15기 정규세션

ToBig's 14기 이정은

CNN 기초

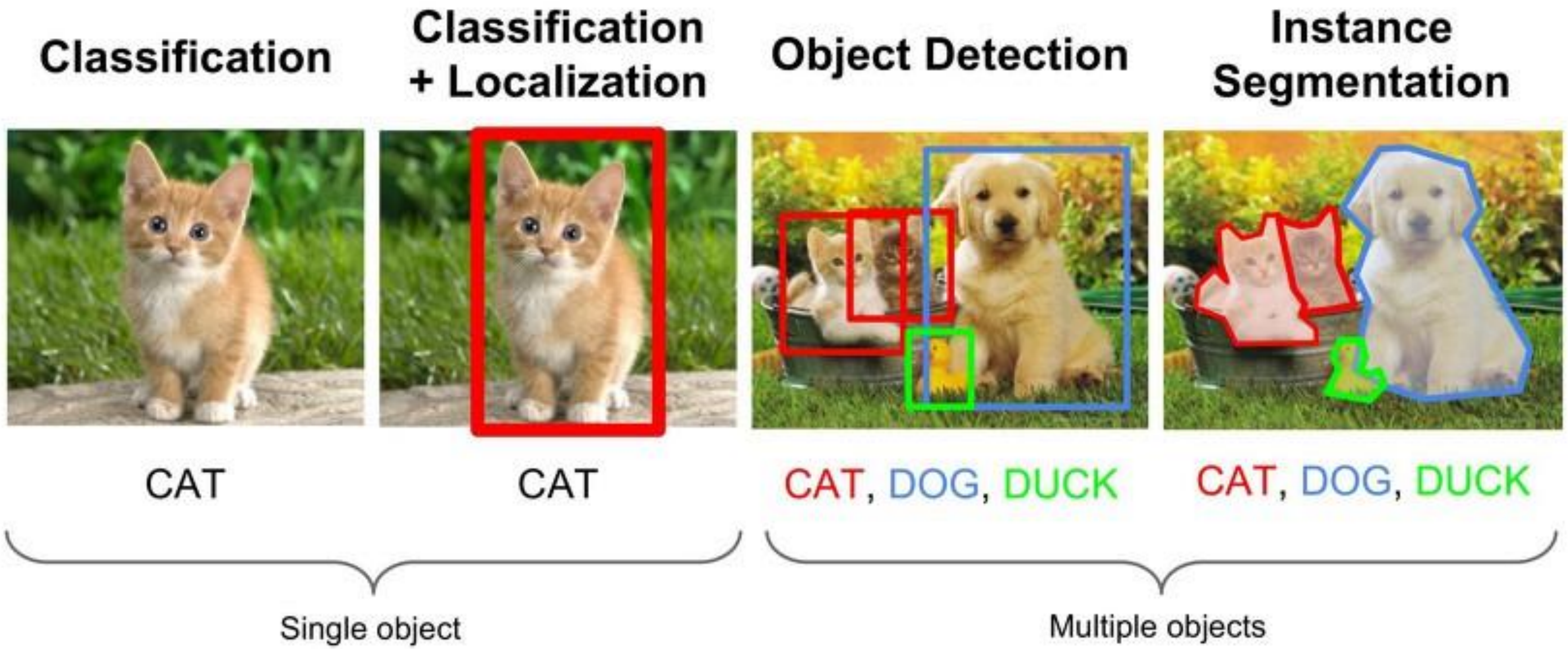
Convolutional Neural Network

Contents

Unit 01 | Intro

https://machinelearningmastery.com/what-is-computer-vision/

컴퓨터 비전 분야의 딥러닝



https://machinelearningmastery.com/applications-of-deep-learning-for-computer-vision/

컴퓨터 비전 분야의 딥러닝

Style Transfer

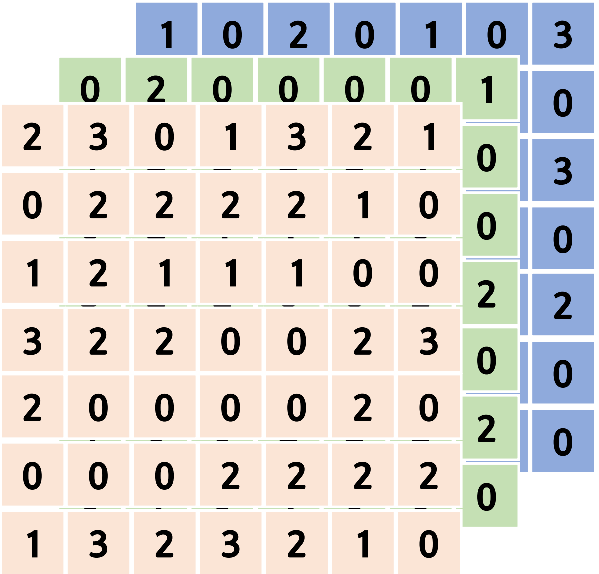
Image Synthesis

Super Resolution

Pose Estimation

Unit 01 | Intro

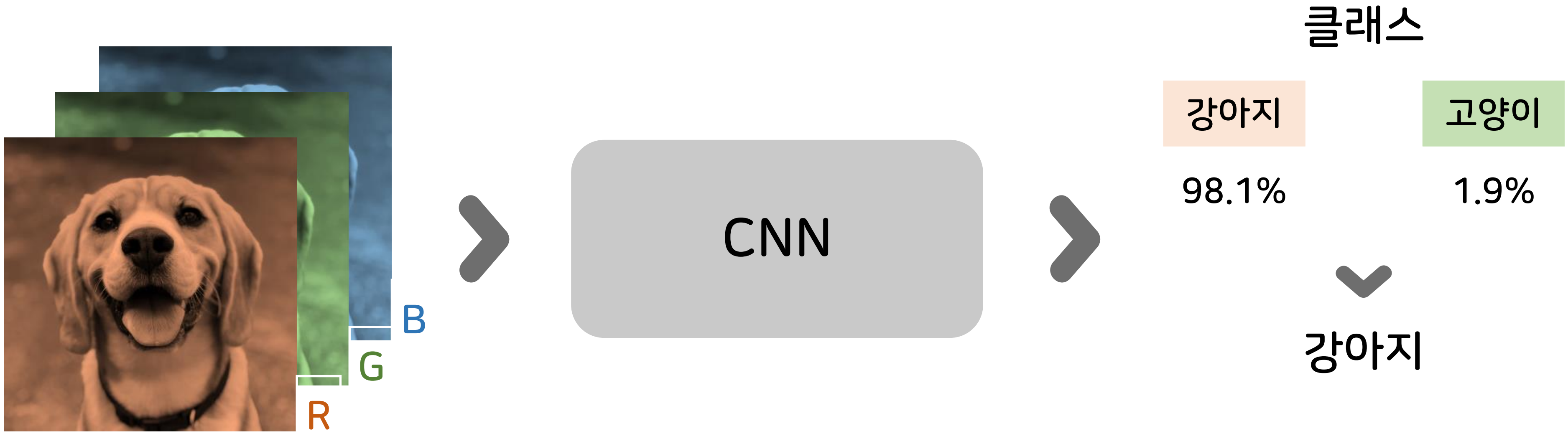
컴퓨터는 이미지를 어떻게 읽을까?



이미지는 픽셀로 이루어져 있다.
컬러 이미지의 경우,
각 픽셀마다 RGB에 해당하는 3개의 값을 가진다.

Unit 01 | Intro

컴퓨터는 이미지를 어떻게 읽을까?

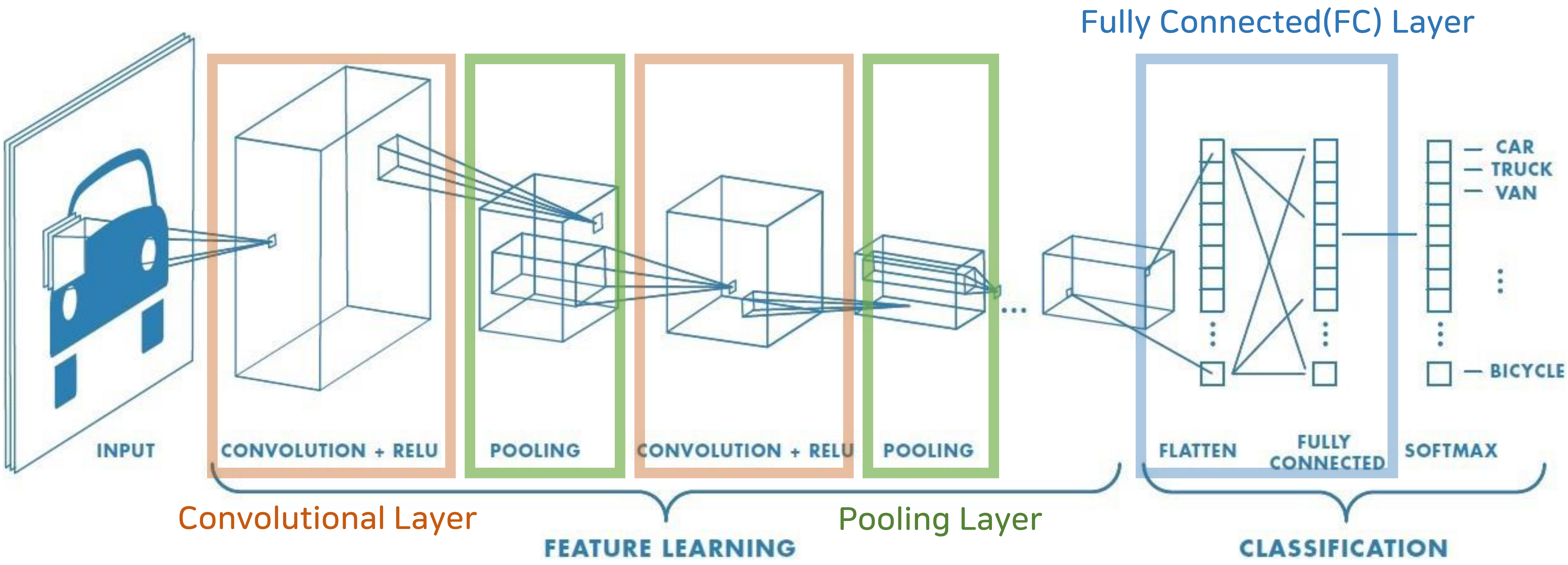


CNN
(Convolutional Neural Network)

