※ Object Detection : Classification + Box level Localization

※ Semantic Segmentation

- Pixel 단위로 영역을 표시 (Pixel의 값은 Class)
- Label 되어 있지 않는 Pixel은 Background로 정의
- Ignore Label : 경계의 회색부분 (숙련된 사람에 따라서 경계가 반영된다)
- Vision이 복잡 할 수록 Label에 들어가는 비용은 증가한다

※ Semantic Segmentation – Training Data

- Pixel 단위로 Label된 Image가 된다 (Raw 단위로 필요)
- Pixel의 Class만 찾지 Object은 찾지 않음 (Instance 단위로 찾지 못한다)

※ Pre-CNN

- 문제점 : Hand Designed Representation
- Labeling의 규모가 방대함 (C^N : N은 Pixel수, C는 Class수)

※ Semantic Segmentation with CNN

- Search Space가 크고 Representation 사용 가능
- Region Based Proposal + Classification

  : Bounding Box의 Candidate를 보는 것. 1) 작은 Pixel들을 뽑고 2) Grouping 3) Tight한 Bounding Box 찾기
- 단점 : Segmentation 정확도가 Region Proposal 정확도에 귀속된다

  : End to End Training이 안됨

  : Pipeline이 독립적으로 각 Layer와 별도의 Module임

※ Convnet for Image Classification

- CNN을 Design한 목적 : End to End를 구성하기 위해
- Combination of Convolutional + Fully Connected Layers

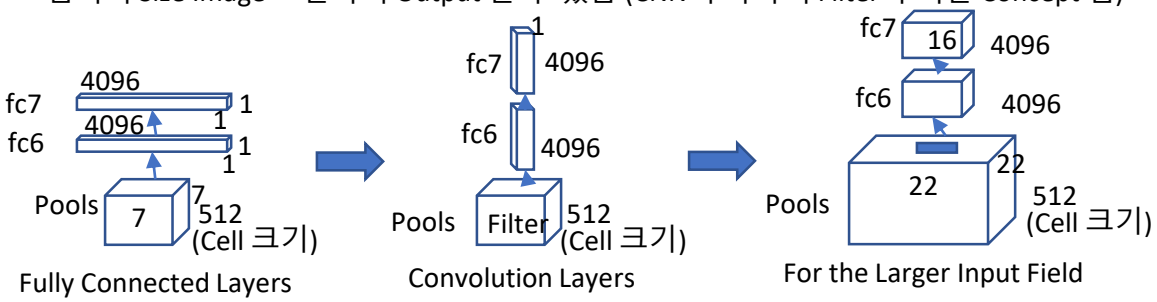
  : 앞단은 Convolution Layer (Operative Filtering, 임의의 Size Image에 사용가능)

  뒷단은 Fully Connected Layers (Input Image에 Size가 주어져야 함

※ Fully Convolution Network

- Region 단위로 하고 싶으면 Fully Connected Layer를 Convolution Operation으로 변경하면

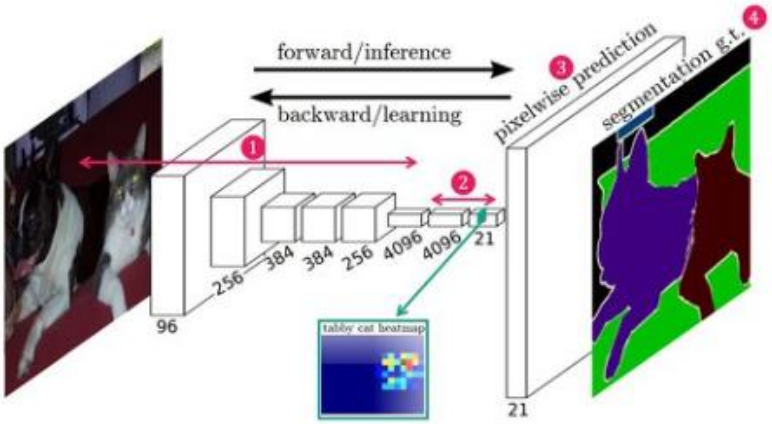
임의의 Size Image로 받아서 Output 할 수 있음 (CNN이 하나의 Filter가 되는 Concept 임)



※ FCN – Fully Convolutional Network

- End to End CNN 구조

- Region Proposal 대비 End to End 가능



네트워크 구조

(1) Feature Extraction : 일반적인 CNN의 구조에서 많이 보이는 conv layer들로 구성되어 있습니다. **(2) Feature-level Classification** : 추출된 Feature map의 pixel 하나하나마다 classification을 수행합니다. 이 때 classification된 결과는 매우 coarse합니다. (그림 3에서 초록색 박스에 tabby cat class에 대한 Classification 결과 참고) **(3) Upsampling** : coarse 한 결과를 backward strided convolution 을 통해 upsampling하여 원래의 image size로 키워줍니다. **(4) Segmentation** : 각 class의 upsampling된 결과를 사용하여 하나의 Segmentation 결과 이미지를 만들어 줍니다.

