ConvNets for Computer Vision

HOML – chapter14

TAVE Research DL001

Changdae Oh

2021. 01. 24

Contents

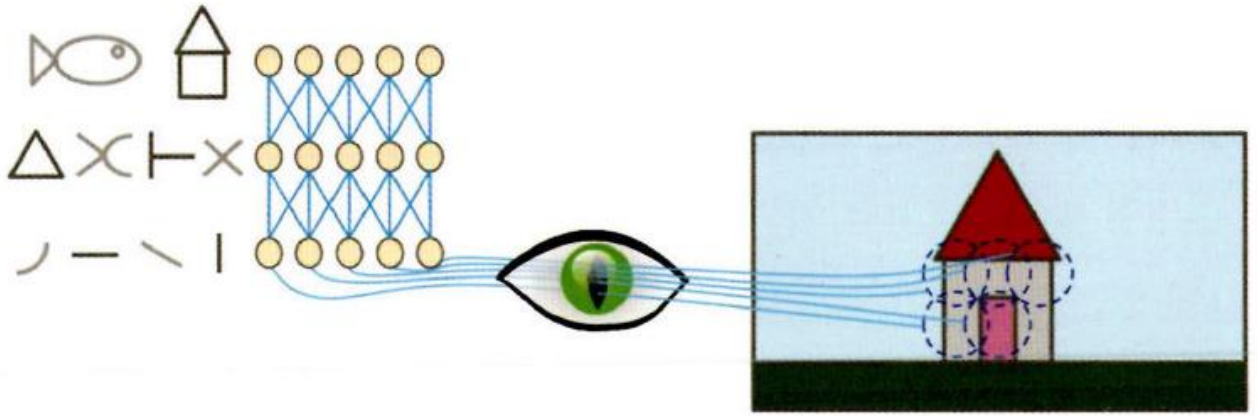
Contents

Introduction to CNN

Insights into the structure of the visual cortex

- 시각 피질 안의 많은 뉴런이 작은 국부 수용장(local receptive field)을 가짐.
- 뉴런들이 시야의 일부 범위(수용장) 안에 있는 시각 신호(패턴)에만 반응한다 !
- 시각 신호가 연속적인 뇌 모듈을 통과하면서 뉴런들이 더 큰 수용장에 있는 더 복잡한 패턴에 반응한다.

➡ *고수준 뉴런이 이웃한 저수준 뉴런의 출력에 기반한다 !*

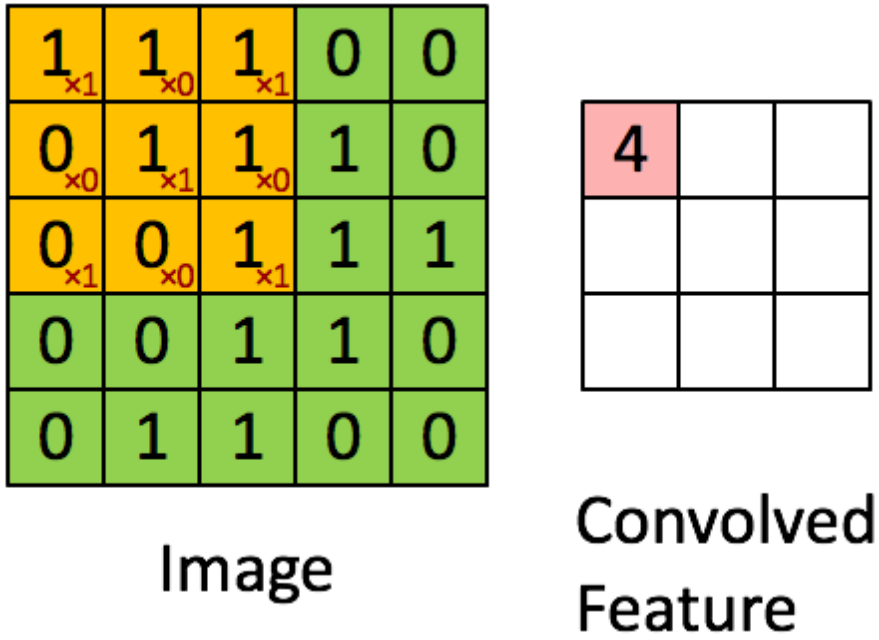


Introduction to CNN

What is a convolution ?

전형적으로 이미지로부터 특성을 추출하기위해 사용됨.

- 행렬에 적용되는 Sliding window 함수라고 생각하자.
- 이 sliding window 는 filter 혹은 kernel 이라고 불린다.
- N*N 크기의 필터를 사용해 그것의 값들을 원본 행렬의 값들과 element-wise multiply하고 결과값들을 더한다.



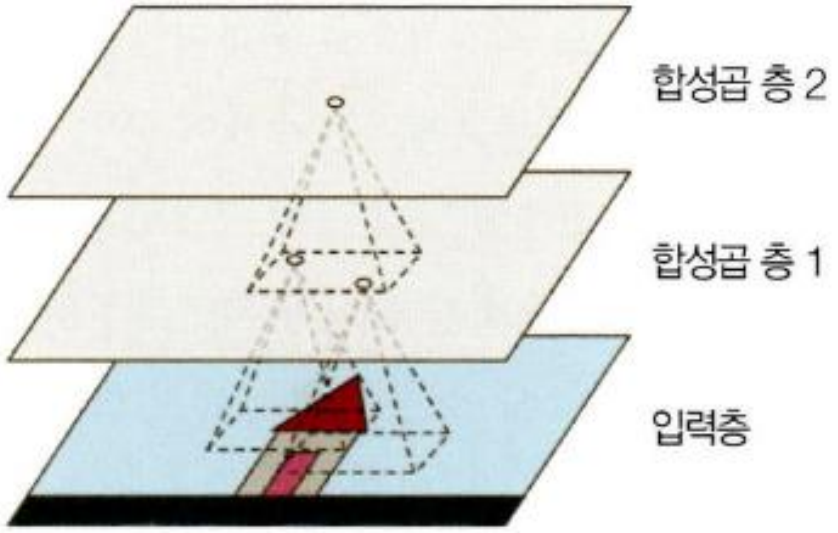
http://deeplearning.stanford.edu/wiki/index.php/Feature_extraction_using_convolution

< Various terms used on CNN >

- Channel
- Filter
- Kernel
- Stride
- Receptive field
- Padding
- Pooling
- Feature Map
- Activation Map

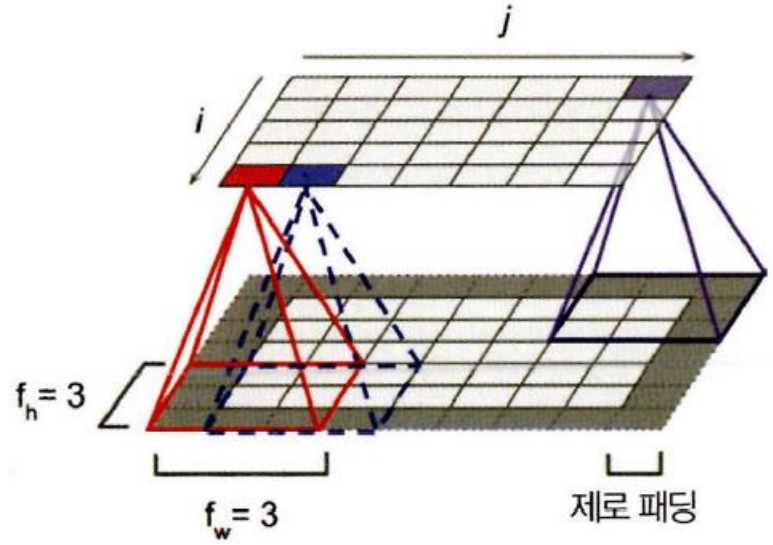
Introduction to CNN

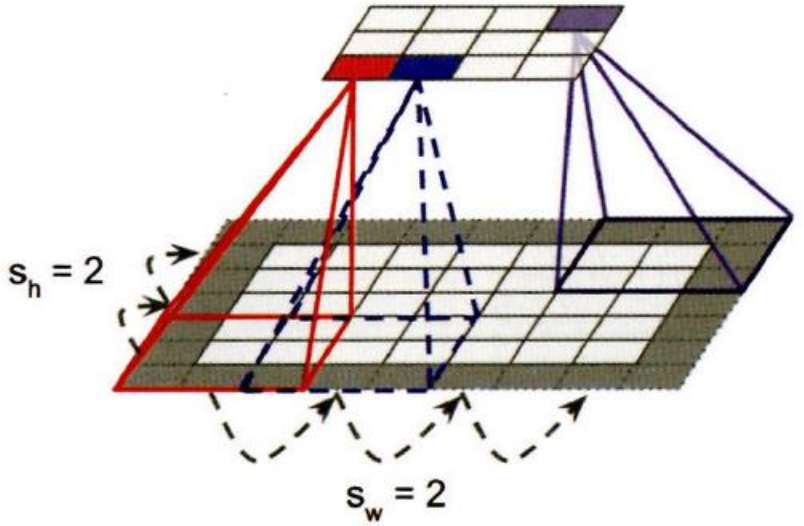
Convolutional layer



- 각 합성곱 층은 원본이미지 혹은 이전 합성곱 층 출력 map의 모든 pixel에 연결되는 것이 아니라 해당 합성곱 층 뉴런의 receptive field 안에 있는 pixel에만 연결된다.
- 하위 층에서는 저수준 특성에 집중하고
  이후의 층들에서 고수준 특성으로 조합해나가는 계층적 구조

