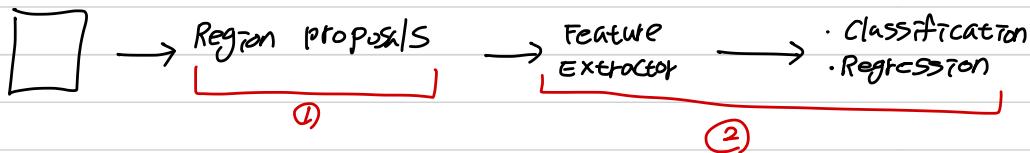


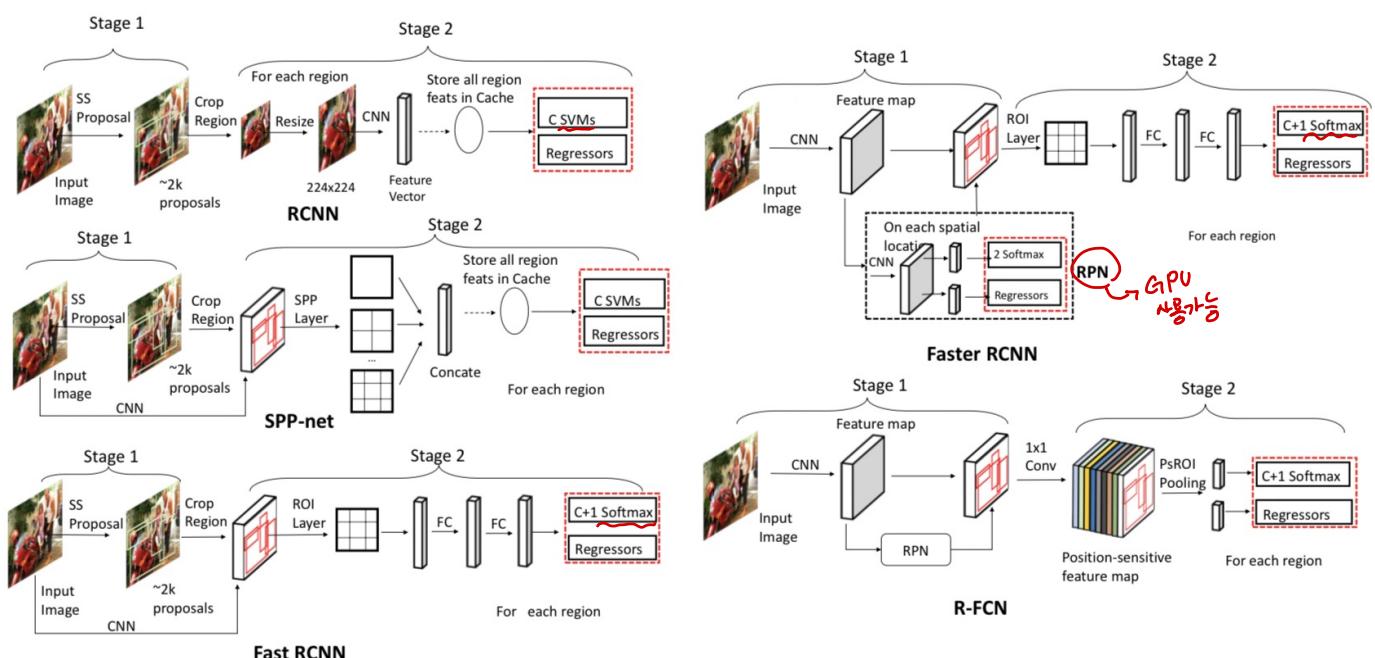
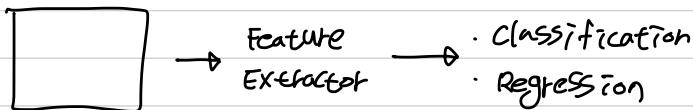
2-Stage Detector : R-CNN

