1 2 기 정 규 세 션

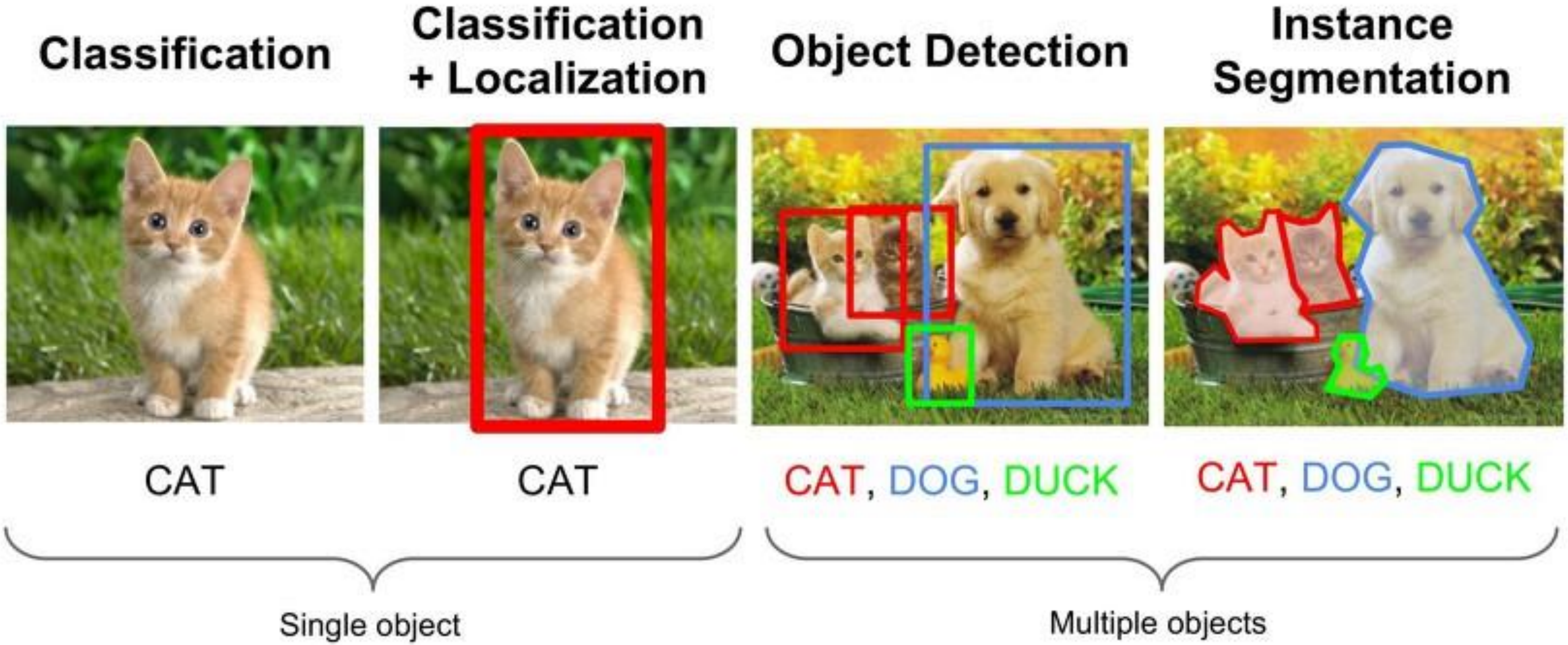
ToBig's 11기 임채빈

Convolutional Neural Networks

Contents

Unit 01 | intro : Applications of cnn

- Applications of cnn

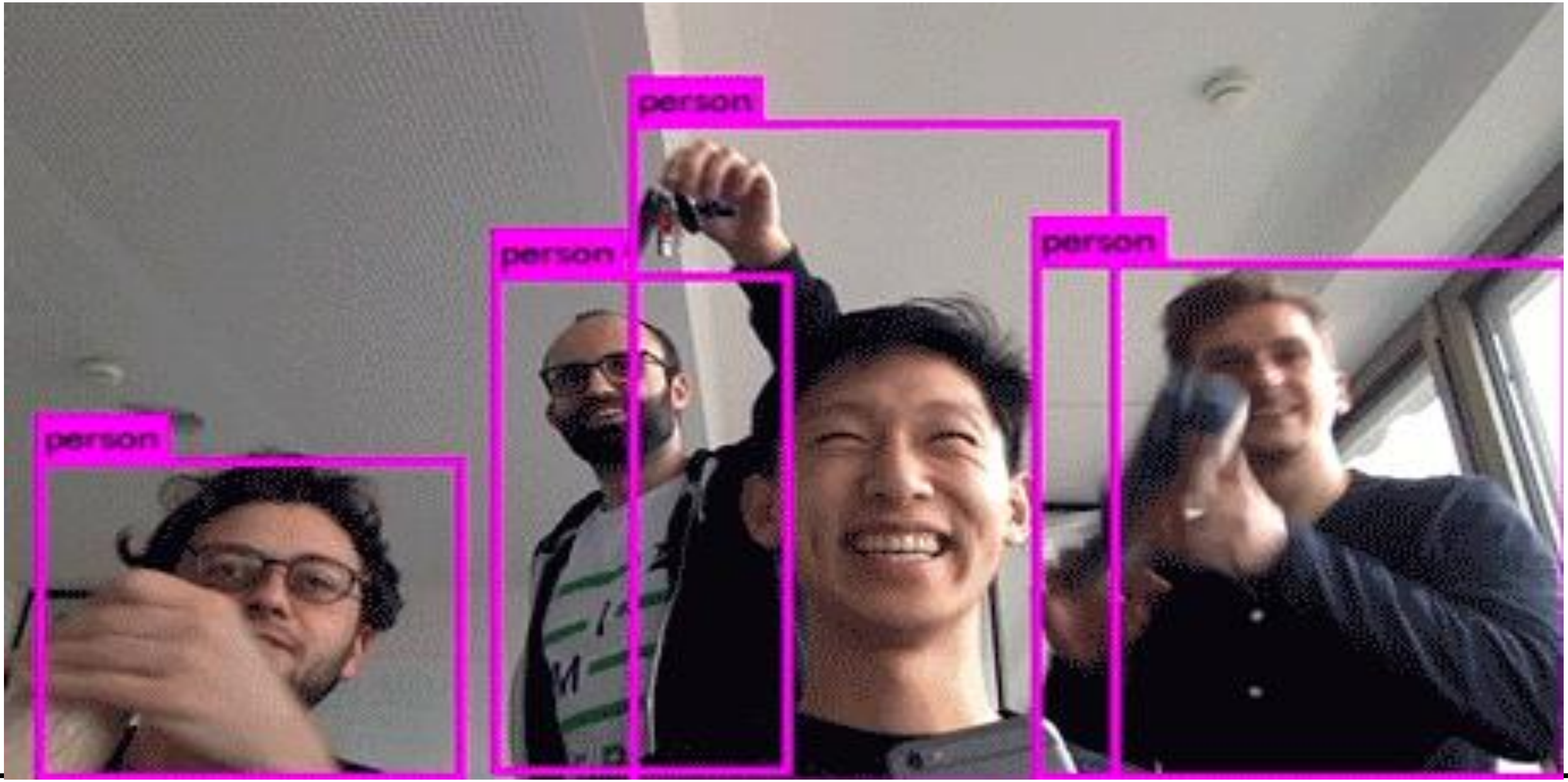


Unit 01 | intro : Applications of cnn

- Image Caption Generation

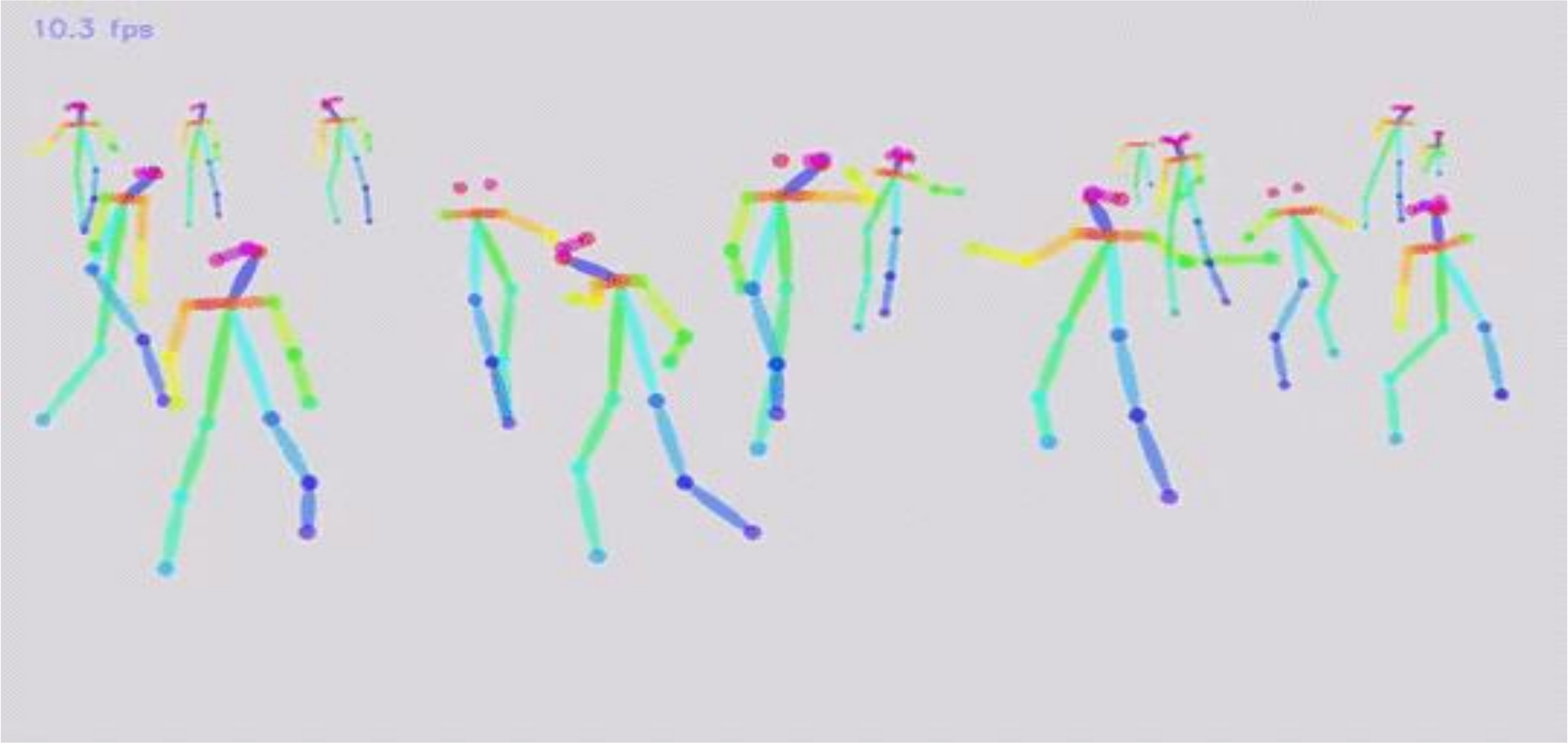
Unit 01 | intro : Applications of cnn

- Real Time Object Detection



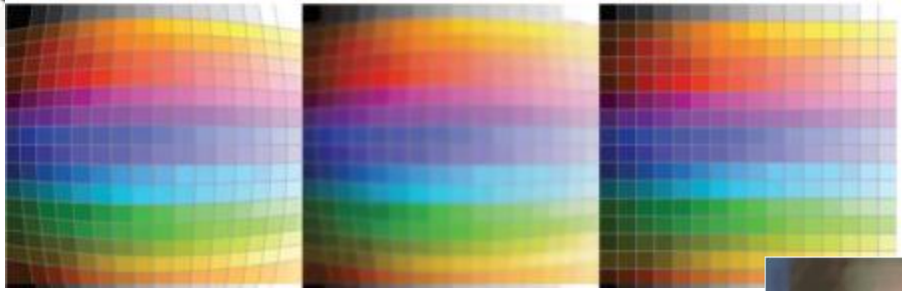
Unit 01 | intro : Applications of cnn

- Real Time Pose Estimation

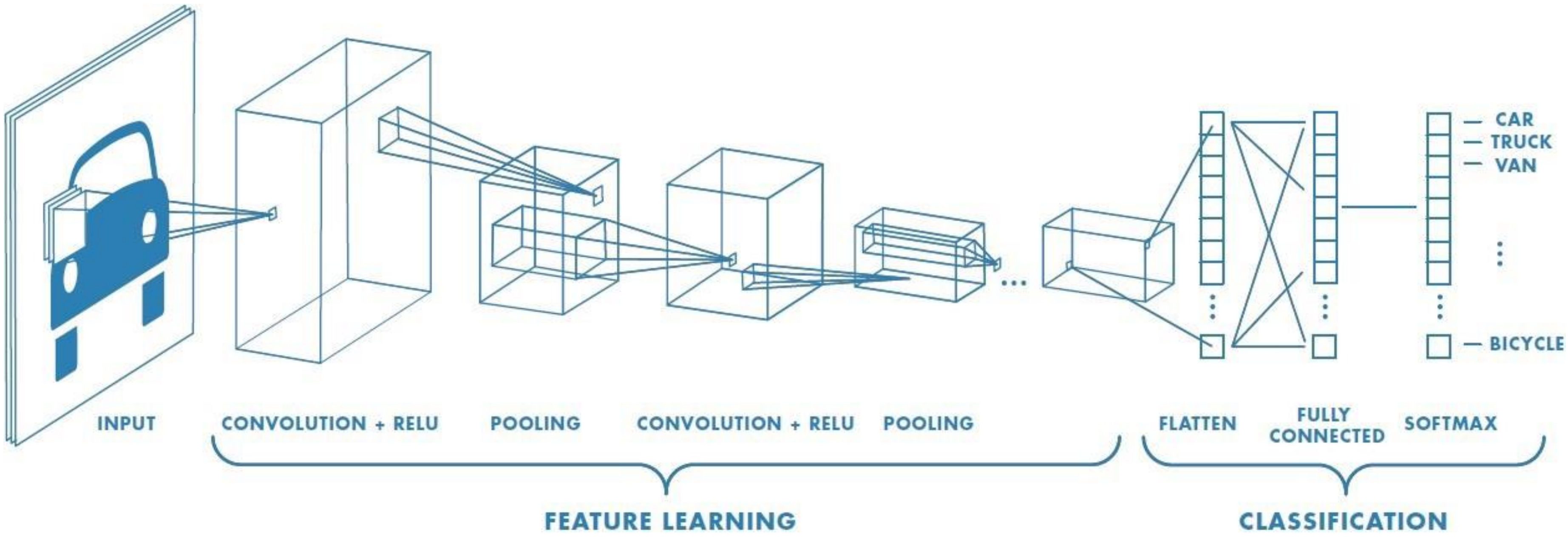


Unit 01 | intro : Applications of cnn

- GAN



Unit 02 | Layers in CNN

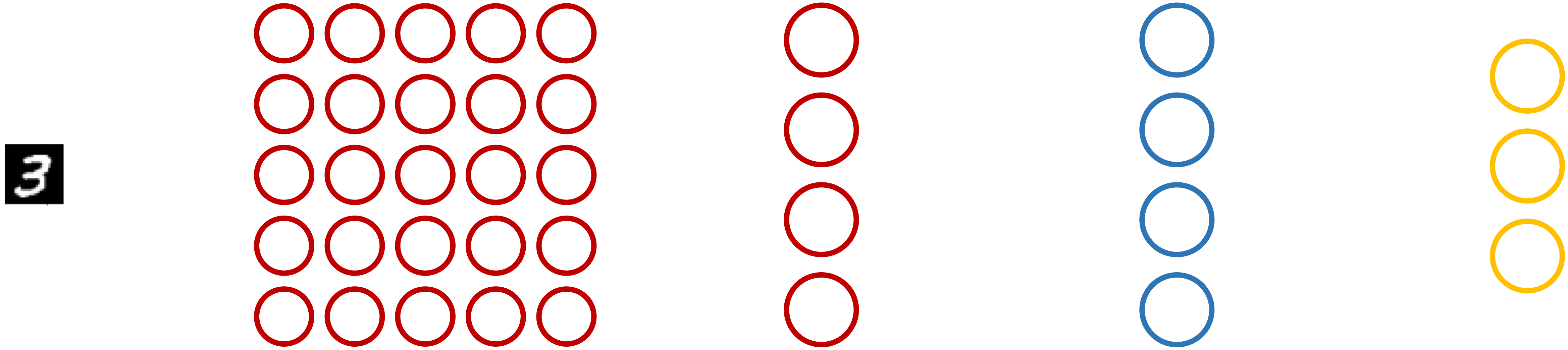


- Convolution Layer + Subsampling Layer + Fully Connected Layer

- Feature extraction : Convolution Layer + Pooling Layer
- Classification : Fully Connected Layer