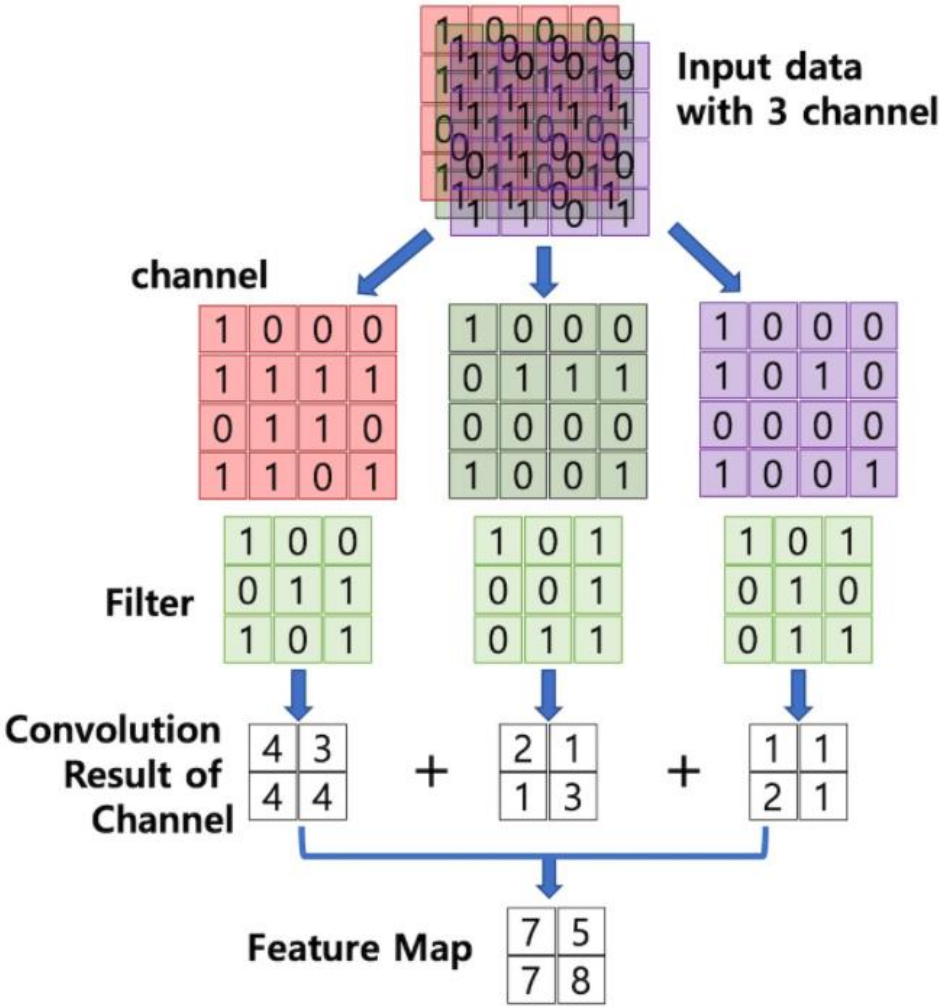
Stride & Padding

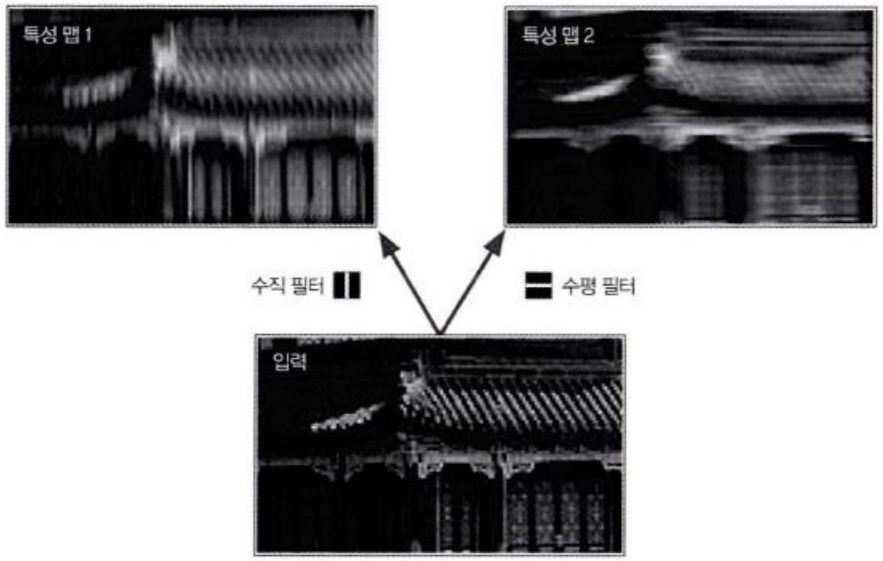




Introduction to CNN

Channel / Filter / feature map





- Feature map은
  필터를 가장 크게 활성화시키는 이미지의 영역을 강조한다.

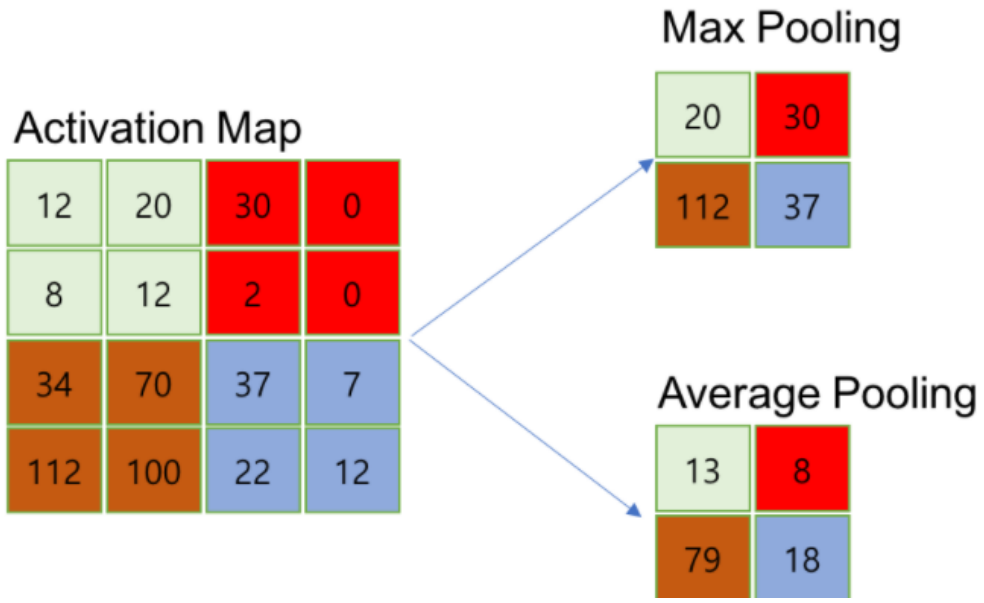
합성곱 층 뉴런의 출력

$$z_{i,j,k} = b_k + \sum_{u=0}^{f_h-1}\sum_{v=0}^{f_w-1}\sum_{k'=0}^{f_{n'}-1} x_{i',j',k'} \times w_{u,v,k',k} \quad \text{여기서} \begin{cases} i' = i \times s_h + u \\ j' = j \times s_w + v \end{cases}$$

Introduction to CNN

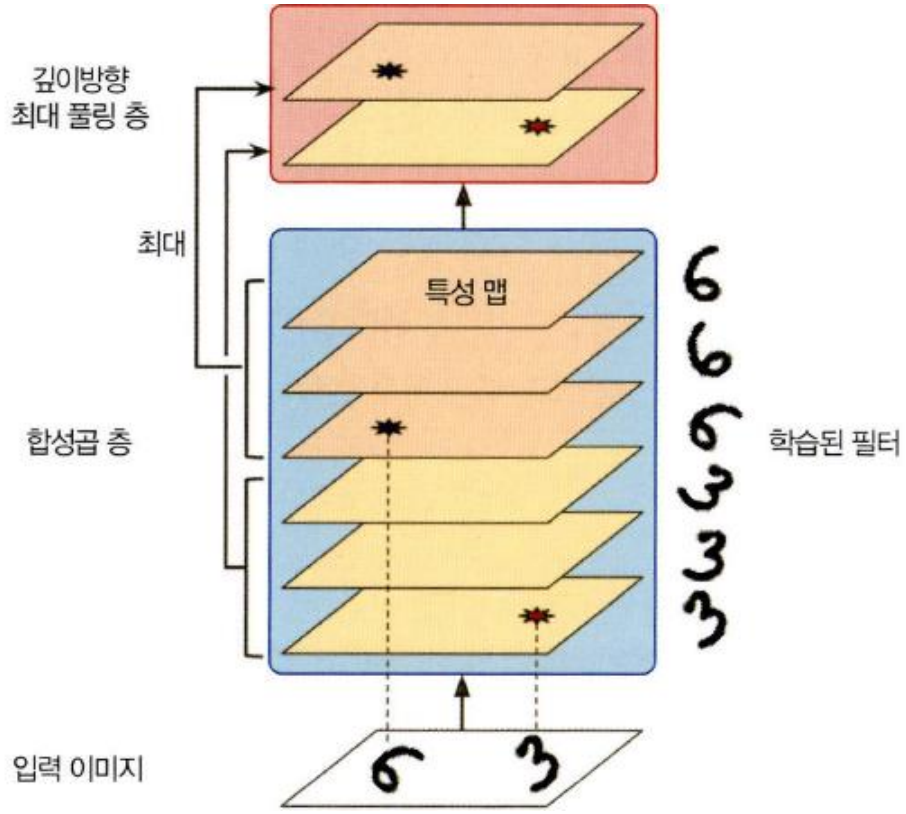
Pooling layer

자연계의 다양한 신호들은 Sparse 함.



