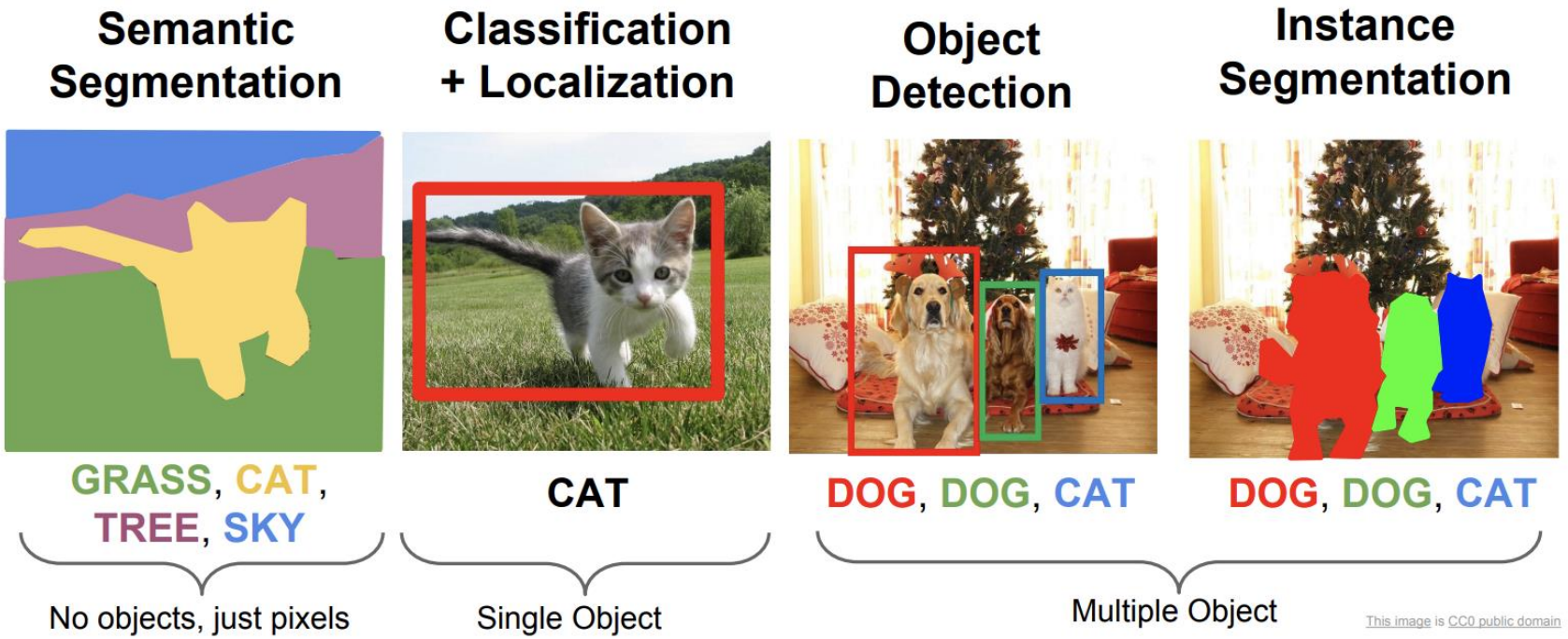
Detection and Segmentation

20192241 이민옥

OverView

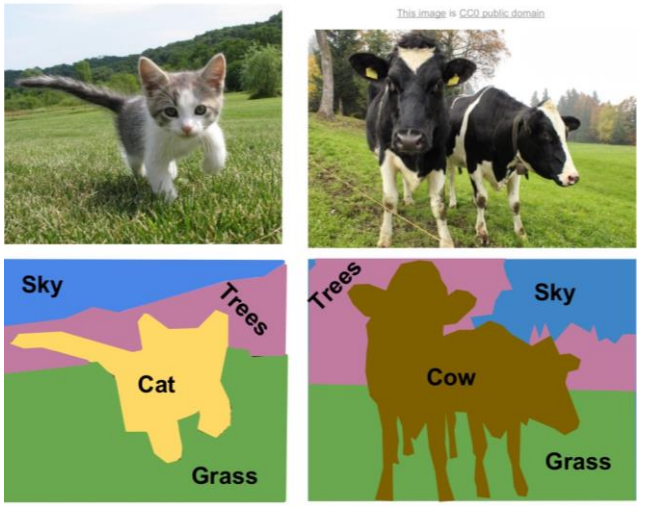
Computer Vision Tasks



02

Computer Vision Tasks

Semantic Segmentation

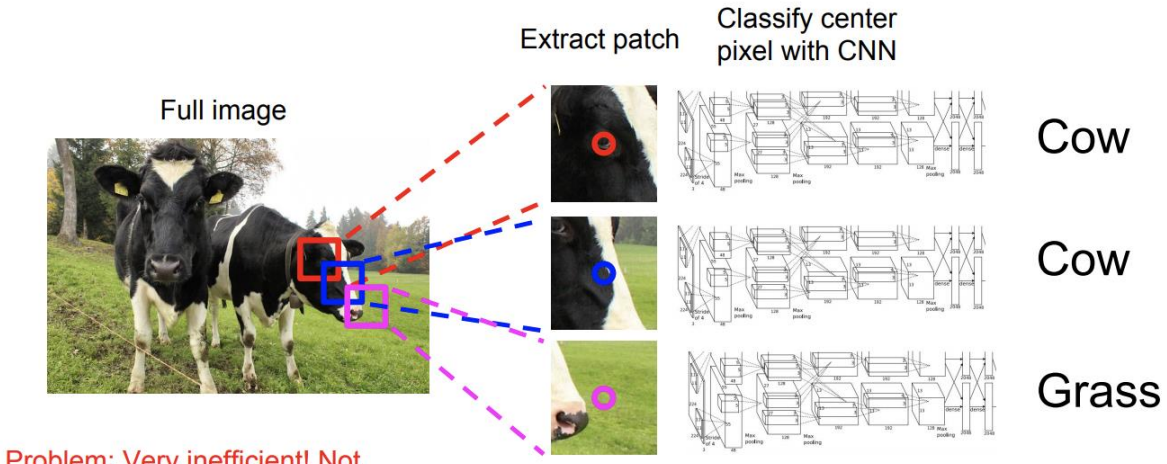


- No objects, just pixels
- Input : Image
- Output : decision of a category for every pixel
  : 픽셀별로 어떤 카테고리에 속하는지 알려줌
- 픽셀이 어떤 것을 나타내는지 알려주지만, 개별에 대해선 분류할 수 없음
  * 2개 이상의 물체를 같은 것으로 인식
  → 추후 instance segmentation에서 이 문제를 해결함
- Semantic Segmentation은 classification을 통해 진행될 수 있음

Computer Vision Tasks

Semantic Segmentation

1. Sliding Window Approach



Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013
Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

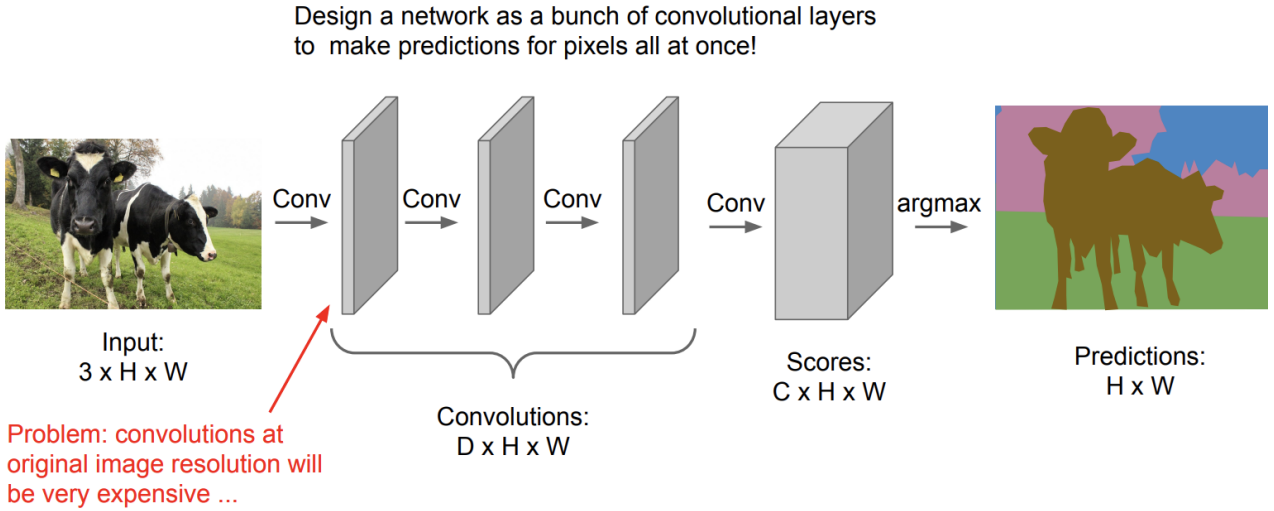
- 잘려진 patch마다 어떤 class인지 유추
- 이 방법은 computation이 비싼 방법
- 중복되는 patch 사이에서 공유된 feature를 재사용하지 않음

02

Computer Vision Tasks

Semantic Segmentation

2. Fully Convolutional



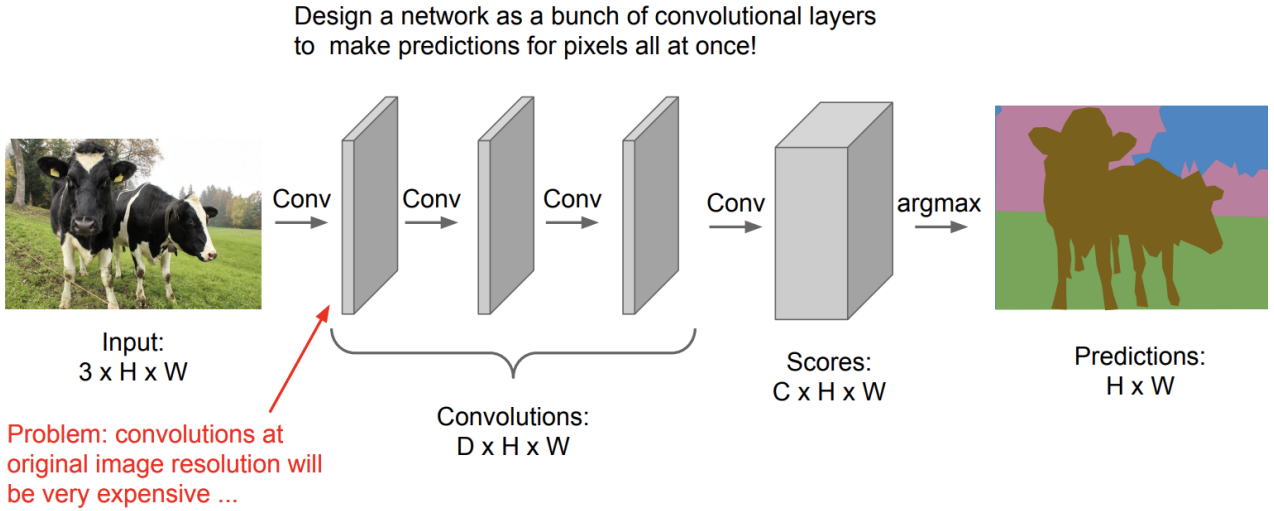
- 3x3 filter를 사용해 이미지 크기를 유지하며 convolution에 넣음
- 한번에 모든 픽셀을 예측할 수 있도록 설계
- Output : C *H * W의 Tensor
  → output Tensor? = 입력 이미지의 모든 픽셀 값에 대해 Classification Scores를 매긴 값

- 그러나 원본 이미지를 그대로 convolution하는 것은 비싼 연산
  → 이 네트워크를 학습시키려면 모든 픽셀의 classification loss를 계산하고 평균값을 취해야하기 때문!

