

# CNN

진행자 : 현시은

# 컴퓨터가 이미지를 인식하는 방식

● 컴퓨터가 이미지를 인식하는 방식

밀집신경망에서 CNN으로

CNN의 구조

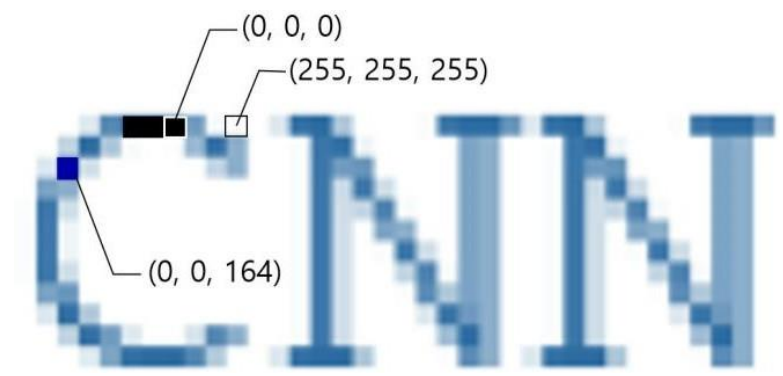
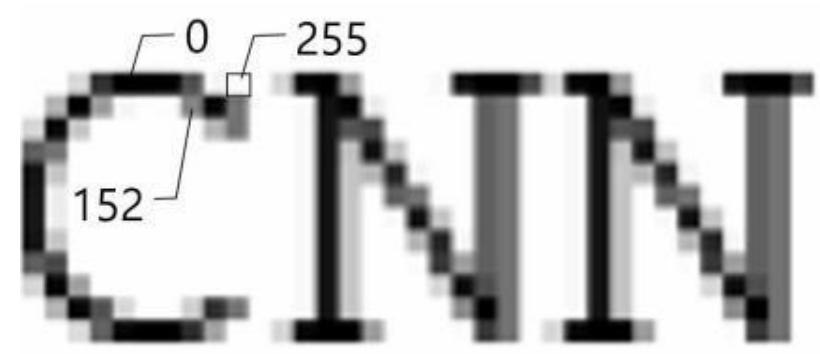
합성곱 계층

풀링 계층

CNN 학습시키기

# 비트맵 이미지

- 비트맵 이미지
  - 각 위치의 픽셀들이 어떤 색상 정보를 담고 있는지 저장
- 흑백 이미지의 경우
  - 검은색 0, 흰색 255를 기준으로 정도를 표현
- 흑백이 아닌 경우
  - 빛의 3원색 (R, G, B) 이용
  - 각각의 색이 합성되는 정도를 0에서 255까지로 표현함



● 컴퓨터가 이미지를 인식하는 방식

밀집신경망에서 CNN으로

CNN의 구조

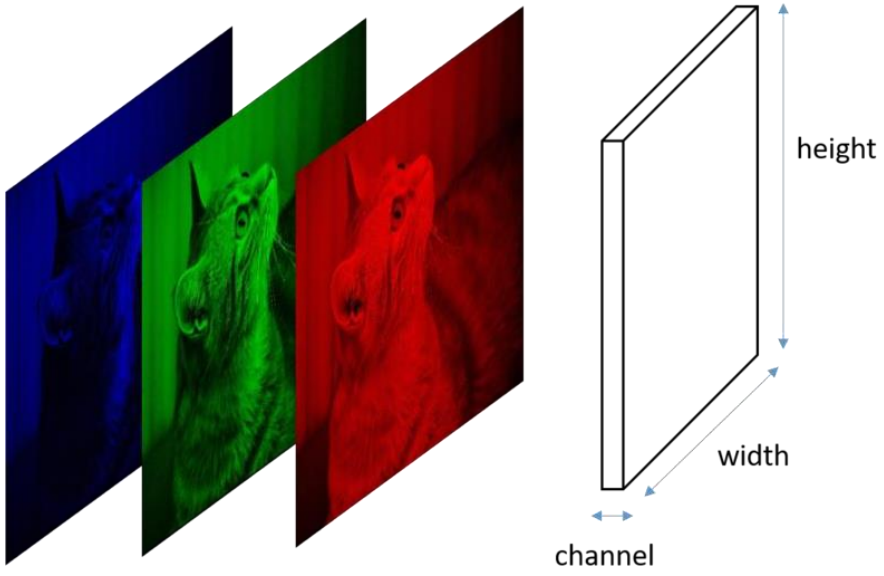
합성곱 계층

풀링 계층

CNN 학습시키기

채널

- 입력 데이터의 속성의 수
- 이미지 데이터에서 색상과 같은 의미 (R/G/B 채널)
- 이미지를 3차원 배열로 생각한다면 width \* height \* (channel 개수)
- Channel 개수 = depth 라고도 부름



AI

# 밀집 신경망에서 CNN으로

컴퓨터가 이미지를  
인식하는 방식

- **밀집신경망에서  
CNN으로**

CNN의 구조

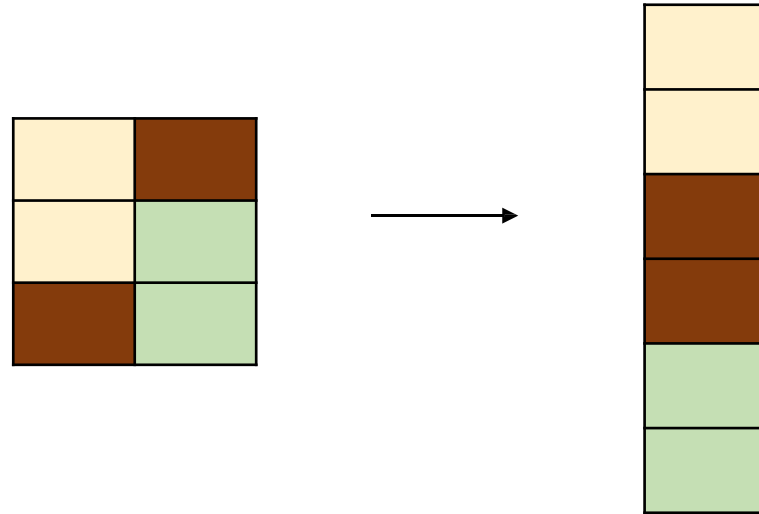
합성곱 계층

풀링 계층

CNN 학습시키기

## 밀집신경망에서 CNN으로!

- 이미지 처리 분야에서 밀집 신경망이 사용되기 힘든 이유
  - 밀집신경망: 입력데이터가 1차원  $((N,1)$  벡터)임
  - 1차원으로 데이터를 바꿔서 입력해야 하기 때문에 이미지의 공간적 정보를 나타내기 힘들다!



- CNN은 공간적 정보를 유지하며 학습 -> 이미지 데이터 처리 가능

# CNN의 구조

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

## • CNN의 구조

합성곱 계층

풀링 계층

CNN 학습시키기

## CNN의 대표적인 구조 예시

- 이번 실습 9에서 다룰 CNN 구조!



풀링 계층, 합성곱 계층..?



## 합성곱 계층

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

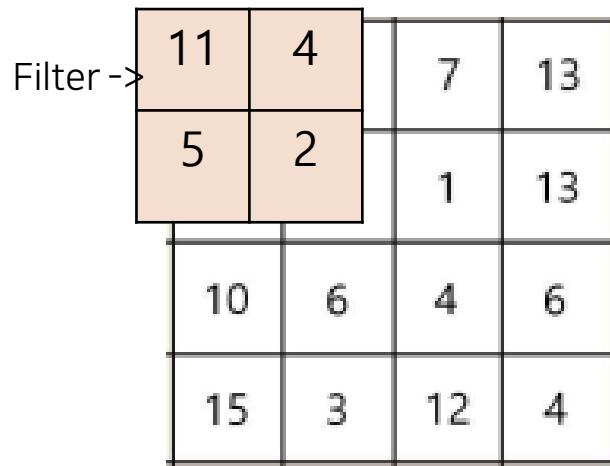
## ● 합성곱 계층

풀링 계층

CNN 학습시키기

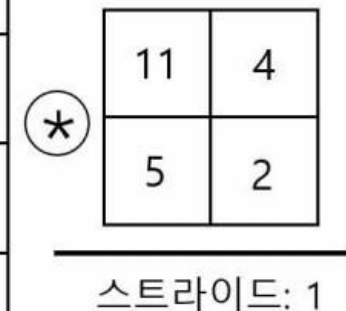
## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지



(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

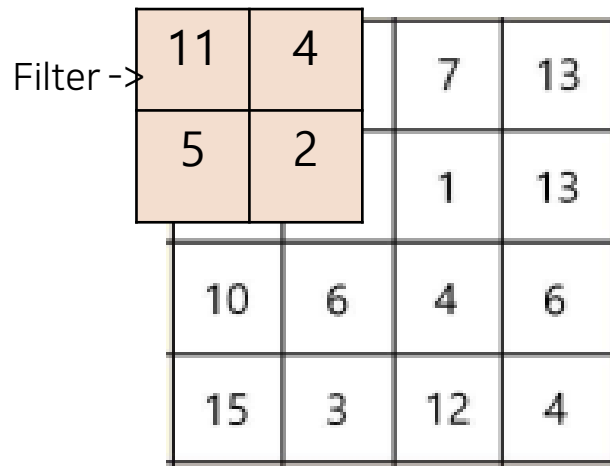
## ● 합성곱 계층

풀링 계층

CNN 학습시키기

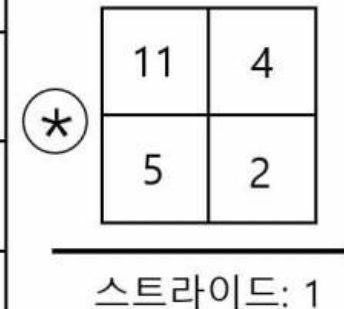
## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지



(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

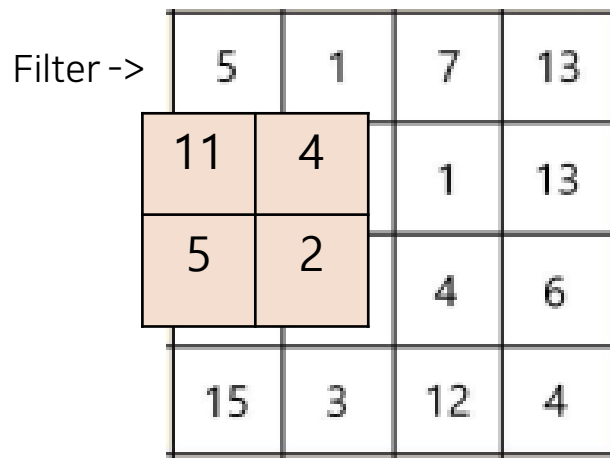
## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지

11	4
5	2

⊗

스트라이드: 1

(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!

