.com/training-yolo-select-anchor-boxes-like-this-3226cb8d7f0b

Faster R-CNN

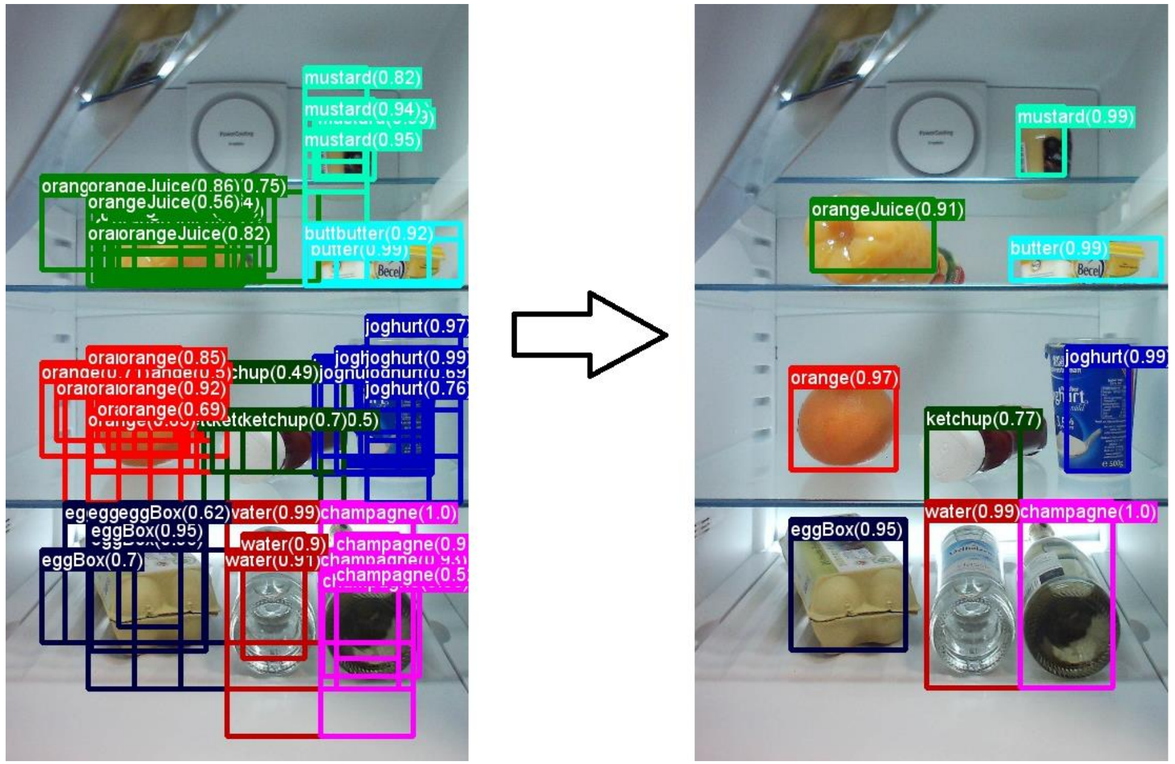
- RPN(Region Proposal Network)
  - Anchor box가 9개일 경우
    - Feature map의 가로*세로 1 칸마다 Anchor box 9개 생성
    - (2048, 14, 14) → (512, 14, 14) → (9, 14, 14)
    - 급격한 차원 축소를 방지하기 위하여 Conv2D(512, 3) 사용
  - 각 격자마다 Confidence score와 Bbox 좌표 예측
    - Confidence(Objectiveness score): (2 * 9, 14, 14)
    - Bbox coordinates: (4 * 9, 14, 14)
- RPN에서 생성 된 여러 Bbox는 NMS를 통해 필터링



https://velog.io/@iissaacc/Region-Proposal-Network

Faster R-CNN

- NMS(Non-Maximum Suppression)
  - 겹치는 Bbox를 제거하여 최종적으로 하나의 객체에 하나의 Bbox만 남기기 위한 기법
  - 여러 Bbox가 동일한 객체를 감지할 때, 중복되는 박스를 제거하고 가장 정확한 박스를 선택하는 데 사용
- 방법
  - 여러 개의 Bbox를 Confidence score 순으로 정렬
    - 너무 값이 낮은 박스들은 제거
  - 가장 값이 큰 기준 Bbox 선택
  - 기준 박스와 다른 박스의 겹치는 면적을 계산
    - 겹치는 면적이 Threshold(0.5)를 넘지 못한다면 제거