- 2*2 pooling kernel, stride = 2
- Local pooling
- 계산량과 메모리 사용량, 파라미터 수를 줄이는 효과
- 입력 데이터의 변화에 대해 일정수준의 불변성을 제공

Depthwise(깊이방향) max pooling



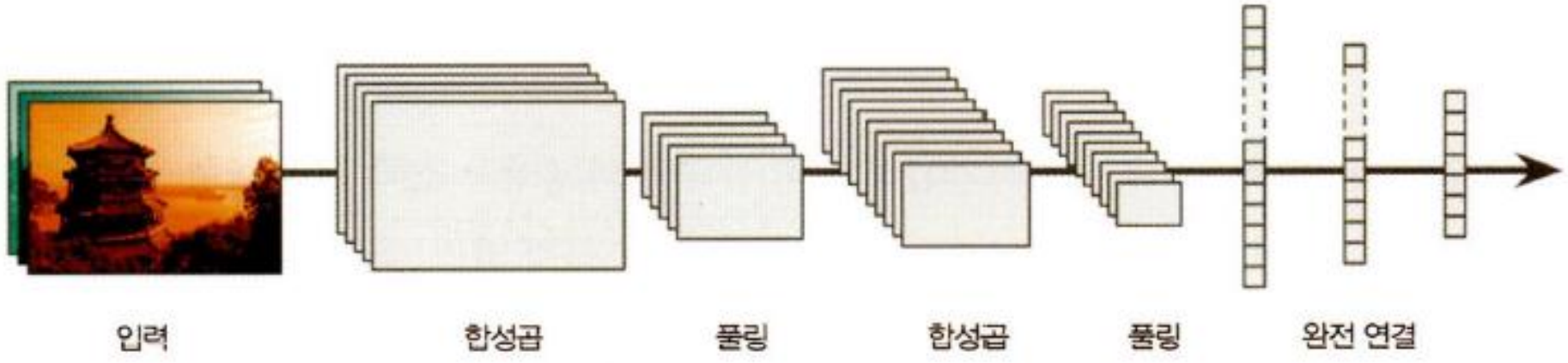
- 데이터의 회전, 두께, 밝기, 왜곡, 색상 변화 등 어떤 것에 대해서도 불변성을 학습할 수 있음

Contents

CNN Architectures (case study)

Traditional CNN architecture

- Task : image classification



- '(Conv + ReLU) 몇 개 + Pooling' 반복
- 데이터 차원은 줄어들고 채널은 증가
- 최상위에 FC layer 몇 개를 추가 후 softmax 층을 거쳐 최종 예측 수행

< Note >

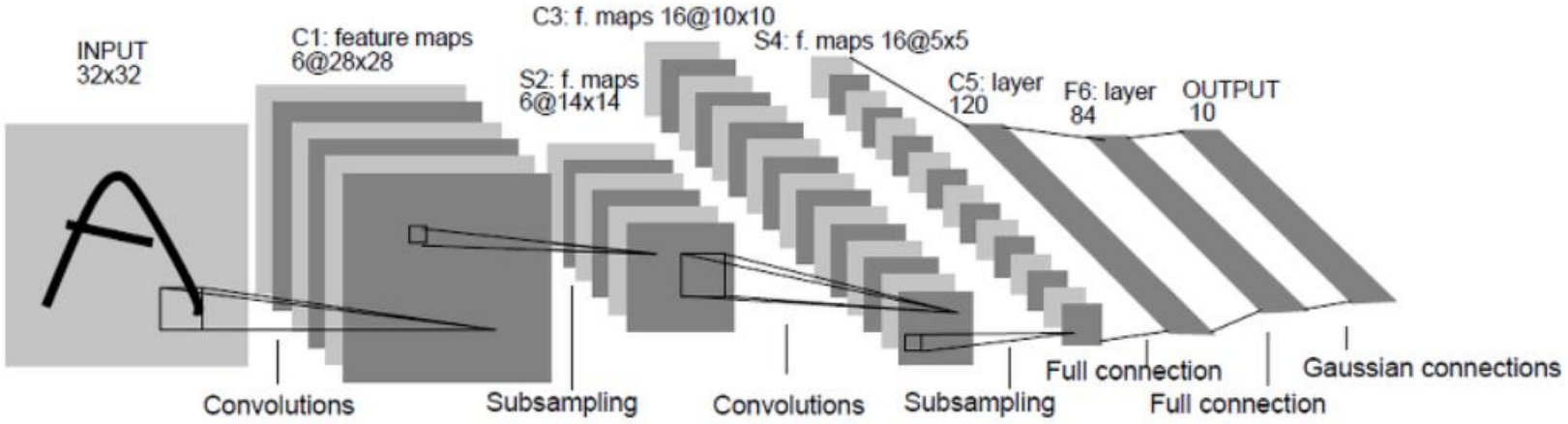
일반적으로 합성곱 층을 쌓을 때, 5*5 커널 합성곱 층 하나 대신
3*3 커널 합성곱 층 2개를 쌓는 것이 더 선호된다.
필요한 파라미터와 계산량이 적어지고 더 나은 성능을 내기 때문.
(*예외) : 입력층 이후에 오는 첫 번째 합성곱 층에서는 비교적 큰 필터 사용.

< idea >

- Local receptive fields
- Shared weights
- Sub-sampling

CNN Architectures (case study)

LeNet-5



- 입력 제외 총 7개의 층 (conv, pool, conv, pool, fc, fc, output)
- Task 대상 이미지가 28*28 pixel인데, 패딩을 통해 32*32 pixel로 변경
  (식별하고자하는 패턴을 centering하기위함.)
- 이외의 층에서는 패딩 사용안했음. -> 점진적으로 데이터 차원 감소
- < Average pooling >
  * 일반적으로 파라미터가 없는 Pooling층에 weight, bias 파라미터 추가
  * 이후 활성화함수 적용
- C3 합성곱 층에서 선택적 연결 수행. Network의 symmetry한 성질을 없애기 위함.

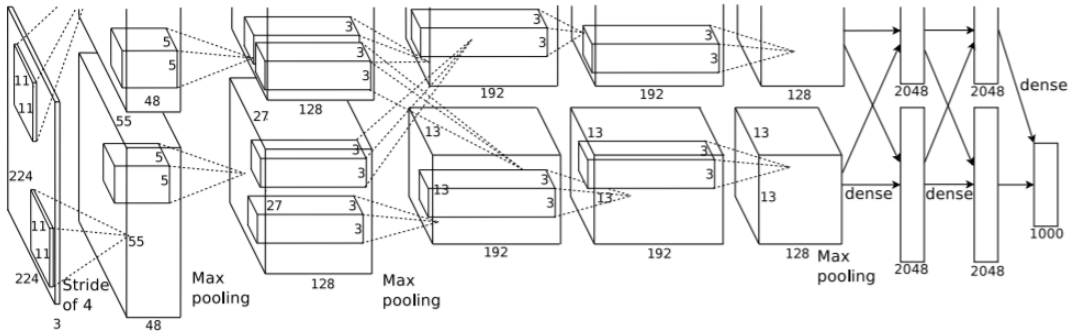
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

TABLE I

EACH COLUMN INDICATES WHICH FEATURE MAP IN S2 ARE COMBINED BY THE UNITS IN A PARTICULAR FEATURE MAP OF C3.

CNN Architectures (case study)

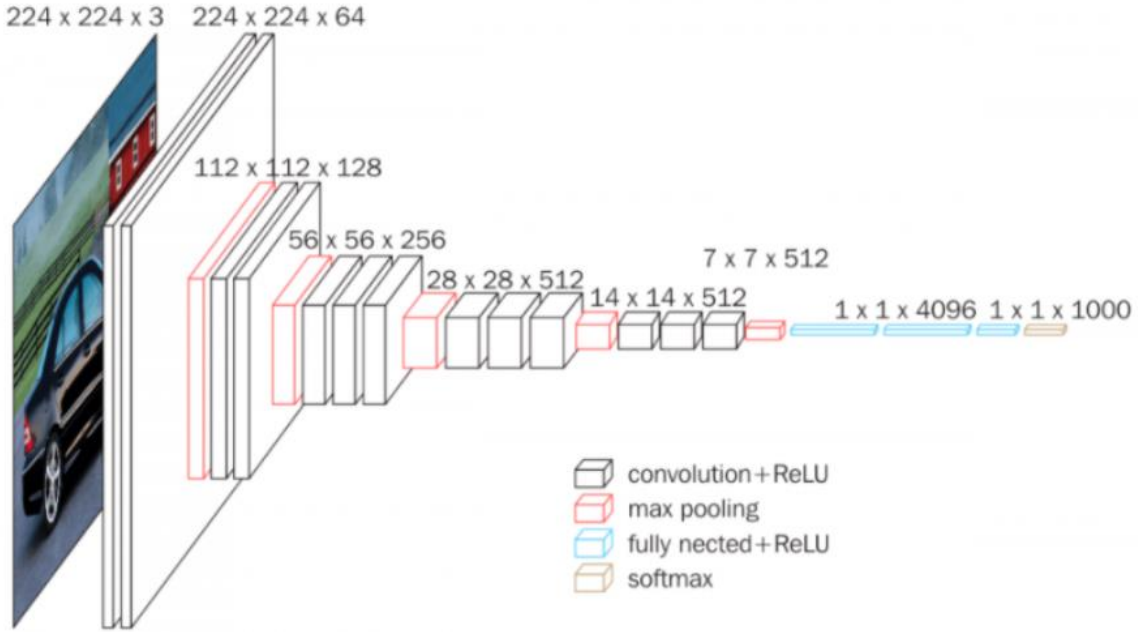
AlexNet



- 최초로 합성곱 층 연속으로 쌓음 (conv, pool) -> (conv, conv)
- 규제
  * Dropout
  * Data augmentation
- 정규화 : LRN (local response normalization)
  * 첫 번째, 두 번째 conv이후 적용
  * 아이디어는 좋으나 효용이 적음