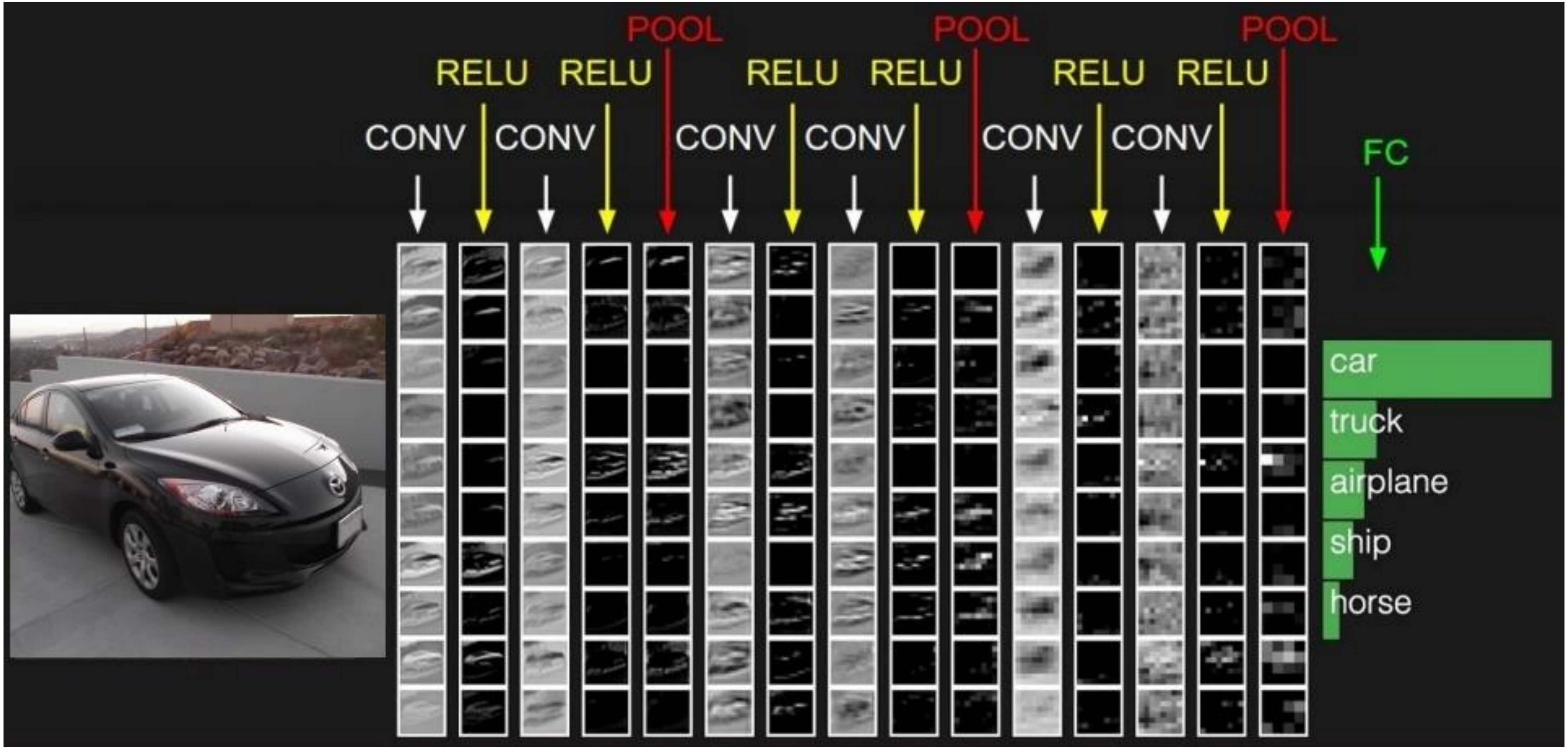
Summary

- Convolutional Layer
- Channel
- Filter
- Stride
- Padding
- Pooling Layer

Unit 02 | CNN


Fully Connected(FC) Layer
INPUT
CONVOLUTION + RELU
POOLING
CONVOLUTION + RELU
POOLING
FLATTEN
FULLY CONNECTED
SOFTMAX
CAR
TRUCK
VAN
BICYCLE
Convolutional Layer
Pooling Layer
FEATURE LEARNING
CLASSIFICATION

Unit 02 | CNN


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

계층적 패턴 인식 (hierarchical pattern recognition)

Input

Output

경계선, 모서리, 가장자리 등
기초엣지

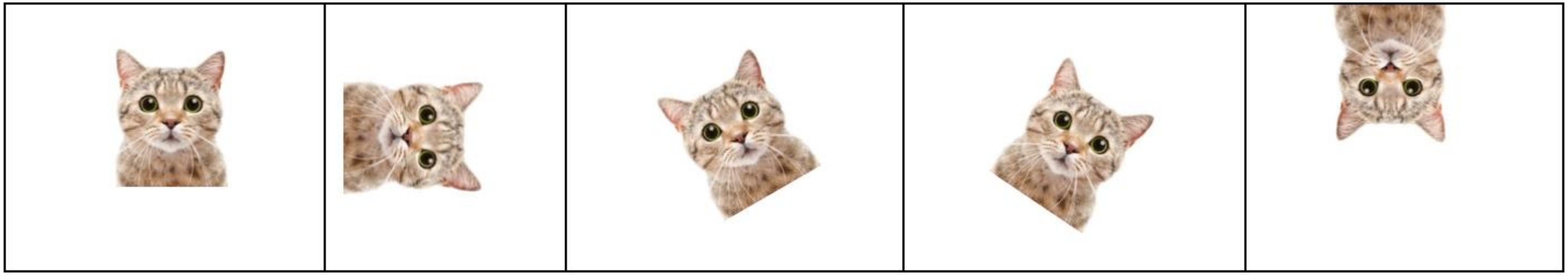
기초엣지를 이용하여
귀, 눈, 입 등 패턴 인식

사람의 얼굴처럼
전체적인 구조 인식

Local Connectivity: receptive field와 유사하게 local 정보를 활용한다.

지역적으로 뉴런을 연결하여 다양한 local feature 추출이 가능하다.

위치이동불변 (translation invariant)



이미지 내에서의 위치와 관계 없이 동일한 패턴이면 동일하게 인식한다.
Shared Weights and Biases (topology invariance)
: 각 Filter는 모든 이미지에 대해 동일하게 사용된다.

Convolutional Layer

Convolution

W

H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

1	0	1
0	1	0
1	0	1

???? (3x3)

Output (?x?)

Convolution

W

H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

1	0	1
0	1	0
1	0	1

???? (3x3)

6

Output (?x?)

Convolution

W

H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

1	0	1
0	1	0
1	0	1

???? (3x3)

6	9

Output (?x?)

Convolution

W

H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

1	0	1
0	1	0
1	0	1

???? (3x3)

6	9	7

Output (?x?)

Convolution

W / H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

1	0	1
0	1	0
1	0	1

???? (3x3)

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

Output (?x?)

Convolution

0	2	0
2	2	2
2	1	0

·

1	0	1
0	1	0
1	0	1

= 4

0*1 + 2*0 + ⋯ + 0*1

Convolution(합성곱) 연산은 벡터 간의 내적을 구하는 연산이다.

Convolution

W

H

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Feature Map (7x7)

1	0	1
0	1	0
1	0	1

Filter (3x3)

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

H W

Output Feature Map (5x5)

Filter

1	0	1
0	1	0
1	0	1

w_{11}	w_{12}	w_{13}
w_{21}	w_{22}	w_{23}
w_{31}	w_{32}	w_{33}

Filter 혹은 Kernel
실제로 얘는 가중치(weight)
-> 즉, 업데이트가 이루어진다.

Bias 개념도 추가할 수 있다.
if) bias가 1이라면?

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

Unit 03 | Convolutional Layer

Filter

Input

$$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

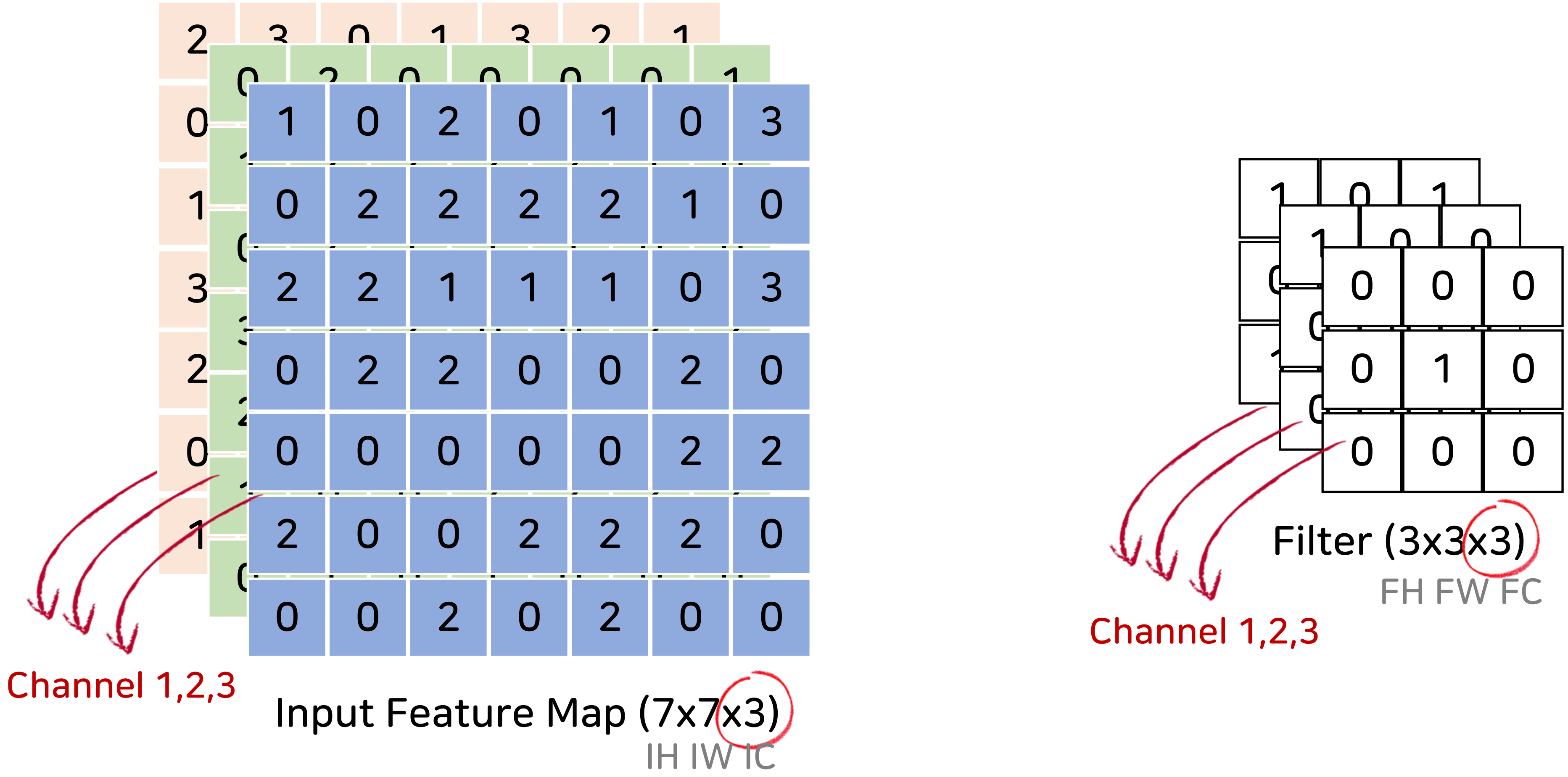
$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Filter는 이미지의 특징을 찾아내기 위한 Shared Weights이다.

Convolution 연산을 진행하면 나오는 Output Feature Map에서는

해당 Filter가 표현하는 특징이 존재하는 부분에서 높은 값을 가진다.

Channel


Channel 1,2,3
Input Feature Map (7x7x3)
IH IW IC
Filter (3x3x3)
FH FW FC
Channel 1,2,3

Unit 03 | Convolutional Layer

Channel 1

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input Feature Map (7x7)

1	0	1
0	1	0
1	0	1

Filter (3x3)

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

Result1 (5x5)

Channel 2

0	2	0	0	0	0	1
1	2	2	2	2	1	0
0	2	1	1	1	0	0
3	2	2	0	0	2	2
2	0	3	0	0	2	0
1	0	0	2	2	2	2
0	1	0	1	0	0	0

Input Feature Map (7x7)

1	0	0
0	0	0
0	0	0

Filter (3x3)

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

Result2 (5x5)

Channel 3

1	0	2	0	1	0	3
0	2	2	2	2	1	0
2	2	1	1	1	0	3
0	2	2	0	0	2	0
0	0	0	0	0	2	2
2	0	0	2	2	2	0
0	0	2	0	2	0	0

Input Feature Map (7x7)

0	0	0
0	1	0
0	0	0

Filter (3x3)

2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

Result3 (5x5)

Channel

다 더하면 된다!

