2023 D&A

Deep Session 5차시

CNN 심화 I

(LeNet, AlexNet, VGG)

CONTENTS

1. CNN 모델 개요

CNN 모델의 발전

1998 — LeNet

2012 — AlexNet

2014 — GoogLeNet, VGG

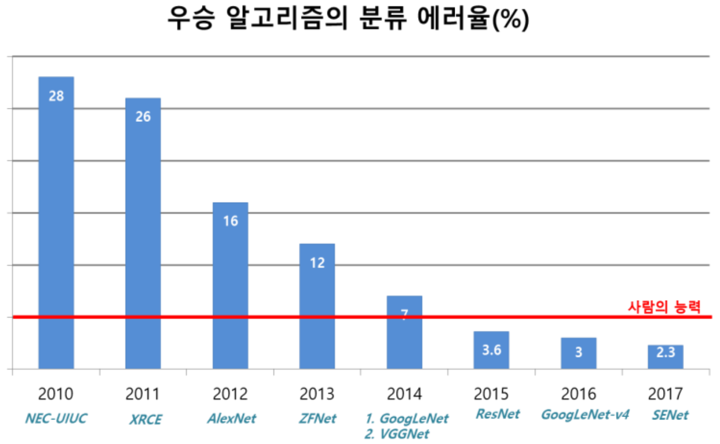
2015 — ResNet

- LeNet: 최초의 CNN 모델

ILSVRC (Imagenet Large Scale Visual Recognition Challenge)

이미지 인식 경진대회로 대용량의 이미지 데이터셋(Imagenet)을 주고 이미지 분류 알고리즘의 성능을 평가

- 2012년 AlexNet이 오류율을 크게 낮추며 딥러닝이 큰 주목을 받게 되었다.
- 그 후 딥러닝을 활용한 기법이 꾸준히 정확도를 개선해 오고, 컴퓨터 비전 분야에 큰 역할을 해 왔다.



2. LeNet

개요

- 1998년 Yann Lecun 연구팀이 개발한 최초의 CNN 알고리즘
- Yann Lecun 팀의 논문 'Gradient-Based Learning Applied to Document Recognition'에 수록되어 있는 LeNet-5가 대표적
- 손글씨 숫자를 인식하는 네트워크 → MNIST 데이터셋 사용 (0~9의 손글씨)

- 32x32 크기의 흑백 이미지에서 학습된 7 layer CNN
- [Input – Conv(C1) – Subsampling(S2) – Conv(C3) – Subsampling(S4) – Conv(C5) – FC6 – FC7(output)]

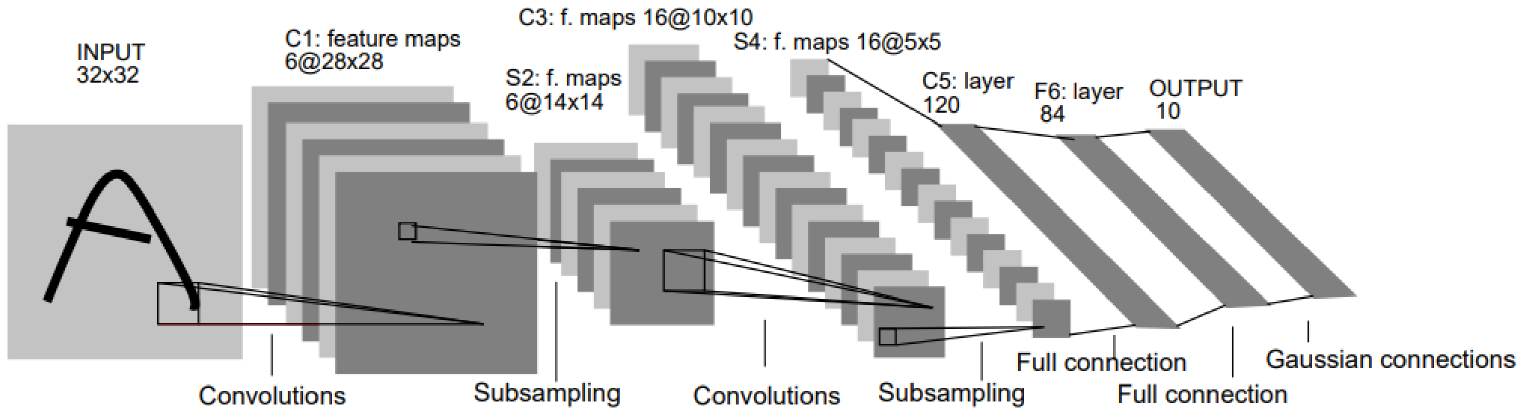
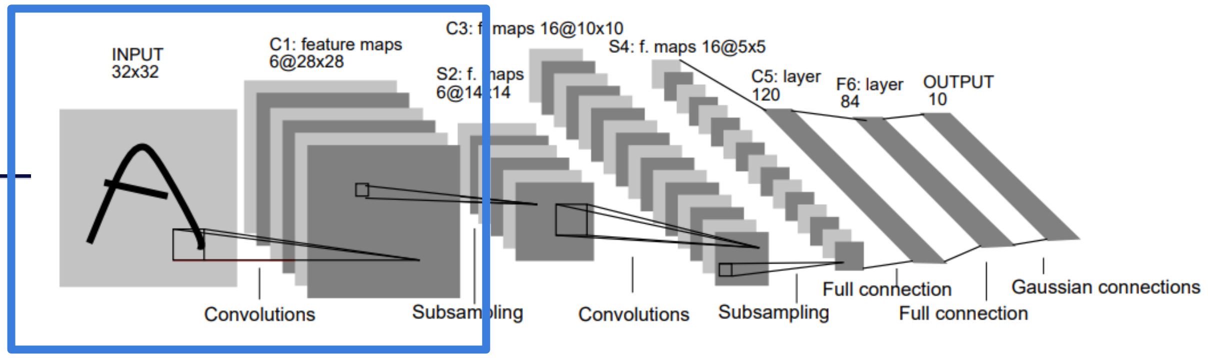


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

2. LeNet – Conv1



Input

Input

Input size: (1, 32, 32)

Convolution

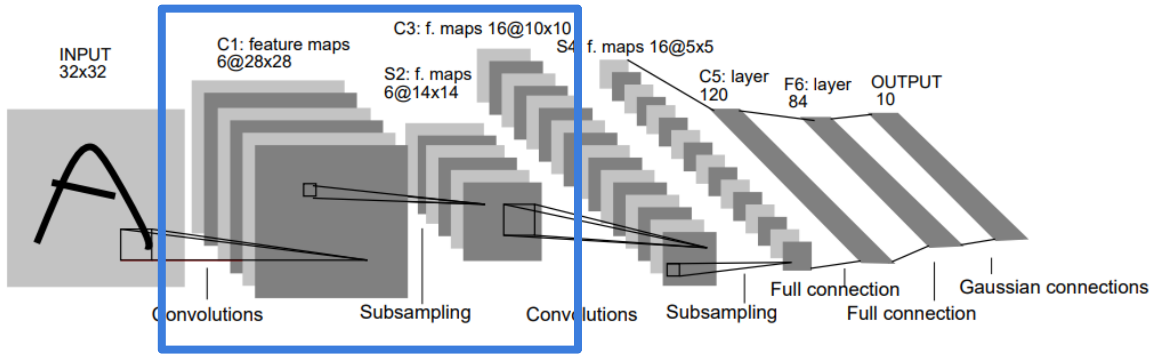
Filter size: 5x5
Filter 수: 6
Stride: 1

C1 layer

C1 Output

size: (6, 28, 28)

2. LeNet – Subsampling2



- 당시에 subsampling이라고 불렸으나 현재의 pooling과 동일한 역할

C1 layer

C1 Output

size: (6, 28, 28)

Average Pooling

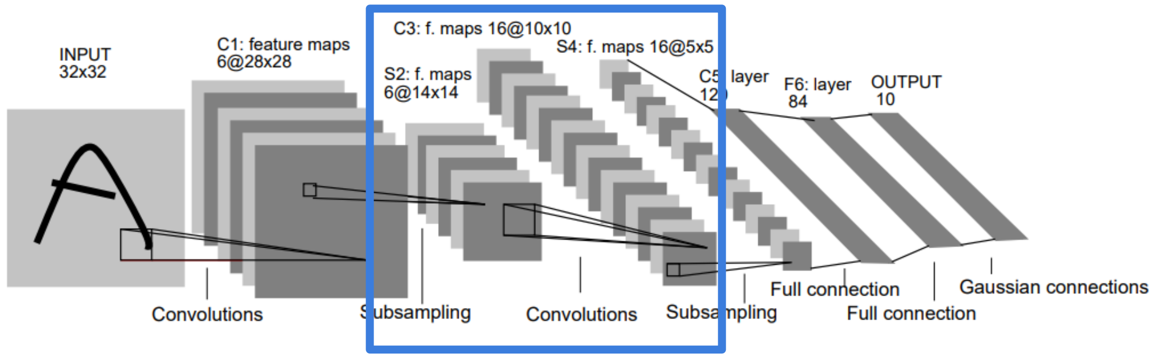
Pooling size: 2x2
Stride: 2

S2 layer

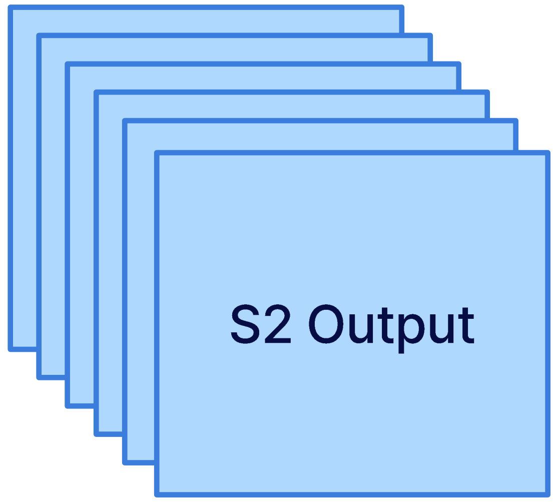
S2 Output

size: (6, 14, 14)