Computer Vision Tasks

Semantic Segmentation

2. Fully Convolutional



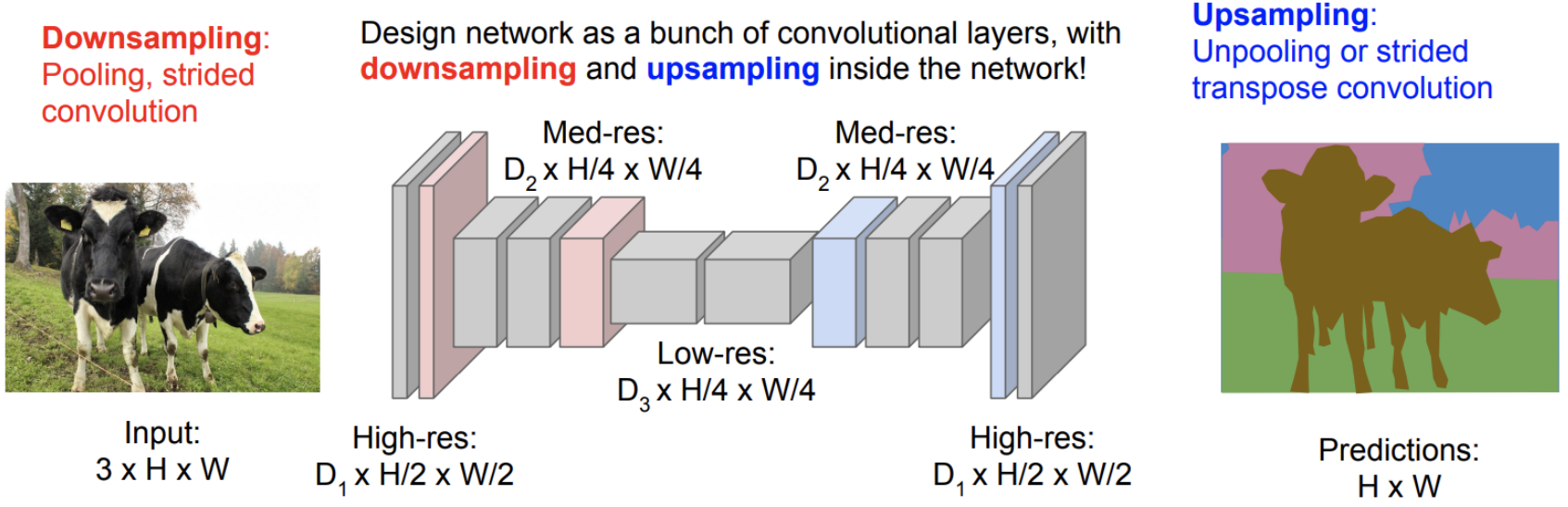
- 앞선 1,2 같은 방법은 네트워크에 고해상도 이미지가 input으로 들어오면 계산량과 메모리가 엄청 커서 감당할 수 없을 것
- 따라서 실제로는 사용을 잘 하지 않는다.
- 앞으로 설명할 것과 같이, feature map을 Down/upsampling하는게 일반적이다.

Computer Vision Tasks

Semantic Segmentation

3. Fully Convolutional with downsampling and upsampling

- 이미지의 크기를 그대로 유지하여 CNN 작업을 하던 전 방법과 다르게,
  이미지의 크기를 한번 Downsampling을 통해서 크기를 줄인다.
- 이후에 Upsampling 으로 원래의 이미지 크기로 복원하여 계산 효율성을 높인다.



- max pooling 또는 strided convolution(stride를 2를 주는 방식)을 통해 downsampling
- Unpooling과 transpose convolution 을 통해 upsampling

Computer Vision Tasks

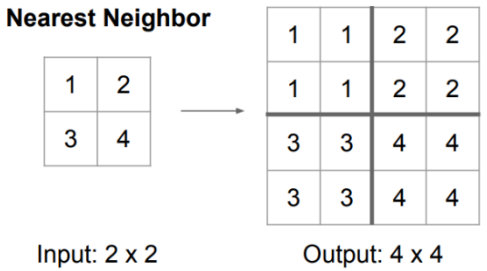
Semantic Segmentation

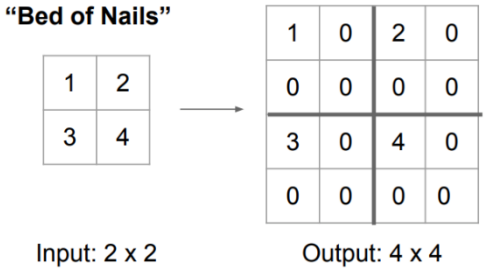
Upsampling

1) Unpooling

- Nearest Neighbor
  - pooling의 반대 작업으로, unpooling 지역의 receptive field의 값을 복제
  - 주변을 이웃값으로 채우는 것
  - Max Pooling stride값이 2인 경우 아래와 같이 같은 값으로 채워줍니다.