Result1

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

+

Result2

0	2	0	0	0
1	2	2	2	2
0	2	1	1	1
3	2	2	0	0
2	0	3	0	0

+

Result3

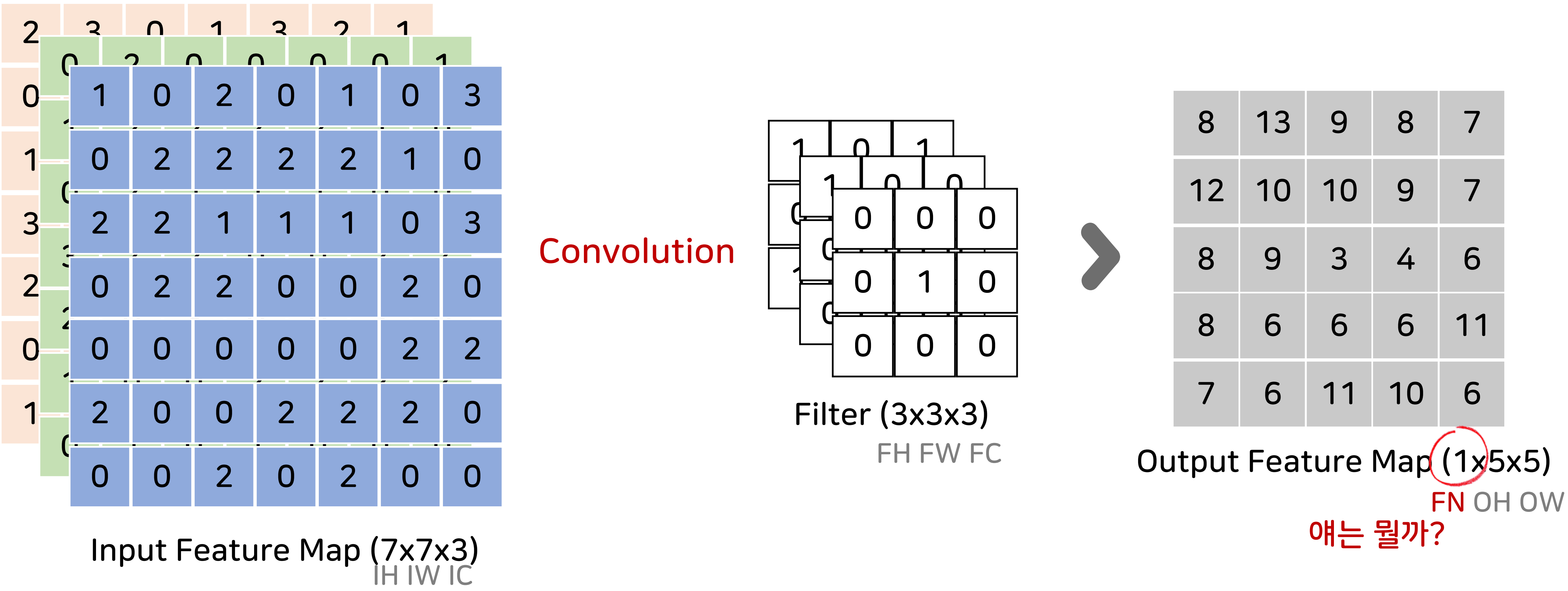
2	2	2	2	1
2	1	1	1	0
2	2	0	0	2
0	0	0	0	2
0	0	2	2	2

=

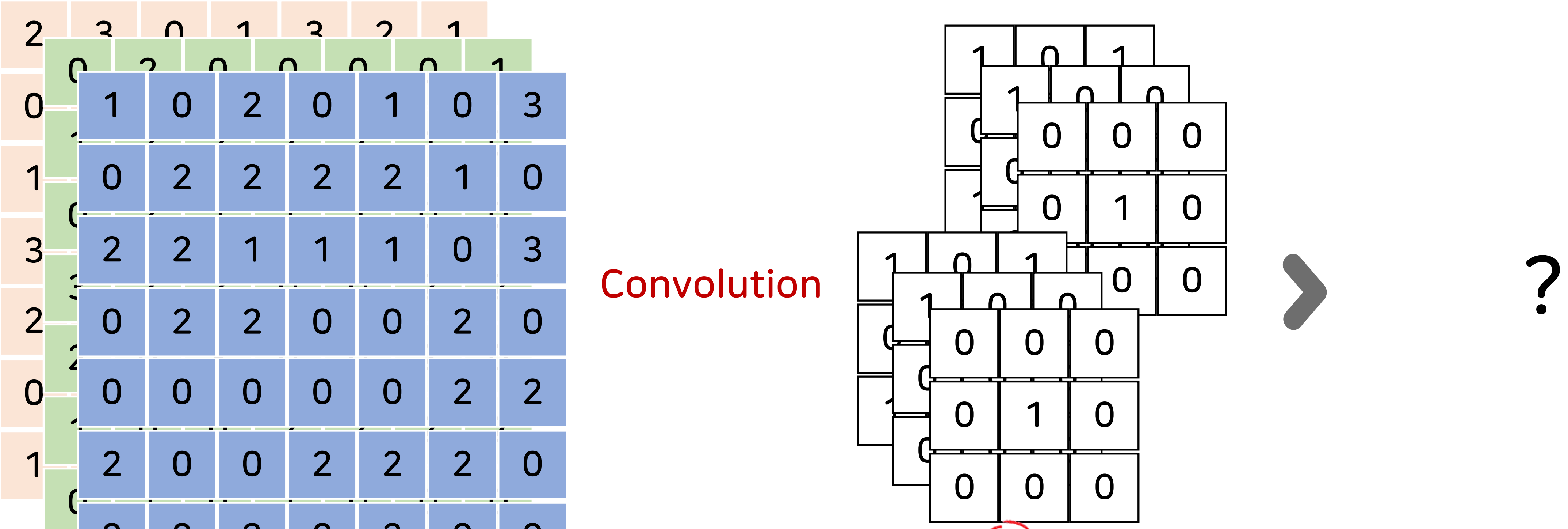
Output Feature Map

8	13	9	8	7
12	10	10	9	7
8	9	3	4	6
8	6	6	6	11
7	6	11	10	6

Channel


1 0 2 0 1 0 3
0 2 2 2 2 1 0
2 2 1 1 1 0 3
0 2 2 0 0 2 0
0 0 0 0 0 2 2
2 0 0 2 2 2 0
0 0 2 0 2 0 0
Input Feature Map (7x7x3)
IH IW IC
Convolution
0 0 0
0 1 0
0 0 0
Filter (3x3x3)
FH FW FC
8 13 9 8 7
12 10 10 9 7
8 9 3 4 6
8 6 6 6 11
7 6 11 10 6
Output Feature Map (1x5x5)
FN OH OW
얘는 뭘까?

Filter



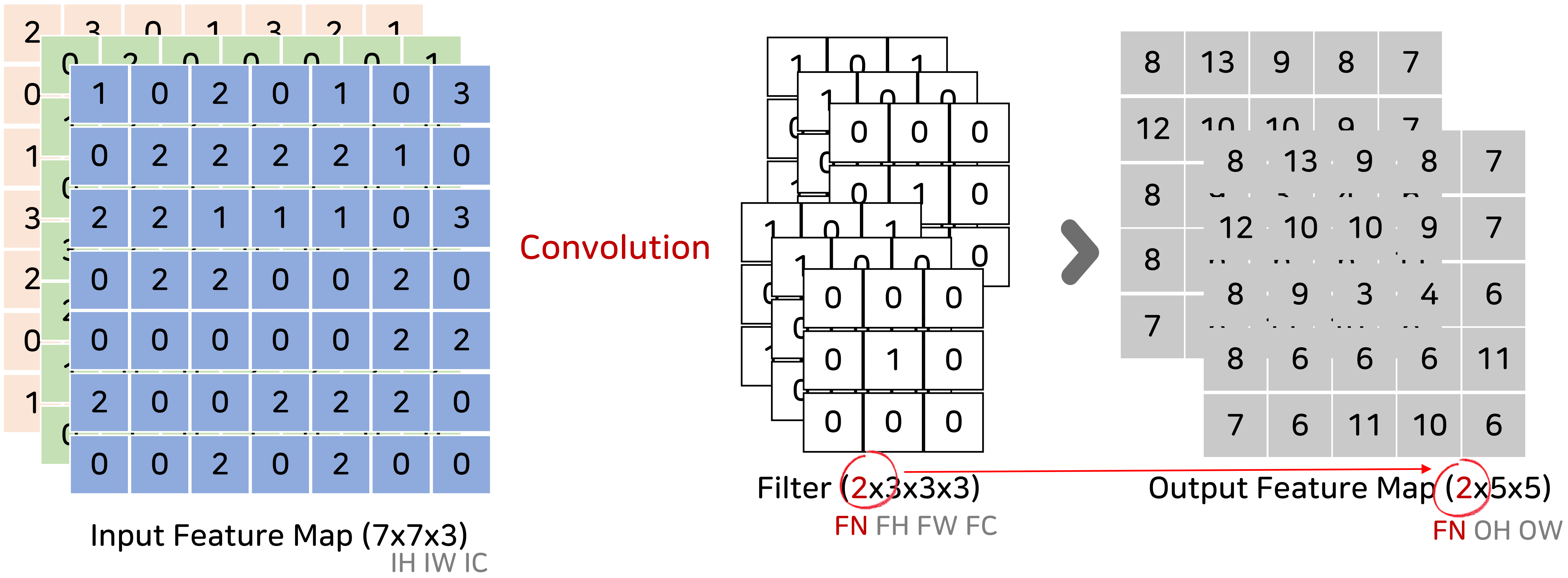
Input Feature Map (7x7x3)
IH IW IC

Filter (2x3x3x3)
FN FH FW FC

Output Feature Map (?x5x5)
FN OH OW

Unit 03 | Convolutional Layer

Filter



Channel / Filter

- Input Feature Map의 Channel = Filter의 Channel

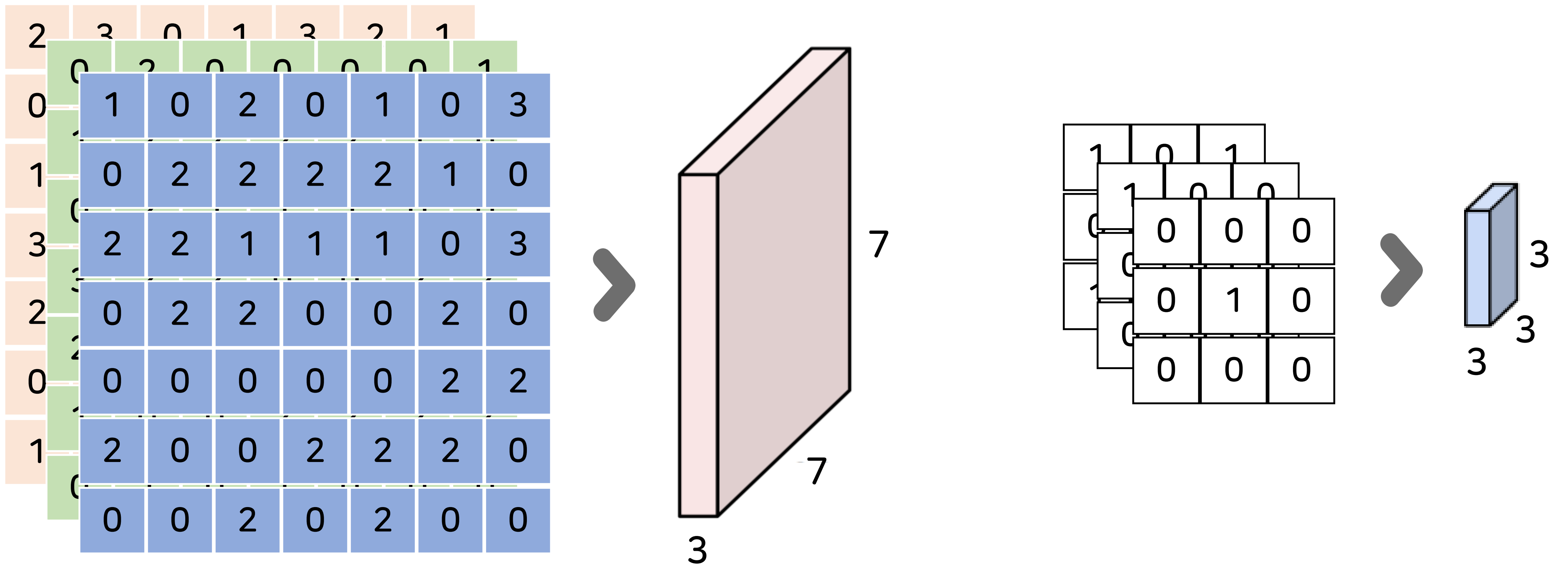
Input Feature Map (7x7x3) Filter (2x3x3x3) Output Feature Map (2x5x5)
IH IW IC FN FH FW FC FN OH OW

