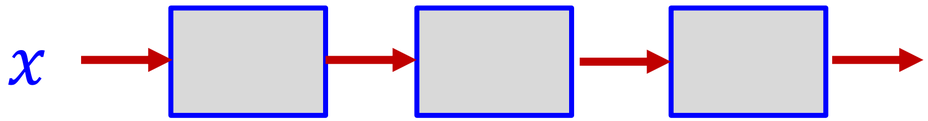
Lecture 11

Convolutional Neural Networks (CNN)

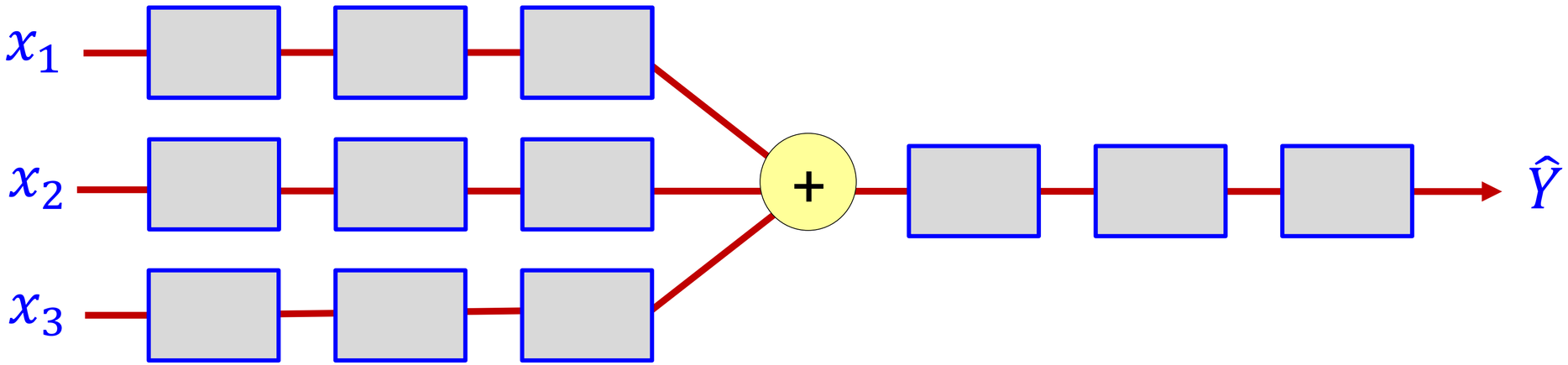
CNN Introduction

CNN Introduction



FC : Fully connected (forward)

Convolutional Neural Network (CNN)



CNN Introduction

A bit of history:

