R-CNN 계열 알고리즘

유튜브: https://youtu.be/0d_F-lyKH7Q

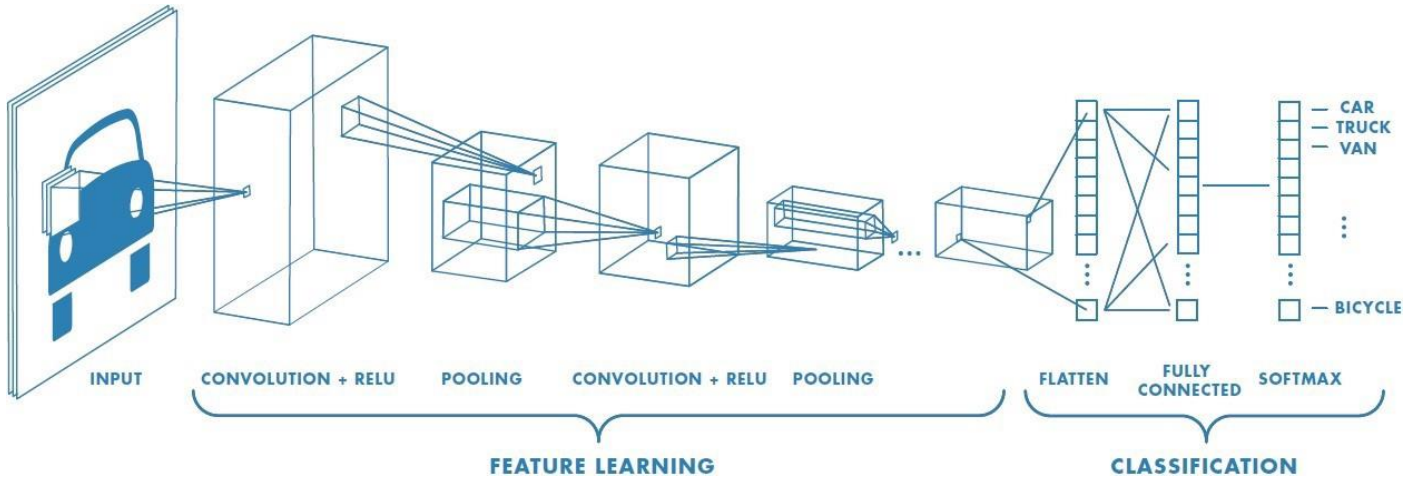
CNN

R-CNN

Fast R-CNN

Faster R-CNN

CNN



- 기존 모델의 문제점
  affine 계층 사용 시 데이터 형상 무시

  = 3차원 데이터를 1차원 데이터로 나열하여 형상 무시, 정보 손실

- CNN(Convolutional neural network): 합성곱 신경망

  인간의 시신경을 모방

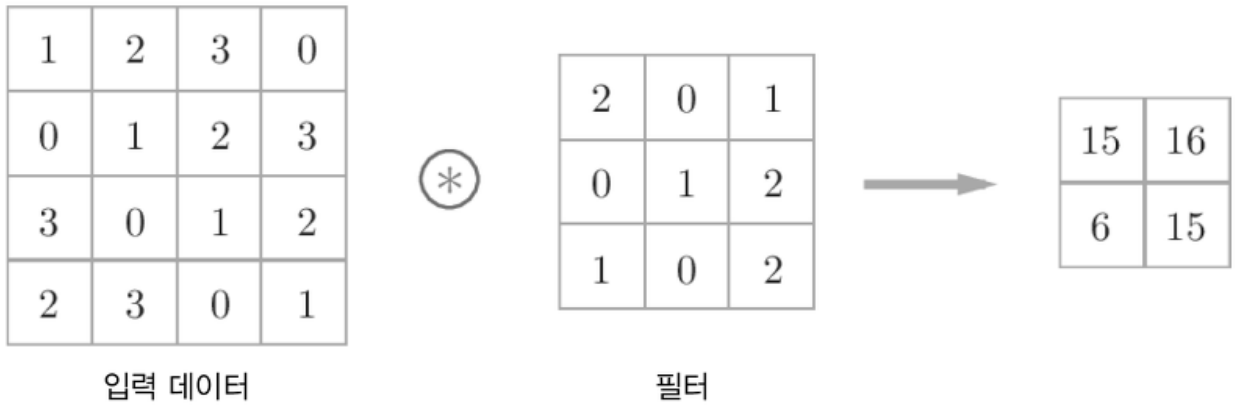
  얼굴인식, 이미지 인식 등 객체 인식과 같은 컴퓨터 비전에서 주로 사용

  합성곱 + 풀링 계층

CNN

- 합성곱 계층(convolution layer)

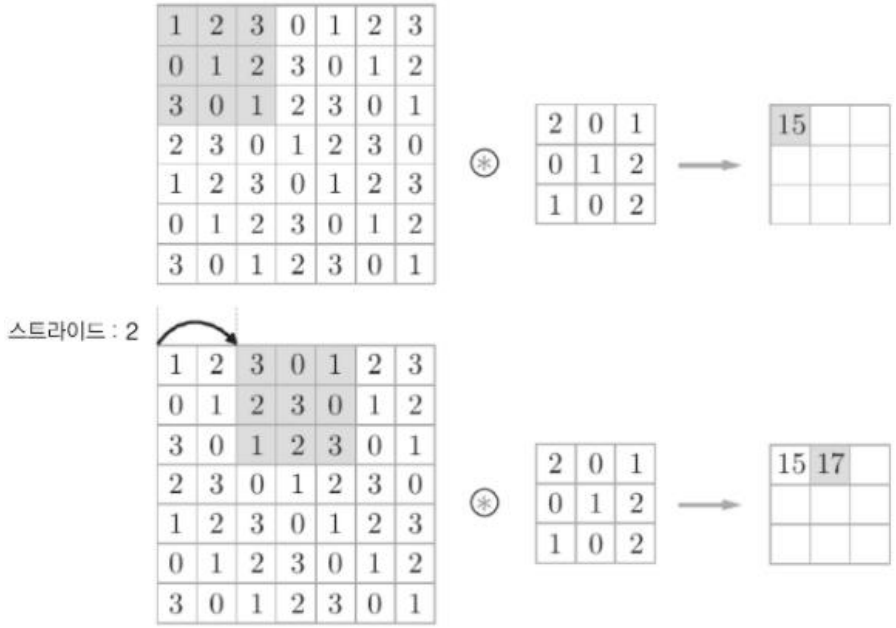
  합성곱 연산 = 필터 연산(필터=커널)