- Filter의 개수 = Output Feature Map의 차원

Input Feature Map (7x7x3) Filter (2x3x3x3) Output Feature Map (2x5x5)
IH IW IC FN FH FW FC FN OH OW

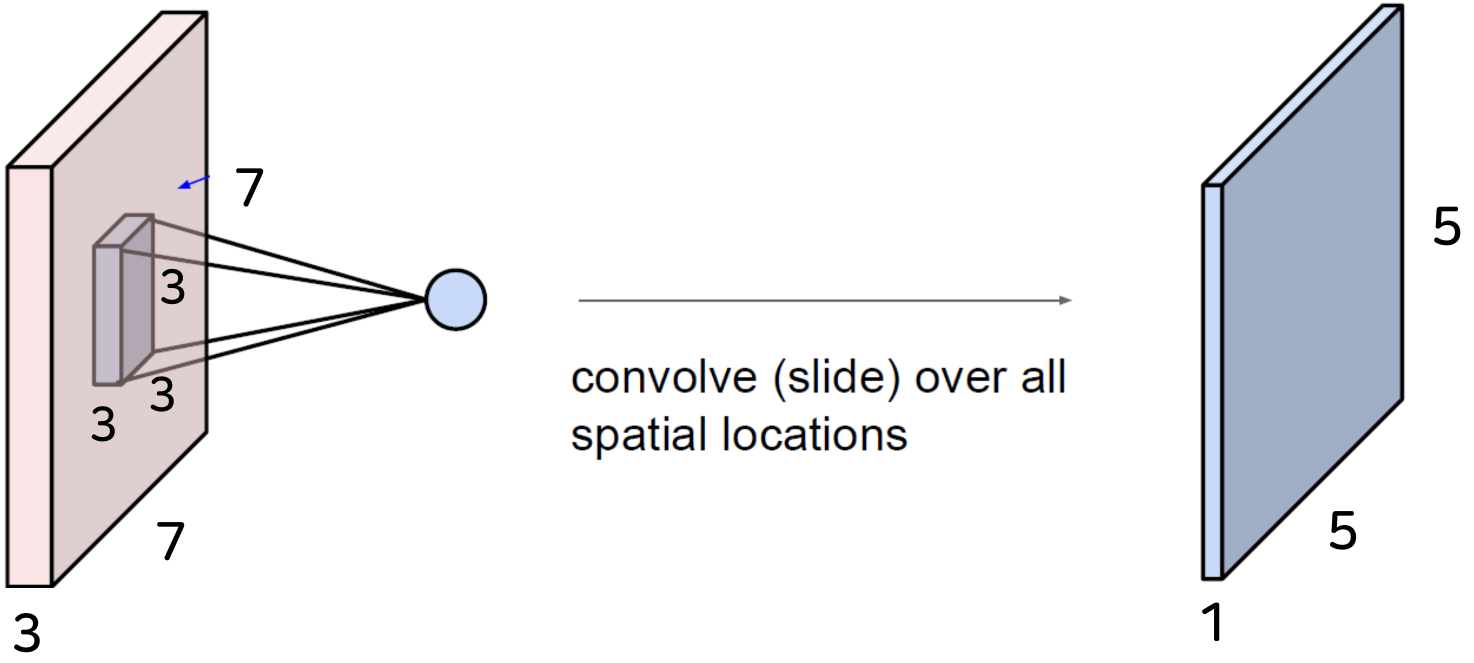
Unit 03 | Convolutional Layer

블록 형태



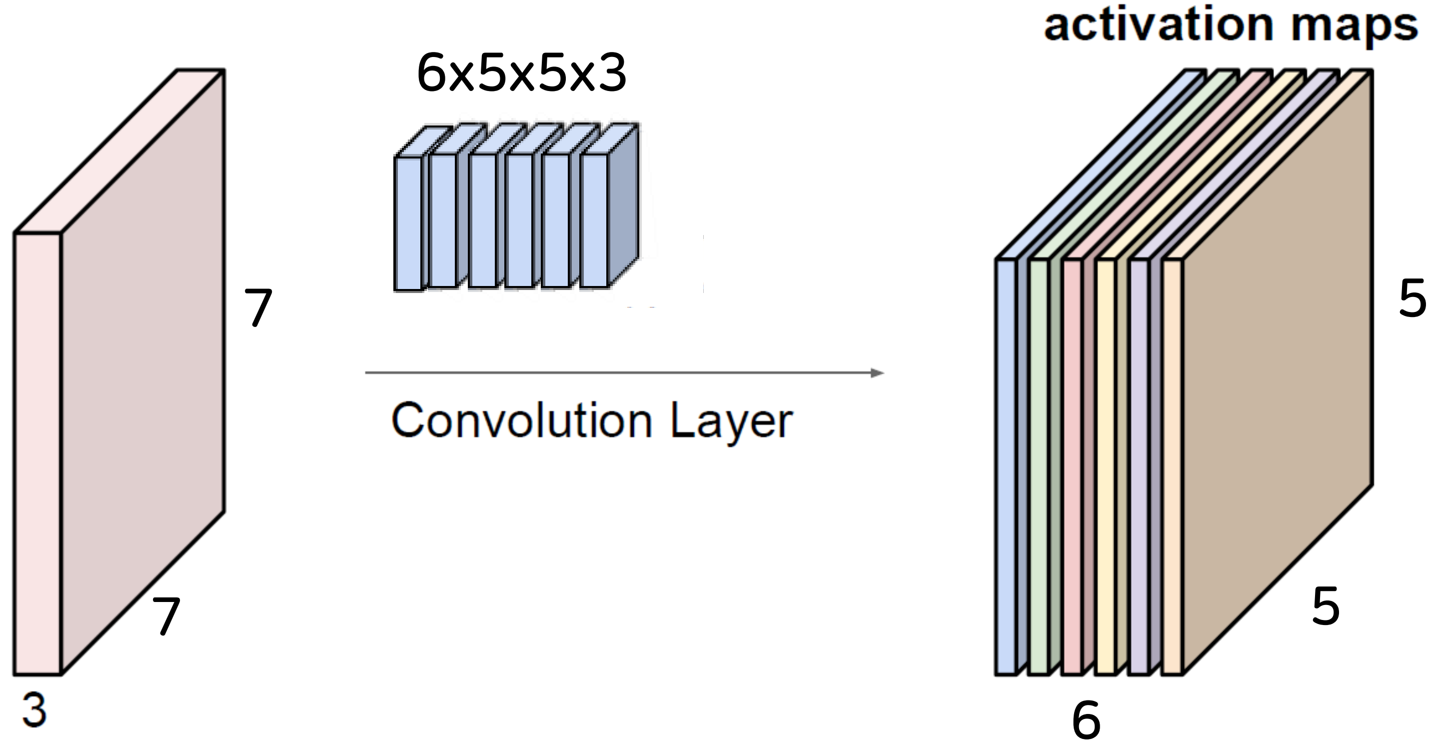
Unit 03 | Convolutional Layer

블록 형태



Unit 03 | Convolutional Layer

블록 형태



Filter가 6개 이므로 6개의 새로운 Feature map이 생성된다.

Stride (S)

Stride는 Filter가 이동하는 거리

차원을 더 빠르게 축소시킬 수 있다.

Stride (S)

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

Stride = 1일 때,

6	9	7	6	6
9	7	7	6	5
6	5	2	3	3
5	4	4	6	9
5	6	6	8	4

Output (5x5)

Stride (S)

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

Input (7x7)

Stride = 2일 때,

6	9	7
9	7	7
6	5	2

Output (3x3)

Stride (S)

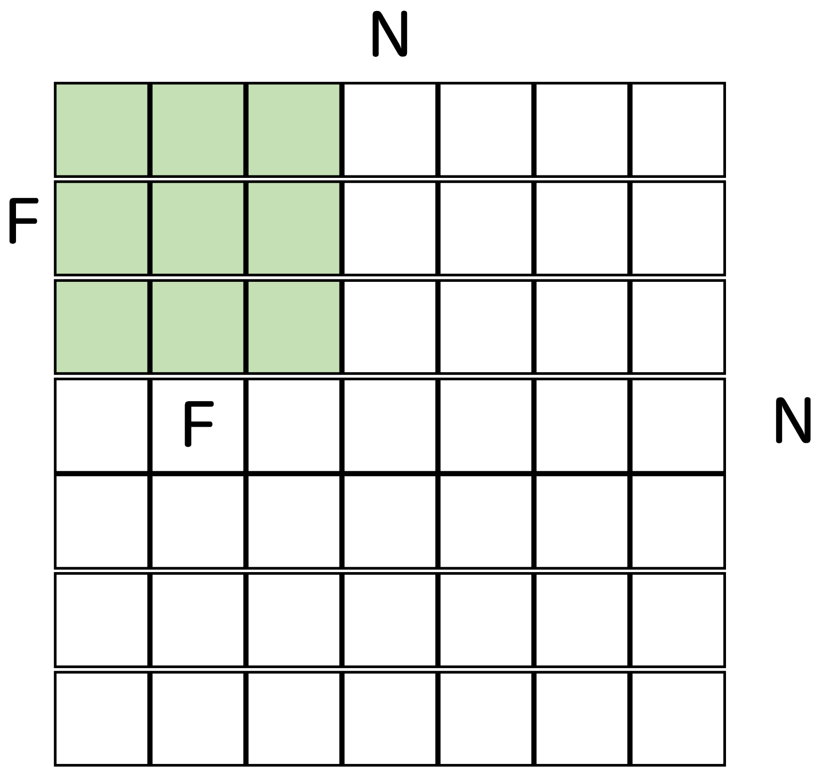
Stride = 3일 때는?

2	3	0	1	3	2	1
0	2	2	2	2	1	0
1	2	1	1	1	0	0
3	2	2	0	0	2	3
2	0	0	0	0	2	0
0	0	0	2	2	2	2
1	3	2	3	2	1	0

크기가 안 맞는다!

Input (7x7)

Stride (S)



Output Size = (N-F)/stride + 1

If) N=7, F=3
stride=1, (7-3)/1 + 1 = 5
stride=2, (7-3)/2 + 1 = 3
stride=3, (7-3)/3 + 1 = 2.33

Output Size가 integer(정수)가 되도록 stride를 설정해야 한다.

Padding (P)

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

기존 데이터의 테두리에 (zero) padding을 더한다.

크기 손실 방지 및 테두리 정보를 활용할 수 있다.

Padding (P)

0	0	0	0	0	0	0	0	0
0								0
0								0
0								0
0								0
0								0
0								0
0								0
0	0	0	0	0	0	0	0	0

Output Size = (N+2*P-F)/stride + 1

If) N=7, F=3, stride=1일 때,
P=1인 zero padding을 준다면?

(7+2*1-3)/1 + 1 = 7
즉, Input size와 Output size가 동일하다.

Input size = Output size
- F=3 & P=1
- F=5 & P=2
- F=7 & P=3

Pooling Layer

Pooling

이미지의 특성을 유지하면서 사이즈를 줄인다.

