



합성곱 신경망

김예진 박보영 박지운

목차

#01 객체 인식을 위한 신경망

- R-CNN
- 공간 피라미드 풀링
- Fast R-CNN
- Faster R-CNN

#02 이미지 분할을 위한 신경망

- 완전 합성곱 네트워크
- 합성곱 & 역합성곱 네트워크
- U-Net
- PSPNet
- DeepLabv3/DeepLabv3+



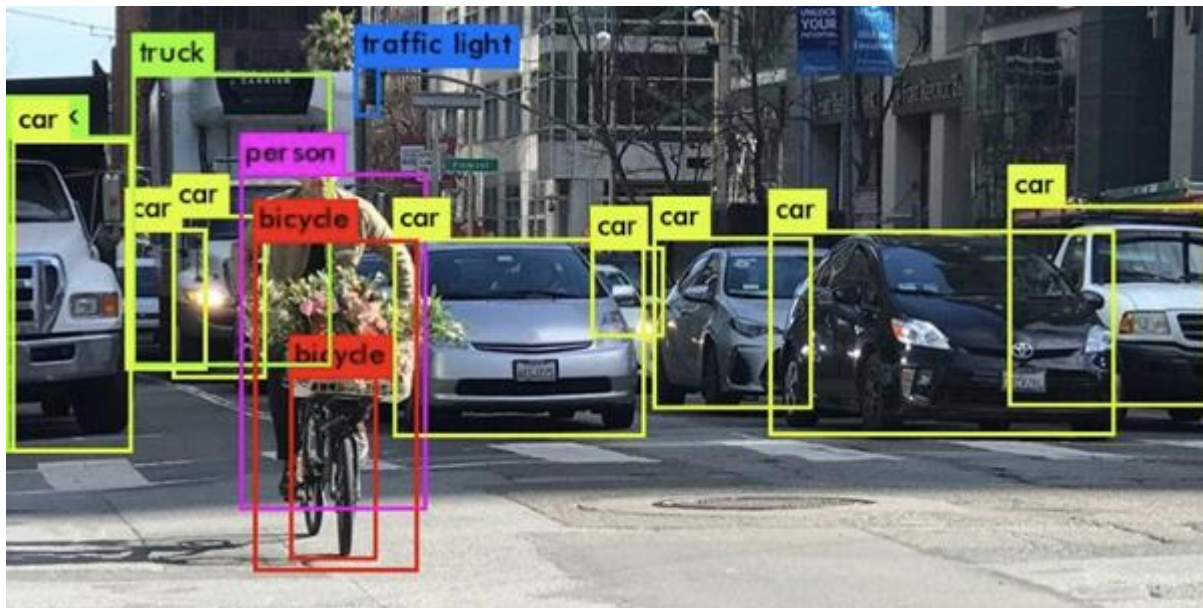
1. R-CNN



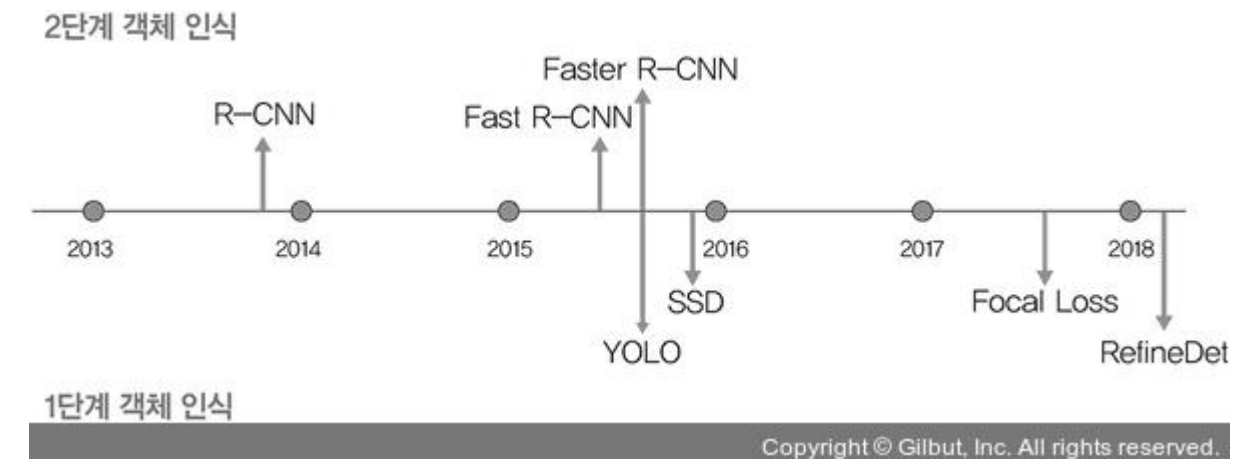
#1-1 객체 인식이란?

객체 인식: 이미지나 영상 내에 있는 객체를 식별하는 컴퓨터 비전 기술

-> 객체 인식 = 여러 가지 객체에 대한 분류 + 객체의 위치 정보를 파악하는 위치 검출



- 1단계 객체 인식: 분류와 위치 검출을 동시에 행하는 방법
-> 비교적 빠르나 정확도가 낮음
- 2단계 객체 인식: 분류와 위치 검출을 순차적으로 행하는 방법
-> 비교적 느리나 정확도가 높음

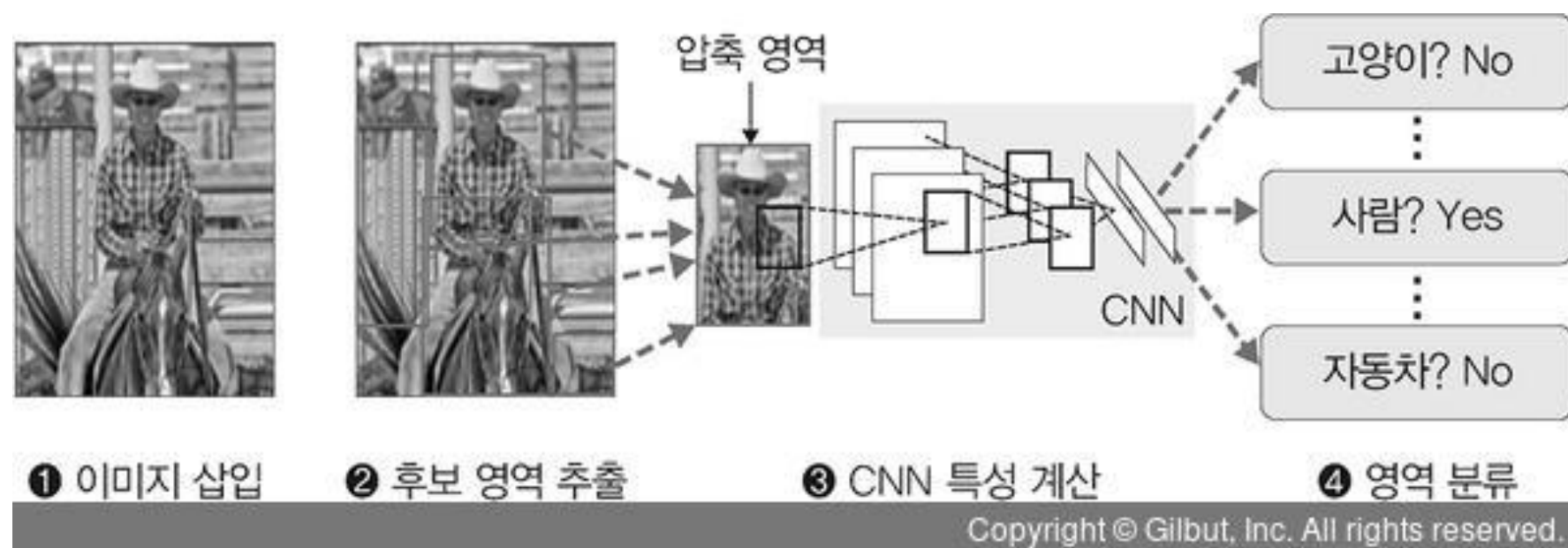


#1-2 R-CNN

R-CNN 이전 객체 인식 알고리즘 : 일정한 크기를 가진 윈도우를 통해 이미지의 모든 영역을 탐색하며 객체를 검출해냄(비효율적)

-> 선택적 탐색 알고리즘을 적용한 후보 영역(객체가 있을 법한 영역) 사용

R-CNN은 CNN(이미지 분류 수행)과 객체가 있을 법한 영역을 제안해주는 후보 영역 알고리즘을 결합한 알고리즘



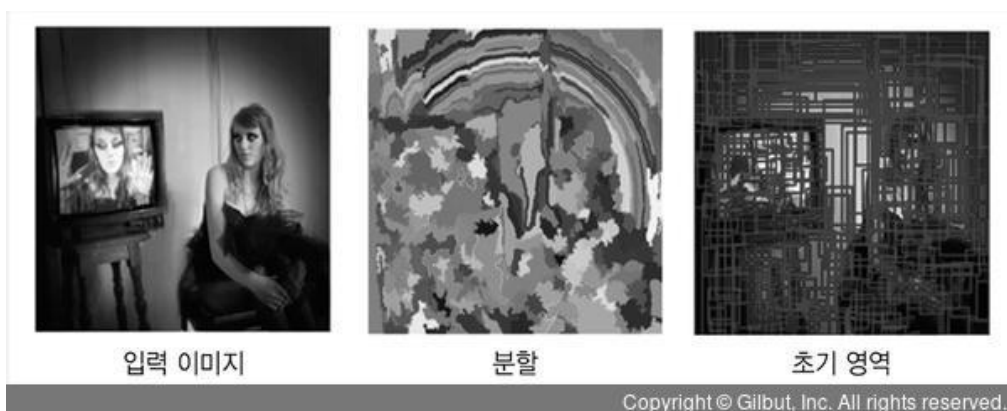
1. 이미지를 입력으로 받는다
2. 2000개의 바운딩 박스를 선택적 탐색 알고리즘으로 추출한 후 잘라내고, CNN 모델에 넣기 위한 크기로 통일(227x227)
3. 크기가 동일한 이미지 2000개에 각각 CNN 모델 적용
4. 각각 분류를 진행하여 결과 도출

1. 세 단계의 복잡한 학습 과정, 2. 긴 학습 시간과 대용량의 저장 공간, 3. 객체 검출의 속도 등의 단점 존재

#1-3 선택적 탐색

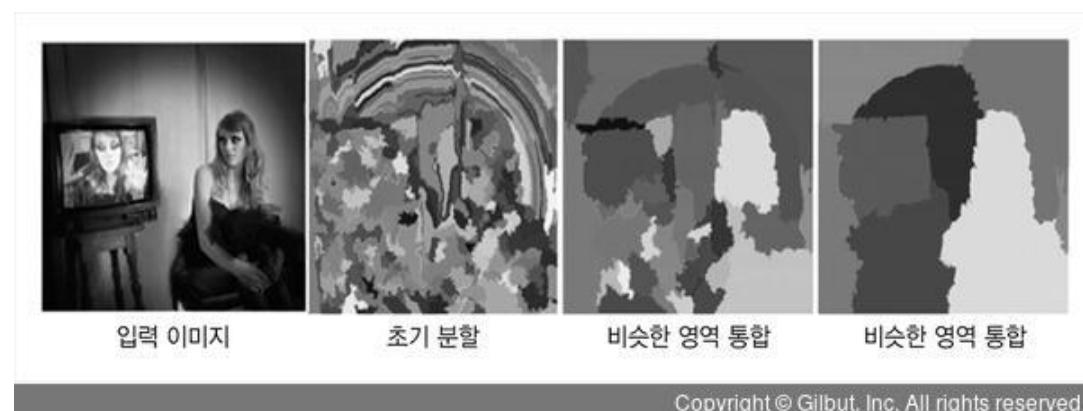
객체 인식이나 검출을 위한 가능한 후보 영역을 알아내는 방법
-> 분할 방식을 이용하여 시드(seed)를 선정하고, 그 시드에 대한 완전 탐색 적용

1단계. 초기 영역 생성



각각의 객체가 영역 한 개에 할당 될 수 있도록 많은 초기 영역 생성(입력된 이미지를 영역 여러 개로 분할함)

2단계. 작은 영역의 통합



앞서 영역 여러 개로 나눈 것들을 비슷한 영역으로 통합
-> 탐욕(greedy) 알고리즘을 사용하여, 비슷한 영역이 하나로 통합될 때까지 반복

3단계. 후보 영역 생성



앞서 통합된 이미지들을 기반으로 후보 영역(바운딩 박스) 추출

#1-4 공간 피라미드 풀링

기존 CNN 구조 : 완전연결층을 위해 입력 이미지 고정 (이미지를 고정된 크기로 자르거나 비율 조정)

-> 물체의 일부분이 잘리거나, 본래의 생김새와 달라지는 문제점 有

=> 공간 피라미드 풀링 도입!

R-CNN



공간 피라미드 풀링



=> 입력 이미지연결의 크기와 관계 없이 합성곱층을 통과시키고, 완전 연결층에 전달되기 전에 특성맵을 동일한 크기로 조절해주는 풀링층 적용

2. R-CNN 개선

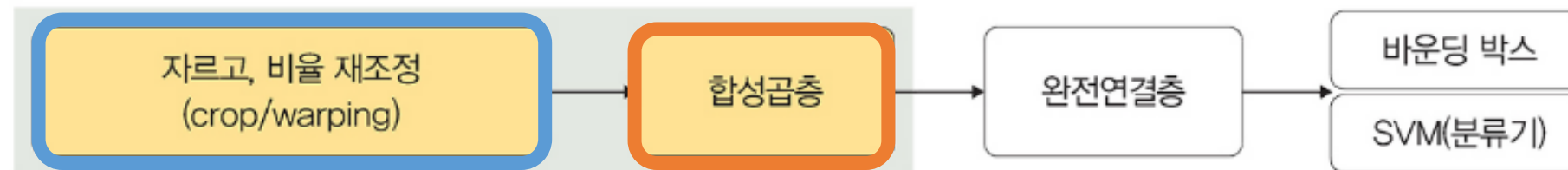


2.1 Fast R-CNN

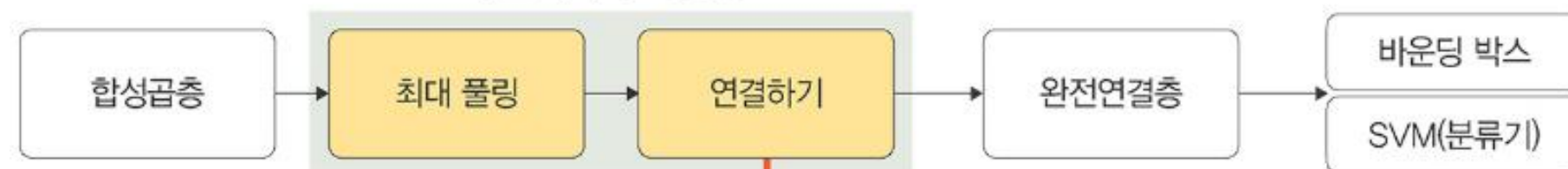
📌 Fast R-CNN

- R-CNN의 문제점 : 바운딩 박스마다 CNN을 돌림. 분류에 긴 학습 시간 필요.
- 속도 문제 개선을 위해 **RoI 풀링** 도입!!
- 바운딩 박스 정보 유지하면서 CNN 먼저 통과 → 최종 CNN 특성 맵에 풀링 적용하여 크기 조정