2. LeNet – Conv3



S2 layer

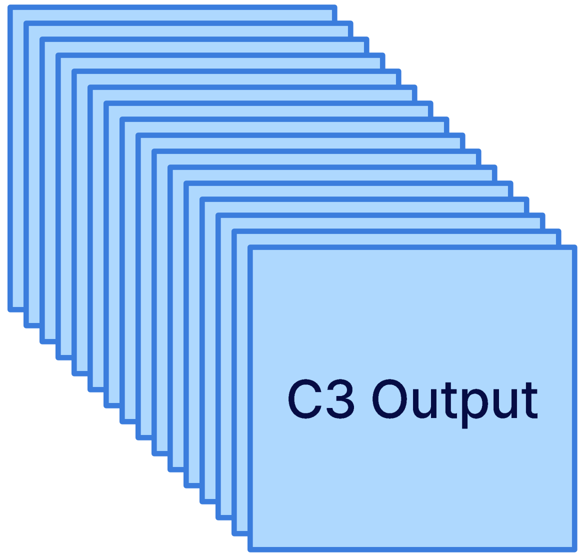


size: (6, 14, 14)

Convolution

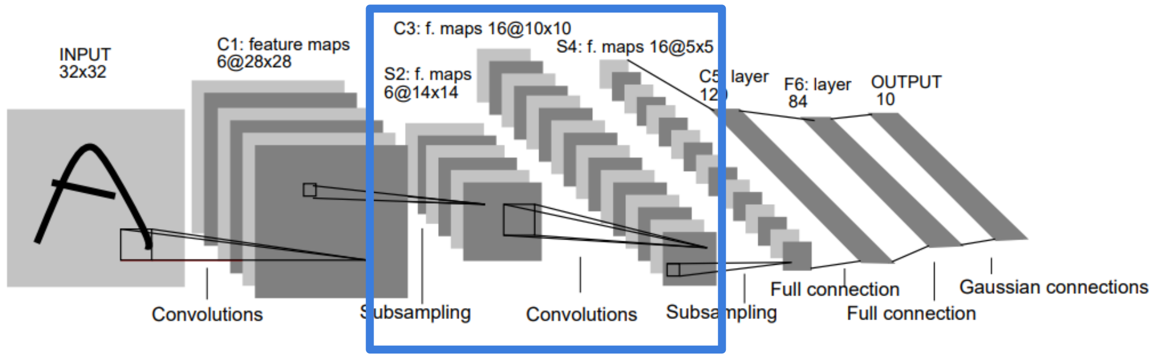
Filter size: 5x5
Filter 수: 16
Stride: 1

C3 layer



size: (16, 10, 10)

2. LeNet – Conv3



C3 layer의 feature map

S2 layer의 feature map

	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.

① 연속된 3장을 모아 Convolution
→ 6장의 10x10 feature map 생성

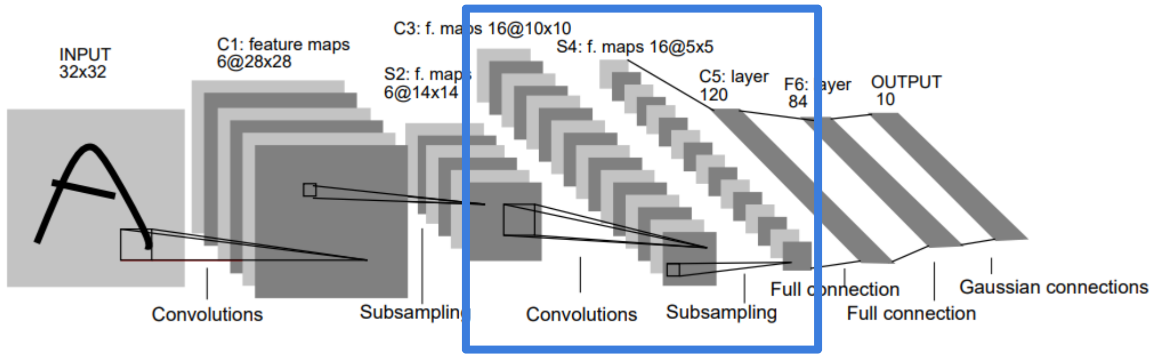
② 연속된 4장을 모아 Convolution
→ 6장의 10x10 feature map 생성

③ 불연속한 4장을 모아 Convolution
→ 3장의 10x10 feature map 생성

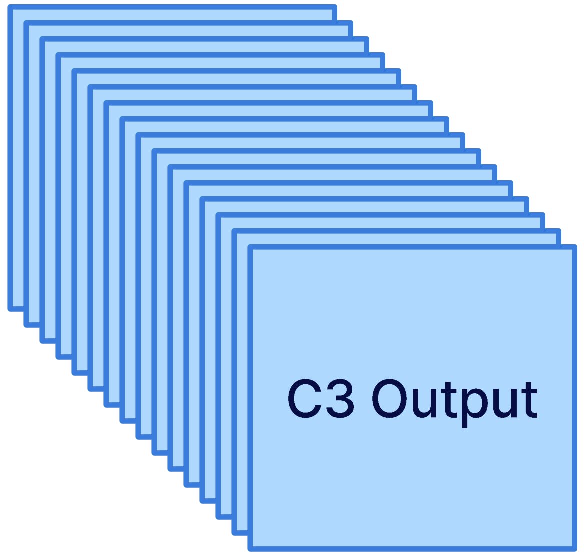
④ 6장 모두 Convolution
→ 1장의 10x10 feature map 생성

연산마다 서로 다른 조합의 입력값을 취해서
보다 다양한 특징을 찾아 global feature로 나타나기를 기대

2. LeNet – Subsampling4



C3 layer

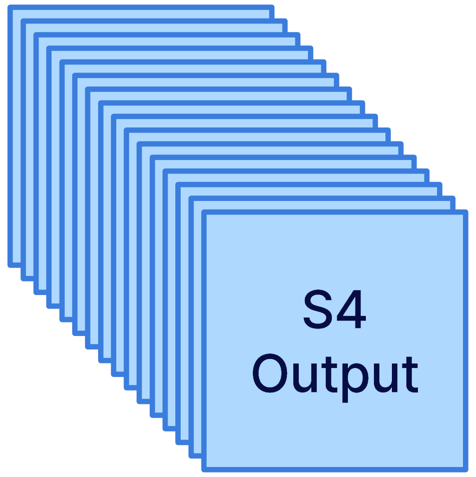


size: (16, 10, 10)

Average Pooling

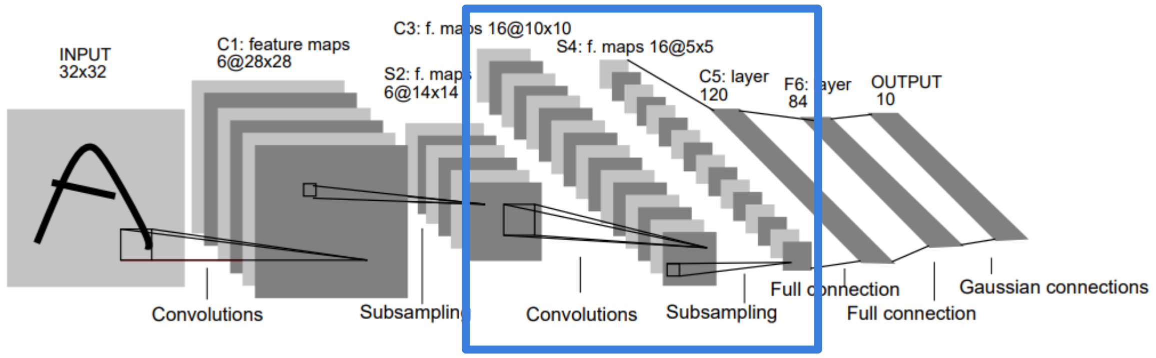
Pooling size: 2x2
Stride: 2

S4 layer

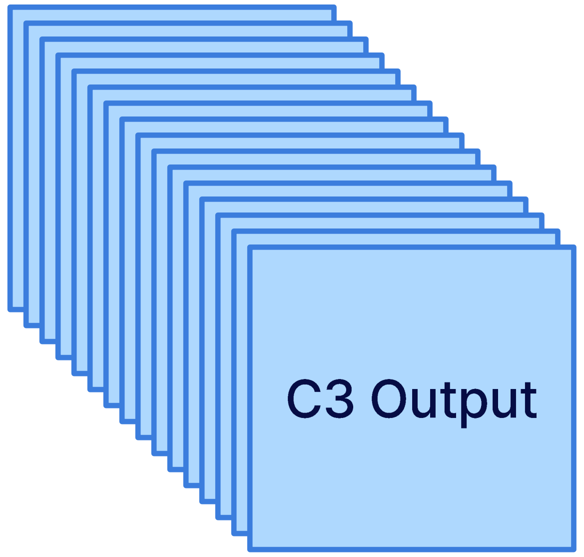


size: (16, 5, 5)

2. LeNet – Conv5



S4 layer

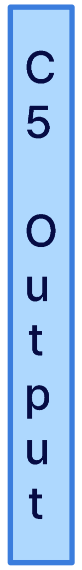


size: (16, 5, 5)

