

19기 정규세션

ToBig's 18기 강의자
이선민

Vision Basic

Content

Unit 01 | Vision

Unit 02 | Convolution

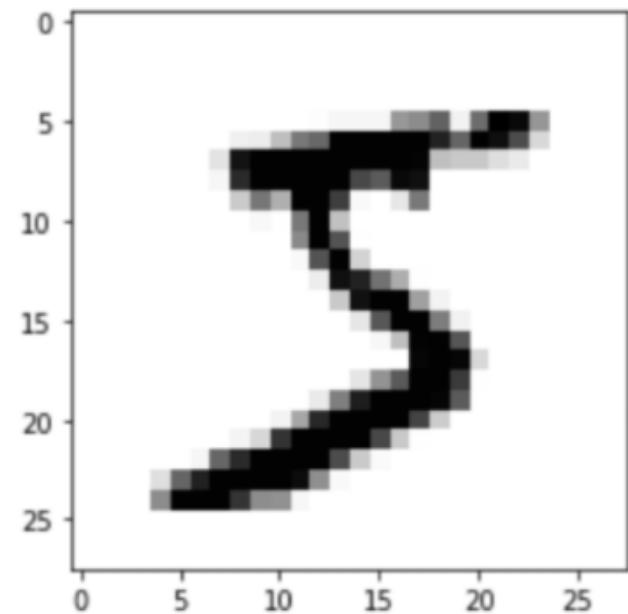
Unit 03 | Convolutional Neural Network

Unit 04 | Number of Parameters

Unit 01 | Vision

Vision

Unit 01 | Vision



Fully Connected Network로 이미지를 다루면 픽셀 하나하나가 입력

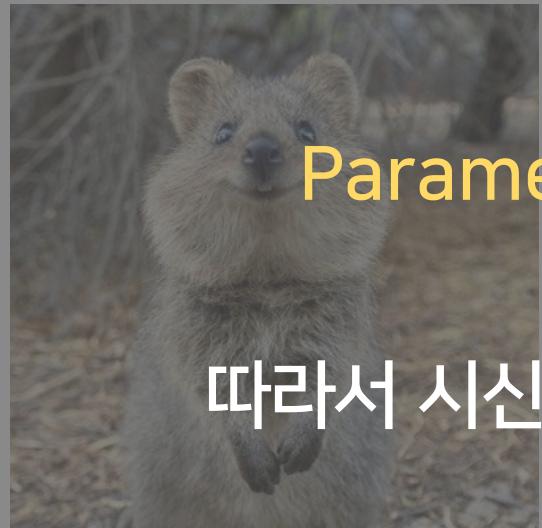
Unit 01 | Vision



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

컬러 이미지의 경우, 각 픽셀마다 RGB(red,green,blue)에 해당하는 값을 3개씩 가진다.

Unit 01 | Vision



Parameter의 개수가 너무 많다!

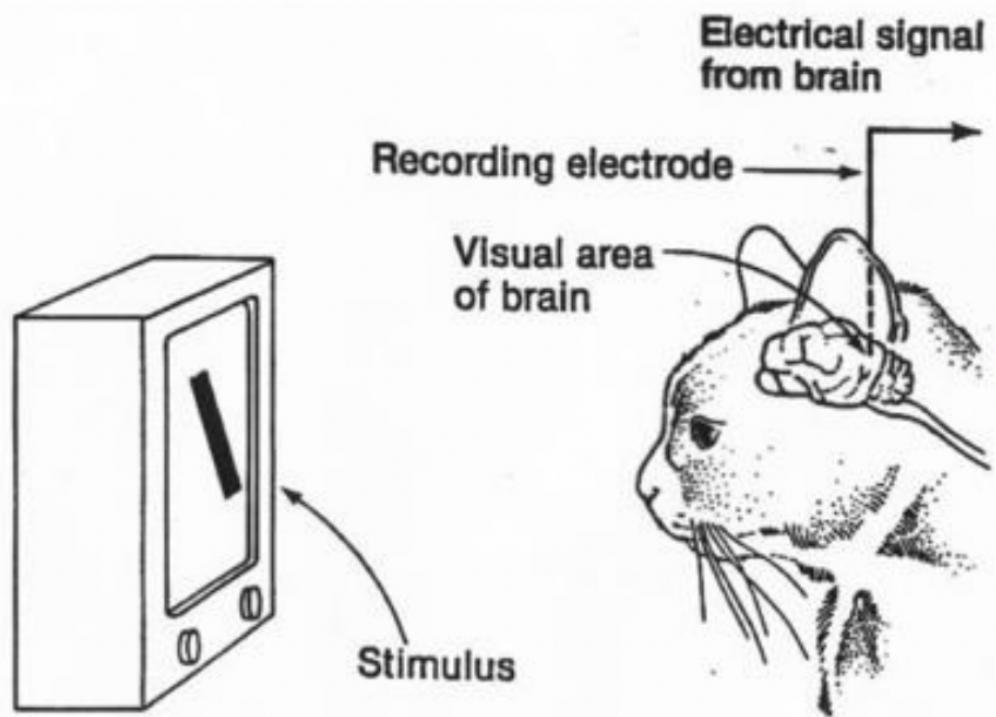


따라서 시신경 구조를 모방한 CNN 등장

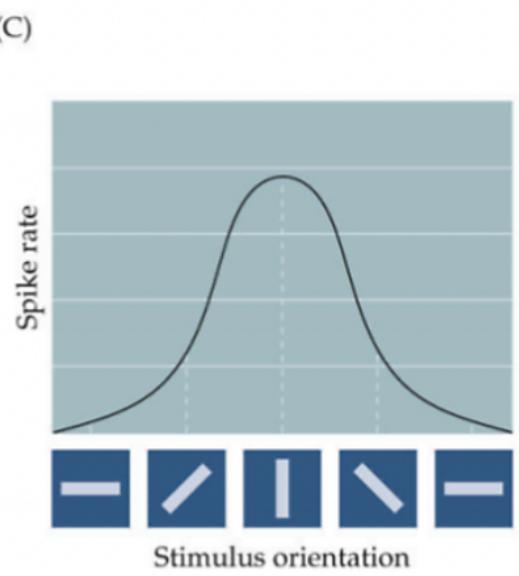
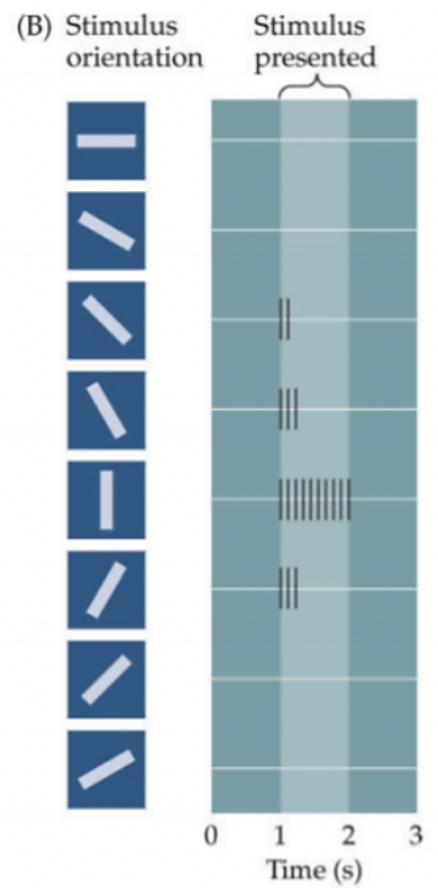
1	0	2	0	1	0	3
0	2	0	0	0	0	1
2	3	0	1	3	2	1
1	2	1	1	1	0	0
3	2	2	0	0	2	3
0	0	0	2	2	2	2
1	3	2	3	2	1	0

컬러 이미지의 경우, 각 픽셀마다 RGB(red,green,blue)에 해당하는 값을 3개씩 가진다.