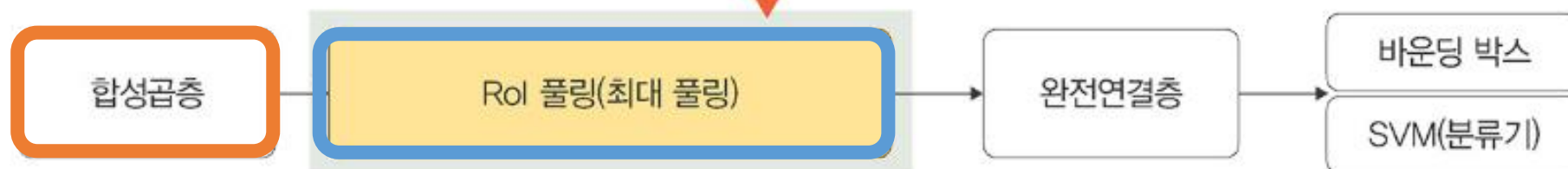
R-CNN



공간 피라미드 풀링



Fast R-CNN

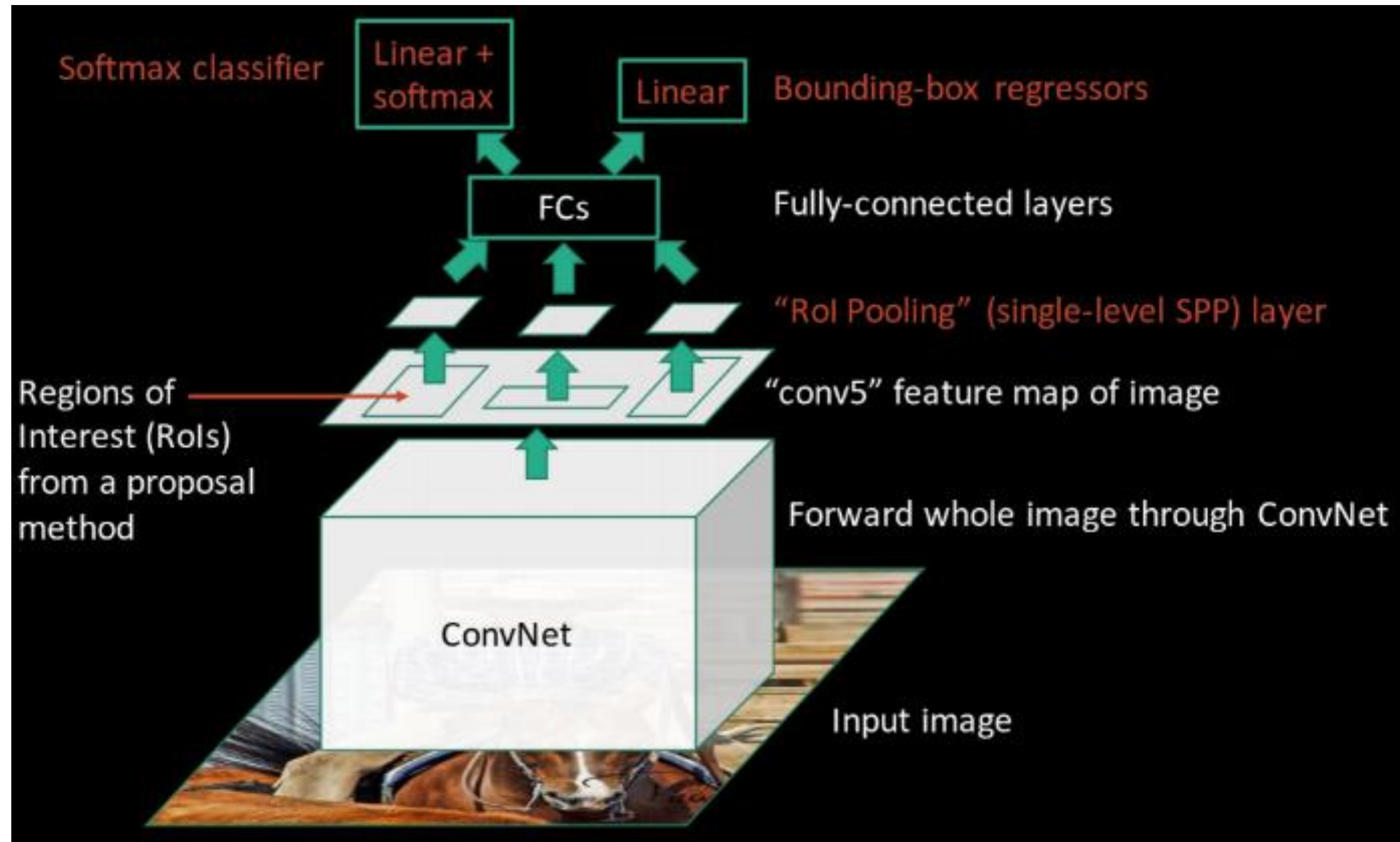


합성곱층 통과

맵 크기 통일

→ R-CNN에서 **순서 바꾼 것**

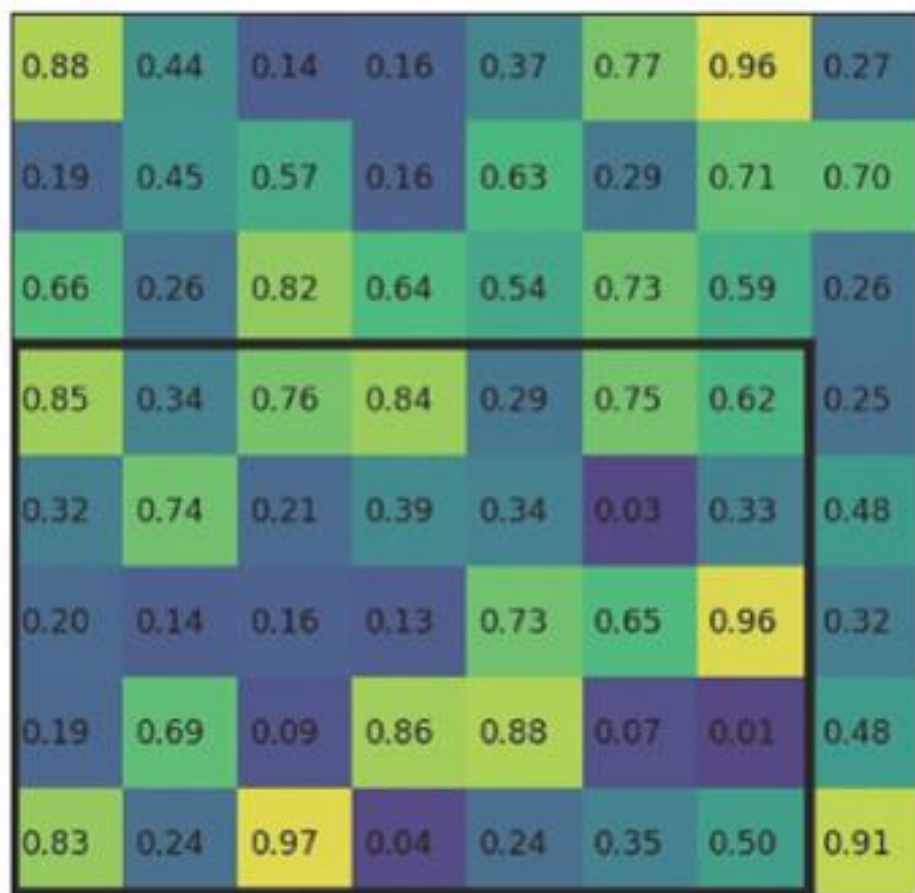
2.1 Fast R-CNN



2.1 Fast R-CNN

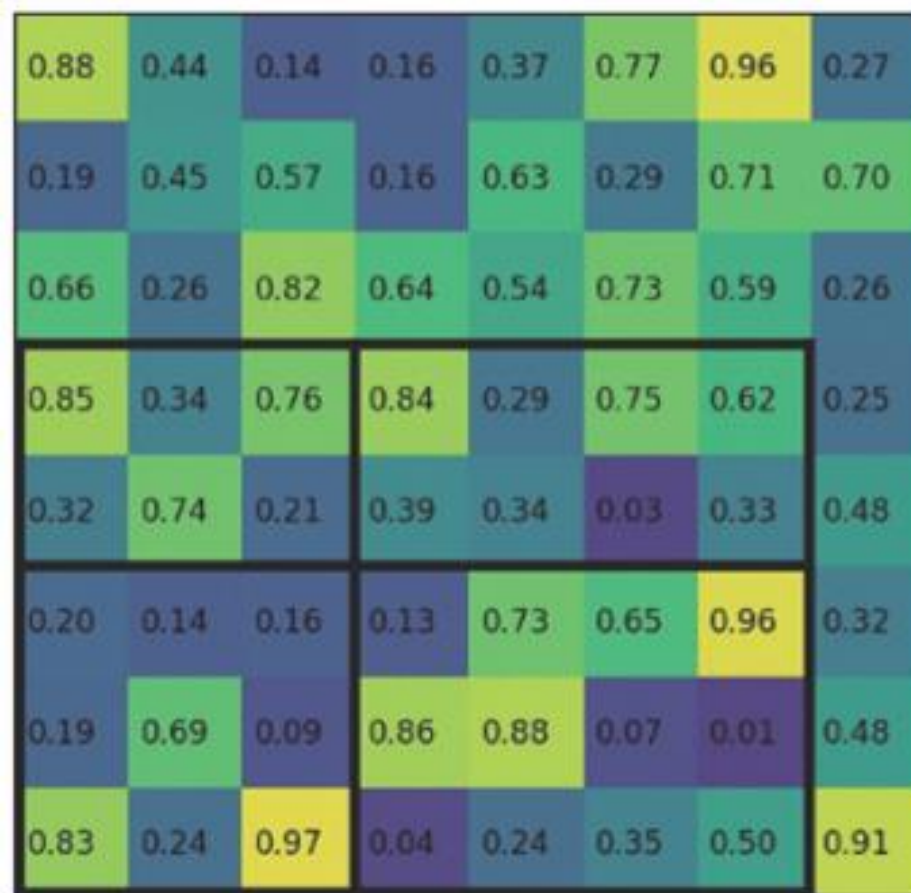
📌 RoI 풀링

- 크기가 다른 특성 맵 → 각 영역에 스트라이드를 다르게 하여 최대 풀링 적용 → 크기 통일



바운딩 박스 크기 : 7*5

통일하고자 하는 크기 : 2*2



$7/2=3, 5/2=2$

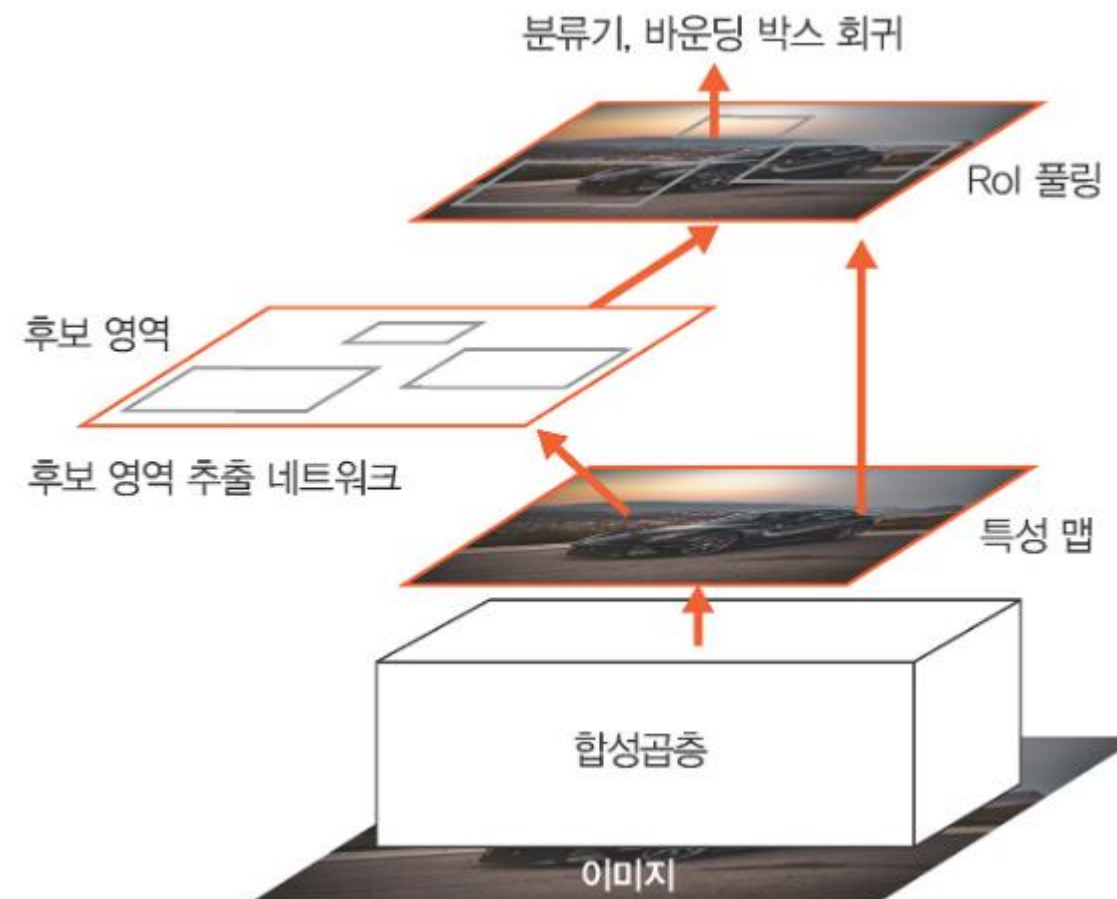
스트라이드 적용

| | |
|------|------|
| 0.85 | 0.84 |
| 0.97 | 0.96 |

2.2 Faster R-CNN

📌 Faster R-CNN

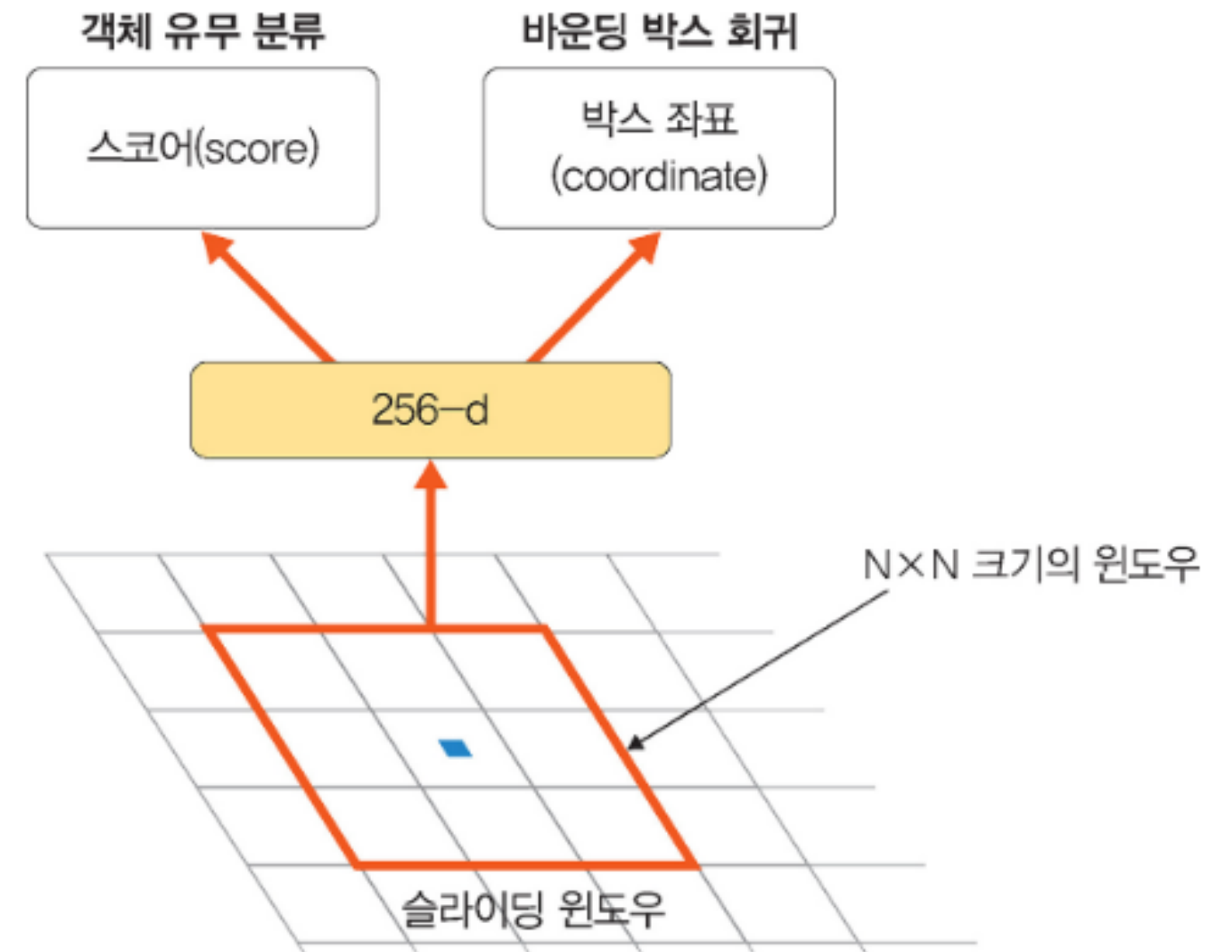
- Fast R-CNN의 문제점 : 후보 영역 생성을 외부에서 진행하여 속도 느림 (CPU 사용)
- 이를 해결하기 위해 **RPN** 도입!!
- RPN (Region Proposal Network) : **후보 영역 추출 네트워크** → 내부에서 빠르게 진행 (GPU 사용)
- 마지막 합성곱층 다음에 위치



