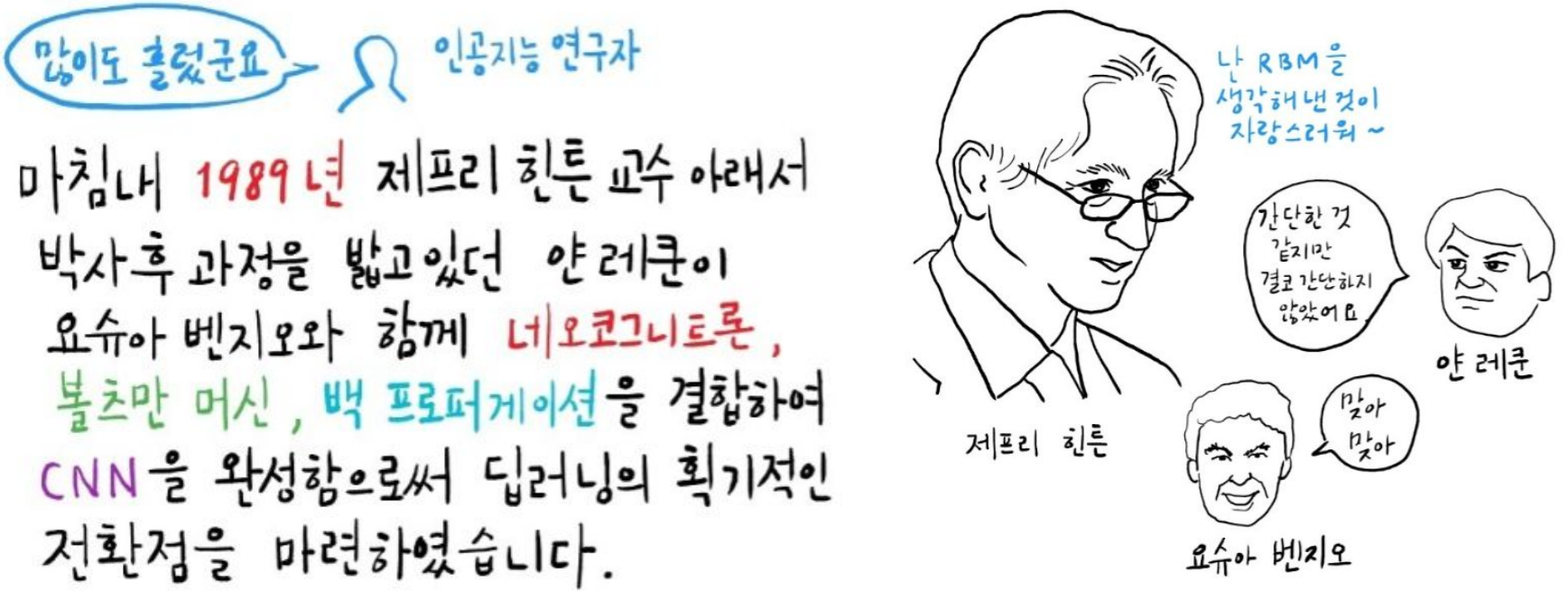
Convolution Neural Net 이론

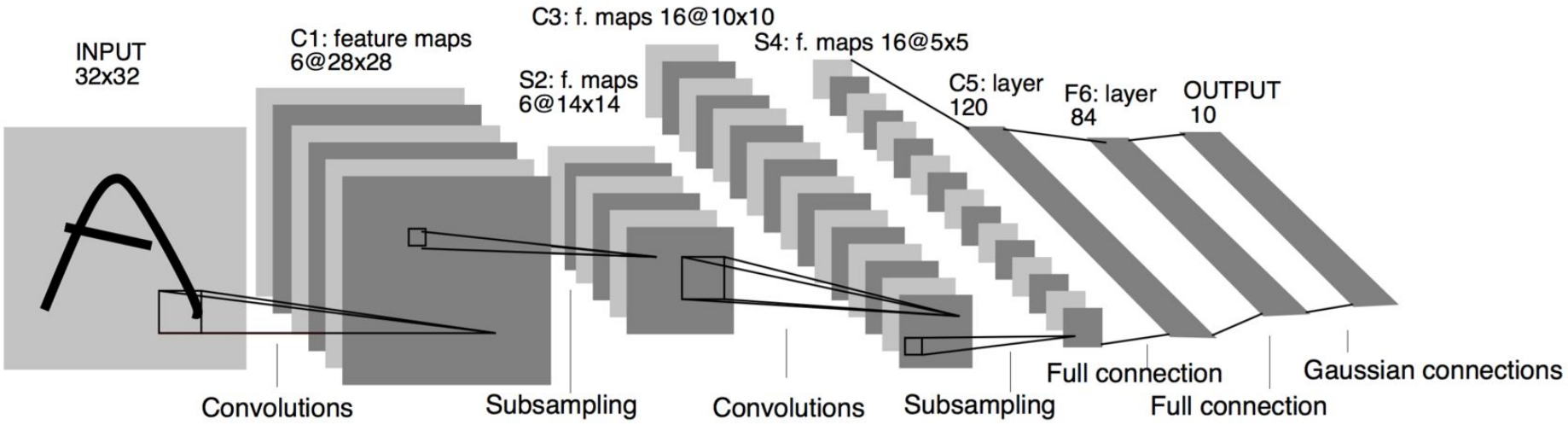
모두의연구소 Rubato Lab.
소준섭

CNN 탄생 배경



https://brunch.co.kr/@hvnpoet/46

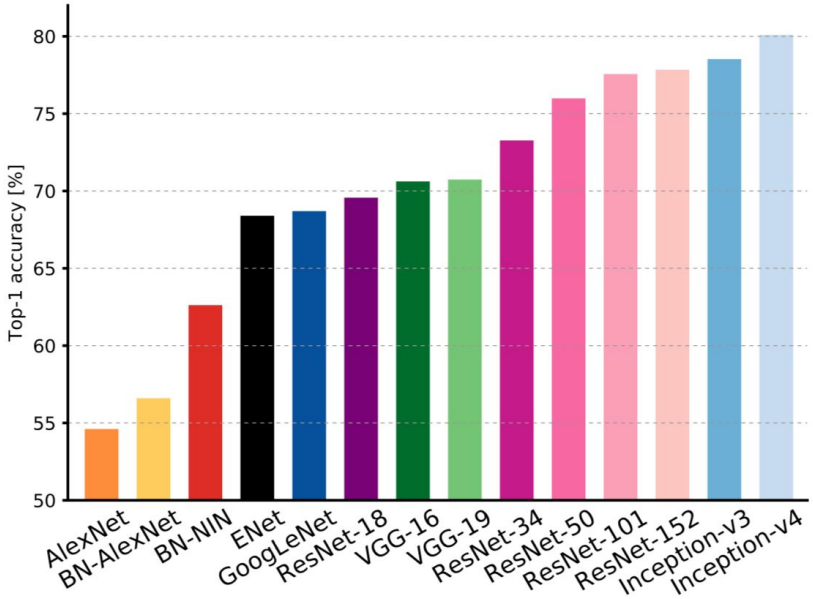
CNN Architecture

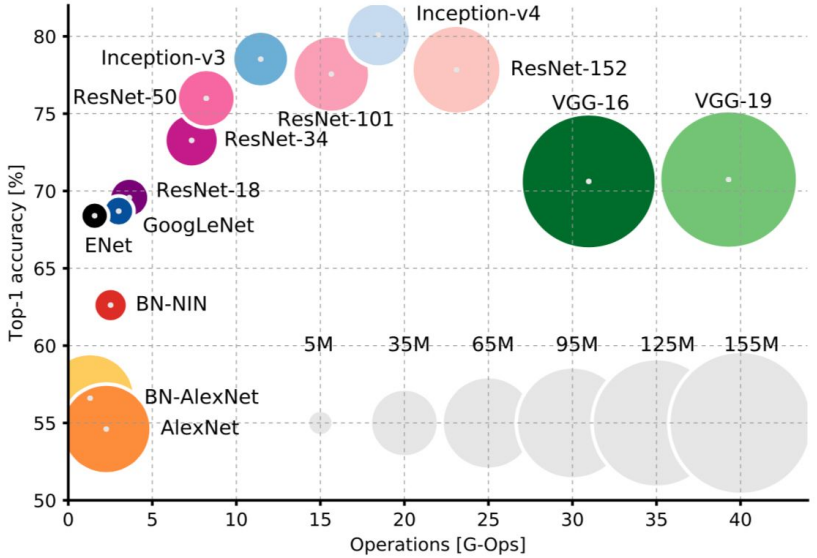


- 이미지와 같은 공간적인 특징을 가지는 고차원 데이터 처리를 위한 Neural Net
- 각 필터가 가중치를 가지며 Filter 단위로 파라미터를 공유하는 특징이 있다.