필터의 윈도우를 일정 간격으로 이동해가며 입력 데이터에 적용

대응하는 원소끼리 곱한 후 합함.(단일-곱셈 누산)

스트라이드:필터를 적용하는 간격

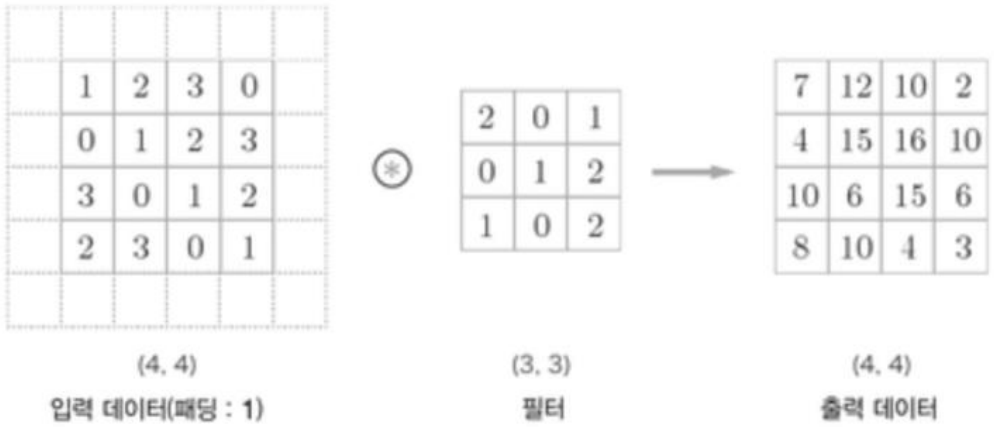


스트라이드를 크게하면 출력 작아짐.

CNN

패딩:합성곱 수행 전 입력 데이터 주변을
특정 값(0)으로 채움.

=> 주로 출력 크기를 조정할 목적으로 사용.



$$OH = \frac{H + 2P - FH}{S} + 1$$

$$OW = \frac{W + 2P - FW}{S} + 1$$

입력 크기:(H,W) , 필터(FH,FW), 출력(OH,OW),패딩:P, 스트라이드:S

3차원 데이터

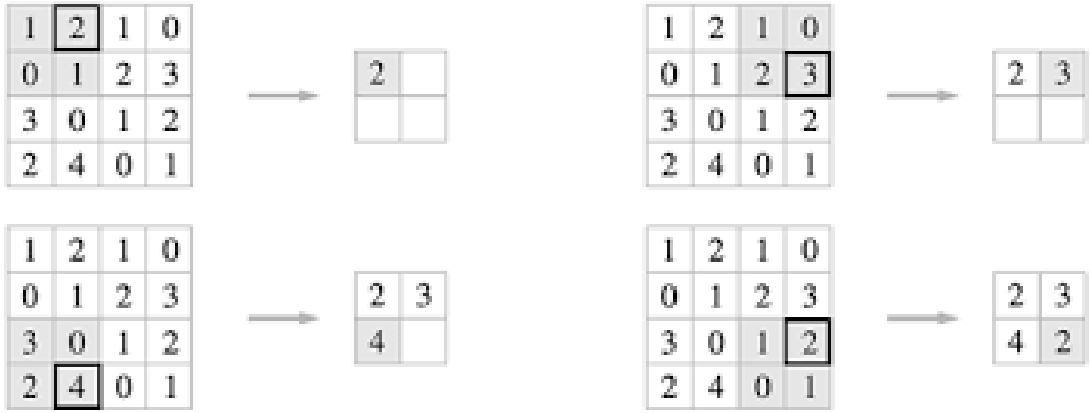
필터의 채널수와 입력데이터의 채널 수가 같아야함

CNN

- 풀링 계층

  : 세로 가로 방향의 공간을 줄이는 연산

  적당히 크기도 줄이고 특정 feature 강조



- Max Pooling(최대 풀링)