Unit 02 | Layers in CNN

- Conventional Neural Network



Unit 02 | Layers in CNN

- Conventional Neural Network

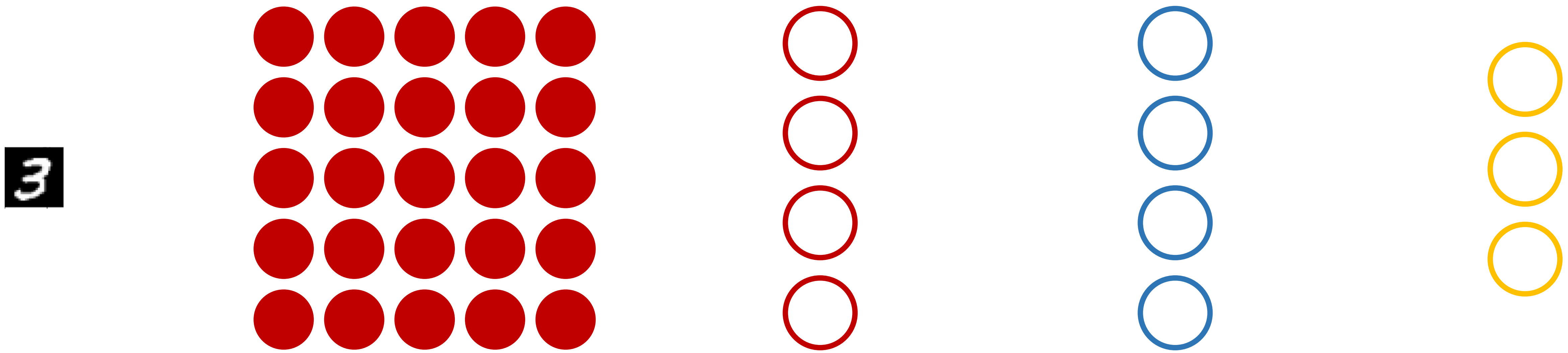
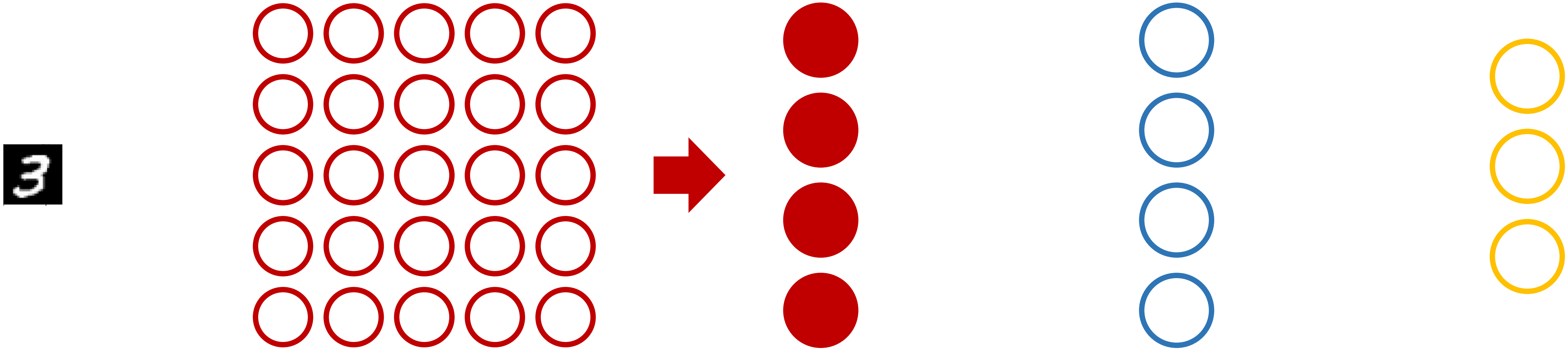


Image pixel (5x5)

Unit 02 | Layers in CNN

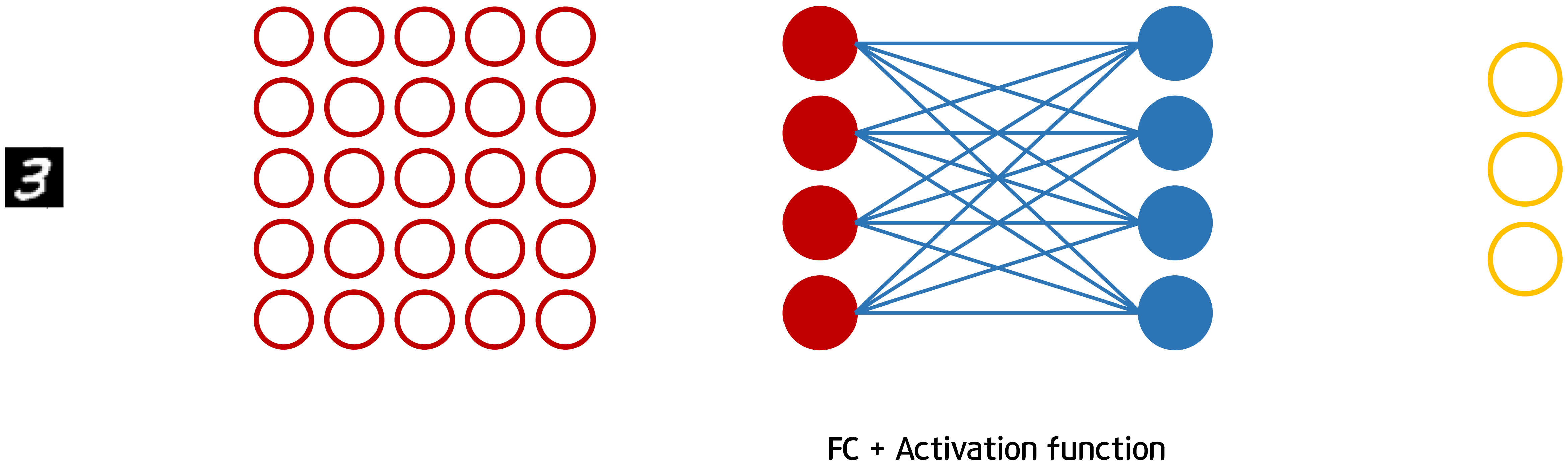
- Conventional Neural Network



Flatten (1 x 25)

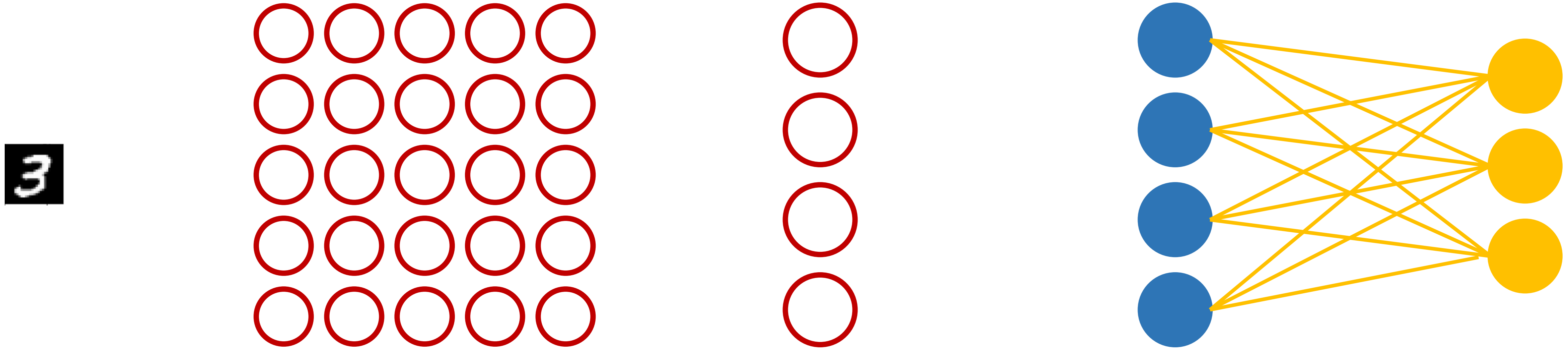
Unit 02 | Layers in CNN

- Conventional Neural Network



Unit 02 | Layers in CNN

- Conventional Neural Network



FC+ Classifier

Unit 02 | Layers in CNN

- Convolutional Neural Network

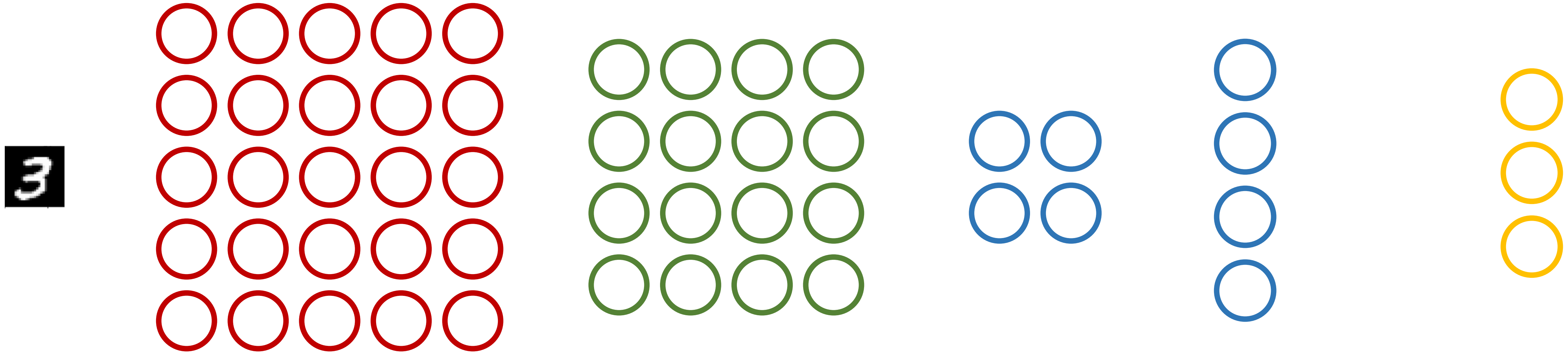
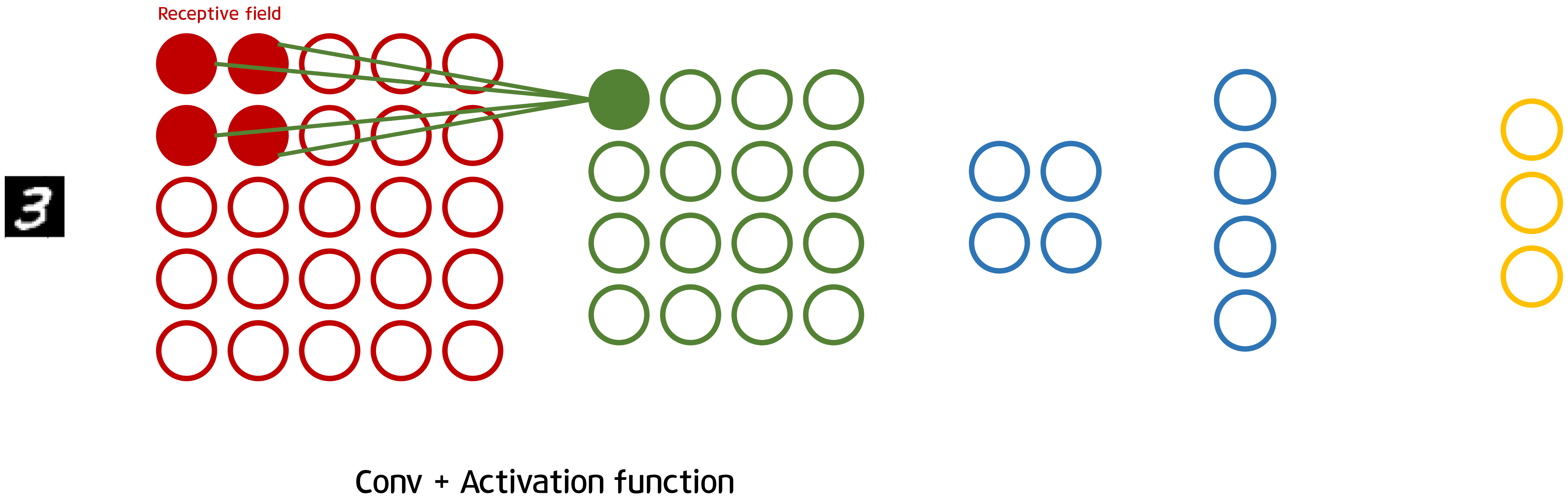


Image pixel (5x5)