Filter ->

5	1	7	13
15	11	4	13
10	5	2	6
15	3	12	4

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지

11	4
5	2

⊗

스트라이드: 1

(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!

Filter ->

5	1	7	13
15	5	11	4
10	6	5	2
15	3	12	4

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지

11	4
5	2

⊗

스트라이드: 1

(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!

Filter ->

5	1	7	13
15	5	1	13
10	11	4	6
15	5	2	4

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지

11	4
5	2

⊗

스트라이드: 1

(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 필터(Filter)

- 필터(Filter) = Kernel과 같은 뜻을 가진 용어
- 합성곱 연산을 수행시킬 필터
- Filter의 크기: Hyperparameter <- 직접 결정!

Filter ->

5	1	7	13
15	5	1	13
10	6	11	4
15	3	5	2

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

(4, 4)  
입력데이터인 흑백이미지

11	4
5	2

⊗

스트라이드: 1

(2, 2)  
합성곱 연산을 수행시킬 필터

144	66	160
247	97	95
215	121	136

(3, 3)  
출력데이터

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 합성곱 연산

입력 이미지 위를 필터가 훑고 지나가면서 합성곱을 수행!

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

⊛

11	4
5	2

스트라이드: 1

144	66	160
247	97	95
215	121	136

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸




결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144		

결과 : 3 X 3

$$5 \times 11 + 1 \times 4 + 15 \times 5 + 5 \times 2 = 144$$

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144		

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	

결과 : 3 X 3



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

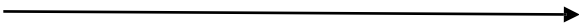
CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144	66	

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144	66	160

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247		

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247		

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144	66	160
247	97	

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	95

결과 : 3 X 3



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	95

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	95
215		

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	95
215		

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

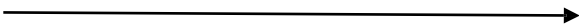
CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144	66	160
247	97	95
215	121	

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

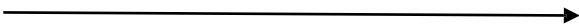
CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸



144	66	160
247	97	95
215	121	

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

이동 간격 1칸

144	66	160
247	97	95
215	121	136

결과 : 3 X 3

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

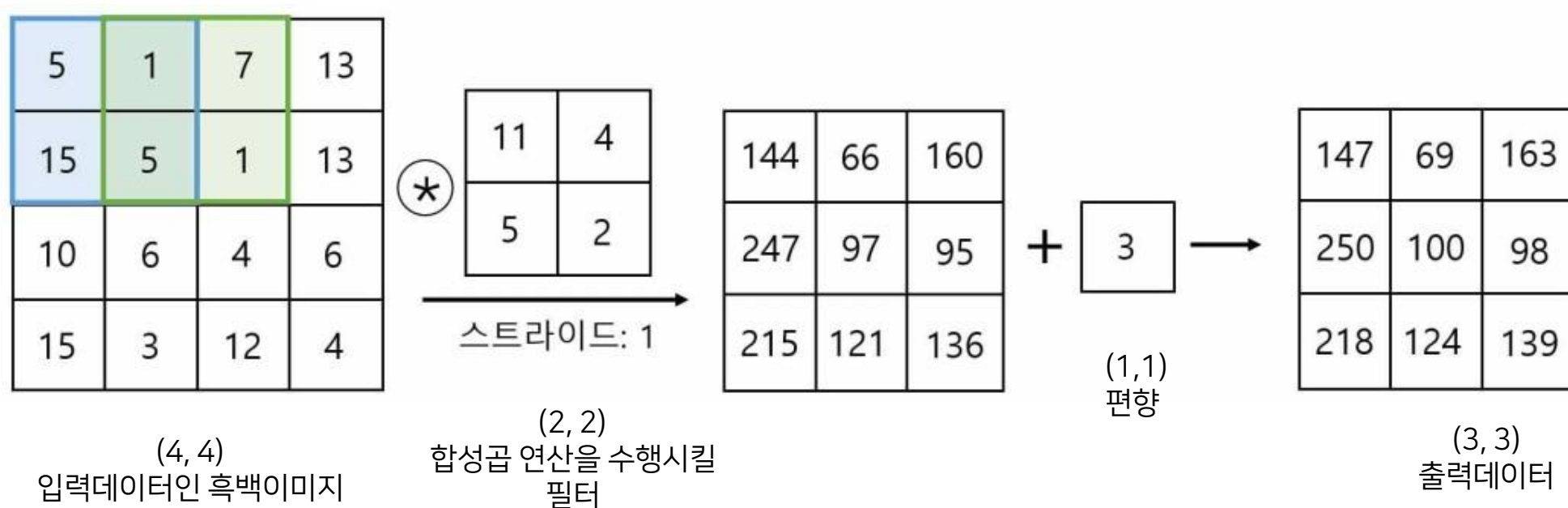
## 합성곱 계층

풀링 계층

CNN 학습시키기

## 합성곱 신경망에서의 편향

- 합성곱 연산에 편향이 포함될 수도 있다!



- 편향은 항상 shape이 (1, 1)인 단일 숫자
- 필터 연산을 거친 출력의 각 성분에 동일하게 더한다!

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

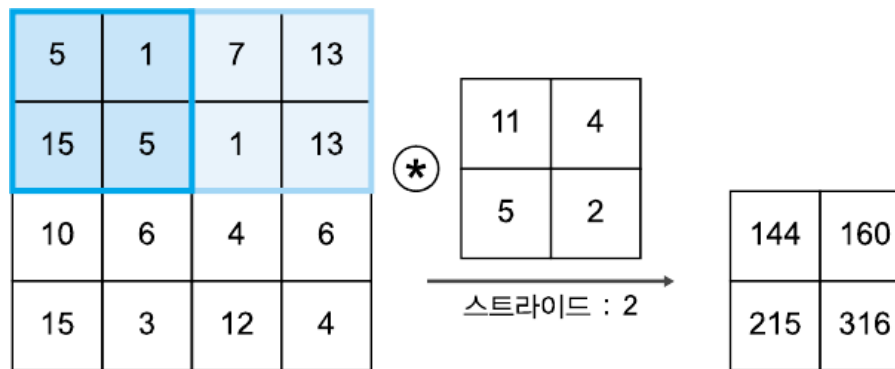
## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 스트라이드(Stride)

- 스트라이드(Stride): 필터 연산을 적용하는 위치의 간격
- 학습을 진행하는 사람이 직접 정하는 하이퍼파라미터



스트라이드가 2인 경우



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

Stride = 2



144	

결과 : 2 X 2

컴퓨터가 이미지를  
인식하는 방식

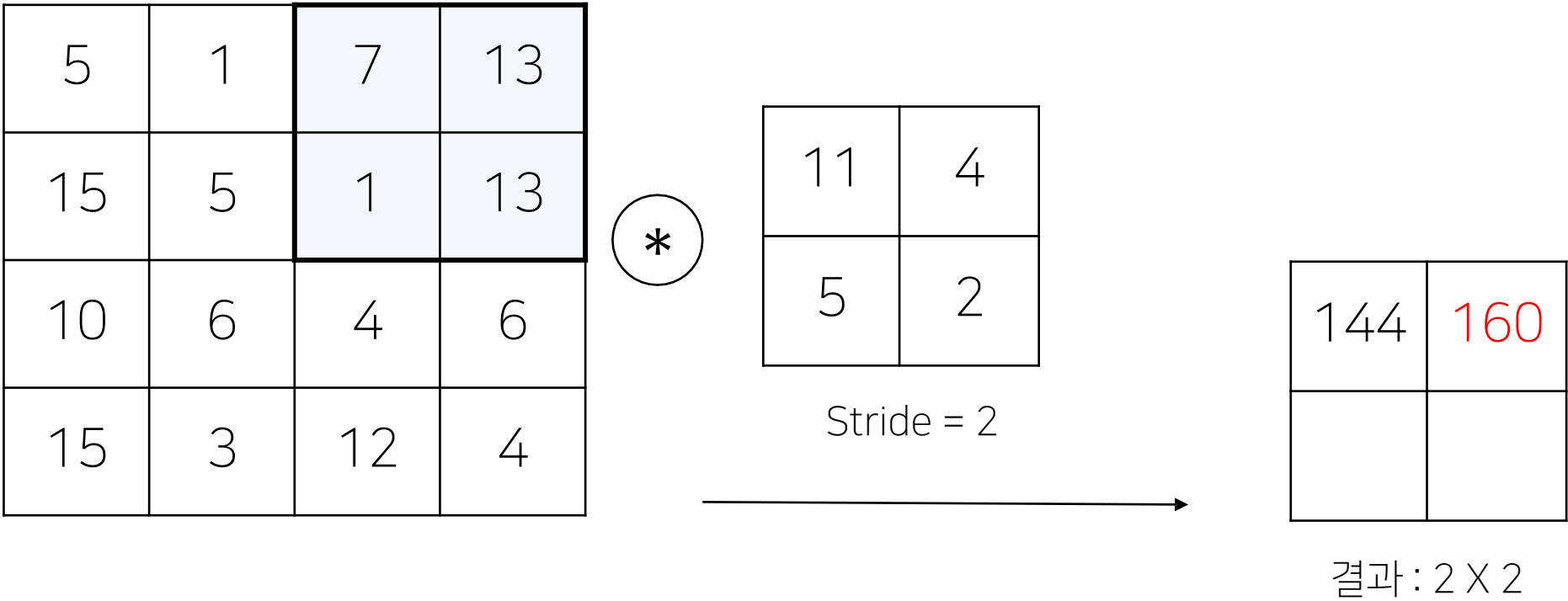
밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