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

CNN Architecture





- ImageNet challenge를 반복하며 다양한 모델들이 개발되었다.

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution

이미지에 필터를 이동시키며
연산해 결과를 얻는 연산 방법

박스

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

가우시안

.0000	.0000	.0002	.0000	.0000
.0000	.0113	.0837	.0113	.0000
.0002	.0837	.6187	.0837	.0002
.0000	.0113	.0837	.0113	.0000
.0000	.0000	.0002	.0000	.0000

샤프닝

0	-1	0
-1	5	-1
0	-1	0

수평 에지

1	1	1
0	0	0
-1	-1	-1

수직 에지

1	0	-1
1	0	-1
1	0	-1

모션

.0304	.0501	0	0	0
.0501	.1771	.0519	0	0
0	.0519	.1771	.0519	0
0	0	.0519	.1771	.0501
0	0	0	.0501	.0304

(a) 원래 영상과 여러 가지 마스크들

> 박스

> 가우시안

> 샤프닝

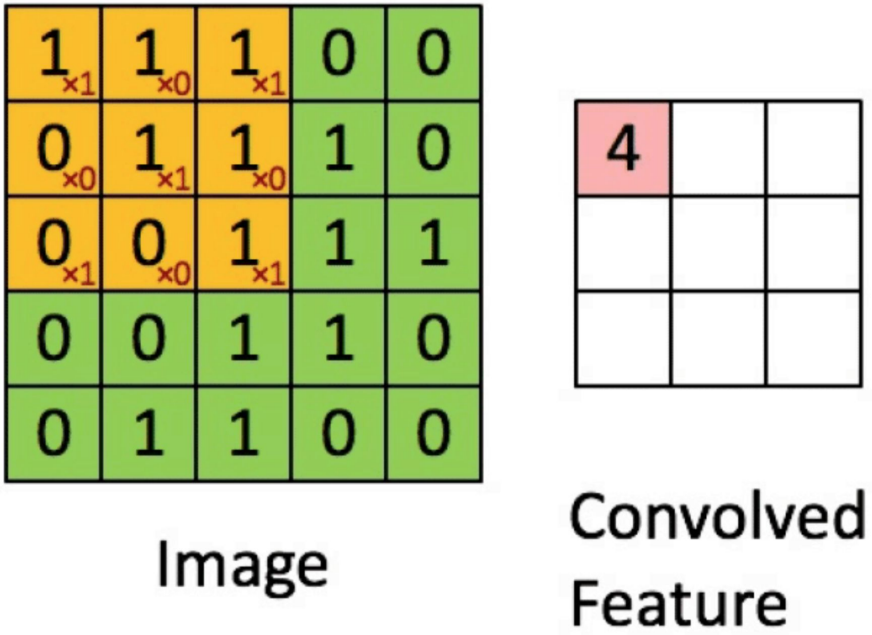
> 수평 에지

> 수직 에지

> 모션

Convolution 연산

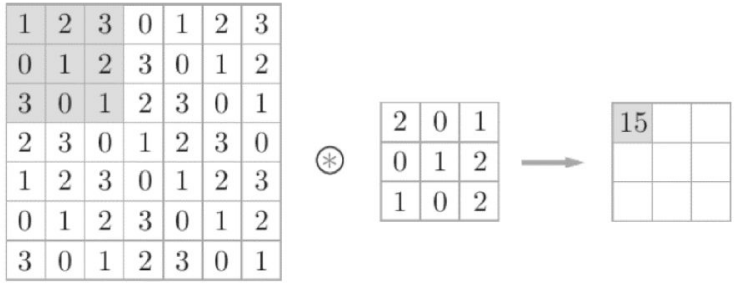
각 필더가 가중치를 가지며, 입력된 데이터와 연산된다.



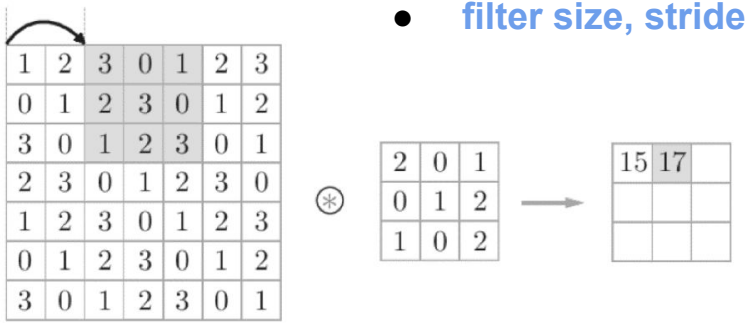
Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution with stride

stride의 크기에 따라 필터가 이동하는 크기가 정해진다.



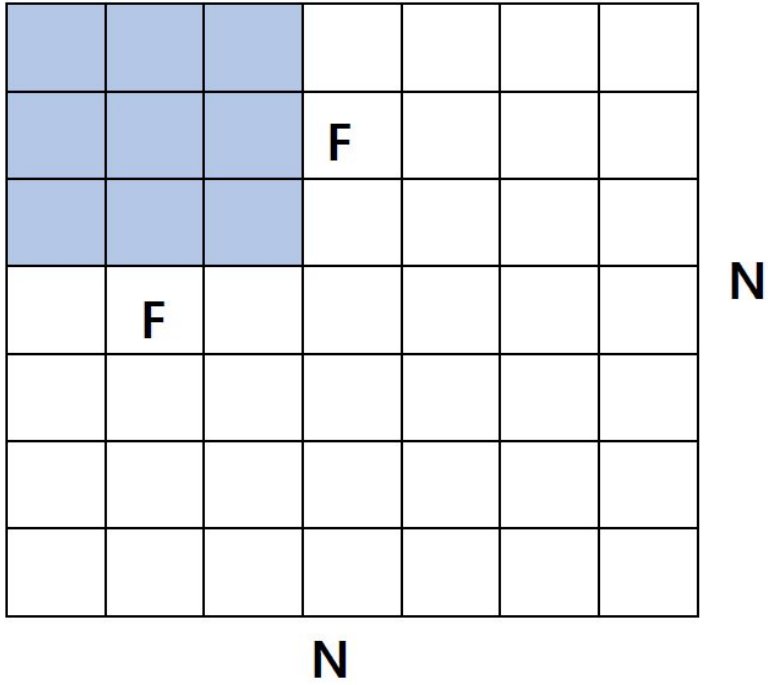
스트라이드 : 2



- filter size, stride의 크기에 따라 output size가 달라진다.