- 단점 : Score Map의 Resolution이 낮다 (만족한 Segmentation을 얻기 힘듬)

→ Resolution이 작아 Slope가 작아진다 (Shape이 Detail하지 못하다)

: Receptive Field가 고정 (유동적이어야 전체나 주변을 볼 수 있다)

→ Receptive Field Size가 Fix되어 있어서 FCN 결과 Class가 혼재 된다

※ Semantic Segmentation 향상 방안

1) Encoder-Decoder Network

2) Atrous Convolution

3) Spatial Pyramid Pooling

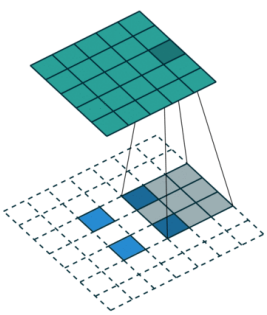
※ Encoder-Decoder Networks

- Original Image의 정보를 뽑아 내는 것 (많은 정보는 손실 됨)

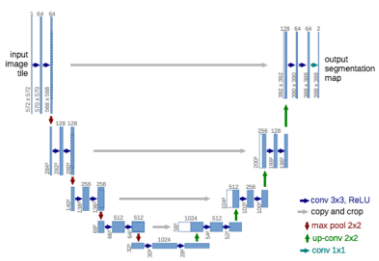
- 정보를 압축 (공간, Semantic) & 압축을 품 (압축을 풀어 정보를 복원 하는 것)

※ Encoder-Decoder 의 종류

- Deconvolution for Upsampling



- U-net



: 높이 – Resolution

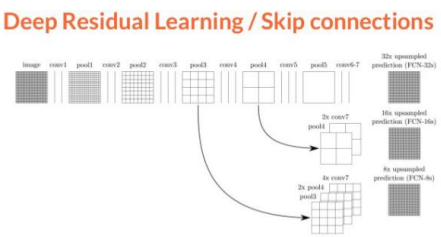
: Width – Feature의 Dimension

: 높이는 줄이고 넓이는 넓어짐

: 공간 정보과 Semantic 모두 사용

: 잃어 버리기 전의 정보를 전달하는 역할도 함

- Skip Connection

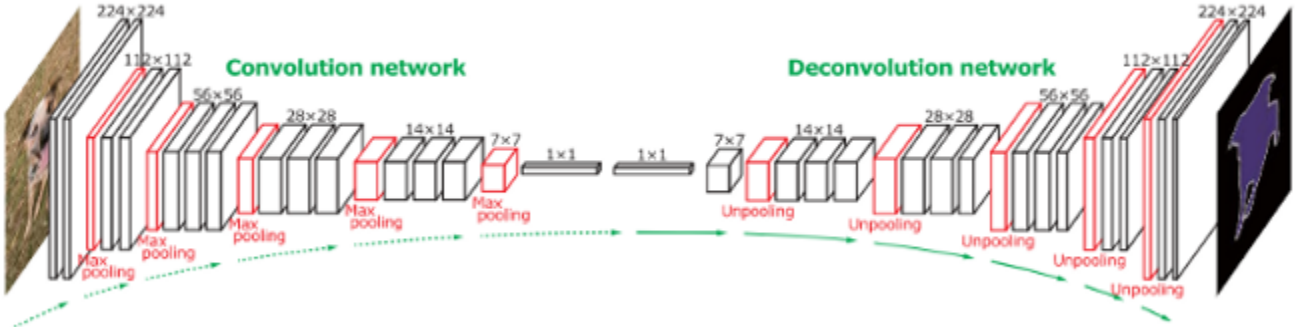


: Pooling 단계에서 Class는 명확치 않음 (정보상 압축은 손실 되지만 Shape은 더 가지고 있다)