● 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

Stride = 2



144	160
215	

결과 : 2 X 2

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4

\*

11	4
5	2

Stride = 2

144	160
215	316

결과 : 2 X 2

컴퓨터가 이미지를 인식하는 방식

밀집신경망에서 CNN으로

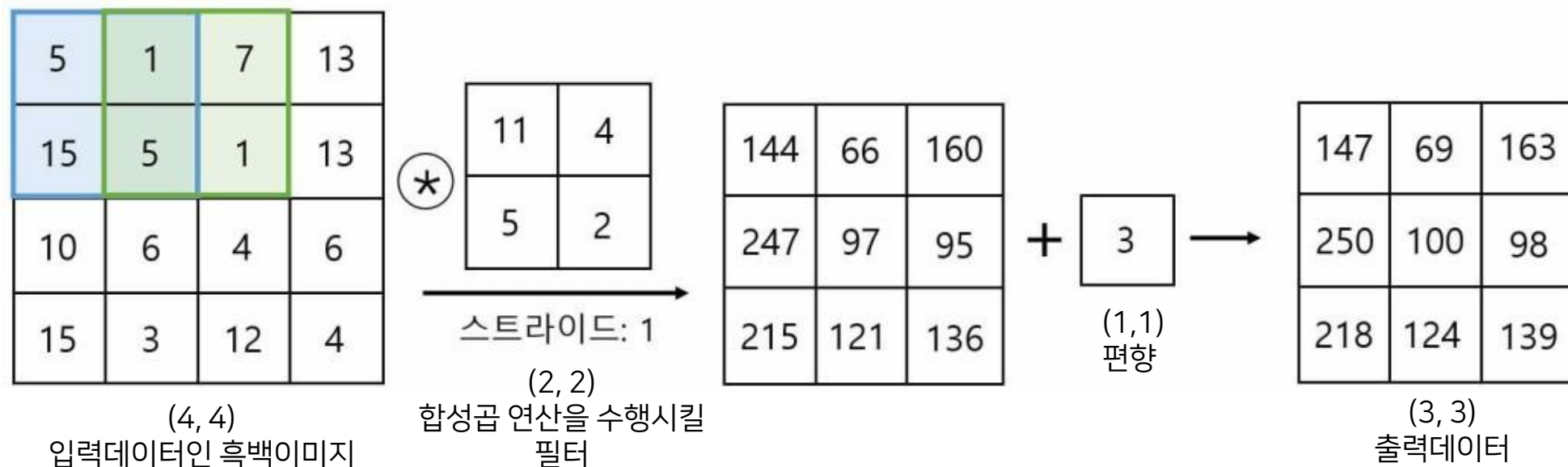
CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 합성곱 연산은 데이터의 차원을 축소시킨다!



- 위의 경우 (4, 4) 크기의 입력 데이터가 (3, 3) 크기의 데이터로 차원이 감소!
- 가장자리에 중요한 데이터가 포함되는 경우 학습 효율에 악영향을 미침

->패딩(Padding) 처리로 이러한 문제 해결!

$$H_{out} = \frac{H_{in} - H_{filter}}{S} + 1$$

$H_{out}$  = 출력 이미지의 높이

$P$  = Padding

$H_{filter}$  = 필터의 높이

$S$  = Stride

컴퓨터가 이미지를 인식하는 방식

밀집신경망에서 CNN으로

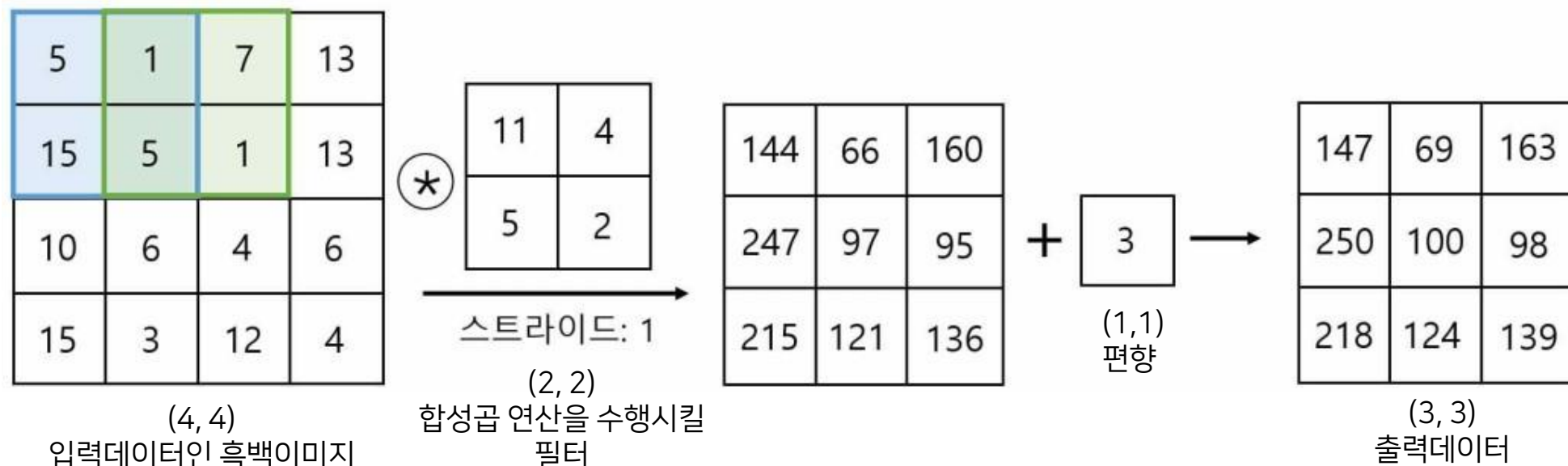
CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 합성곱 연산은 데이터의 차원을 축소시킨다!



- 위의 경우 (4, 4) 크기의 입력 데이터가 (3, 3) 크기의 데이터로 차원이 감소!
- 가장자리에 중요한 데이터가 포함되는 경우 학습 효율에 악영향을 미침

-> **패딩(Padding)** 처리로 이러한 문제 해결!

$$H_{out} = \frac{H_{in} - H_{filter}}{S} + 1$$

$H_{out}$  = 출력 이미지의 높이

$P$  = Padding

$H_{filter}$  = 필터의 높이

$S$  = Stride

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

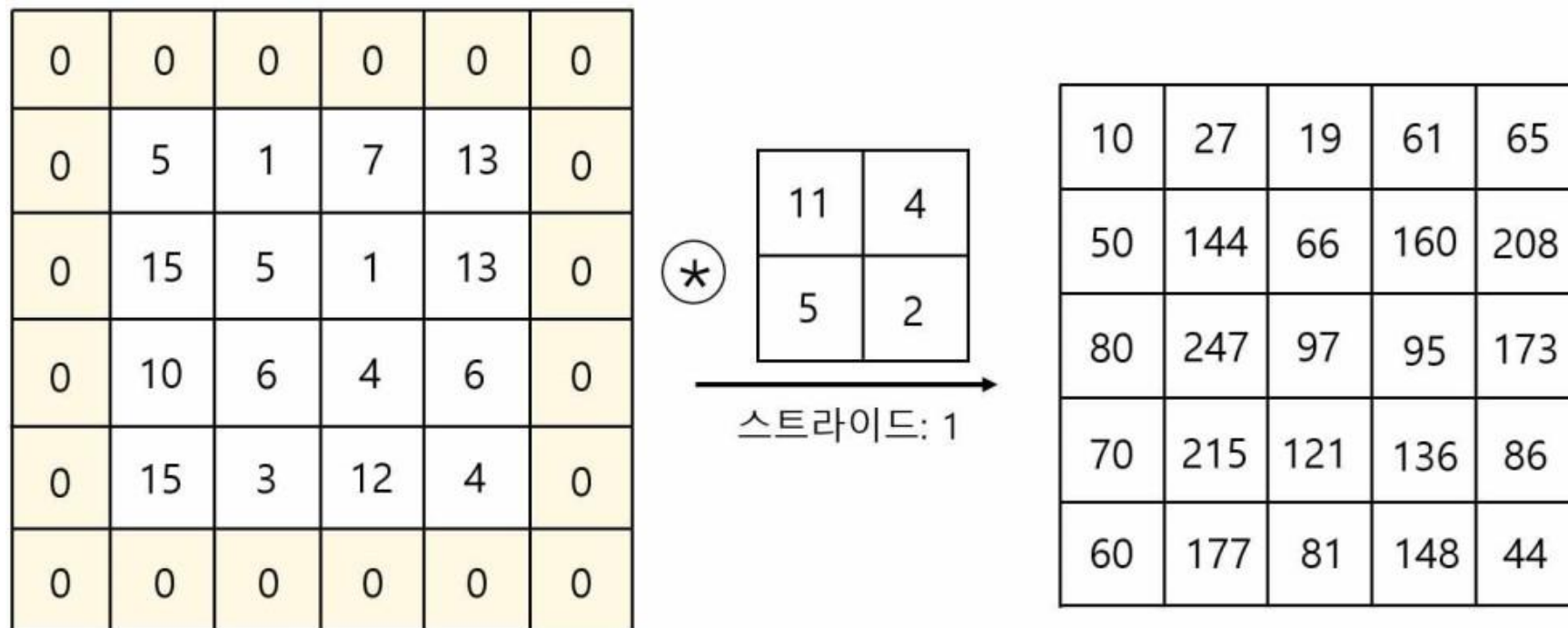
## ● 합성곱 계층

풀링 계층

CNN 학습시키기

## 패딩(Padding)

- 합성곱 연산을 거치기 전, 입력 데이터 주변을 특정 값 (0)으로 적절히 채워 넣는 기법



패딩 처리를 하면 입력 데이터의 가장자리와 내부 픽셀은 동일한 횟수의 필터 연산을 거치게 됨!

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

11	4
5	2

\*



Stride = 1

10				

결과 : 5 X 5



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

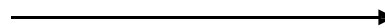
CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

11	4
5	2

\*



Stride = 1

10	27			

결과 : 5 X 5

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

\*

11	4
5	2



Stride = 1

10	27	19		

결과 : 5 X 5

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

11	4
5	2

\*



Stride = 1

10	27	19	61	

결과 : 5 X 5

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

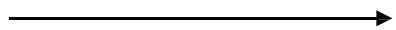
CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

11	4
5	2

\*



Stride = 1

10	27	19	61	65

결과 : 5 X 5

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 패딩(Padding)

0	0	0	0	0	0
0	5	1	7	13	0
0	15	5	1	13	0
0	10	6	4	6	0
0	15	3	12	4	0
0	0	0	0	0	0

11	4
5	2

\*



Stride = 1

10	27	19	61	65
50				

결과 : 5 X 5

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

풀링 계층

CNN 학습시키기

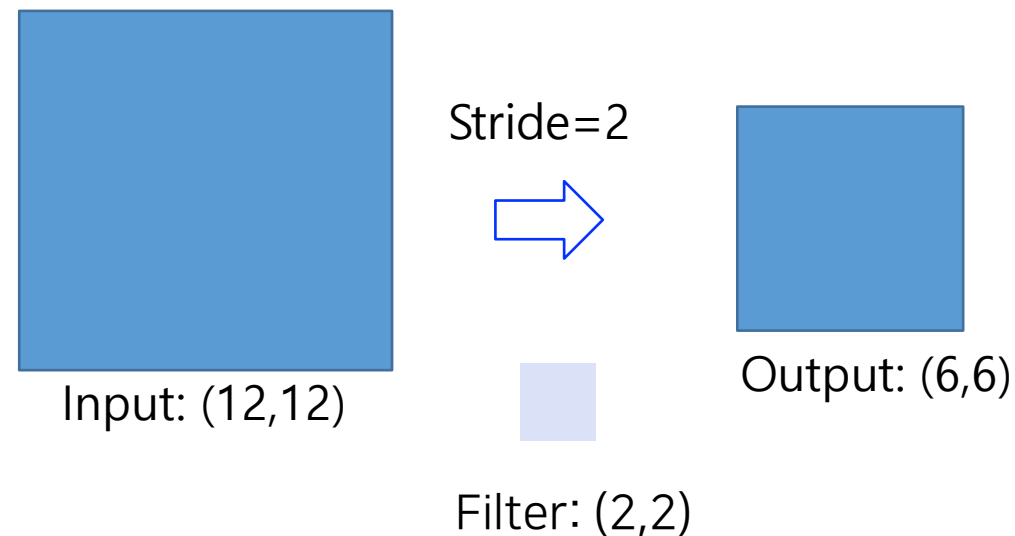
## 패딩(Padding)

패딩(Padding)이 없다면?