Convolution with stride

NxN Image, FxF Filter



$$O = (N-F)/Stride + 1$$

예시 1) 7x7 Image, 3x3 Filter, Stride 1인 경우

5 = (7-3)/1 + 1

따라서, 5x5 Output 생성

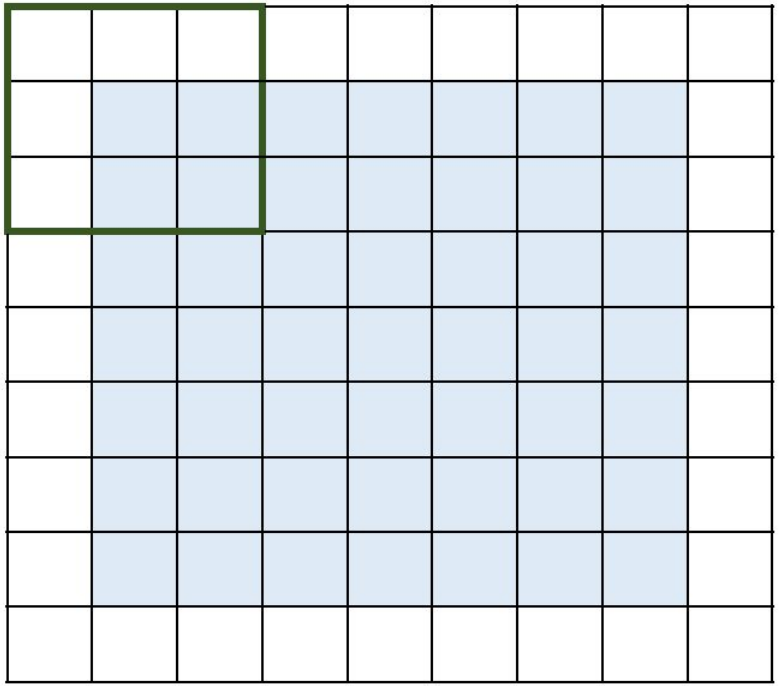
예시 2) 7x7 Image, 3x3 Filter, Stride 2인 경우

3 = (7-3)/2 + 1

따라서, 3x3 Output 생성

Convolution with padding

상하좌우로 1 픽셀 씩 Padding



7x7 Image, Padding 1, 3x3 Filter, Stride 1인 경우 Output의 크기는 얼마일까?

$$O = (N-F)/Stride + 1$$

$$7 = (9-3)/1 + 1$$

크기가 유지될 수 있음

$$O = ((N+2*P)-F)/Stride + 1$$

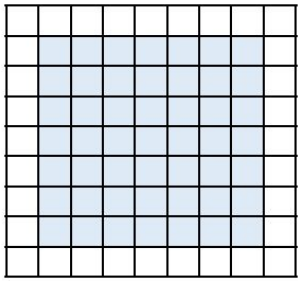
P : Padding

$$P = \frac{((O-1) * Stride - (N-F))}{2}$$

Convolution with padding

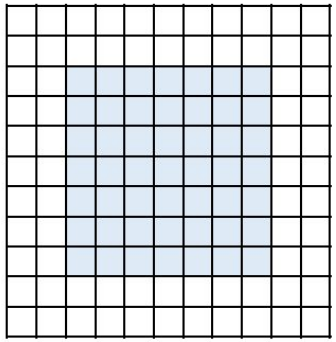
7x7 image, stride=1 일때 Same size를 위한 padding

3x3 Filter



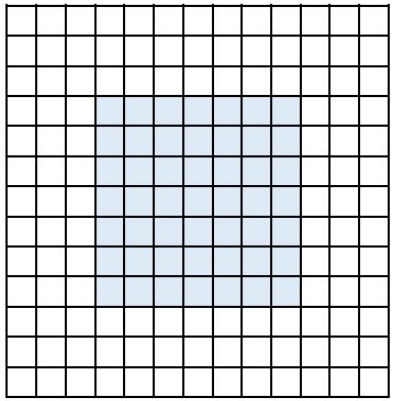
$P = ((7 - 1) * 1 - (7 - 3))/2$
$= (6 - 4)/2$
$= 1$