  : 최댓값을 구하는 연산

- Average Pooling

  : 대상 영역의 평균 계산

- Min Pooling

  : 최솟값을 구하는 연산

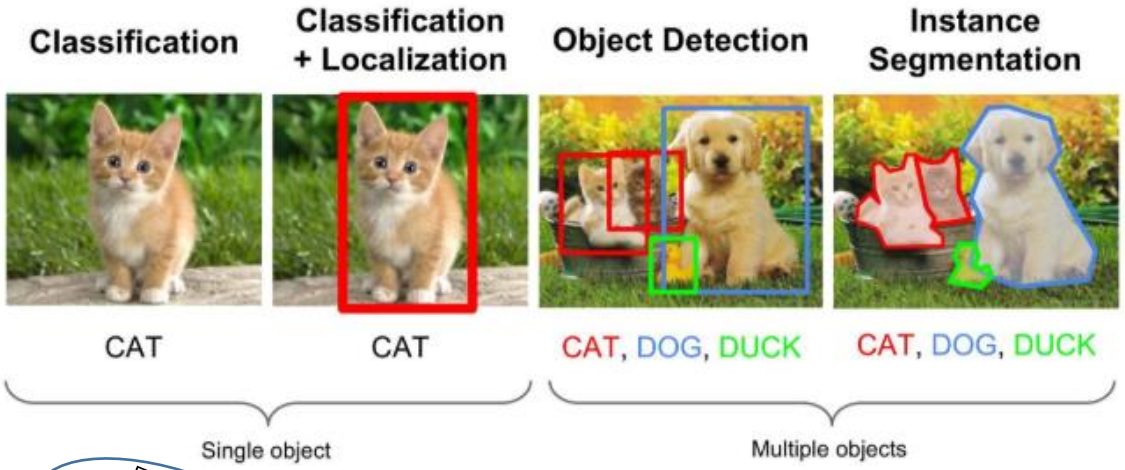
특징: 1. 학습해야 할 매개변수 x

2. 채널 수가 변하지 않음

3. 입력 변화에 영향을 적게 받음

풀링 윈도우 크기와 스트라이드는 같은 값 설정이 보통.

Object Detection



- **Classification :** Single object에 대해서 object의 클래스를 분류
- **Classification + Localization :** Single object에 대해서 object의 위치를 bounding box 로 찾고 **(Localization)** +클래스를 분류**(Classification)**
- **Object Detection :** Multiple objects에서 각각의 object에 대해 Classification + Localization을 수행
- **Instance Segmentation :** Object Detection과 유사하지만, object의 위치를 bounding box가 아닌 실제 edge로 찾음

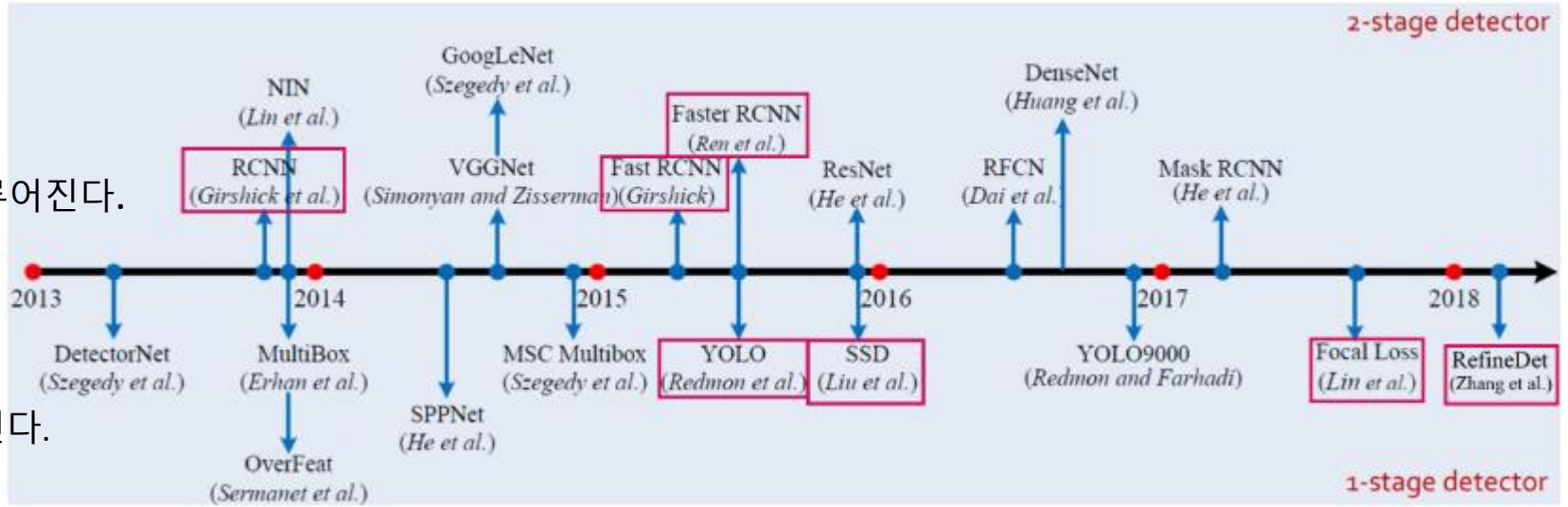
물체가 있을만 한 영역을 빠르게 찾아내는 알고리즘

<2-Stage Detector>

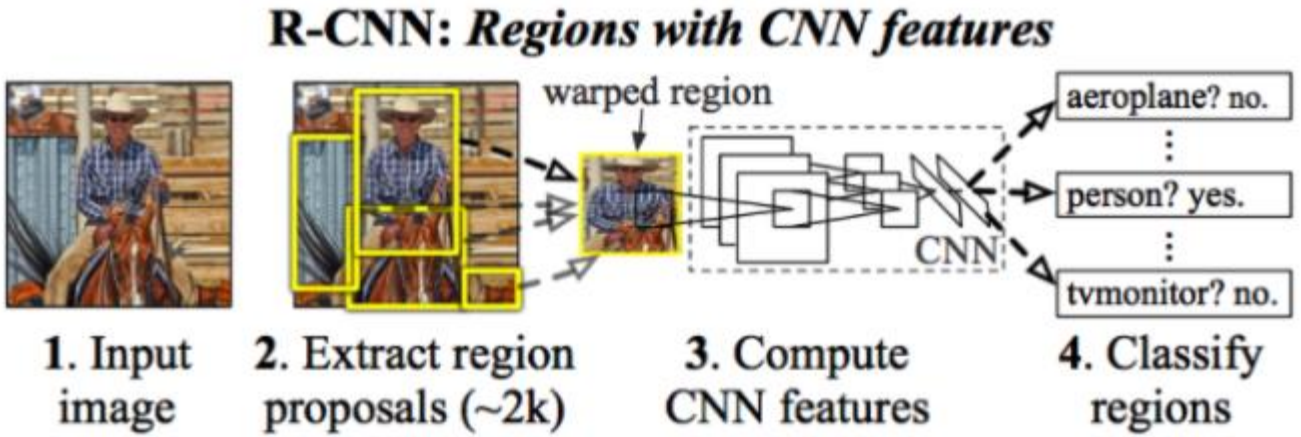
Regional Proposal과 Classification이 순차적으로 이루어진다.