- 가장자리 쪽의 픽셀은 이미지 안쪽의 픽셀보다 윈도우가 거쳐 가는 횟수가 적음
- 가장자리 데이터는 합성곱 신경망의 순전파를 진행할 때 약하게 전달
- 가장자리에 중요한 데이터가 포함되는 경우 학습 효율에 악영향을 미침

패딩(Padding)의 이점

- 1) 가장자리 정보 소실 방지
- 2) 출력 크기가 너무 작아지는 문제 방지



Padding이 없는 합성곱 계층을 지나면 크기가 점점 작아진다..!

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

- **합성곱 계층**

풀링 계층

CNN 학습시키기

## CNN에서의 차원 계산

$H_{out}$  = 출력 이미지의 높이

$P = \text{Padding}$

$H_{filter} = \text{filter(kernel)의 높이}$

$S = \text{Stride}$

$$H_{out} = \frac{H_{in} + 2P - H_{filter}}{S} + 1, \quad W_{out} = \frac{W_{in} + 2P - W_{filter}}{S} + 1$$

필터 크기  
스트라이드  
패딩



학습시키는 사람이 정하는 하이퍼 파라미터  
(모두 정수이어야 함!!)

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

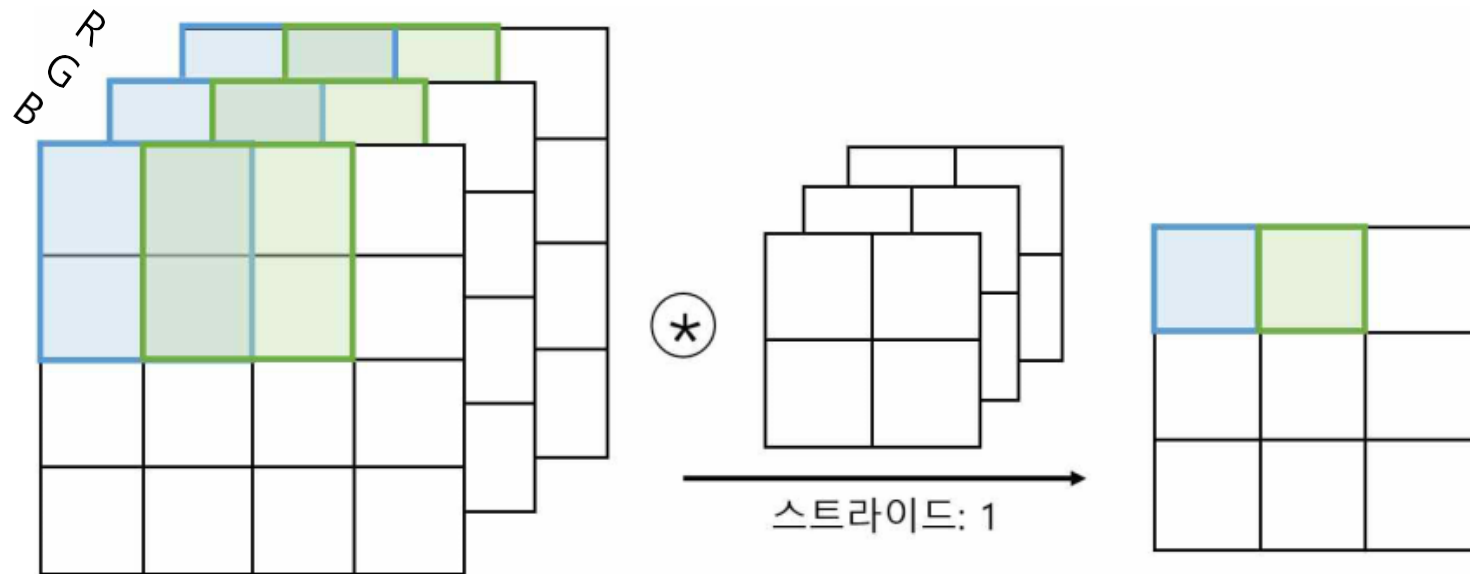
- 합성곱 계층

풀링 계층

CNN 학습시키기

## 2개 이상의 채널로 표현되는 이미지에 대한 합성곱 연산

예) 컬러 이미지: 채널 개수 3개 (R, G, B)



합성곱 연산에서는 Input data의 channel 수와 filter의 채널 수는 같아야 함 (그림에서 3개)



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

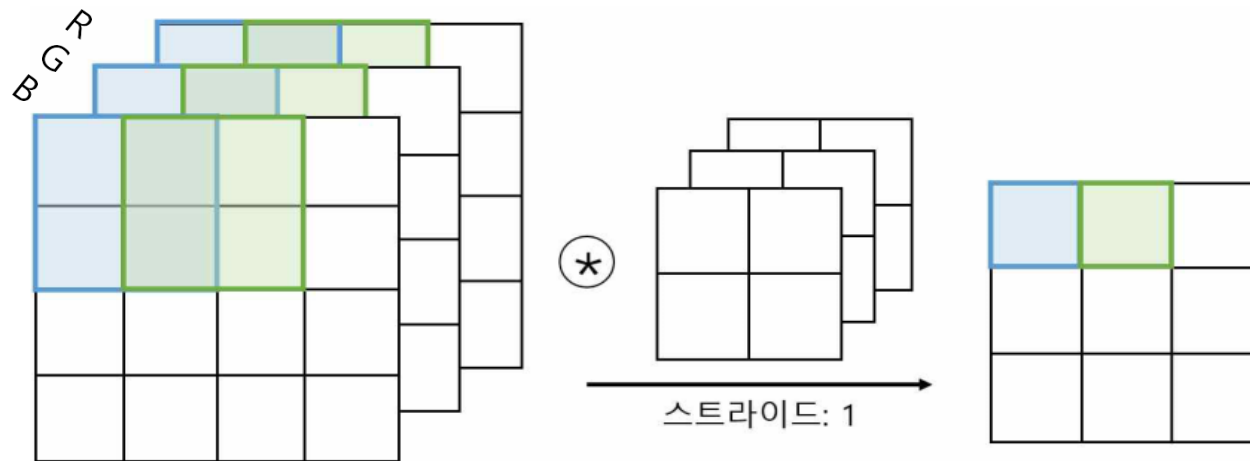
CNN의 구조

- 합성곱 계층

풀링 계층

CNN 학습시키기

## 2개 이상의 채널로 표현되는 이미지에 대한 합성곱 연산



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

## ● 합성곱 계층

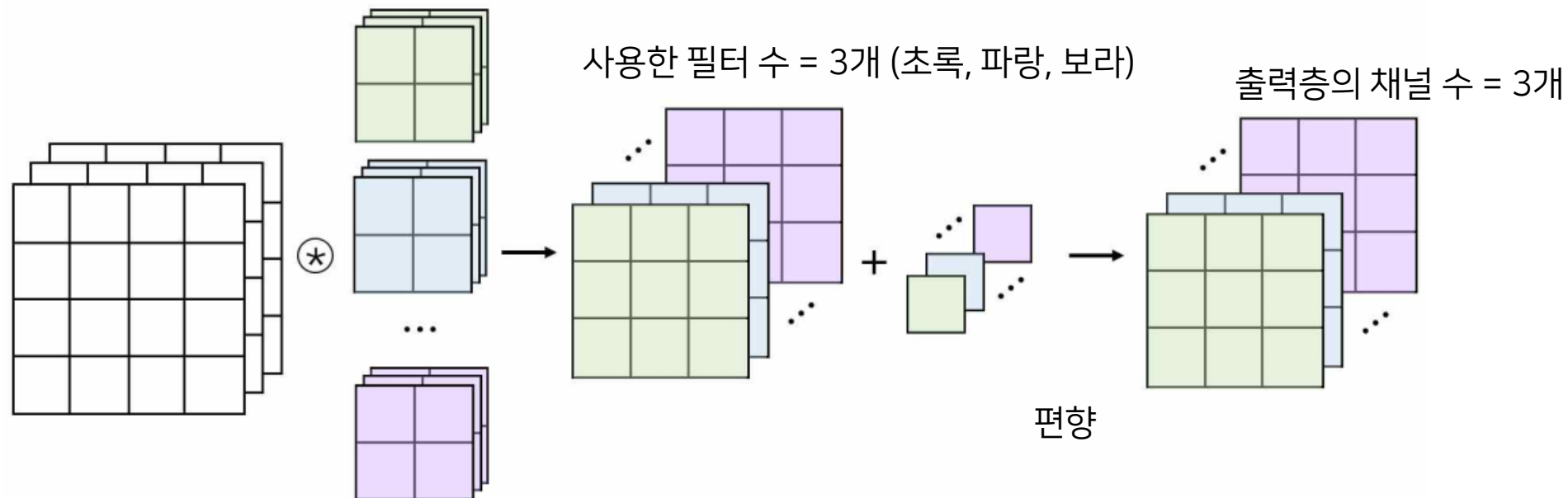
풀링 계층

CNN 학습시키기

## 2개 이상의 채널로 표현되는 이미지에 대한 합성곱 계층

그렇다면, 출력 채널 개수가 여러 개인 경우에는? → 필터 개수를 늘리면 된다!