Fast R-CNN to Faster R-CNN

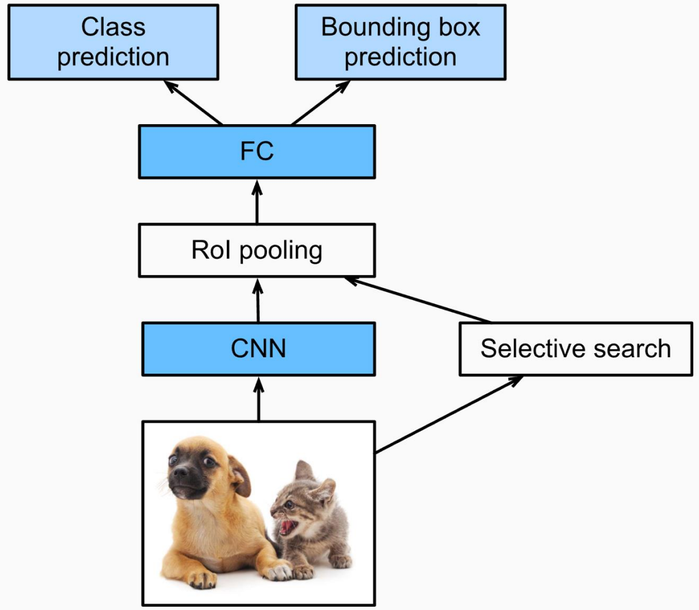


Fig. 14.8.2 The fast R-CNN model.

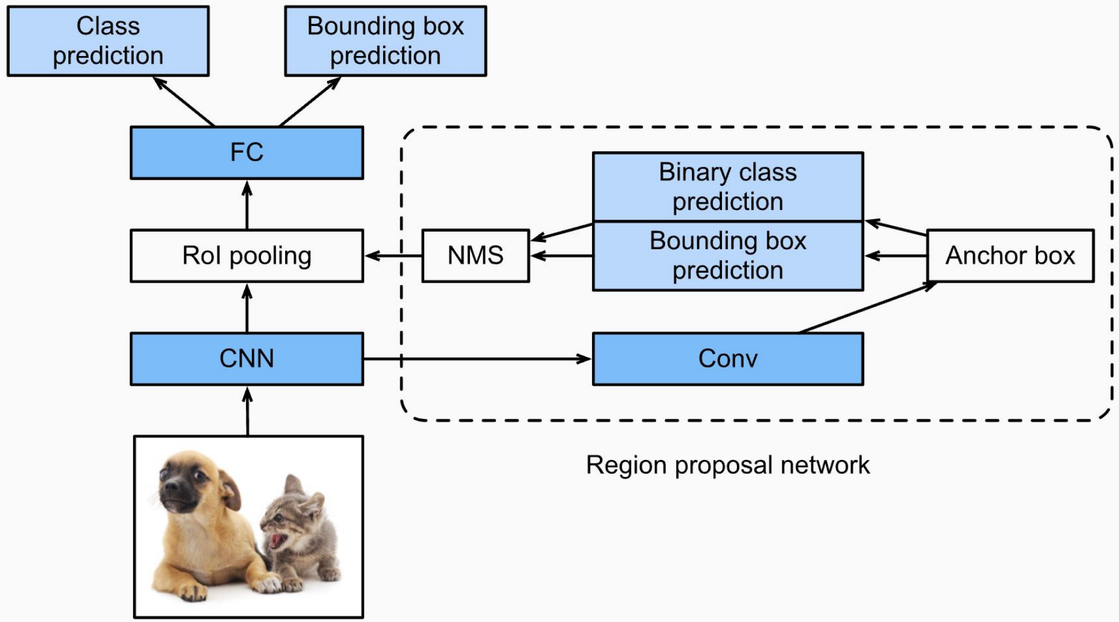


Fig. 14.8.4 The faster R-CNN model.

- RPN 도입을 통한 RoI 탐색 효율화
  - 신경망을 통한 RoI 탐색
  - 제한된 수의 Bbox 생성
  - Thresholding으로 Bbox 필터링하여 연산 속도 개선