5x5 Filter



$P = ((7 - 1) * 1 - (7 - 5))/2$
$= (6 - 2)/2$
$= 2$

7x7 Filter

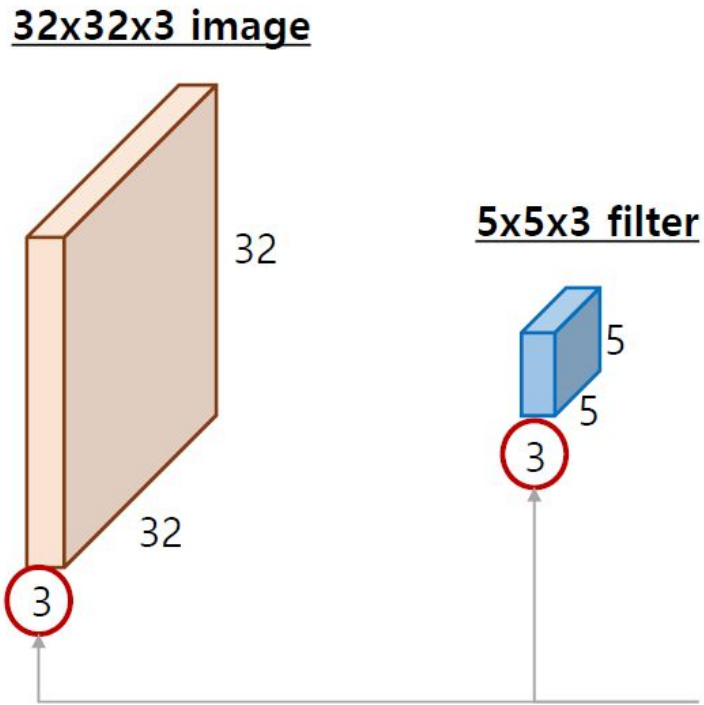


$P = ((7 - 1) * 1 - (7 - 7))/2$
$= (6 - 0)/2$
$= 3$

Convolution 연산

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산

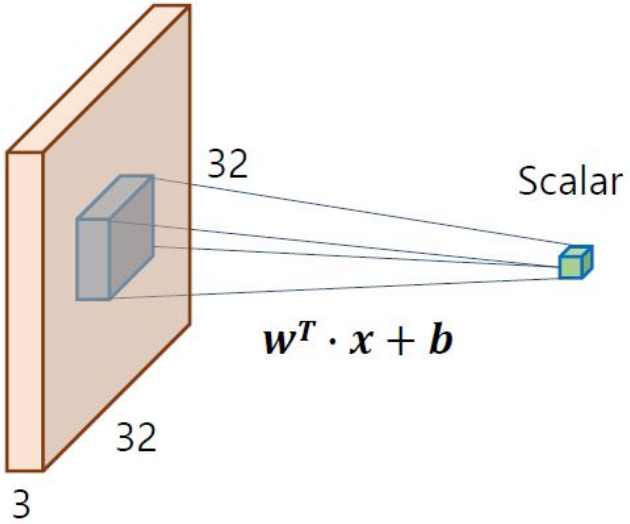


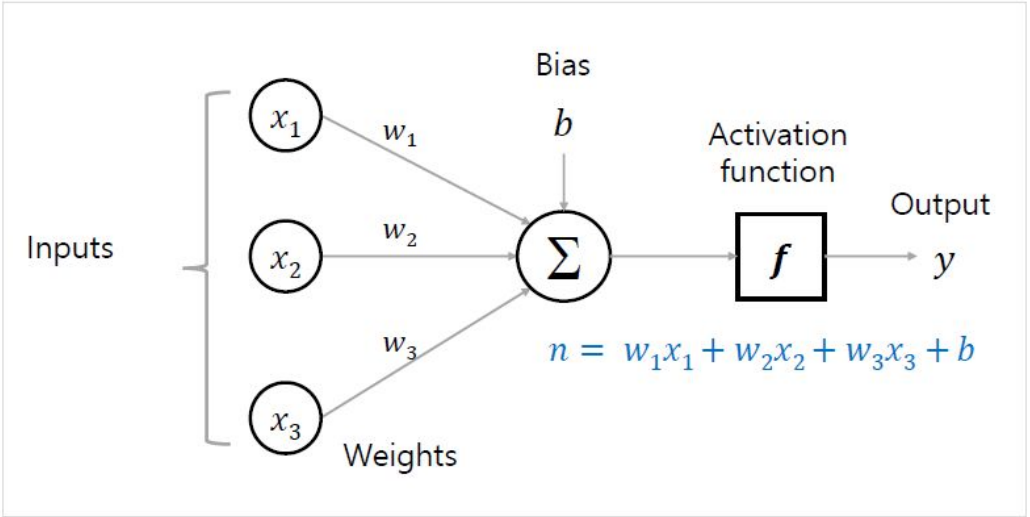
Image와 Filter의 Depth가 동일해야
Convolution 연산을 할 수 있다.

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산

뉴런의 관점을 생각해보자!

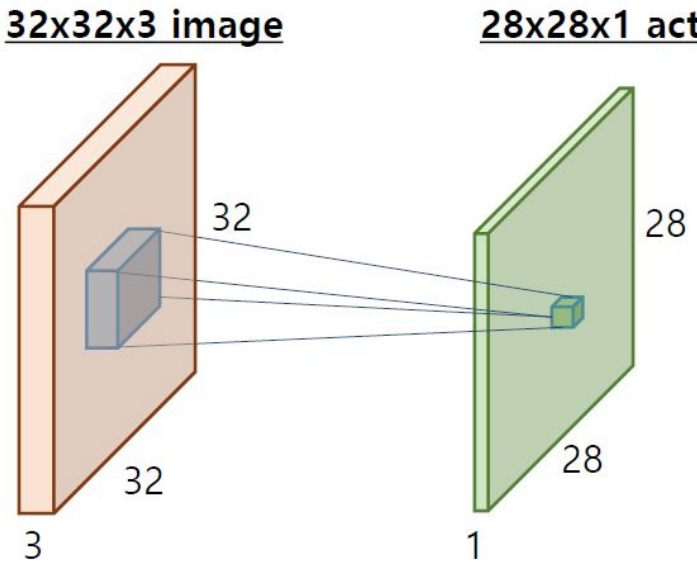




Local Connectivity를 갖는 뉴런

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산



- 모든 pixel에 대해 convolution을 하면 activation map이 한 개 생성됨

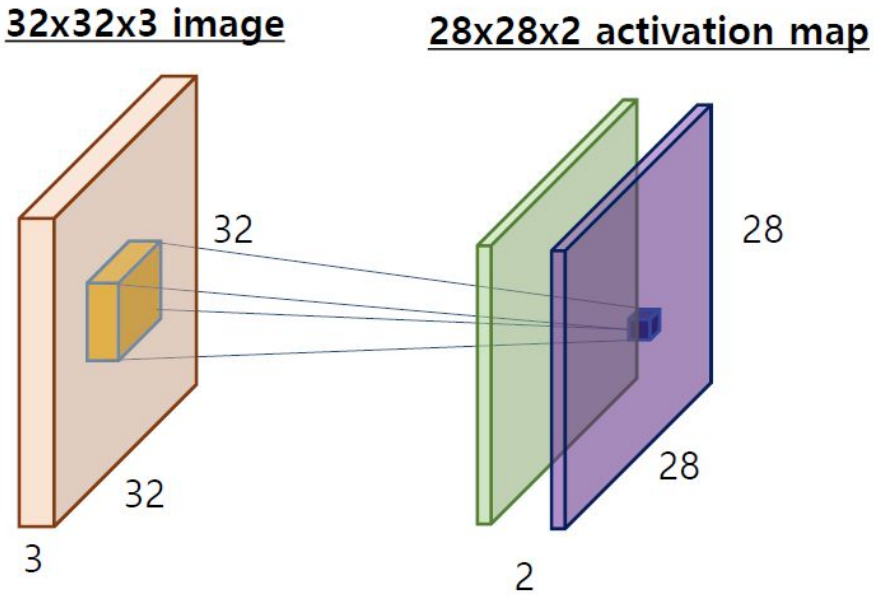
따라서, activation map은 28x28 뉴런 sheet이다.

1. 각 뉴런은 입력의 작은 영역에 연결
2. 모든 뉴런은 파라미터를 공유함

"5x5 filter" -> "5x5 receptive field for each neuron"

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, "Gradient-based learning applied to document recognition"

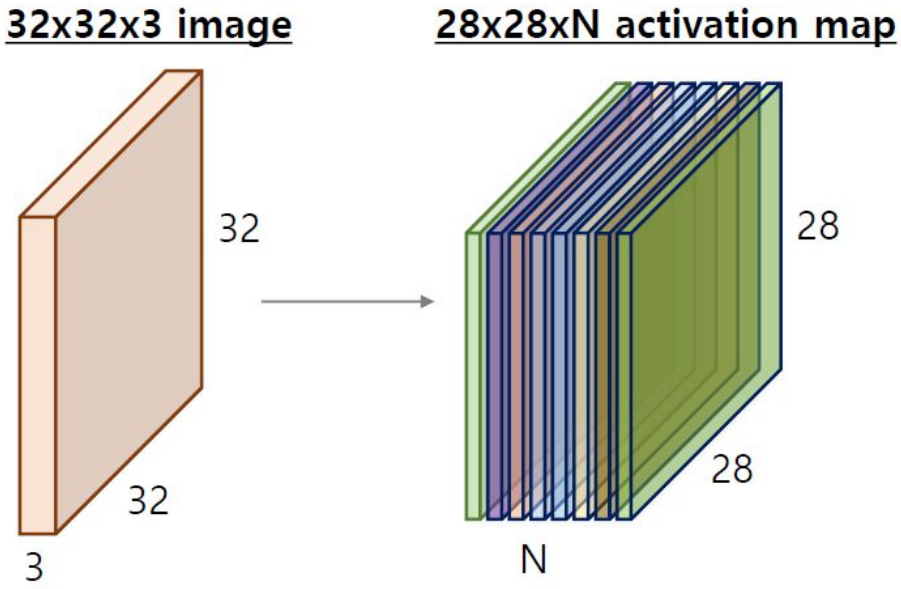
Convolution 연산



서로 다른 filter 별로 activation map이 생성된다.

