**이미지 위치 찾기**. (기존 CNN활용은 이미지 분류하기였음)

크게 localization, detection, segmentation의 방법이 있음.

* Localization : 이미지에 single object 존재. 여기에 bounding box 그리기
* Detection : 이미지에 multiple objects 존재. Objects들 찾아 bounding box 그리기
* Segmentation : 이미지에 multiple objects 존재. Objects들 pixel단위로 나누어 체크

Classification의 dataset은 feature, label 필요.

Localization의 dataset은 feature, label, box의 (x,y,w,h)값들 필요

Detection의 dataset은 feature, [label, box의 (x,y,w,h)] \* n 값들 필요

Segmentation의 dataset은 object가 존재하는 pixel정보값 data 필요.

1. Localization 알고리즘

Convolutional layer와 작동방식 거의 비슷함. Sliding window를 좌상단부터 우하단까지 스캔하며 object가 있을 확률 저장해 object 위치 찾기.

1. Detection 알고리즘
2. One-Stage Method : ex) YOLO, SSD
3. Two-Stage Method : ex) R-CNN, Fast R-CNN, Faster R-CNN

Image object detection 동향 : R-CNN, Fast R-CNN, Faster R-CNN, Mask R-CNN

간단한 원리 정도로

R-CNN

논문 링크 : <https://arxiv.org/pdf/1311.2524.pdf>

* Region proposal 알고리즘 : 이미지에서 selective search 방법으로 비슷한 색의 인접 픽셀을 연결해 bounding box들 구성. 이 부분을 CNN layer에 넣어 feature vector추출
* Feature vector를 분류기(SVM linear classifier)에 통과시켜 객체가 맞는지, 어떤 객체인지 분류

Fast R-CNN

* Feature Extractor : 기존 R-CNN은 selective search로 뽑은 RoI(Region of Interest) 모두를 CNN에 넣었으나 fast R-CNN에서는 이미지의 좌표와 크기정보만 저장. 그리고 CNN에 이미지 한 개만 넣어 공통 feature map을 추출하고 각 RoI들의 좌표만 비율에 따라서 변경(RoI projection)
* Roi pooling layer : 공통 feature map과 roi 좌표 정보를 roi pooling layer에 통과시킴. 여기서 roi pooling layer는 다양한 크기의 roi를 고정 크기로 바꿔줌.(Fully-connected layer에 연결하기 위하여)
* Classifier & Regressor : FCL에서 classification을 위한 classifier, detection을 위한 regressor(bounding box맞춤)로 병렬로 연결됨. 최종 출력은 각 region에 있는 class와 bounding box들

Faster R-CNN

* image들을 CNN에 넣음
* 나온 feature map을 Region Proposal Network(RPN)에 넣어 학습
* Rol pooling으로 FCL에 넣을 수 있게 resize
* Fast R-CNN과 동일하게 진행

Mask R-CNN

* 해당 픽셀이 객체인지 masking하는 CNN 추가(Binary Mask)
* RoI pooling 대신에 RoI Align을 사용(RoI Pool을 더 정확하게 다시 정렬. 픽셀수준 정확도 필요하므로)