$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$

VGGNet

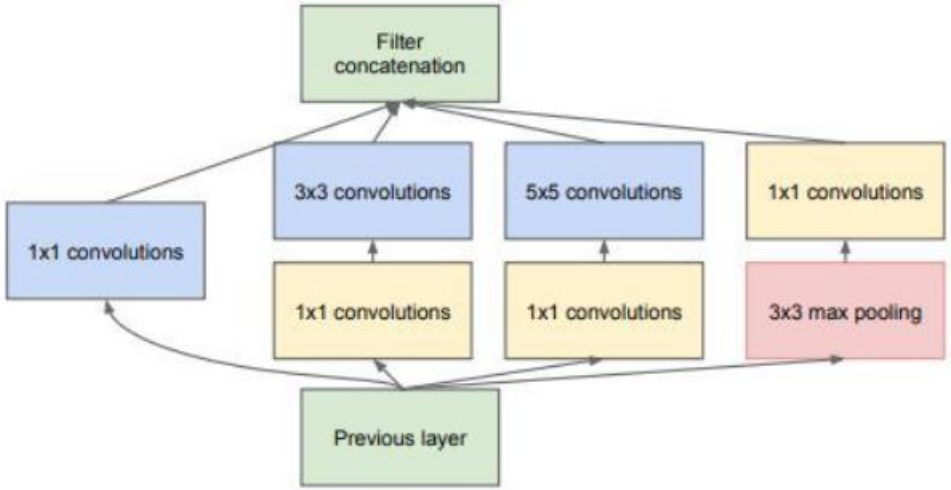


- Basic한 CNN구조. 이전보다 깊어진게 특징
- 사용하는 conv 필터 크기가 항상 3*3
- 구성이 간단하여 다양하게 응용하기 좋다.

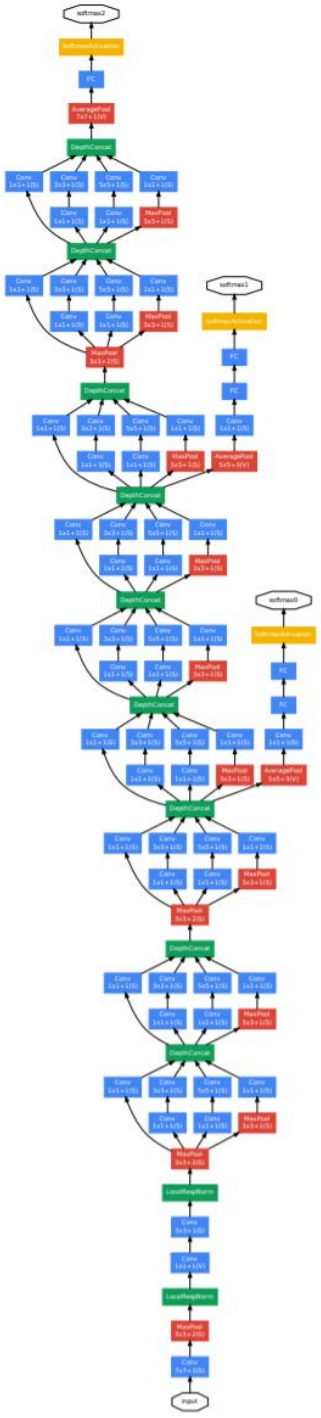
CNN Architectures (case study)

GoogLeNet

< Inception module >

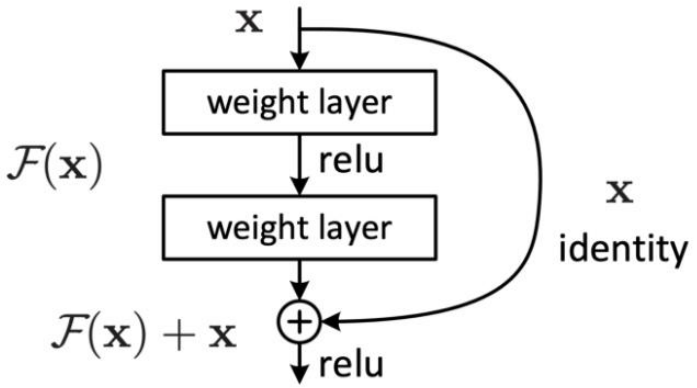


- '인셉션 모듈' 이라는 서브 네트워크 도입 (Network in Network)
- 1*1 conv 사용
  * <channel 수 조절>, <연산량 감소>, <풍부한 비선형성 추가>
- 학습과정 중간 평가를 위한 부가적인 분류기 추가 포함 – 전체손실과 가중합 (아이디어만 좋았다..)
- Global Average Pooling 을 출력층 전에 두어 파라미터 효과적으로 감소
- LRN, drop out

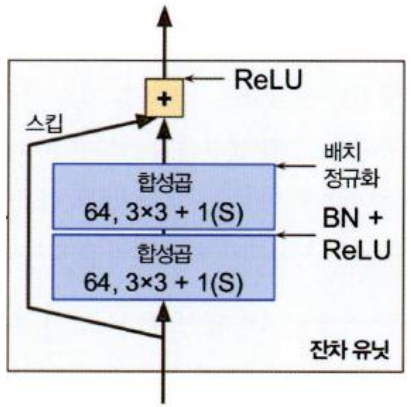


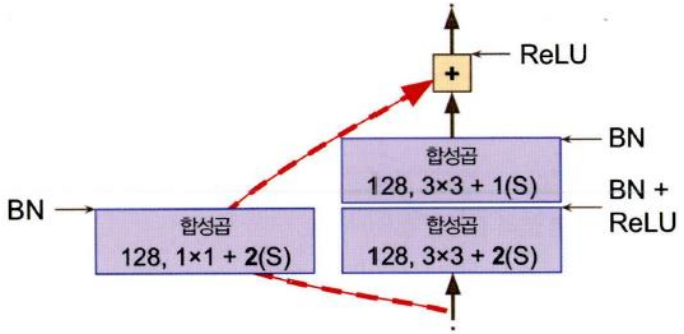
CNN Architectures (case study)