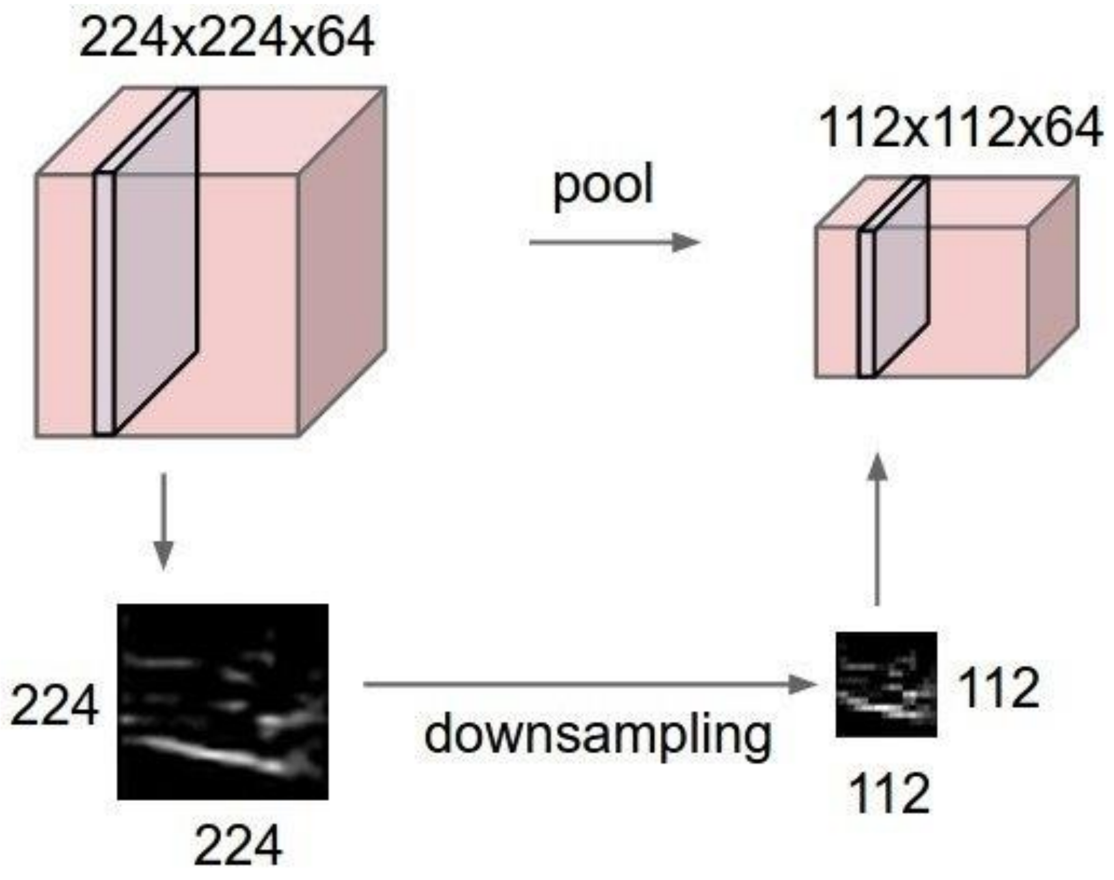
-> 불필요한 연산을 줄이고, 모수의 수를 줄여 Overfitting을 방지한다.

채널 수는 변함이 없다.

선형 결합이 아니므로 학습되는 weight가 없다.

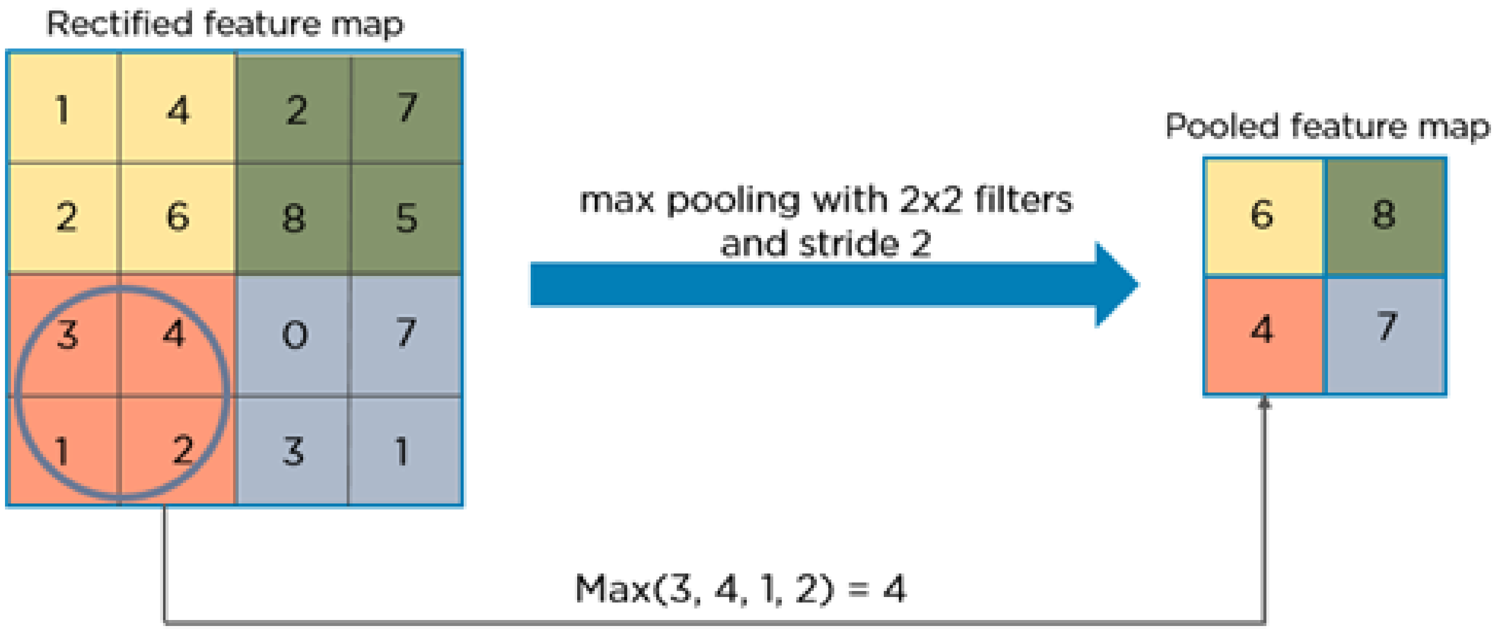
Unit 04 | Pooling Layer

Pooling



Unit 04 | Pooling Layer

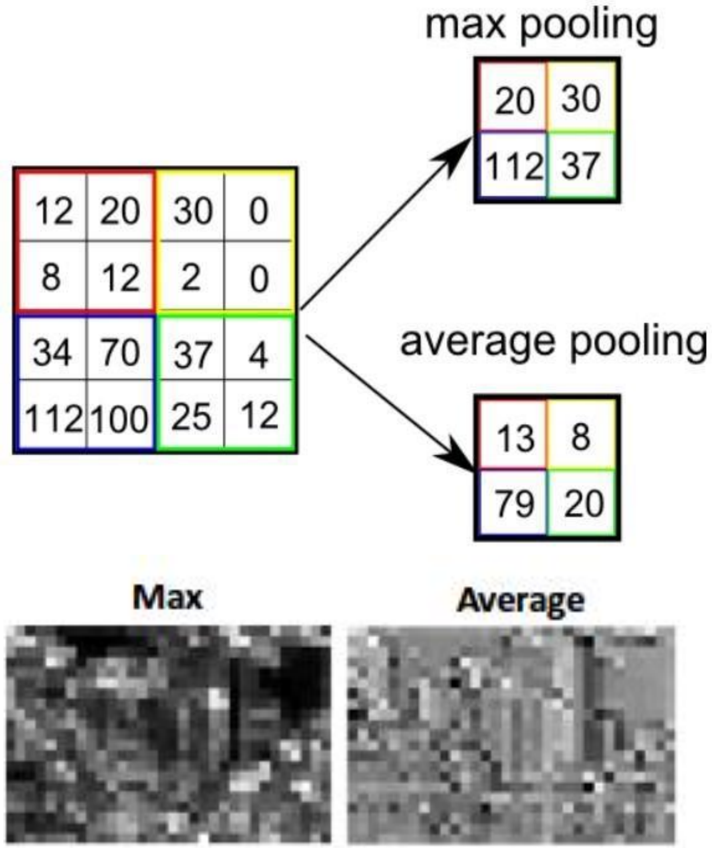
Max Pooling



Filter size(F)와 Stride(S)가 존재한다.