Unit 01 | Vision



Hubel & Wiesel's Experiment



Unit 02 | Convolution

Convolution

Unit 02 | Convolution

■ Convolution 연산

Convolution 연산은 signal을 kernel을 이용해 국소적으로 증폭 또는 감소시켜 정보를 추출 또는 필터링하는 것을 의미.

continuous $[f * g](x) = \int_{\mathbb{R}^d} f(z)g(x - z)dz = \int_{\mathbb{R}^d} f(x - z)g(z)dz = [g * f](x)$

discrete $[f * g](i) = \sum_{a \in \mathbb{Z}^d} f(a)g(i - a) = \sum_{a \in \mathbb{Z}^d} f(i - a)g(a) = [g * f](i)$

Unit 02 | Convolution

■ Convolution 연산

Convolution은 해당하는 요소끼리 곱하고 결과를 모두 더하는 선형 연산.

CNN에서 사용하는 연산은 cross-correlation을 사용하지만 관용적으로 convolution으로 부른다.

continuous $[f * g](x) = \int_{\mathbb{R}^d} f(z)g(x+z)dz = \int_{\mathbb{R}^d} f(x+z)g(z)dz = [g * f](x)$

discrete $[f * g](i) = \sum_{a \in \mathbb{Z}^d} f(a)g(i+a) = \sum_{a \in \mathbb{Z}^d} f(i+a)g(a) = [g * f](i)$

Unit 02 | Convolution

■ 다양한 차원의 Convolution

$$[f * g](i) = \sum_{p=1}^d f(p)g(i+p) \quad \longleftarrow \quad 1\text{D-conv}$$

$$[f * g](i, j) = \sum_{p,q} f(p, q)g(i+p, j+q) \quad \longleftarrow \quad 2\text{D-conv}$$

$$[f * g](i, j, k) = \sum_{p,q,r} f(p, q, r)g(i+p, j+q, k+r) \quad \longleftarrow \quad 3\text{D-conv}$$

데이터에 따라 사용하는 커널이 달라진다.

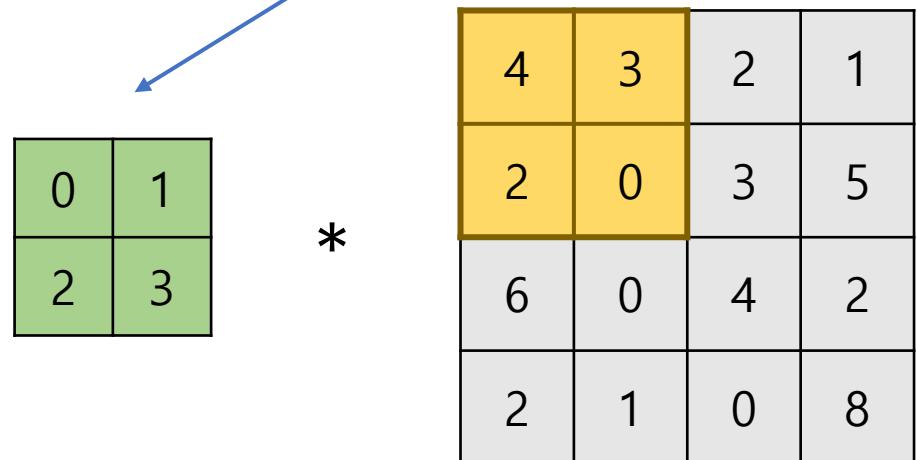
좌표계 (i, j, k) 가 바뀌어도 커널 f 의 값은 바뀌지 않는다.

Unit 02 | Convolution

■ 2D-Convolution

고정된 커널(kernel)을 입력벡터 상에서 이동해가며 선형모델과 합성함수를 적용

$$[f * g](i, j) = \sum_{p,q} f(p, q)g(i + p, j + q)$$



$$0 \times 4 + 1 \times 3 + 2 \times 2 + 3 \times 0 = 7$$

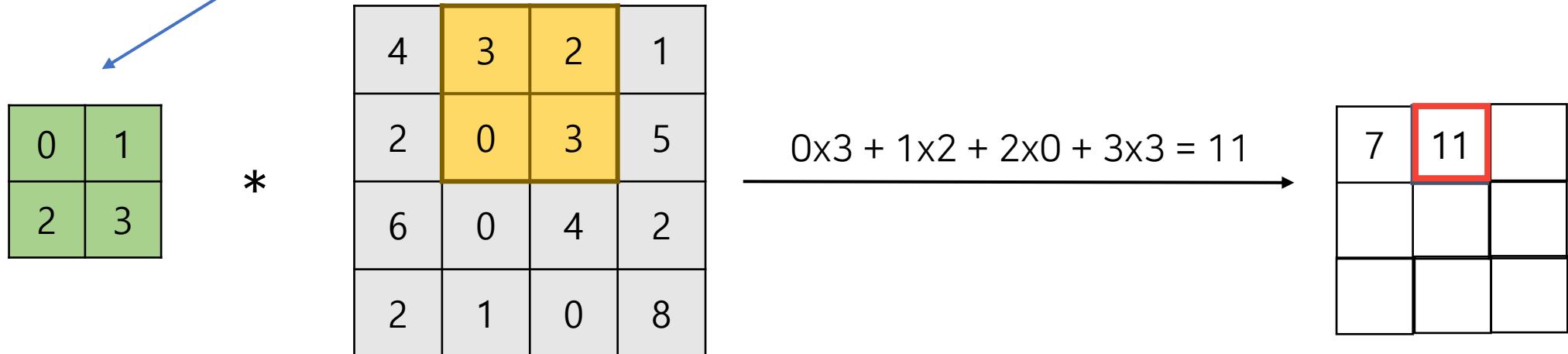
7		

Unit 02 | Convolution

■ 2D-Convolution

입력만 바뀌고 커널은 바뀌지 않는다.

$$[f * g](i, j) = \sum_{p,q} f(p, q)g(i + p, j + q)$$



Unit 02 | Convolution

■ 2D-Convolution

입력의 크기와 커널의 크기를 통해 출력의 크기를 계산할 수 있다.

$$O_H = H - K_H + 1$$

$$O_W = W - K_W + 1$$

입력 사이즈: (H, W) , 커널 사이즈: (K_H, K_W) , 출력 사이즈: (O_H, O_W)

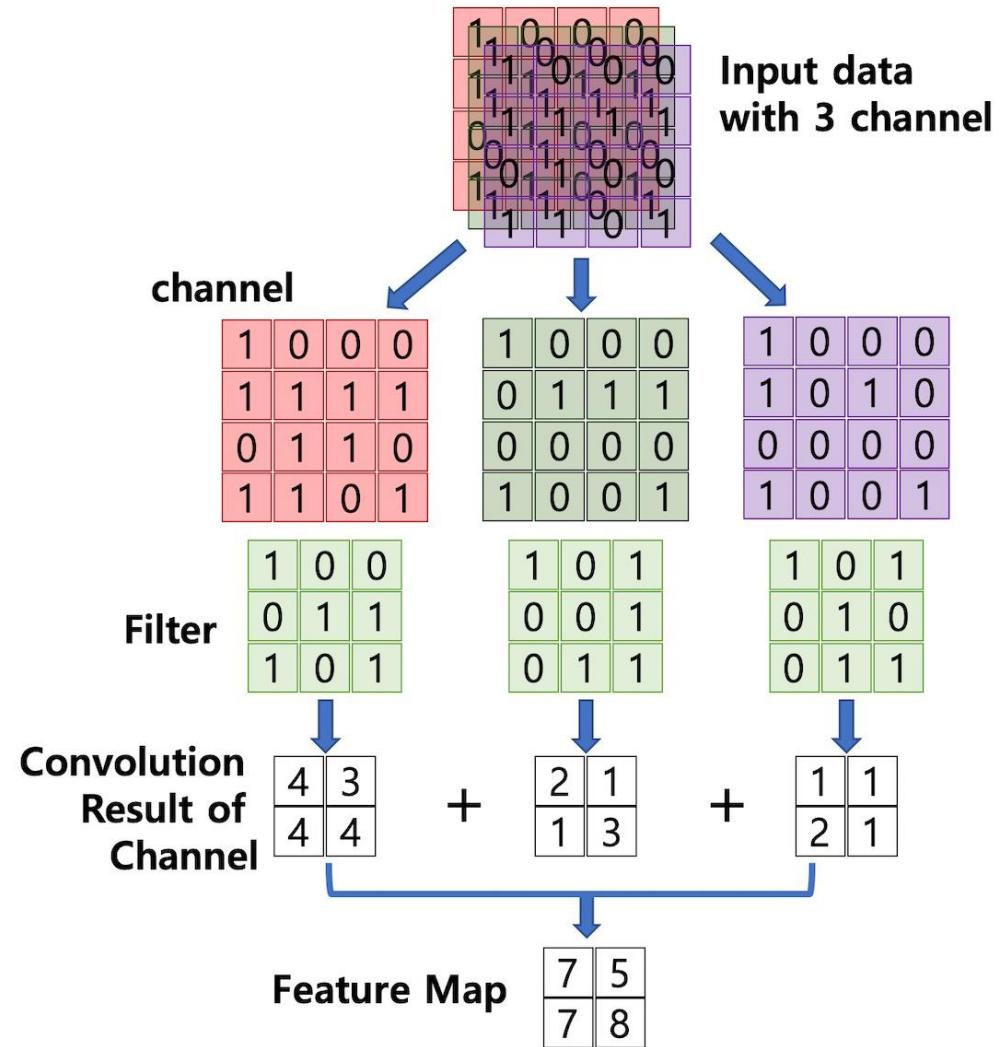
Ex) 입력 사이즈: 28x28, 커널 사이즈: (3x3) => 출력 사이즈: (26x26)