Convolution

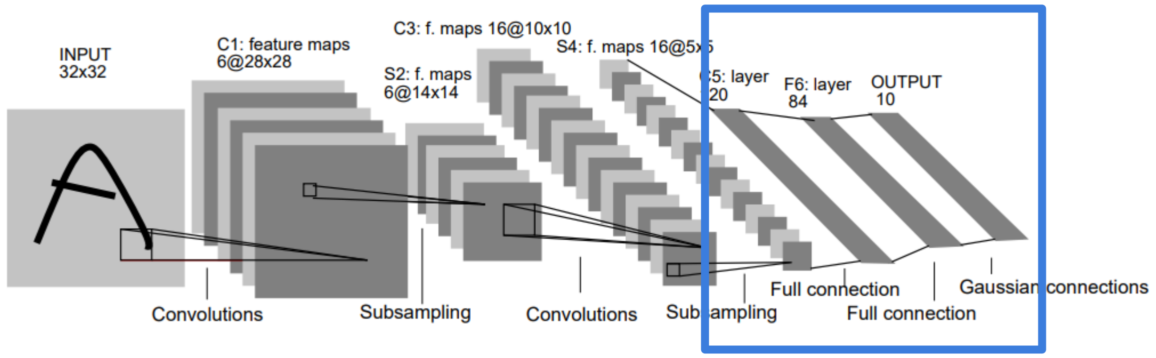
Filter size: 5x5
Filter 수: 120
Stride: 1

C5 layer



size: (120, 1, 1)

2. LeNet – Fully Connected



C5 layer

C5 Output

size: (120)

tanh

Fully connected

F6 layer

F6 Output

size: (84)

Eclidean Radial Basis Function

Fully connected

Output

Output

Output size: (10)

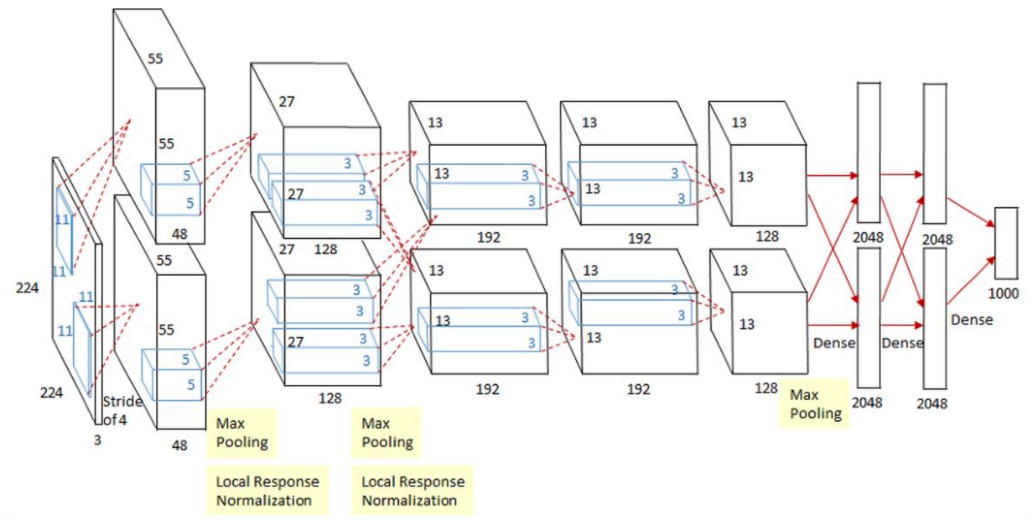
* Eclidean Radial Basis Function
: 입력값과 학습된 가중치들 사이의 거리를 계산 후,
이 거리값을 가지고 가우시안 분포 함수의 값을 계산하여 확률을 계산

3. AlexNet

개요

- 2012년 ILSVRC 대회 우승 모델
- 딥러닝 열풍을 일으키는 데 큰 역할
- 2개의 GPU로 병렬연산을 수행하기 위해 병렬적인 구조로 설계

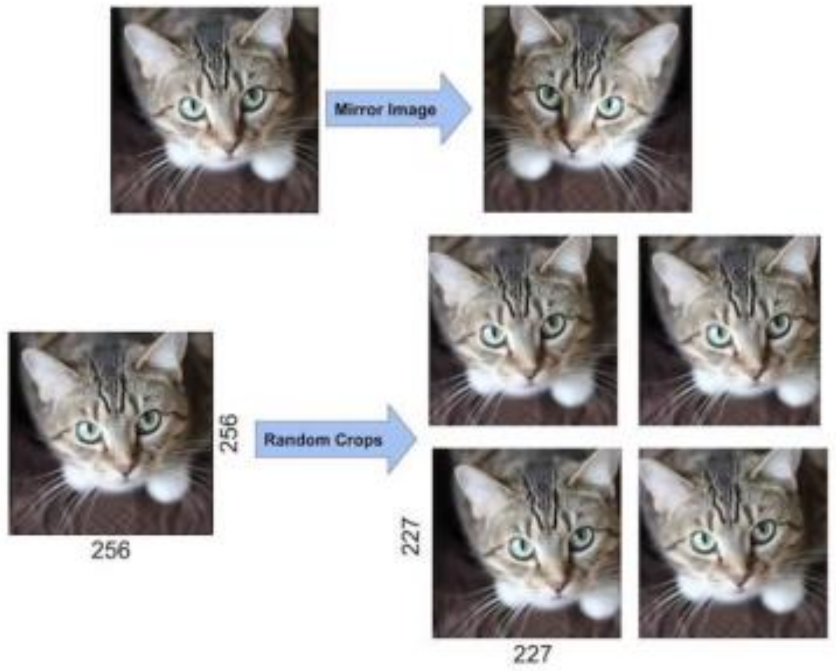
- 227x227 크기의 RGB 3 Channel 이미지를 Input으로 사용
- [Input – Conv1 – MaxPool1 - Conv2 – MaxPool2 – Conv3 – Conv4 – Conv5 – MaxPool5 – FC6 – FC7 – FC8(output)]



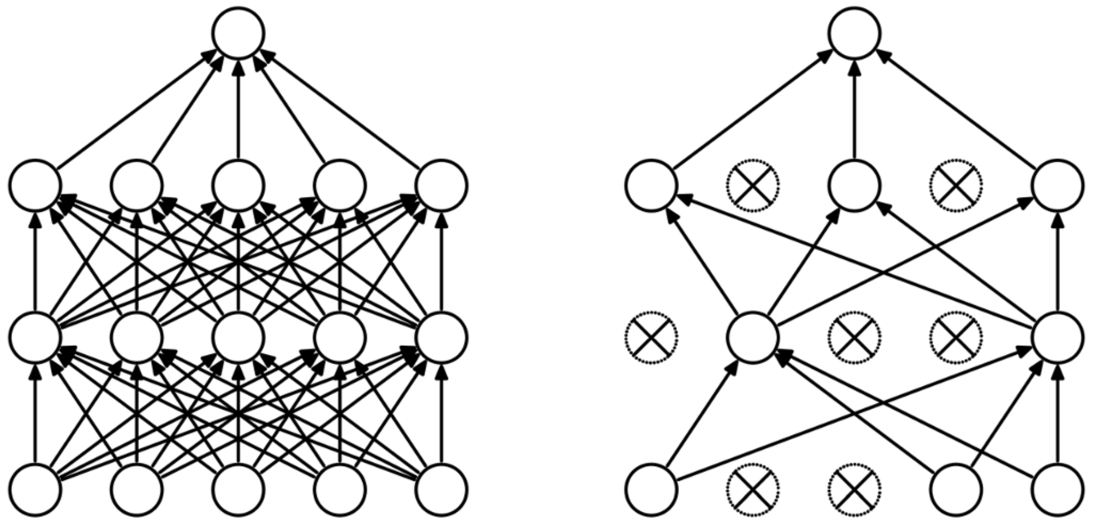
3. AlexNet

Data Augmentation

1. 좌우 반전을 통해 이미지 양 2배 증가
2. 256x256 이미지를 랜덤으로 잘라서 227x227 만듦



Dropout


(a) Standard Neural Net
(b) After applying dropout.