<1-Stage Detector>

regional proposal와 classification이 동시에 이루어진다.



R-CNN



1. Image를 입력받는다.
2. Selective search알고리즘에 의해 regional proposal output 약 2000개를 추출한다.
추출한 regional proposal output을 모두 동일 input size로 만들어주기 위해 warp해준다.
3. 2000개의 warped image를 각각 CNN 모델에 넣는다.
4. 각각의 Convolution 결과에 대해 classification을 진행하여 결과를 얻는다.

•**1. Region Proposal :** "Object가 있을법한 영역"을 찾는 모듈 (기존의 Sliding window 방식의 비효율성 극복)

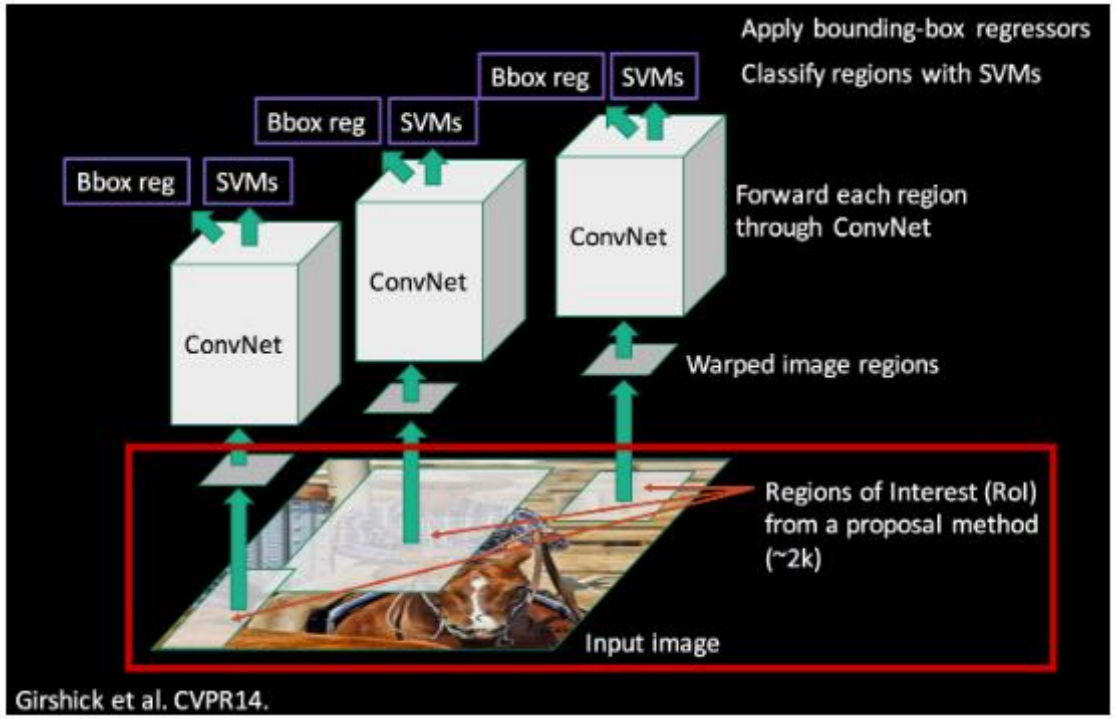
•**2. CNN** : 각각의 영역으로부터 고정된 크기의 Feature Vector를 뽑아낸다.(+warp)

•**3. SVM** : Classification을 위한 선형 지도학습 모델
(**Classifier로 Softmax를 쓰지 않고 SVM을 사용한 이유**: CNN fine-tuning을 위한 학습 데이터가 시기상 많지 않아서 Softmax를 적용시키면 오히려 성능이 낮아져서 SVM을 사용했다.

Warp 해주는 이유:
(Convolution Layer에는 input size가 고정이지 않음)

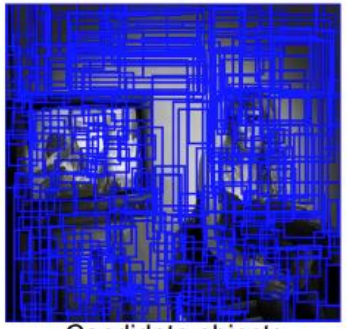
마지막 FC layer에서의 input size는 고정이므로 Convolution Layer에 대한 output size도 동일해야함.
따라서 Convolution Layer에 입력에서부터 동일한 input size로 넣어주어서 output size를 동일하게 하는 것이다.)