2.2 Faster R-CNN

📌 후보 영역 추출 네트워크 (RPN)

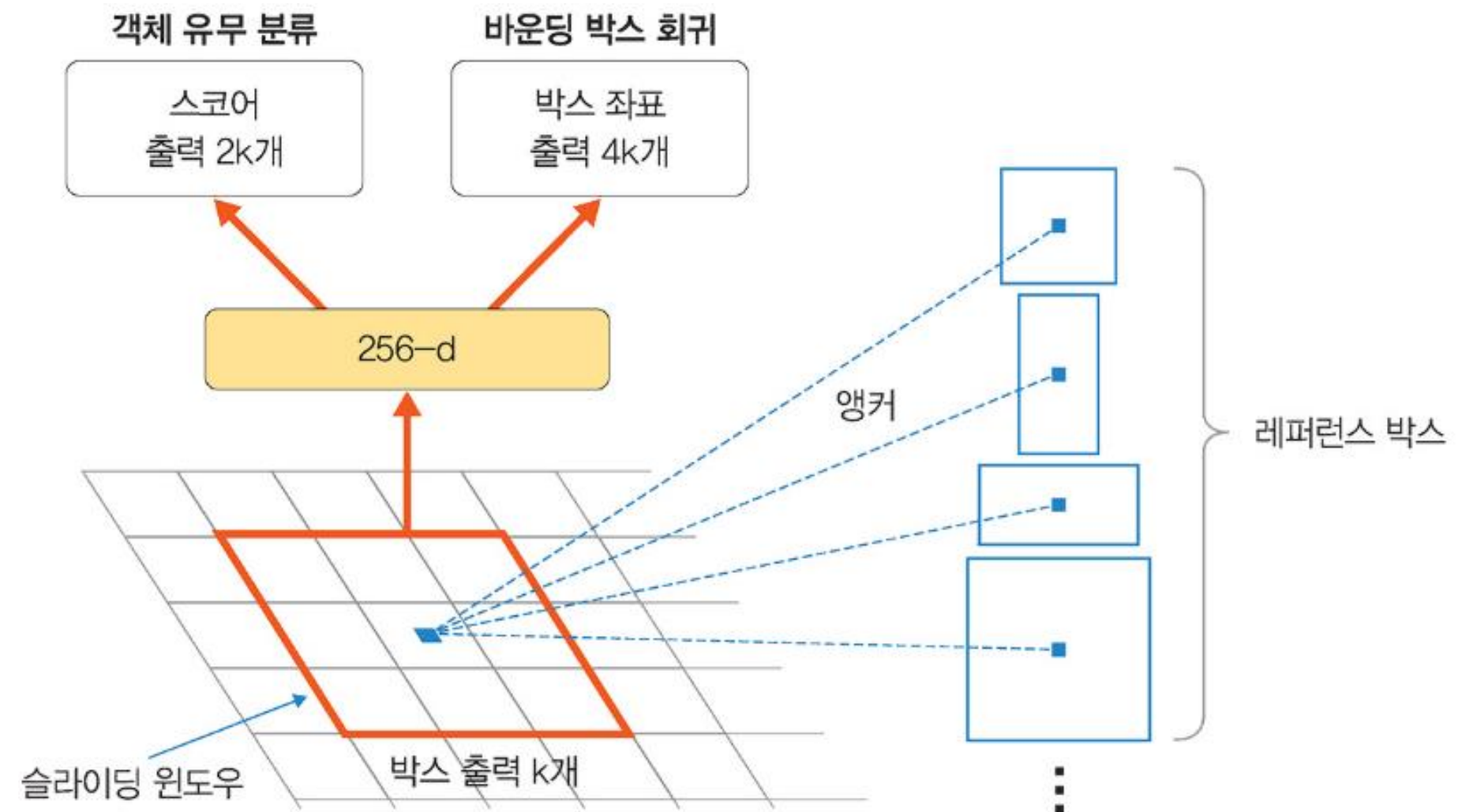
- $N \times N$ 크기의 작은 윈도우 영역 → 슬라이딩 윈도우 방식으로 객체 탐색
- Score : 객체 유무 분류 → object / not object
- Coordinate : 바운딩 박스 회귀 → 박스 위치 보정 (좌표점 추론)



2.2 Faster R-CNN

📌 후보 영역 추출 네트워크 (RPN)

- 이미지에 존재하는 객체의 크기와 비율 다양 → 고정된 $N \times N$ 크기로 수용하기 어려움
- **Anchor** : 여러 크기와 비율의 레퍼런스 박스 k 개 미리 정의 → 각 슬라이딩 윈도우 위치에서 박스 k 개 출력
- 객체 / 배경 → $2k$ 개의 분류 출력
- X, y, w, h 위치 보정 → $4k$ 개의 회귀 출력



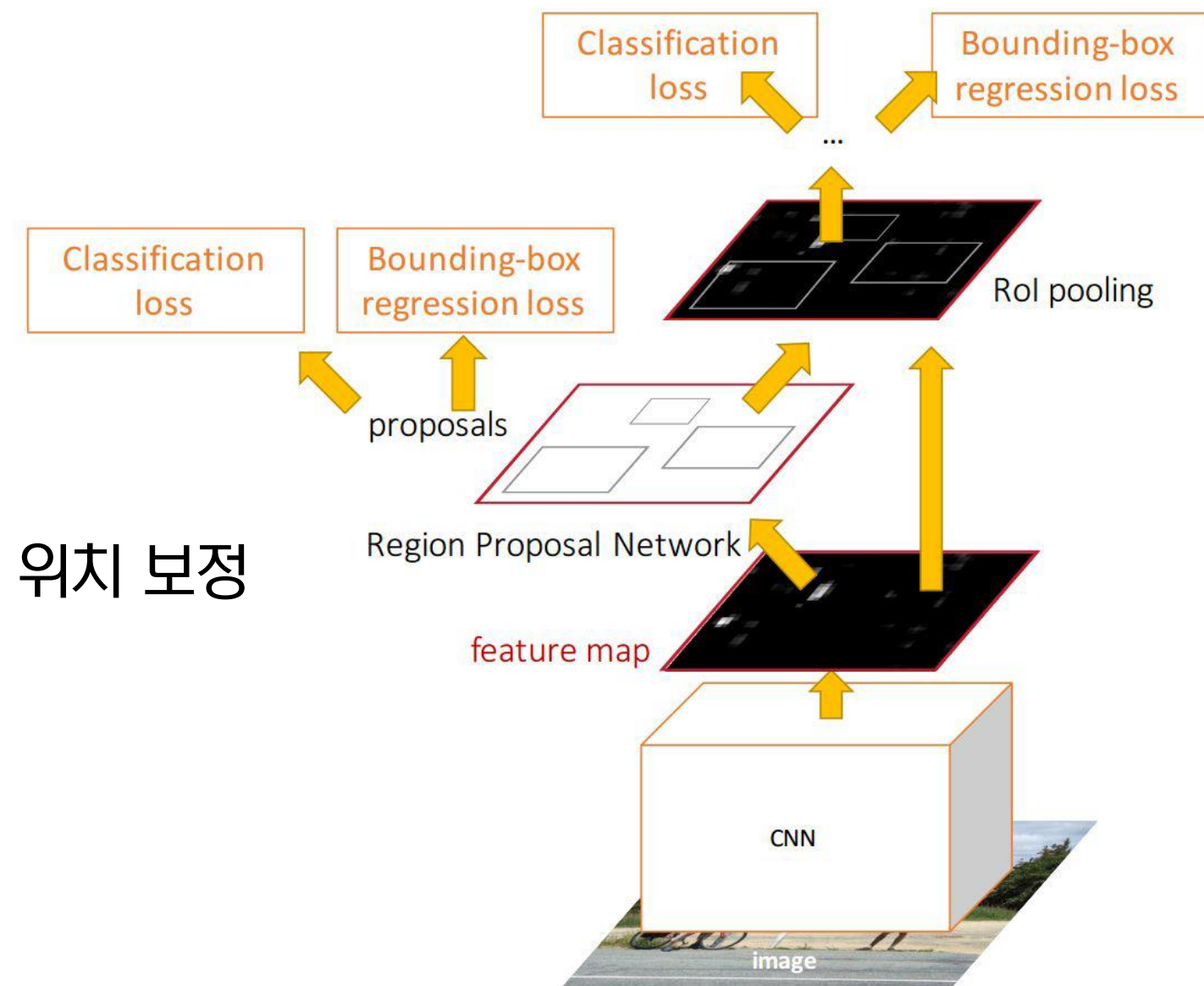
2.2 Faster R-CNN

분류 : 해당 object가 어떤 class인지

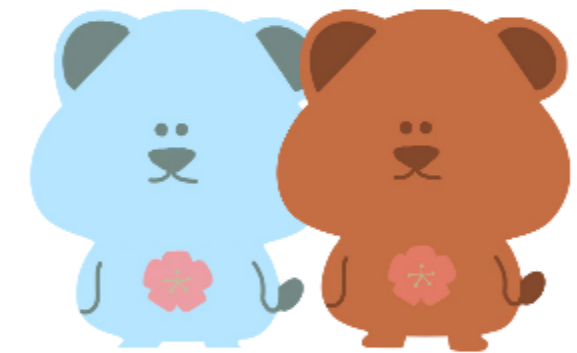
회귀 : 박스 위치 보정

분류 : object / not object

회귀 : anchor 활용하여 박스 위치 보정



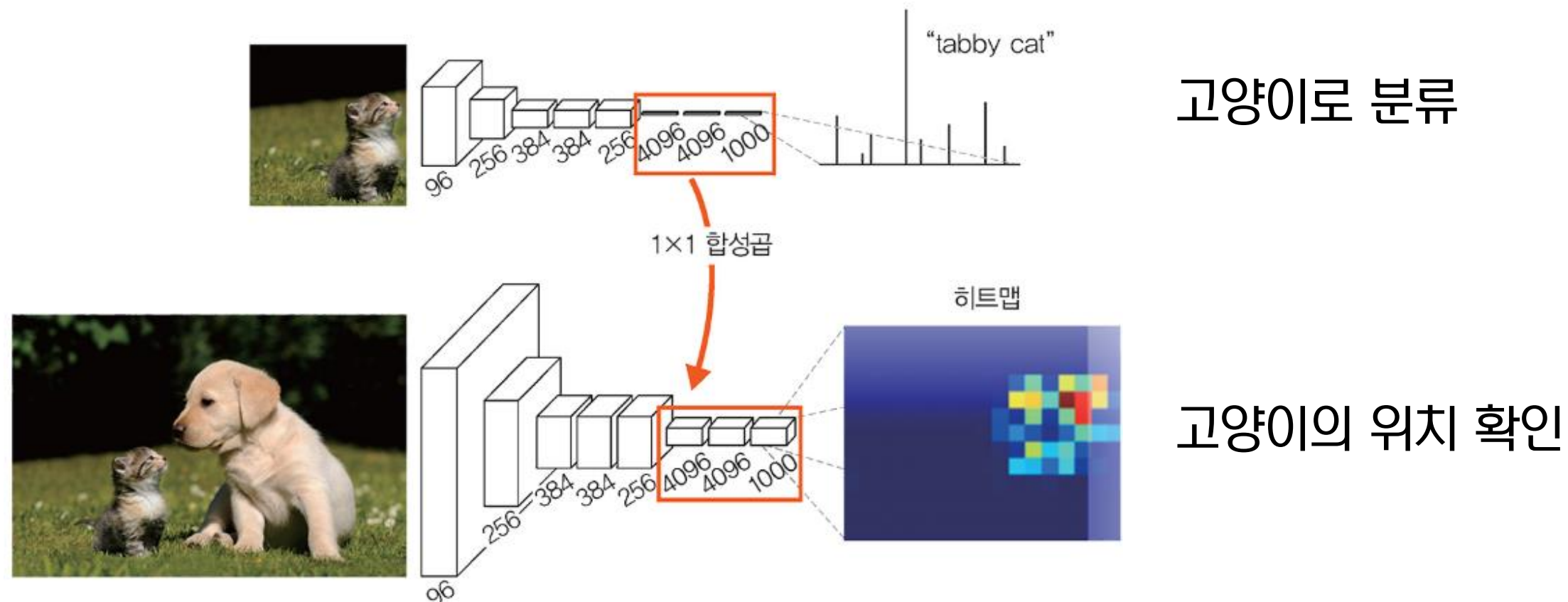
3. 이미지 분할을 위한 신경망



3.1 완전 합성곱 네트워크

📌 완전 합성곱 네트워크

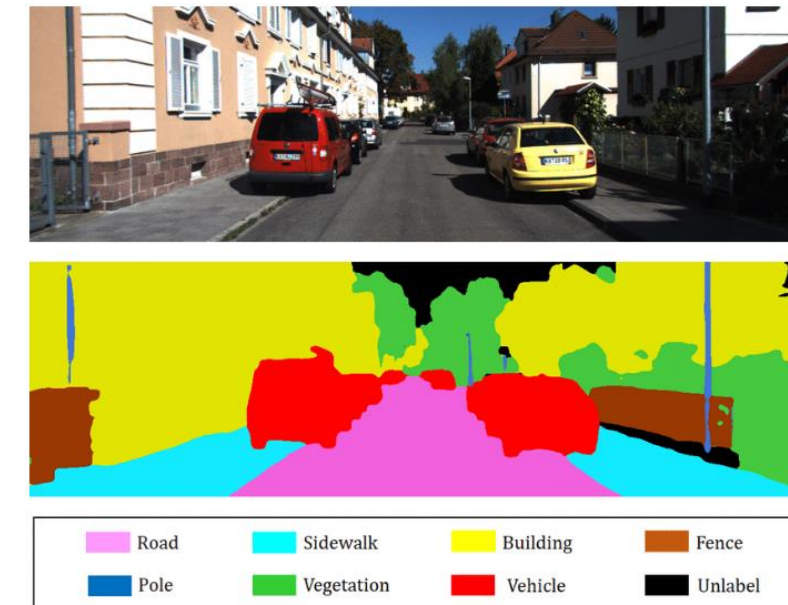
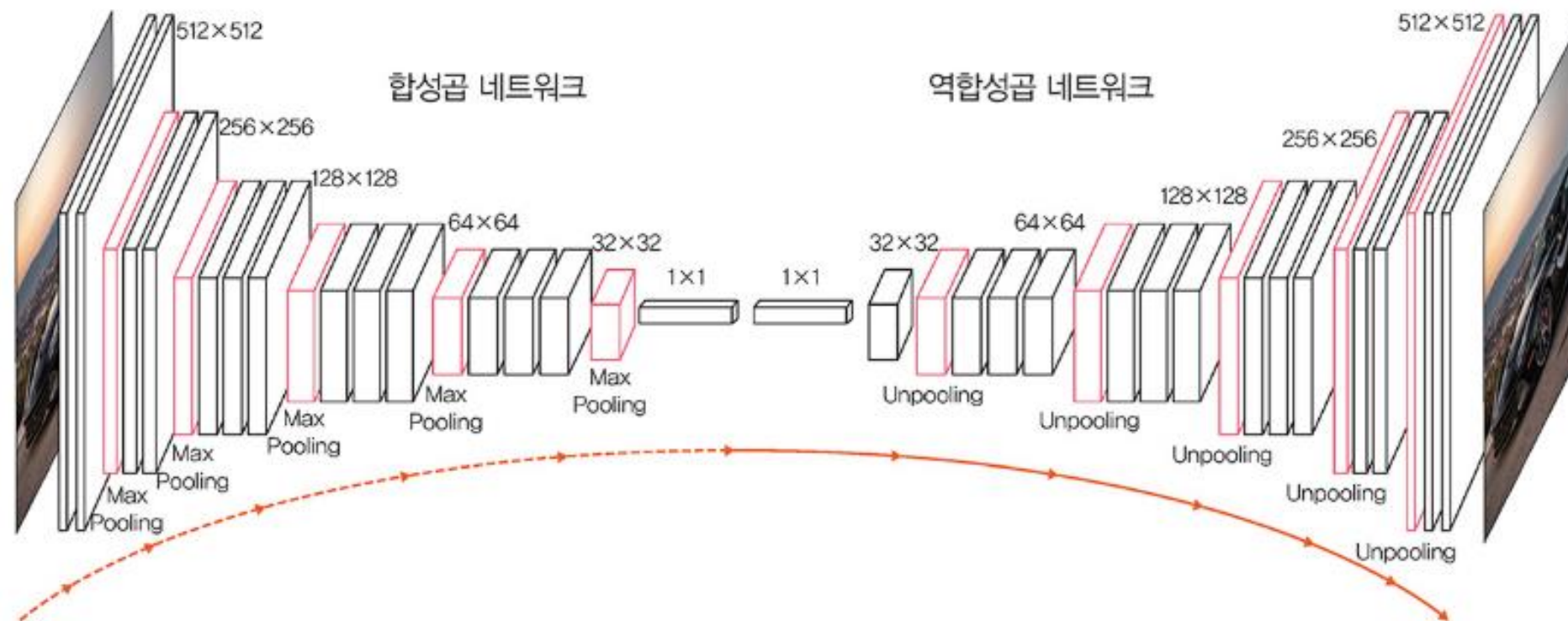
- 완전연결층의 문제점 : 고정된 크기의 입력만 가능 / 완전 연결층을 거친 후에 위치 정보가 사라짐
- 이를 해결하기 위해 완전연결층을 **1*1 합성곱**으로 대체!!
- 이미지 분류 CNN 기반 모델의 완전연결층을 1*1 합성곱으로 대체 → 이미지 분할
- 입력 이미지에 대한 크기 제약 사라짐 / 위치 정보 보존 가능