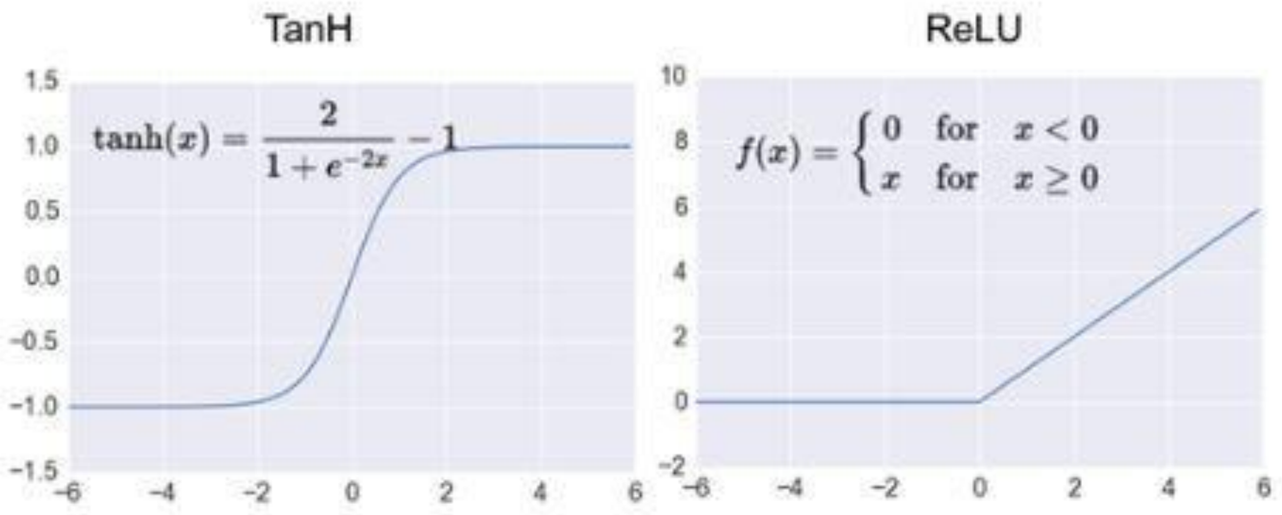
→ overfitting 방지

3. AlexNet

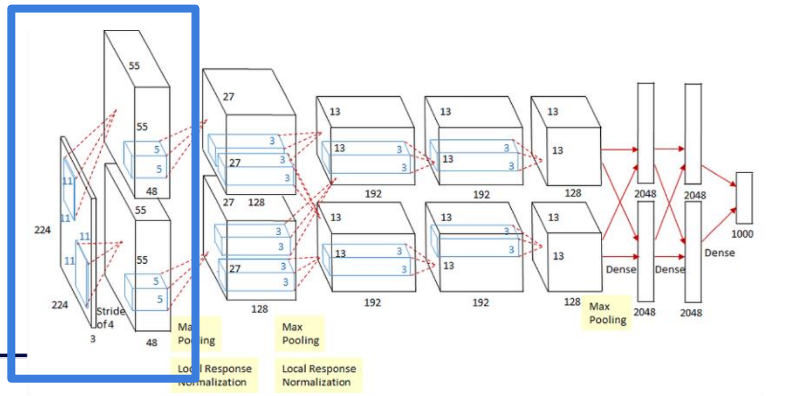
ReLU 함수

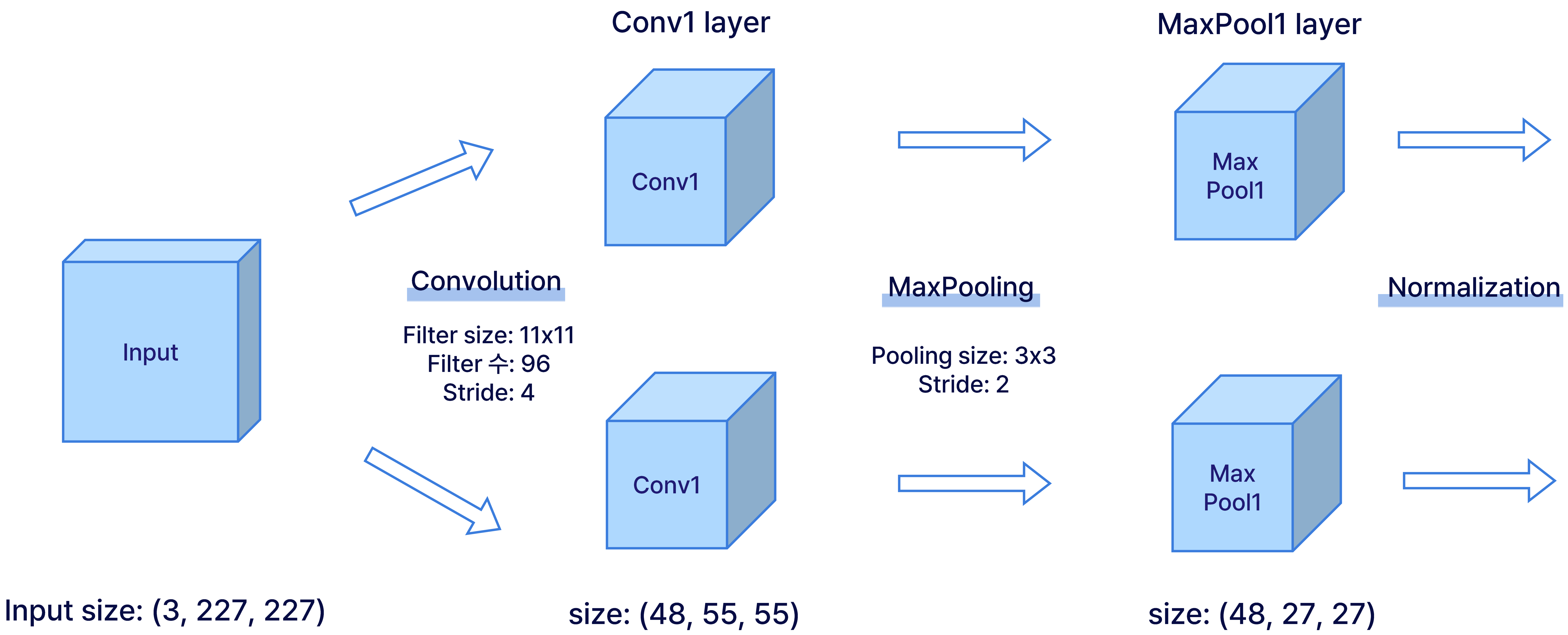
활성화 함수로 ReLU 사용

- LeNet-5에서는 tanh 사용
  → 점점 기울기가 0에 수렴하여 역전파의 경우 기울기가 소실하게 되는 문제
- ReLU를 사용하는 것이 같은 정확도를 유지하면서 tanh를 사용하는 것보다 6배나 빨라 AlexNet 이후로는 ReLU 함수를 주로 사용



3. AlexNet – Conv1, MaxPool1



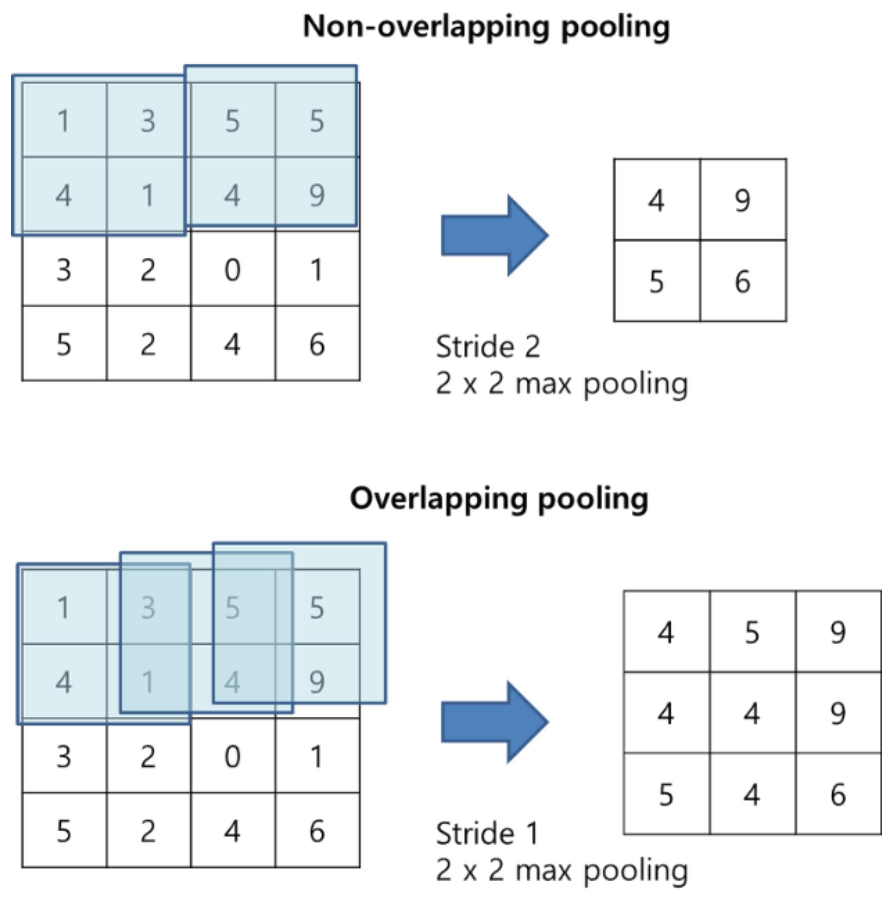


3. AlexNet – Conv1, MaxPool1

MaxPooling layer

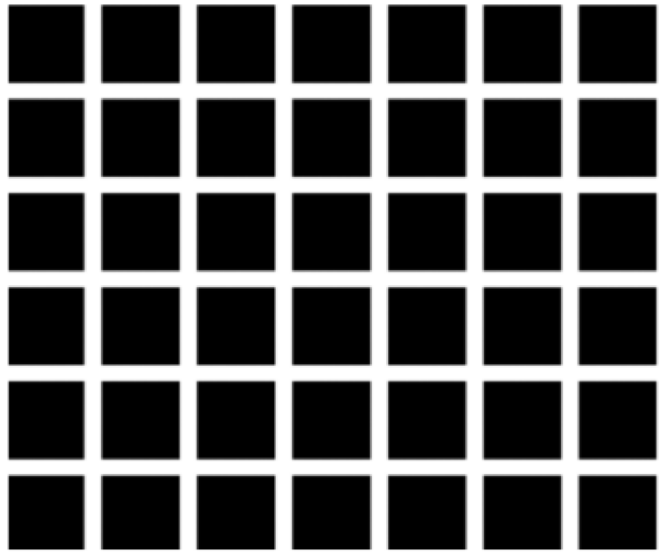
Overlapping maxpooling 사용

→ 정보의 손실을 최소화하고 Overfitting 방지



3. AlexNet – Conv1, MaxPool1

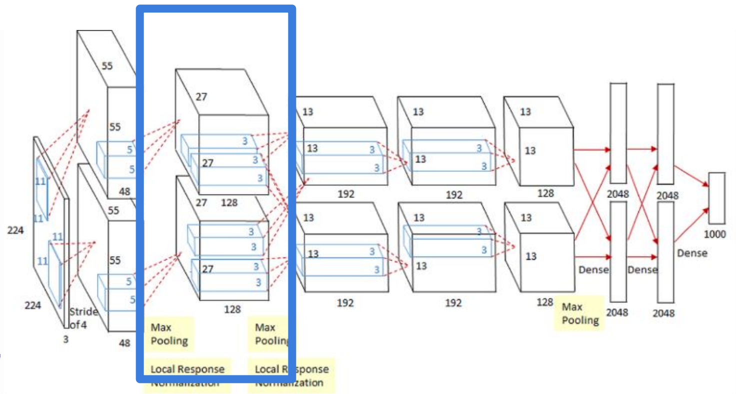
LRN (Local Response Normalization)