- 여러 개의 필터를 사용하는 합성곱 계층



입력층에 대해 사용한 필터 수 = 출력층의 채널 수

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

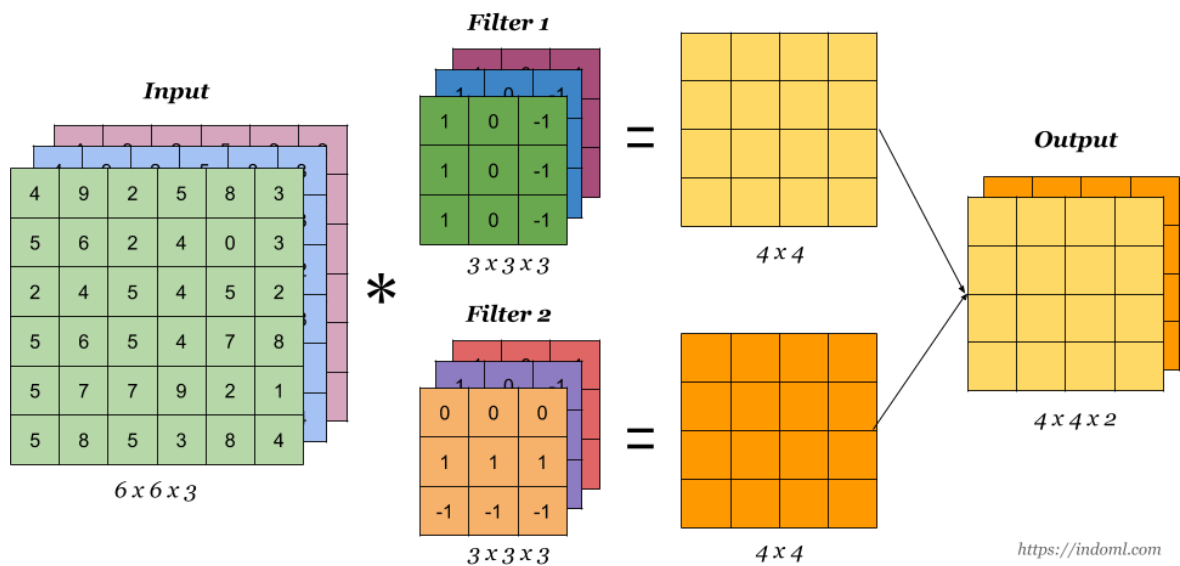
- 합성곱 계층

풀링 계층

CNN 학습시키기

## 정리!

Input data의 channel 수와 관계없이 filter의 개수만큼 output data가 나옴(그림에서 2개)



필터의 값에 따라 신경망의 최종 출력 값이 달라짐

따라서 필터는 학습 과정에서 갱신해야 하는 모델 파라미터로,  
학습을 통해 적절한 필터를 찾아내는 것이 CNN의 학습과정!

# 풀링 계층

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

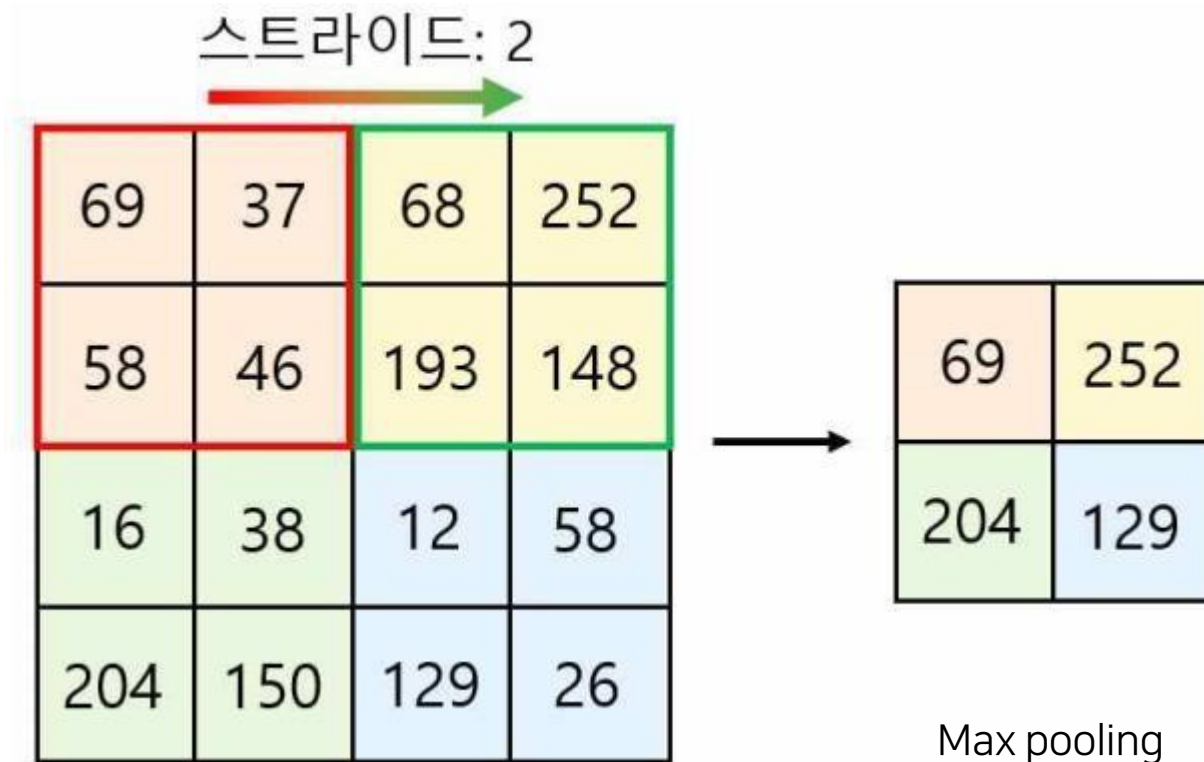
합성곱 계층

## 풀링 계층

CNN 학습시키기

## 풀링 계층

- 데이터의 크기를 줄이기 위해 사용하는 층
- 별도의 학습이 일어나지 않음 -> 풀링 계층에서 학습시킬 파라미터: 0개!
- $n \times m$  풀링 :  $n \times m$  영역을 그 영역을 대표하는 원소 하나로 축소하여 나타내는 연산



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층

풀링의 종류

- 최대 풀링 (Max Pooling) : 풀링 영역의 최댓값을 출력
- 평균 풀링 (Average Pooling) : 풀링 영역의 평균값을 출력

One Feature Map

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

Pooling

5	8
-1	7

(a) Max-Pooling

One Feature Map

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

Pooling

2	3
-4	4

(b) Average-Pooling

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층

Max Pooling  
스트라이드 = 2

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4



15	

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층

Max Pooling  
스트라이드 = 2

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4



15	13



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층

Max Pooling  
스트라이드 = 2

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4



15	13
15	

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층

Max Pooling  
스트라이드 = 2

5	1	7	13
15	5	1	13
10	6	4	6
15	3	12	4



15	13
15	12

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

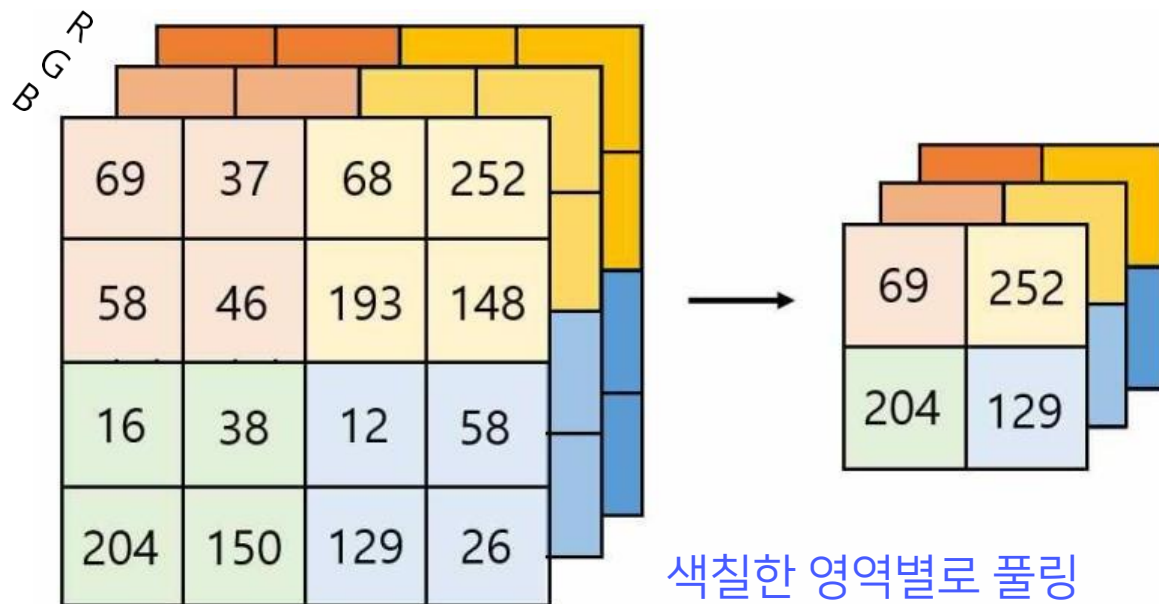
## ● 풀링 계층

CNN 학습시키기

# 풀링 계층

풀링계층을 통과하면 입력데이터의 변화에 민감하지 않게 된다!

- 최대 풀링 (Max Pooling) : 최대값이 갱신되어야만 결과값이 변화하기 때문
- 평균 풀링 (Average Pooling) : 1개의 입력 데이터값의 변화가 평균에 반영되면 작게 나타남