3.2 합성곱 & 역합성곱 네트워크

📌 합성곱 & 역합성곱 네트워크

- 완전 합성곱 네트워크의 문제점 : 여러 단계의 합성곱, 풀링층을 거치면서 해상도 낮아짐
- 이를 해결하기 위해 **역합성곱** 네트워크 도입!!
- 합성곱 : 특성 맵 크기 줄임 ↔ 역합성곱 : 특성 맵 크기 증가
- CNN의 최종 출력 결과 → 원래 입력 이미지와 같은 크기로 만들고 싶을 때 사용

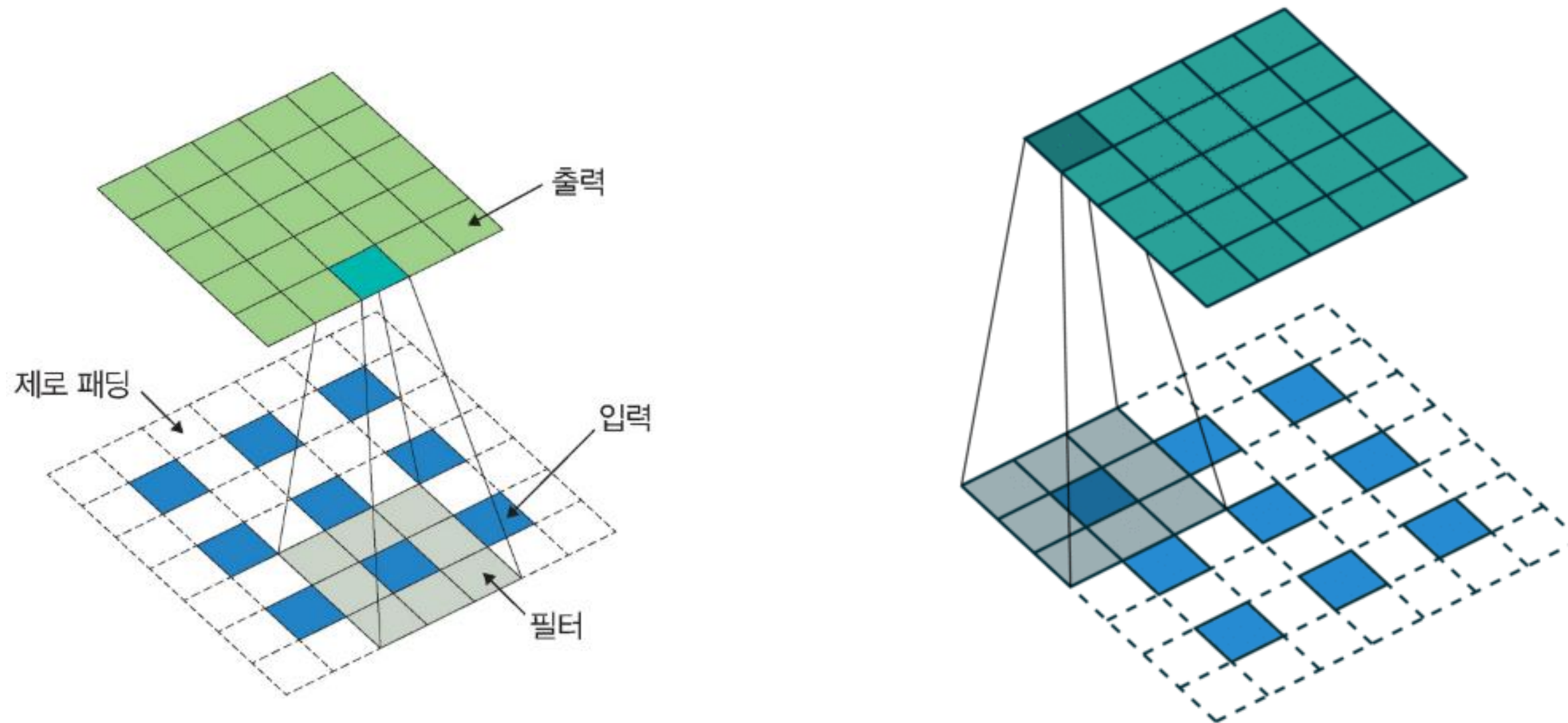


Semantic segmentation

3.2 합성곱 & 역합성곱 네트워크

📌 역합성곱 진행방식

- 1) 각 픽셀 주위에 제로 패딩 추가
- 2) 패딩된 것에 합성곱 연산 수행

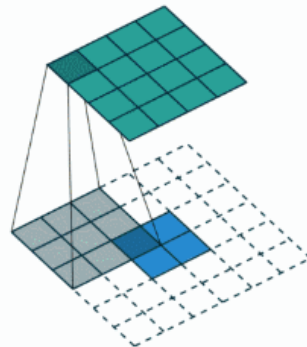
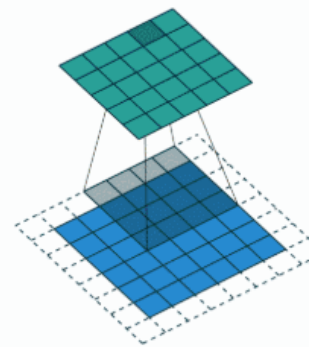
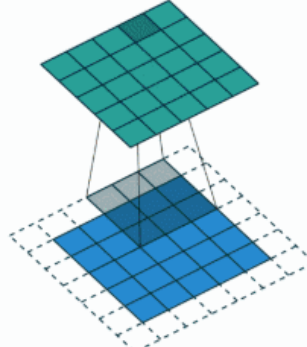
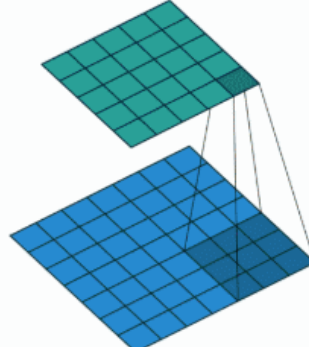
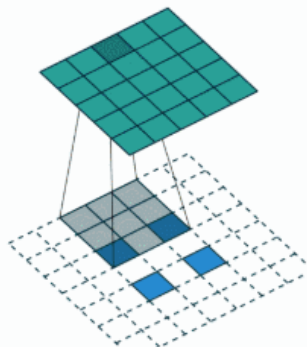
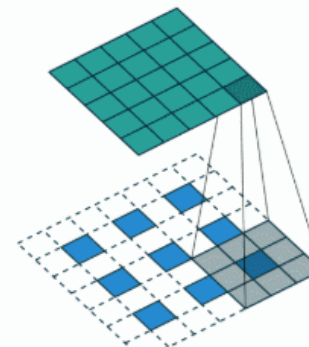
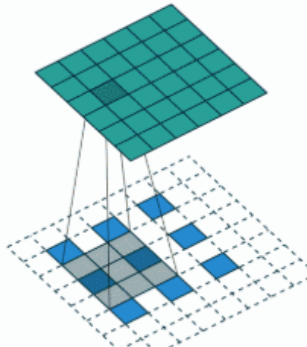


3.2 합성곱 & 역합성곱 네트워크

📌 역합성곱의 여러 이름들.. 종류들..

- Deconvolution
- Upsampling
- Convolution transpose
- Backward strided convolution
- $\frac{1}{2}$ strided convolution
- Upconvolution

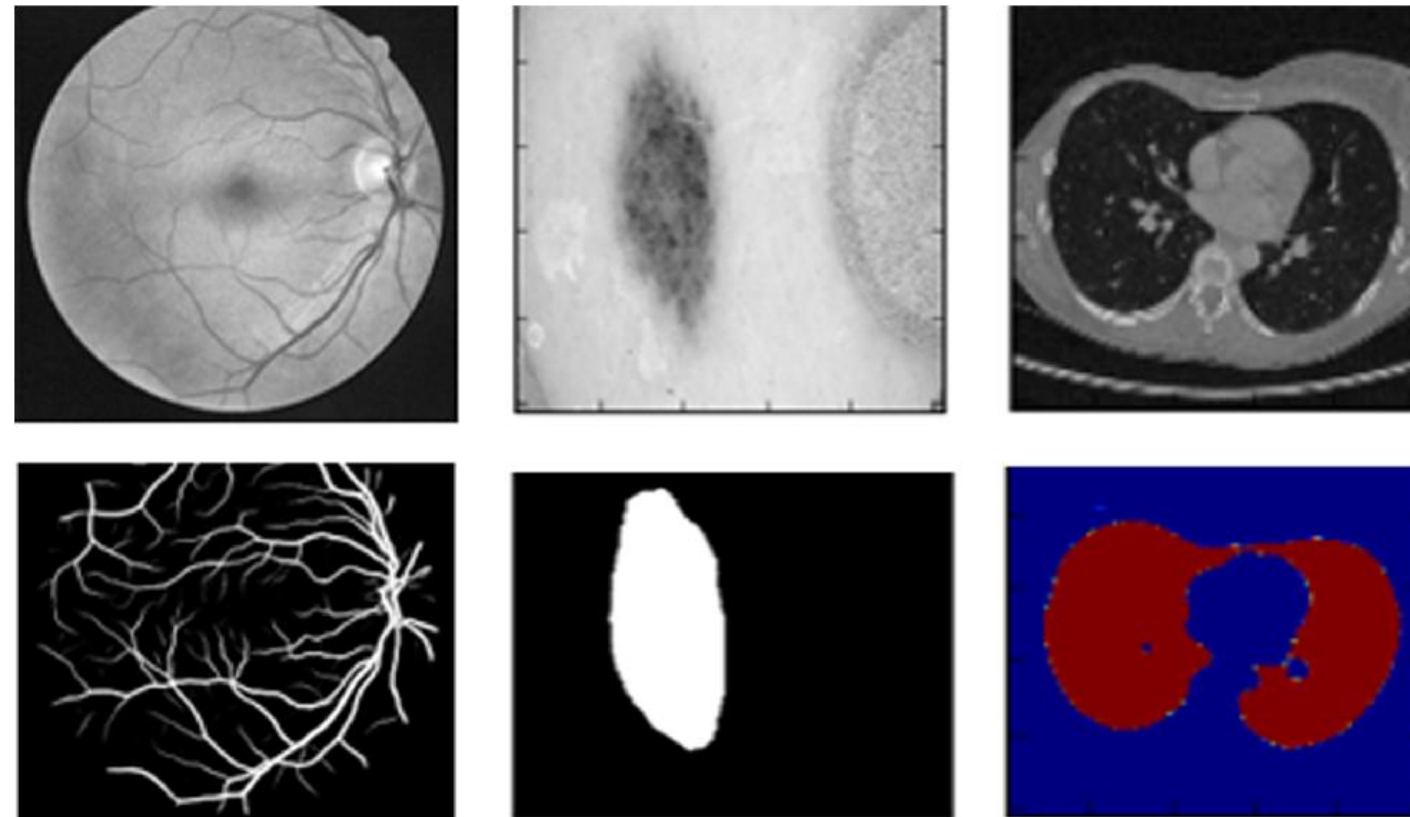
N.B.: Blue maps are inputs, and cyan maps are outputs.

| | | | |
|--|--|--|---|
|  |  |  |  |
| No padding, no strides, transposed | Arbitrary padding, no strides, transposed | Half padding, no strides, transposed | Full padding, no strides, transposed |
|  |  |  | |
| No padding, strides, transposed | Padding, strides, transposed | Padding, strides, transposed (odd) | |

3.1. U-Net

Q. 어디에 쓰이나요?

MRI, CT 상에서 병변을 진단하거나 장기, 세포 조직 등을 분할하는 등 의료 영상(Biomedical) 분야
메디컬 이미지 분할과 관련하여 항상 회자됨

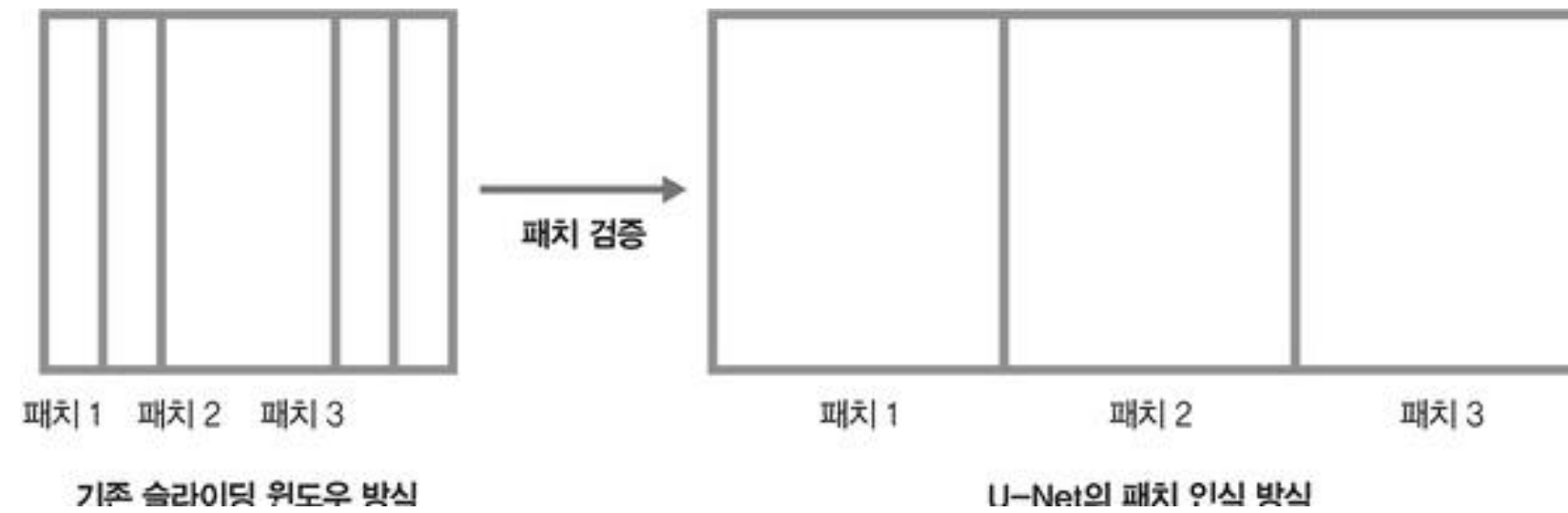


<https://medium.com/@msmapark2/u-net-%EB%85%BC%EB%AC%B8-%EB%A6%AC%EB%B7%B0-u-net-convolutional-networks-for-biomedical-image-segmentation-456d6901b28a>

