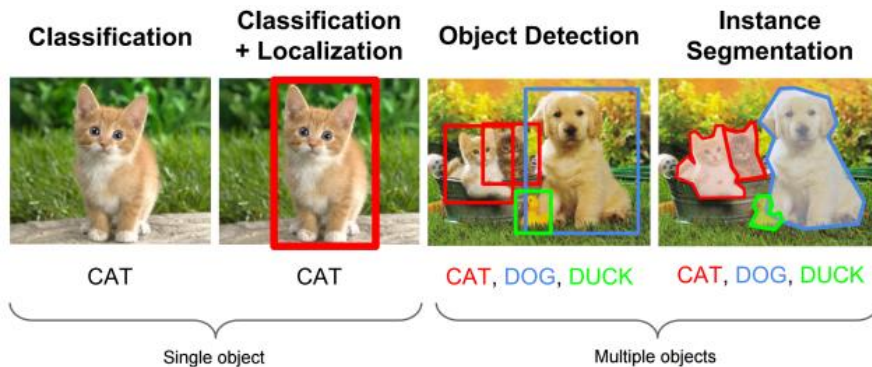


R-CNN

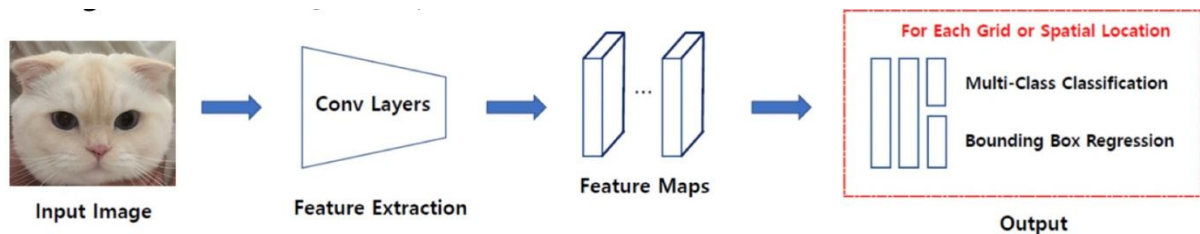
1. 배경 지식



Object detection이란?

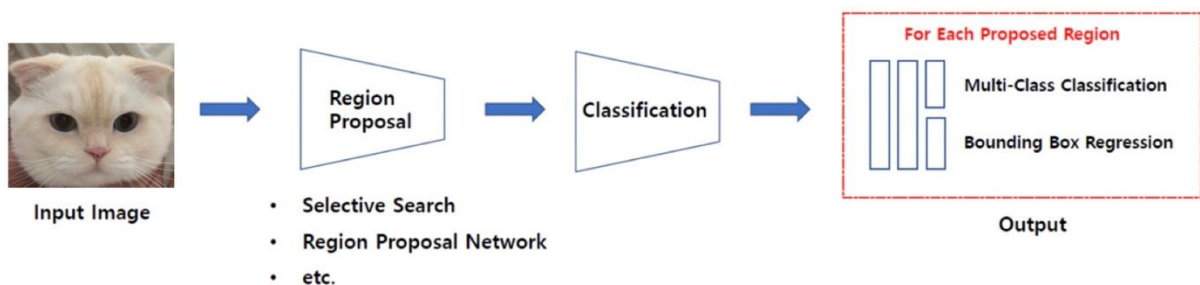
이미지에 존재하는 다양한 객체들에 대해서 Classification과 Bounding Box를 찾아 위치를 찾는 Localization을 모두 실행하는 것

Object detection Architecture



1-Stage Detector – Region Proposal과 Classification이 동시에 이루어짐

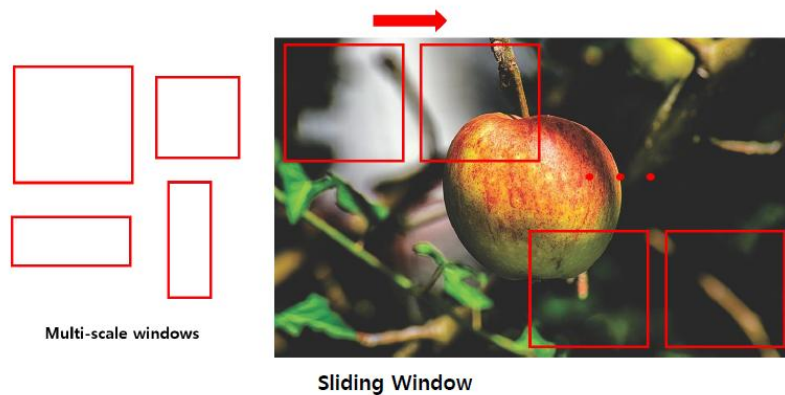
Ex) YOLO 계열(YOLO v1,v2,v3), SSD 계열(SSD, DSSD, DSOD, RetinaNet, RefineDet ...)