: 중간 Output의 Layer에서 Decoding 함

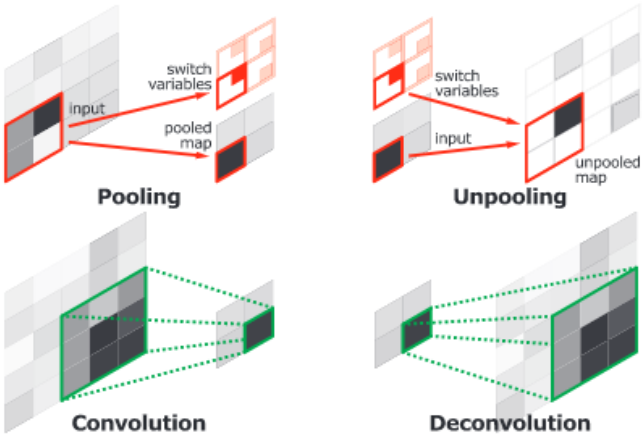
※ Deconvolution Network

- Shared Pooling Switches
- 공간 정보를 잃어 버렸는데 어떻게 복원을 할까?
- Convolution → Unpooling → Unconvolution



※ Operations in Deconvolution Network

- Pooling Switches : Pooling에서 어느 위치였는지 정보를 가지고 있다
- Unpooling과 Deconvolution은 set의 개념임



※ Summary : Encoder – Decoder Network

- Encoder는 공간 정보를 잃어 버리고 Decoder 하면서 학습한다 (잃어 버린 정보를 유추)
- Skip Connection, Upsampling, Pooling Switches

※ Semantic Segmentation : Atrous Convolution

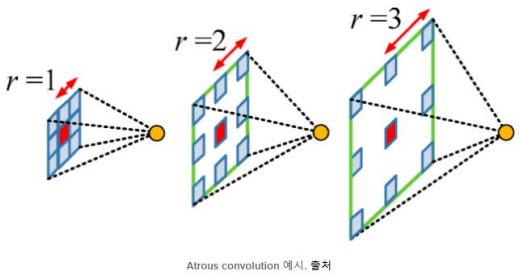
- FOV (Field of View) in Segmentation
- Receptive Field : Input Region의 Size

  → Receptive Field를 늘리면 공간 정보를 많이 잃어 버린다

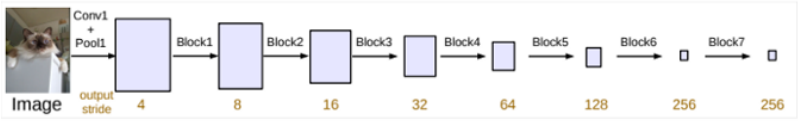
  → Convolution Field를 늘리면 Overfitting이 생김

※ Atrous Convolution

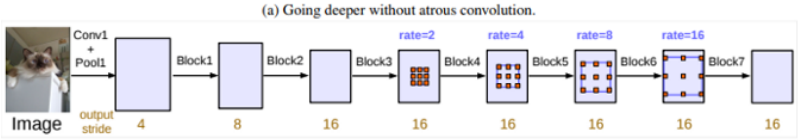
- Convolution Filter의 Weight을 늘리지 않고 영역을 늘리는 것



Atrous convolution 예시, 출처



(a) Going deeper without atrous convolution.



(b) Going deeper with atrous convolution. Atrous convolution with $rate > 1$ is applied after block3 when $output\_stride = 16$.

※ Self - Attention

- 각 Feature의 Point가 서로 얼마나 비슷 한가? (Simility)

※ Semantic Segmentation : Spatial Pyramid Pooling

- 다양한 FOV로 보는 것
- Atrous의 Rate을 다양하게 하면 Spatial Pyramid Pooling과 같은 효과를 볼 수 있다
- Object Detection을 FOV Size별로 하여 Detection 한다

※ Multi-Scale FOV by Spatial Pyramid Pooling

- 적절한 FOV를 찾기 어려워 다 보겠다
- Pooling Layer의 Size를 Multi

※ Visual Object Tracking

- Object을 시간에 따라서 Detection 하는 것
- Initial Frame에서 사람을 주어지면 그 외 Frame에서 찾아야 함

※ Visual Object Tracking의 Learning Perspective

- One Shot Learning
- Label Data는 Initial Frame에서 주어진다
- 변화하는 물체와 Appearance에 적용
- 추적된 결과에 따라 Training 한다 (Selfvised Learning)

※ Probabilistic Tracking

- 각 Frame에서 Object를 추적하는 확률 (물체가 존재할 확률)
- Bayes Rule : p(z|x) = p(x|z)p(z) x: object loction(z관측할 수 있음), z: frame(관측할 수 없음)

  P(x|z) = Object가 주어 졌을때 해당 영역에 물체가 존재할 확률 (Likelihood)

  p(z) : 사전지식
- Sequential Bayesian Filtering : 시간대 별로 유추 하는 것
- Sequential Markov Chain Monte Carlo : 해당 위치에 물체가 위치할 확률

단점 : 이전 Frame의 물체의 위치에 ego서 Integration 해야 한다