- "Bed of Nails"
  이와 달리 Bed of Nails는 그냥 가장자리 한쪽에만 채우고 나머지는 0으로 채웁니다.
  *Bed of Nails : zero region은 평평하고 non-zero region은 바늘처럼 값이 튀기 때문



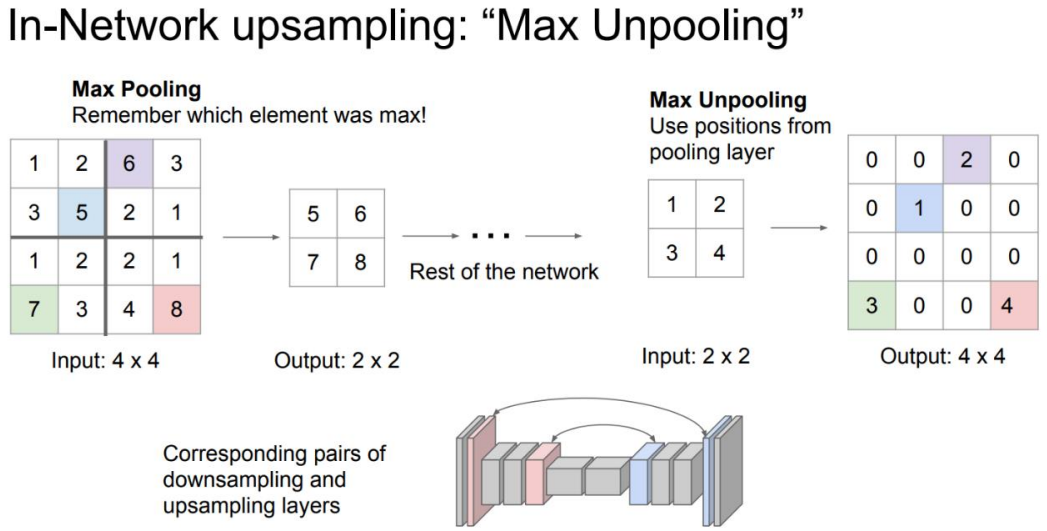


Computer Vision Tasks

Semantic Segmentation

Upsampling

2) Max Unpooling



- max pooling을 할 때 max인 값을 기억했다가 Max Unpooling 할 때 사용. 해당 위치 말고는 모두 0으로 채워넣음
- fixed function, 별도로 학습을 시키지 않는다.

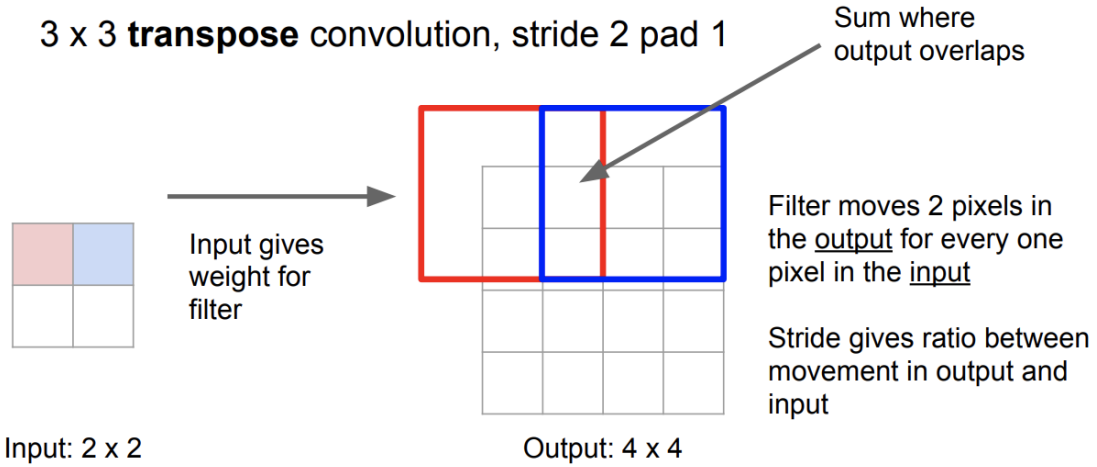
Computer Vision Tasks

Semantic Segmentation

Upsampling

3) Transpose Convolution

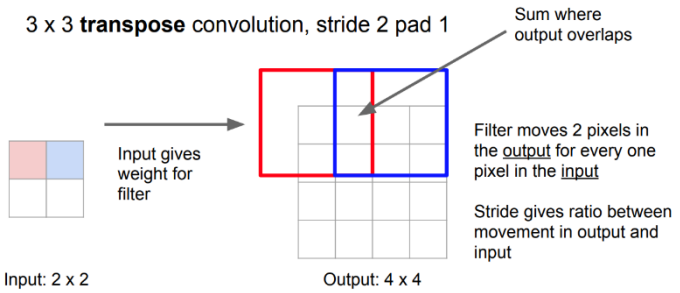
Max Pooling의 방법이 아니라,
Conv의 stride값을 이용해 Downsampling한후 Upsampling하는 방법



- Convolution과 같은 경우, 빨간색과 파란색 Convolution 영역이 겹치는 부분이 생김
- 이러한 점에서 Upsampling 시에 이를 반영하여 Upsampling 될 필요성이 있습니다.