3.1.1. U-Net 특징

특징 1. 속도가 빠르다

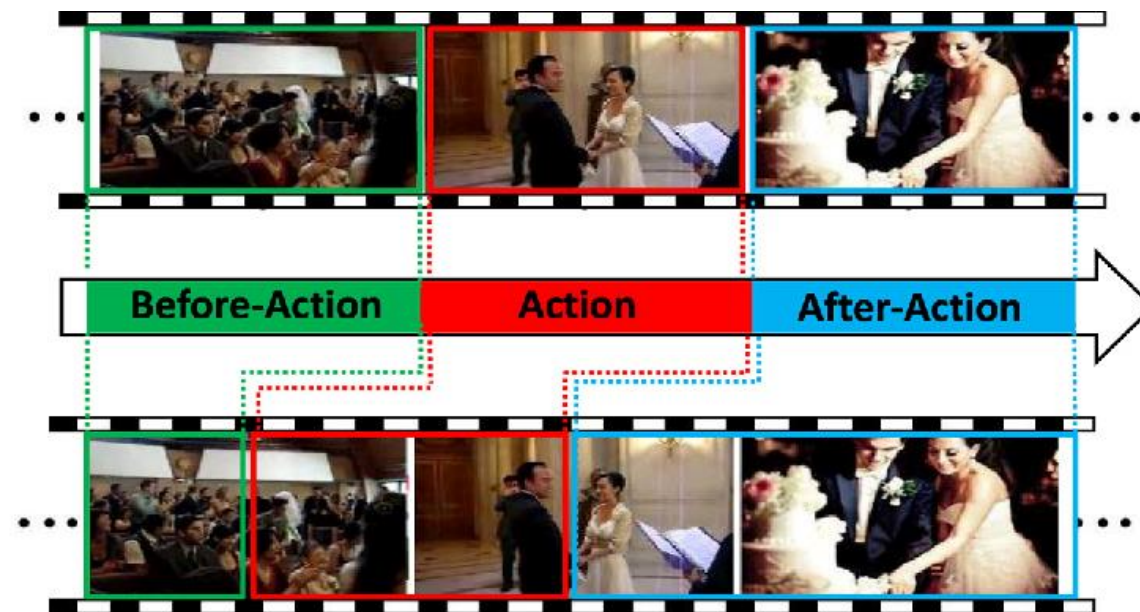
이전 패치에서 검증이 끝난 부분은 건너뛴다



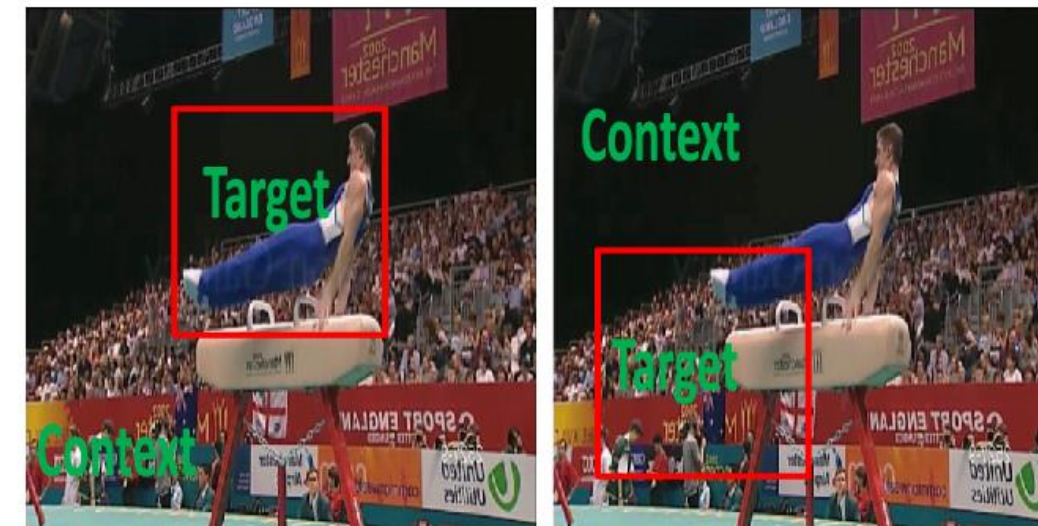
3.1.1. U-Net 특징

특징 2. 트레이드오프 없다

컨텍스트(context) 인식과 지역화(localization) 트레이드오프 문제를 개선했다



(a) Temporal localization.



(b) Spatial localization.

(a) The top one is the accurate temporal localization and the bottom one is inaccurate.

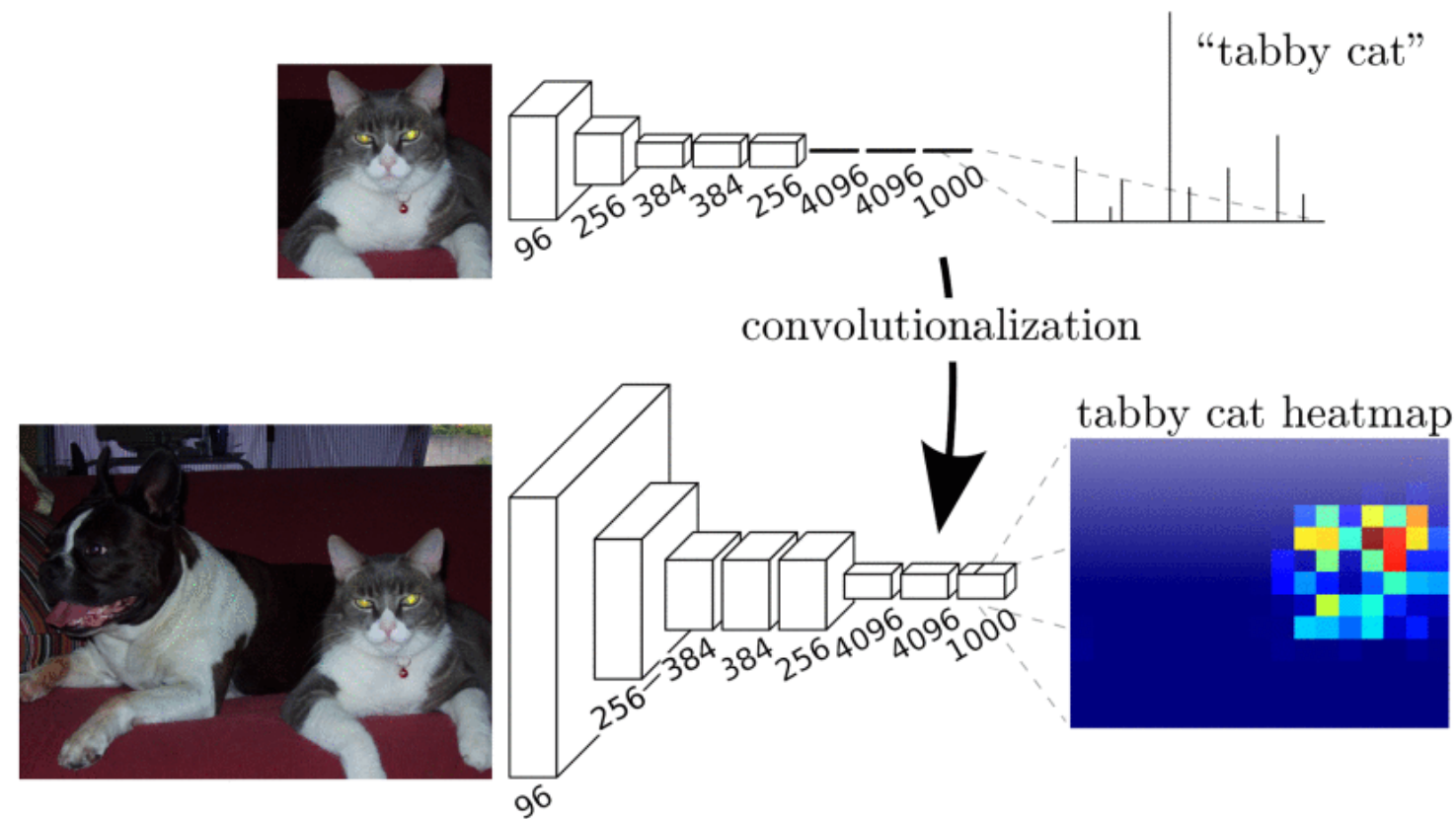
(b) The left one is the accurate spatial localization and the right one is inaccurate.

Inaccurate localization can not only affect the action itself (target), but also has negative effect on its spatio-temporal contexts.

3.1.2. U-Net 구조

U-Net 구조는 FCN을 기반으로 구축되었다.

FCN, 문제는 네트워크의 Pooling을 거치면서 ① 해상도가 떨어지고, ② 객체의 경계가 너무 거칠어(coarse)진다.



VGG 16 for Segmentation

3.1.2. U-Net 구조

① Upsampling (복원)

Ex. 양선형보간법(bilinear interpolation)

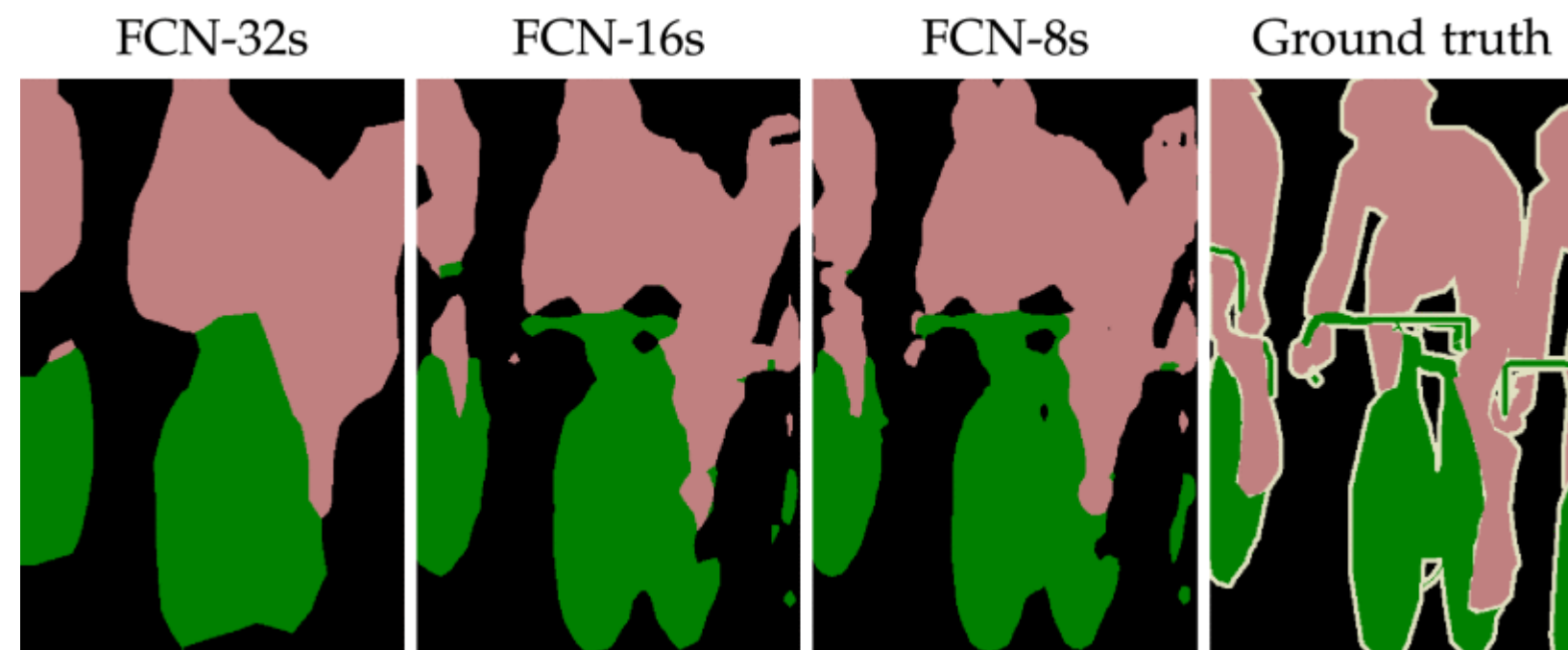
인접한 픽셀 값 사이의 평균값을 **중간 픽셀로 채워 넣는** 방식이다.

Upsampling 시키는 필터의 변수를 학습시키는 Backward Convolution 방식을 적용한다.

② Skip Connection (경계 보존)

앞단의 **피쳐맵**을 upsampling시에 **추가**하여 경계 정보를 한다.

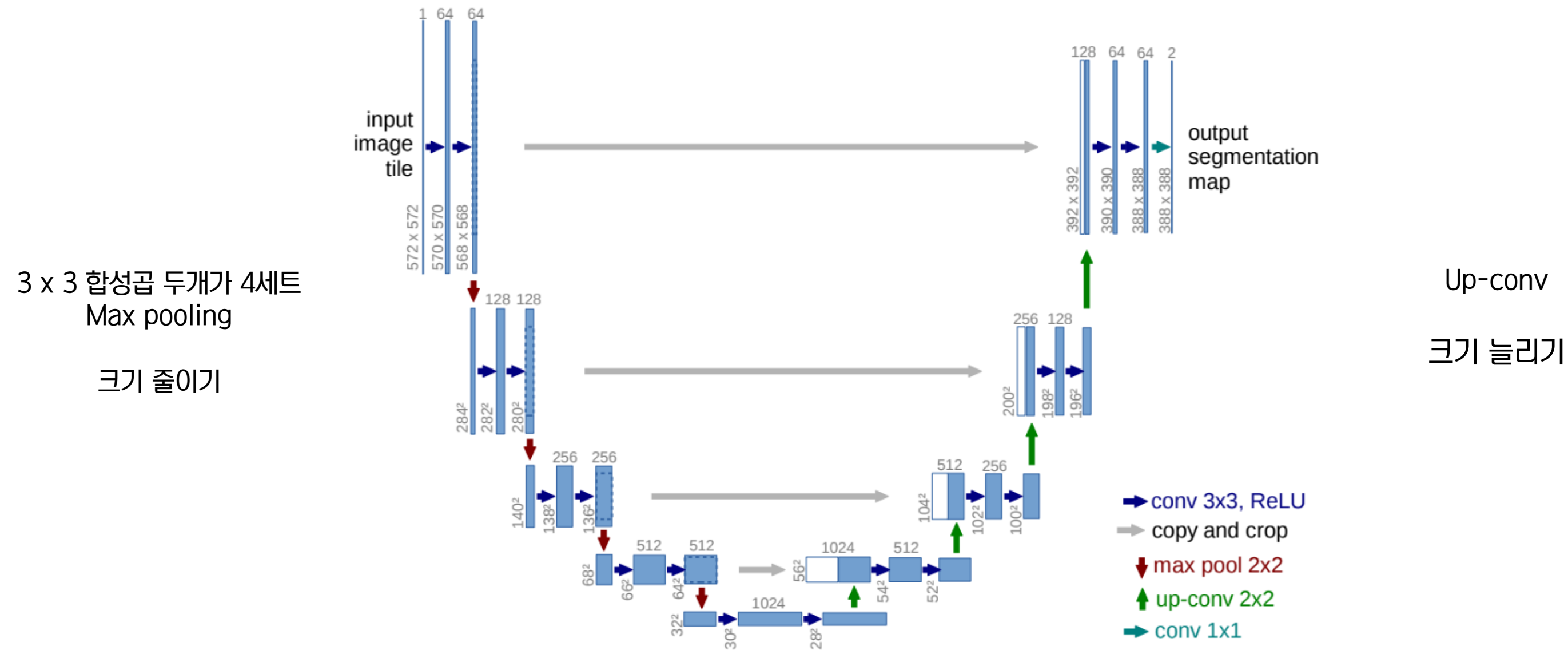
FCN-32s → FCN-16s → FCN-8s 이는 각각 원본 이미지의 1/32, 1/16, 1/8 크기로 압축된 피쳐맵을 추가한 것