Unit 02 | Convolution

■ 채널이 여러 개인 2차원 입력

채널 개수만큼 2차원 convolution을 적용하고,
생성된 채널 별 feature map을 element wise sum하여
output의 feature map을 구한다.

=> output의 채널은 1

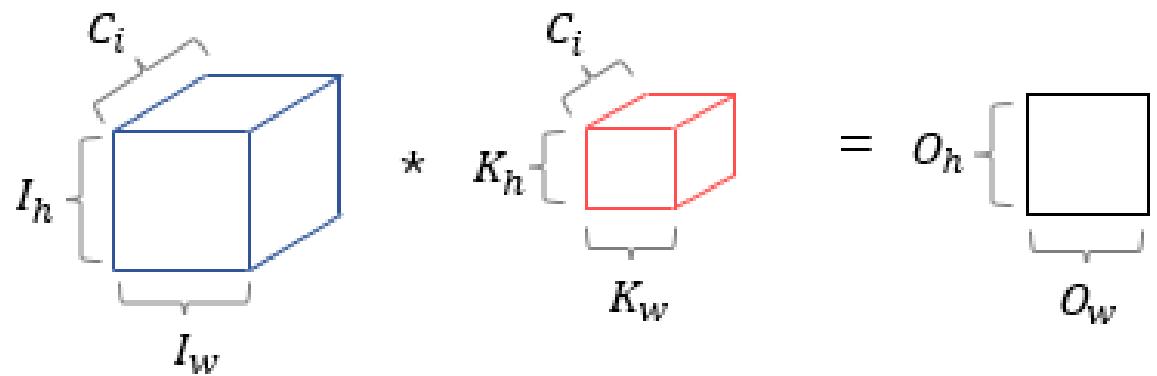


Unit 02 | Convolution

■ 채널이 여러 개인 2차원 입력

채널이 여러 개인 2차원 입력 tensor를 블록으로 표현

입력의 채널 = 커널의 채널 = C_i



I_h : 입력의 높이

I_w : 입력의 너비

C_i : 입력 데이터의 채널

K_h : 커널의 높이

K_w : 커널의 너비

C_i : 입력 데이터의 채널

O_h : 특성 맵의 높이

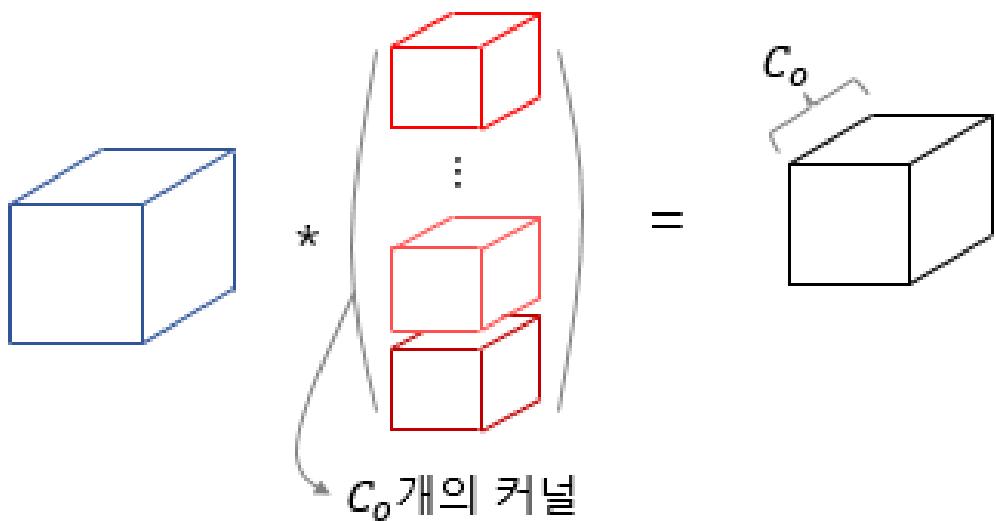
O_w : 특성 맵의 너비

Unit 02 | Convolution

■ 채널이 여러 개인 2차원 입력

커널의 개수 = output의 depth = C_o

커널을 여러 개 사용하면 출력도 tensor가 된다.



Unit 02 | Convolution

■ Fully Connected Layer vs. Convolution Layer

• FC layer

각 뉴런들이 선형모델과 활성함수로 모두 연결된 구조

각 성분 h_i 에 대응하는 가중치 행 W_i 존재

$$\boxed{y = Wx + b} \quad h_i = \sigma \left(\sum_{j=1}^p W_{ij} x_j \right)$$

• Conv layer

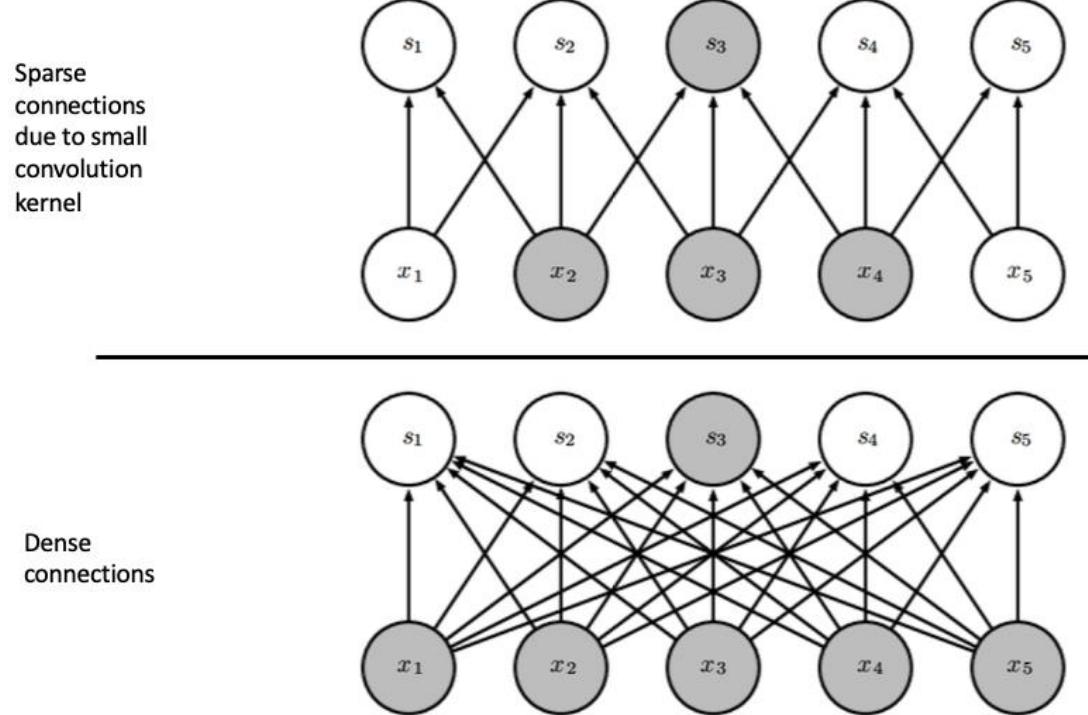
고정된 커널을 입력벡터 상에서 이동해가며 선형모델과 활성함수가 적용되는 구조