Hubel & Wiesel,

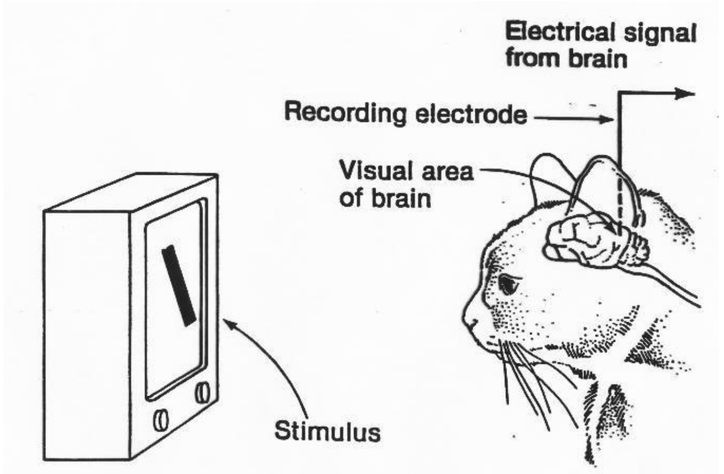
1959

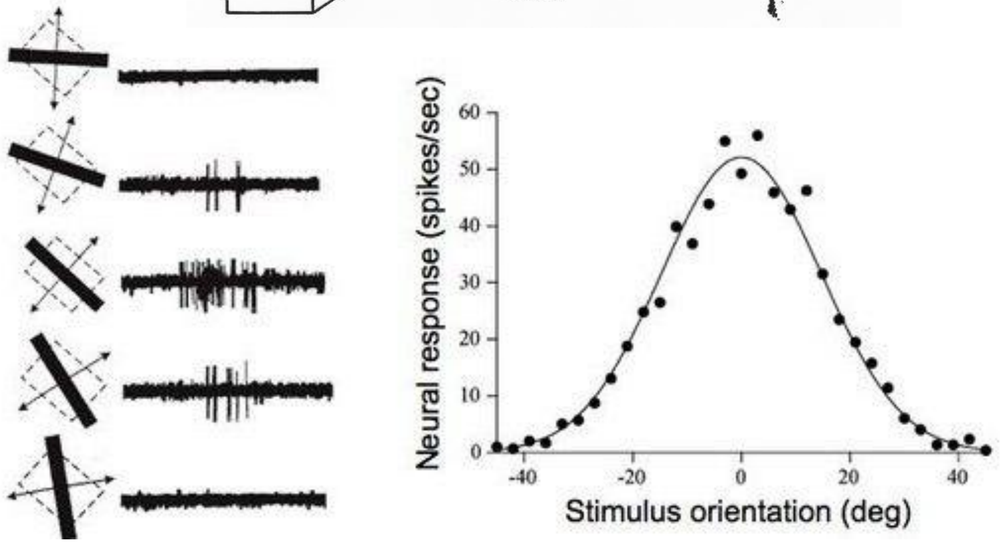
RECEPTIVE FIELDS OF SINGLE NEURONES IN THE CAT'S STRIATE CORTEX

1962

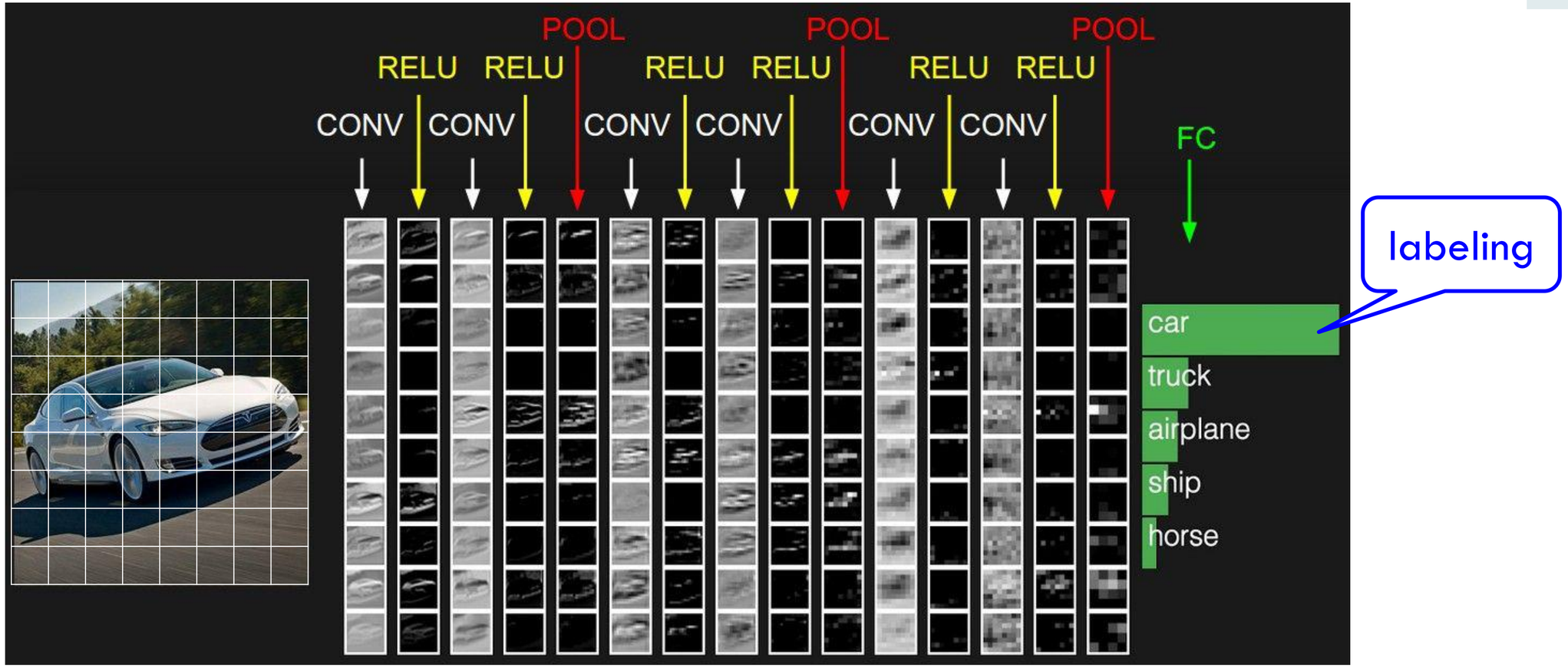
RECEPTIVE FIELDS, BINOCULAR INTERACTION AND FUNCTIONAL ARCHITECTURE IN THE CAT'S VISUAL CORTEX

1968...