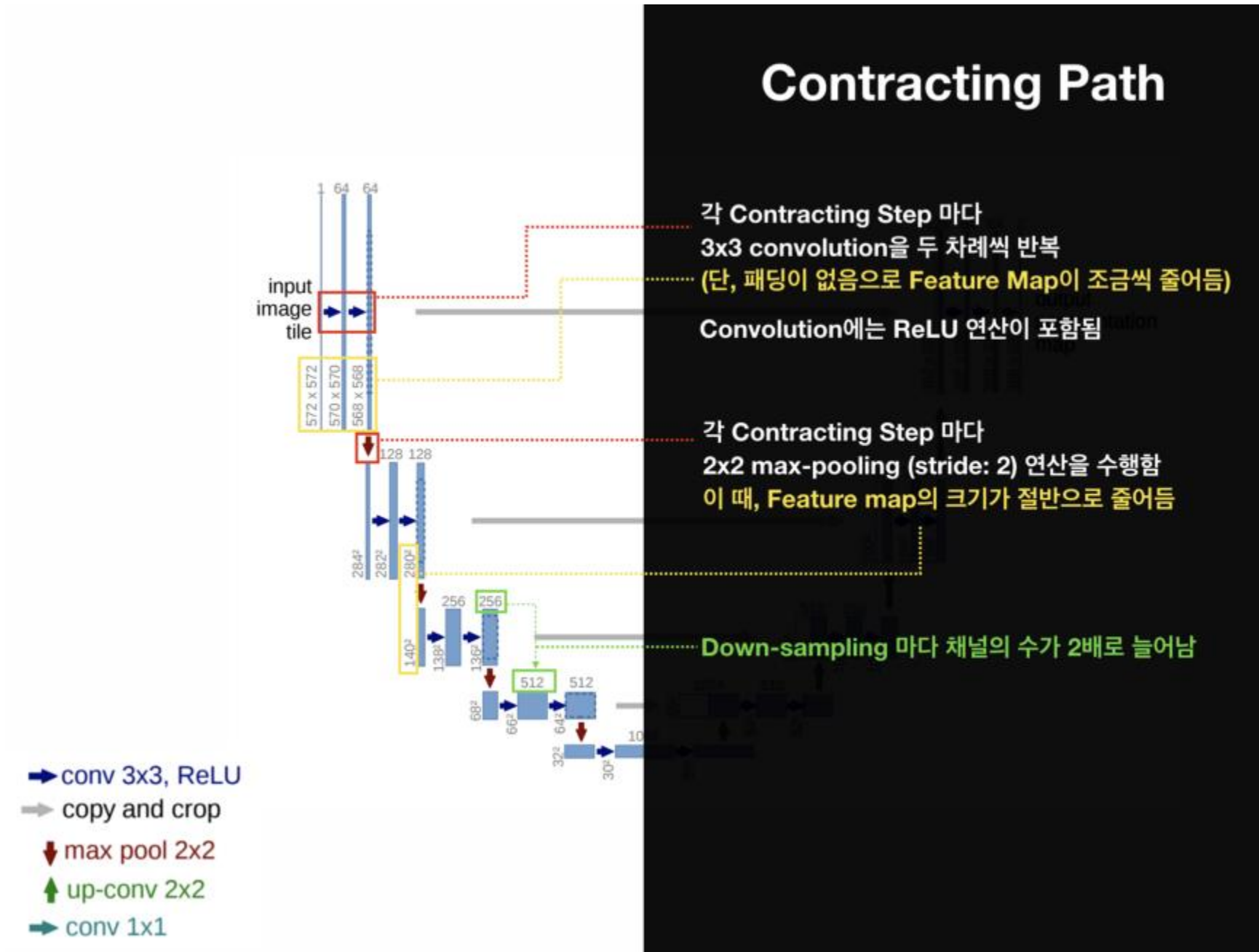
3.1.2. U-Net 구조

이미지를 압축하는 **수축 경로**(contracting path)와 원본 이미지의 크기로 복원하는 **확장 경로**(expansive path)

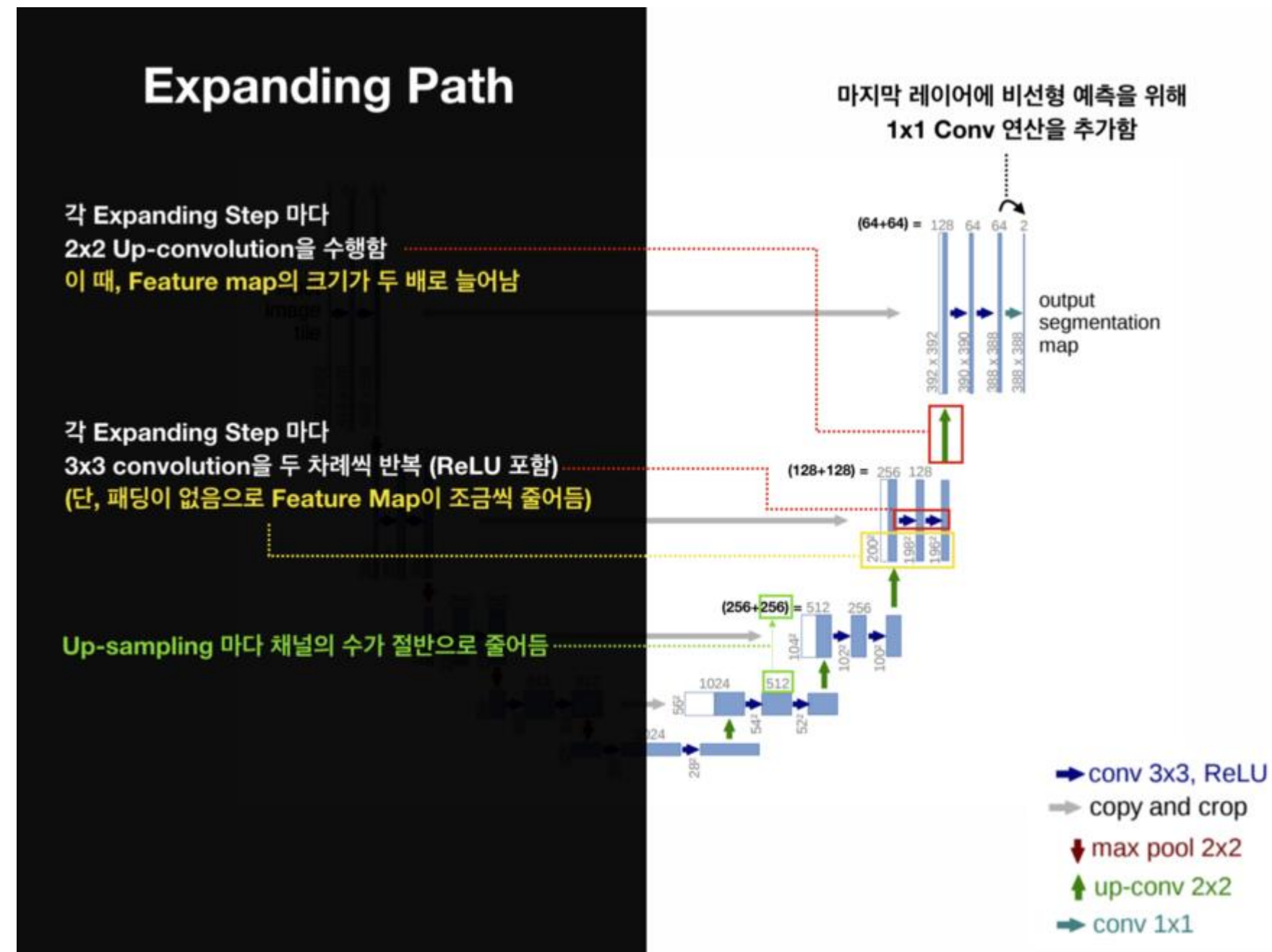
- 수축경로는 컨텍스트를 포착하며
- 확장 경로는 특성 맵을 업 샘플링하고 수축경로에서 포착한 특성 맵의 컨텍스트와 결합하여 정확한 지역화를 수행한다.



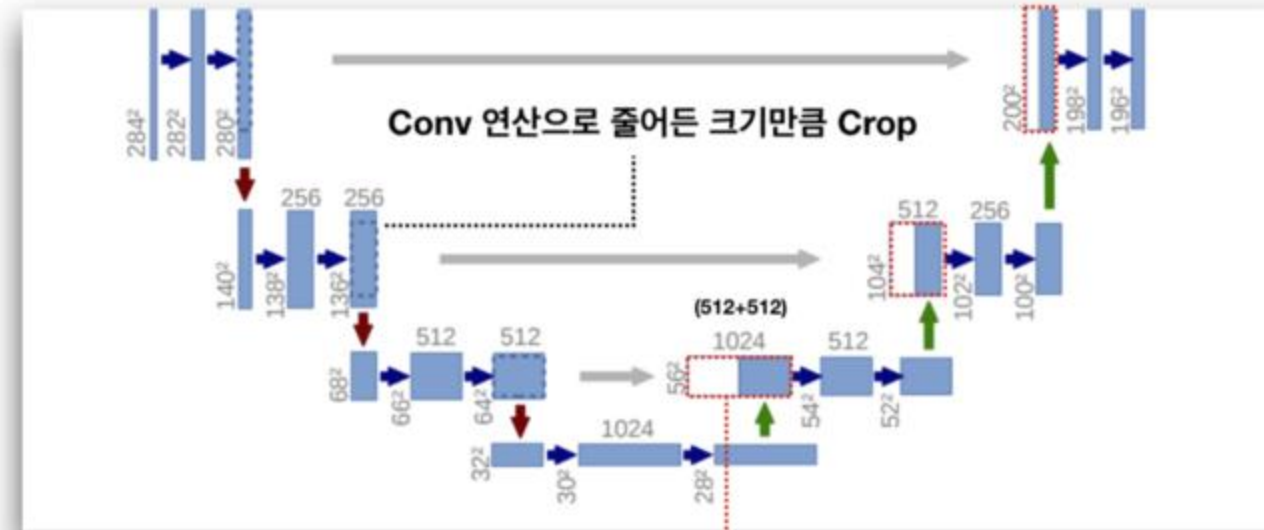
3.1.2. U-Net 구조



3.1.2. U-Net 구조



3.1.2. U-Net 구조



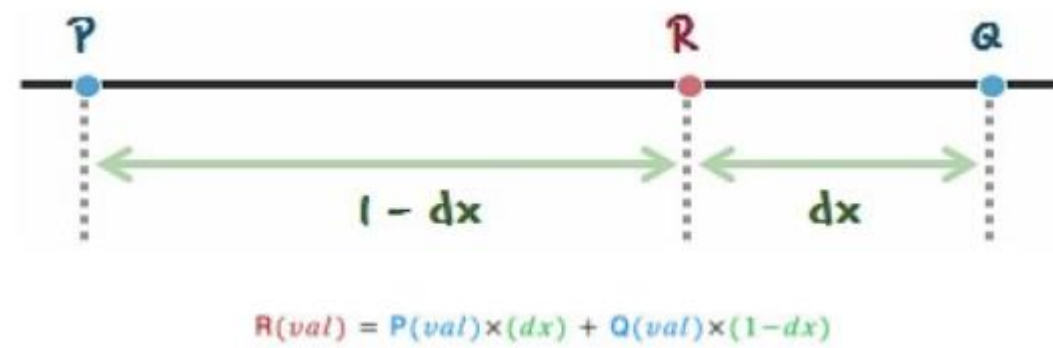
Up-conv 된 특징맵은 Contracting path의 Cropped된 특징맵과 Concatenation 함

Skip Architecture

- ➡ conv 3x3, ReLU
- ➡ copy and crop
- ⬇ max pool 2x2
- ⬆ up-conv 2x2
- ➡ conv 1x1