$$h_i = \sigma \left(\sum_{j=1}^k V_j x_{i+j-1} \right)$$

Unit 02 | Convolution

■ Fully Connected Layer vs. Convolution Layer

1. Sparse Connectivity



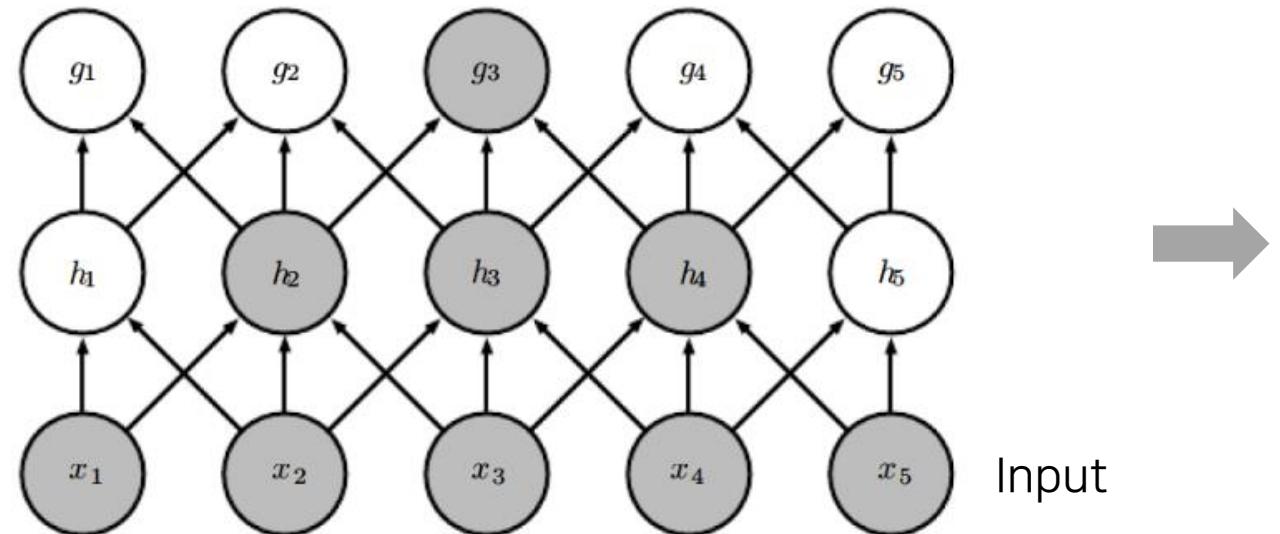
Parameter의 개수가 많이 줄어드는 효과
하지만 receptive field가 좁아지는 문제

receptive field: 출력 레이어의 뉴런 하나에 영향을 미치는 입력 뉴런들의 공간 크기

Unit 02 | Convolution

- Fully Connected Layer vs. Convolution Layer

1. Sparse Connectivity



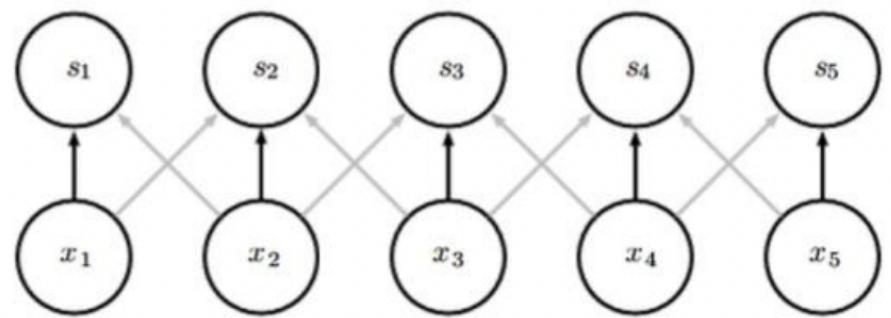
Layer를 여러 개 쌓음으로써 문제 해결
(Growing Receptive Fields)

Unit 02 | Convolution

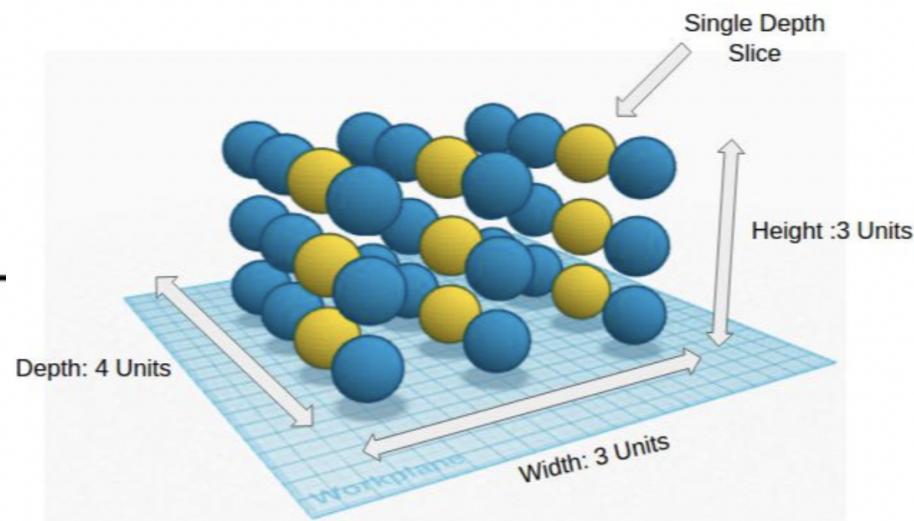
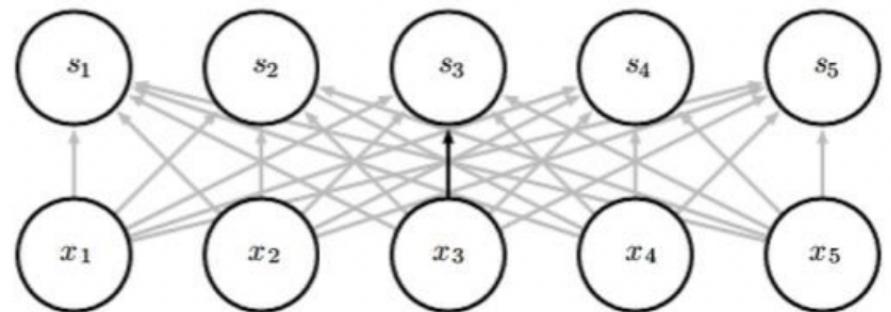
■ Fully Connected Layer vs. Convolution Layer

2. Parameter Sharing

Convolution shares the same parameters across all spatial locations



Traditional matrix multiplication does not share any parameters



Unit 02 | Convolution

■ Fully Connected Layer vs. Convolution Layer

Efficiency of Convolution

Input size : 320 x 280 / Kernel size: 2 x 1 / output size : 319 x 280

	CNN	FCN
Memory (# of params)	2	$\frac{\text{output 픽셀 수}}{319 \times 280} \times \frac{\text{Input 픽셀 수}}{320 \times 280}$ $\approx 8 \times 10^9$
Computation (# of muls & adds)	$\frac{\text{output 픽셀 수}}{319 \times 280 \times 3} \times \frac{\text{곱하기 2}}{\text{더하기 1}}$ $= 267,960$	$\frac{\text{output 픽셀 수}}{319 \times 280} \times \frac{\text{Input 픽셀 수}}{320 \times 280} \times 2$ $\approx 16 \times 10^9$ 곱하기 1 더하기 1