R-CNN




Input Image


Segmentation


Candidate objects

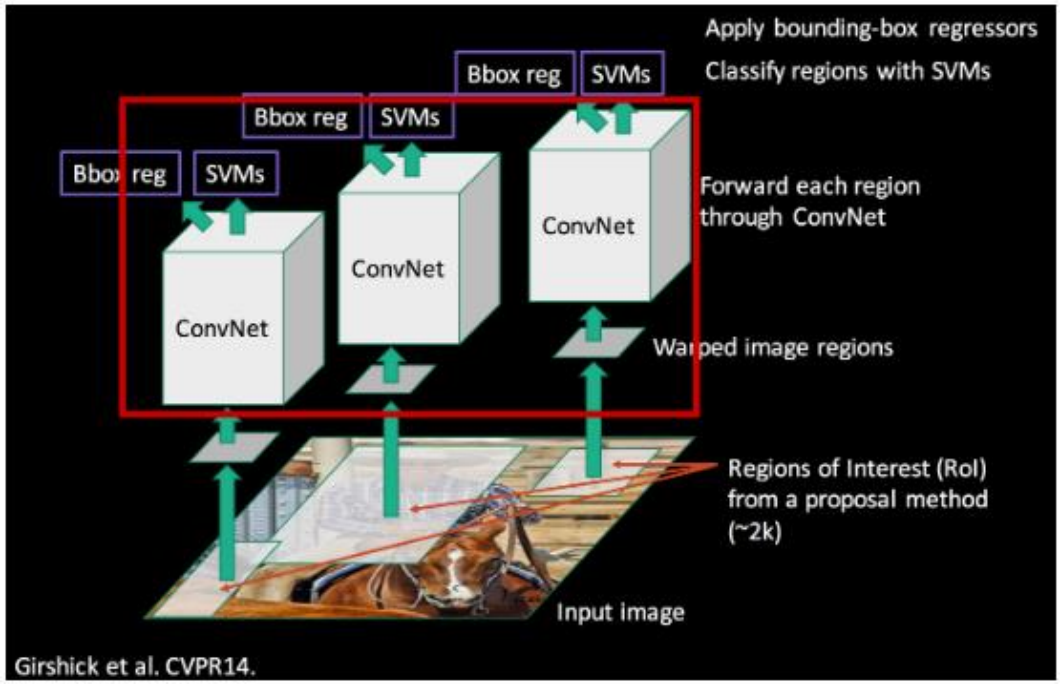
1.Region Proposal

Selective Search는 주변 픽셀 간의 유사도를 기준으로 Segmentation을 만들고, 이를 기준으로 물체가 있을 법한 박스를 추론

Selective search알고리즘에 의해 2000개의 region proposal이 생성되면 이들을 모두 CNN에 넣기 전에 같은 사이즈로 warp

기존 sliding window 방식:
이미지에서 물체를 찾기 위해 window의 (크기, 비율)을 임의로 마구 바꿔가면서 모든 영역에 대해서 탐색
=> 너무 느림

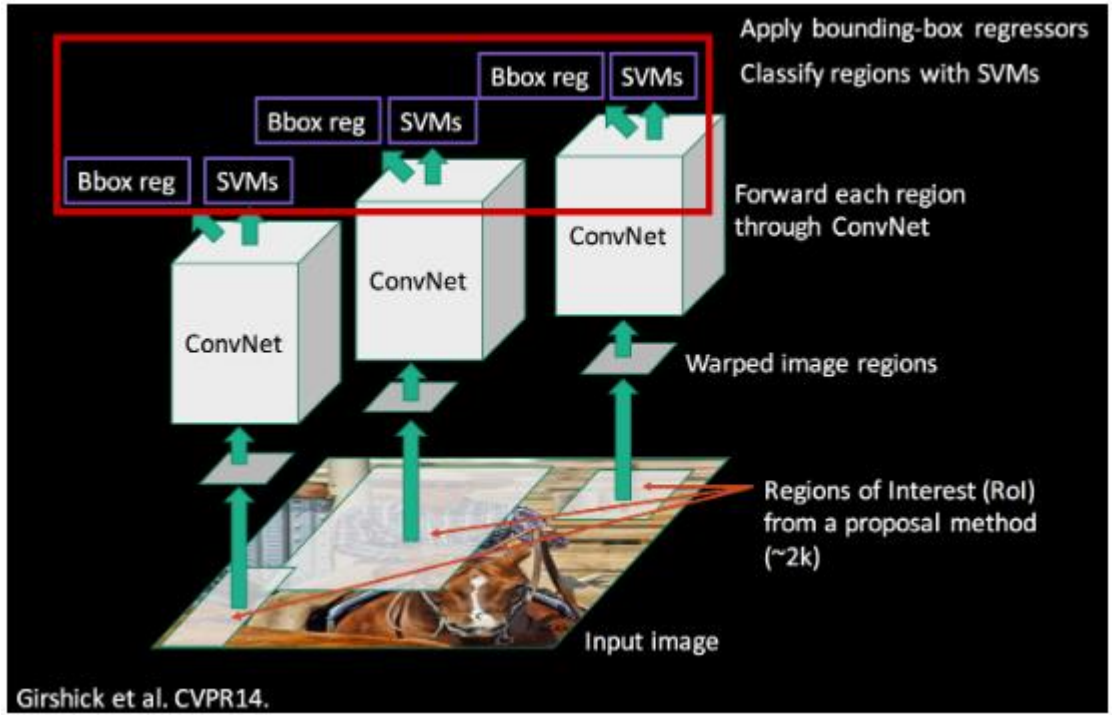
R-CNN



2. CNN

Warp작업을 통해 region proposal 모두 227x227 크기로 되면 CNN 모델에 넣음.
(Alexnet을 가져와 사용->fine tuning)
CNN을 거쳐 각각의 region proposal로부터 4096-dimentional feature vector를 뽑아내고,
고정길이의 Feature Vector를 만듬