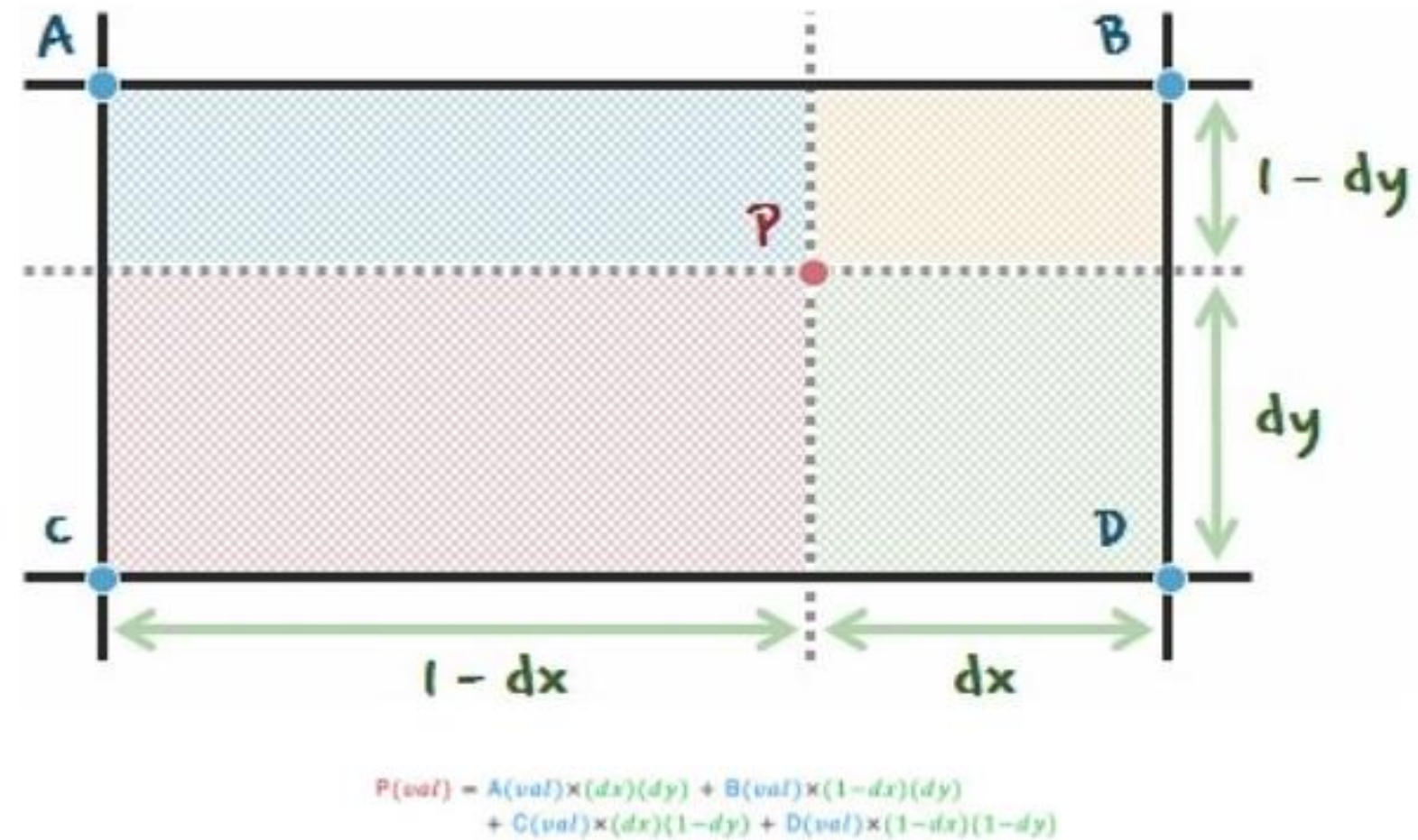
3.1.3. 양선형 보간법

Linear interpolation



Deslauriers-Dubuc Algorithm

bilinear interpolation

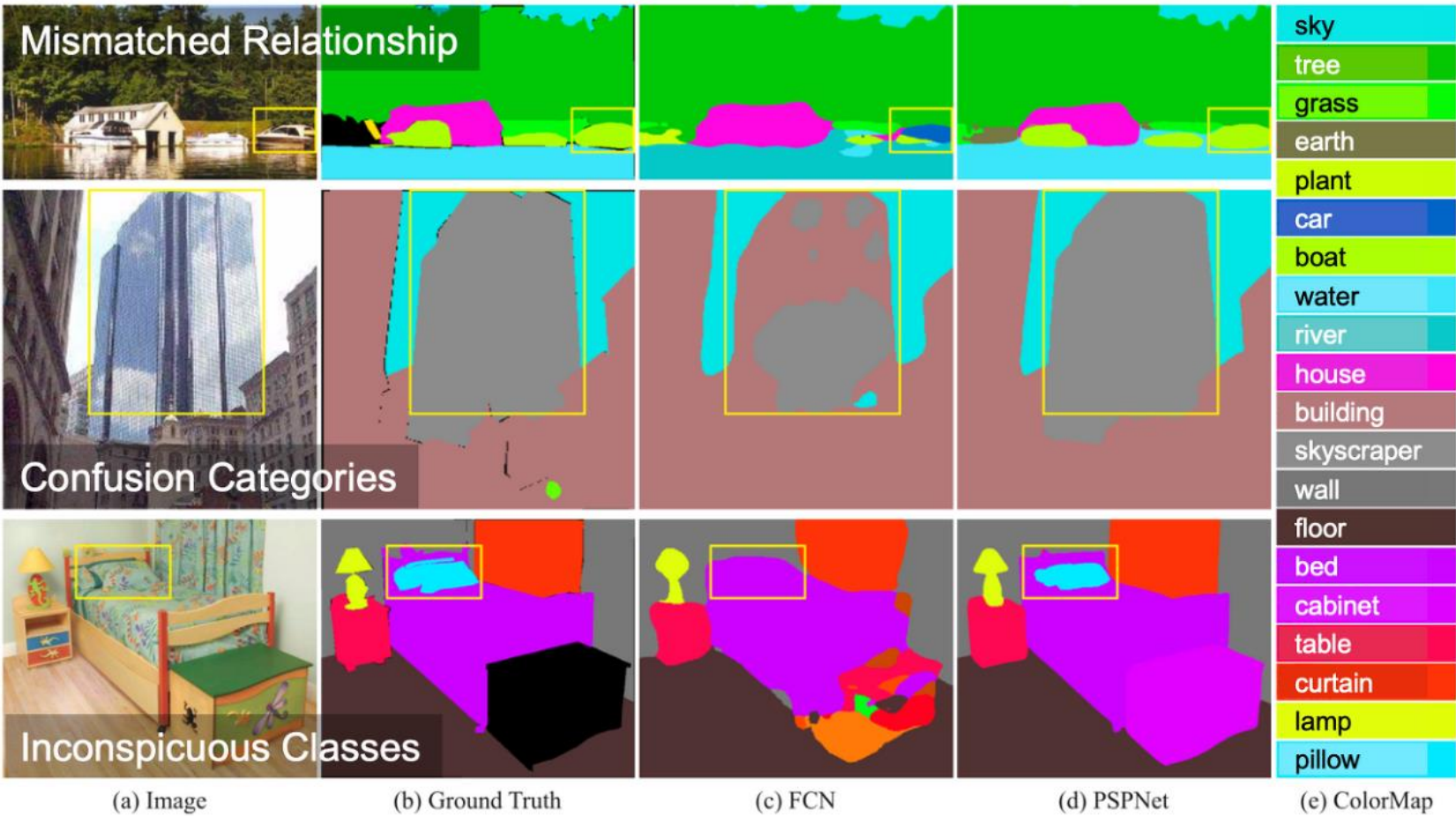


3.2. PSPNet

Q. 어디에 쓰이나요?

시멘틱 분할을 위해 쓰인다 (global contextual information)

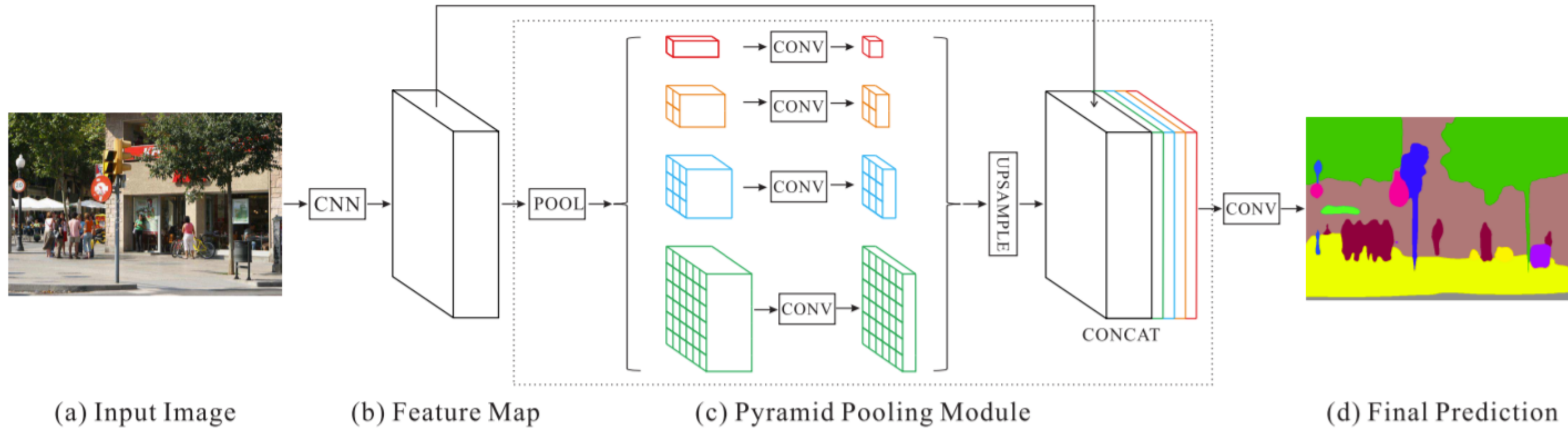
- Mismatched Relationship : 주변 환경(contextual information)과 맞지 않는 픽셀 분류
- Confusion Categories : 헷갈릴 수 있는 픽셀 분류
- Inconspicuous Classes : 눈에 잘 띄지 않는 물체의 픽셀 분류



| Method | Mean IoU(%) | Pixel Acc.(%) |
|-----------------------|--------------|---------------|
| ResNet50-Baseline | 37.23 | 78.01 |
| ResNet50+B1+MAX | 39.94 | 79.46 |
| ResNet50+B1+AVE | 40.07 | 79.52 |
| ResNet50+B1236+MAX | 40.18 | 79.45 |
| ResNet50+B1236+AVE | 41.07 | 79.97 |
| ResNet50+B1236+MAX+DR | 40.87 | 79.61 |
| ResNet50+B1236+AVE+DR | 41.68 | 80.04 |

Table 1. Investigation of PSPNet with different settings. Baseline is ResNet50-based FCN with dilated network. ‘B1’ and ‘B1236’ denote pooled feature maps of bin sizes $\{1 \times 1\}$ and $\{1 \times 1, 2 \times 2, 3 \times 3, 6 \times 6\}$ respectively. ‘MAX’ and ‘AVE’ represent max pooling and average pooling operations individually. ‘DR’ means that dimension reduction is taken after pooling. The results are tested on the validation set with the single-scale input.

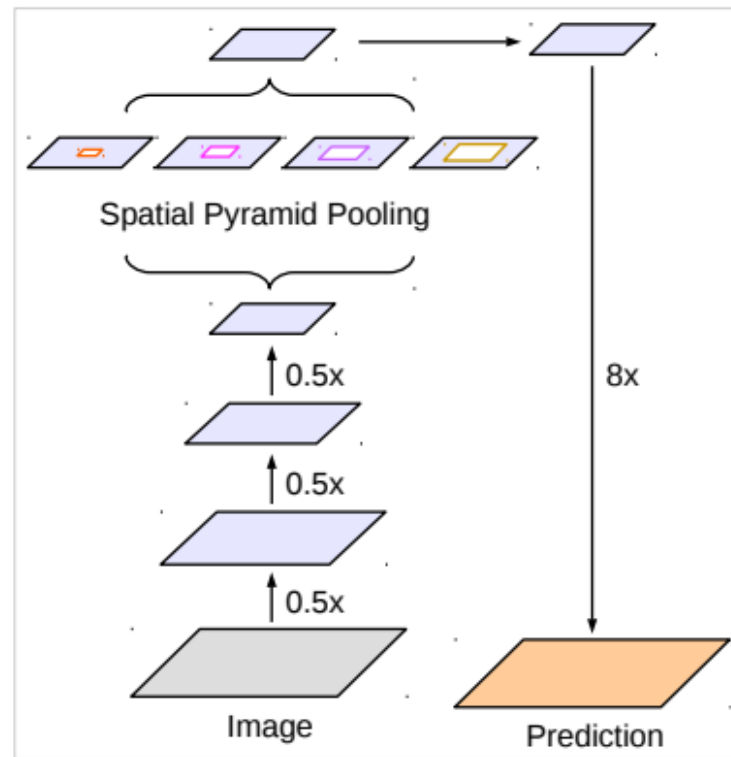
3.2. PSPNet



A. 완전 연결층 한계를 극복하기 위해 피라미드 풀링 모듈을 추가했다

1. 여러 차례 풀링을 수행하여, 각각 다른 크기의 특성 맵이 서로 다른 영역들의 정보를 담게 한다.
2. 1×1 합성곱을 사용하여 채널수를 조절한다. 출력채널수=입력채널수/풀링층개수
3. 특성맵을 업 샘플링(양산형 보간법)
4. 특성 맵 병합

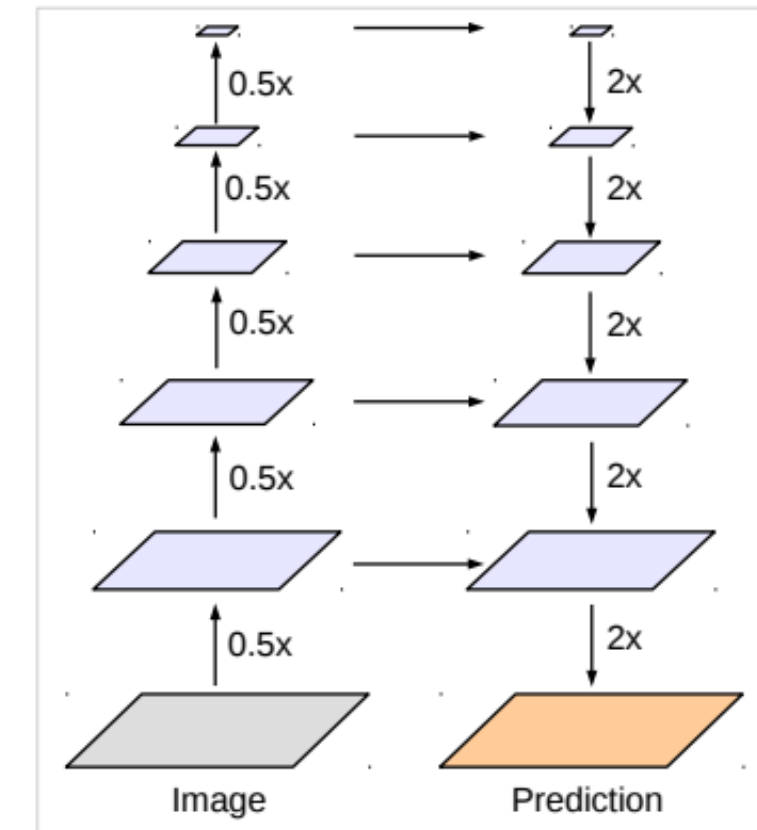
3.3. DeepLabv3 / DeepLabv3+



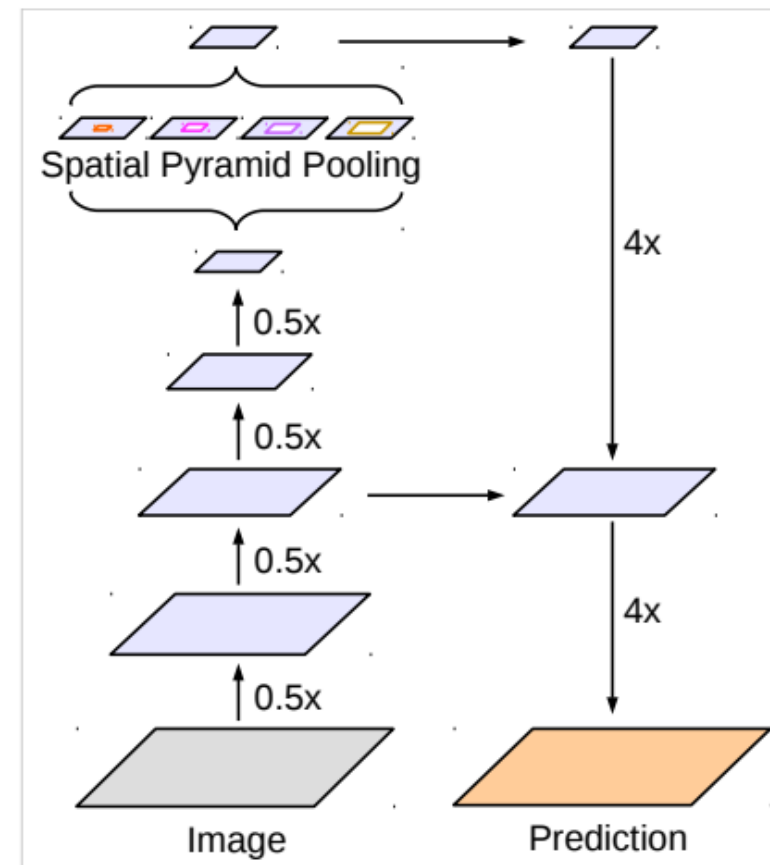
(a) Spatial Pyramid Pooling

(a) mult-scale contextual 정보 추출

(b) 점진적으로 공간정보 확장하여 디테일한 경계 추출



(b) Encoder-Decoder

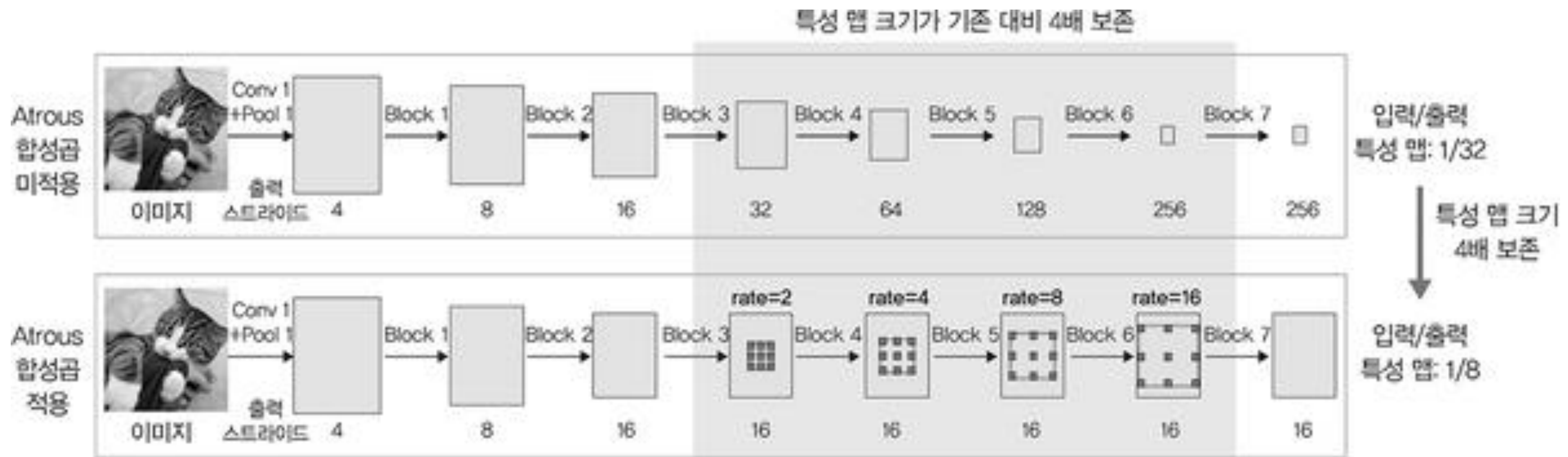


(c) Encoder-Decoder with Atrous Conv

(c) 크기는 **encoder-decoder** 구조를 사용,
encoder에서 **atrous spatial pyramid pooling** 모듈을
사용해서 multi-scale 문맥정보를 통합, 해상도 제어