Unit 04 | Pooling Layer

Max Pooling vs Average Pooling



- Max Pooling

해당 window의 max값을 추출

= 밝은 픽셀 값을 선택

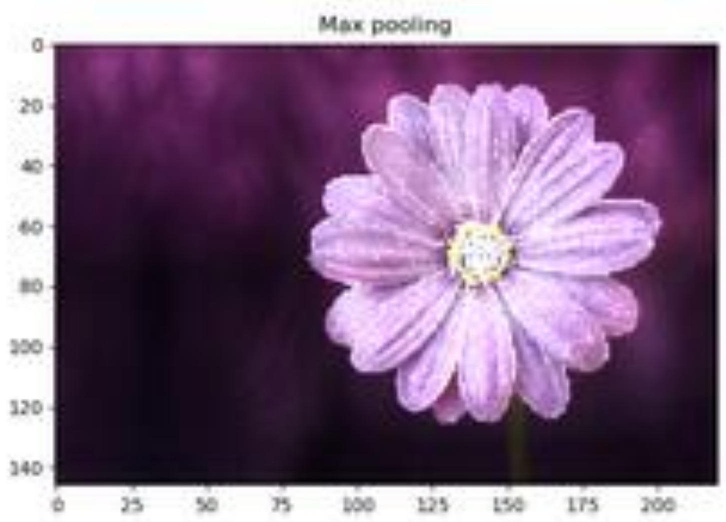
- Average Pooling

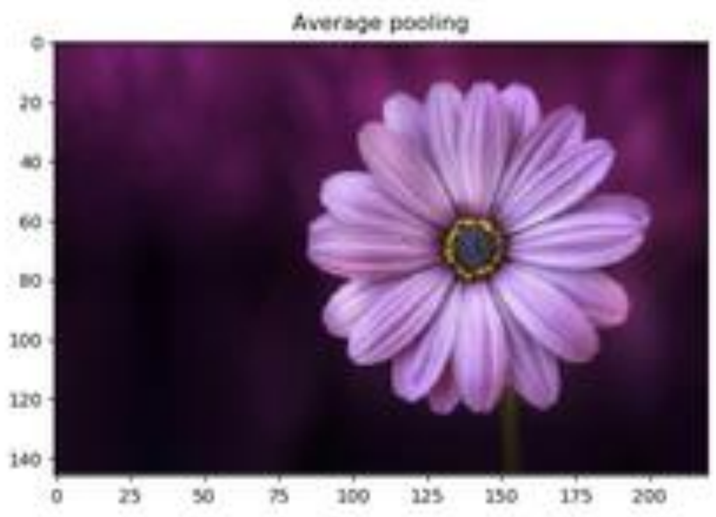
해당 window의 average 값을 추출

= smoothing

Unit 04 | Pooling Layer

Max Pooling vs Average Pooling





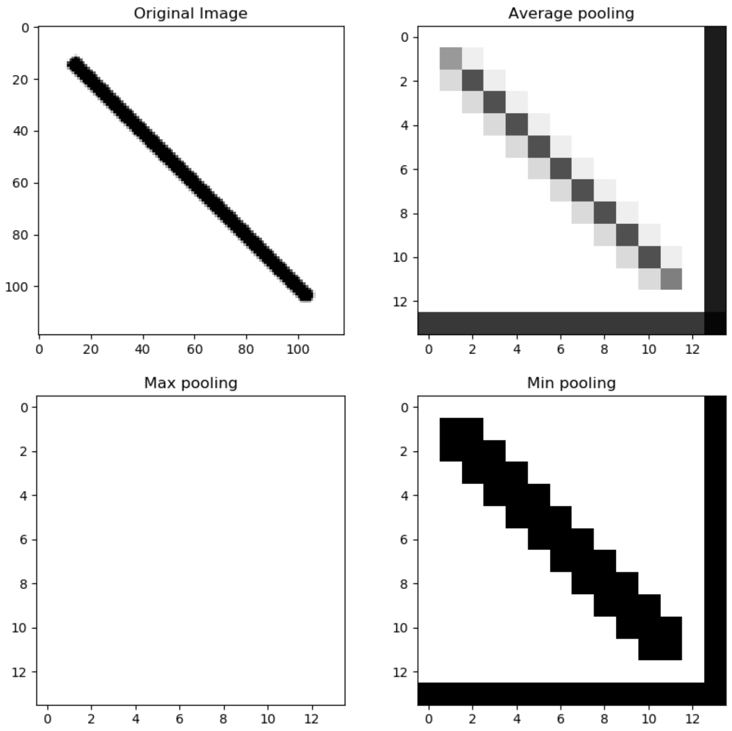
Original

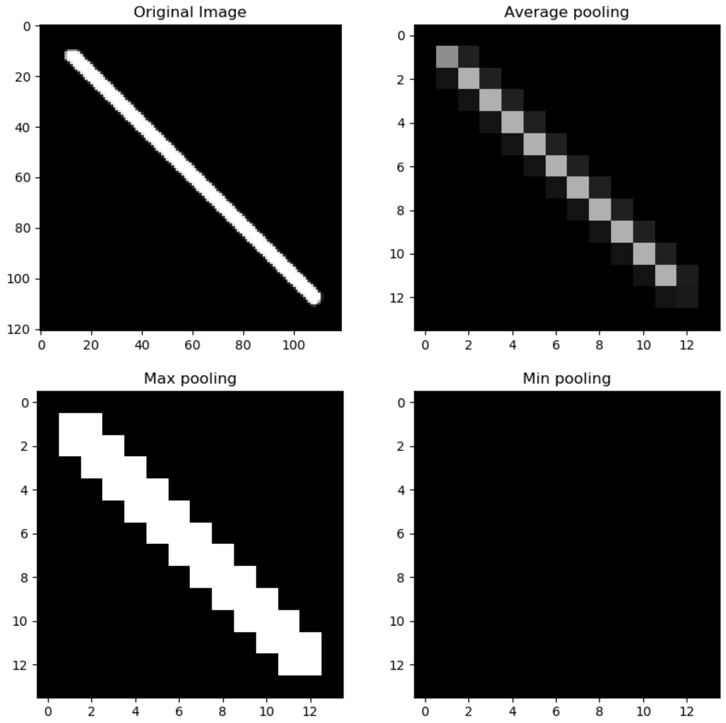
Max Pooling

Average Pooling

Unit 04 | Pooling Layer

Max Pooling vs Average Pooling

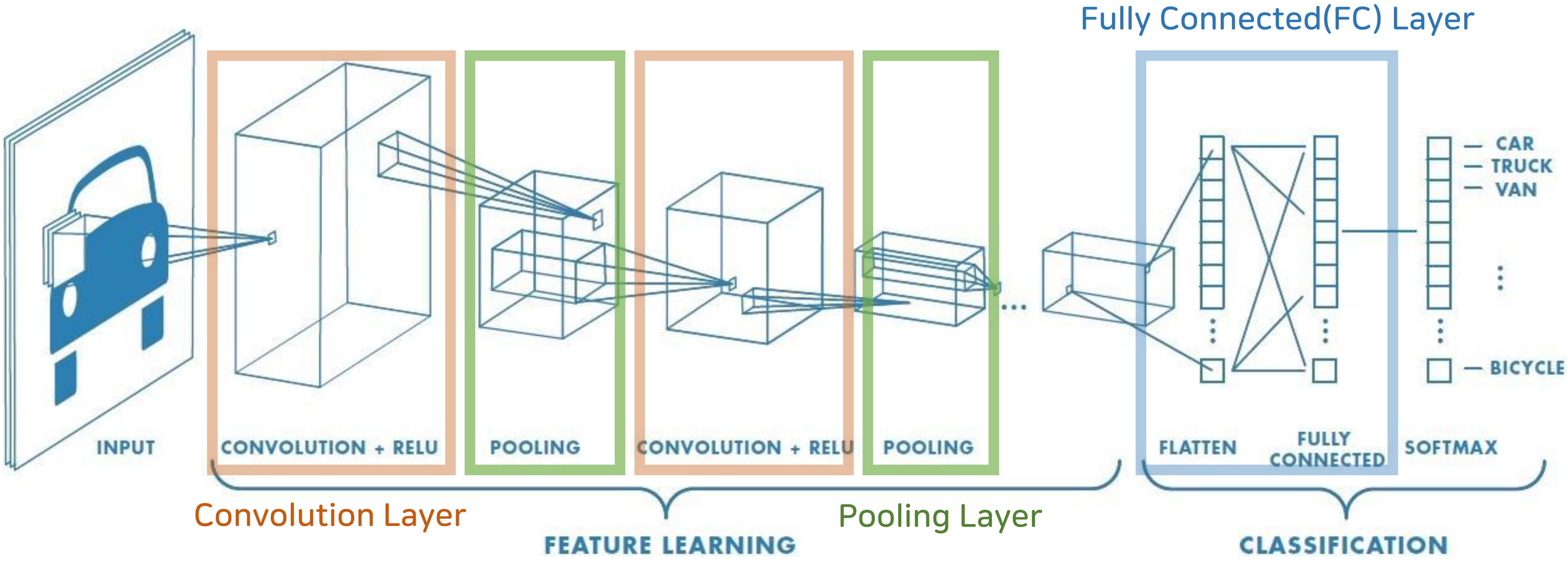




Most Important Features를 뽑는다는 관점에서 일반적으로 Max Pooling을 많이 사용한다.

Summary with code

Unit 05 | Summary with code



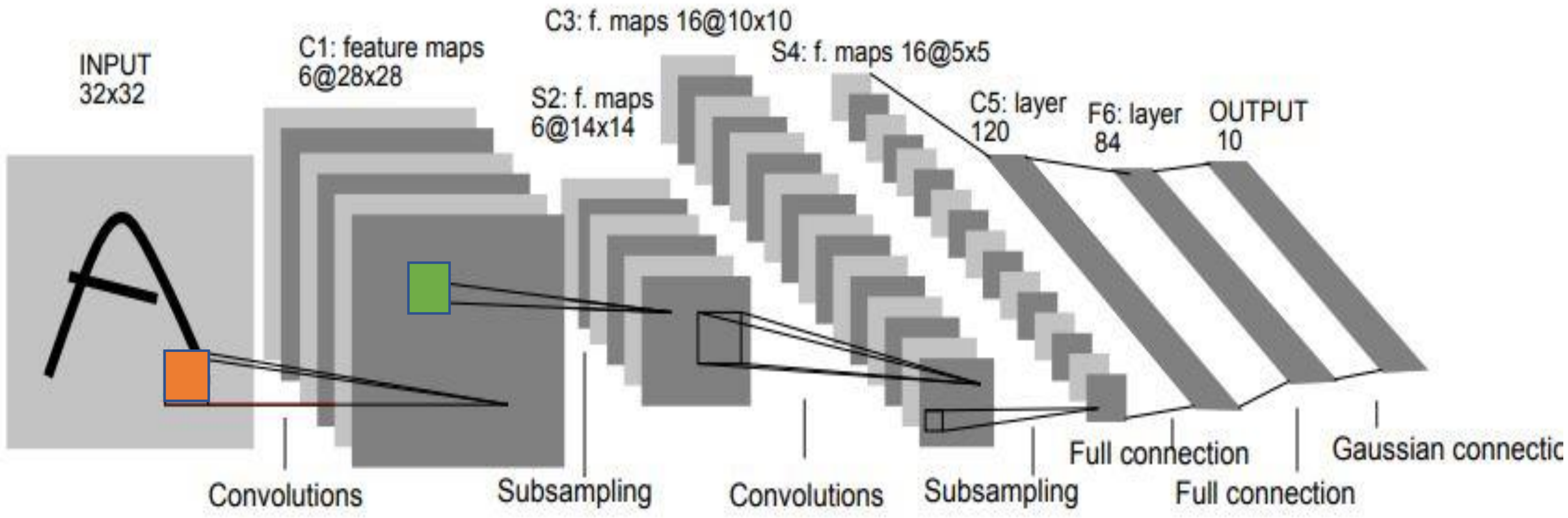
Unit 05 | Summary with code

Summary

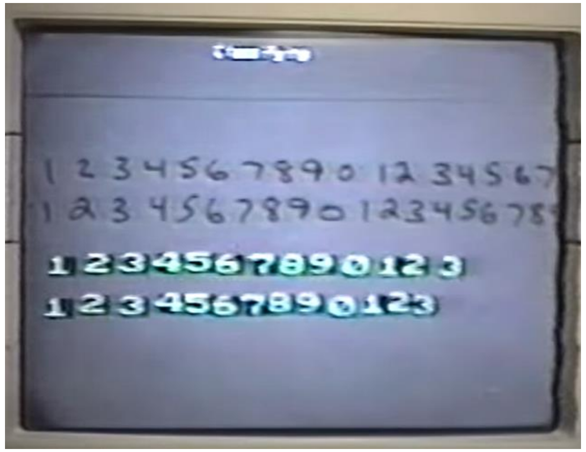
- Convolutional Layer
  - Channel
  - Filter
  - Stride
  - Padding
- Pooling Layer