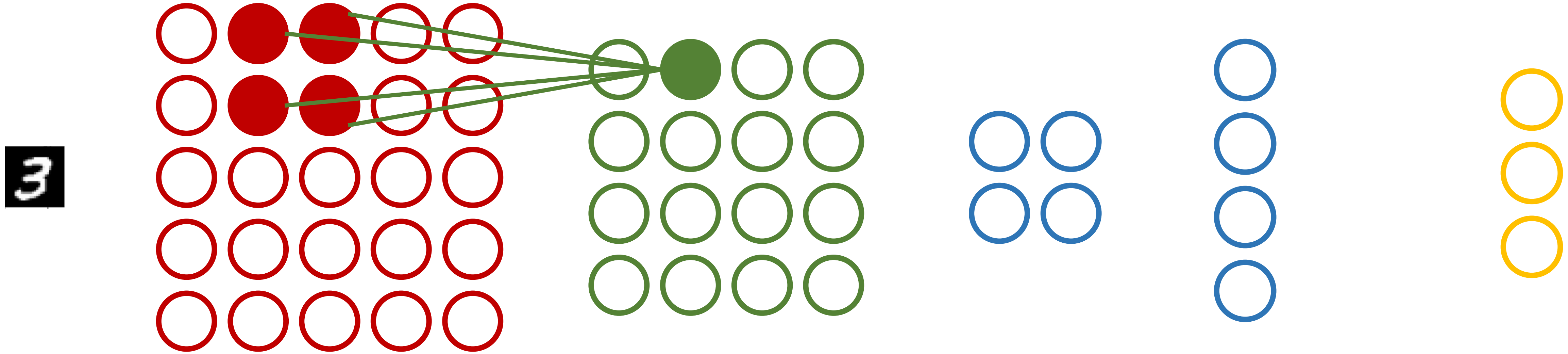
Unit 02 | Layers in CNN

- Convolutional Neural Network
  - Receptive field : 출력 레이어의 뉴런 하나에 영향을 미치는 입력 뉴런들의 공간