Computer Vision Tasks

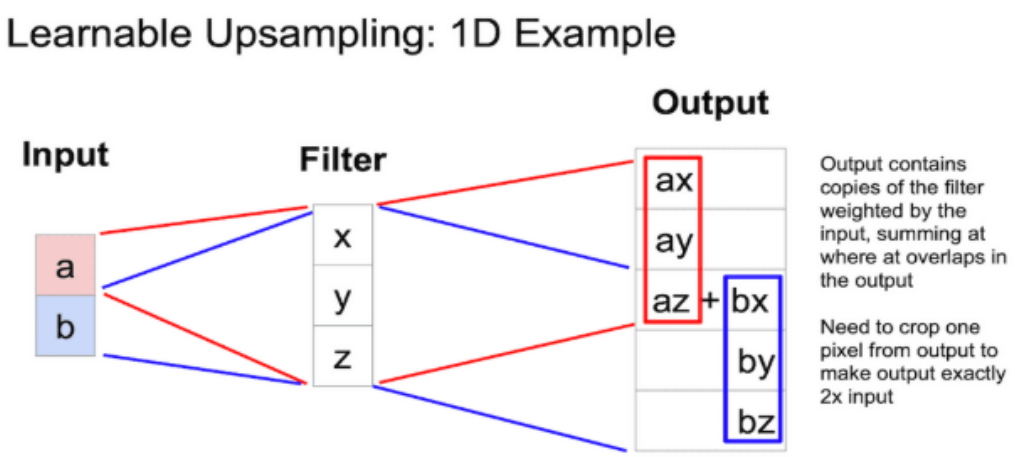
Semantic Segmentation



Upsampling

3) Transpose Convolution

- 2차원이 아니라 1차원으로 나타낸 그림
- Input에 대해서 Transpose Convolution 으로 Upsampling 하는 방법을 나타냄



- 위는 가로 방향의 성분에 대해서만 생각한 그림
- 여기서 Convolution의 pad가 1이라고 가정하고 생각
- az+bx : 3번째 요소가 겹치는 부분을 표현합니다.

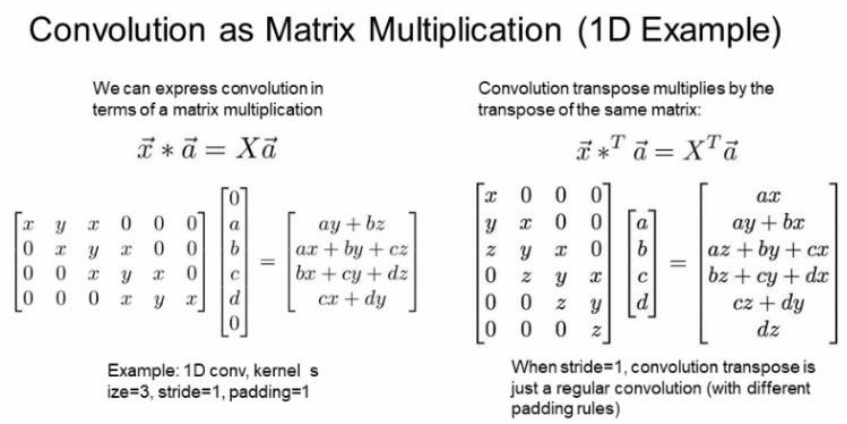
Computer Vision Tasks

Semantic Segmentation

Upsampling

3) Transpose Convolution

- 1D에서 Convolution과 Transpose Convolution을 행렬곱 형태로 나타낸 식



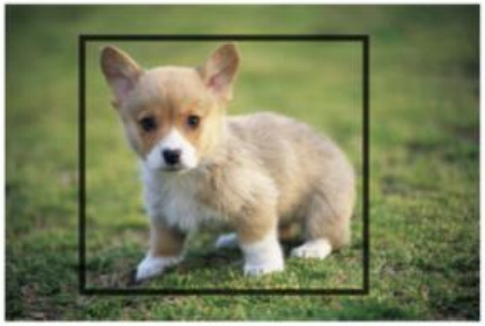
- Convolution은 필터크기 3, stride 2, pad 1인 경우
- 여기서 x,y,z 값은 Filter의 값이고, a,b,c,d는 입력값
- pad를 표현하기 위해서 a의 위와 d의 아래에 0의 값으로 채워 넣은 것을 볼 수 있음
- 이러한 방식으로 행렬곱으로 transpose convolution을 할 수 있다

Computer Vision Tasks

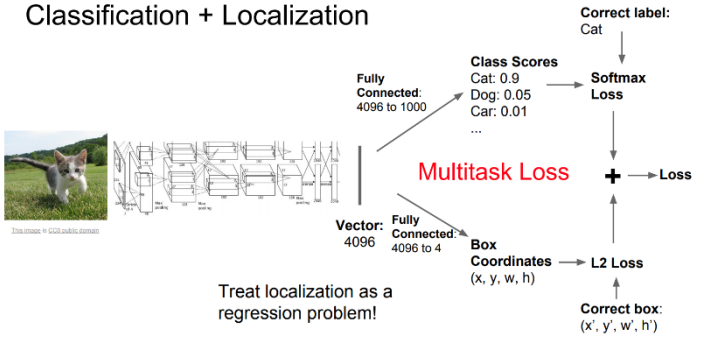
Classification + Localization

Classification + Localization

- 물체 인식(Object Detection)은 Classification + Location 두 가지 모두 필요
- 한 마리의 개를 찾아내고(Classification)
  이 개의 위치를 찾아(Location)를 찾아 Bounding Box를 찾아야함