채널이 여러 개인 경우 :  
각 채널에 대해 독립적으로 연산 진행

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

- 풀링 계층

CNN 학습시키기

## 풀링 계층에서 차원 계산

$$W_{\text{out}} = \frac{W_{\text{in}} - W_{\text{filter}}}{S} + 1$$

$$H_{\text{out}} = \frac{H_{\text{in}} - H_{\text{filter}}}{S} + 1$$

# CNN 학습시키기

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

풀링 계층

- CNN 학습시키기

## CNN 학습시키기



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

풀링 계층

- CNN 학습시키기

## CNN 학습시키기



합성곱을 진행하고 풀링하기 이전 활성화함수(ReLU) 통과

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

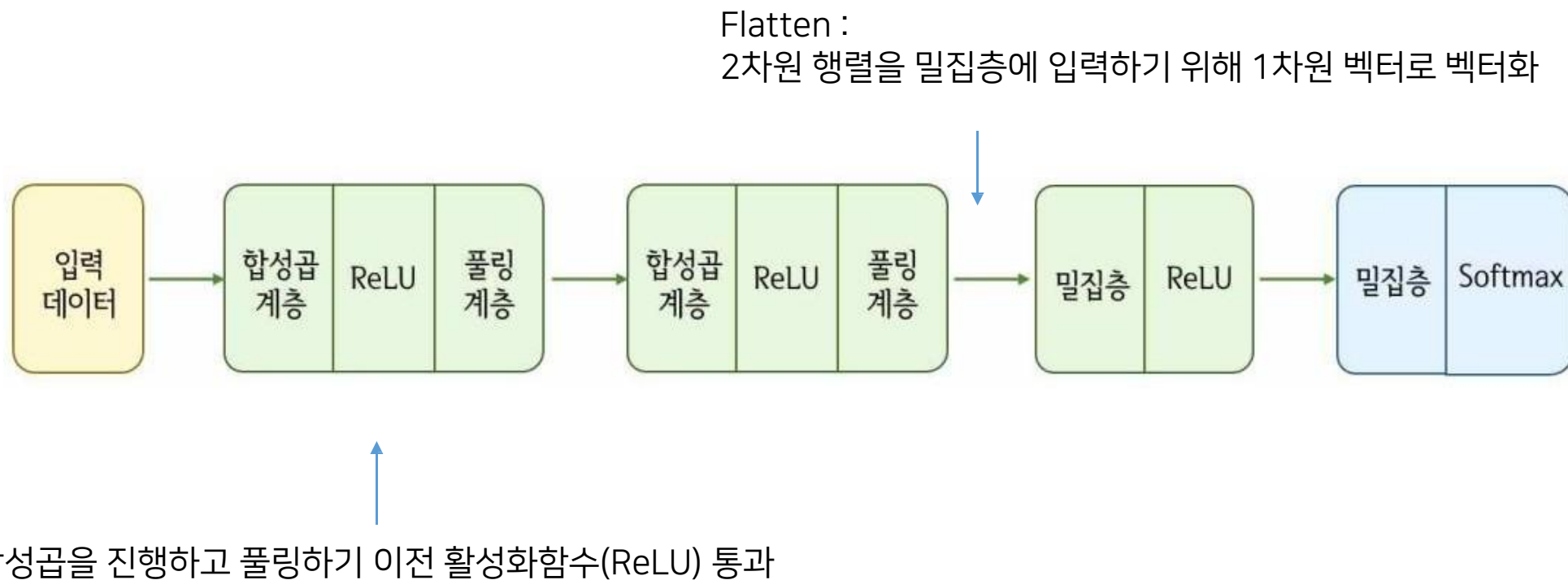
CNN의 구조

합성곱 계층

풀링 계층

## • CNN 학습시키기

# CNN 학습시키기





컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

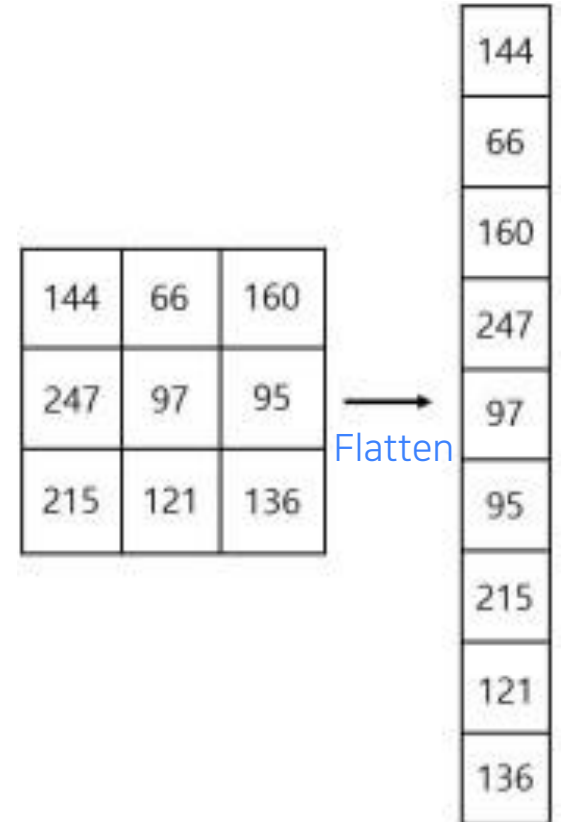
합성곱 계층

풀링 계층

## • CNN 학습시키기

# CNN 학습시키기

- 입력데이터가 합성곱 계층을 지나고 활성화 함수를 통과한 후  
(주로 ReLU), 풀링계층을 통과
- 풀링계층까지 통과한 데이터가 최종 목표인 분류를  
수행하기 위해서는 밀집층에 입력되어야 함
- 풀링계층을 통과한 데이터는 2차원, 밀집층에 입력될 데이터  
는 1차원이므로 벡터화하는 과정(=Flatten) 필요



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

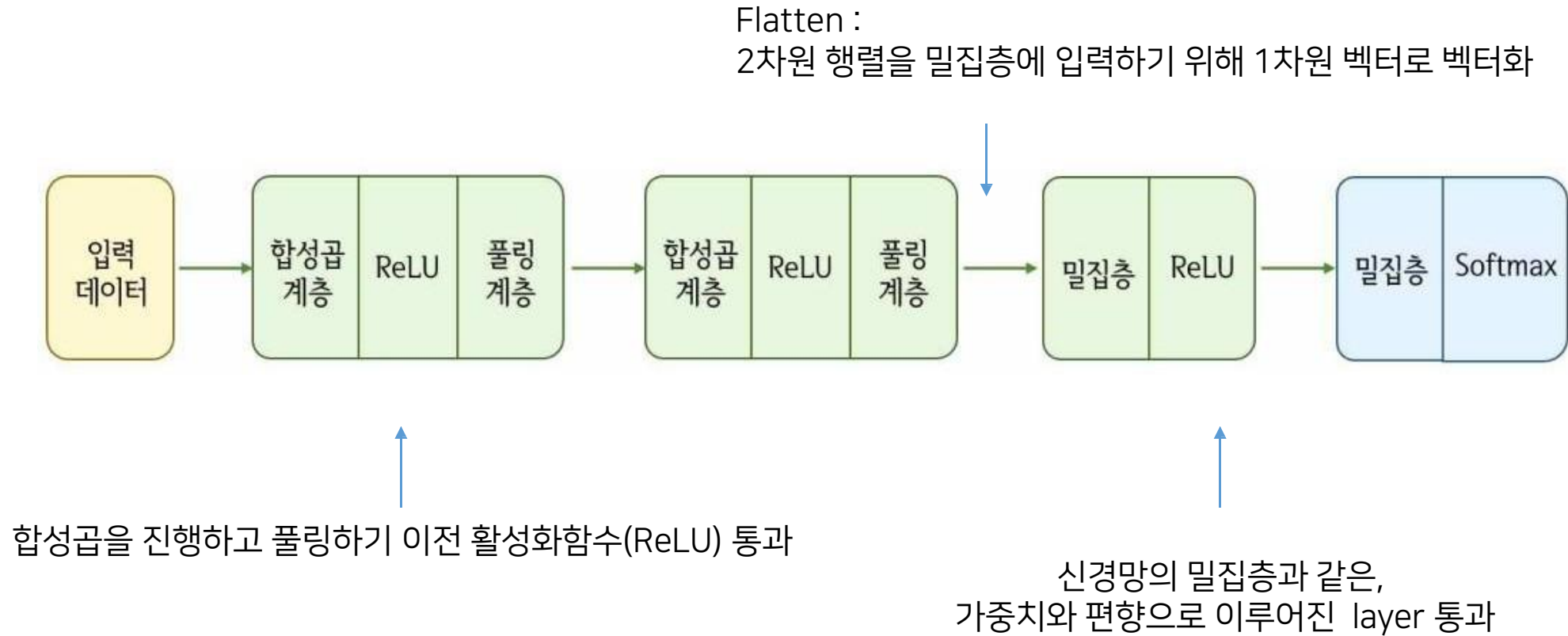
CNN의 구조

합성곱 계층

풀링 계층

## • CNN 학습시키기

# CNN 학습시키기



컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

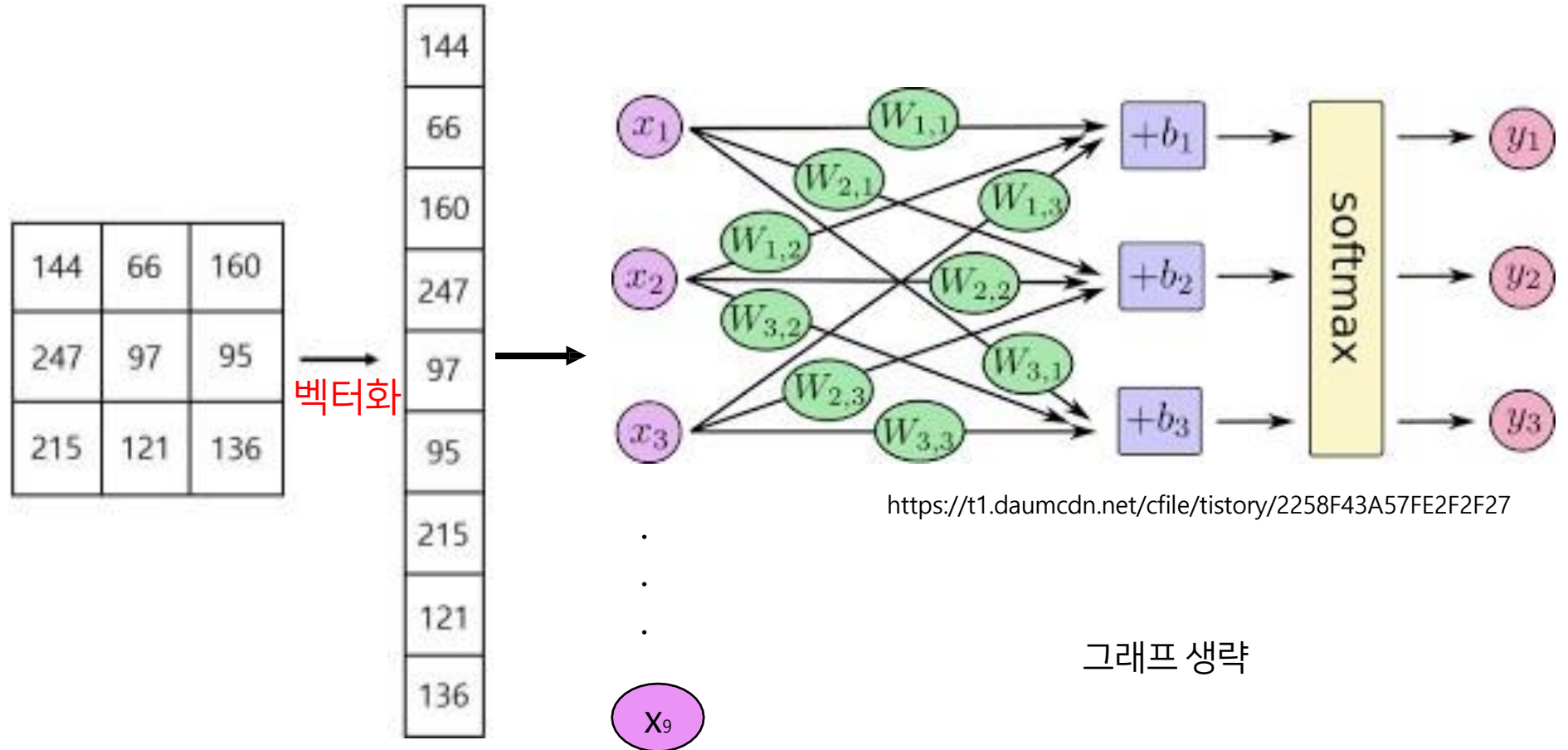
CNN의 구조

합성곱 계층

풀링 계층

- CNN 학습시키기

## CNN 학습시키기



- 벡터화 과정을 거친 데이터를 밀집층에 입력하여 분류 수행!