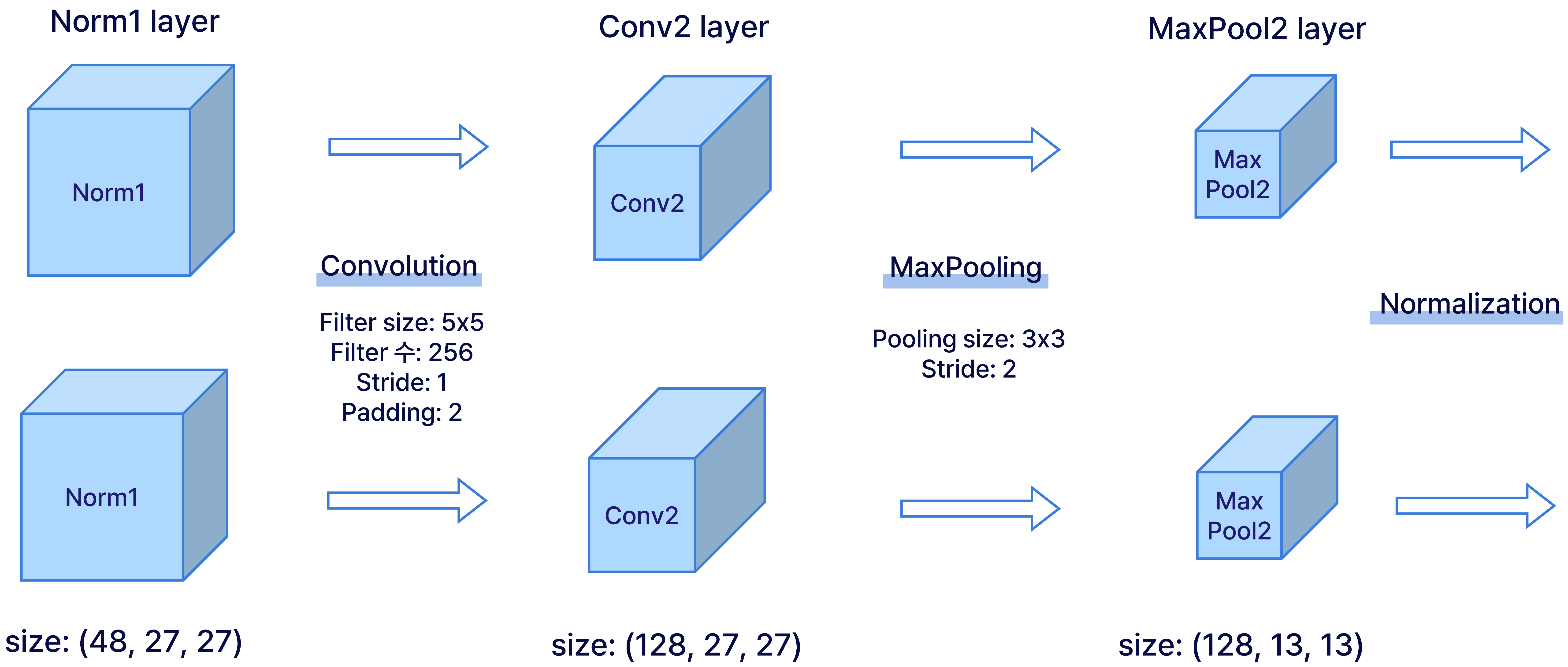


- 신경생물학에서 원리를 가져온 것으로, 우측 그림의 검은 부분을 집중하여 보면 회색의 점이 보인다.
  → 강한 자극인 검정색이 약한 자극인 흰색의 인식을 막아 발생하는 '측면억제' 현상
- ReLU는 양수의 방향으로 입력의 값을 그대로 사용하기 때문에
  매우 높은 하나의 픽셀값이 주변의 픽셀에 영향을 미치게 될 수 있다.