Classification , Localization



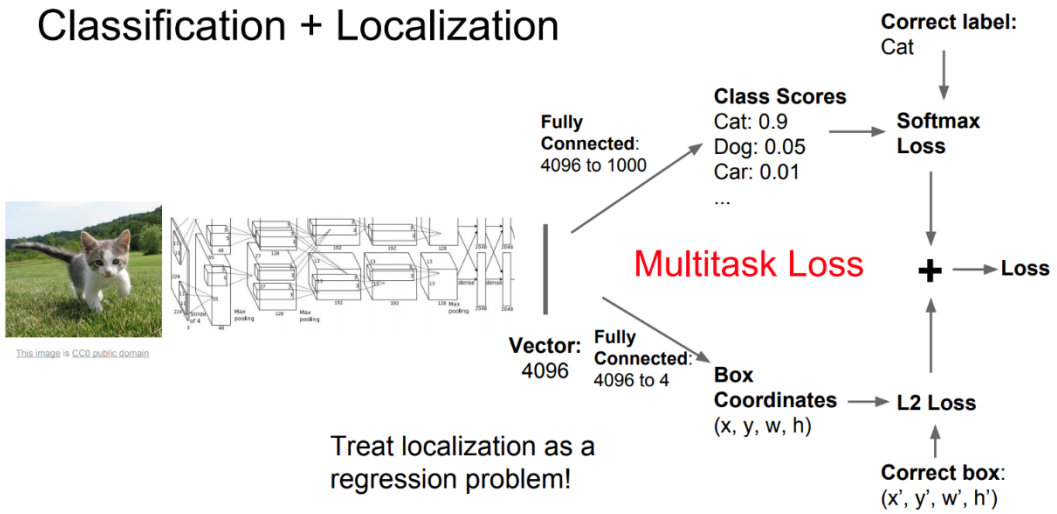
- 이미지 내에서 Classification과 Localization은 image classifier의 마지막 단에 두갈래로 나누어 작업 진행
- 한가지는 이미지에 대해서 어떤 물체가 있는지 알아내는 FC와
  나머지 하나는 이 물체의 위치를 찾아내는 FC로 여러가지 일을 해야함
- 일반적으로 앞에서 이용하는 이미지 분류기는 pre-training 된 모델을 사용하고 나머지 2개의 task 에 대해서 학습 진행

Computer Vision Tasks

Classification + Localization

Classification + Localization

- Single Object
- what the category is, where is that object in the image?
- Output : Bounding Box(around the region of label), label

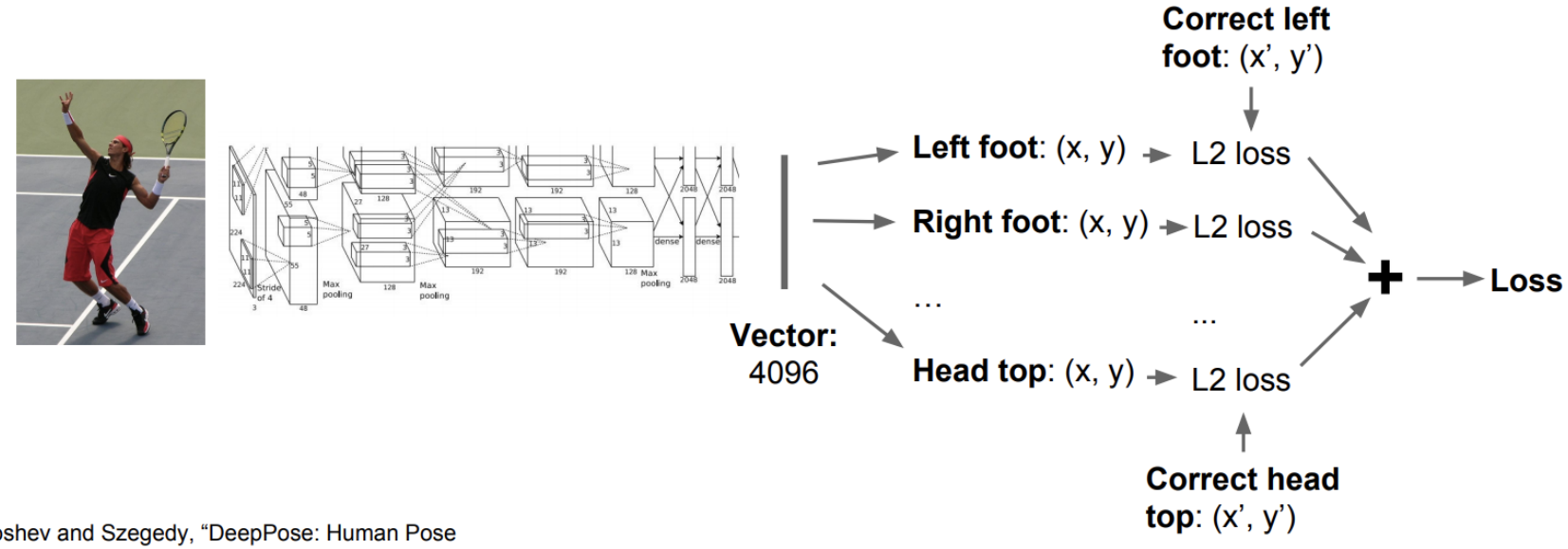


- localization은 regression 문제!
- Multi Task Loss를 계산
- ImageNet의 pretrain 모델을 사용하기도 함(Transfer Learning) * 처음부터 학습하기 어려울 수 있으므로

Computer Vision Tasks

Classification + Localization

Aside: Human Pose Estimation



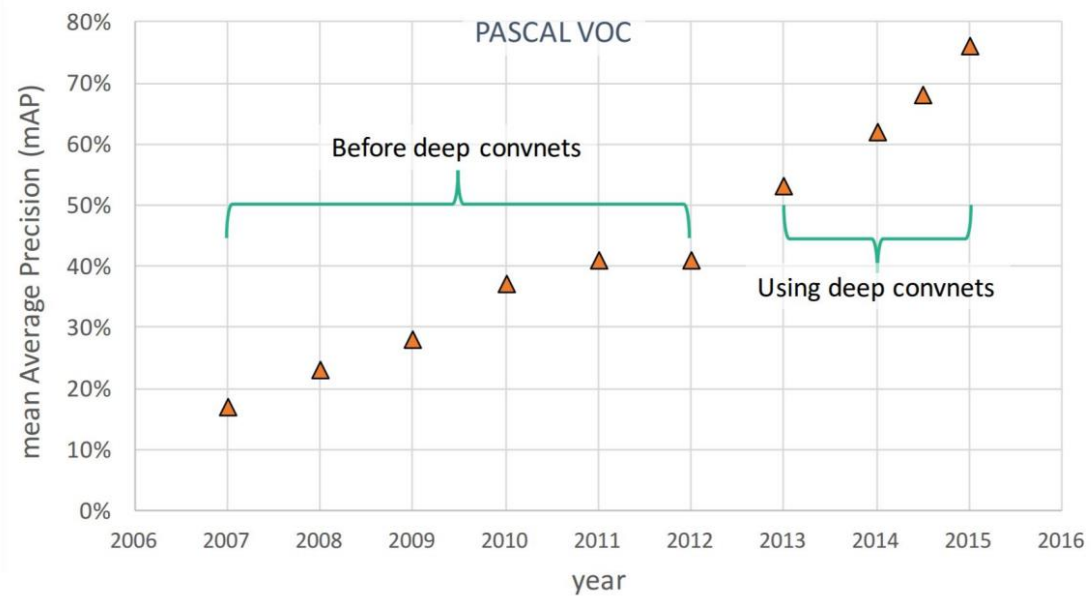
Toshev and Szegedy, "DeepPose: Human Pose Estimation via Deep Neural Networks", CVPR 2014