Unit 03 | Convolutional Neural Network

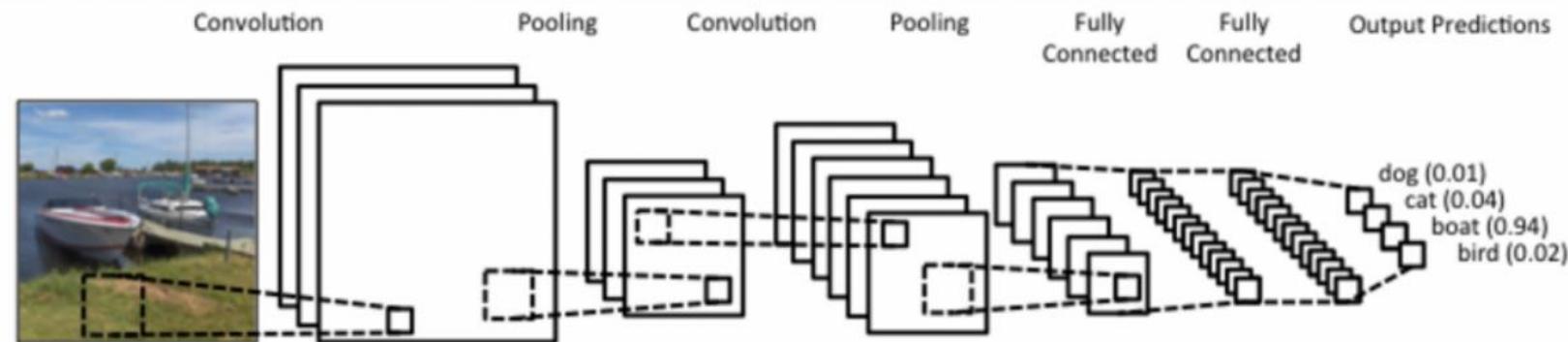
Convolutional Neural Network

Unit 03 | Convolutional Neural Network

■ CNN의 구조

CNN은 크게 Convolution layer, Pooling layer, Fully connected layer로 구성된다.

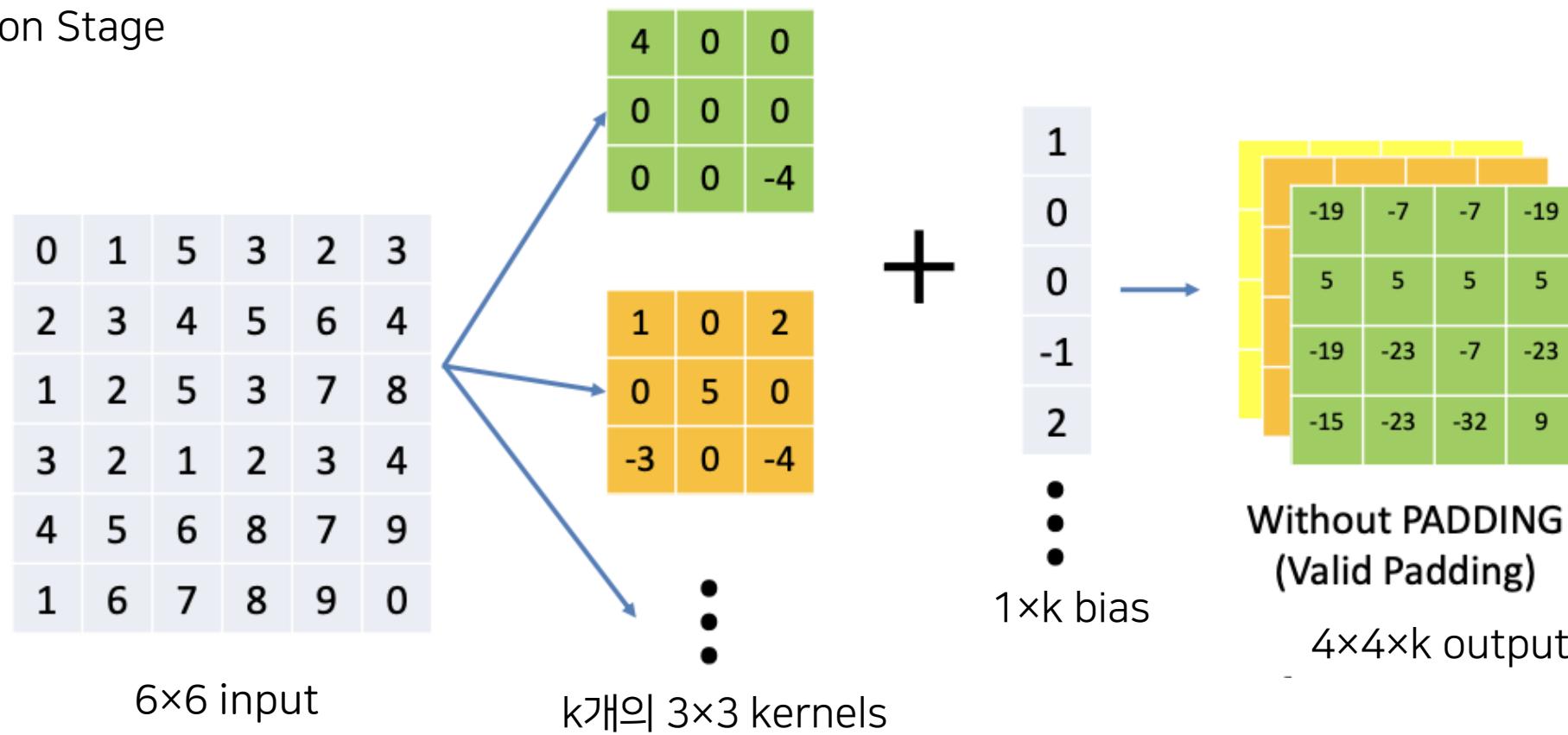
- Convolution layer, Pooling layer → feature extraction
- Fully connected layer → classification



Unit 03 | Convolutional Neural Network

■ CNN의 구조

Convolution Stage



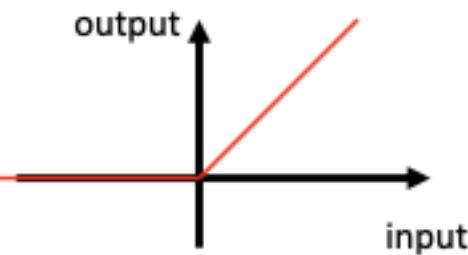
Unit 03 | Convolutional Neural Network

■ CNN의 구조

Detector Stage

-19	-7	-7	-19	
5	5	5	5	
-19	-23	-7	-23	
-15	-23	-32	9	

4×4 ×k input feature map



ReLU(Retified Linear Unit)