R-CNN



3.SVM +bounding box

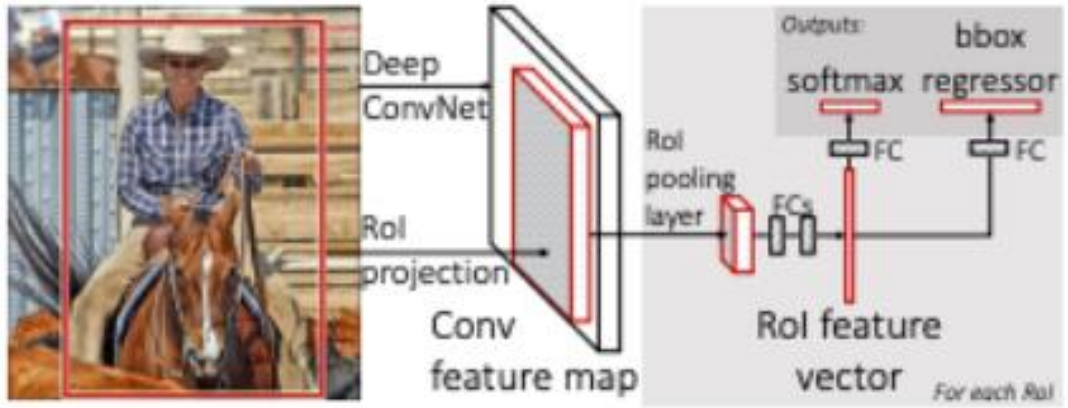
CNN모델로부터 feature가 추출되면 Linear SVM을 통해 classification을 진행
Selective search로 만든 bounding box는 정확하지 않기 때문에 물체를 정확히 감싸도록 조정해주는
bounding box regression(선형회귀 모델) 사용 (CNN을 거치기 전의 region proposal 데이터 사용)

R-CNN 단점!

1.느리다-
selective search로 2000개의 region proposal을 뽑고 각 영역마다 CNN을 수행하기 때문에
CNN연산 * 2000 만큼의 시간이 걸려 수행시간이 매우 느리다.

2. 복잡하다
CNN, SVM, Bounding Box Regression 총 세가지의 모델이 multi-stage pipelines으로 한 번에 학습되지 않는다.
end-to-end 로 학습할 수 없다.

Fast R-CNN



1-1. R-CNN에서와 마찬가지로 Selective Search를 통해 RoI를 찾음
1-2. 전체 이미지를 CNN에 통과시켜 feature map을 추출

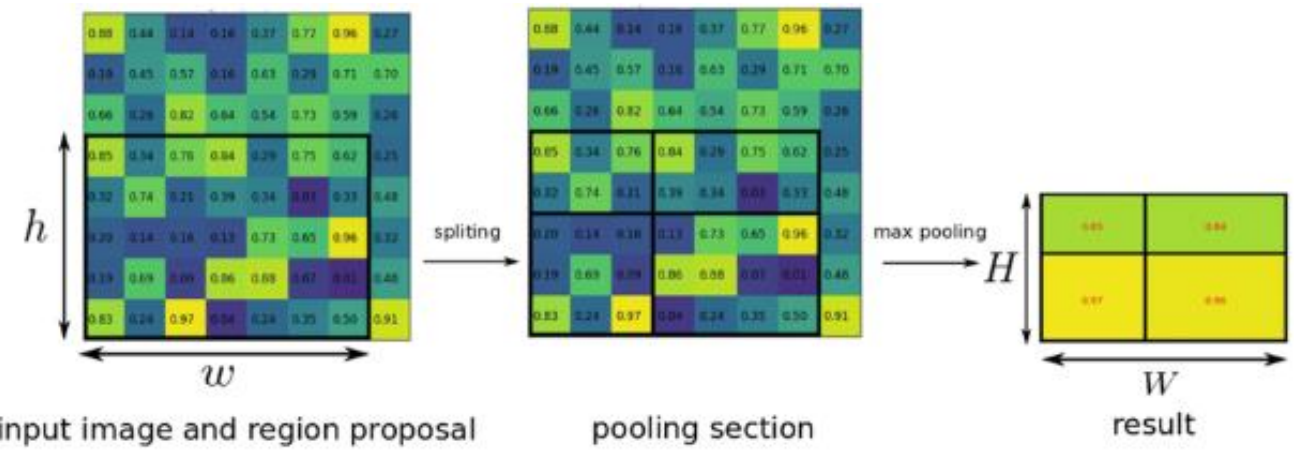
2. Selective Search를 통해서 찾은 각각의 RoI에 대해 RoI Pooling을 진행하여 고정된 크기의 feature vector를 얻음

3. feature vector는 FC layer를 통과한 뒤, 두 브랜치로 나뉨

4-1. 하나는 softmax를 통과하여 RoI에 대해 classification
4-2. bounding box regression을 통해 selective search로 찾은 box의 위치를 조정

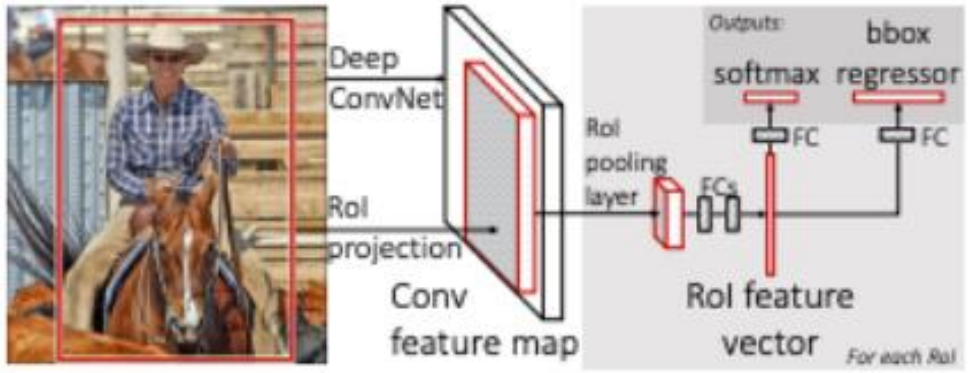
장점
1. CNN후에 region proposal 연산 - 2000xCNN연산 → 1번의 CNN연산
2. 변경된 feature vector가 결국 기존의 region proposal을 projection시킨 후 연산한 것이므로
해당 output으로 classification과 bbox regression도 학습 가능