- Pose Estimation에도 활용할 수 있음. Image를 넣으면 14개의 join position이 나옴

Computer Vision Tasks

Object Detection

- Computer Vision의 핵심 Task
- Multiple Object
- Output : Bounding Box(around the region of label), label

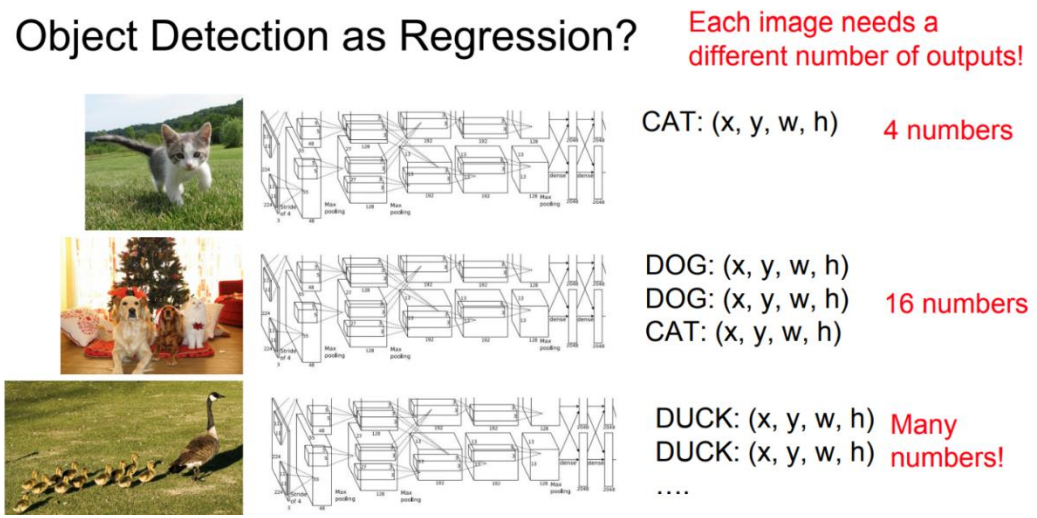


- 딥러닝을 활용한 이후부터 점점 성능이 좋아지고 있음

Computer Vision Tasks

Object Detection

- Localization과의 차이점은 동일한 종류의 물체가 여러 개 있다면
- Object Detection은 모두 잡음(Localization은 1개로 취급)



- 이제 까지는 하나의 물체에 대한 Detection과 Location에 대해서 배움
- 하지만 실제 상황에서는 한 이미지에 물체가 몇 개인지 예측할 수 없다.
- 따라서 이를 해결하기 위한 방법이 필요함

Computer Vision Tasks

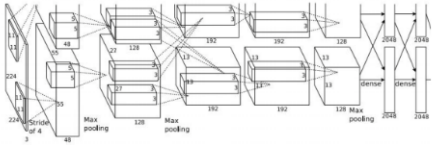
Object Detection

방법1. Sliding Window Approach

- 이미지를 임의에 크기로 잘라 탐색하며 물체인지, 배경(아무것도 아닌지) 탐색

- 단점
  1) 어디에 존재하는지 알지 못함
  2) 몇개 인지 알지 못함
  3) 크기도 알 수 없음
  4) 하나씩 CNN으로 하기에 시간이 부족함

Apply a CNN to many different crops of the image, CNN classifies each crop as object or background



Dog? NO
Cat? YES
Background? NO

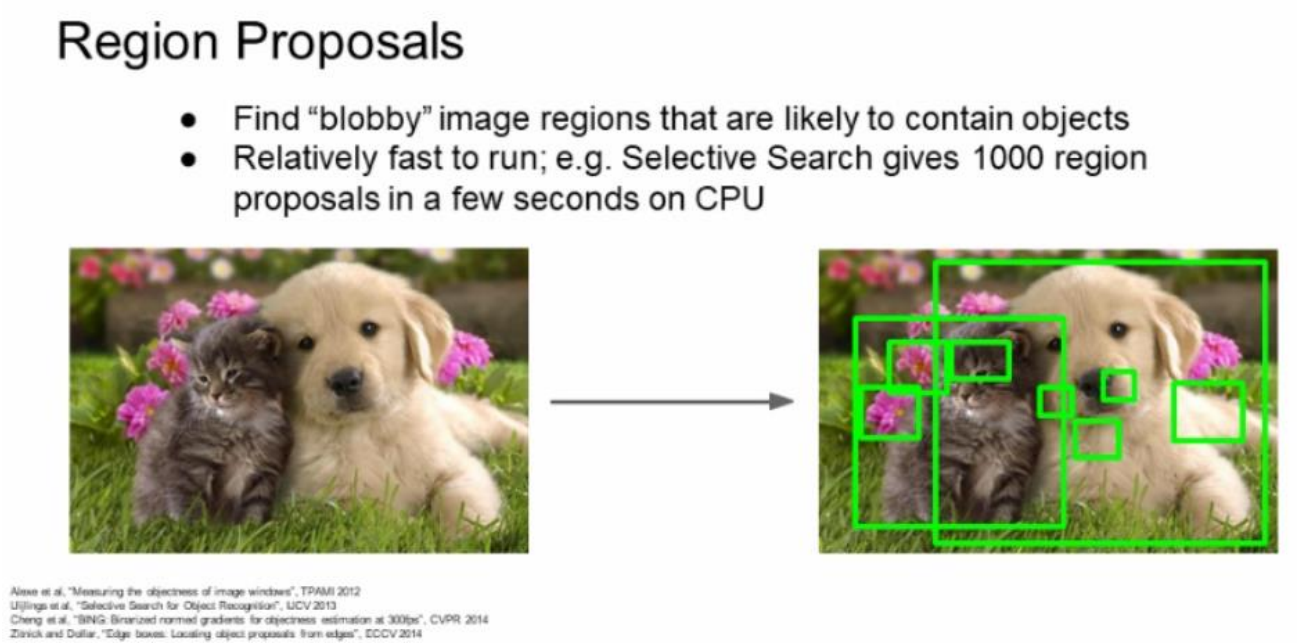
Problem: Need to apply CNN to huge number of locations and scales, very computationally expensive!