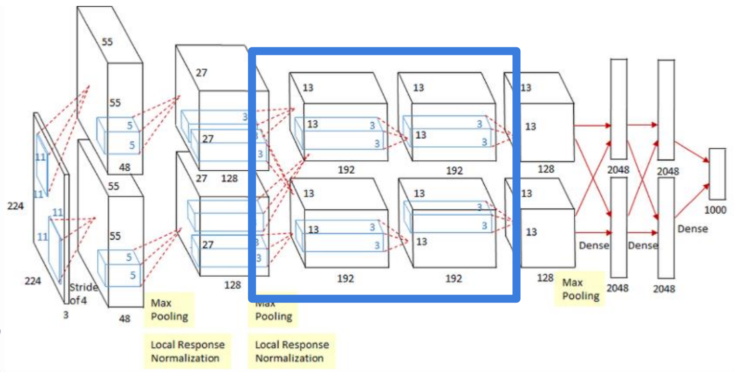
  → map의 같은 위치에 있는 pixel끼리 정규화

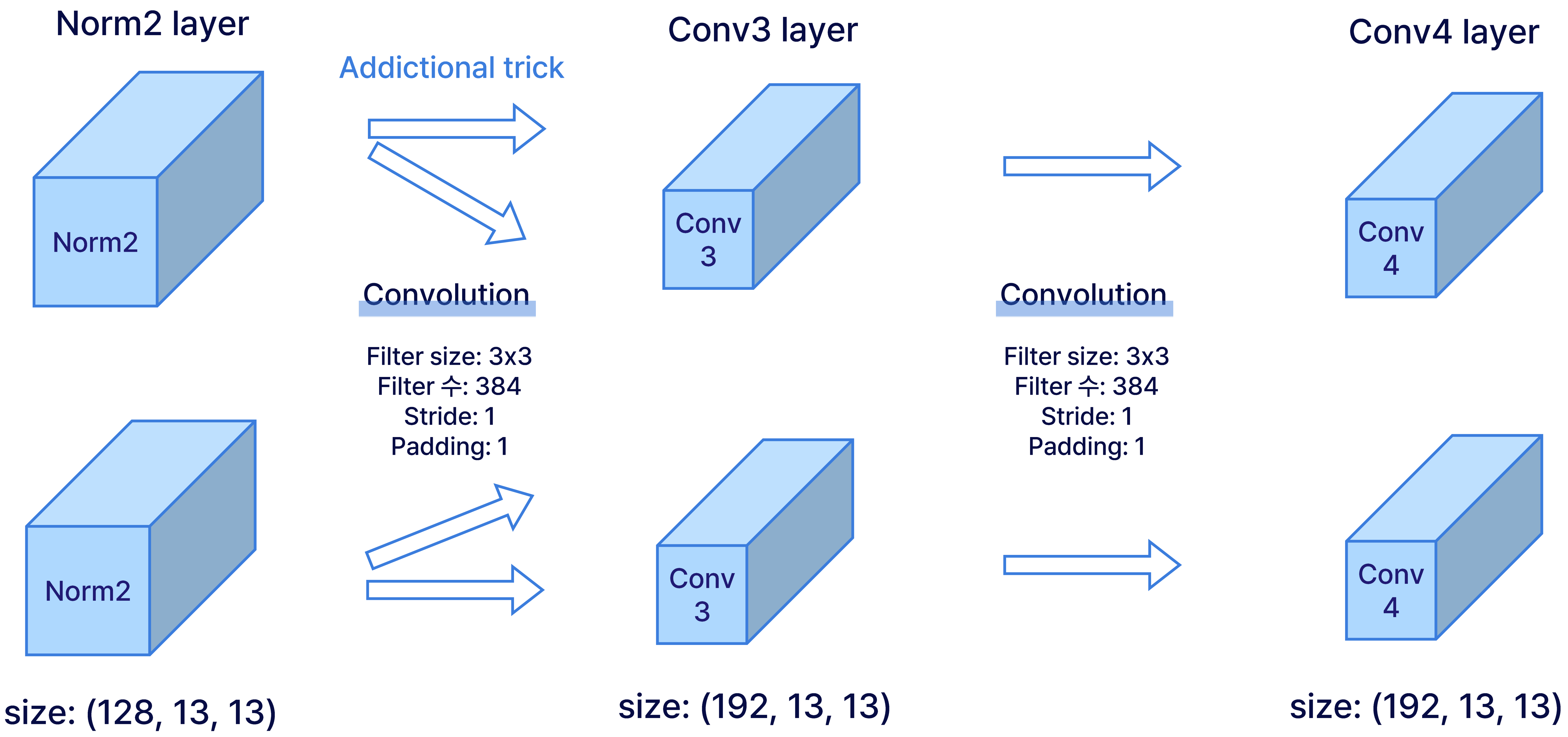
3. AlexNet – Conv2, MaxPool2



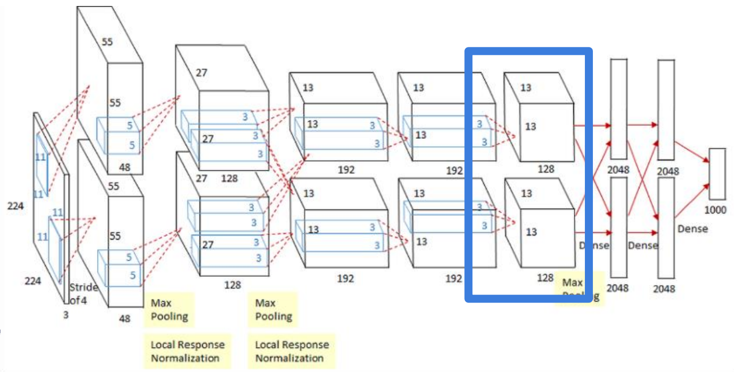


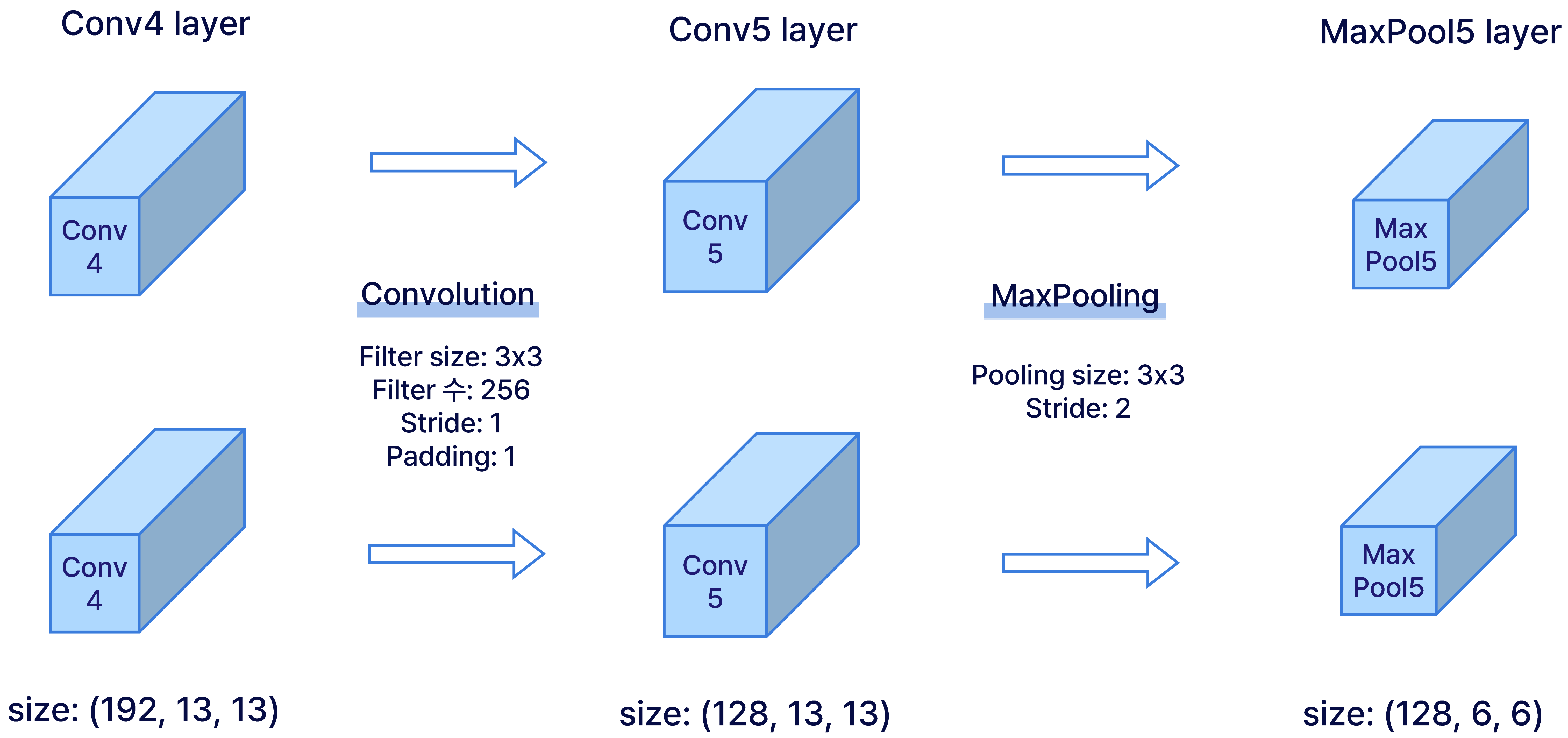
3. AlexNet – Conv3, Conv4



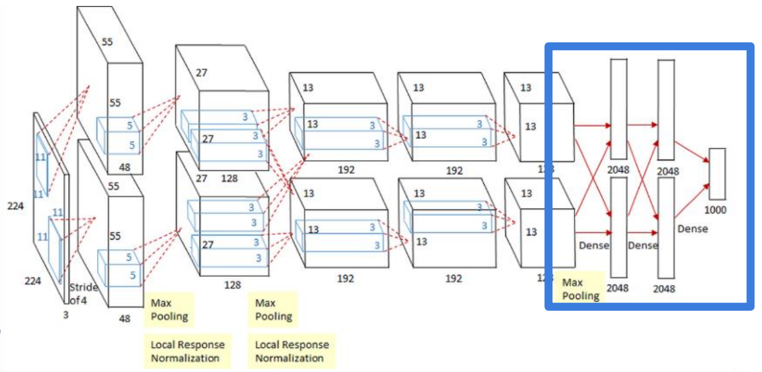


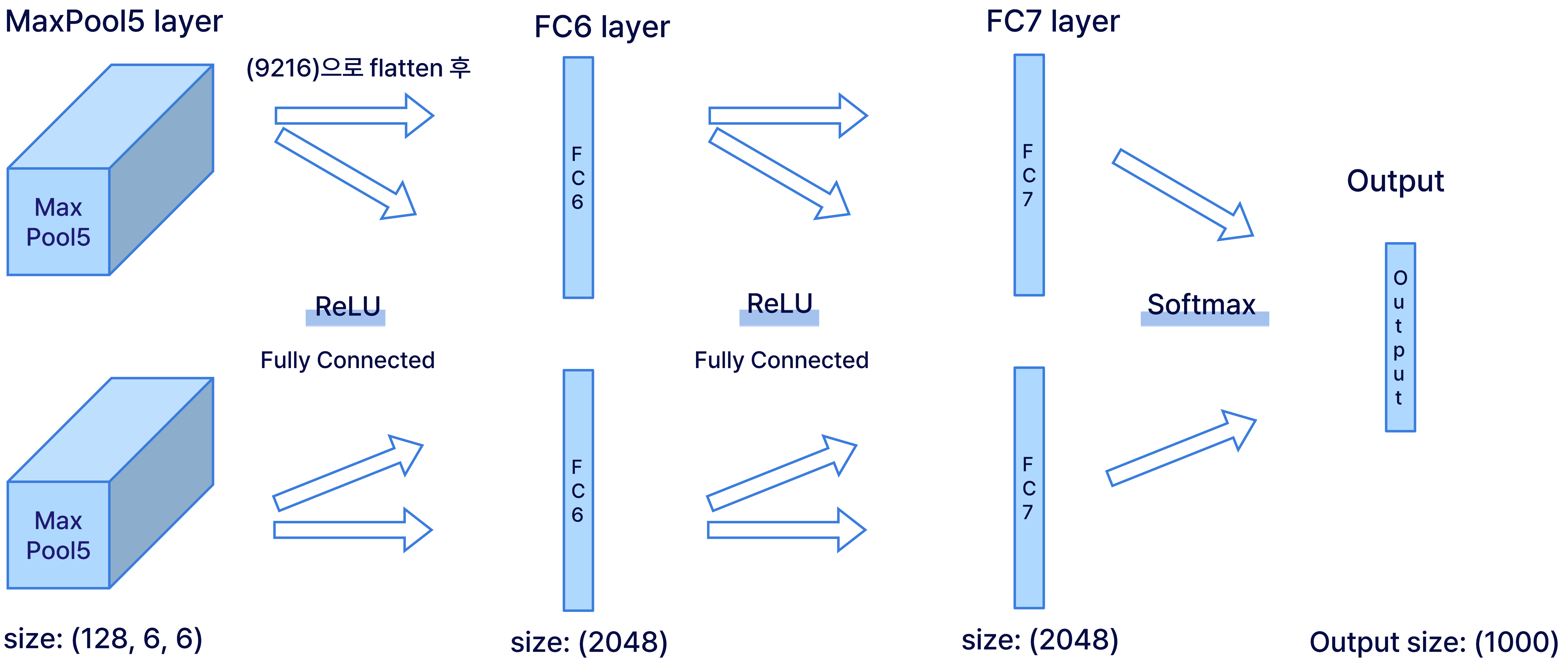
3. AlexNet – Conv5, MaxPool5





3. AlexNet – FC6, FC7, FC8



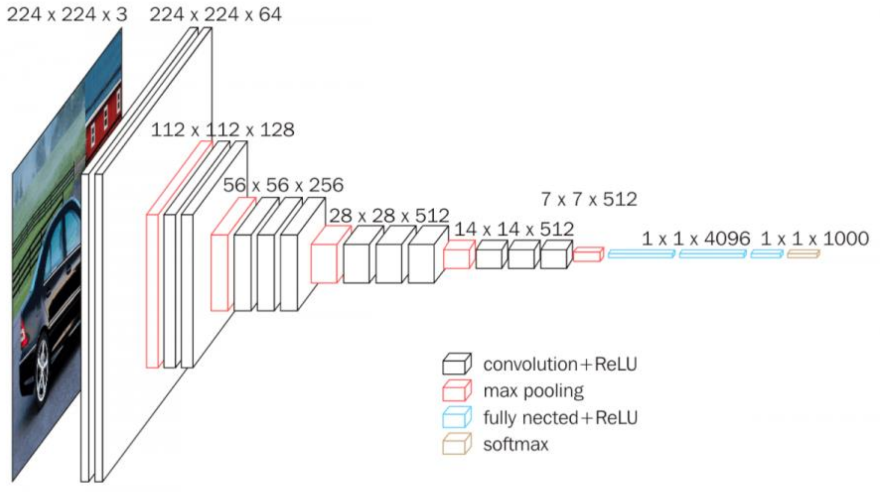


4. VGG

개요

- 2014년 ILSVRC 대회에서 2위를 한 CNN 모델
- 논문 'Very deep convolutional networks for large-scale image recognition'에 수록
- VGG부터 네트워크의 깊이가 확 깊어졌다.