2-Stage Detector - Region Proposal과 Classification이 순차적으로 해결

Ex) R-CNN 계열(R-CNN, Fast R-CNN, Faster R-CNN, R-FCN, Mask R-CNN ...)

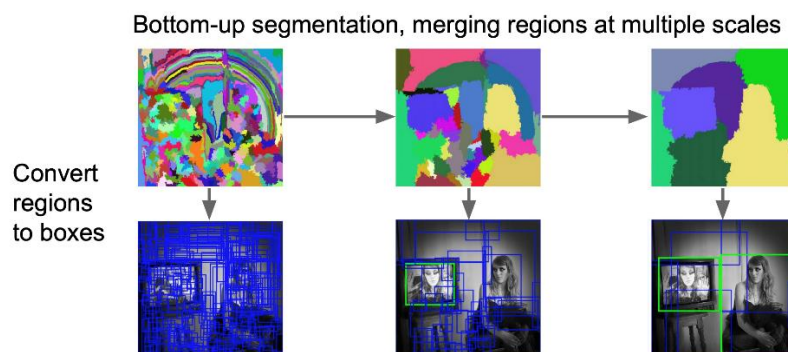
Sliding Window 다양한 scale의 window를 이미지의 왼쪽 위부터 오른쪽 아래까지 sliding하며 score를 계산하는 방법



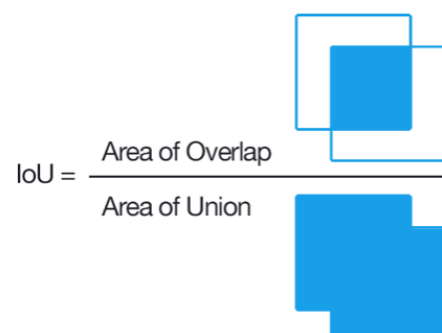
여러 번의 score를 계산해야 하므로 속도 측면에서 비효율적

➔ Deformable Part Model 등의 방안 제시

Selective Search 영역의 유사한 질감, 색, 강도 등을 갖는 인접 픽셀들을 찾아 물체가 있을 법한 box나 영역을 찾는 방법

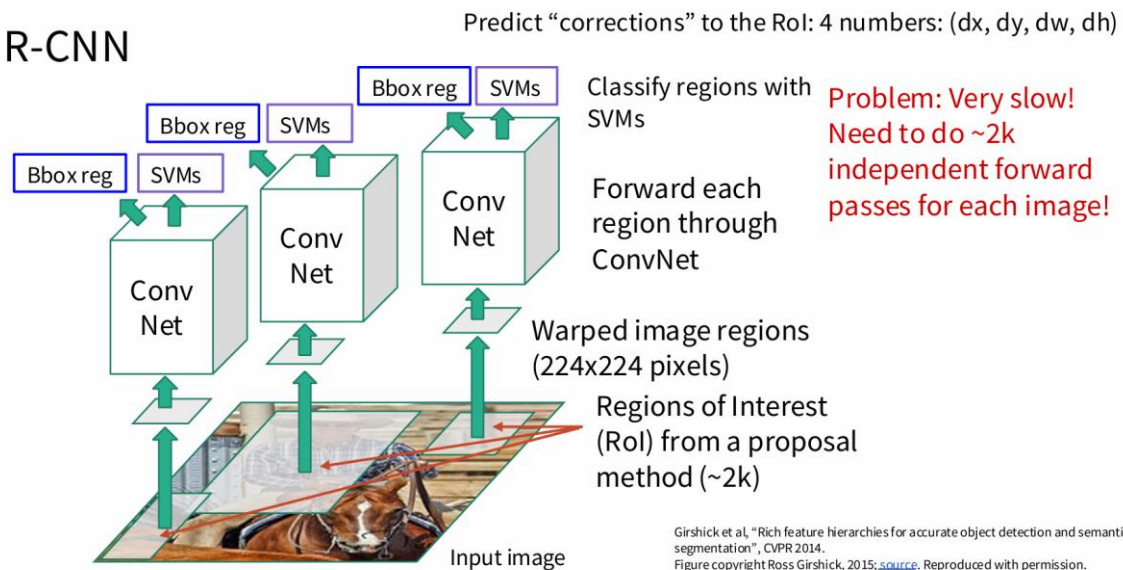


IoU(Intersection over Union) bounding box의 교집합 / 합집합



2. R-CNN에 대해

R-CNN



1. proposal method(Selective Search)