activation functions

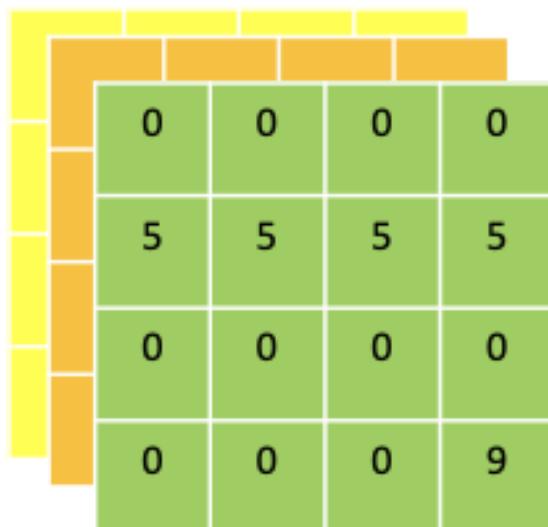
0	0	0	0	
5	5	5	5	
0	0	0	0	
0	0	0	9	

4×4 ×k output feature map

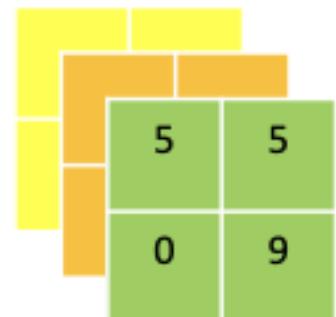
Unit 03 | Convolutional Neural Network

■ CNN의 구조

Pooling Stage



4×4×k input

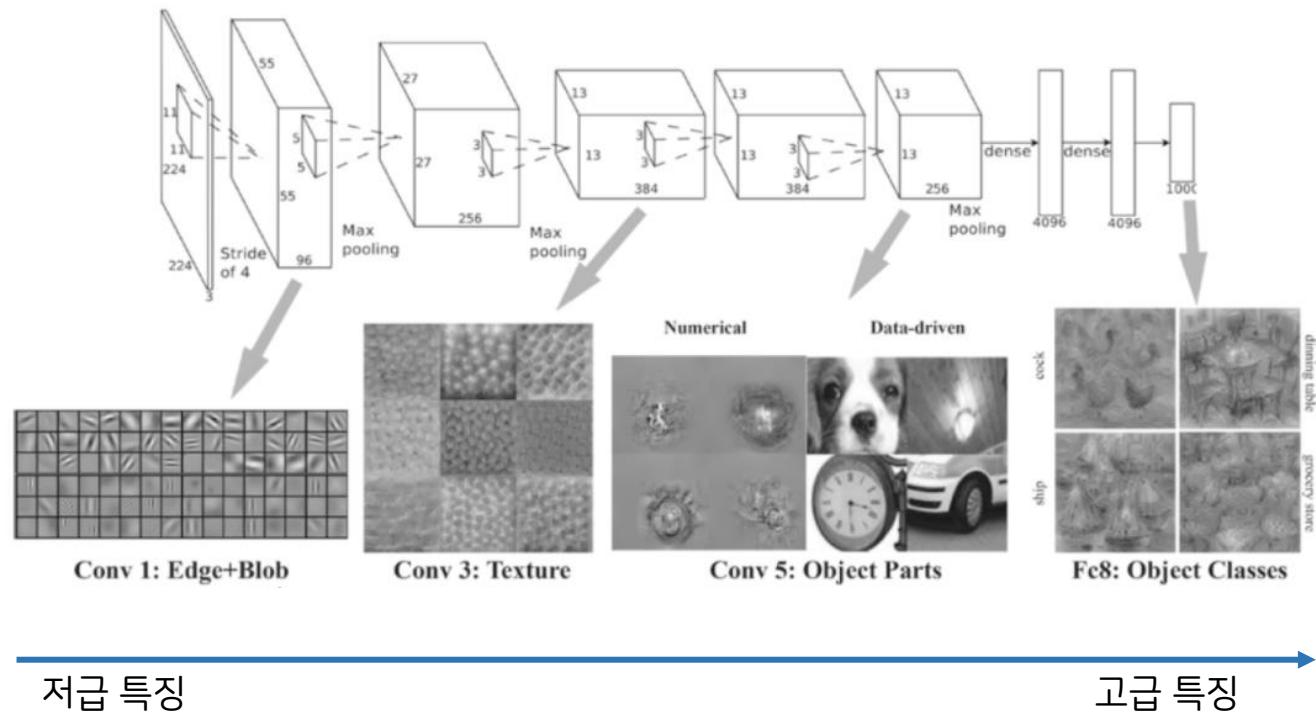
 2×2 max pooling

2×2×k output

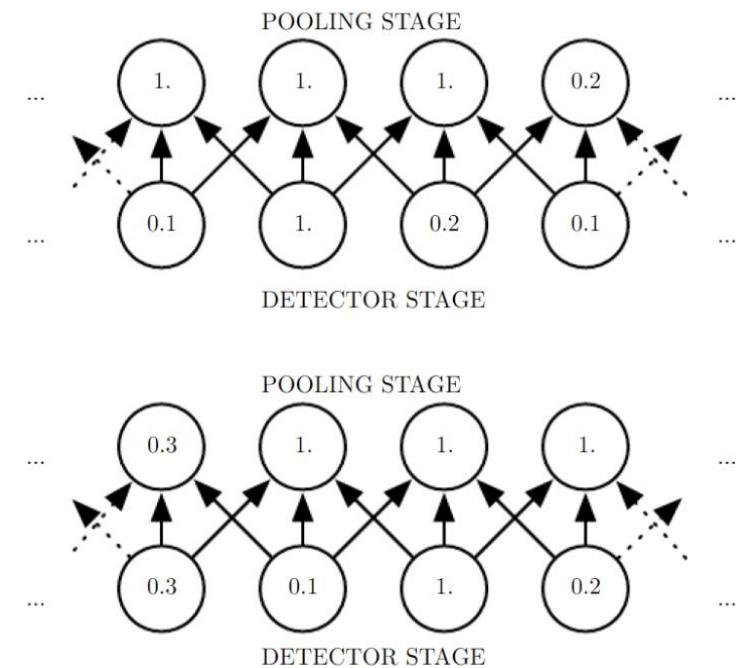
Unit 03 | Convolutional Neural Network

■ CNN의 특징

- Hierarchical Pattern Recognition



- Translation Invariance



Unit 03 | Convolutional Neural Network

■ Max Pooling vs. Average Pooling

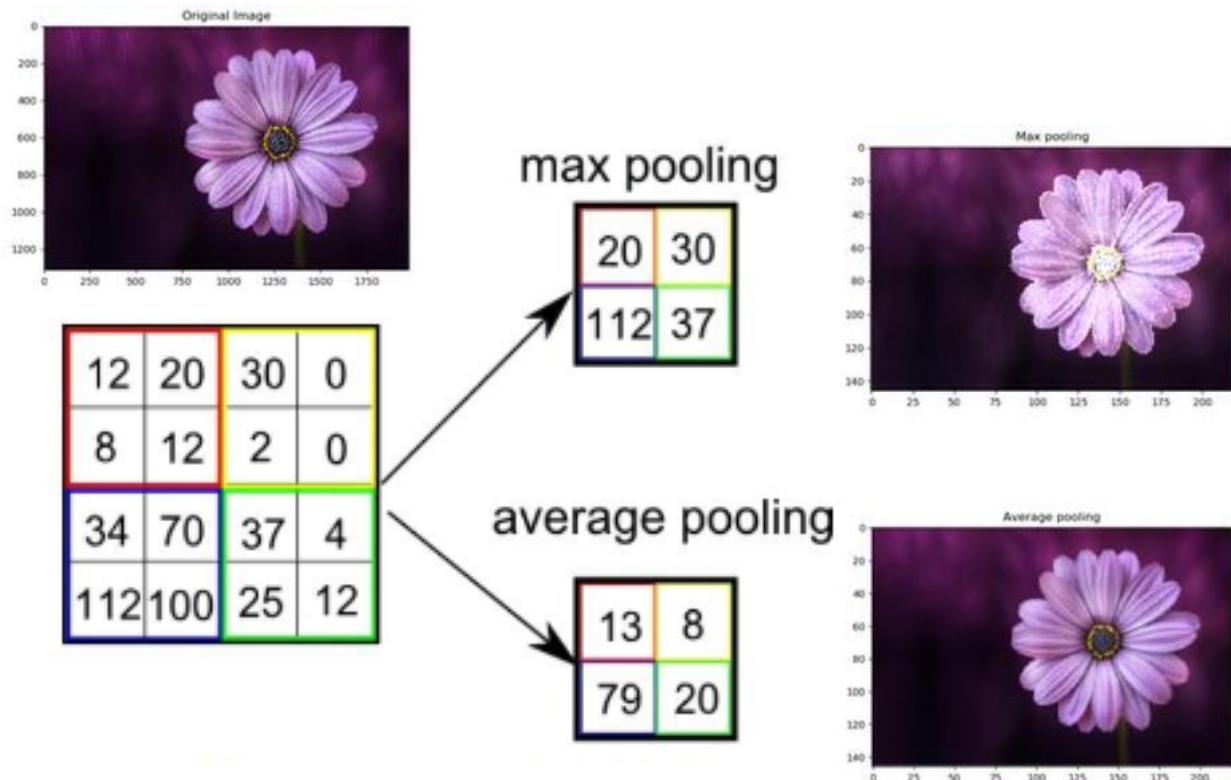
• Max Pooling

- 해당 window의 max 값 추출
- 가장 밝은 픽셀 값이 선택됨

• Average Pooling

- 해당 window의 average 값 추출
- Smoothing 효과

→ Most important feature를 뽑는다는 관점에서 일반적으로 Max Pooling을 사용한다.