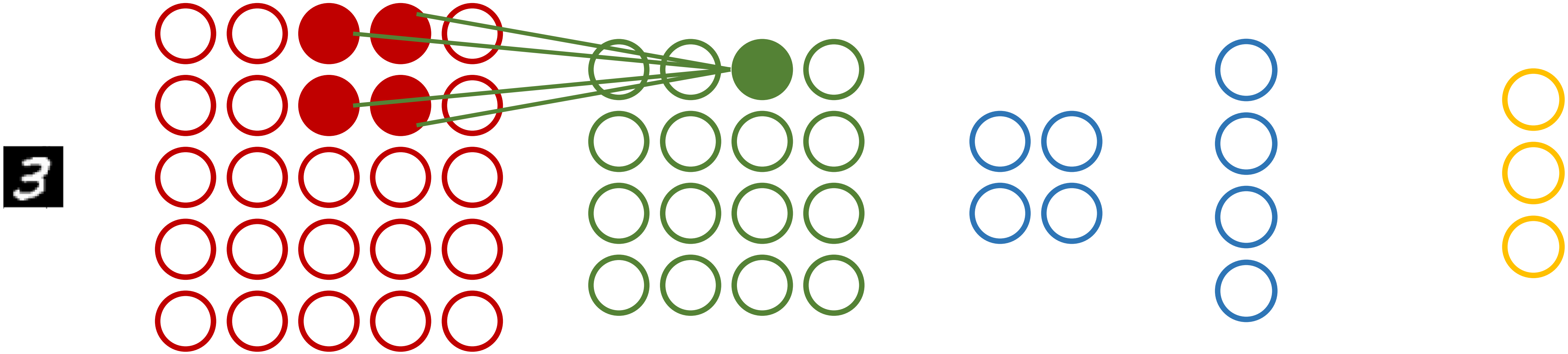
Unit 02 | Layers in CNN

- Convolutional Neural Network



Conv + Activation function

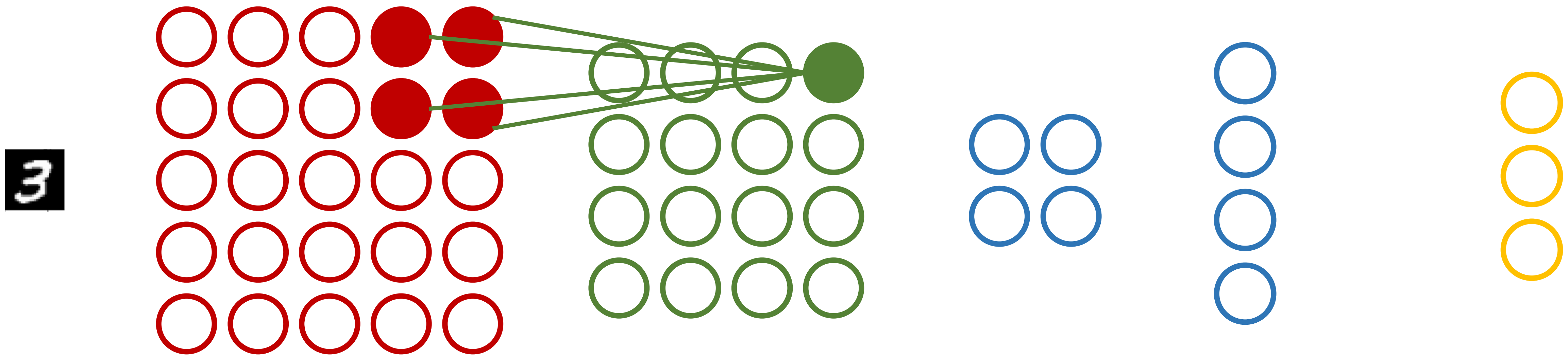
- Convolutional Neural Network



Conv + Activation function

Unit 02 | Layers in CNN

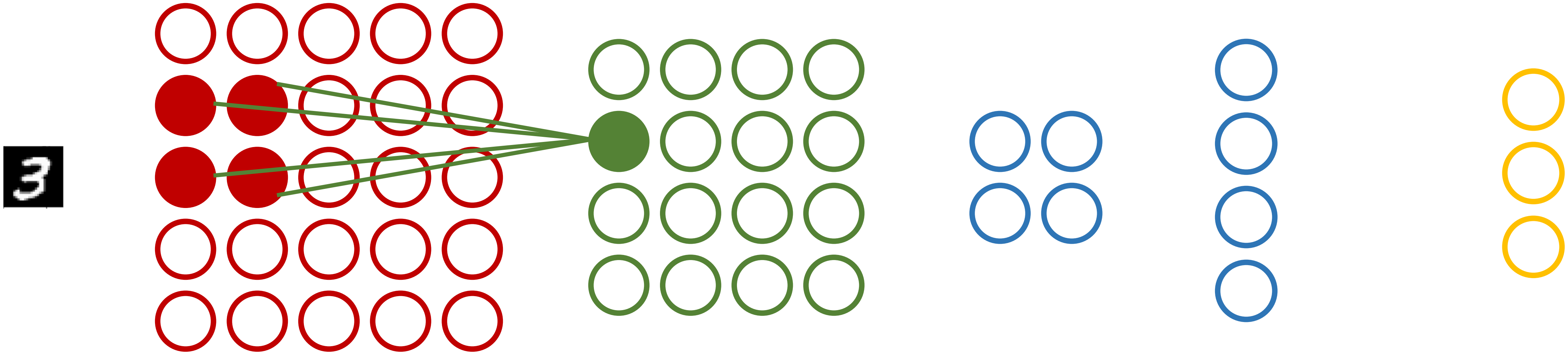
- Convolutional Neural Network



Conv + Activation function

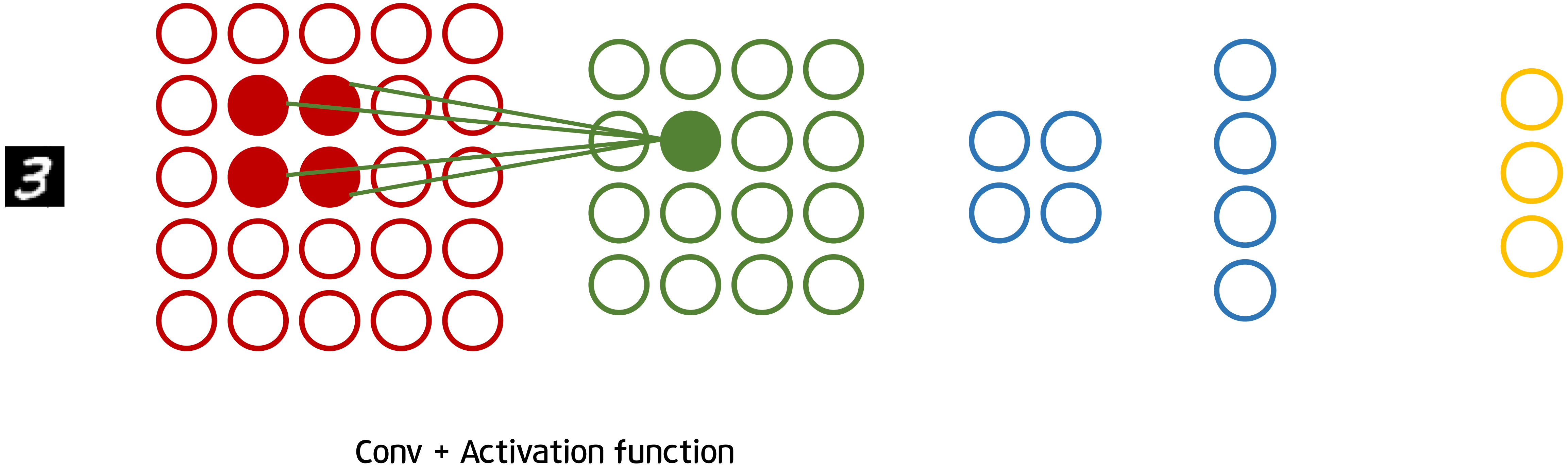
Unit 02 | Layers in CNN

- Convolutional Neural Network



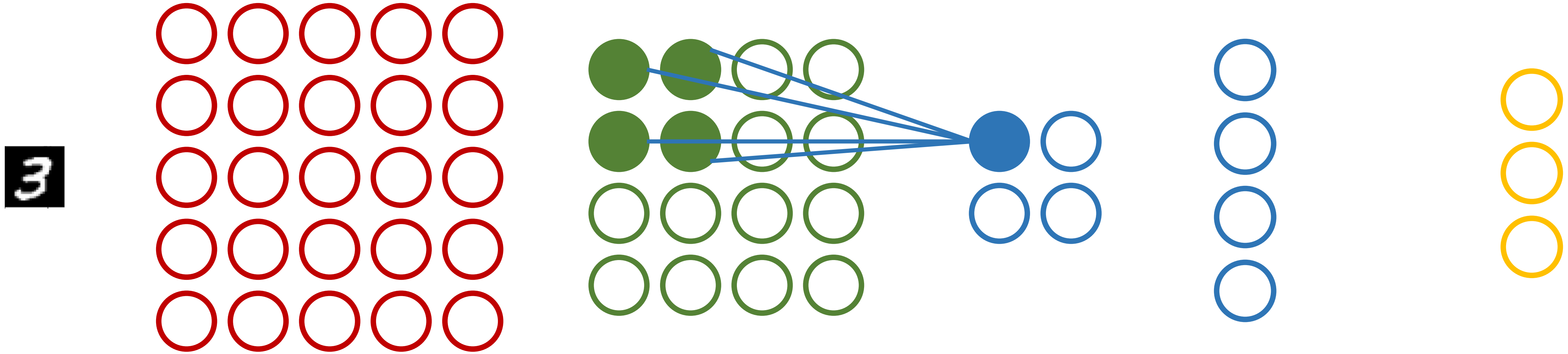
Conv + Activation function

- Convolutional Neural Network



Unit 02 | Layers in CNN

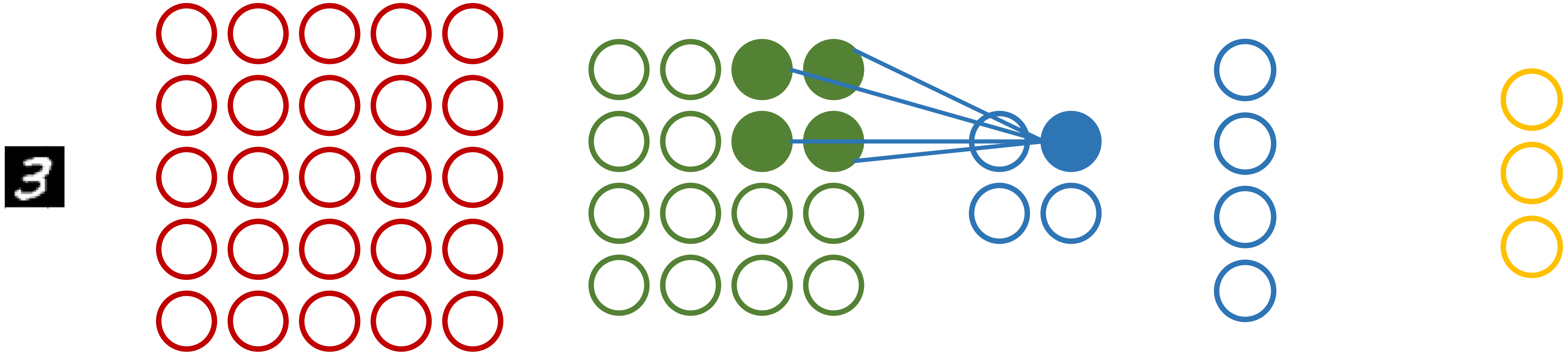
- Convolutional Neural Network



pooling

Unit 02 | Layers in CNN

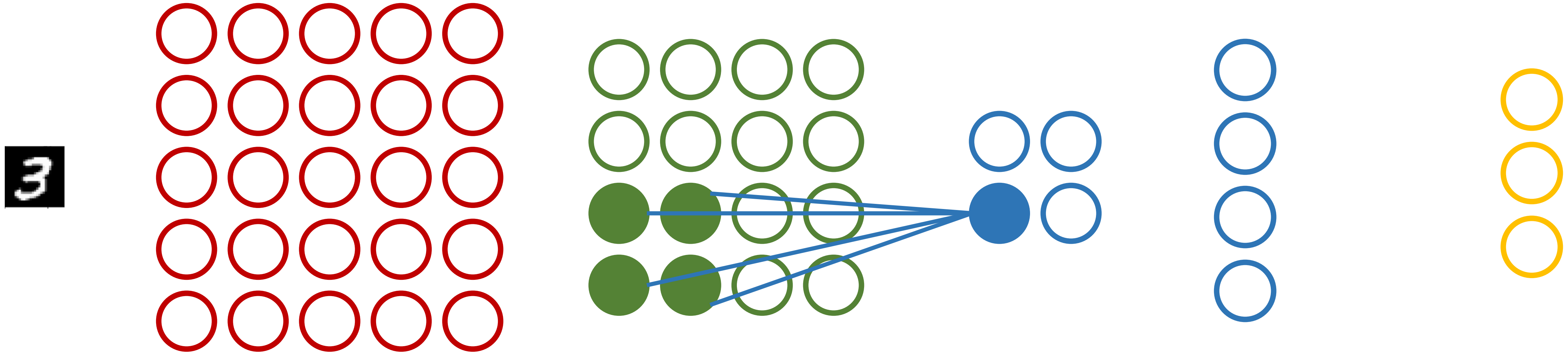
- Convolutional Neural Network



pooling

Unit 02 | Layers in CNN

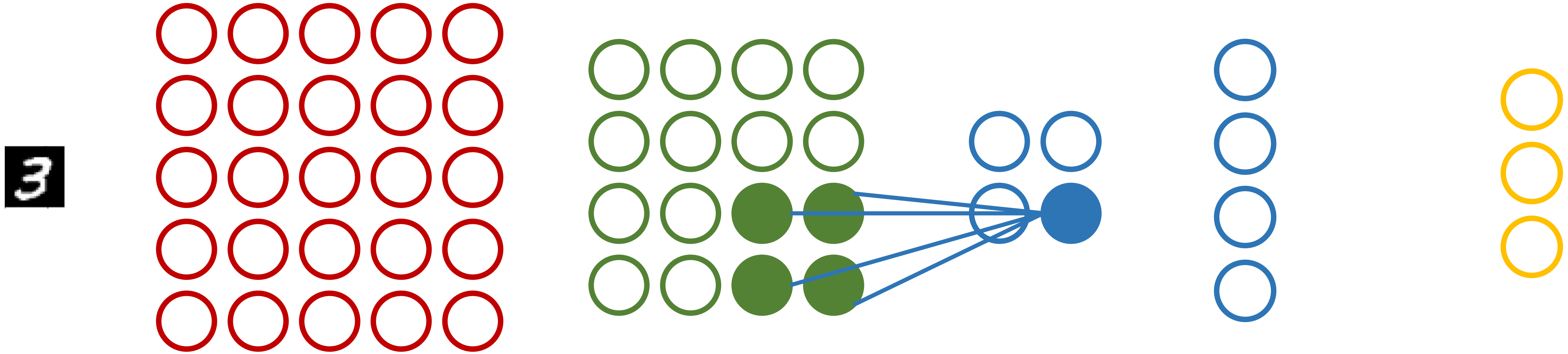
- Convolutional Neural Network



pooling

Unit 02 | Layers in CNN

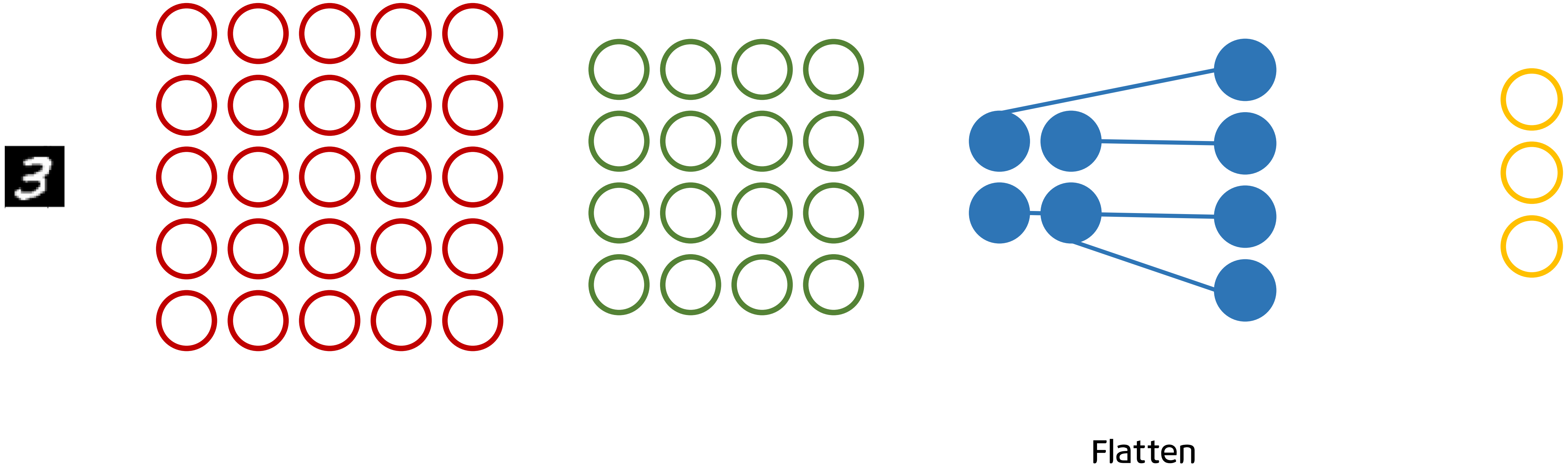
- Convolutional Neural Network



pooling

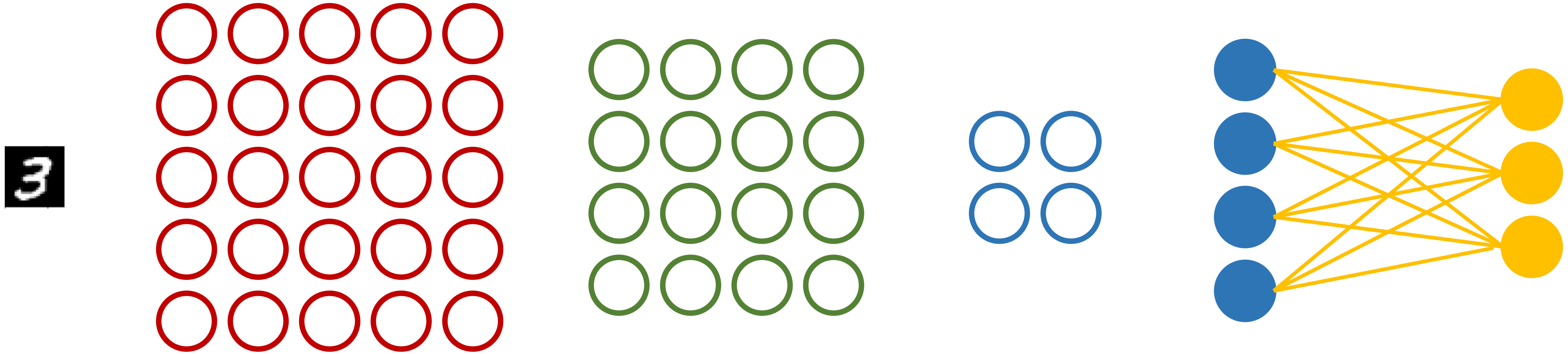
Unit 02 | Layers in CNN

- Convolutional Neural Network



Unit 02 | Layers in CNN

- Convolutional Neural Network

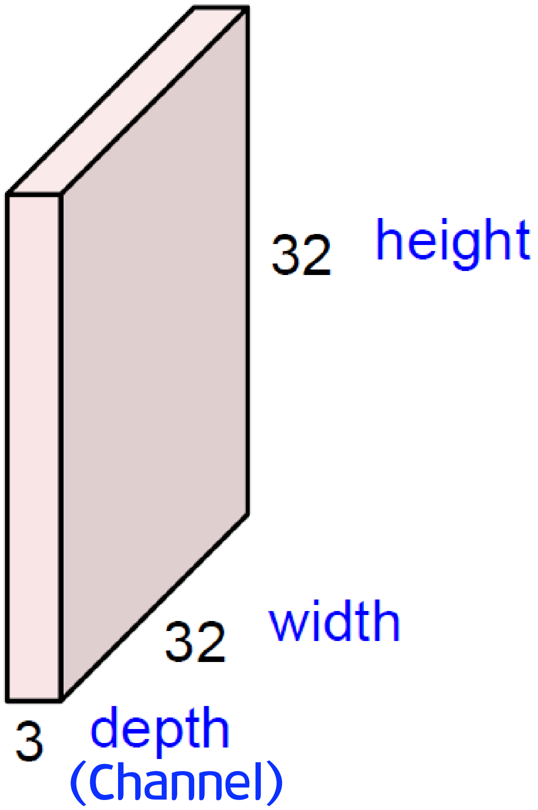


FC + Classifier

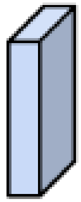
Unit 03 | Convolution Layer

- Convolution

32x32x3 image -> preserve spatial structure



5x5x3 filter (Shared Weight)

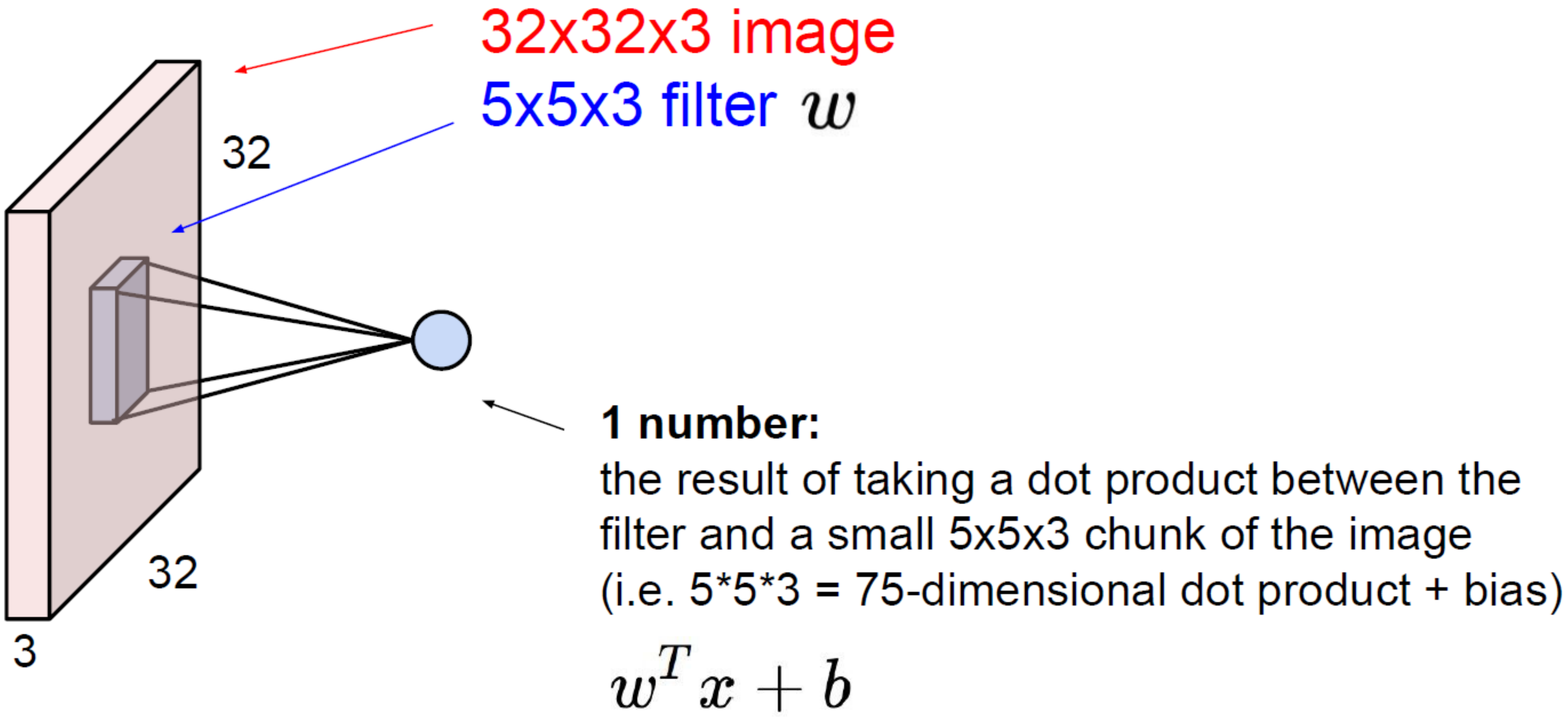


- Image와 filter의 channel은 항상 같다.

  Ex) RGB = 3 channel, grayscale = 1channel

Unit 03 | Convolution Layer

- Convolution



Unit 03 | Convolution Layer

- Convolution

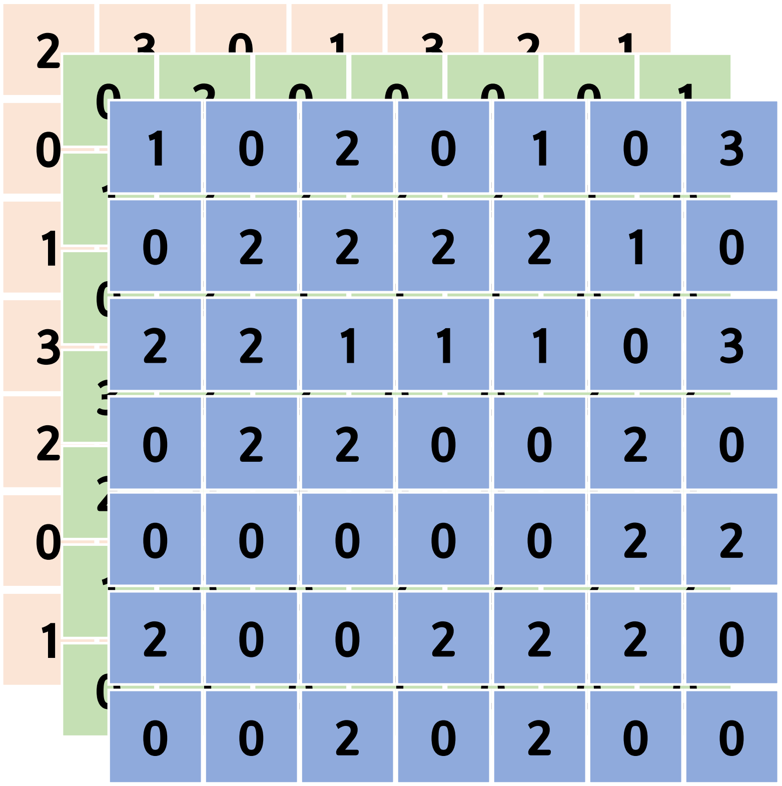
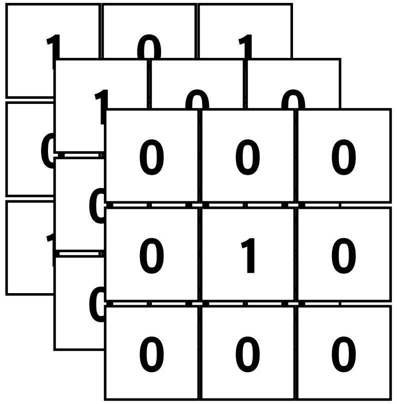


Image 7x7x3



filter 3x3x3

Unit 03 | Convolution Layer

- Convolution

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 Volumn (7x7)

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

Dot Product

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 Volumn (7x7)

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6				

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

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 Volumn (7x7)

Dot Product

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6	9			

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

Dot Product

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 Volumn (7x7)

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

6	9	7		

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

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 Volumn (7x7)

1	0	1
0	1	0
1	0	1

Filter (3 x 3)

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

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

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 Volumn (7x7)

1	0	0
0	0	0
0	0	0

Filter (3 x 3)

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

Convolved Feature Volumn (5 x 5)

Unit 03 | Convolution Layer

- Convolution

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 Volumn (7x7)

0	0	0
0	1	0
0	0	0

Filter (3 x 3)