- 3x3의 작은 filter를 사용한 convolution layer를 깊게 중첩한다는 것이 가장 큰 특징
- [Input – C1 – C2 – MaxPool2 – C3 – C4 – MaxPool4 – C5 – C6 – C7 – MaxPool7 – C8 – C9 - C10 – MaxPool10 – C11 – C12 – C13 – MaxPool13 – FC14 - FC15 – FC16(Output)



4. VGG

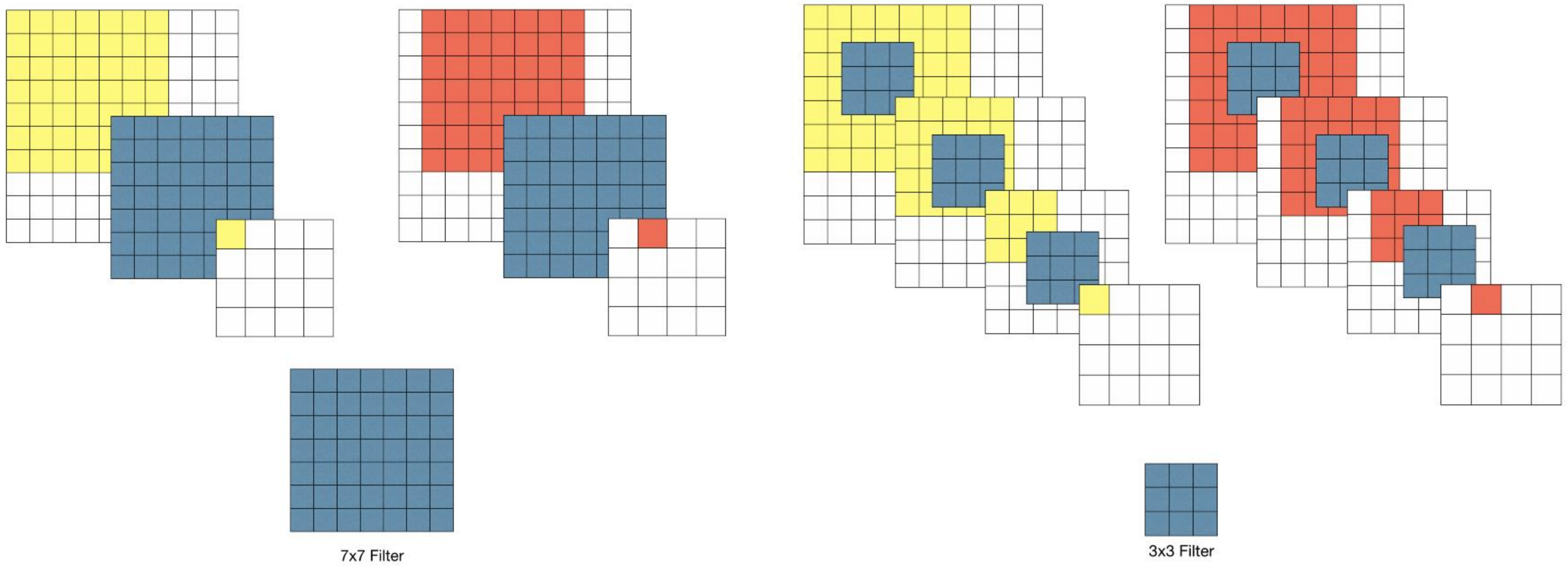
3x3 filter 사용

깊이의 영향만을 최대한 확인하고자 filter size를 3x3으로 고정
- filter 사이즈가 크면 금방 이미지 사이즈가 작아져서 깊게 만들기 어려움

1. 비선형성 증가

- 각 Convolution 연산은 ReLU 함수를 포함
→ layer가 증가함에 따라 비선형성 증가

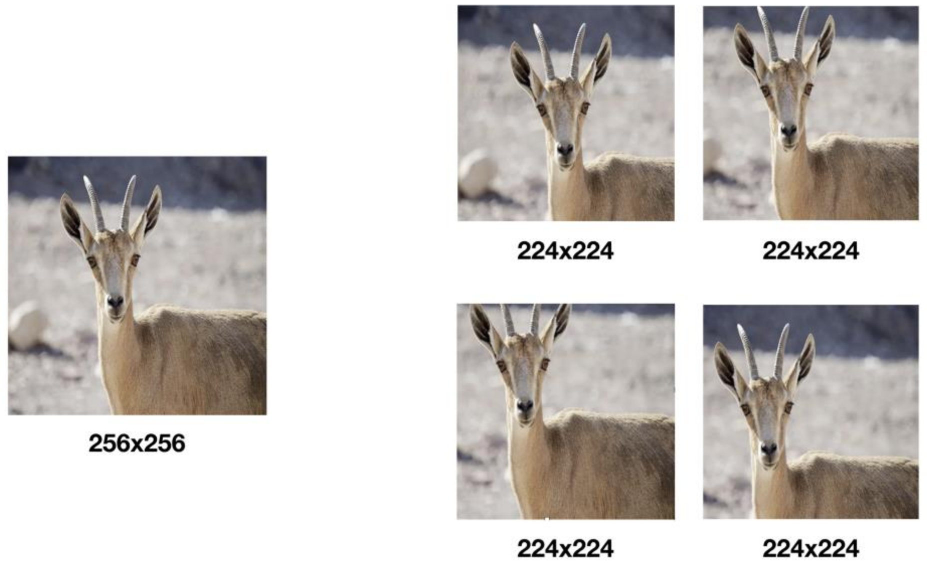
2. 학습 파라미터 수 감소



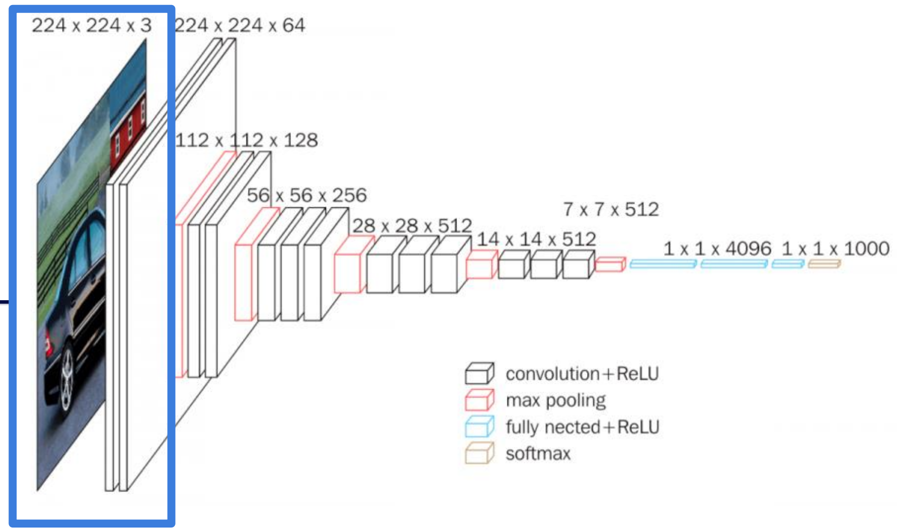
4. VGG

Data Augmentation

256x256 ~ 512x512 중 임의로 scaling 후 224x224로 crop



4. VGG – Conv1, Conv2, MaxPool2



Input

Input

Input size: (3, 224, 224)

Convolution

Filter size: 3x3
Filter 수: 64
Stride: 1
Padding: 1

Conv1 layer

Conv1

size: (64, 224, 224)

Convolution

Filter size: 3x3
Filter 수: 64
Stride: 1
Padding: 1

C2 layer

Conv2

size: (64, 224, 224)