• 물체의 ① 위치를 찾는 문제 (localization) 와 분류 (Classification) 문제를 순차적으로 해결



1-Stage Detector : YOLO

• 물체의 위치를 찾는 문제 (localization) 와 분류 (classification) 문제를 한 번에 해결



Object detection의 성능 평가지표

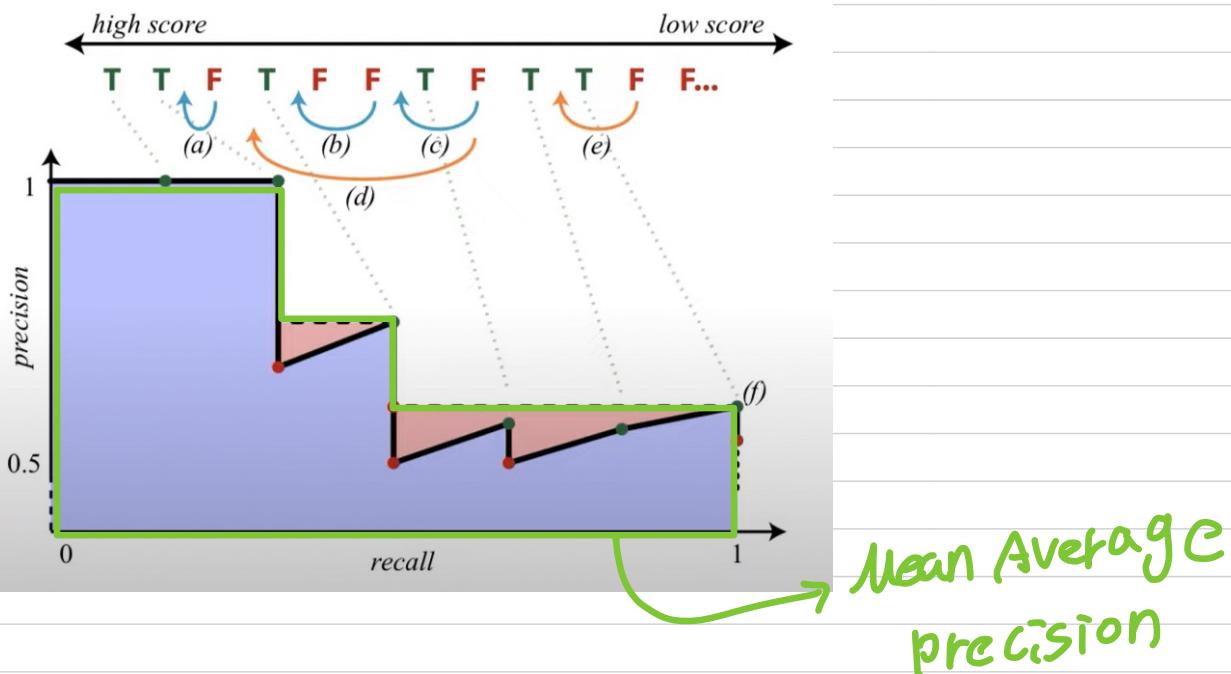
Accuracy → Precision: 정확도 $\rightarrow TP / (TP + FP) \rightarrow$ 올바르게 탐지 / 모델이 탐지한 물체의 수 \rightarrow 올바르거나 탐지한 비율
 Recall: 재현율 $\rightarrow TP / (TP + FN) \rightarrow$ 올바르게 탐지 / 실제 정답 물체의 수 \rightarrow 물체가 있을 때 얼마나 맞추었는가

	P	N
P	TP	FN
N	FP	TN

→ TP: 존재○ → 맞음
 TN: 존재 X → 맞음
 FP: 존재○ → 틀림
 FN: 존재 X → 틀림.

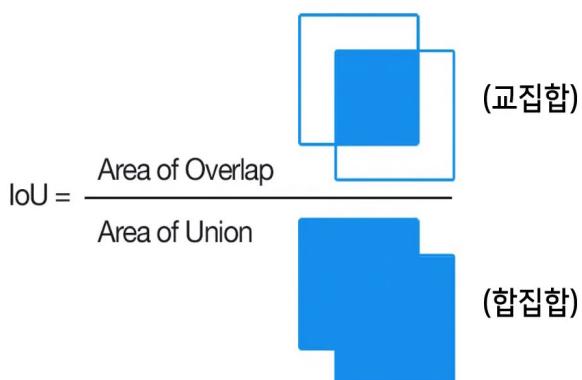
Average Precision

- 일반적으로 정확도와 재현율은 반비례
- Average Precision으로 위의 단점을 해결 (단조 감소 그래프로 넓이를 계산)



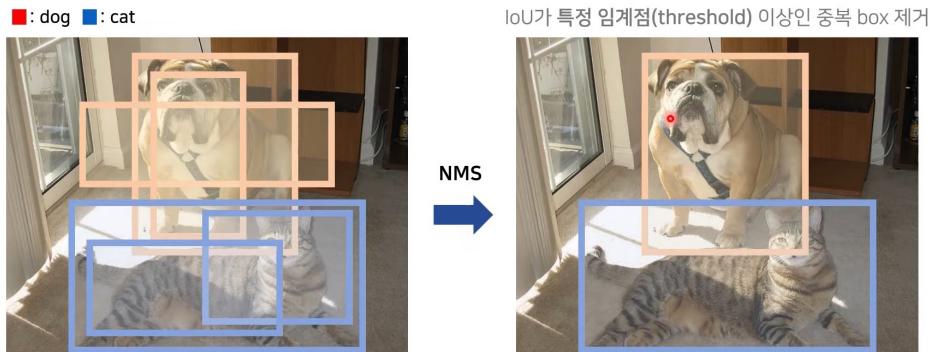
Intersection over Union (IoU)