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

Convolved Feature Volumn (5 x 5)

- Convolution

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

+

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

+

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

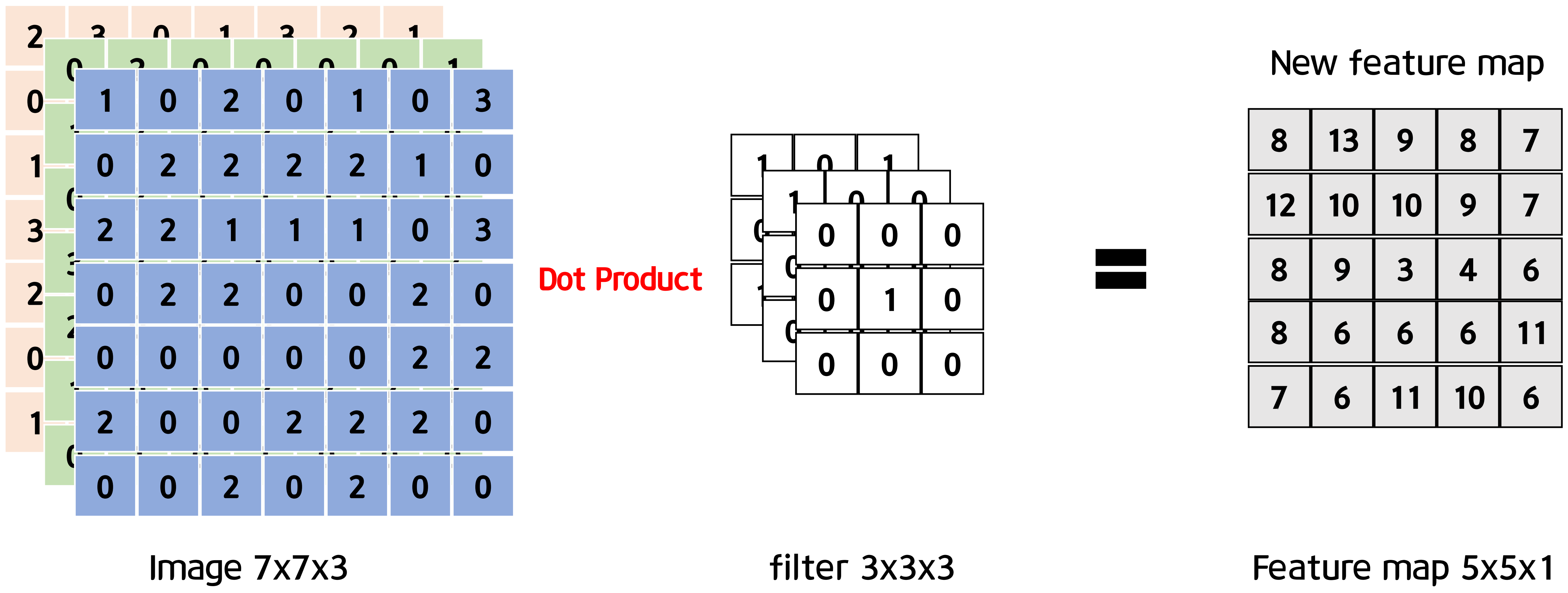
=

New 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

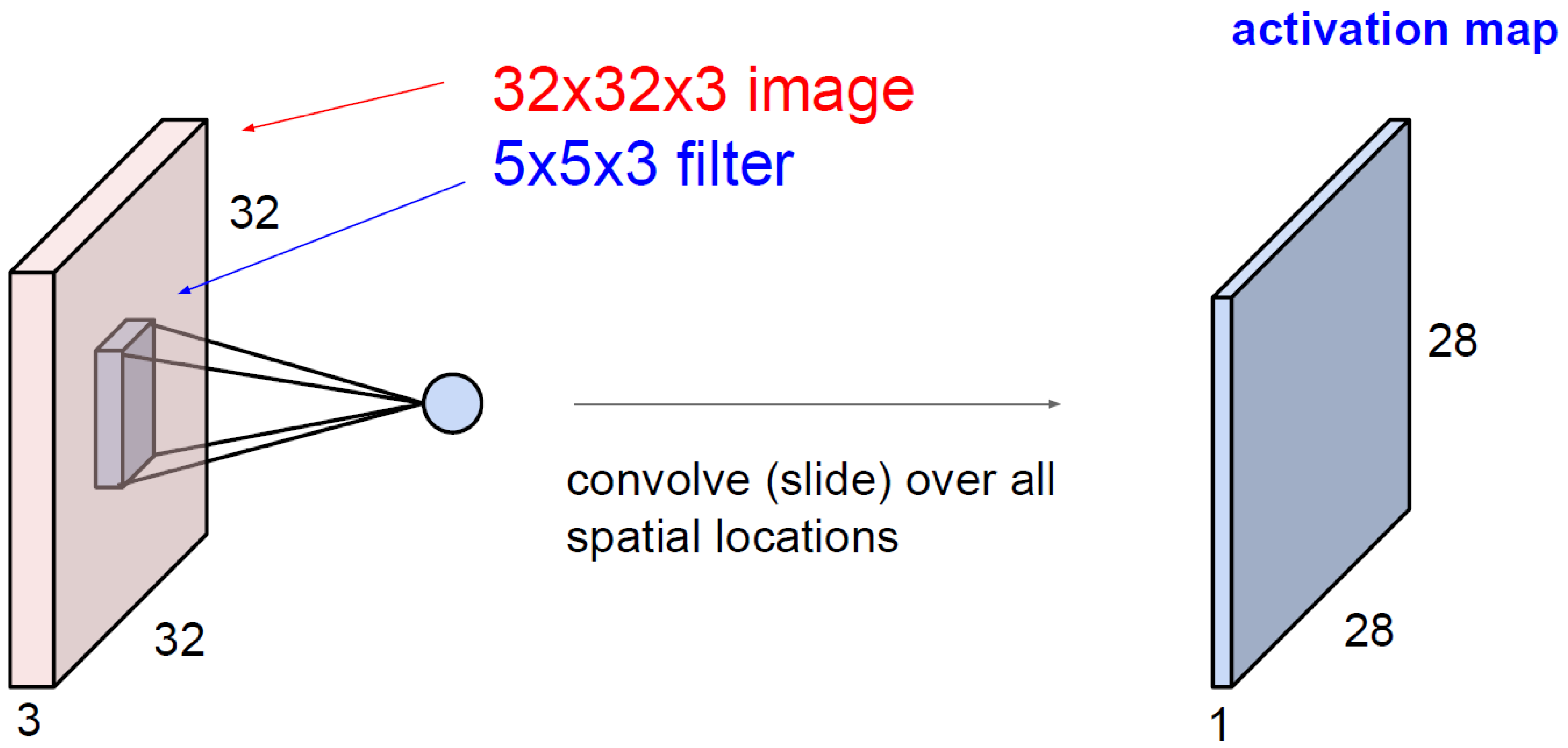
Unit 03 | Convolution Layer

- Convolution



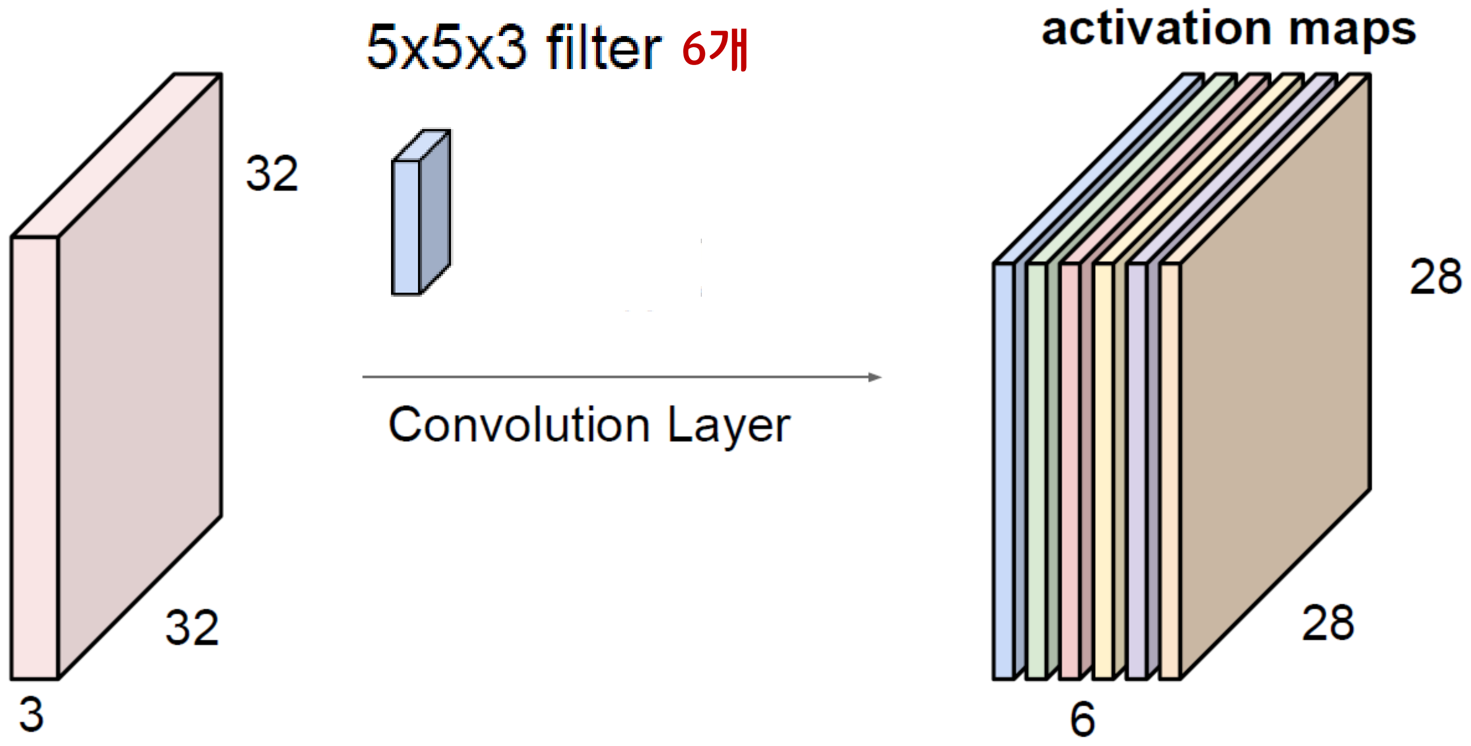
Unit 03 | Convolution Layer

- Convolution



Unit 03 | Convolution Layer

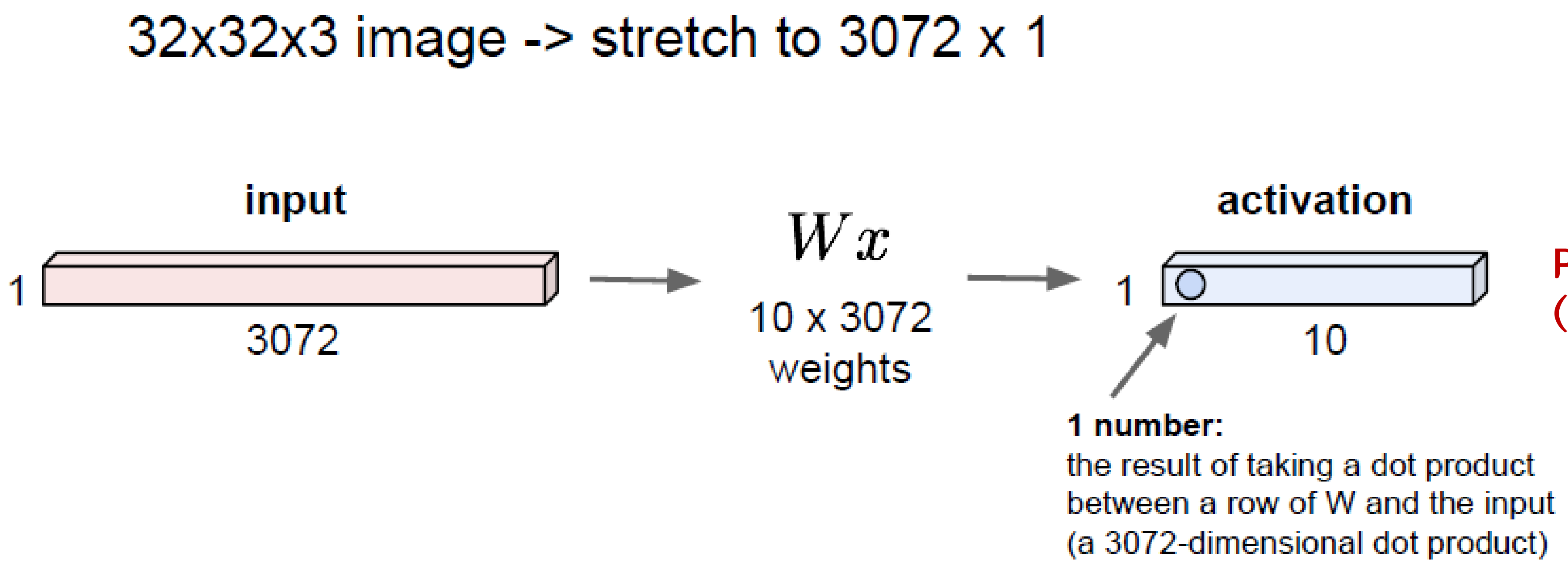
- Convolution
  - Filter 수가 6개라면 6개의 새로운 feature map이 생성됨. (hyper parameter)
  - 6개의 feature map을 stack up하여 28x28x6의 새로운 데이터 생성



Parameter 개수?
(weight + bias)

Unit 03 | Convolution Layer

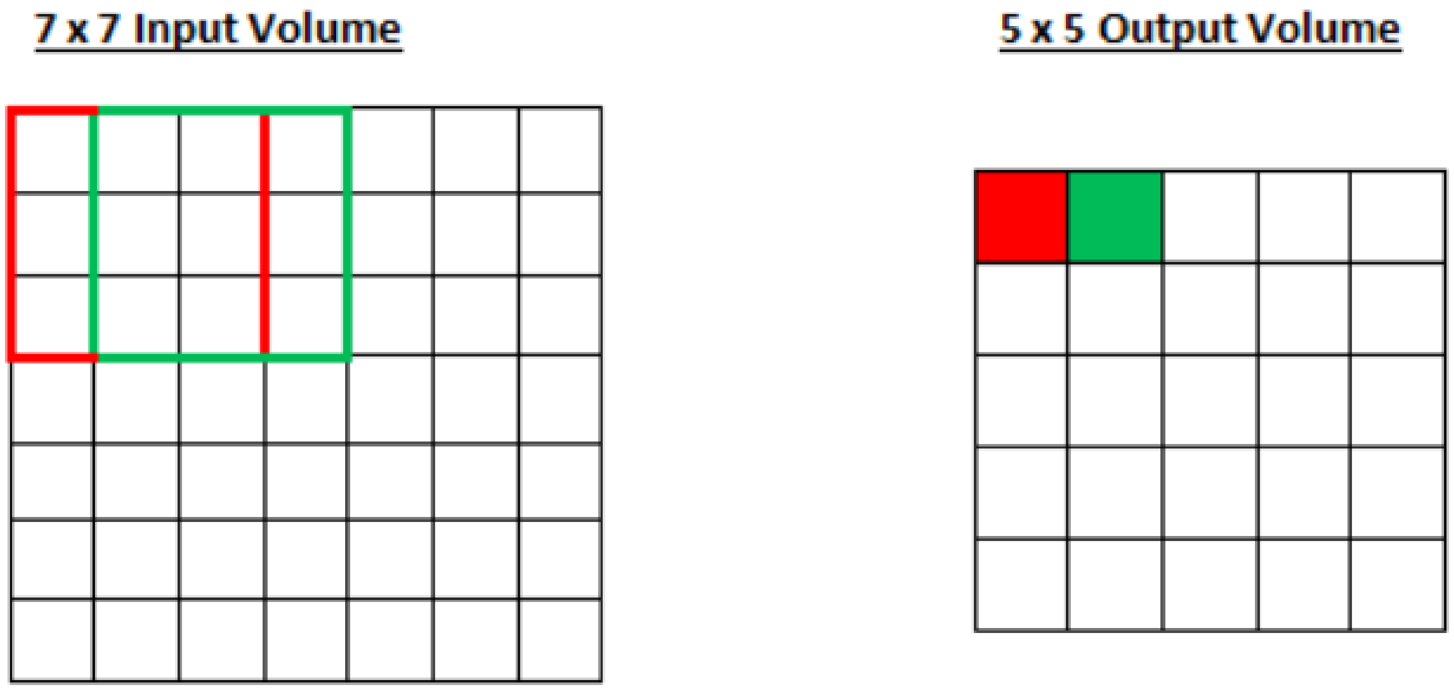
- Convolution
  - 각각의 pixel을 하나의 channel로 flatten 후 1x1x3072 filter 10개를 사용
  - FC?