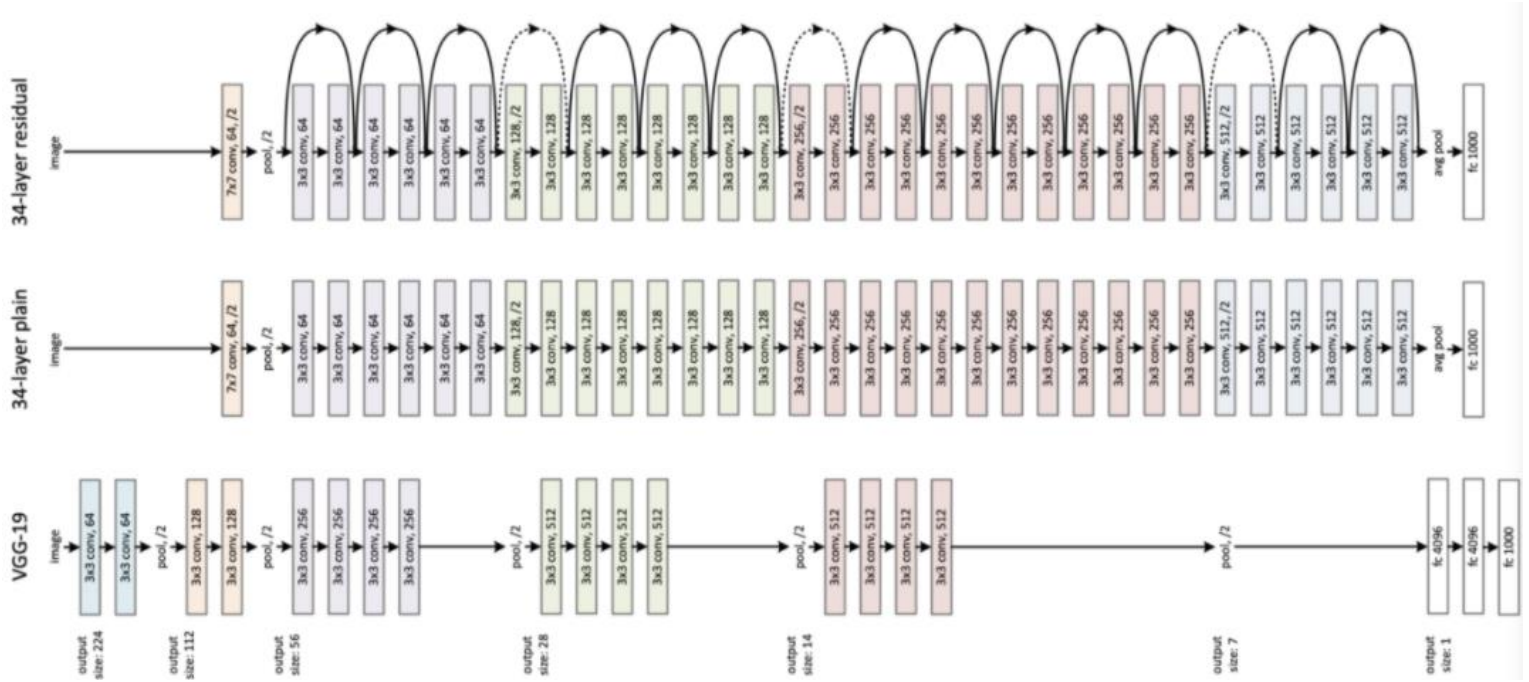
ResNet



< residual unit >



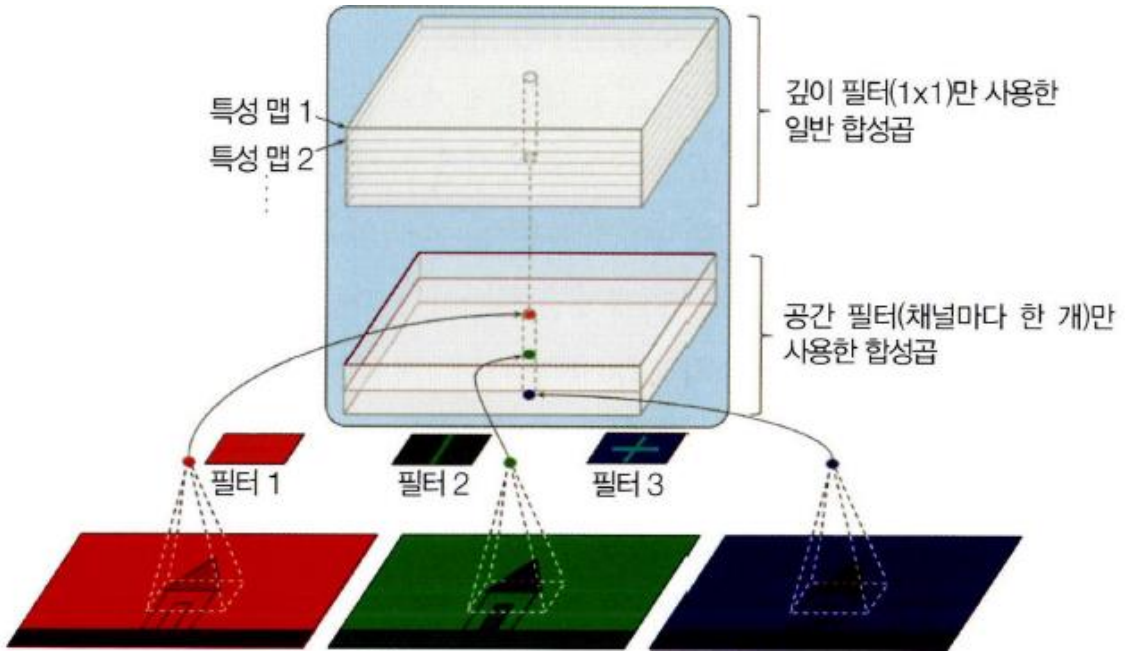




- Shortcout 연결을 통해
  Vanishing / Exploding gradients 문제 해소
- BN을 모든 conv layer이후 사용

CNN Architectures (case study)

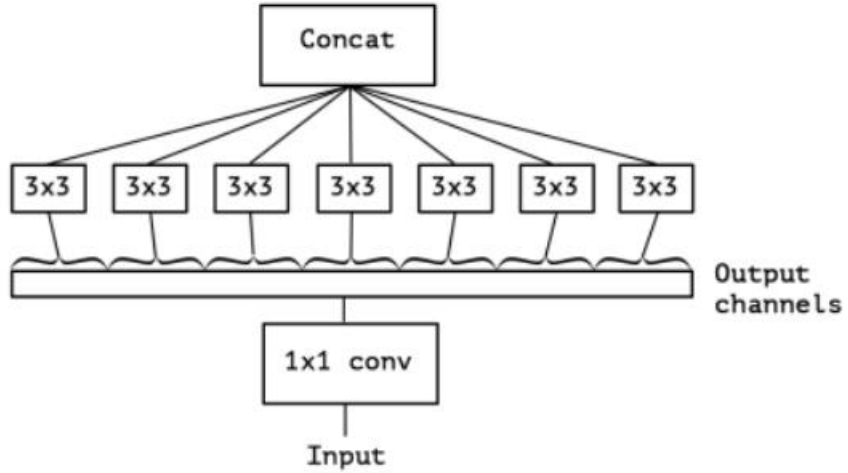
Xception



Depthwise separable convolution layer

(깊이별 분리 합성곱 층)

Figure 4. An "extreme" version of our Inception module, with one spatial convolution per output channel of the 1x1 convolution.



- Cross-channel conv : depthwise conv (3*3)
- Spatial conv : pointwise conv (1*1)

- 기존의 conv layer에서 n*n 필터를 여러 개 사용하여 공간기준, 채널기준 패턴을 한 번에 학습하려 했다면, Xception은 공간, 채널 패턴을 따로따로 고려
- 파라미터/메모리사용/계산량 측면에서 효율적이며, 일반 합성곱 층보다 성능이 잘 나오는 경향이 있음.

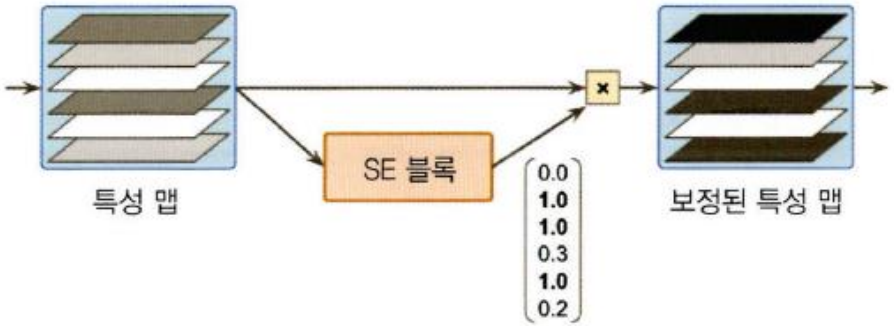
CNN Architectures (case study)

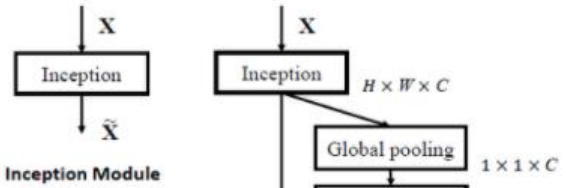
SENet

(squeeze-and-excitation network)

- 인셉션, ResNet 같은 기존 구조를 확장하여 성능향상
- SE block이라는 작은 신경망 추가

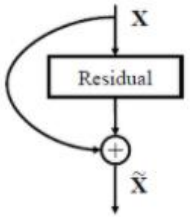
- Conv를 통해 추출된 각 채널 별 특성들을
  SE block에서 구한 채널당 중요도를 고려하여 재보정(recalibration)



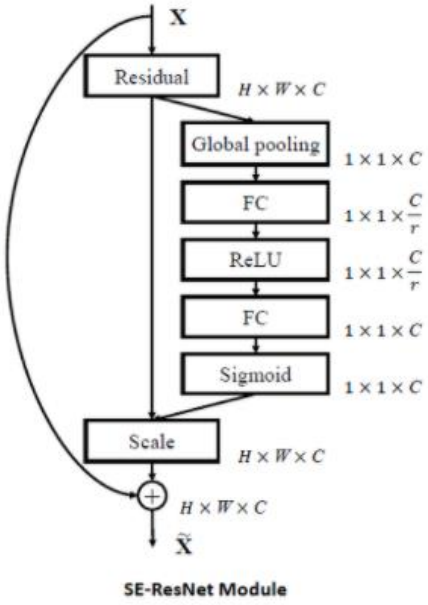


Inception Module

SE-Inception Module



ResNet Module



SE-ResNet Module

< Note >

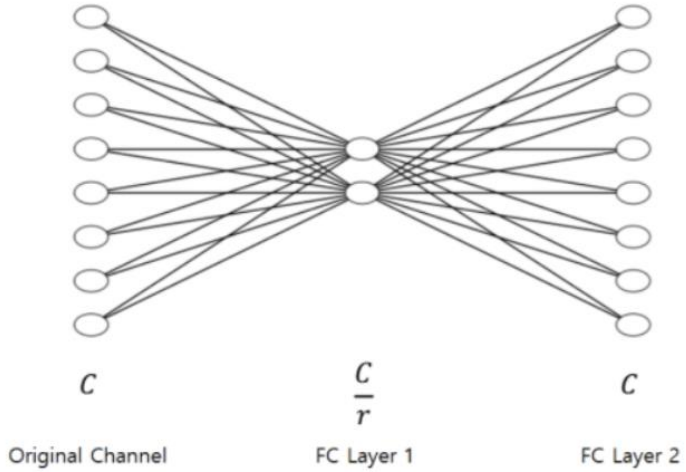
SE block은 어떤 특성이 일반적으로
동시에 가장 크게 활성화되는지를 학습한다.