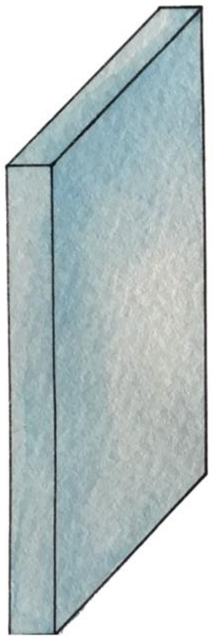




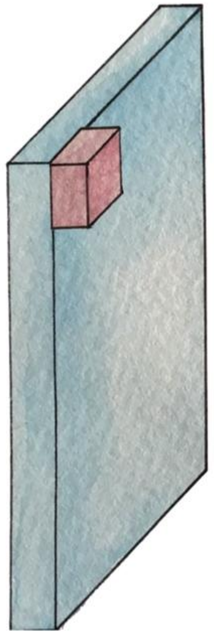
CNN Introduction



Start with an image (width x height x depth)

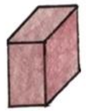


32x32x3 image

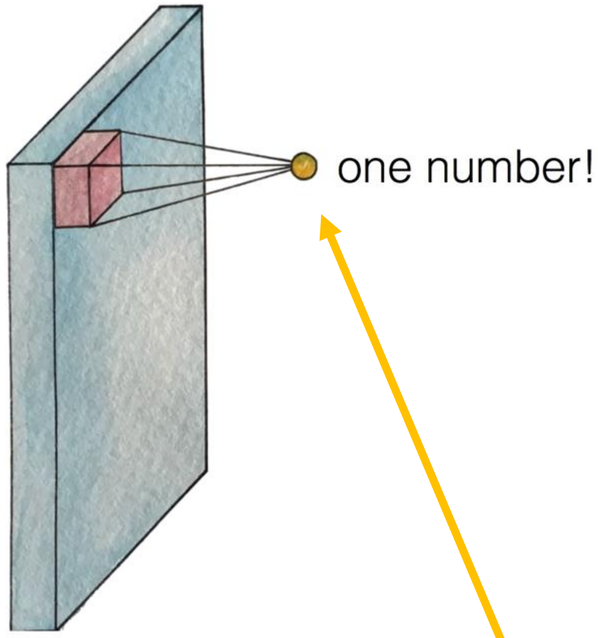


32x32x3 image

Let's focus on a small area only (5x5x3)



5x5x3 filter

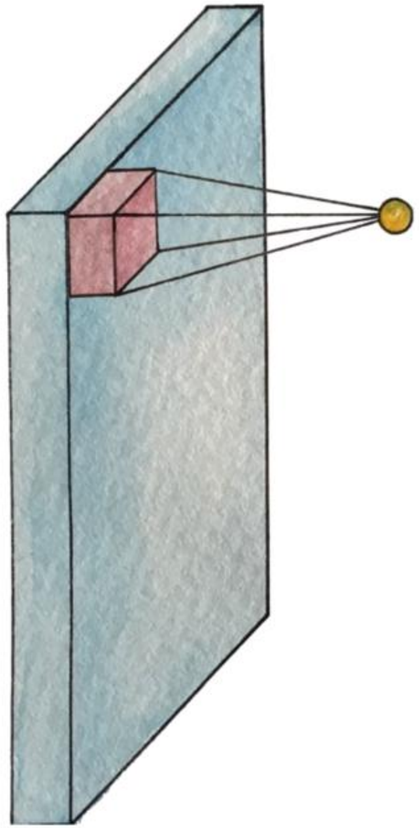


32x32x3 image

Start with an image (width x height x depth)

Get one number using the filter

Get one number using the filter



32x32x3 image

one number! =Wx+b

e.g. =ReLU(Wx+b)

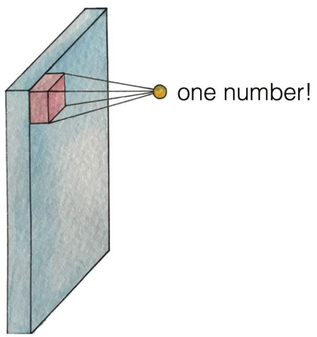
5x5x3 filter

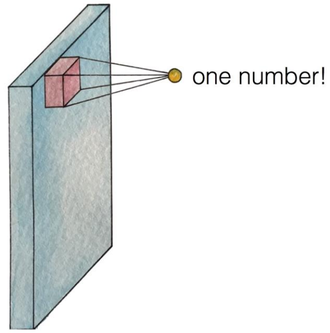
R $\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + b$