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산

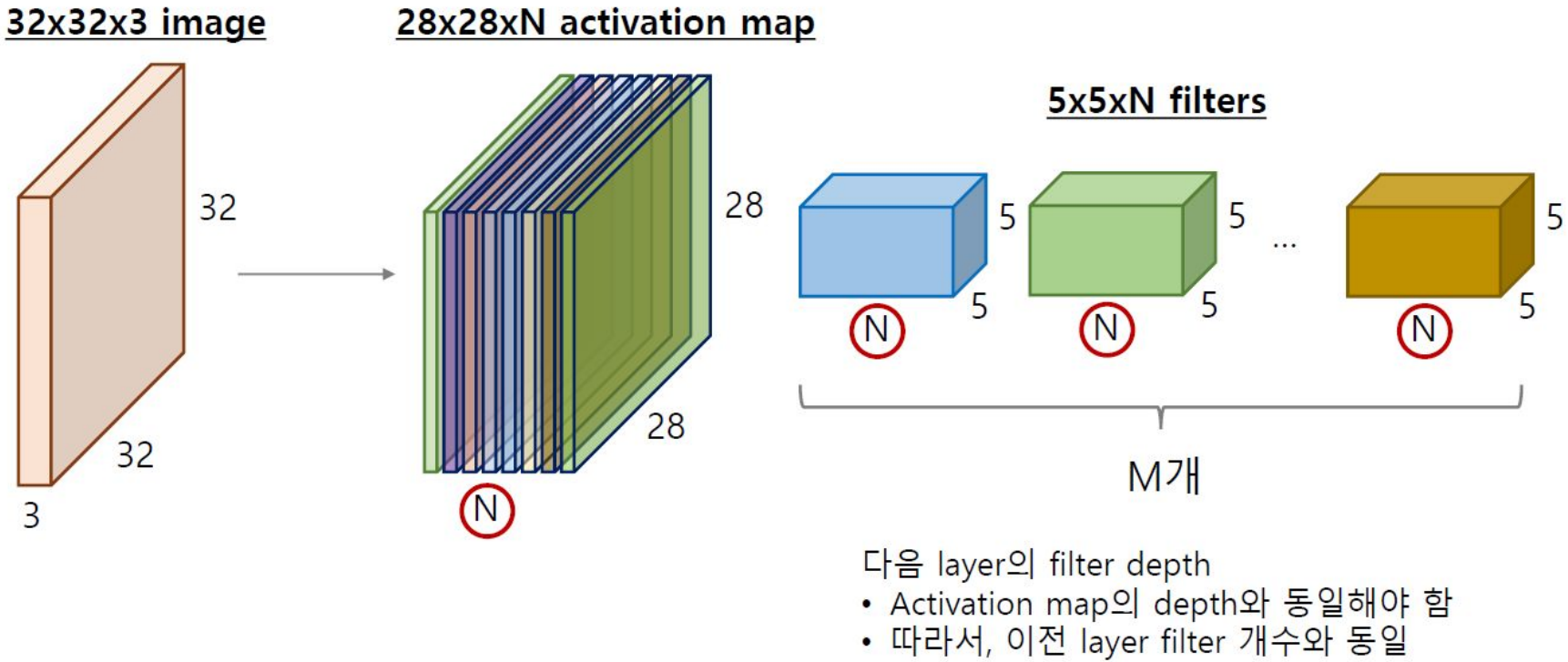


N 개의 5x5x3 filter 적용 후

- 여러 Filter를 사용해서 다양한 feature를 학습할 수 있도록 함
- Activation map은 filter 개수만큼 depth를 갖게 됨

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

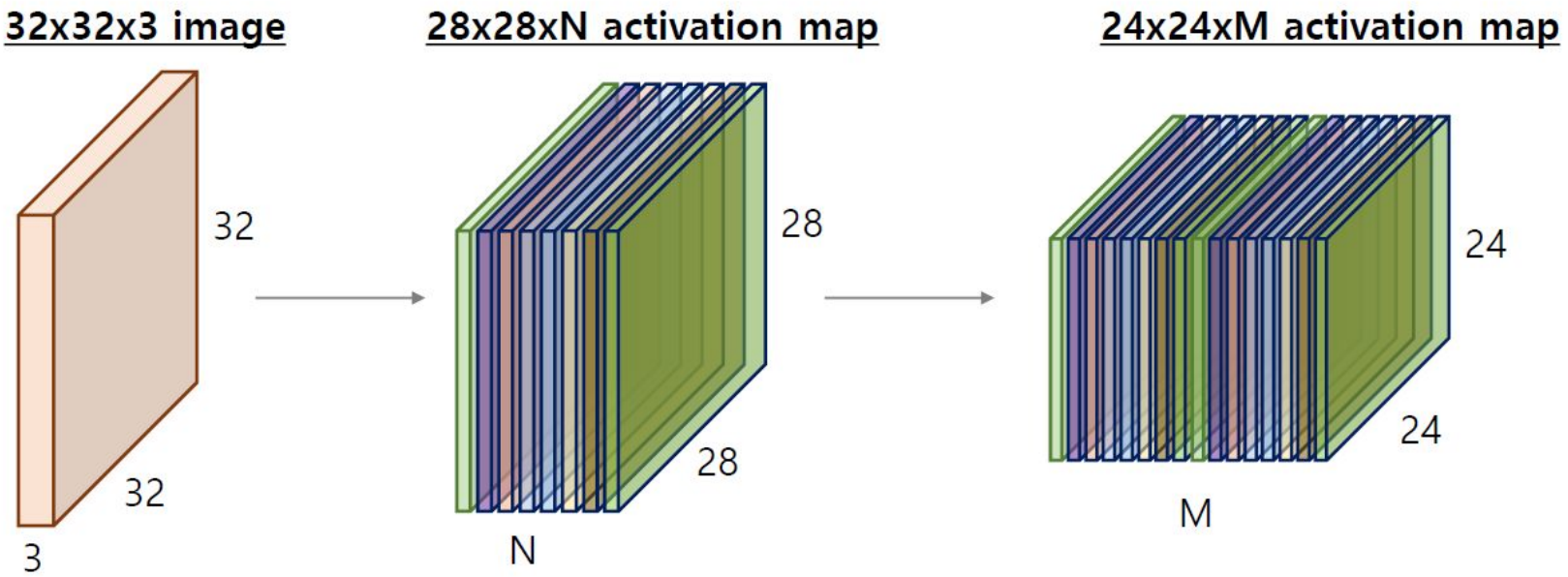
Convolution 연산



다음 layer의 filter depth
- Activation map의 depth와 동일해야 함
- 따라서, 이전 layer filter 개수와 동일

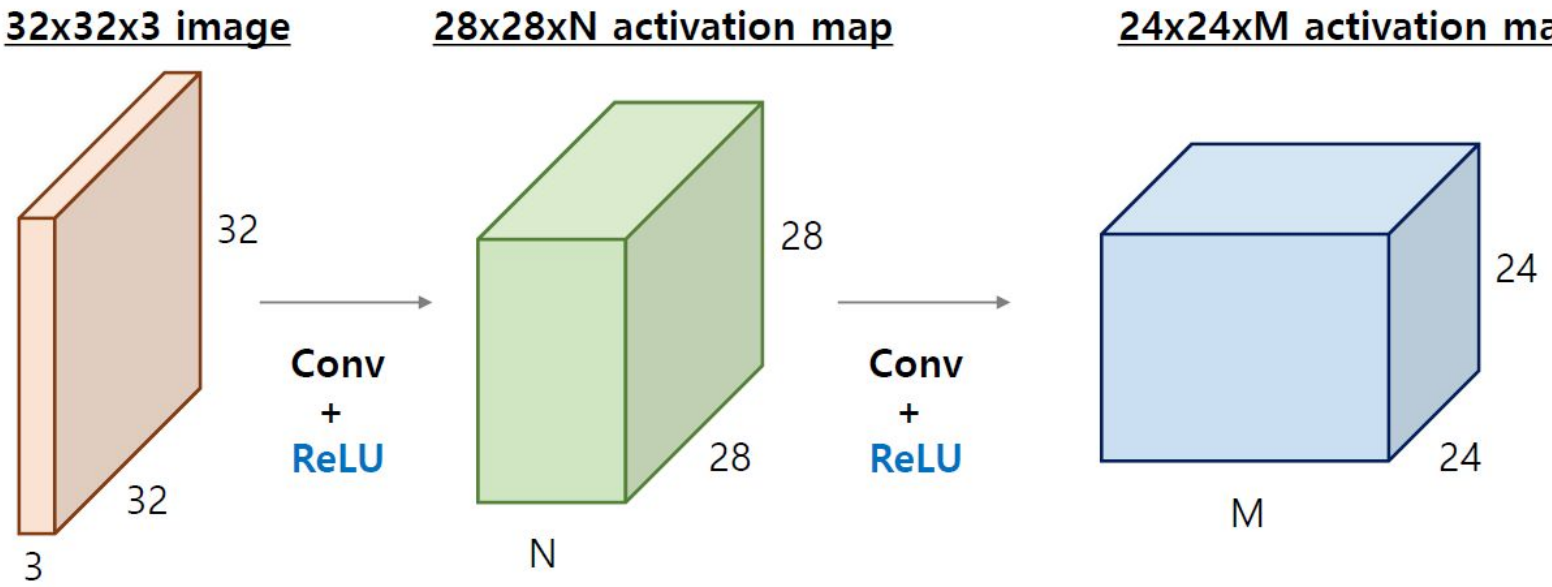
Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산



Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산

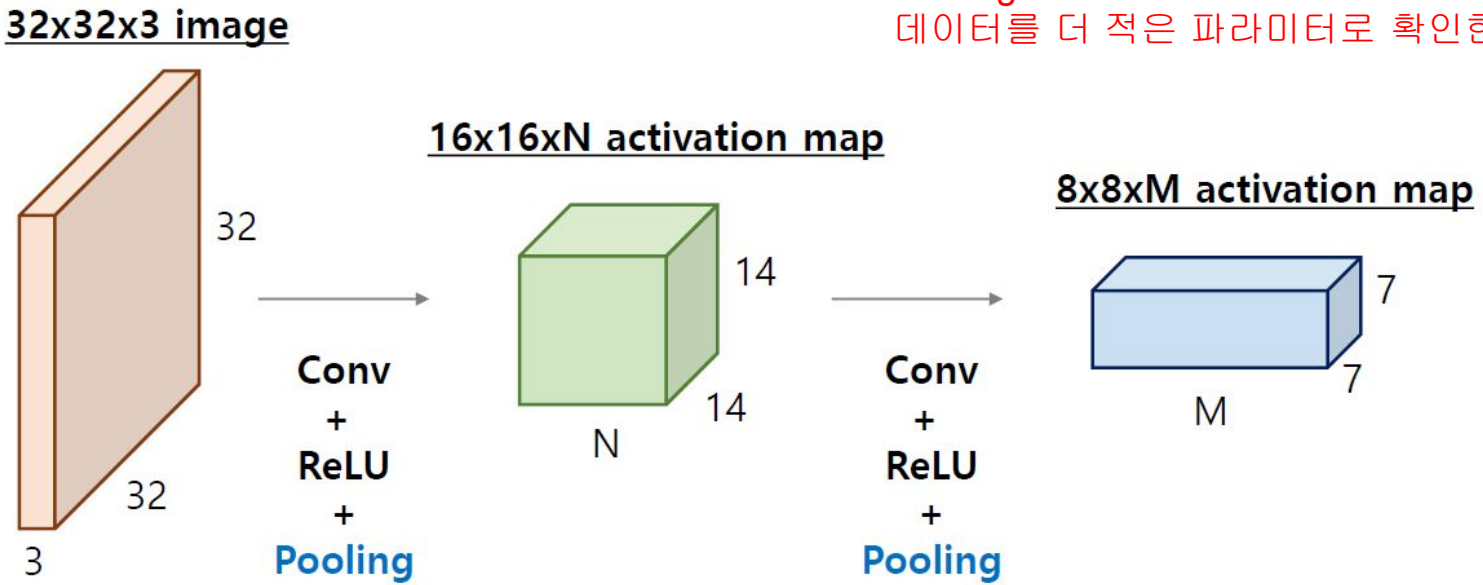


- 각 Layer 별로 convolution을 수행하고 activation function을 수행
- 비선형 연산을 통해 복잡한 분류 및 이미지 처리 연산을 수행

Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution 연산

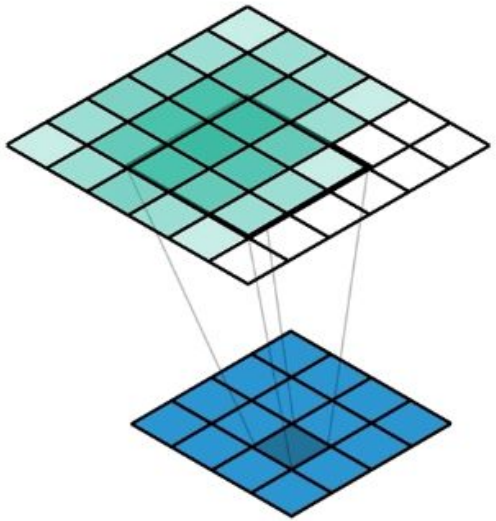
- Pooling을 통해 size를 줄이며 중요한 데이터를 더 적은 파라미터로 확인한다.



Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, “Gradient-based learning applied to document recognition”

Convolution NN Transpose

Convolution의 역연산으로 Upsampling을 진행



- 주어진 이미지의 한 픽셀을 생성된 Conv Transpose filter를 이용해 Upsampling

Input

1	1	1	1	
1	1	1	1	
1	1	1	1	
1	1	1	1	

Kernel

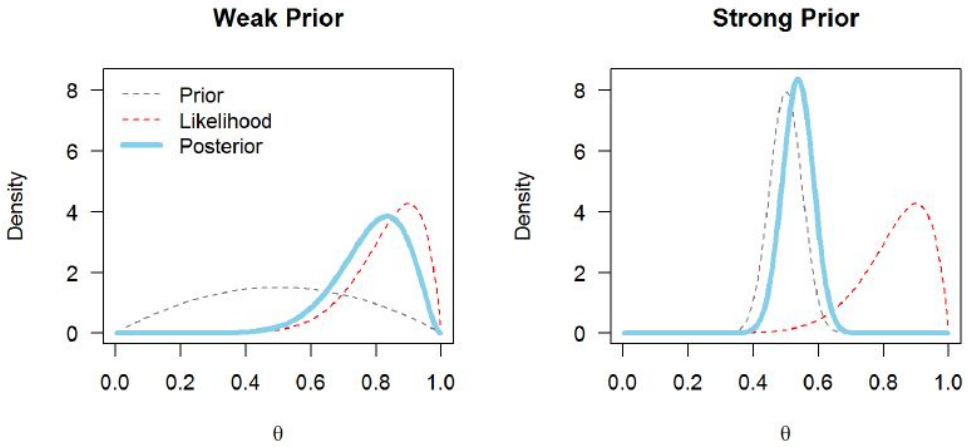
1	1	1
1	1	1
1	1	1

Output

1	2	3	3	2	1
2	4	6	6	4	2
3	6	9	9	6	3
3	6	9	9	6	3
2	4	6	6	4	2
1	2	3	3	2	1

https://medium.com/apache-mxnet/transposed-convolutions-explained-with-ms-excel-52d13030c7e8

CNN 가정사항



모델에 대한 믿음(Belief)을 파라미터의 사전 분포(Prior)로 표현

Weak Prior

- 높은 엔트로피를 갖는 분포
- Gaussian with high variance
- 데이터에 의해 파라미터가 자유롭게 변화함

Strong Prior

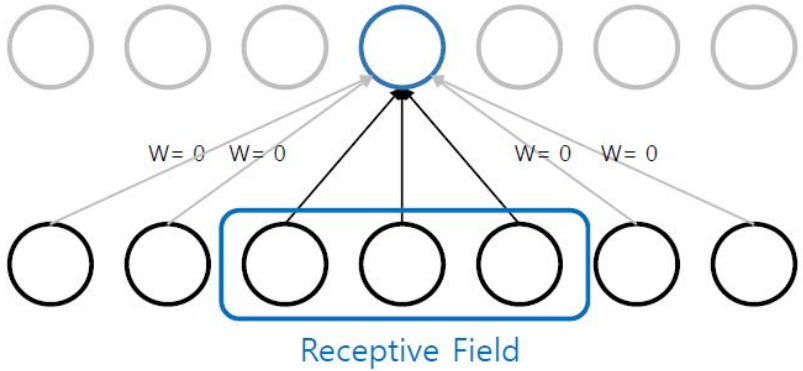
- 낮은 엔트로피를 갖는 분포
- Gaussian with low variance
- 파라미터를 결정할 때 사전 분포의 영향력이 매우 큼