Fast R-CNN



1.RoI Pooling(Region Of Interest)
크기가 다른 Feature Map의 Region마다 Stride를 다르게 Max Pooling을 진행하여 결과값을 맞추는 방법

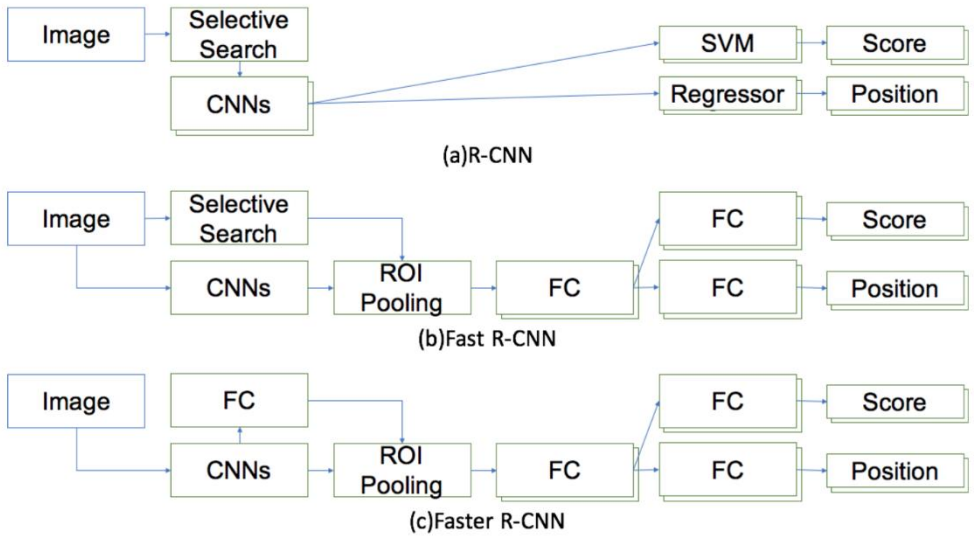
-먼저 입력 이미지를 CNN에 통과시켜 feature map을 추출
-그 후 이전에 미리 Selective search로 만들어놓은 RoI(=region proposal)을 feature map에 projection시킨다.
- 각각의 칸 별로 가장 큰 값을 추출하는 max pooling



2. Multi-task loss
feature vector를 **multi-task loss**를 사용하여 Classifier와 Bounding box regression을 동시에 학습. R-CNN 모델과 같이 **각 모델을 독립적으로 학습시켜야 하는 번거로움이 없다는 장점**

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

Faster R-CNN



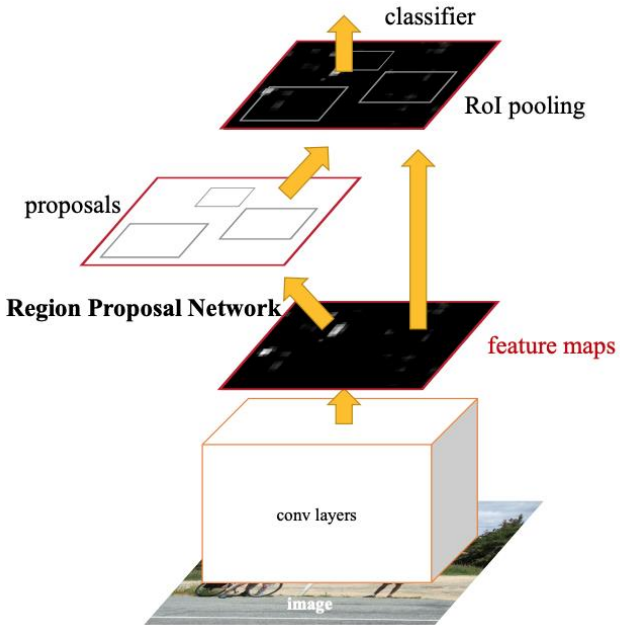
(1)region proposal 추출 → 각 region proposal별로 CNN 연산 → **(2)**classification, **(3)**bounding box regression

(1)region proposal 추출 → 전체 image CNN 연산 → RoI projection, RoI Pooling → **(2)**classification, bounding box regression