Computer Vision Tasks

Object Detection

방법2. Region Proposals

- 물체가 있을법한 후보(Region proposals)을 찾아내는 것
- 물체가 '뭉친'곳을 찾아내 region을 selective search
- 이러한 selective search 방법은 딥러닝 방식이 아님

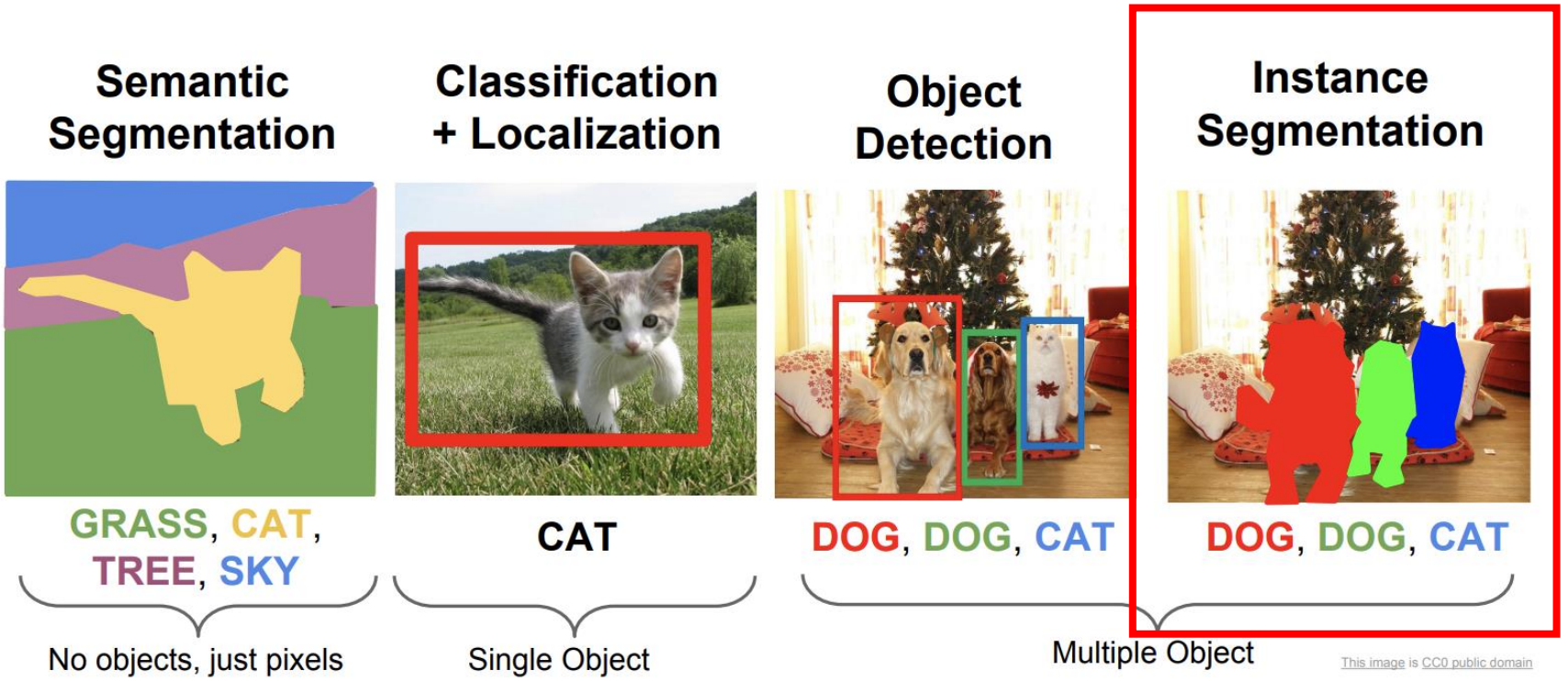


Computer Vision Tasks

Instance Segmentation

Instance Segmentation

- 이는 Semantic segmentation과 Object detection을 섞은 것
- 이미지 내의 두 마리의 개가 있으면 이 두 마리를 구분해내야 한다



Computer Vision Tasks

Instance Segmentation

- 일반적으로 Mask R-CNN과 같은 2-stage detector에서는
  먼저 object들을 bounding box를 통해 localization시킨다

- 그 후 위에서 class별로 output 채널을 만든 것과 같이 localize된 RoI마다 class의 개수만큼
  binary mask(instance인지 아닌지) 마스크를 씌워준다