Infinitely Strong Prior

- 일부 파라미터의 확률이 zero
- 데이터에 상관없이 확률이 zero인 파라미터는 사용할 수 없음

CNN as infinitely strong prior

Convolution as infinitely strong prior



- 계층 별로 가중치에 Infinitely Strong Prior를 적용
- Receptive Field 이외의 가중치는 0
- 모든 Hidden Unit에 대해 동일한 파라미터 사용

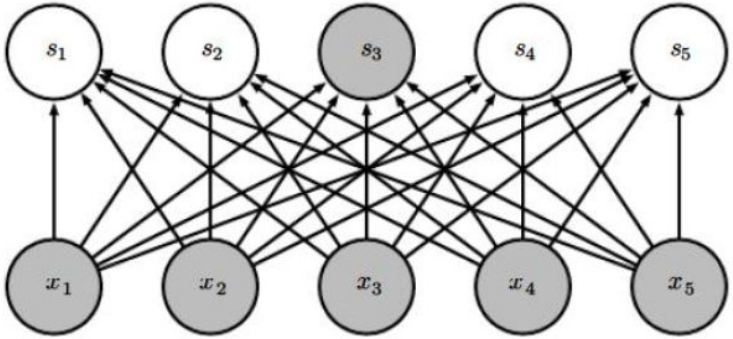
Convolution의 infinitely strong prior에 따라 다음 세가지 성질을 갖게 됨

- Sparse Interaction
- Parameter Sharing
- Equivariant

딥러닝의 성능을 향상시키는 중요한 아이디어!

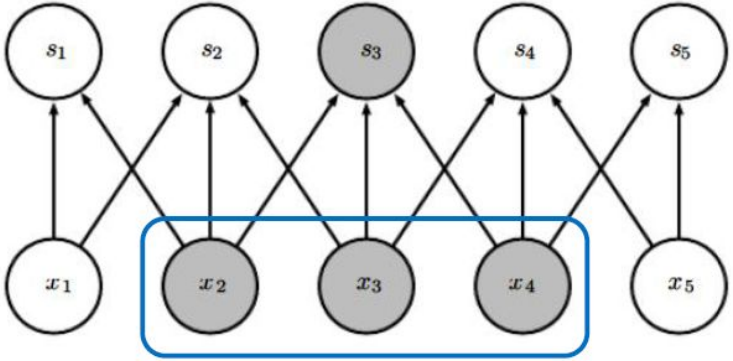
Sparse Interaction

Full Connectivity



- Output은 모든 input에 연결됨
- Parameter 수 : $O(m \times n)$
  - m : input 개수
  - n : output 개수

Sparse Connectivity



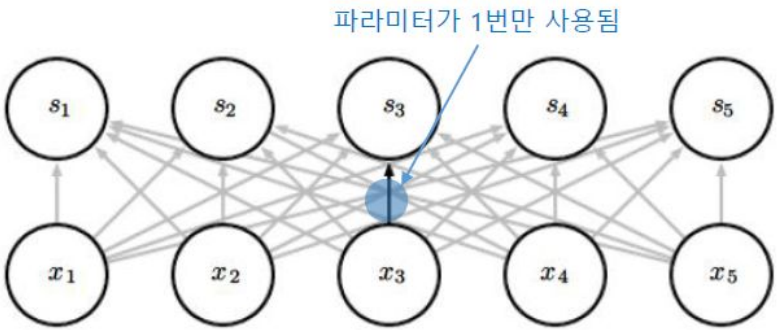
Receptive Field

- Output에 연결된 input이 제한적
- Parameter 수 : $O(k \times n)$
  - k : output과 연결된 connection 수
  - n : output 개수

메모리 및 계산 절약, 통계적 효율 향상

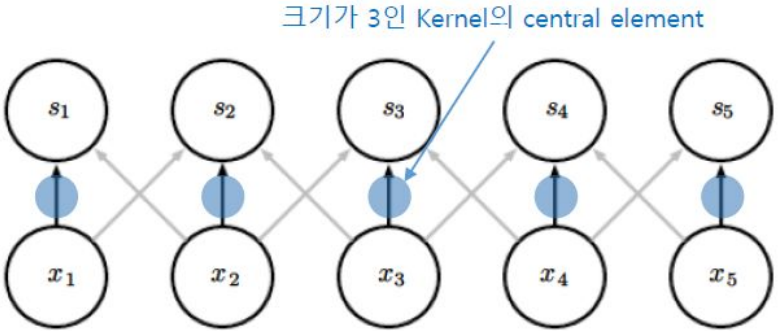
Parameter Sharing

No Parameter Sharing



- 각 파라미터는 한번만 사용됨
- Parameter 수 : $O(m \times n)$
  - m : input 개수
  - n : output 개수

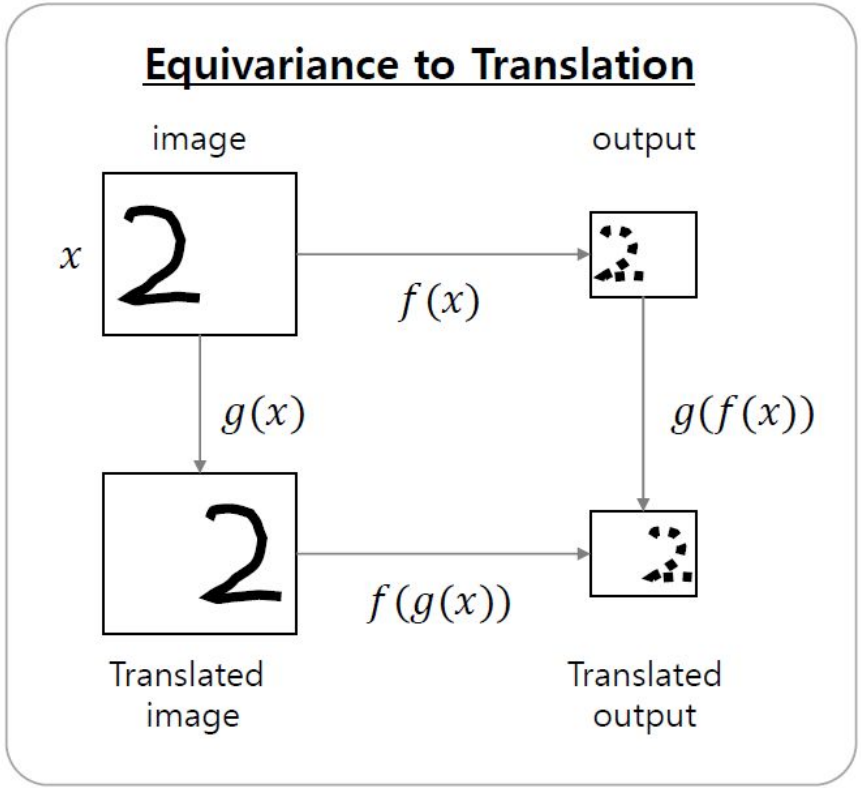
Parameter Sharing



- 모든 파라미터가 재사용됨
- Parameter 수 : $O(k)$
  - k : 각 output이 갖는 connection 수
  - n : output 개수