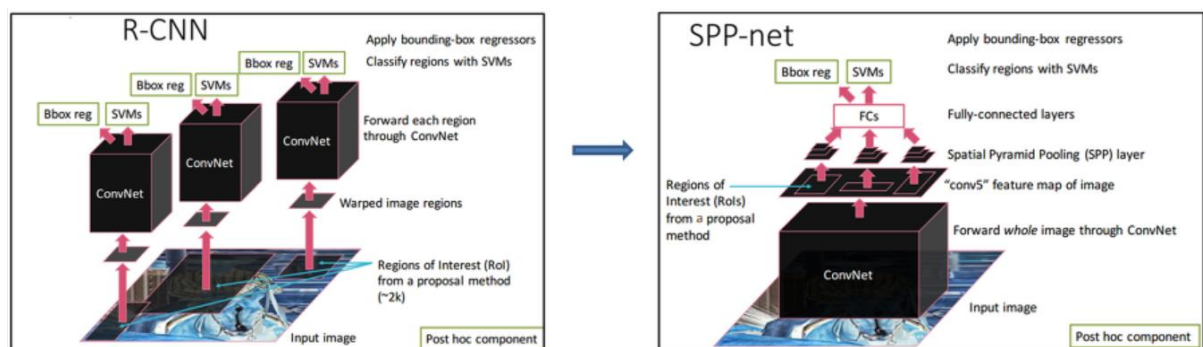
아래 식을 통해 의미 있는 2천개의 Bounding box 추출

$$S(r_i, r_j) = a_1 s_{\text{colour}}(r_i, r_j) + a_2 s_{\text{texture}}(r_i, r_j) + a_3 s_{\text{size}}(r_i, r_j) + a_4 s_{\text{fill}}(r_i, r_j)$$

유사도 : color, texture, size, fill의 선형결합

$$s_{\text{fill}}(r_i, r_j) = 1 - \frac{\text{size}(BB_{ij}) - \text{size}(r_i) - \text{size}(r_j)}{\text{size}(\text{image})}$$

2천개의 Bounding box에 대해 Conv 연산을 하기에 시간이 매우 오래 걸린다는 문제점



2천개의 region을 뽑지 않고 이미지를 먼저 Conv 연산을 하여 한 이미지에 대해서

한 번의 Conv 연산만으로 해결 가능

어 그러면 크기는 어떻게 맞출까? Spatial Pyramid Pooling을 통해서(추후 다룸)

2. Warping for Pre-trained Model

Bounding box의 비율 고려X -> 정보 왜곡 발생 가능

어떻게 해결하였을까? SPP-net에서 Spatial pyramid pooling 도입

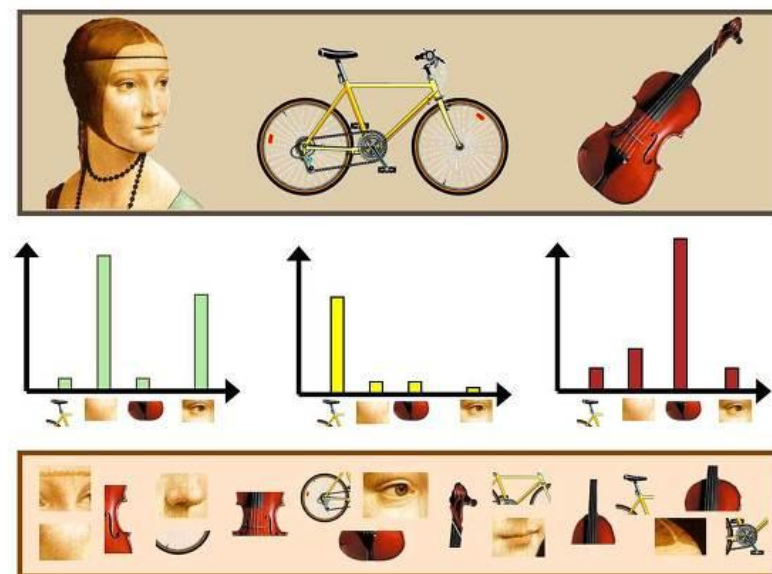
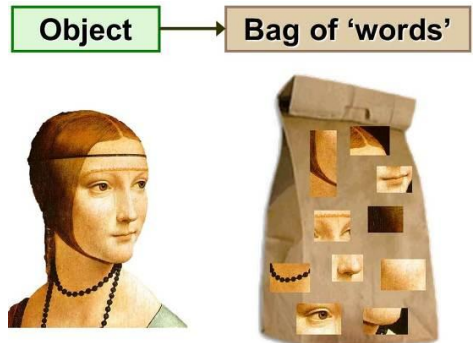
Spatial pyramid pooling에 대하여