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

풀링 계층

## • CNN 학습시키기

# CNN 학습시키기

- 손실함수를 통해 모델을 평가하면서, 모델 파라미터인 필터와 밀집층의 가중치( $W$ )와 편향( $b$ )을 갱신

Flatten :

2차원 행렬을 밀집층에 입력하기 위해 1차원 벡터로 벡터화



합성곱을 진행하고 풀링하기 이전 활성화함수(ReLU) 통과

신경망의 밀집층과 같은,  
가중치와 편향으로 이루어진 layer 통과

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

합성곱 계층

풀링 계층

- CNN 학습시키기

## CNN 학습시키기

- CNN을 학습시키면 얻는 것 :

합성곱 계층에서 사용되는 필터 + 밀집층의 가중치와 편향 -> 모델 파라미터!

- 이미지 데이터를 처리하는 경우

- 합성곱 계층의 필터 = 이미지의 특징을 추출해주는 필터
- 합성곱 계층의 깊이가 깊어질수록 복잡하고 추상화된 정보 추출된다!

컴퓨터가 이미지를  
인식하는 방식

밀집신경망에서  
CNN으로

CNN의 구조

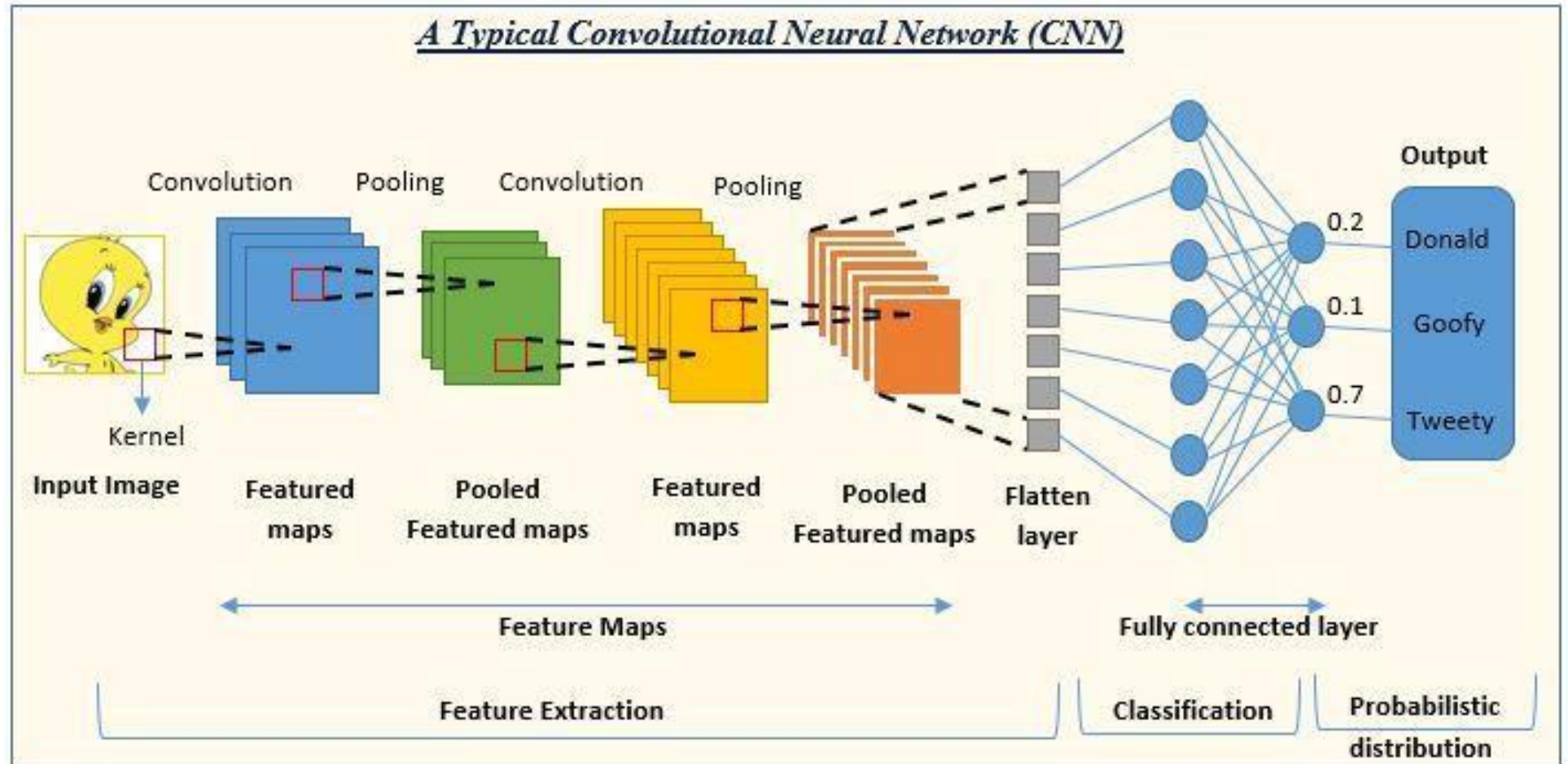
합성곱 계층

풀링 계층

## • CNN 학습시키기

# CNN 학습시키기

- CNN 전체 과정

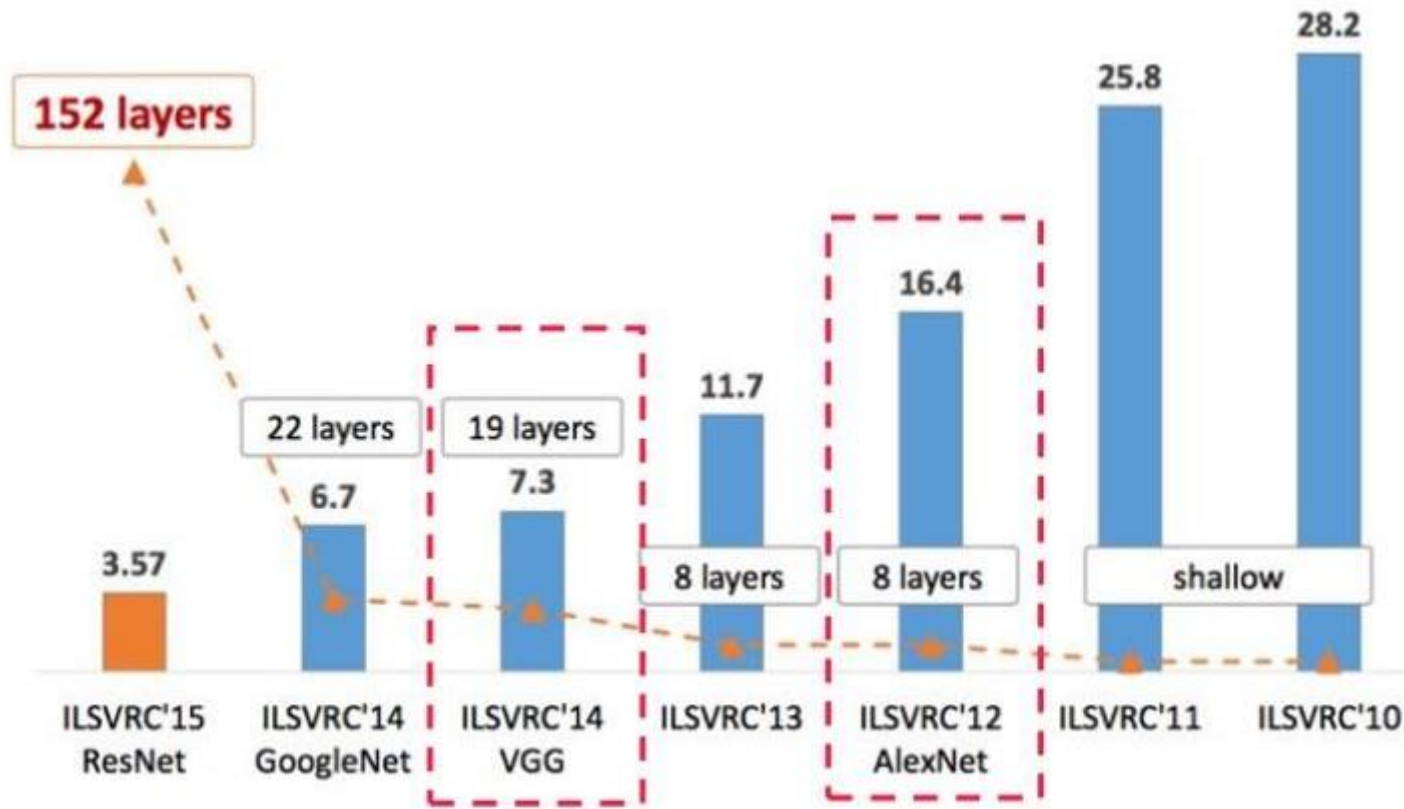


<https://www.analyticsvidhya.com/blog/2022/01/convolutional-neural-network-an-overview/>