- 두 bounding box가 겹치는 비율
 \rightarrow 성능 평가 예시: mAP@0.5는 정답과 예측의 IoU가 50% 이상일 때 정답으로 판정하였다.
- NMS 계산 예시: 같은 클래스끼리 IoU가 50% 이상일 때 낮은 confidence의 box를 삭제



NMS (Non Maximum Suppression)

- 각 객체 검출에서는 하나의 인스턴스(instance)에 하나의 bounding box가 적용되어야 한다.
- 따라서 여러개의 bounding box가 겹쳐 있는 경우에 하나로 합치는 방법이 필요.



Region Proposal 이란?

- 객체가 있을 믿을 만한 곳을 선정하는 기술 → Naive
- 가장 단순한 방식: window 기반의 단순 slide 방식 해 ROI로 추출
- Top-down: 이미지 전체에서 객체를 수집
- bottom-up: 픽셀 → 이미지 → Selective Search algorithm → R-CNN

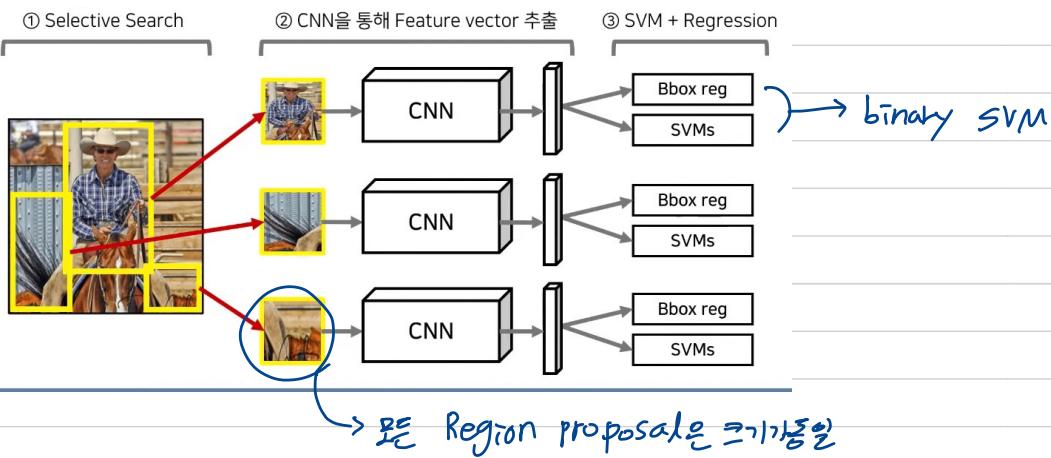


↳ Selective Search algorithm

1. Efficient Graph Based Image Segmentation을 통해 후보군 선정
 2. 초기 후보군들을 주변의 후보군들과 비교하여 인근하는 후보들끼리 유사도를 구함
 3. Greedy 알고리즘을 통해 유사도가 높은 것들끼리 합치기를 반복
- ② → 0~1 사이로 정규화된 4가지 요소 (color, texture, size, fill)들의 가중합

R-CNN

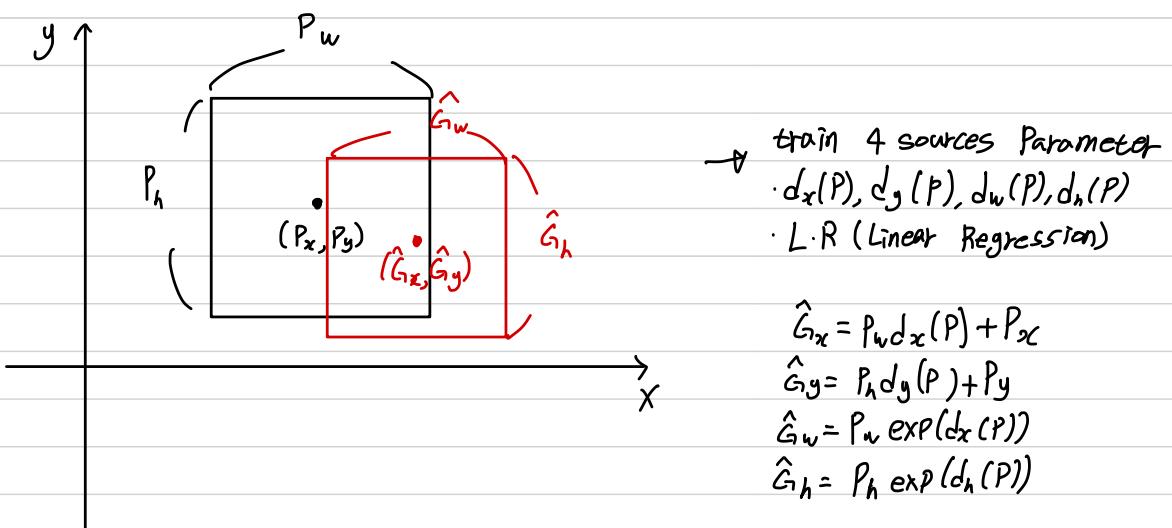
- Selective Search를 이용해 2000개의 Region Proposal을 생성
→ 이때 각 R.P를 일일이 CNN에 넣어 (forward) 결과를 계산