병목 현상

해결-> gpu에서 region proposal 계산

->conv layer에서 해결

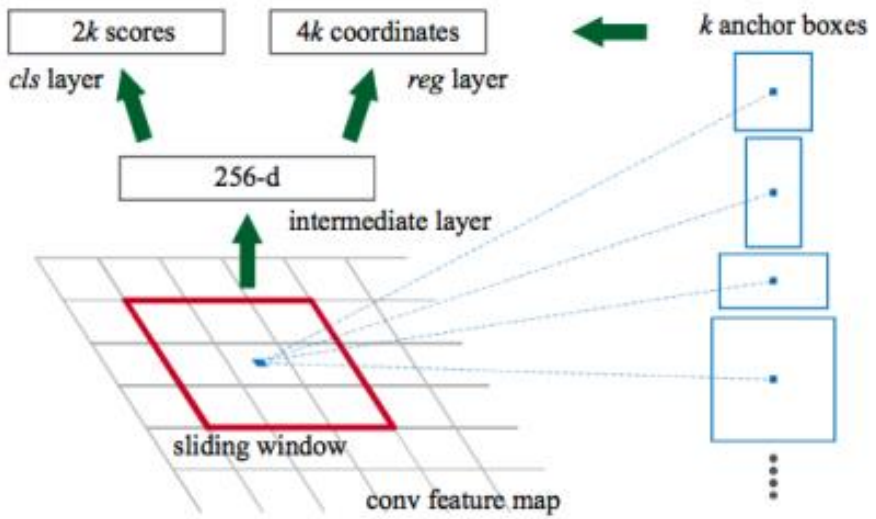


Faster R-CNN

RPN+ Fast R-CNN

: selective search 대신Region proposal Network 도입

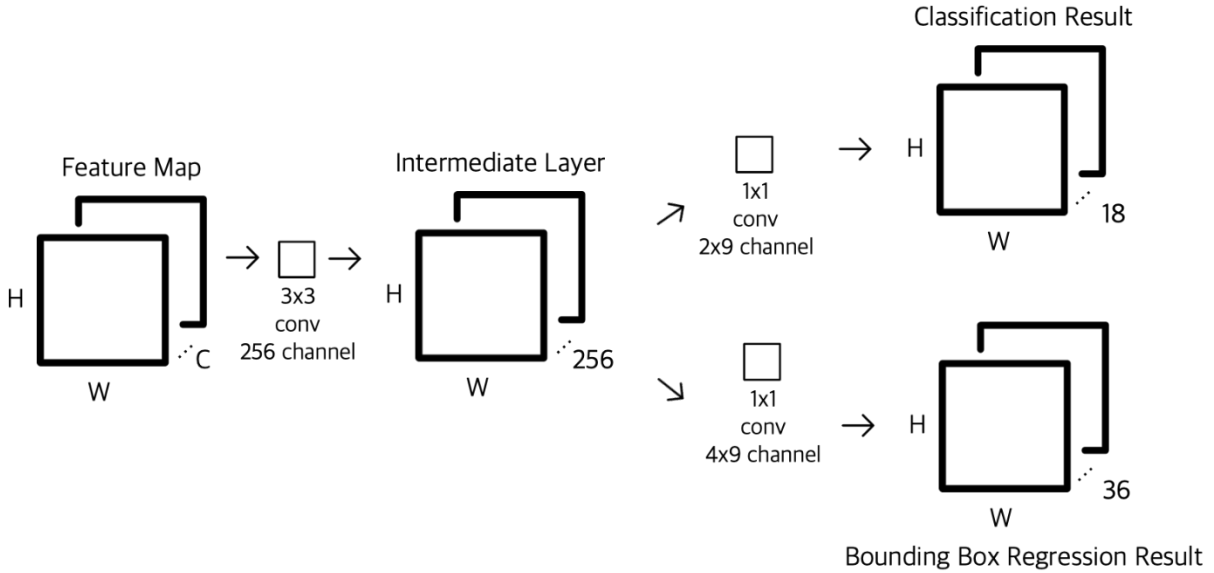
Faster R-CNN



RPN(Region Proposal Network)

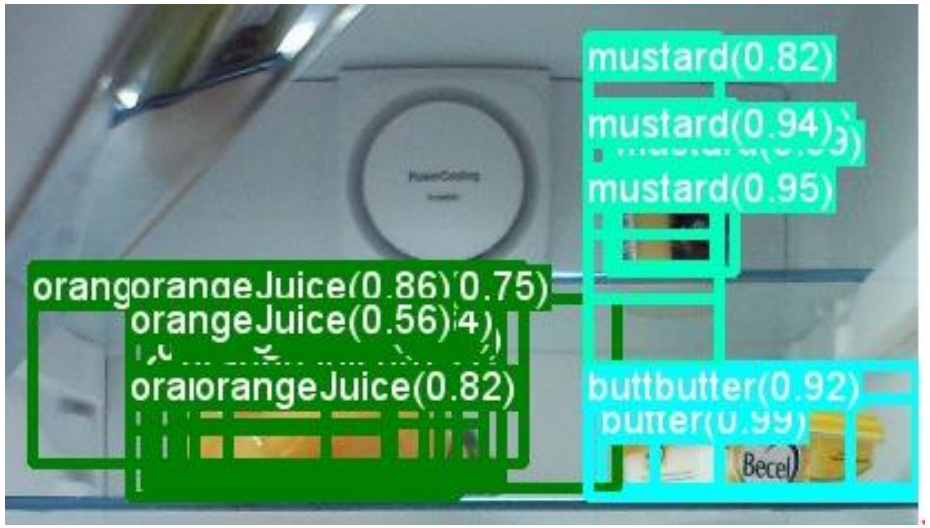
Region proposal을 생성하기 위해 feature map위에 nxn window를 sliding window
이때, k개의 anchor box를 미리 정의해놓는다.
여기서는 가로세로길이 3종류 x 비율 3종류 = 9개의 anchor box를 이용

1x1 convolution을 이용하여 classification과 bbox regression을 계산
네트워크를 가볍게 만들기 위해 binary classification으로 bbox에 물체가 있나 없나만 판단
무슨 물체인지 classification하는 것은 마지막 classification 단계



Faster R-CNN

NMS (Non-Maximum Suppression)



1. box들의 score(confidence)를 기준으로 정렬한다.
2. score가 가장 높은 box부터 시작해서 다른 모든 box들과 IoU를 계산해서 0.7이상이면 같은 객체를 detect한 box라고 생각하여 낮은 score box를 지움.
3. 최종적으로 각 object별로 score가 가장 높은 box 하나씩만 남음.

이 후 Fast R-CNN 실행

Q & A