Unit 05 | Summary with code

LeNet-5

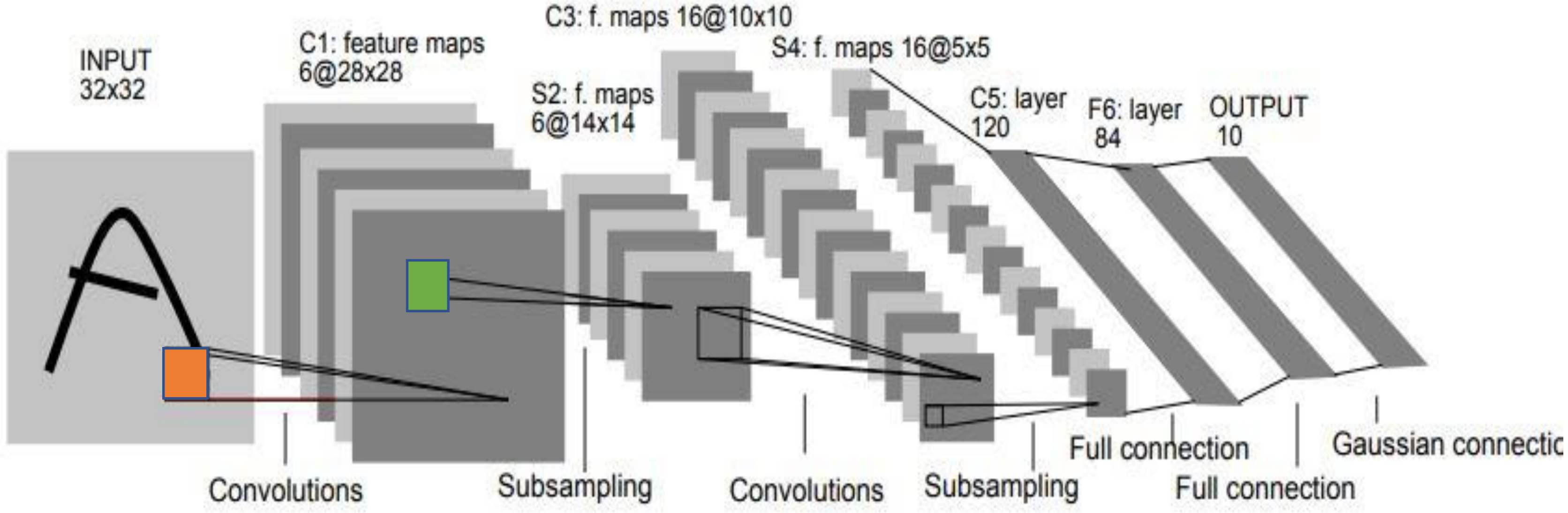


(1998) CNN의 시초



손으로 쓴 zipcode(우체국 코드)를
컴퓨터가 인식
이미지 기반 AI 기술의 가능성을 증명

LeNet-5 계산 연습



Layer1:
Convolution

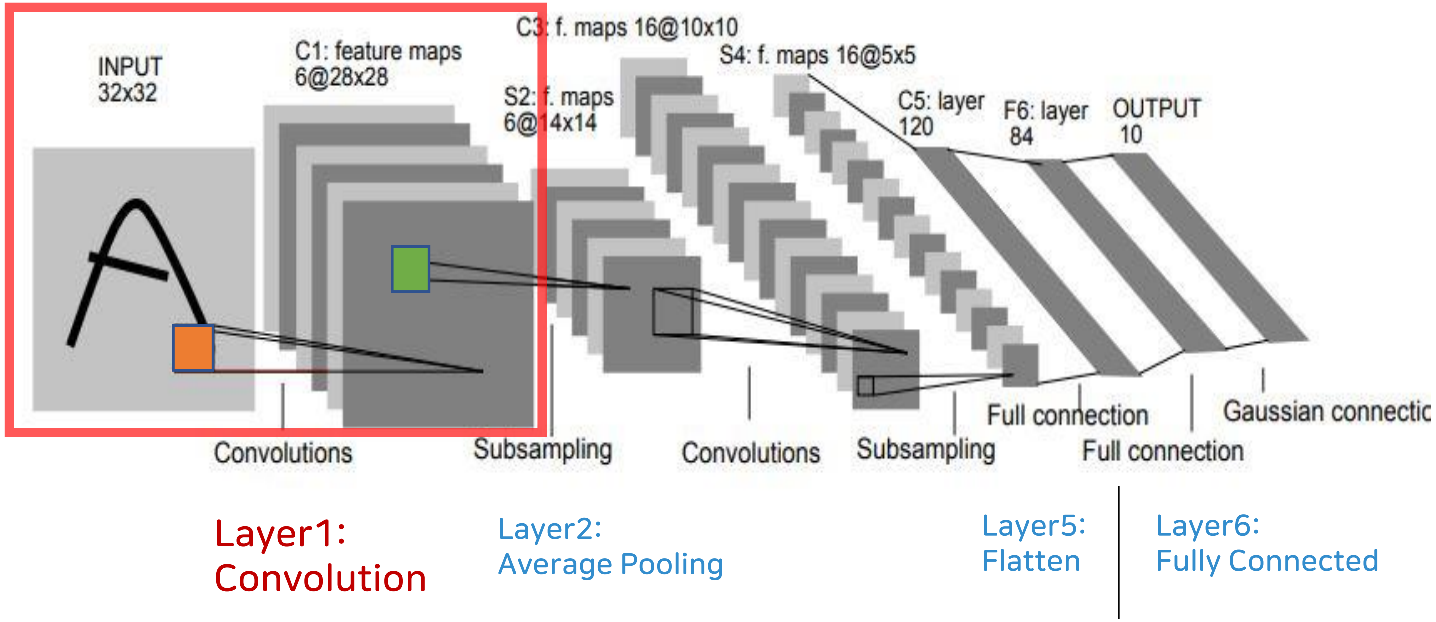
Layer2:
Max Pooling

Layer5:
Flatten

Layer6:
Fully Connected

Unit 05 | Summary with code

Layer1 (Conv)



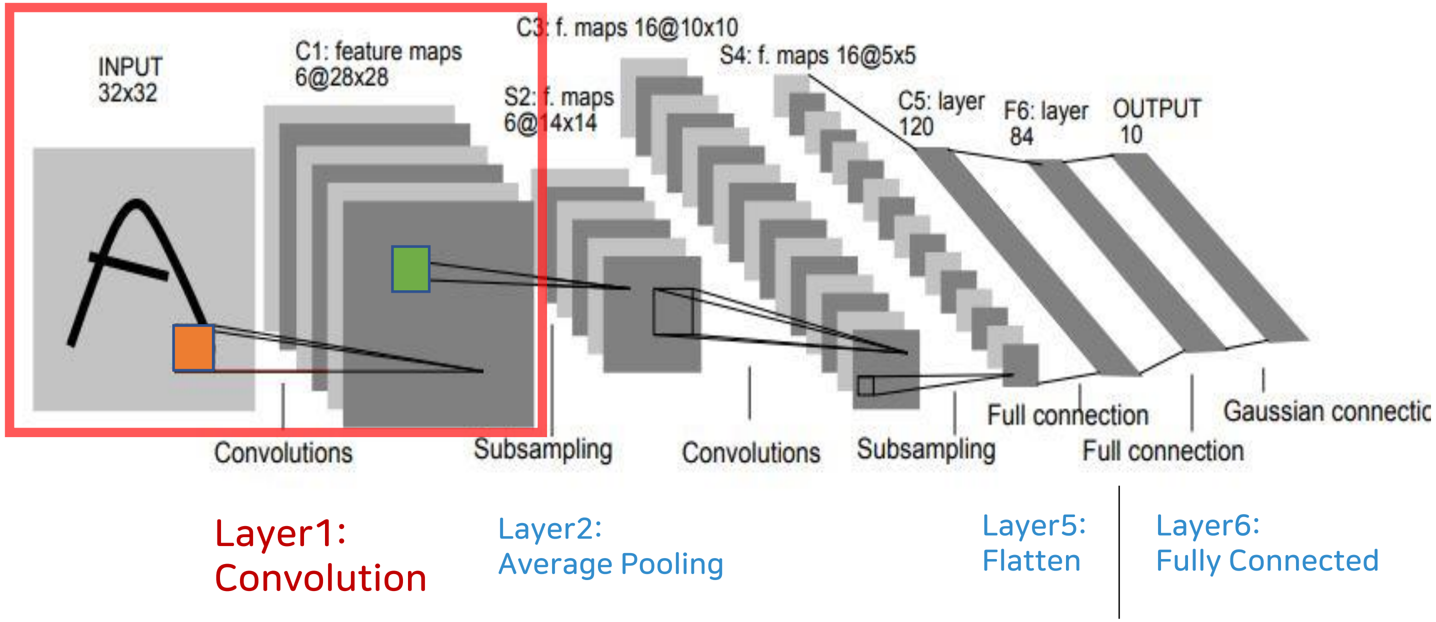
1. Input/output shape
   - Input : 32x32x1
     - filter : (5,5), 6개
     - stride : 1
     - padding : x
   - Output :

2. 파라미터(모수) 개수

   :

Unit 05 | Summary with code

Layer1 (Conv)



1. Input/output shape
   - Input : 32x32x1 (IH IW IC)
     - filter : (5,5), 6개 (FH FW FN), FC 1
     - stride : 1
     - padding : x
   - Output :

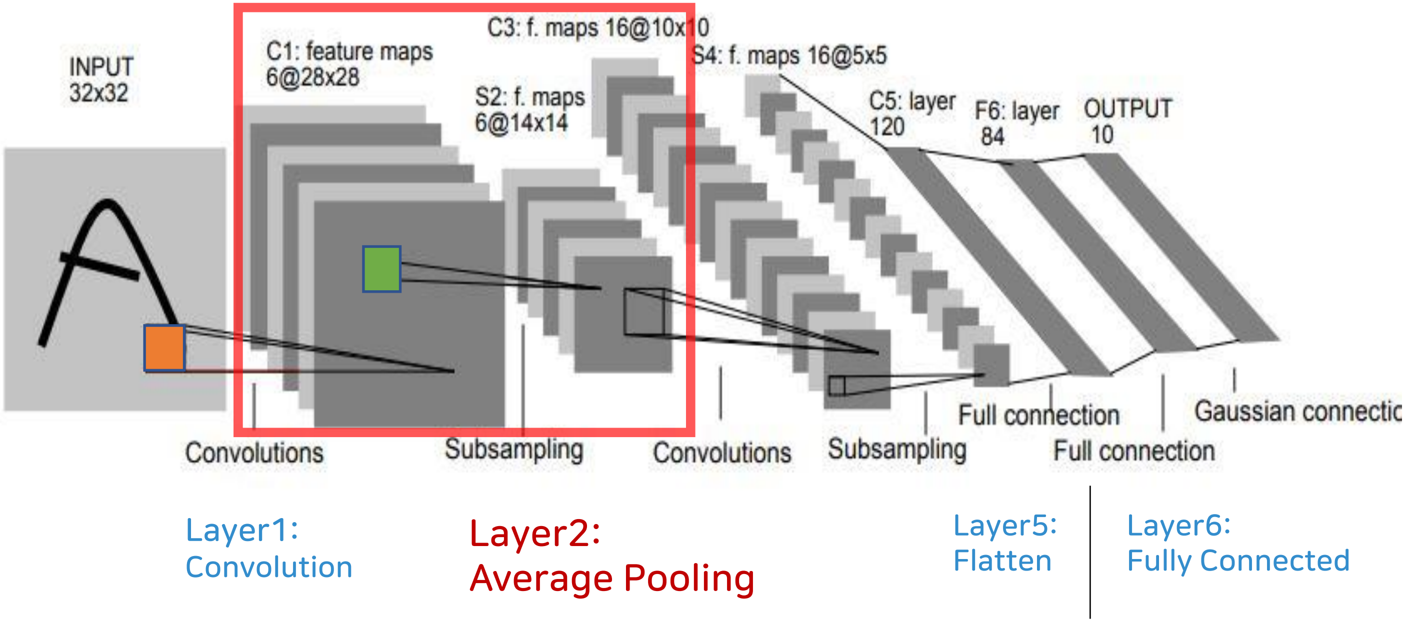
2. 파라미터(모수) 개수

   :

[Hint1]
(input_size + 2*padding – filter_size) / stride +1
= (IH + 2*P – FH) / stride + 1

[Hint2]
(filter 하나 당 가중치 개수 + bias 개수) x filter 개수
= (FH x FW x FC + 1) x FN

Layer2 (Average Pooling)

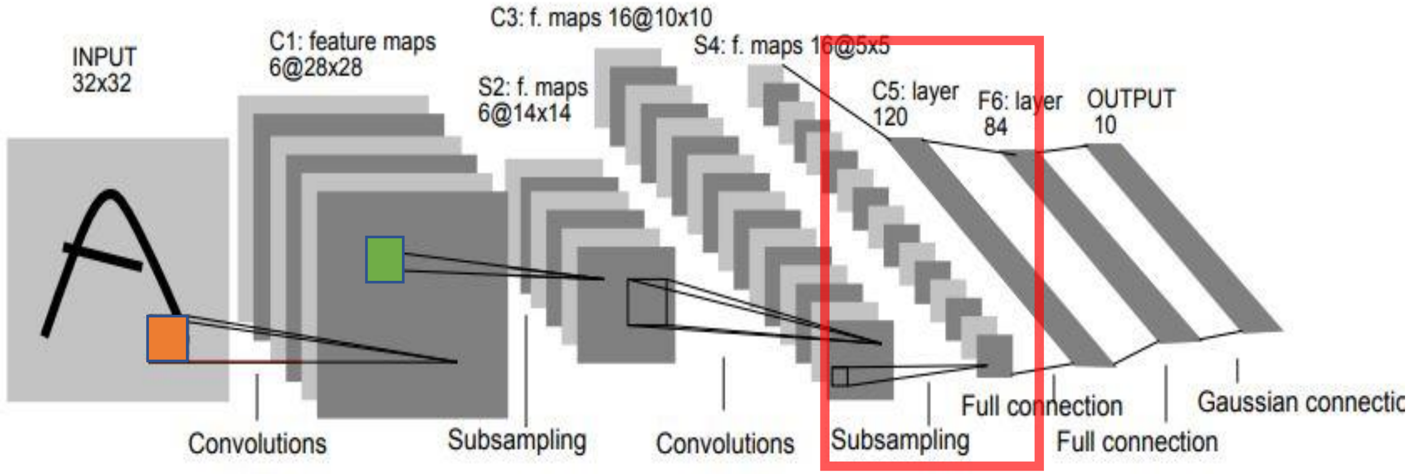


1. Input/output shape
   - Input : 6x28x28
     - filter : (2,2)
     - stride : (2,2)
   - Output :

2. 파라미터(모수) 개수
   : 원래는 x, Lenet 한정 존재
   - 평균값(a)
     -> a*(weight + bias) -> sigmoid
     = (1+1)*6 = 12