2-1. 사전 지식

Object → **Bag of 'words'**

본래 Bag of Words 기법은 문서의 자동 분류를 위한 방법이었다. 예를 들어 어떤 문서에서 '환율', '주가', '금리' 등의 단어가 많이 나온다면 이 문서는 경제학에 관련된 문서로 분류한다.

이미지 분류에서는 어떻게 적용할까?



1. Feature Extraction : SIFT등의 방법으로 feature를 추출한다

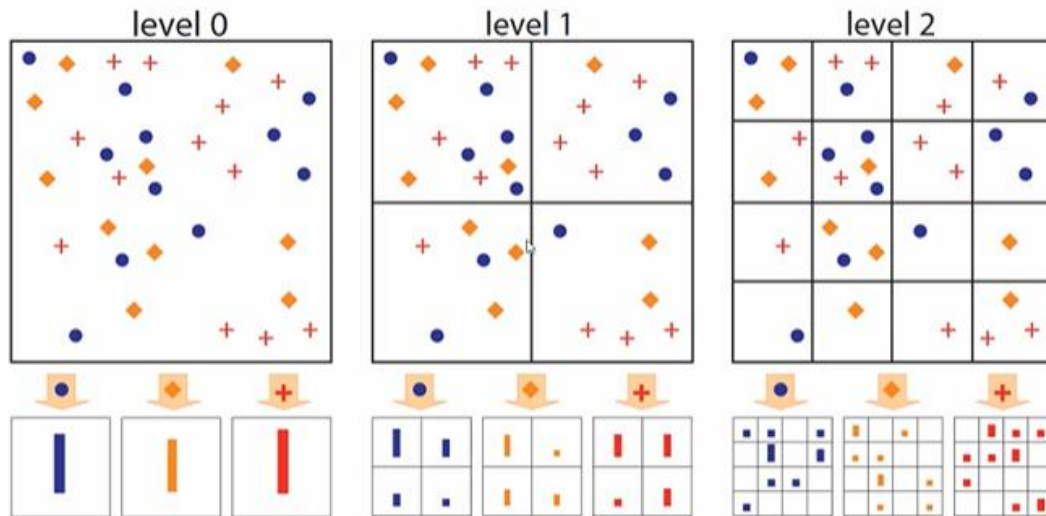
2. Clustering : 추출된 feature들을 k-means와 같은 방법으로 clustering을 하여 클러스터 중심들인 codeword들을 찾아낸다.

3. Codebook Generation : 찾아진 codeword들로 구성되는 codebook를 생성한다.

4. Image Representation : 각각의 이미지들을 codeword들의 histogram으로 표현한다.

이때 histogram bin의 개수는 codebook의 크기와 동일하다.

2-2. Spatial pyramid pooling



분면을 나눠 각 분면에 대해 codeword histogram을 진행한다

그러면 모든 이미지에 대해서 같은 크기의 vector가 나온다

위 이미지를 예로 들어 codeword가 3개인 이미지가 있으면

1분면일 때 3*1개의 vector, 4분면일 때 4*3개의 vector, 16분면일 때 16*3개의 vector
가 추출되므로 이미지의 size에 상관없이 63개의 vector로 표현 가능해진다.

3. Fine-tuning

ImageNet을 학습한 모델을 PASCAL VOC에 적용하기 위해서

이때 object가 없는 이미지를 고려해 class 수는 20(class)+1(background)

학습하는 데이터는 Positive Sample : Negative Sample = 1:3

*Positive Sample : Selective Search로 만들어낸 region과 할당된 실제 Box의 IoU값이 0.5
이상인 것(0.5미만인 값들은 Background class로 labeling)

4. Feature extract by conv

Conv연산을 통하여 feature들을 추출한다

5. SVM classification

왜 classifier로 softmax를 사용하지 않았을까?