CNN Architectures (case study)

SENet

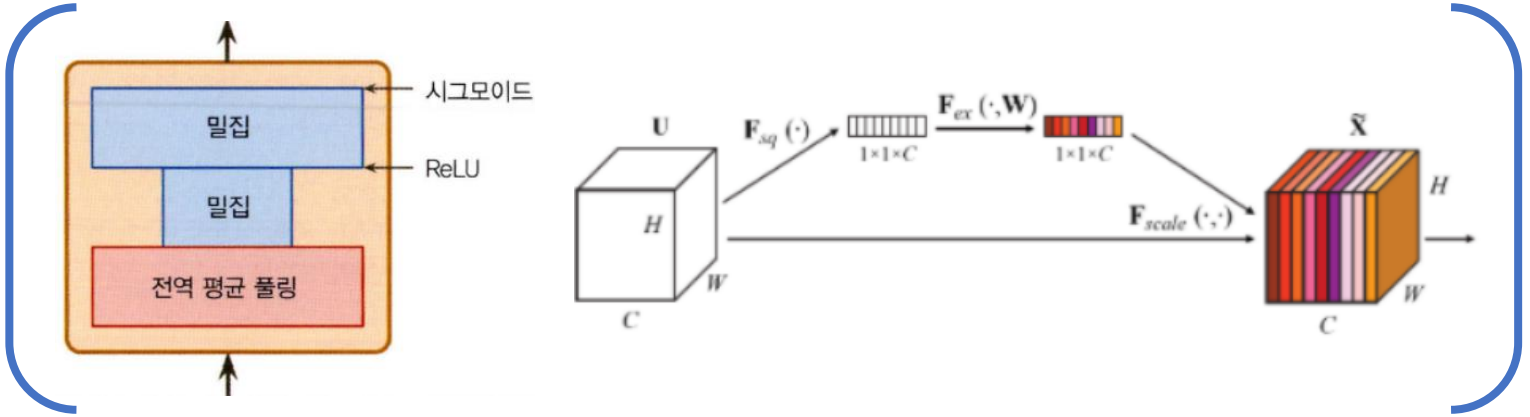
(squeeze-and-excitation network)

- Squeeze 과정 : Global Avg Pooling 등 무난한 압축함수 적용
- Excitation 과정 : 병목층을 거쳐 특성 조합에 대한
  일반적인 representation 학습
  (은닉층 뉴런 개수를 입력층보다 매우 작게 설정)



r : reduction ratio

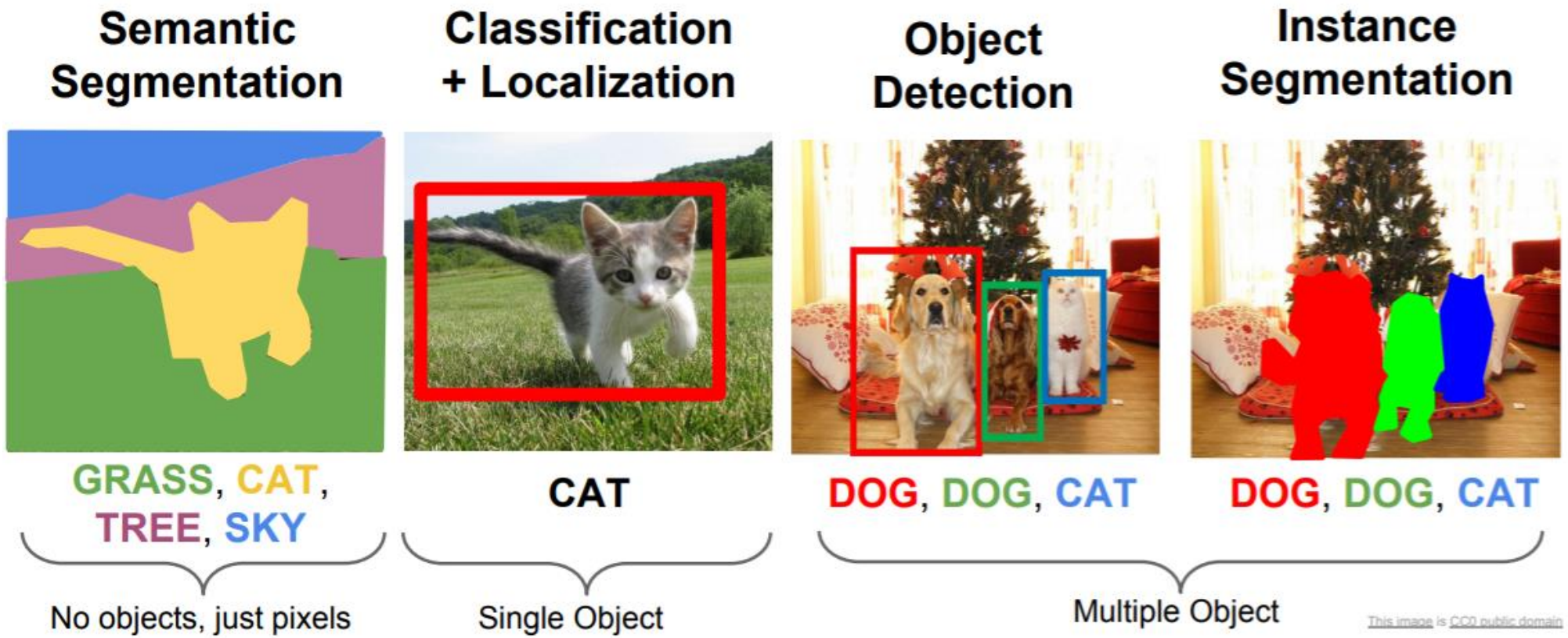
< SE block >



Contents

Classification & Localization

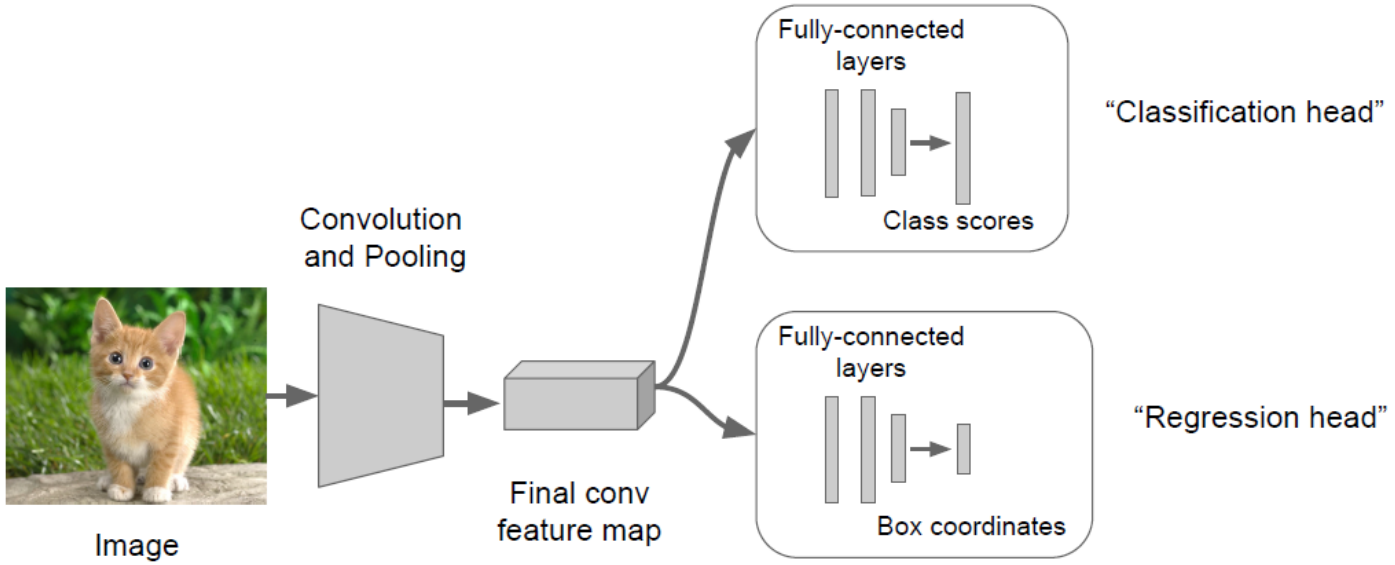
Computer Vision Tasks



Classification & Localization

Localization as regression

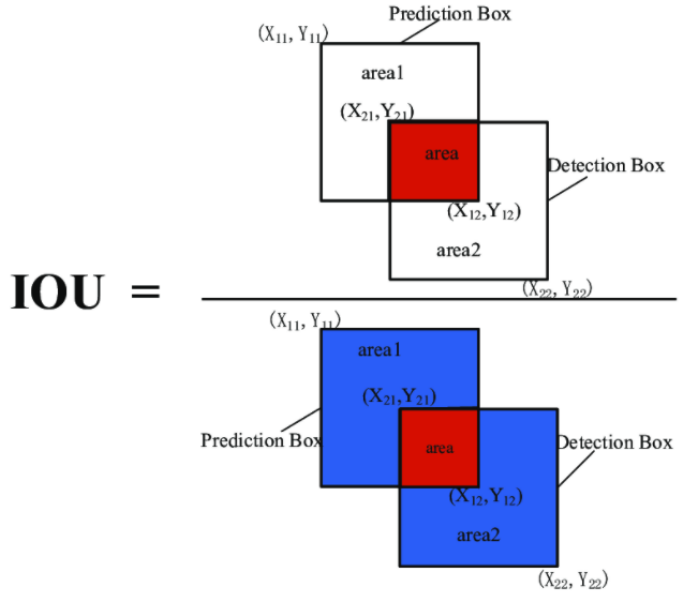
- Object의 위치를 추정하는 것
  = Bounding box의 좌표(x,y)와 높이(h),너비(w)를 예측하는 것
  = 4개의 output을 주는 다중회귀 문제



1. Train classification model (AlexNet, VGG, GoogLeNet, etc…)
2. Attach new fully-connected "regression head" to the network
3. Train the regression hear only with SGD, L2 loss
4. At test time, use both heads