메모리 및 계산 절약, 통계적 효율이 극적으로 향상

Equivariance



Input이 이동한 만큼 output도 이동하는 성질

$$f(g(x)) = g(f(x))$$

g : Translation

f : Convolution

- Parameter Sharing으로 나타나는 효과
- Convolution은 scale, rotation에 대해서는 equivariant하지 않음

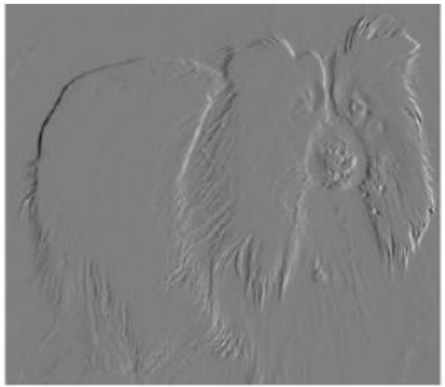
Parameter Sharing

Input의 여러 위치에 동일한 패턴의 정보를 처리할 때 유용

Edge Detection

320

280



Kernel

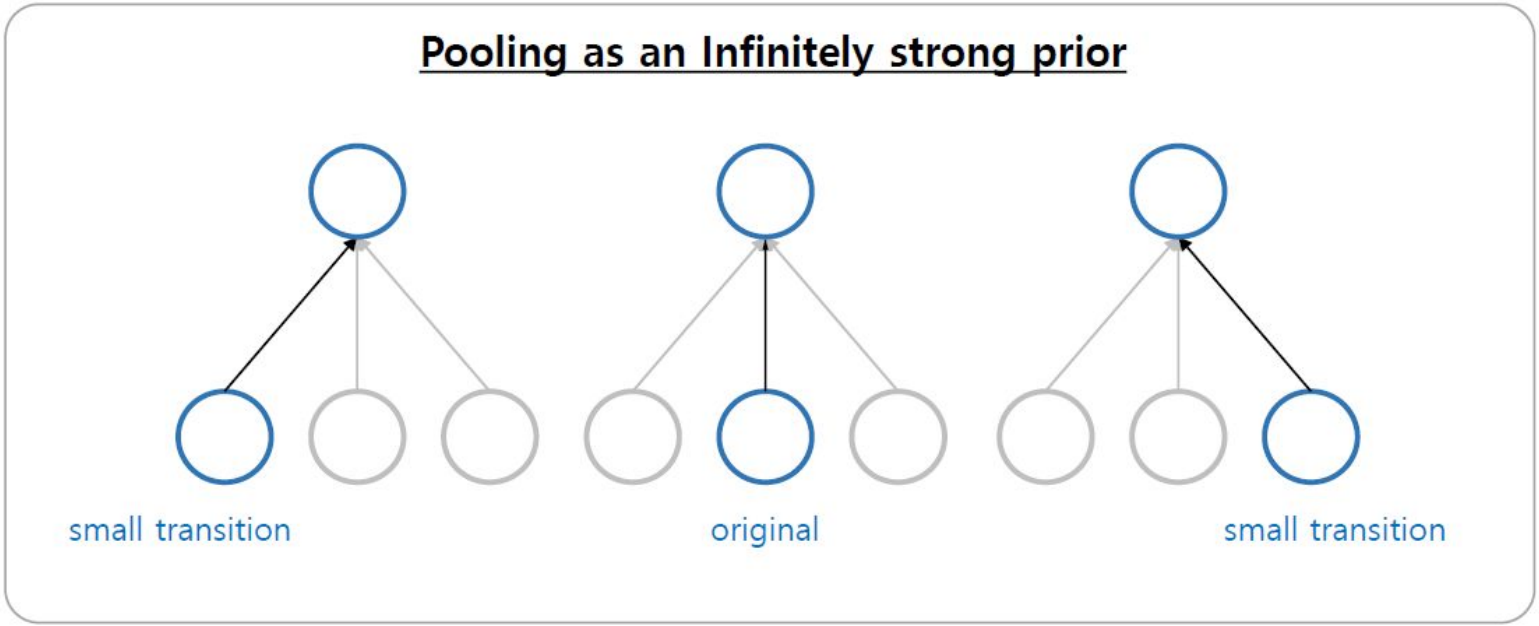
1	-1

Convolution : [319 x 280] x 3 = 267,960 float point operations

약 60,000 배 이상 계산 효율성 향상

행렬 연산 : [320 x 280] x [319 x 280] = 약 16,000,000 float point operations

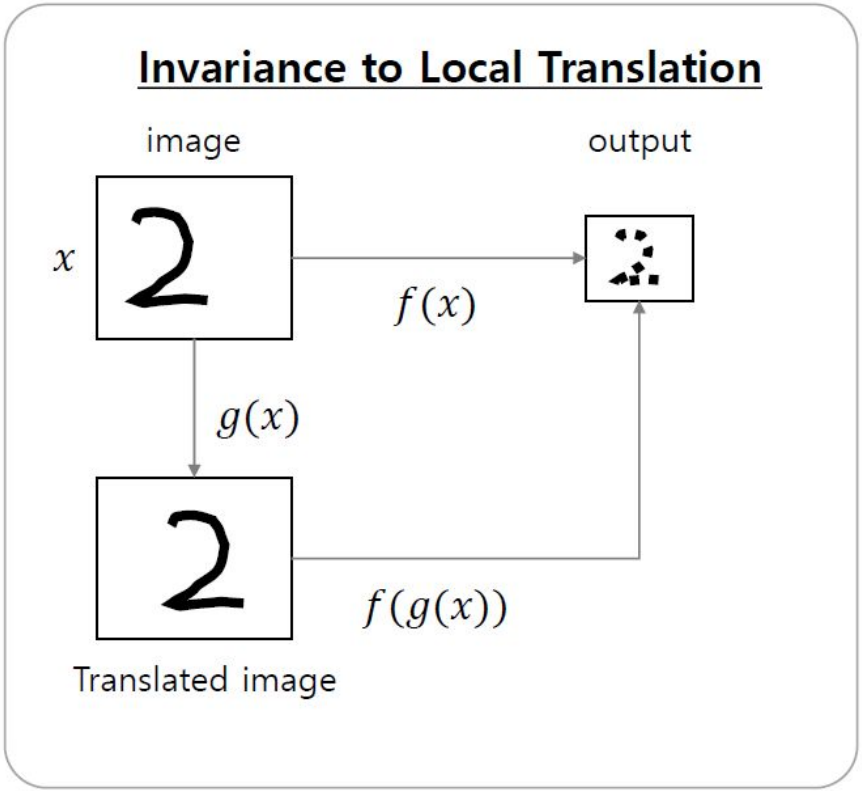
Pooling as infinitely strong prior



- Input에 조금의 변화가 있어도 Pooling은 동일한 결과를 얻을 수 있게 해 줌

Pooling의 Infinitely Strong Prior에 따라 Invariance to Local Translations을 갖게 됨

Invariance to Local Translation



Input이 조금 이동해도 output은 바뀌지 않는 성질

$$f(x) = f(g(x))$$

g : Translation

f : Pooling

"특징의 정확한 위치 보다 특징의 존재에 대해
더 관심이 있을 때 유용한 성질."

참고자료

- 밑바닥부터 시작하는 딥러닝 1, 2
  http://www.yes24.com/Product/Goods/34970929?Acode=101
  http://www.yes24.com/Product/Goods/72173703

- 모두를 위한 딥러닝 시즌2
  https://www.edwith.org/boostcourse-dl-tensorflow/joinLectures/22150

- 모두의 연구소 이일구, 윤성진님(CRAS Lab) 강의 자료
  https://github.com/ilguyi