Unit 03 | Convolutional Neural Network

■ Pooling을 사용하는 이유

1. Down Sampling

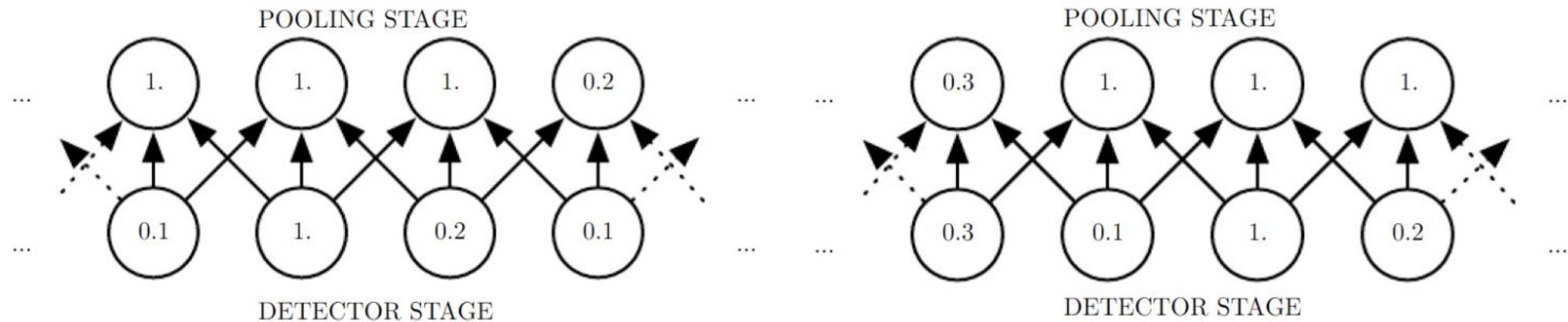
- 사이즈를 줄여 불필요한 연산을 줄이고 parameter 수를 줄여 overfitting을 방지한다.
- 선형결합이 아니기 때문에 weight가 없어 학습이 일어나지 않는다.
- 채널 수는 변함 없다.

Unit 03 | Convolutional Neural Network

■ Pooling을 사용하는 이유

2. Translation Invariance

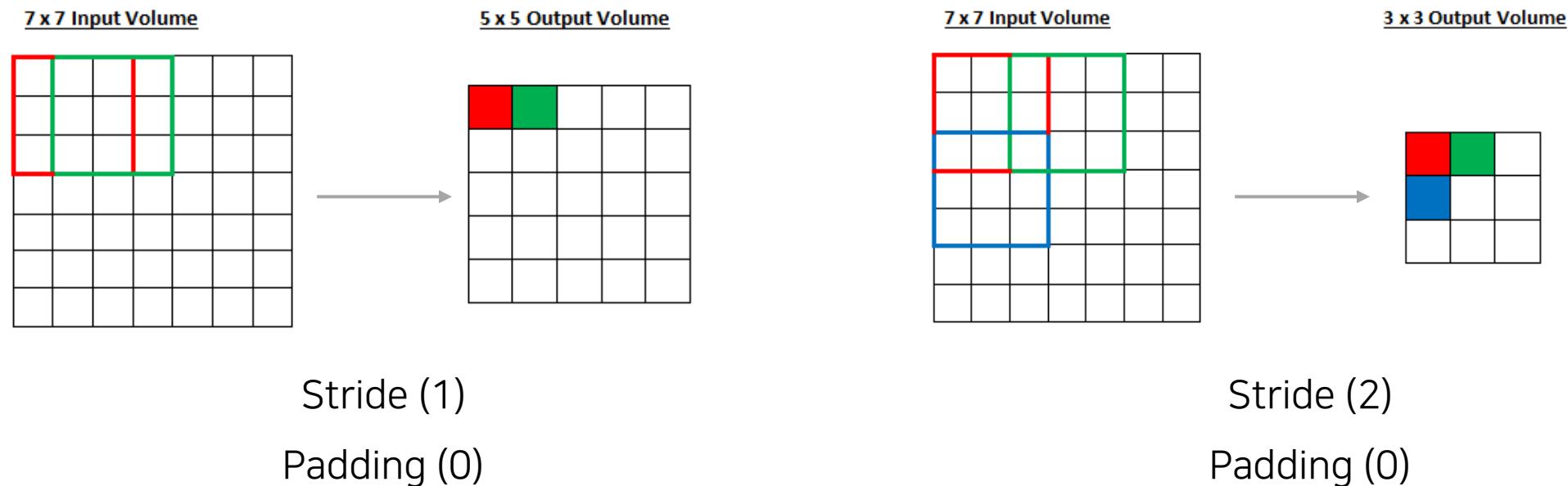
- 작은 이동에 둔감하다. → 이미지 내에서의 위치에 관계없이 동일한 패턴을 동일하게 인식한다.



Unit 03 | Convolutional Neural Network

■ Stride

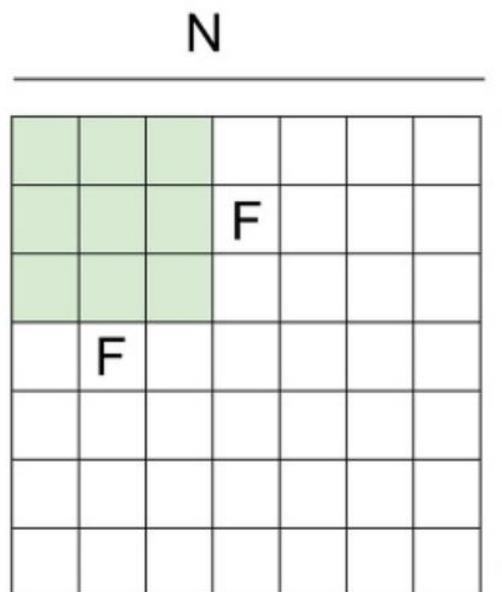
- Filter 적용 시 이동 간격을 의미한다.
- Stride를 키우면 차원을 더 급격히 줄일 수 있다.



Unit 03 | Convolutional Neural Network

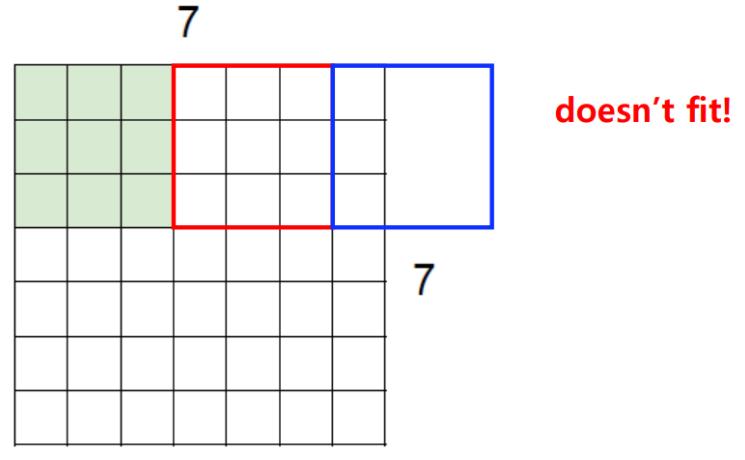
■ Stride

- Filter 적용 시 이동 간격을 의미한다.
- Stride를 키우면 차원을 더 급격히 줄일 수 있다.
- Output의 차원이 정수(integer)가 되도록 Stride를 설정한다.



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 => $(7 - 3)/3 + 1 = 2.33$



Stride = 3 ?

Unit 03 | Convolutional Neural Network

■ Padding

- 데이터 가장자리에 fake pixel을 붙여 연산.
- 일반적으로 zero padding을 사용한다.

→ 크기 손실 방지, 테두리 정보 활용

- Padding 값은 어떻게 결정하나?

$$\text{Padding size} = (F - 1)/2$$

$$\text{Output size} = (N - F + 2P)/S - 1$$

The diagram illustrates a convolution operation with padding. It shows an input image of size 7x7 (labeled "Image") with padding, multiplied by a 3x3 kernel (labeled "Filter / Kernel") to produce a feature map of size 5x5 (labeled "Feature").