< Intersection over Union >

$$IOU = \frac{\text{area}}{\text{area1} + \text{area2} - \text{area}}$$

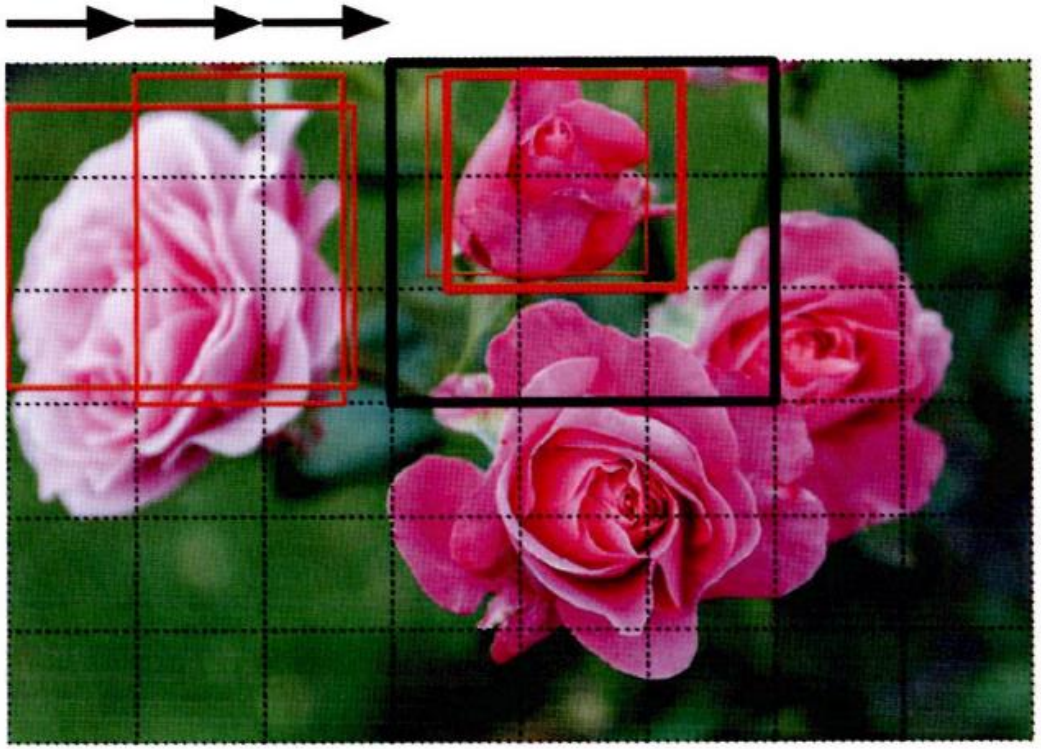


Contents

Object Detection

- 하나의 이미지에 여러 개의 Object가 있어 그것들의 class를 분류하고 위치를 추정해야 하는 task를 객체 탐지라고 함.

Sliding window method



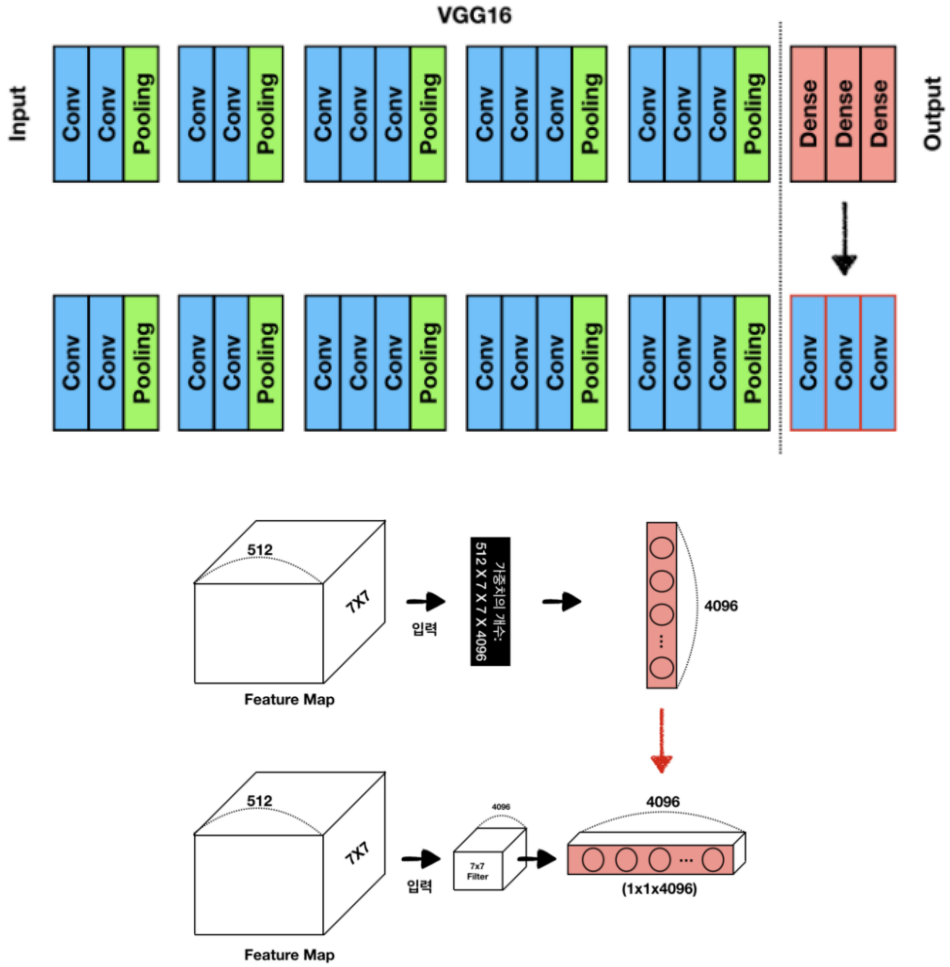
< non-max suppression, NMS >

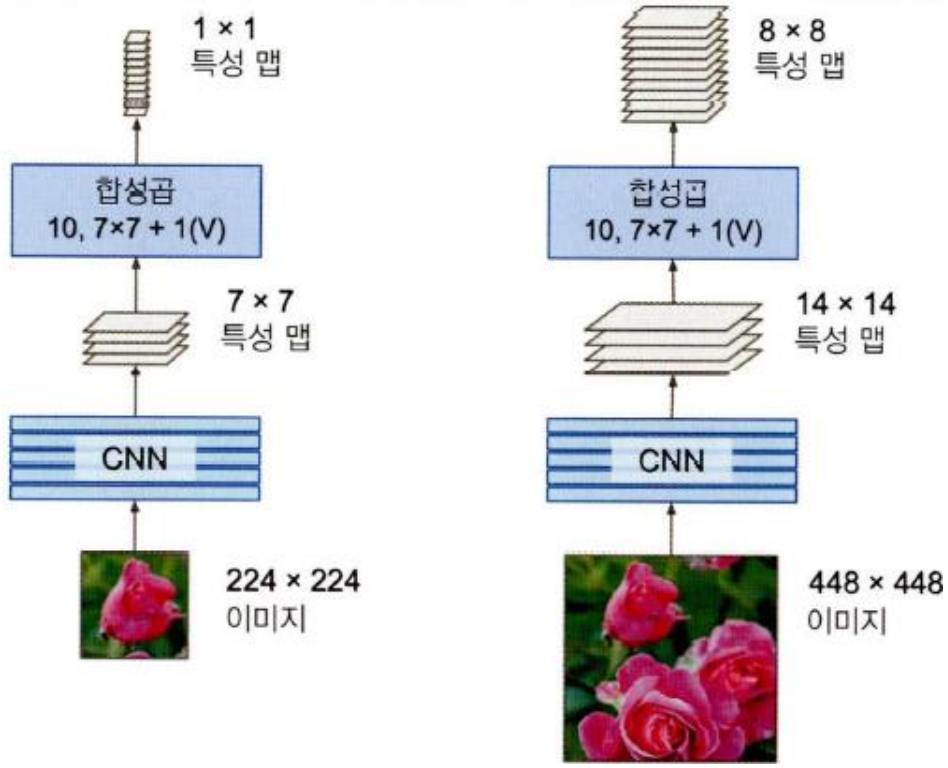
간단하지만 cost 큼 (각 window에 대해 cnn을 여러 번 실행해야 함.)

Object Detection

Fully Convolutional Nework (FCN)

기존 cnn의 최상위층 FC layer들을 conv layer로 바꾸자 !





- 파라미터 공유를 통한 계산량 감소
- 입력 이미지 크기가 자유로워짐

Object Detection

YOLO (you only look once)

- Detection as regression
- 입력 이미지를 그리드로 나누고, 각 그리드별로 bbox 찾기 + 분류 수행.
  (각 셀에서 B개의 bounding box와 box에 대한 confidence score,
  conditional class probabilities를 예측)
- NMS를 통해 최종 bbox와 class label 추출

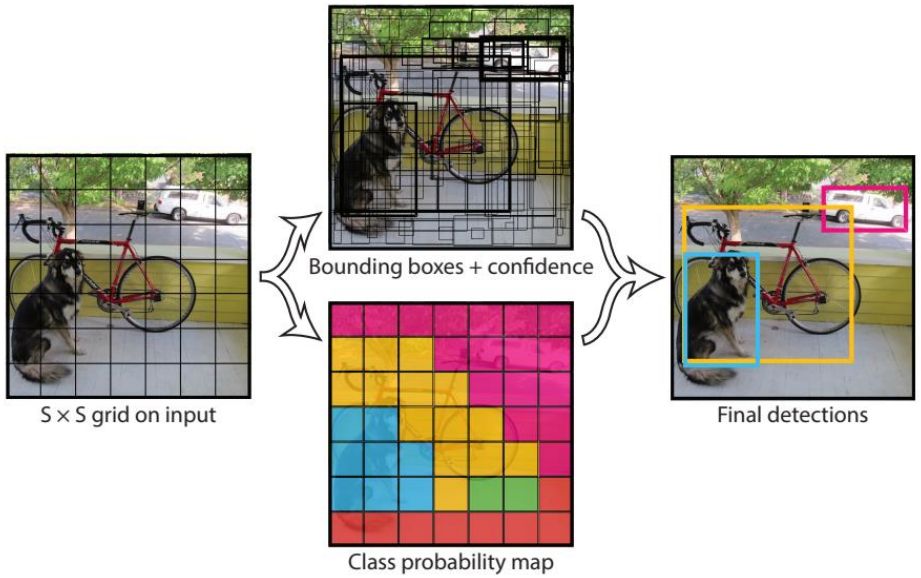
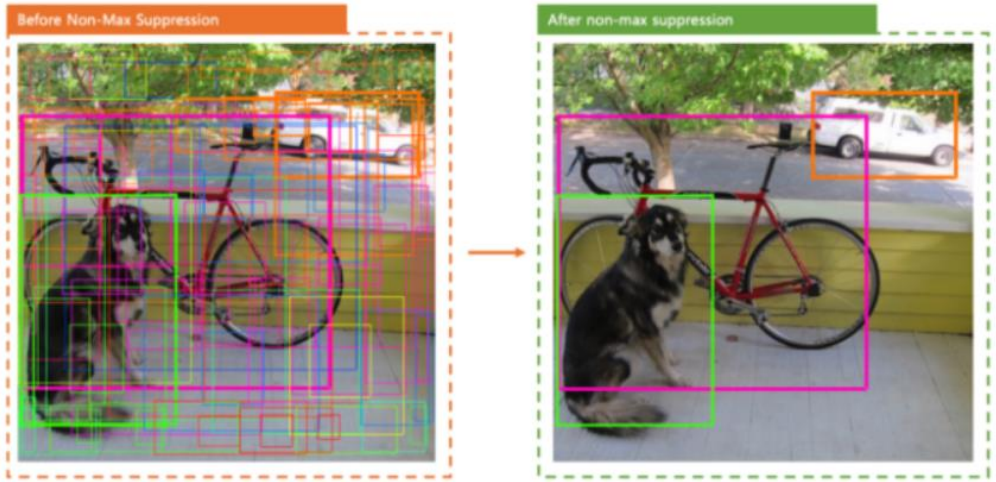