# AlexNet

# AlexNet

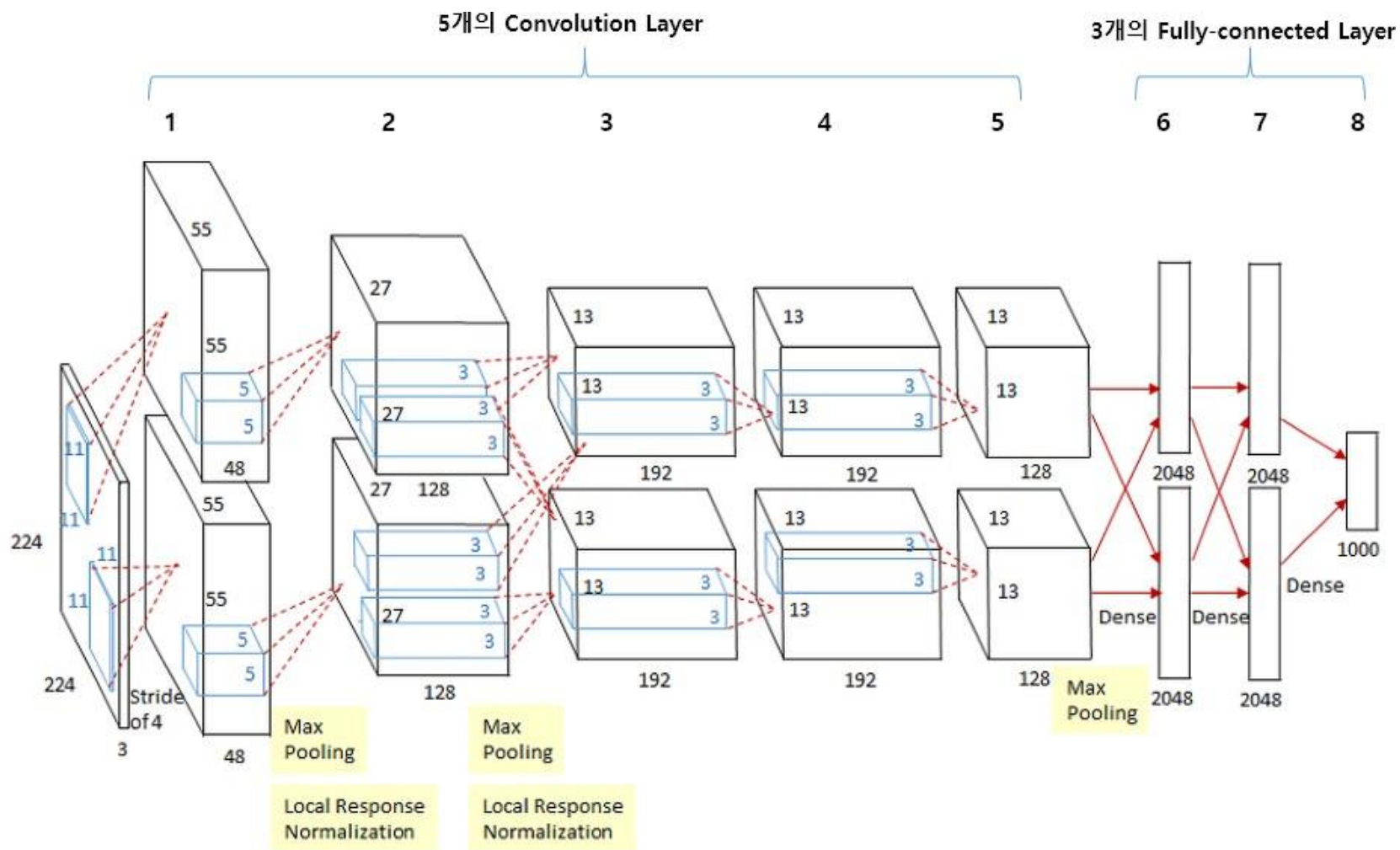
- 2012년에 개최된 ILSVRC 대회에서 큰 차이로 우승
- CNN의 부흥을 일으키는데 큰 역할



<https://velog.io/@lighthouse97>



# AlexNet의 구조



<https://velog.io/@lighthouse97>

# AlexNet의 구조

Layer	Input	Filters	Output
CONV1	[224×224×3]	96 11×11 filters at stride 4,	[55×55×96]
POOL1	[55×55×96]	3×3 filters at stride 2	[27×27×96]
NORM1	[27×27×96]	Normalization layer	[27×27×96]
CONV2	[27×27×96]	256 5×5 filters at stride 1, pad 2	[27×27×256]
POOL2	[27×27×256]	3×3 filters at stride 2	[13×13×256]
NORM2	[13×13×256]	Normalization layer	[13×13×256]
CONV3	[13×13×256]	384 3×3 filters at stride 1, pad 1	[13×13×384]
CONV4	[13×13×384]	384 3×3 filters at stride 1, pad 1	[13×13×384]
CONV5	[13×13×384]	256 3×3 filters at stride 1, pad 1	[13×13×256]
POOL3	[13×13×384]	3×3 filters at stride 2	[6×6×256]
FC6	[6×6×256]	Fully-connected	[4096]
FC7	[4096]	Fully-connected	[4096]
FC8	[4096]	Fully-connected	[1000]

## AlexNet의 차원 계산

Layer	Input	Filters	Output
CONV2	[27×27×96]	256 5×5 filters at stride 1, pad 2	

$$H_{out} = \frac{H_{in} + 2P - H_{filter}}{S} + 1, \quad W_{out} = \frac{W_{in} + 2P - W_{filter}}{S} + 1$$

## AlexNet의 차원 계산

Layer	Input	Filters	Output
CONV2	[27×27×96]	256 5×5 filters at stride 1, pad 2	[27×27×256]

$$H_{out} = \frac{H_{in} + 2P - H_{filter}}{S} + 1, \quad W_{out} = \frac{W_{in} + 2P - W_{filter}}{S} + 1$$

$$H_{out} = W_{out} = \frac{27 + 2 \times 2 - 5}{1} + 1 = 27$$

입력층에 대해 사용한 필터 수 = 출력층의 채널 수

## AlexNet의 차원 계산

Layer	Input	Filters	Output
POOL2	[27×27×256]	3×3 filters at stride 2	

$$W_{\text{out}} = \frac{W_{\text{in}} - W_{\text{filter}}}{S} + 1$$

$$H_{\text{out}} = \frac{H_{\text{in}} - H_{\text{filter}}}{S} + 1$$

## AlexNet의 차원 계산

Layer	Input	Filters	Output
POOL2	[27×27×256]	3×3 filters at stride 2	[13×13×256]

$$W_{\text{out}} = \frac{W_{\text{in}} - W_{\text{filter}}}{S} + 1$$

$$H_{\text{out}} = \frac{H_{\text{in}} - H_{\text{filter}}}{S} + 1$$

$$H_{\text{out}} = W_{\text{out}} = \frac{27 - 3}{2} + 1 = 13$$

## AlexNet의 차원 계산

Layer	Input	Filters	Output
NORM2	[13×13×256]	Normalization layer	
CONV3	[13×13×256]	384 3×3 filters at stride 1, pad 1	
CONV4	[13×13×384]	384 3×3 filters at stride 1, pad 1	
CONV5	[13×13×384]	256 3×3 filters at stride 1, pad 1	
POOL3	[13×13×256]	3×3 filters at stride 2	

$$W_{\text{out}} = \frac{W_{\text{in}} - W_{\text{filter}}}{S} + 1 \quad W_{\text{out}} = \frac{W_{\text{in}} + 2P - W_{\text{filter}}}{S} + 1$$
$$H_{\text{out}} = \frac{H_{\text{in}} - H_{\text{filter}}}{S} + 1 \quad H_{\text{out}} = \frac{H_{\text{in}} + 2P - H_{\text{filter}}}{S} + 1$$

## AlexNet의 차원 계산

Layer	Input	Filters	Output
NORM2	[13×13×256]	Normalization layer	[13×13×256]
CONV3	[13×13×256]	384 3×3 filters at stride 1, pad 1	[13×13×384]
CONV4	[13×13×384]	384 3×3 filters at stride 1, pad 1	[13×13×384]
CONV5	[13×13×384]	256 3×3 filters at stride 1, pad 1	[13×13×256]
POOL3	[13×13×256]	3×3 filters at stride 2	[6×6×256]

$$W_{\text{out}} = \frac{W_{\text{in}} - W_{\text{filter}}}{S} + 1 \quad W_{\text{out}} = \frac{W_{\text{in}} + 2P - W_{\text{filter}}}{S} + 1$$
$$H_{\text{out}} = \frac{H_{\text{in}} - H_{\text{filter}}}{S} + 1 \quad H_{\text{out}} = \frac{H_{\text{in}} + 2P - H_{\text{filter}}}{S} + 1$$



## AlexNet의 차원 계산

Layer	Input	Filters	Output
FC6	$[6 \times 6 \times 256]$	Fully-connected	

# AlexNet의 차원 계산

Layer

FC6

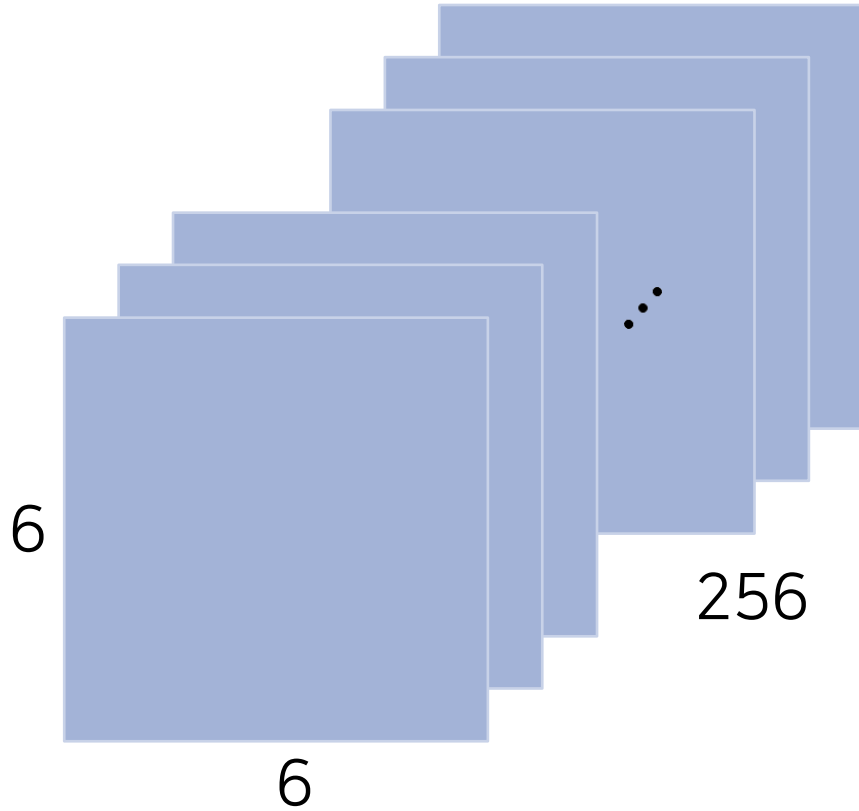
Input

[6×6×256]

Filters

Fully-connected

Output




벡터화



6 x 6 x 256

## AlexNet의 차원 계산

Layer	Input	Filters	Output
FC6	[6×6×256]	Fully-connected	[4096]



9216

1



감사합니다