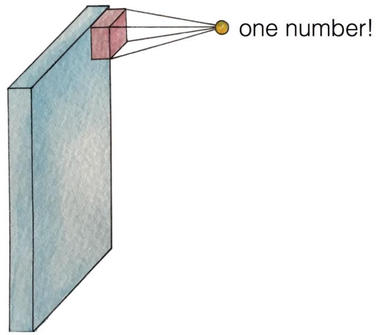
G $\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + b$

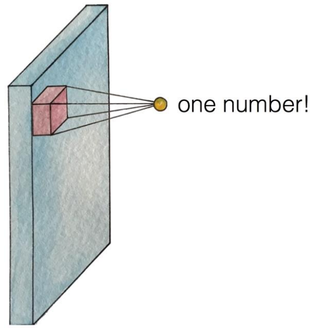
B $\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + b$

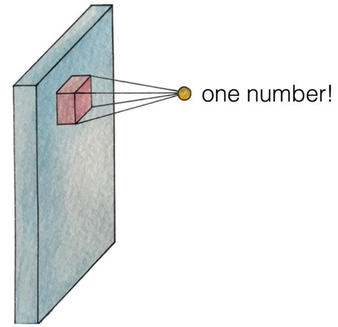
Let's look at other areas **with the same filter (w)**







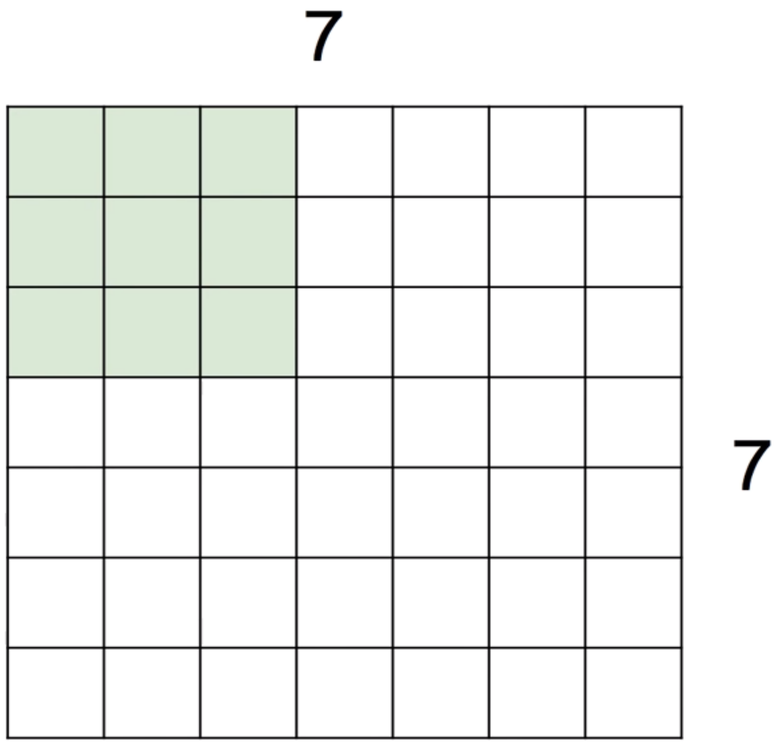




32x32x3 image

How many numbers can we get?

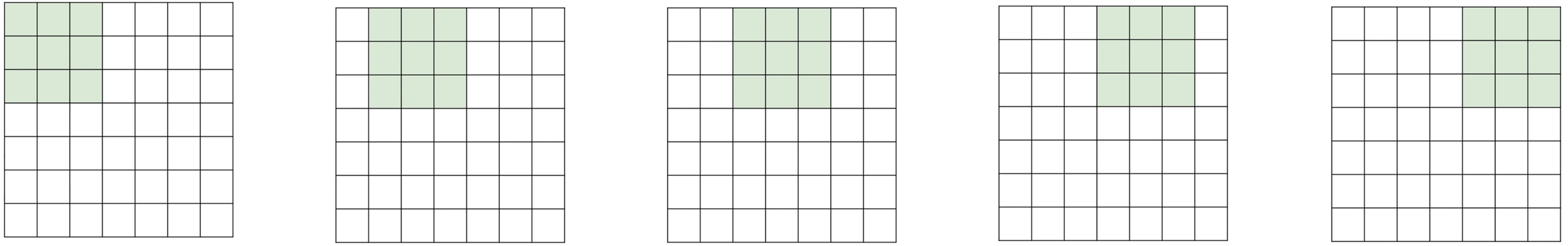
A closer look at spatial dimensions



7x7 input (spatially)
assume 3x3 filter

A closer look at spatial dimensions

7x7 input (spatially)
assume 3x3 filter
applied with stride 1



How many numbers can we get?