Parameter 개수?
(weight + bias)

Unit 03 | Convolution Layer

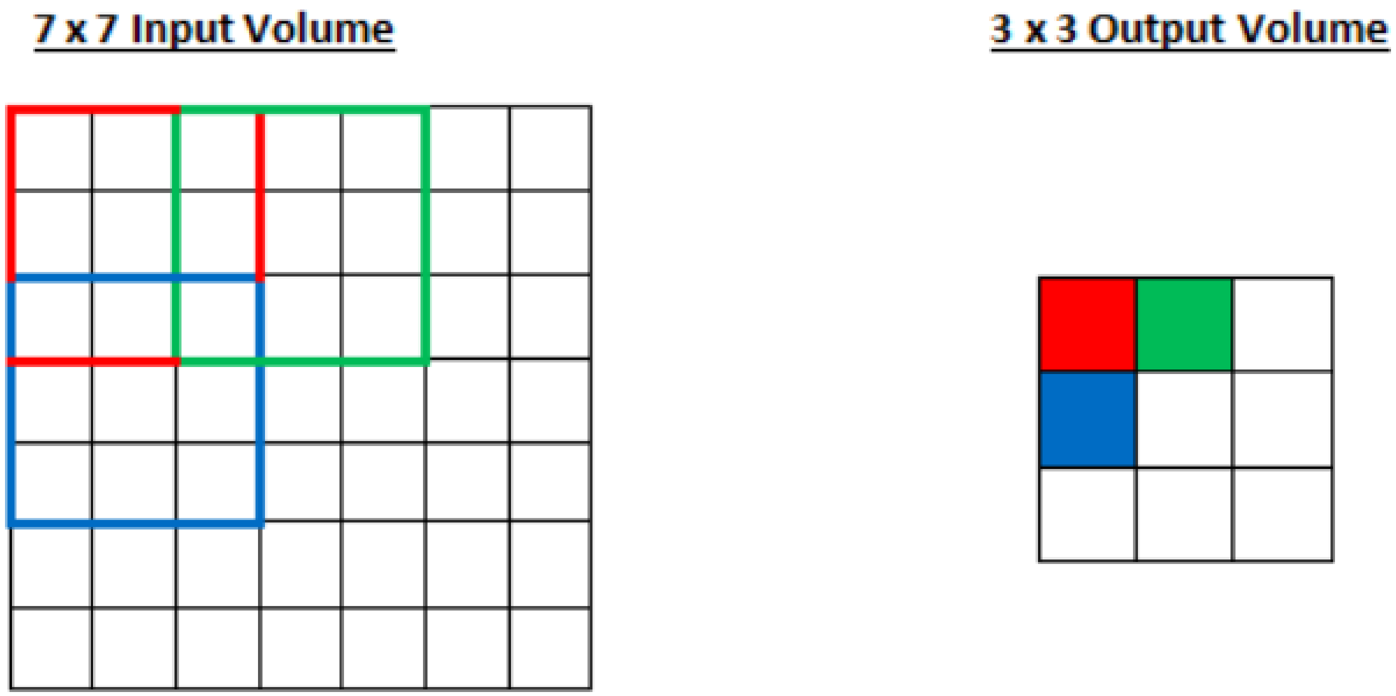
- Stride
  - Filter가 이동하는 거리



Stride = 1

Unit 03 | Convolution Layer

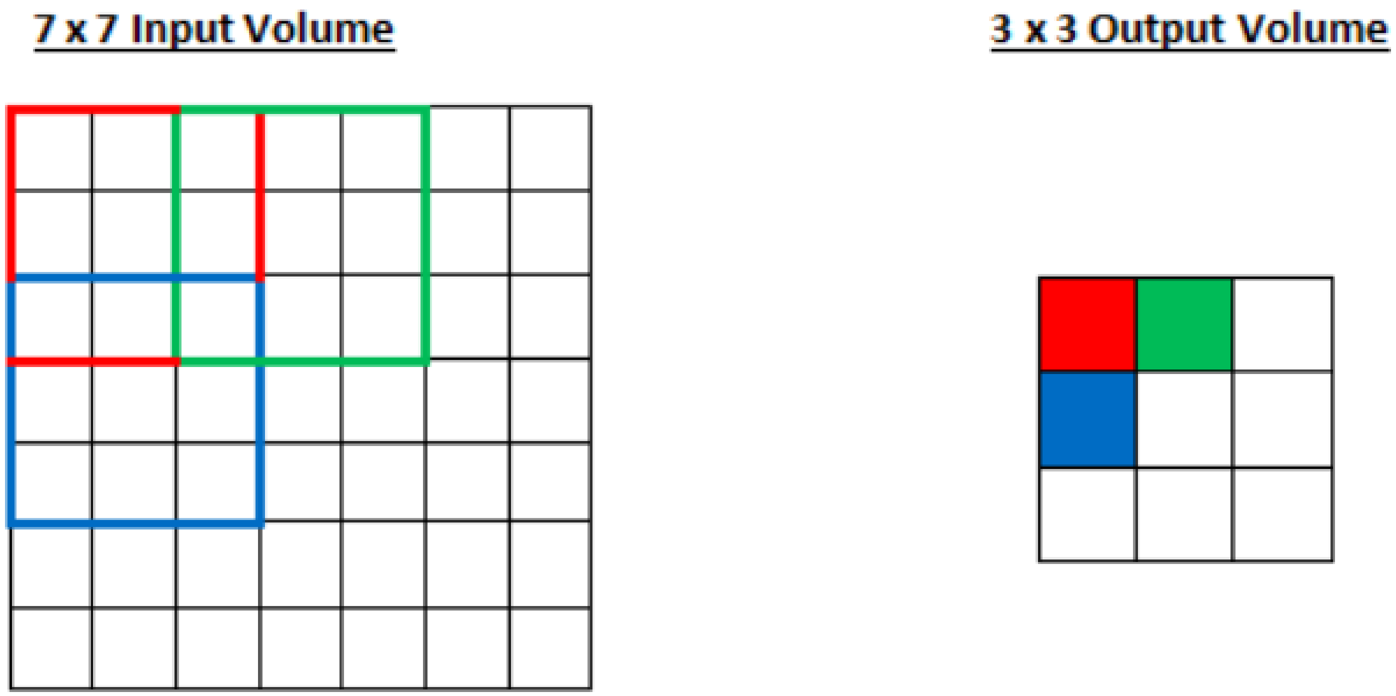
- Stride
  - Filter가 이동하는 거리



Stride = 2

Unit 03 | Convolution Layer

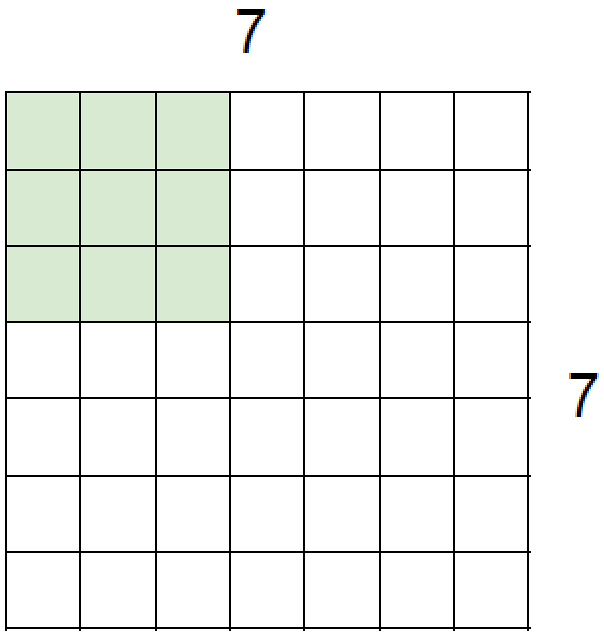
- Stride
  - Filter가 이동하는 거리



Stride = 2

Unit 03 | Convolution Layer

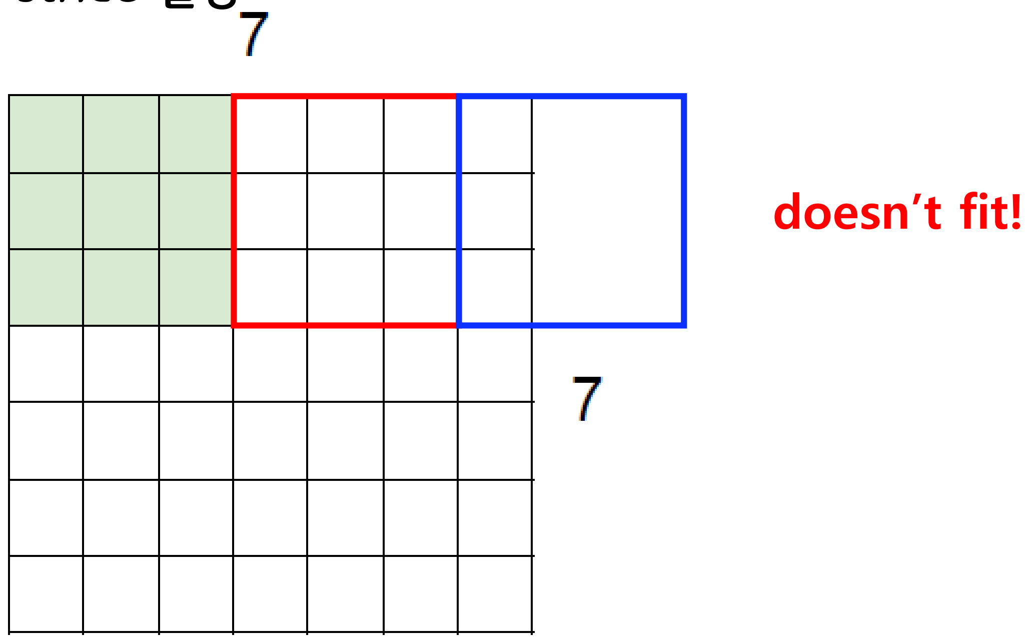
- Stride
  - Filter가 이동하는 거리



Stride = 3?

Unit 03 | Convolution Layer

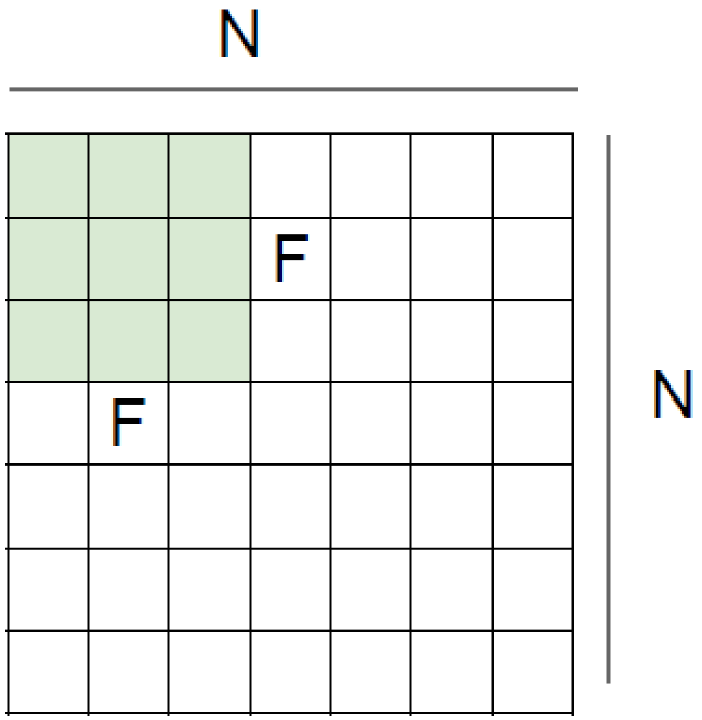
- Stride
  - Filter가 이동하는 거리
  - Output의 volume이 integer가 되도록 stride 설정



Stride = 3?