Softmax가 성능이 더 좋지 않아서(경험적 원인)

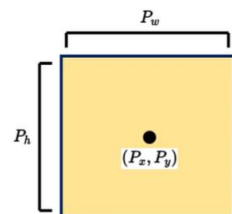
Fine-tuning된 CNN 모델로부터 Feature vector가 추출되면 이를 SVM으로 classifiy

이때, class 별로 SVM Classifier를 구성하여 학습(메모리적으로 효율적)

Positive Sample : Class별 object의 ground-truth box

Negative Sample : IoU값이 0.3미만인 영역

6. Bounding Box Regression



- 학습 데이터셋: $\{(P^i, G^i)\}_{i=1, \dots, N}$
- 예측 위치: $P^i = (P_x^i, P_y^i, P_w^i, P_h^i)$
- 실제 위치: $G^i = (G_x^i, G_y^i, G_w^i, G_h^i)$

x, y, w, h는 각각 Bounding Box의 x좌표, y좌표, Width(너비), Height(높이)

예측된 P를 G에 근사하는 것이 Bounding Box Regression의 목표

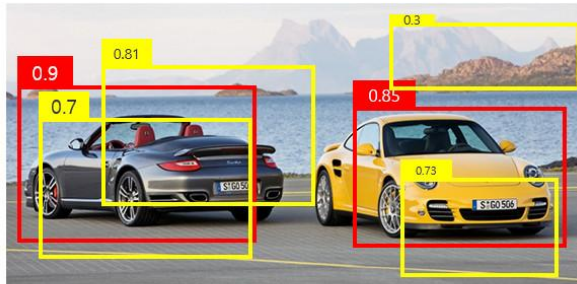
	학습해야 할 Parameter	G와 최대한 가까워질 변수	추정값과 실제값의 차이
	$d_x(P), d_y(P), d_w(P), d_h(P)$	$\hat{G}_x = P_w d_x(P) + P_x$ $\hat{G}_y = P_h d_y(P) + P_y$ $\hat{G}_w = P_w \exp(d_w(P))$ $\hat{G}_h = P_h \exp(d_h(P))$	$t_x = (G_x - P_x)/P_w$ $t_y = (G_y - P_y)/P_h$ $t_w = \log(G_w/P_w)$ $t_h = \log(G_h/P_h)$

$$\mathbf{w}_* = \underset{\hat{\mathbf{w}}_*}{\operatorname{argmin}} \sum_i^N \left(t_*^i - \hat{\mathbf{w}}_*^T \phi_5(P^i) \right)^2 + \lambda \|\hat{\mathbf{w}}_*\|^2 \quad d_*(P) = \mathbf{w}_*^T \phi_5(P)$$

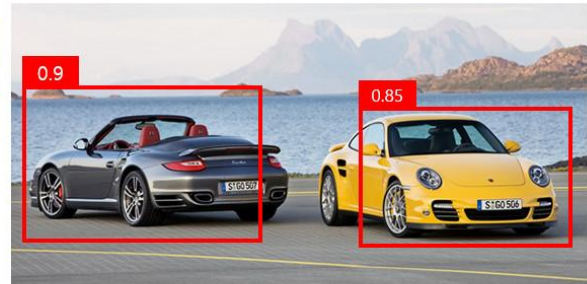
이에, 다음과 같이 t와 d의 차이인 Loss를 줄여 나가는 방향으로 학습한다.

7. Non Maximum Suppression(NMS)

왜 사용할까? 하나의 객체에 대해서 많은 bounding box들이 겹칠 수 있기 때문에
객체 탐지의 정확도가 떨어질 수 있다



Before Non Maximum Suppression



After Non Maximum Suppression

1. confidence score threshold

Confidence score threshold = 0.5 라면 위 그림에서 0.5이하인 0.3과 같은 것들은 제거됨

2. IoU threshold

IoU threshold = 0.4 라면 confidence score에 따라 내림차순으로 box들을 정렬

가장 높은 confidence score를 가지는 box와 IoU 값 연산

IoU threshold값인 0.4 이하인 것들은 제거 후 위 과정 반복