The input image (7x7) has values:

0	0	0	0	0	0	0
0	2	4	9	1	4	0
0	2	1	4	4	6	0
0	1	1	2	9	2	0
0	7	3	5	1	3	0
0	2	3	4	8	5	0
0	0	0	0	0	0	0

The kernel (3x3) has values:

1	2	3
-4	7	4
2	-5	1

The resulting feature map (5x5) has values:

21	59	37	-19	2
30	51	66	20	43
-14	31	49	101	-19
59	15	53	-2	21
49	57	64	76	10

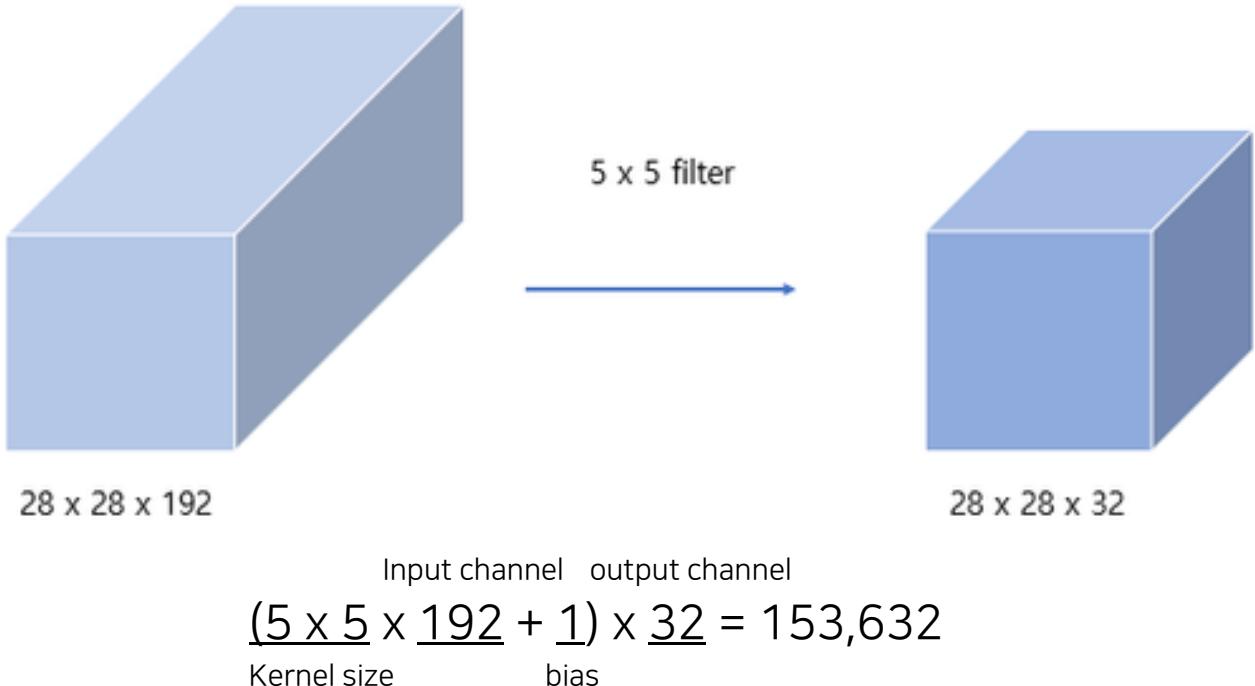
Unit 04 | CNN Visualization

The number of parameters on CNN

Unit 03 | Convolutional Neural Network

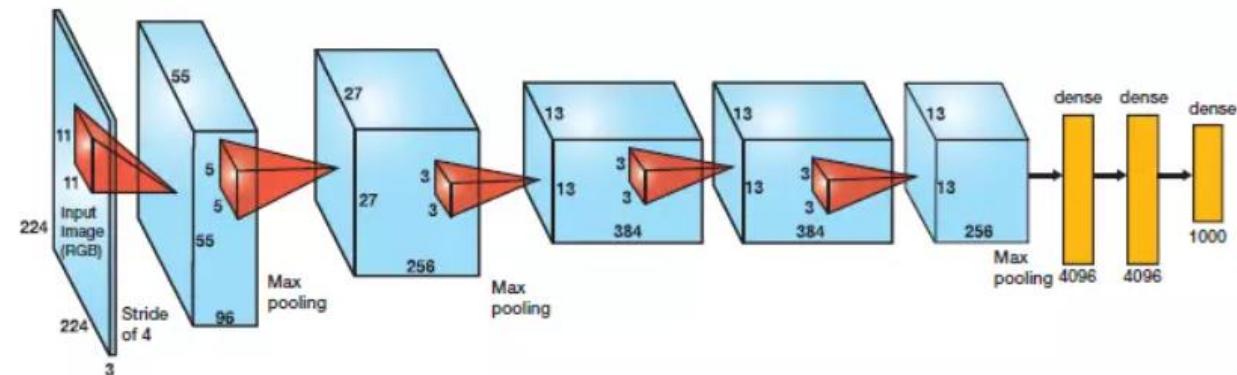
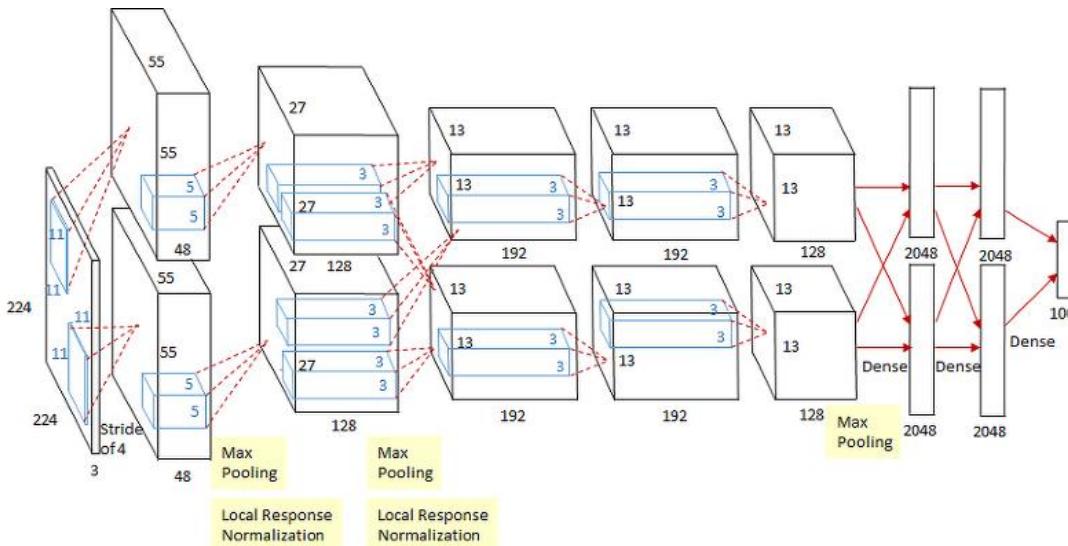
■ CNN 모델의 parameter 개수 계산

Padding (1), Stride (1), 5 x 5 Kernel



Assignments

과제 - AlexNet model



과제 1. AlexNet의 파라미터 개수 구하기

week7_CNNbasic_AlexNet_parameters.ipynb의 물음표를 채워주세요.

과제 2. AlexNet model의 코드 구현하기

week7_CNNbasic_AlexNet_modeling.ipynb에 모델 구현 후 summary로 전체 모델 구조 보이고 주석을 통해 간단한 설명을 해주세요.

References

- 14기 이정은님 강의
http://www.datamarket.kr/xe/board_jPWY12/74345
- 15기 황보진경님 강의
- 이정우 교수님 딥러닝의 기초 강의 - 201029 Chapter7
https://www.youtube.com/playlist?list=PLKs7xpqpX1bd-UDMAe_vl2vZFQ05bzizQ
- Stanford cs231n 강의
<http://cs231n.stanford.edu/syllabus.html>
- 김성훈 교수님 PyTorch Lecture 10: Basic CNN
<https://www.youtube.com/watch?v=LgFNRIFxuUo>
- <http://taewan.kim/post/cnn/>
- <https://yjjo.tistory.com/8>

Q & A

들어주셔서 감사합니다.