Unit 03 | Convolution Layer

- Stride
  - Filter가 이동하는 거리
  - Output의 volume이 integer가 되도록 stride 설정

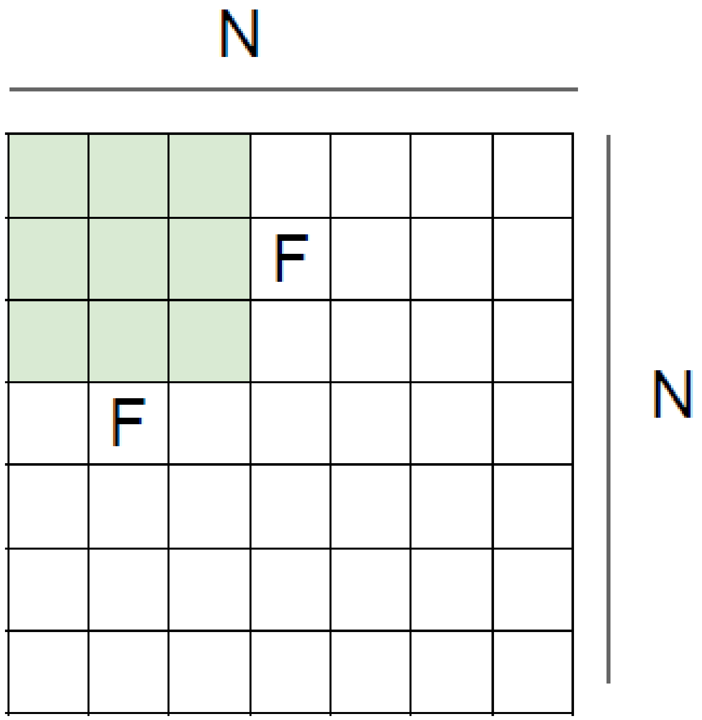


Output size:
(N - F) / stride + 1

e.g. 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 **Fraction!**

Unit 03 | Convolution Layer

- Stride
  - Filter가 이동하는 거리
  - Output의 volume이 integer가 되도록 stride 설정

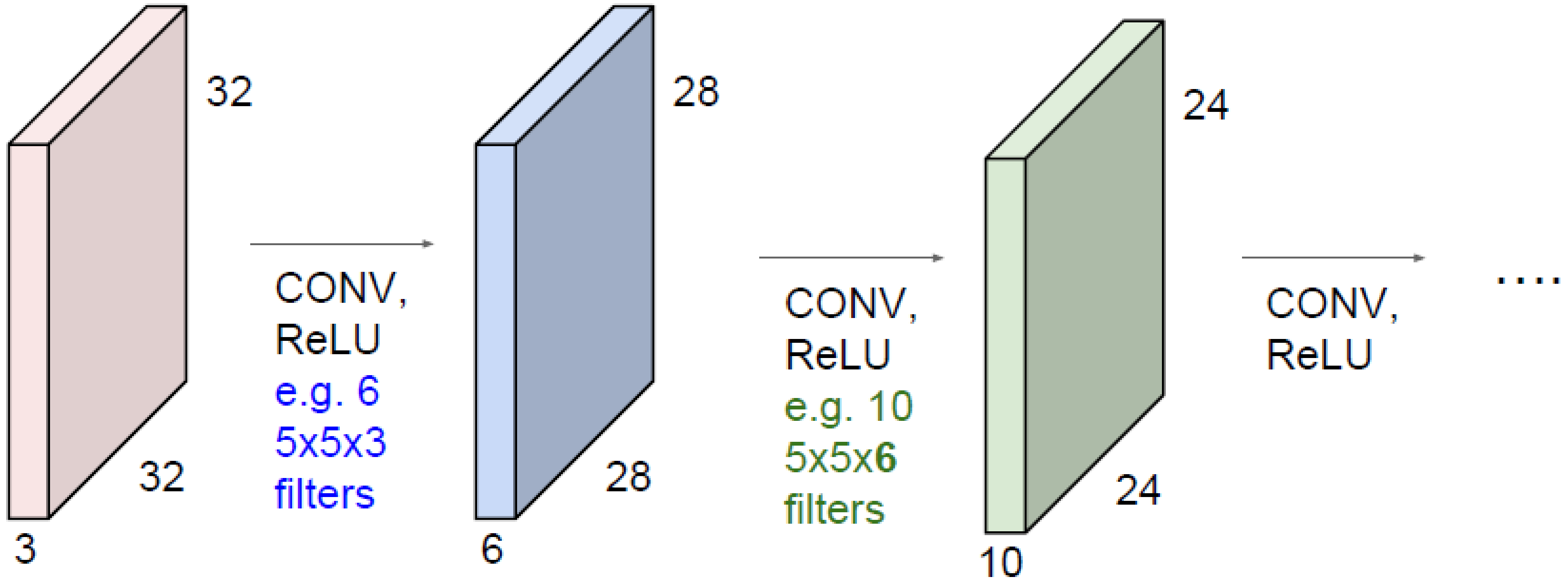


Output size:
(N - F) / stride + 1

e.g. 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 **Fraction!**

Unit 03 | Convolution Layer

- Padding
  - Convolution layer 층이 깊어질 수록 data의 size가 줄어든다.
  - Data size가 너무 빠르게 줄어들면 잘 작동하지 않음!



32 -> 28 -> 24 -> 20 -> 16 -> …

Unit 03 | Convolution Layer

- Padding
  - 데이터 테두리에 zero-padding을 더해 크기 손실 방지

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

e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

Unit 03 | Convolution Layer

- Padding
  - 데이터 테두리에 zero-padding을 더해 크기 손실 방지

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

e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(N + P – F) / stride + 1
= (7 + 2 – 3) / 1 + 1
= 7

Input size = Output size

Unit 03 | Convolution Layer

- Padding
  - 데이터 테두리에 zero-padding을 더해 크기 손실 방지

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

e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(N + P – F) / stride + 1
= (7 + 2 – 3) / 1 + 1
= 7

Input size = Output size

e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

- Padding
  - 데이터 테두리에 zero-padding을 더해 크기 손실 방지

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

e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(N + P – F) / stride + 1
= (7 + 2 – 3) / 1 + 1
= 7

Input size = Output size

e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

- Padding
  - 데이터 테두리에 zero-padding을 더해 크기 손실 방지

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

e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(N + P – F) / stride + 1
= (7 + 2 – 3) / 1 + 1
= 7

Input size = Output size

e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

Unit 03 | Convolution Layer

- Example

  Input volume: 32x32x3

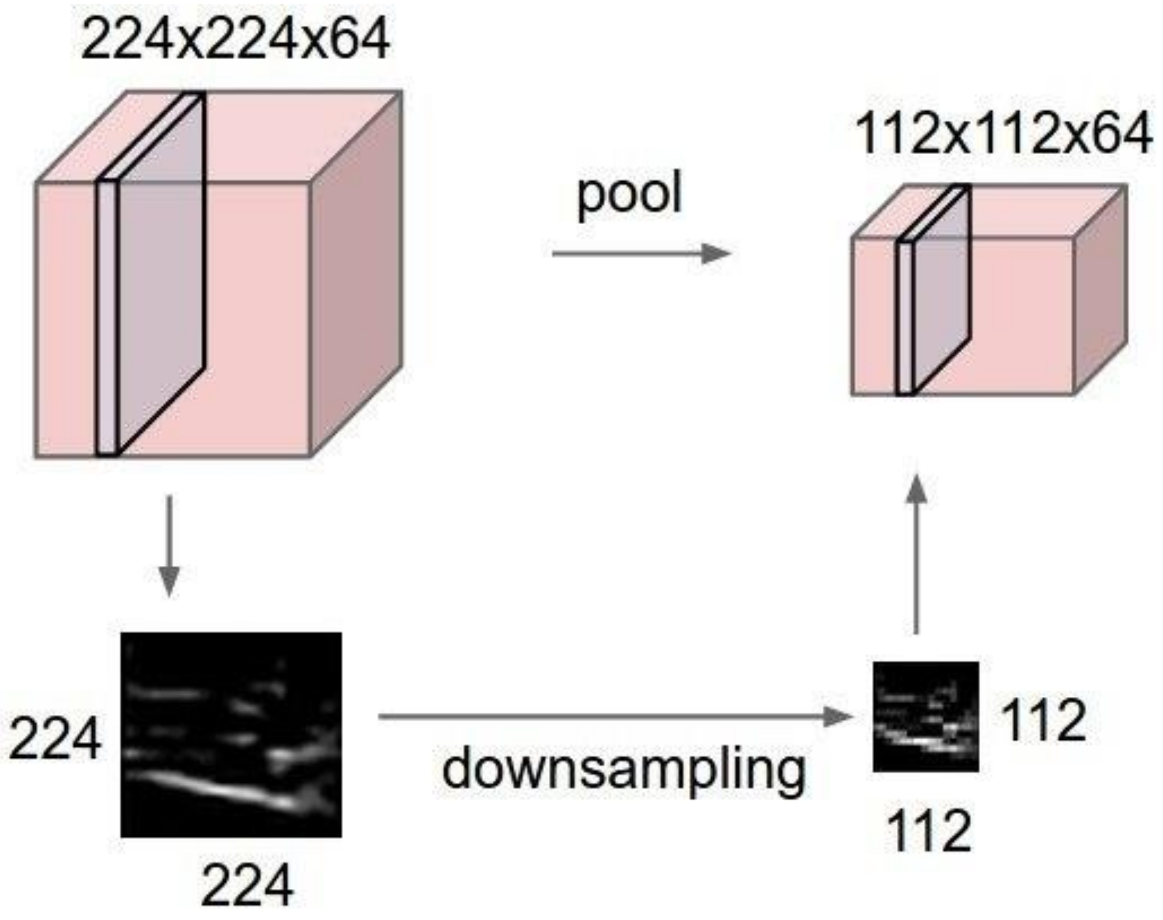
  10 5x5 filters with stride 1, pad 2

  1. Output volume size: ?

  2. Number of parameters in this layer : ?

Unit 04 | Sub-Sampling

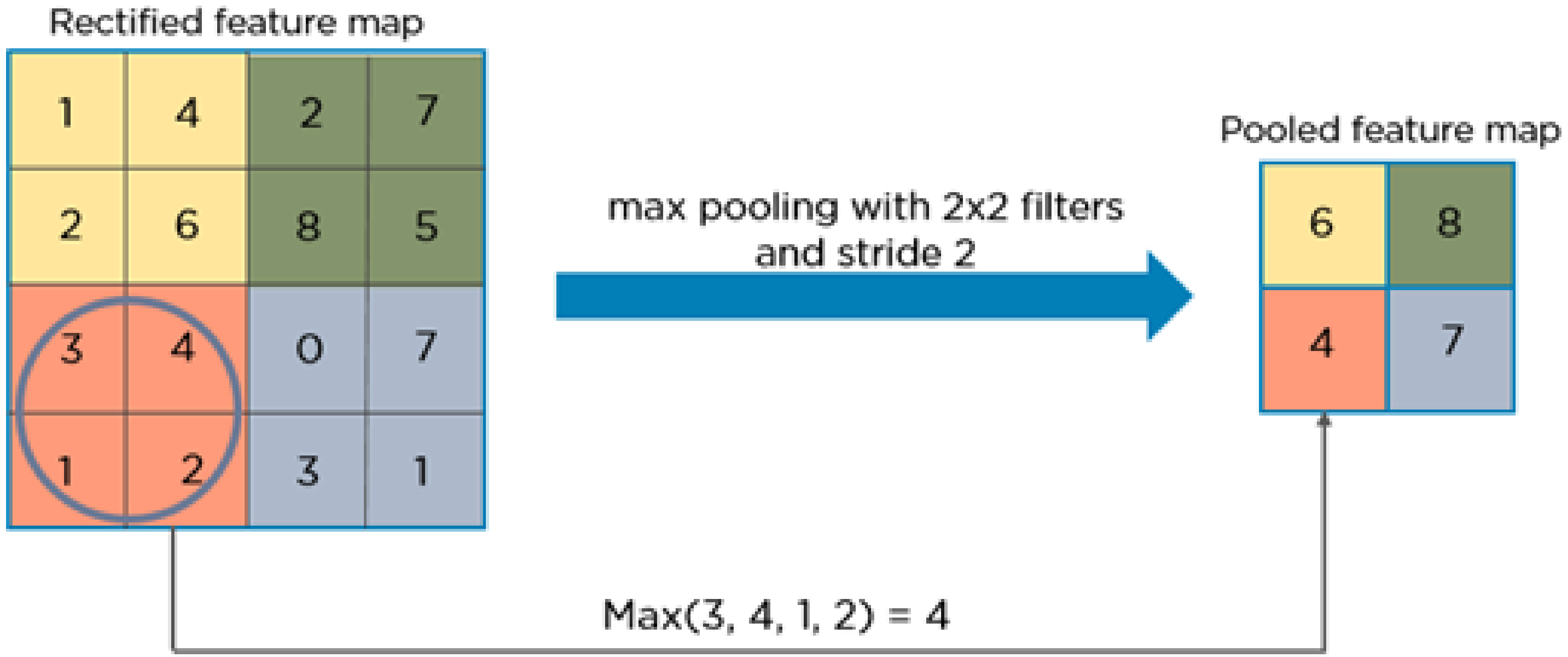
- Sub Sampling
  - 이미지의 특성은 유지
  - 사이즈를 줄여 관리하기 쉽게 만듬
  - 각 feature map마다 독립적으로 작용