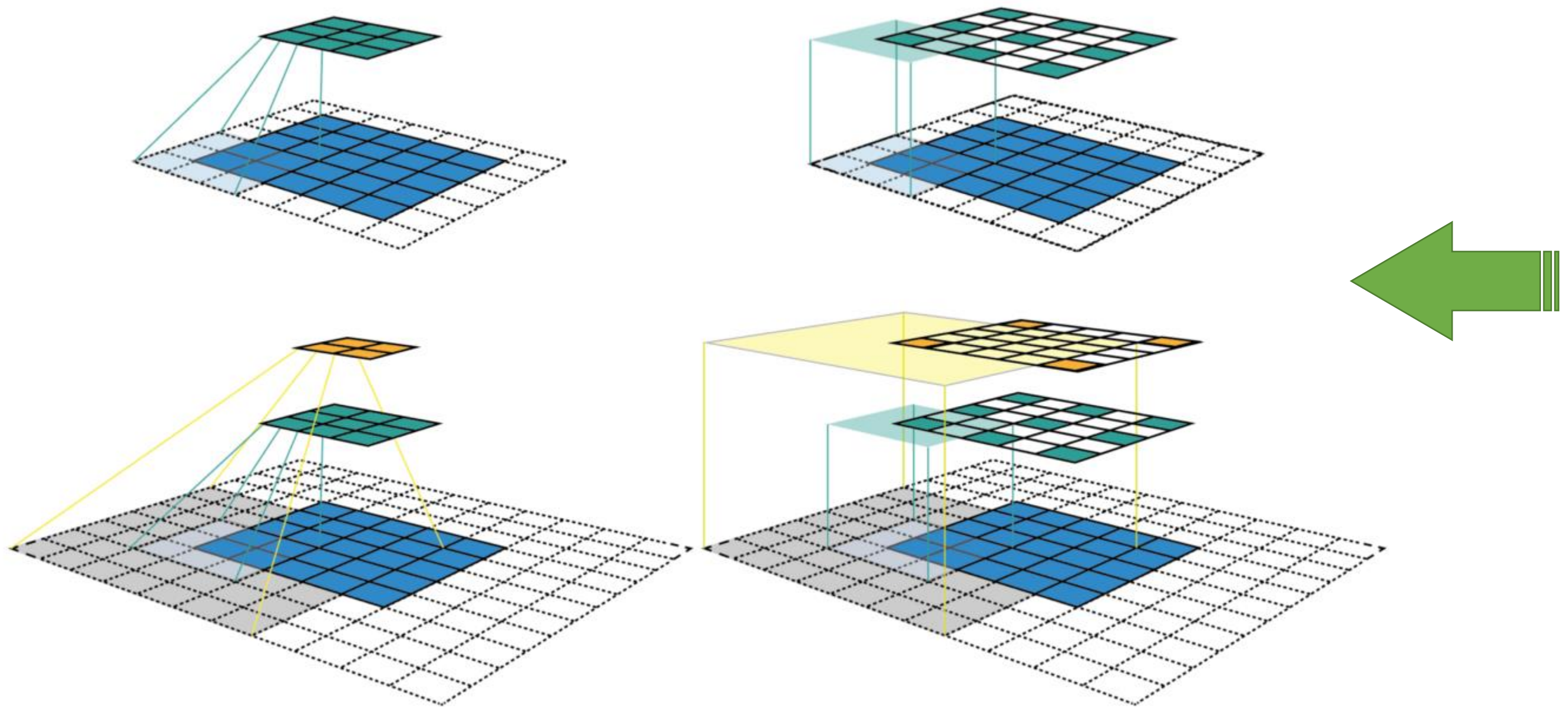
3.3. DeepLabv3 / DeepLabv3+

인코더, Atrous 합성곱을 적용하여, **특성 맵을 기존 대비 4배** 보존한다.



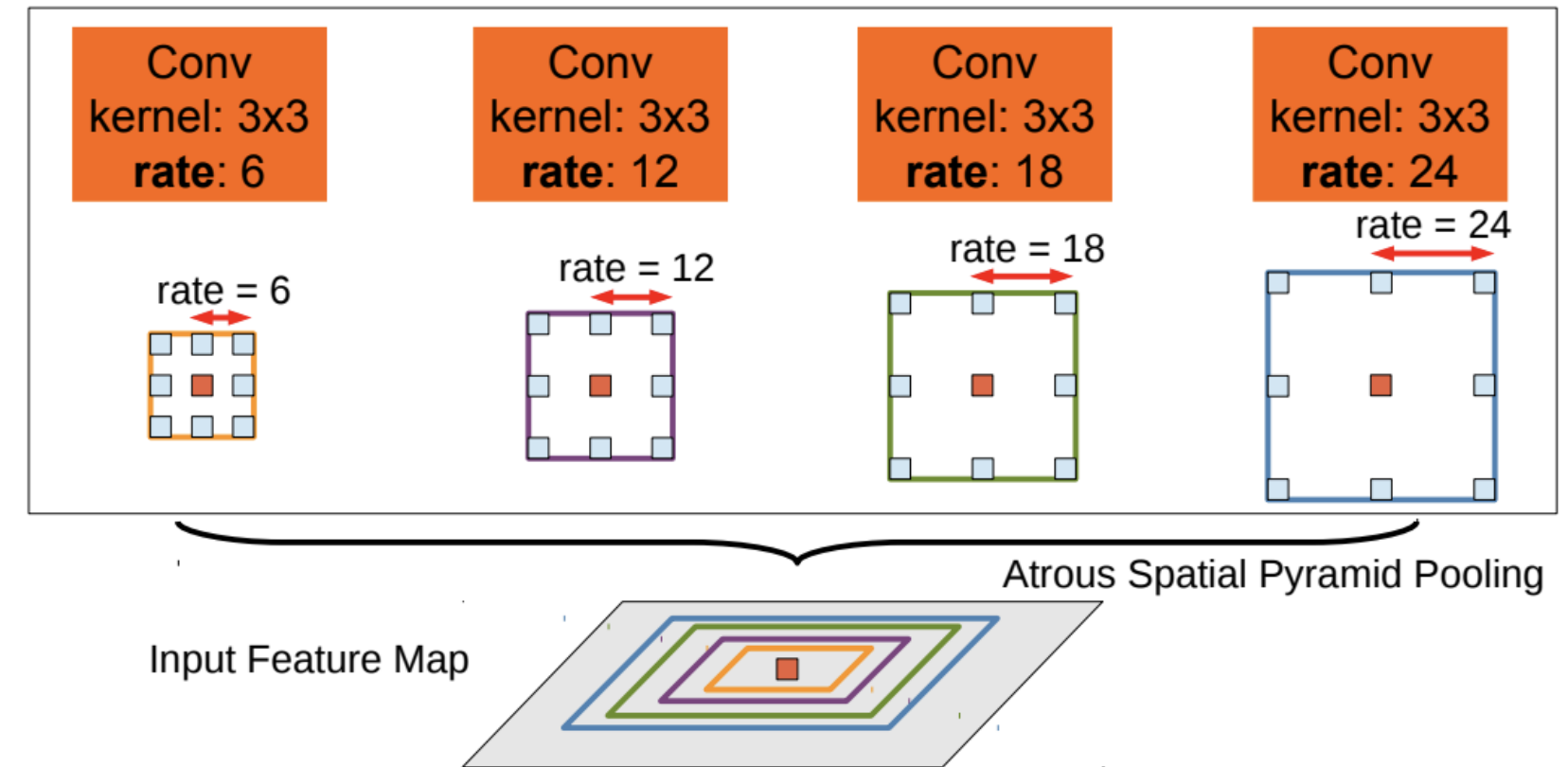
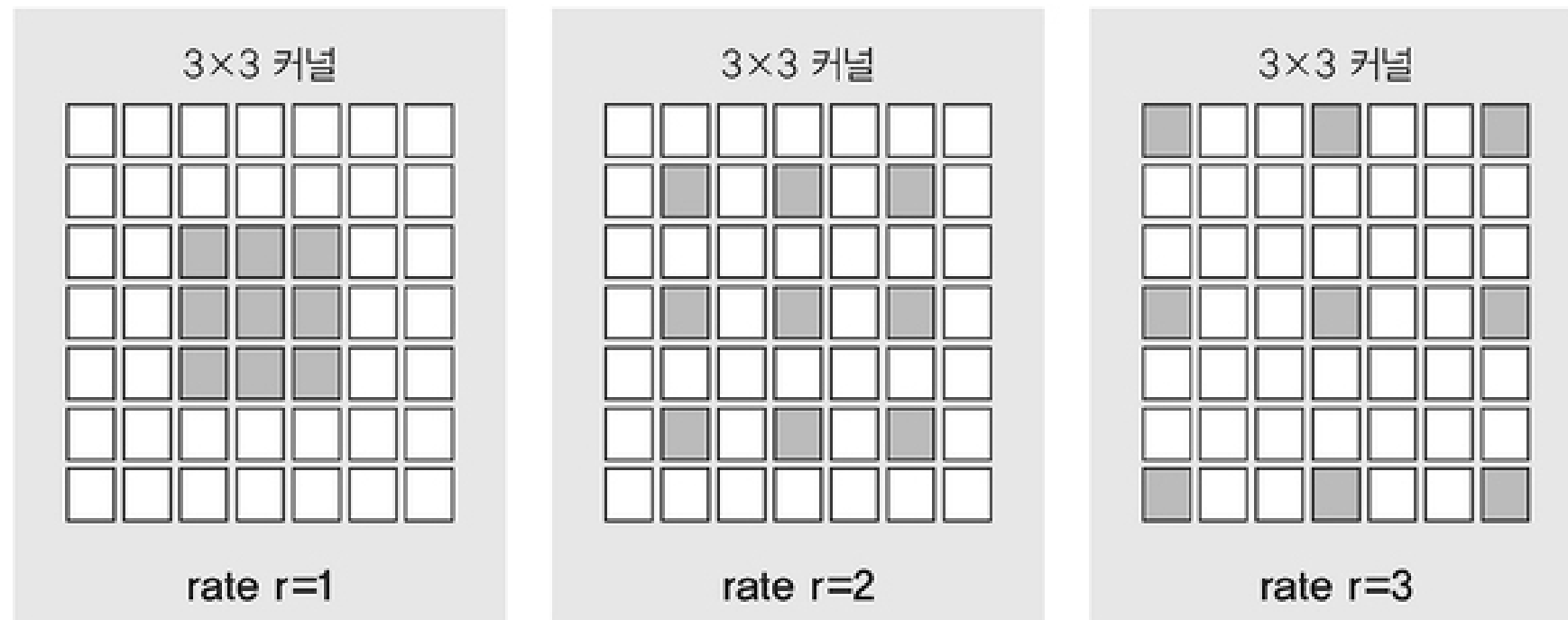
3.3. 1. 수용영역

수용영역 (receptive field),
특성 위치에 있는 픽셀은 거리가 먼 픽셀과의 연관성이 떨어지며, 영향력이 감소한다. 따라서 특정범위를 한정시켜 전체
영역에 대해 서로 다른 중요도를 처리한다.



3.3. DeepLabv3 / DeepLabv3+

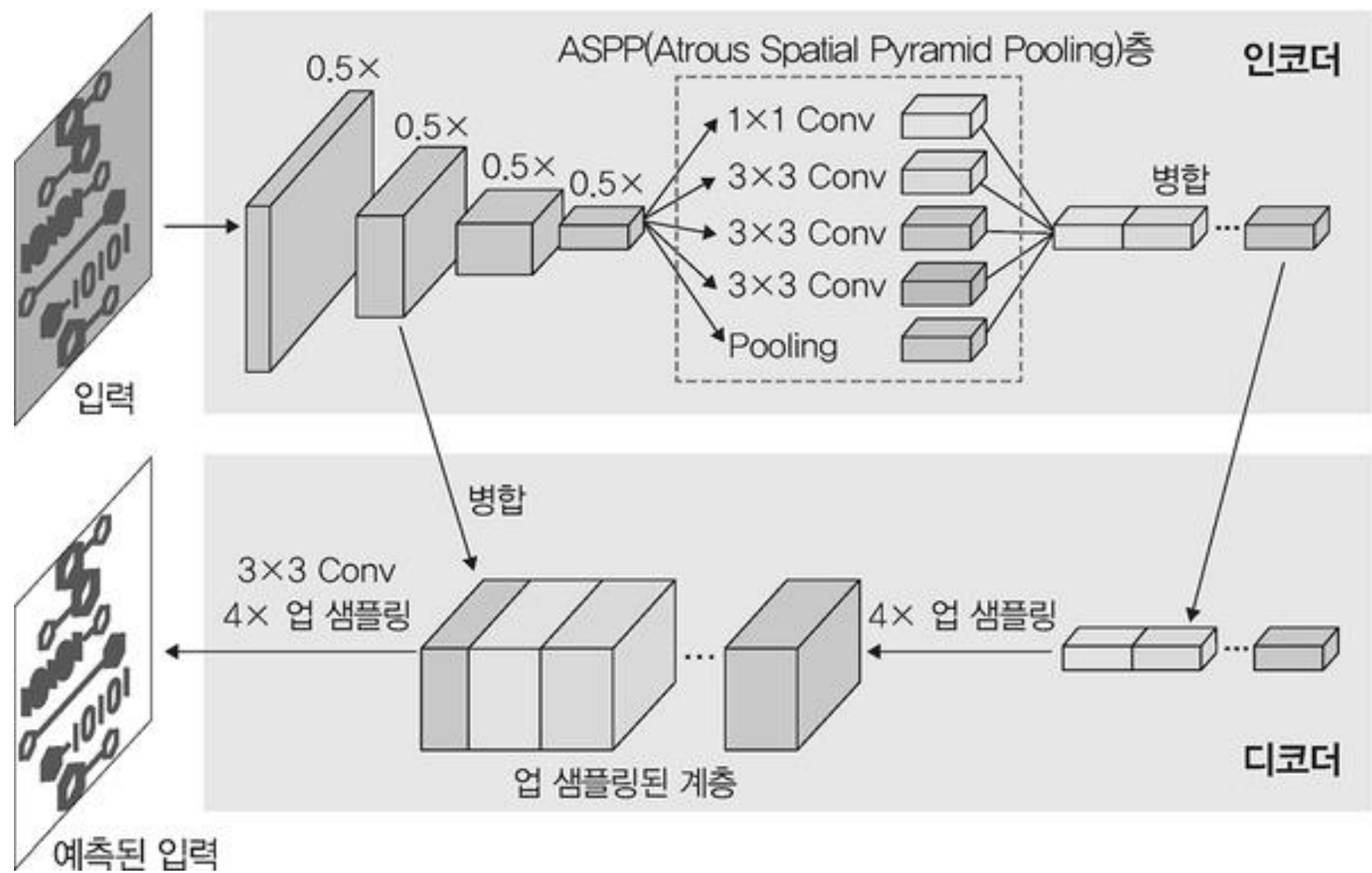
R=1, 기존합성곱



ASPP(atrous spatial pyramid pooling),
이미지 분할에서 수용역역을 확대하면, 특성을 찾는 범위를 넓게 해 주기 때문에, 파라미터 수를 늘리지 않고 높은 성능을 낸다.
빈 공간을 결정하는 파라미터 rate가 커질 수록 빈 공간 커진다.

3.3. DeepLabv3 / DeepLabv3+

<https://arxiv.org/abs/1808.08448>
Encoder-Decoder with Atrous Separable Convolution for ...
LC Chen 저술 · 2018 · 3232회 인용 — Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. ... Specifically, our proposed model, DeepLabv3+, extends DeepLabv3 by adding a simple yet effective decoder module to refine the segmentation results especially along object boundaries.



| DeepLab v3 (디코더 X) | DeepLab v3+ (디코더 O) |
|-----------------------|------------------------|
| 85.7 | 87.8 |

- 디코더
- 1차적으로 4배 upsampling 되고, 백본에서 상응하는 low-level features와 concat된다.
 - 3x3 conv를 통해 features를 정교화시켜준 뒤, 2차적으로 4배 양산형보간법을 시켜줘서 크기를 input과 맞춰준다.

THANK YOU