Layer5 (Flatten)

LeNet 한정



Layer1:
Convolution

Layer2:
Average Pooling

Layer5:
Flatten

Layer6:
Fully Connected

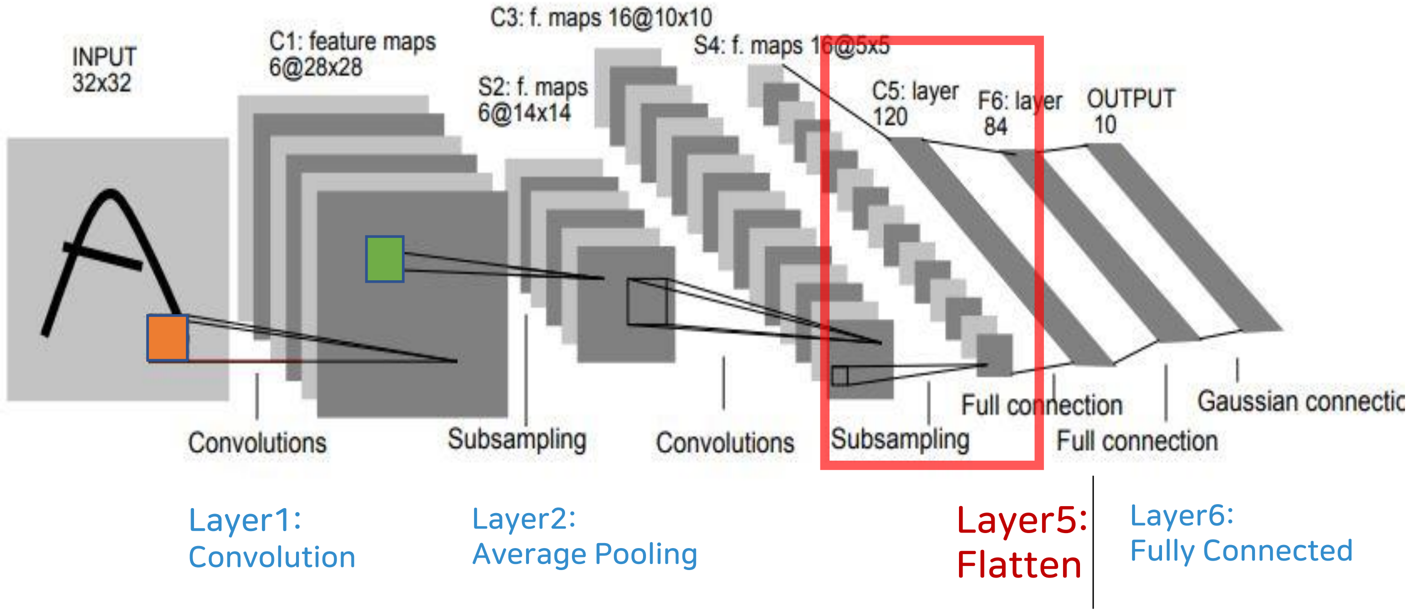
1. Input/output shape
   - Input : 16x5x5
     - filter : (5,5), 120개
     - stride : 1
     - padding : x
   - Output : 120x1x1

2. 파라미터(모수) 개수

: (5x5x16+1) x 120 = 48,120

Unit 05 | Summary with code

Layer5 (Flatten) 실제로는

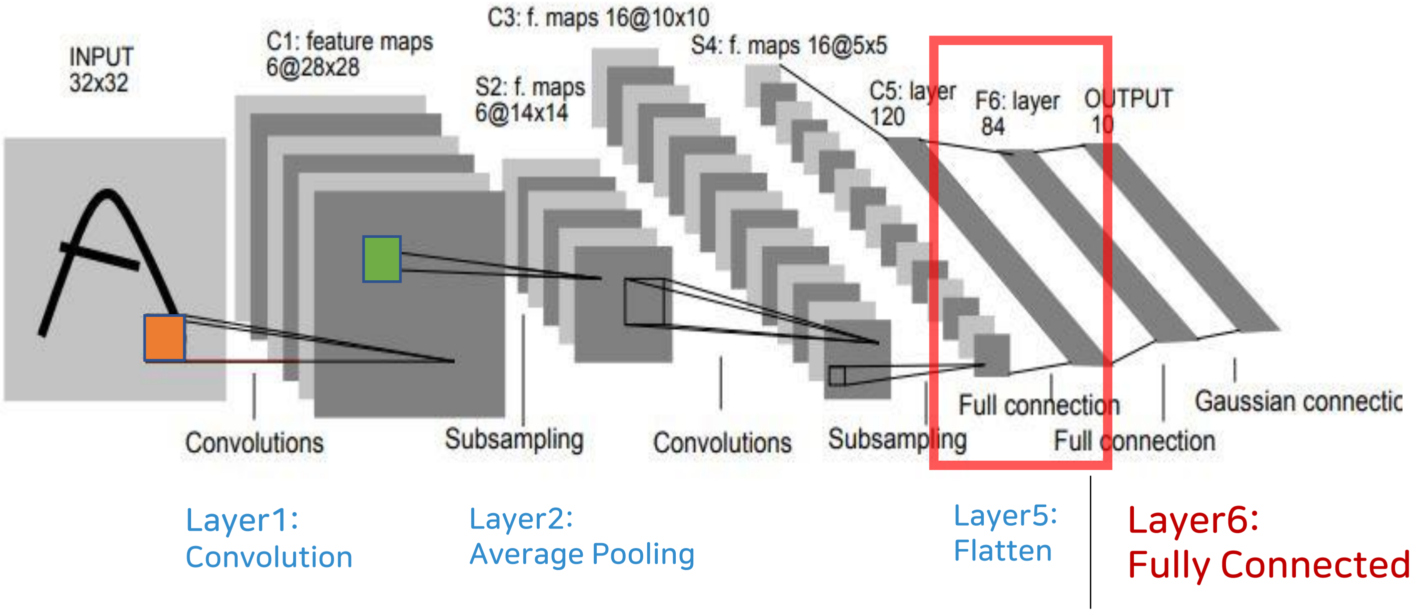


1. Input/output shape
   - Input : 16x5x5
   - Output : 120

2. 파라미터(모수) 개수

   : 없음

Layer6 (FC)



1. Input/output shape
   - Input : (,120)
   - Output : (,84)

2. 파라미터(모수) 개수

   :

Unit 05 | Summary with code

LeNet-5 (Tensorflow)

```
from tensorflow.keras.layers import Conv2D, AveragePooling2D,Flatten,Dense
from tensorflow.keras.models import Sequential

n_classes = 10 # 분류 범주 개수

model=Sequential()

model.add(Conv2D(6,(5,5),activation='tanh',input_shape=[32,32,1]))
model.add(AveragePooling2D((2,2)))

model.add(Conv2D(16,(5,5),activation='tanh'))
model.add((AveragePooling2D((2,2)))

model.add(Conv2D(120,(5,5),activation='tanh'))

model.add(Dense(84,activation='tanh'))
model.add(Dense(n_classes,activation='softmax'))
```

Unit 05 | Summary with code

LeNet-5 (Pytorch)

```
class LeNet5(nn.Module):

    def __init__(self, n_classes):
        super(LeNet5, self).__init__()

        self.feature_extractor = nn.Sequential(
            nn.Conv2d(in_channels=1, out_channels=6, kernel_size=5, stride=1),
            nn.Tanh(),
            nn.AvgPool2d(kernel_size=2),
            nn.Conv2d(in_channels=6, out_channels=16, kernel_size=5, stride=1),
            nn.Tanh(),
            nn.AvgPool2d(kernel_size=2),
            nn.Conv2d(in_channels=16, out_channels=120, kernel_size=5, stride=1),
            nn.Tanh()
        )

        self.classifier = nn.Sequential(
            nn.Linear(in_features=120, out_features=84),
            nn.Tanh(),
            nn.Linear(in_features=84, out_features=n_classes),
        )
```

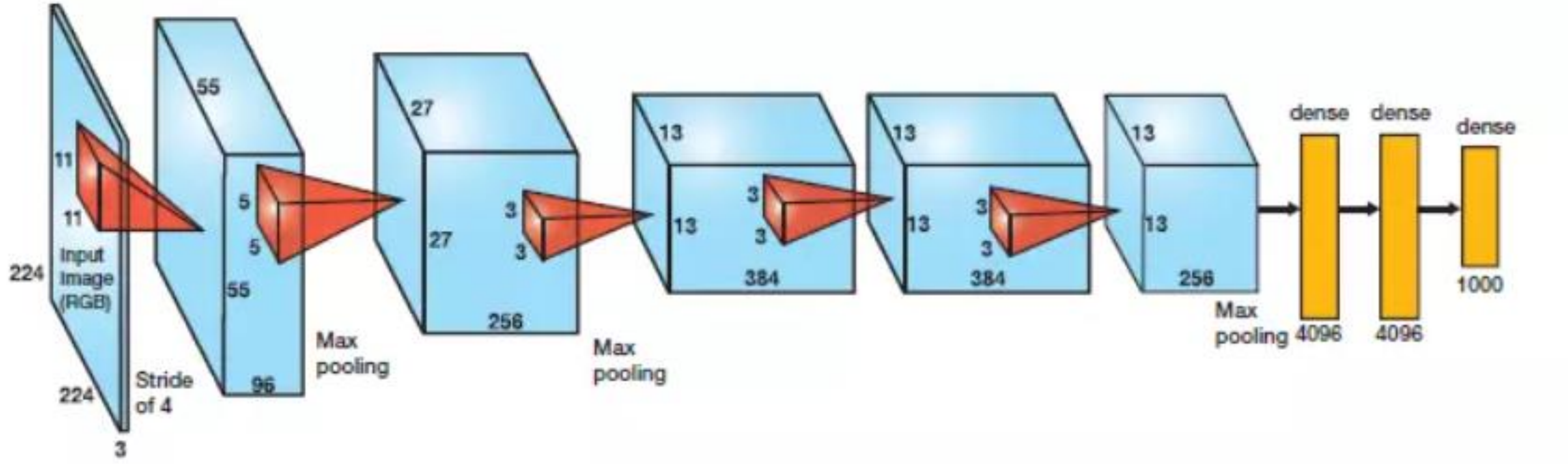
```
    def forward(self, x):
        x = self.feature_extractor(x)
        x = torch.flatten(x, 1)
        logits = self.classifier(x)
        probs = F.softmax(logits, dim=1)
        return logits, probs
```

Unit 05 | Summary with code

실습

Assignment

AlexNet



과제 1. week8_CNNbasic_assignment_parameters.ipynb
`???` 채우기

과제 2. week8_CNNbasic_assignment_modeling.ipynb
AlexNet model 구현
-모델 구현 후 summary로 전체 모델 구조 출력 + 주석으로 간단한 설명
(프레임워크는 자유, 각 프레임워크 별 summary 방법 구글링)

Bonus!

! Bonus Assignment !

참여 방법:

1. 열심히 공부를 해서 질문이 생긴다.
2. 도움을 받거나(멘토, 강의자) 스스로 답을 찾는다.
3. 멘티가 강의자에게 [질문+답+(도움 준 사람)] 카톡을 보낸다.
4. 참여 완료!

추첨을 통해서 [멘티+도움 준 사람]에게
이모티콘을 선물로 드립니다.
많은 참여 부탁드려요~!!

참고 자료

13기 고유경님 강의
http://www.datamarket.kr/xe/board_jPWY12/70471

13기 강미경님 이미지 세미나 (CS231n) 강의 리뷰
https://tobigs-staff.gitbook.io/-1/lecture-5-convolutional-neural-networks

Stanford CS231n
http://cs231n.stanford.edu/syllabus.html

Towards data science
https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

케라스 창시자에게 배우는 딥러닝
https://github.com/gilbutITbook/006975

Q & A

들어주셔서 감사합니다.