1. Selective Search를 통해 2000개의 ROI (Region of Interest) 추출
2. 각 ROI에 대해 warping을 수행하여 일정한 크기의 입력 이미지로 변경
3. Warped image를 CNN에 넣어 이미지 feature 추출
4. 해당 feature를 SVM에 넣어 클래스 분류 결과 획득
 \rightarrow 이때 각 클래스에 독립적으로 훈련된 이진 SVM을 사용
5. 해당 feature를 regressor에 넣어 위치 (bounding box) 예측

Bounding box regression

- 지역화 (localization) 성능 향상을 위해 bounding-box regression 사용
- train dataset : $\{(P^i, G^i)\}_{i=1,\dots,n}$
- real location: $P^i = (P_x^i, P_y^i, P_w^i, P_h^i)$
- Pred location: $G^i = (G_x^i, G_y^i, G_w^i, G_h^i)$
 \hookrightarrow Ground truth

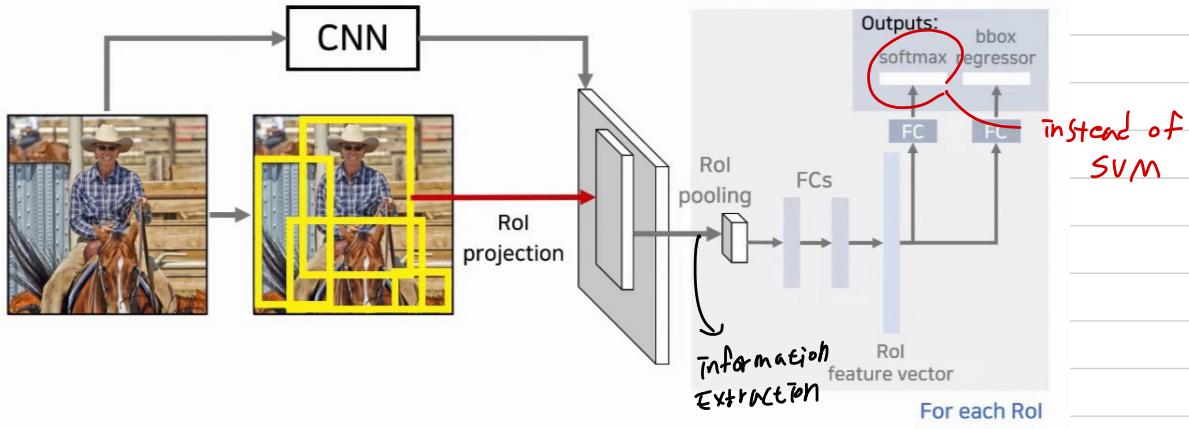


R-CNN limitation

- Input Images에 대해 CPU 기반의 Selective Search를 진행함으로 많은 시간 소요.
- 전체 이미지에서 SVM, Regressor 모듈이 CNN과 분리되어 있음.
 - CNN은 고정되므로 SVM과 Bounding Box Regression 결과로 CNN을 업데이트할 수 없음.
 - end-to-end 방식으로 학습 불가
- 모든 ROI를 CNN에 넣어야 하기에 2000번의 CNN 연산이 필요
 - 학습과 평가 과정에서 많은 시간 소요

Fast R-CNN

- 동일한 Region proposal을 이용해도 이미지를 한번만 CNN에 넣어 Feature Map을 생성



RoI Pooling Layer.

- 각 ROI 영역에 대해 max pooling → 고정된 벡터를 생성

Faster R-CNN

- 이전 fast R-CNN에서 병목현상을 해결
- Region Proposal → GPU에서 수행 (RPN 적용)
- end-to-end train 가능

RPN (Region Proposal Network)

- Selective Search 대체
- Input: feature map
- Output: RPN

∴ 1. 다양한 시그널 이미지를 입력받아 object score와 object proposal을 출력.

2. Fast R-CNN의 합성곱 신경망을 공유
3. Feature