Figure 2: The Model. Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts B bounding boxes, confidence for those boxes, and C class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

$$\Pr(Object) * IOU_{pred}^{truth}$$

$$C(conditional\ class\ probabilities) = \Pr(Class_i|Object)$$



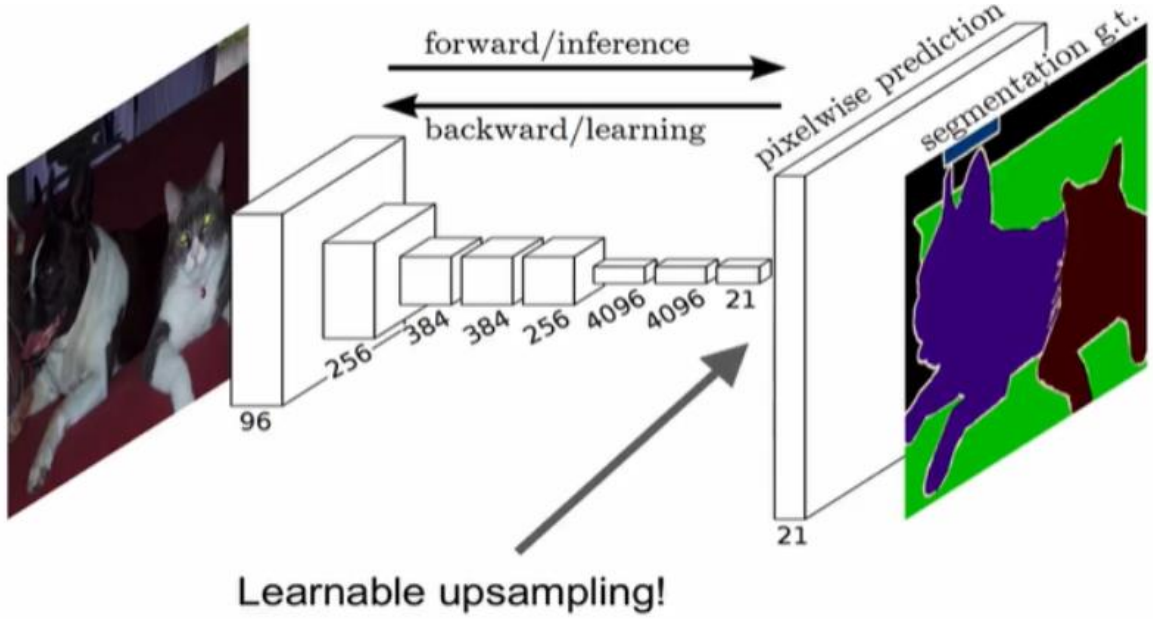
$$class\ specific\ confidence\ score$$
$$= \Pr(Class_i|Object) * \Pr(Object) * IOU_{pred}^{truth}$$
$$= \Pr(Class_i) * IOU_{pred}^{truth}$$

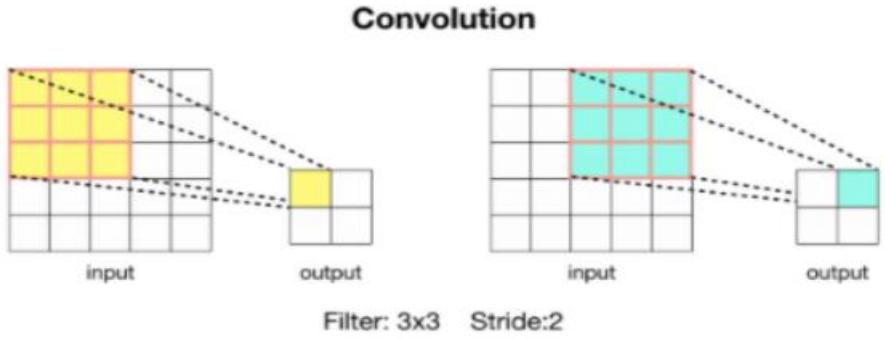
Contents

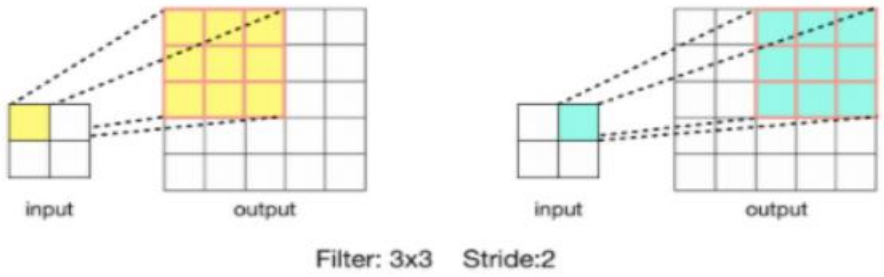
Semantic Segmentation

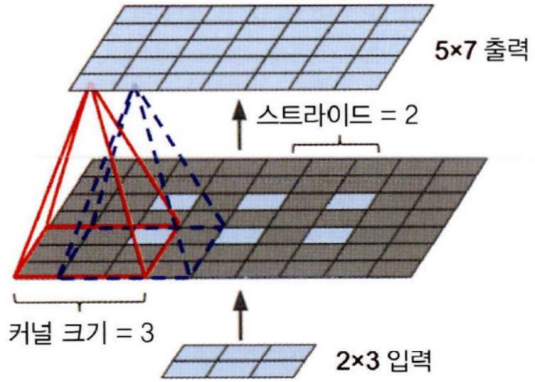
FCN & upsampling

FCN으로 위치정보 보존! 근데 해상도는..?



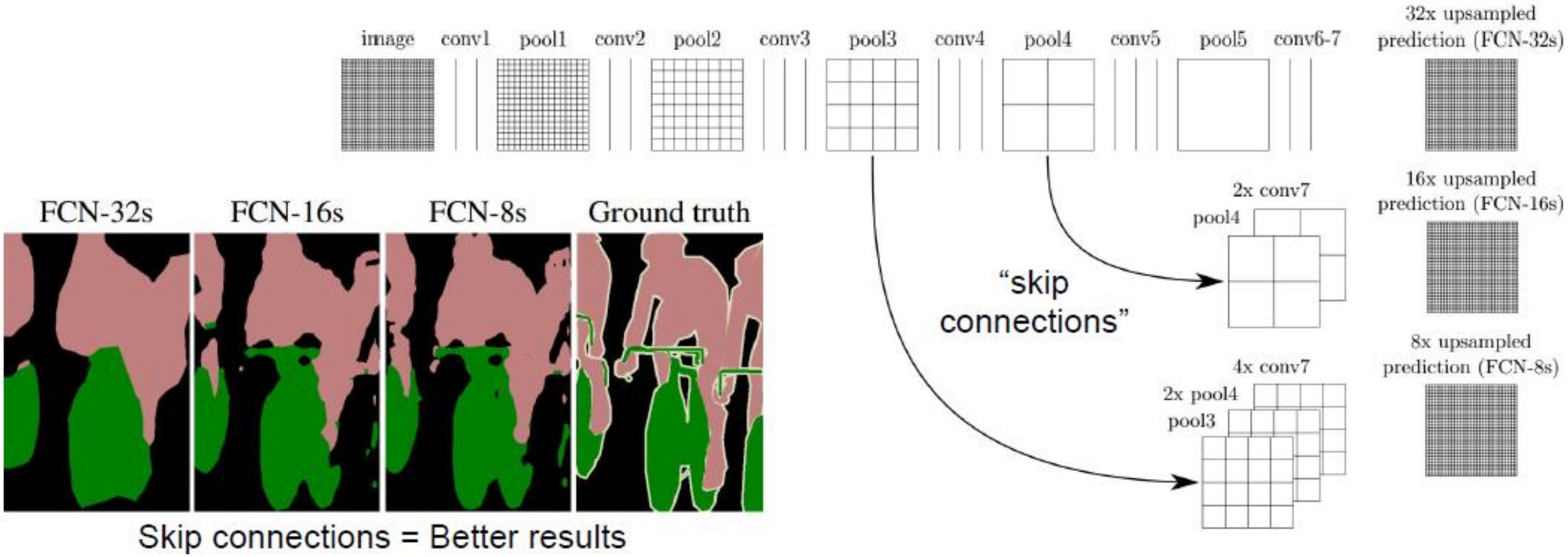






Semantic Segmentation

Skip connection



Reference