But 기기의 성능 향상으로
사이즈를 줄일 필요가 적어지면서
성능을 위해 잘 사용하지 않는 추세

Unit 04 | Sub-Sampling

- Sub Sampling
  - Max pooling
    - Filter size와 stride 존재

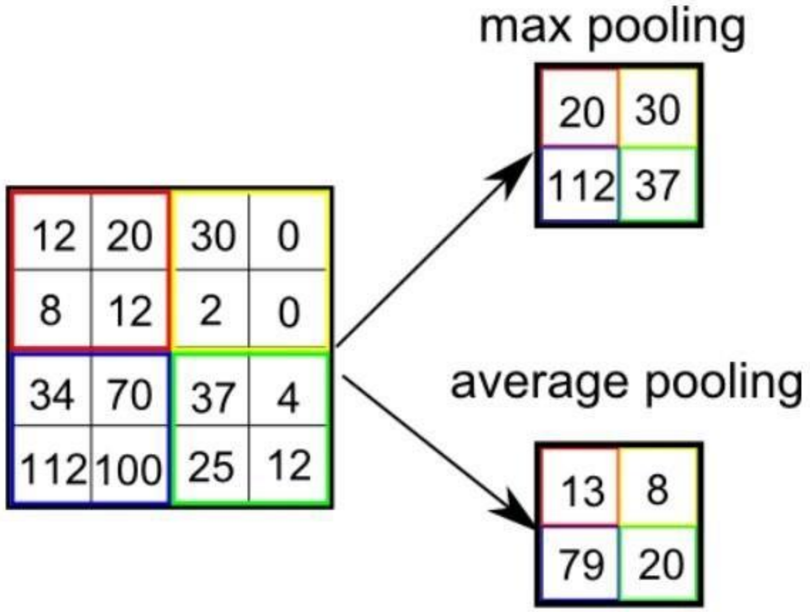


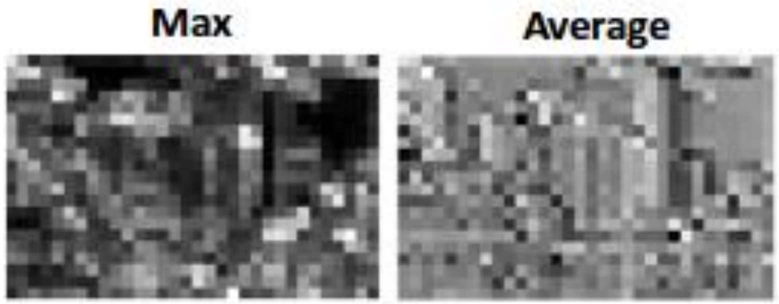
Unit 04 | Sub-Sampling

- Max pooling vs Average pooling
  - Max pooling

    해당 window의 max값을 추출
  - Average pooling

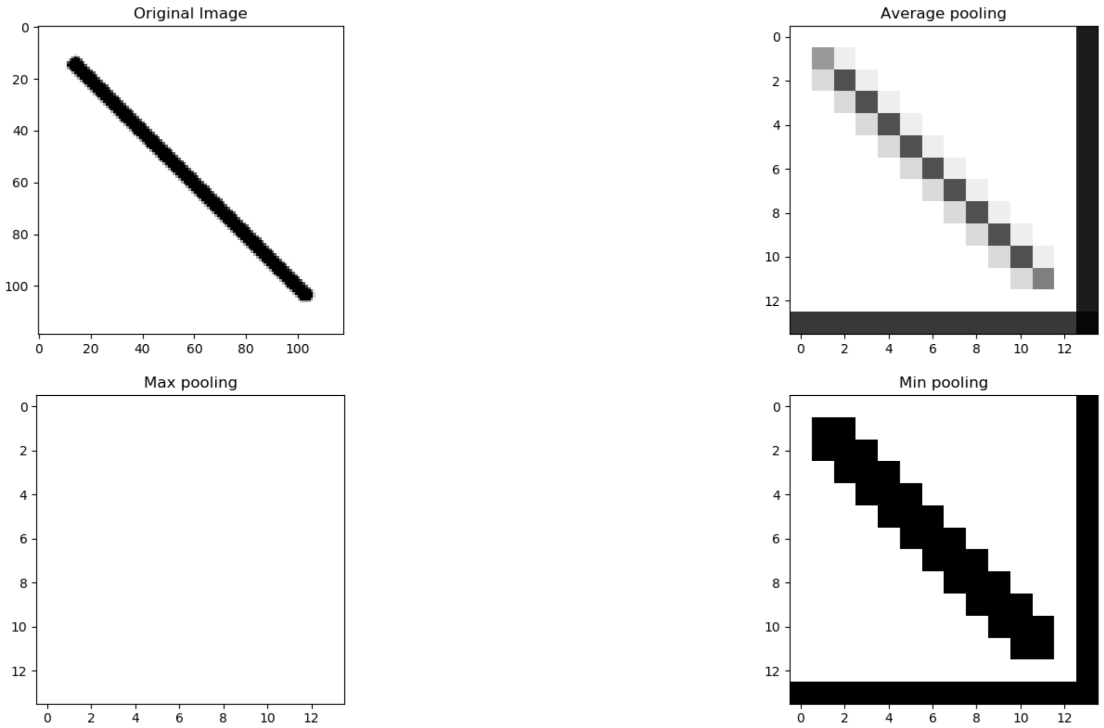
    해당 window의 평균 값 추출 (smoothing 됨)

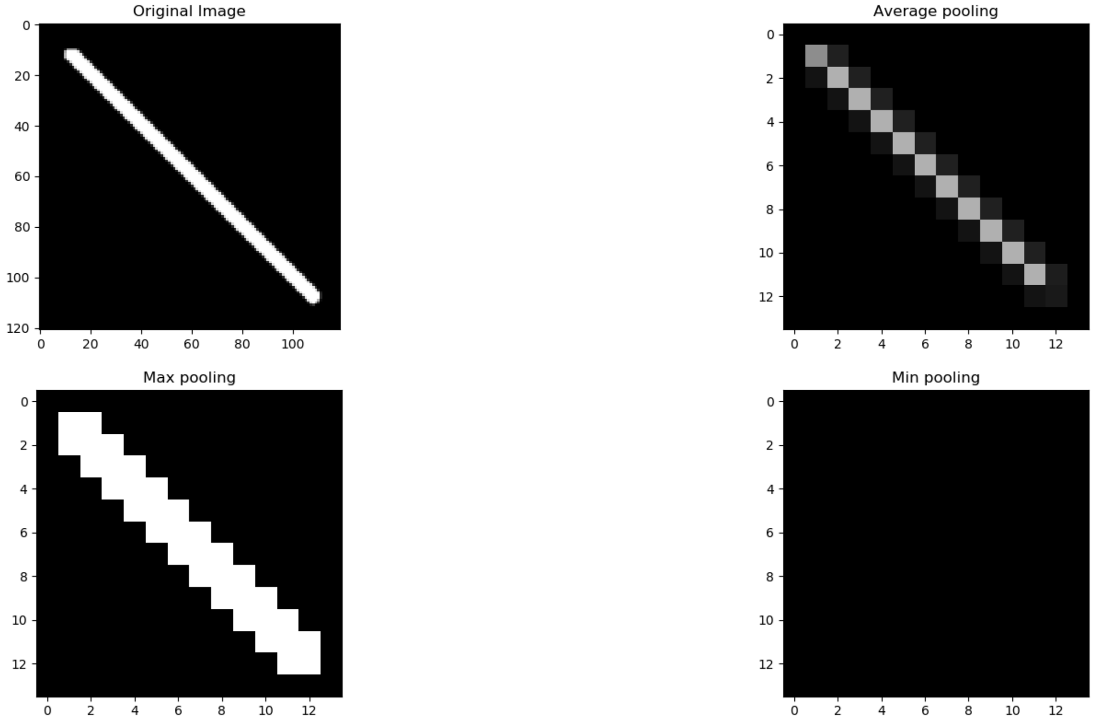




Unit 04 | Sub-Sampling

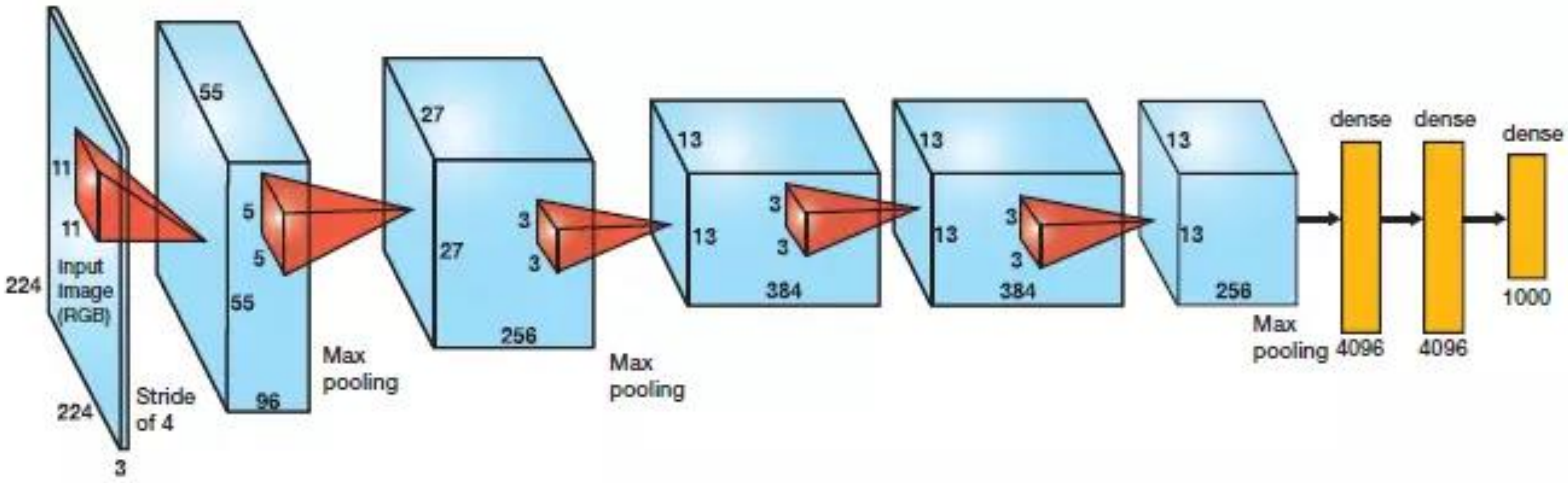
- Max pooling vs Average pooling
  - Most important features를 뽑는다는 관점에서 일반적으로 Max pooling을 사용



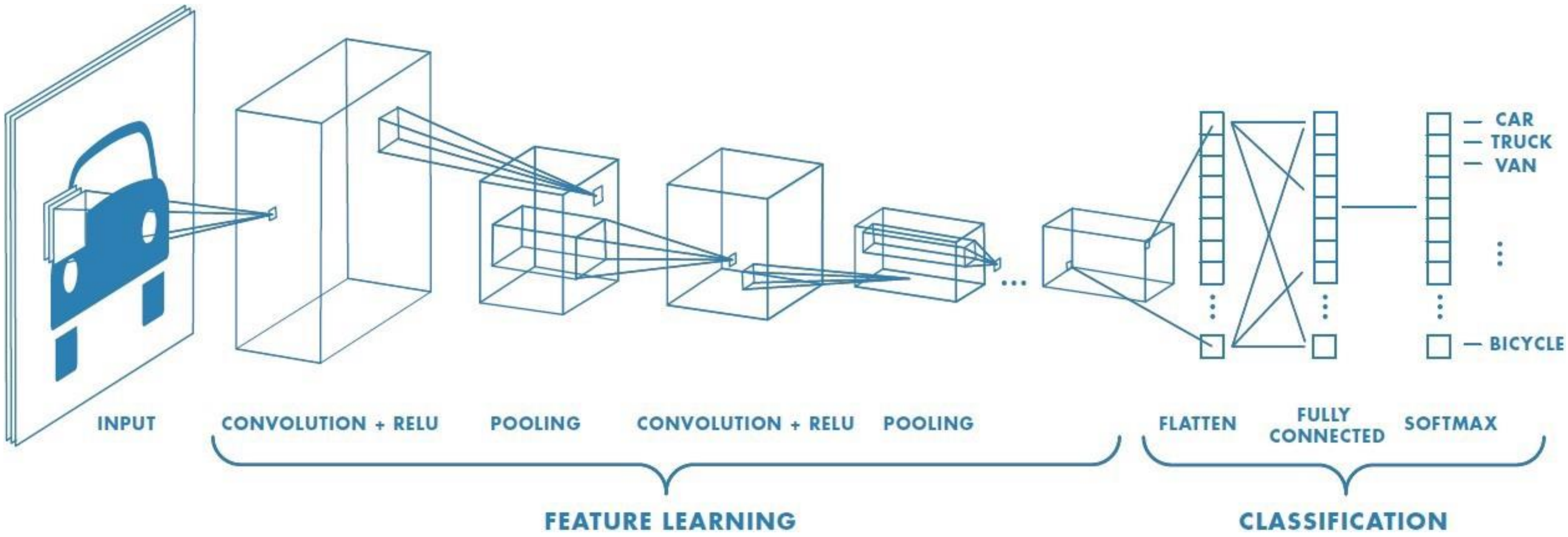


Unit 04 | Sub-Sampling

- pretty much everything of 'CNN' (AlexNet)



Unit 05 | Summary



- Convolution Layer + Subsampling Layer + Fully Connected Layer

- Feature extraction : Convolution Layer + Pooling Layer
- Classification : Fully Connected Layer

Unit 02 | Layers in CNN

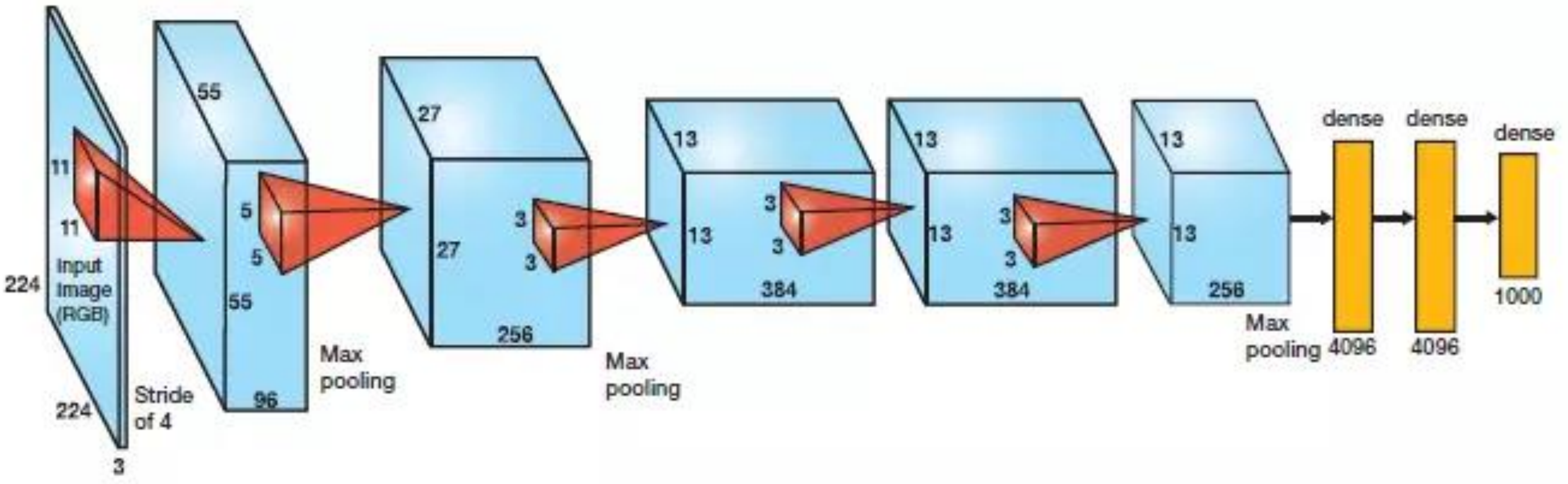
- Convolutional Neural Network
  - Local connectivity(receptive field)
    - 지역적으로 뉴런을 연결하여 다양한 local feature 추출 가능
  - Shared Weights and Biases (topology invariance)
    - Filter의 weight을 공유하여 parameter를 획기적으로 줄일 수 있다.
    - 찾고자 하는 특징이 이미지 어디에 위치해도 알 수 있다.
  - Compositionality
    - 저레벨 특징을 고레벨 특징으로 compose함
    - ex) 눈의 특징, 귀의 특징, 코의 특징 등등이 결합하여 사람의 얼굴의 특징으로 귀결

Unit 05 | Summary

- 용어 정리
  - Convolution(합성곱)
  - 채널(Channel)
  - 필터(Filter) = 커널(Kernel)
  - 스트라이드(Stride)
  - 패딩(Padding), zero-Padding
  - 피처 맵(Feature Map) = 액티베이션 맵(Activation Map)
  - 풀링(Pooling) 레이어
  - receptive field(수용공간)

Assignment

AlexNet



과제 1. assignment_1.ipynb 물음표 채우기

과제 2. AlexNet model 구현 (프레임워크 자유)
-모델 구현 후 summary로 전체 모델 구조 보이고 주석을 통해 간단한 설명
(각 프레임워크 별 summary 방법 구글링)

Q & A

들어주셔서 감사합니다.

Appendix

- 참고자료

\# 10기 박성진님 강의

http://www.datamarket.kr/xe/index.php?mid=board_jPWY12&page=2&document_srl=52335

\# Stanford cs231n 강의

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