MaxPooling

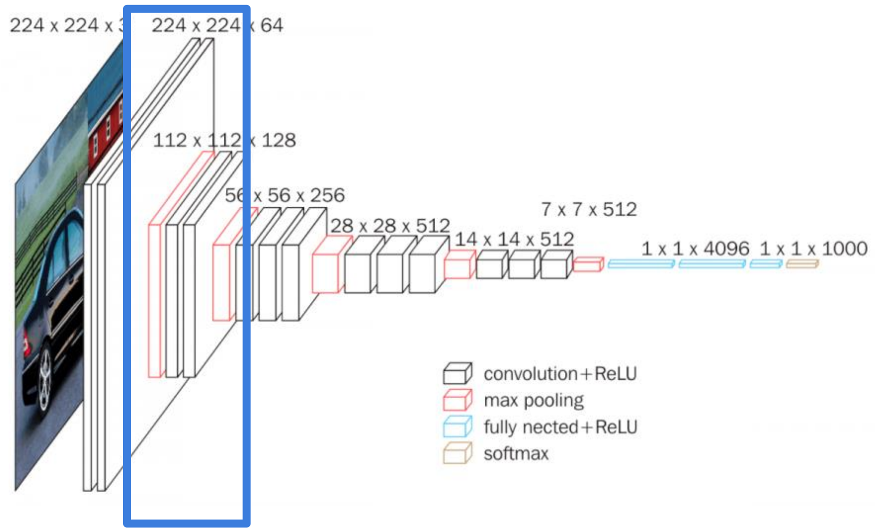
Pooling size: 2x2
Stride: 2

MaxPool2 layer

MaxPool
2

size: (64, 112, 112)

4. VGG – Conv3, Conv4, MaxPool4



MaxPool2 layer

MaxPool2

Output size: (64, 112, 112)

Convolution

Filter size: 3x3
Filter 수: 128
Stride: 1
Padding: 1

Conv3 layer

Conv3

(128, 112, 112)

Convolution

Filter size: 3x3
Filter 수: 128
Stride: 1
Padding: 1

Conv4 layer

Conv4

size: (128, 112, 112)

MaxPooling

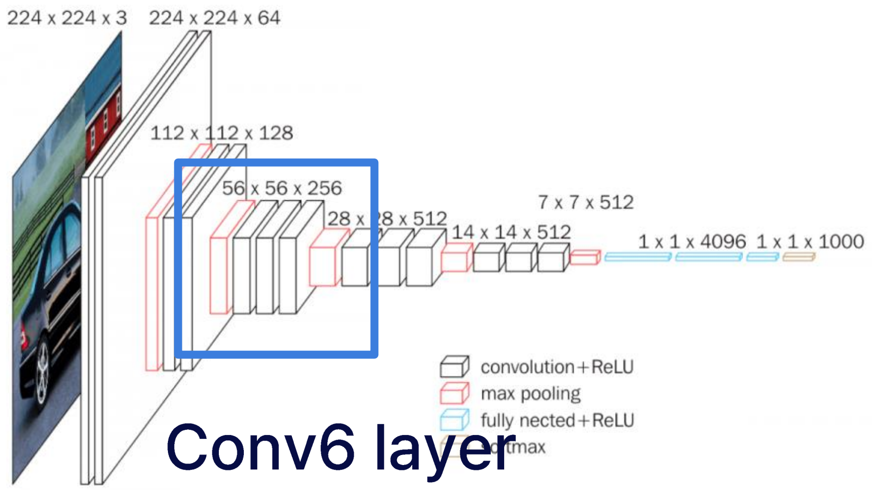
Pooling size: 2x2
Stride: 2

MaxPool4 layer

Max Pool4

size: (128, 56, 56)

4. VGG – Conv5, Conv6, Conv7, MaxPool7



MaxPool4 layer

MaxPool4

size: (128, 56, 56)

Convolution

Filter size: 3x3
Filter 수: 256
Stride: 1
Padding: 1

Conv5 layer

Conv5

size: (256, 56, 56)

Convolution

Filter size: 3x3
Filter 수: 256
Stride: 1
Padding: 1

Conv6 layer

Conv6

size: (256, 56, 56)

Convolution

Filter size: 3x3
Filter 수: 256
Stride: 1
Padding: 1

Conv7 layer

Conv7

size: (256, 56, 56)

MaxPooling

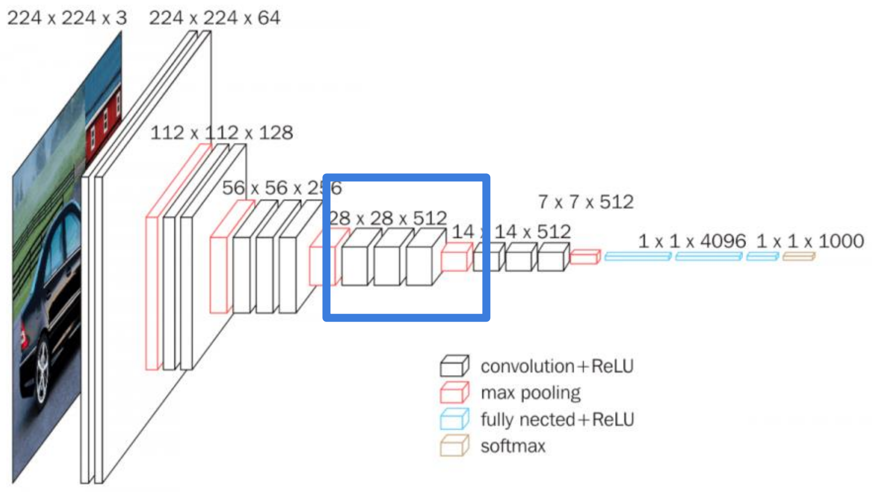
Pooling size: 2x2
Stride: 2

MaxPool7 layer

Max Pool7

size: (256, 28, 28)

4. VGG – Conv8, Conv9, Conv10, MaxPool10



MaxPool7 layer

Max Pool7

size: (256, 28, 28)

Conv8 layer   Conv9 layer

Convolution

Filter size: 3x3
Filter 수: 512
Stride: 1
Padding: 1

Conv10 layer

Conv 10

size: (512, 28, 28)

MaxPooling

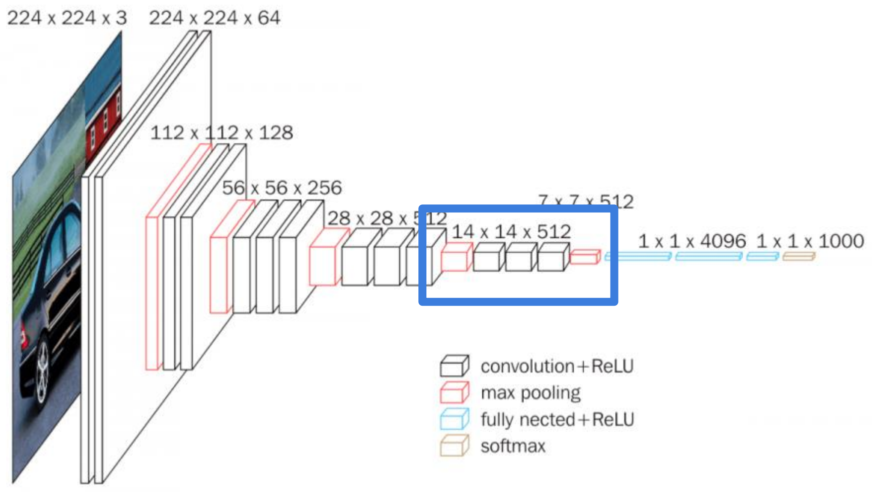
Pooling size: 2x2
Stride: 2

MaxPool10 layer

Max Pool10

size: (512, 14, 14)

4. VGG – Conv11, Conv12, Conv13, MaxPool13



MaxPool10 layer

Max Pool10

size: (512, 14, 14)

Conv11 layer　Conv12 layer

Convolution

Filter size: 3x3
Filter 수: 512
Stride: 1
Padding: 1

Conv13 layer

Conv 13

size: (512, 14, 14)

MaxPooling

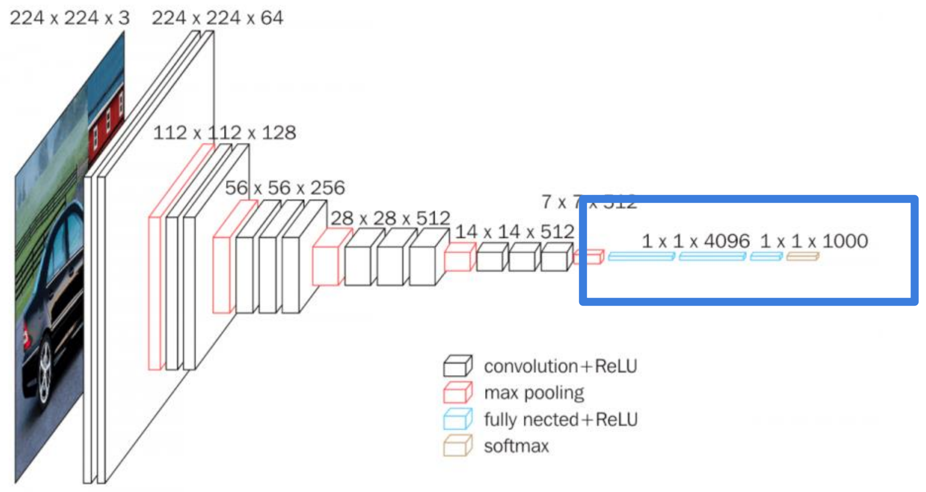
Pooling size: 2x2
Stride: 2

MaxPool13 layer

Max Pool13

size: (512, 7, 7)

4. VGG – FC14, FC15, FC16



MaxPool13 layer

Max Pool 13

size: (512, 7, 7)

(25088)으로 flatten 후

ReLU

Fully Connected

FC14 layer

FC14

size: (4096)

ReLU

Fully Connected

FC15 layer

FC15

size: (4096)

Softmax

Fully Connected

Output

Output

Output size: (1000)

과제

1. AlexNet 주석 달기
2. VGG 논문 review

2023 D&A

Deep Session 5차시

THANK YOU

2023. 04. 06