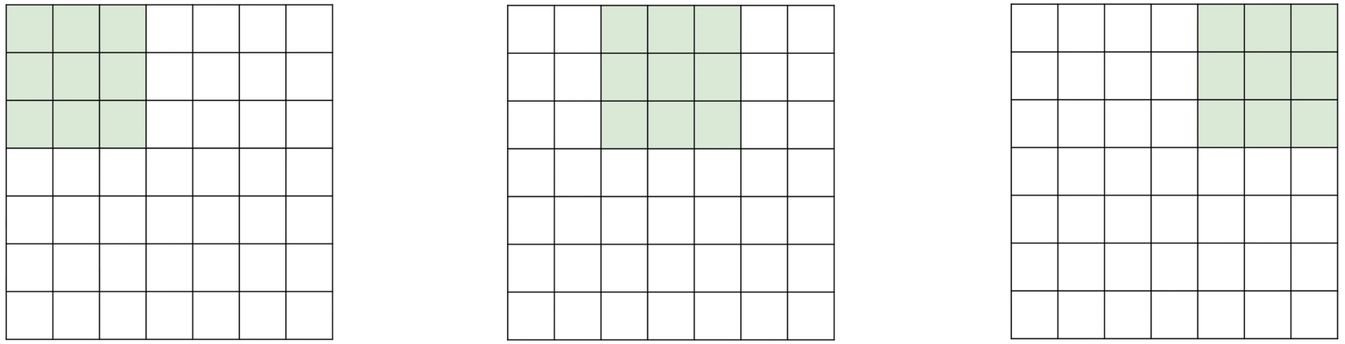
if stride size = 1

then

Horizontal direction : 5번
Vertical direction : 5번
→ 5 X 5 output

A closer look at spatial dimensions

7x7 input (spatially)
assume 3x3 filter
applied with stride 2



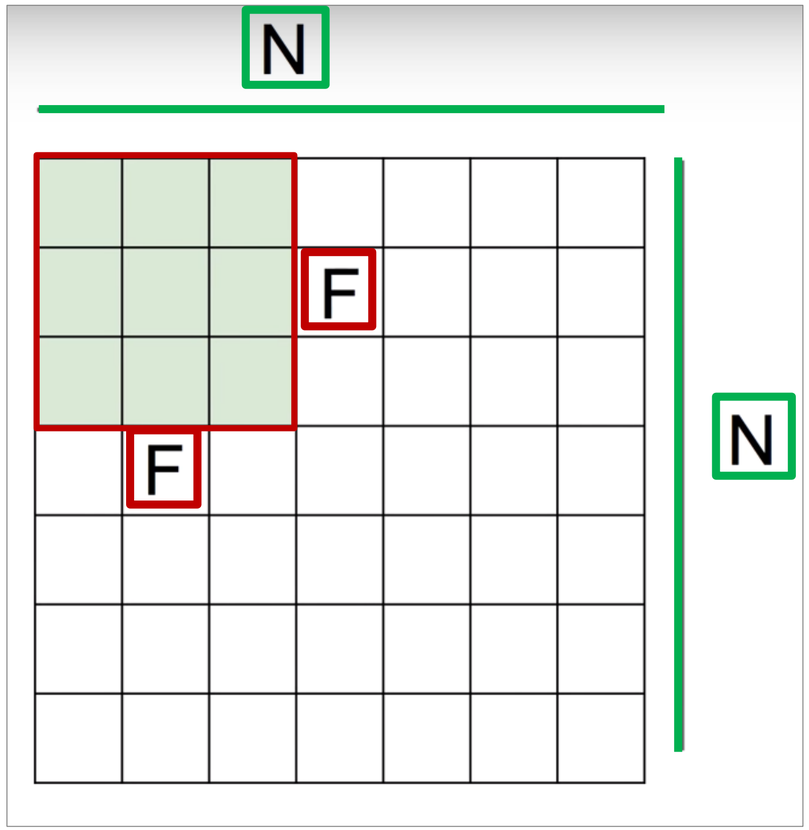
How many numbers can we get?

if stride size = 2

then

Horizontal direction : 3번
Vertical direction : 3번
→ 3 X 3 output

A closer look at spatial dimensions



수평 또는 수직의 한 방향에서 보면…

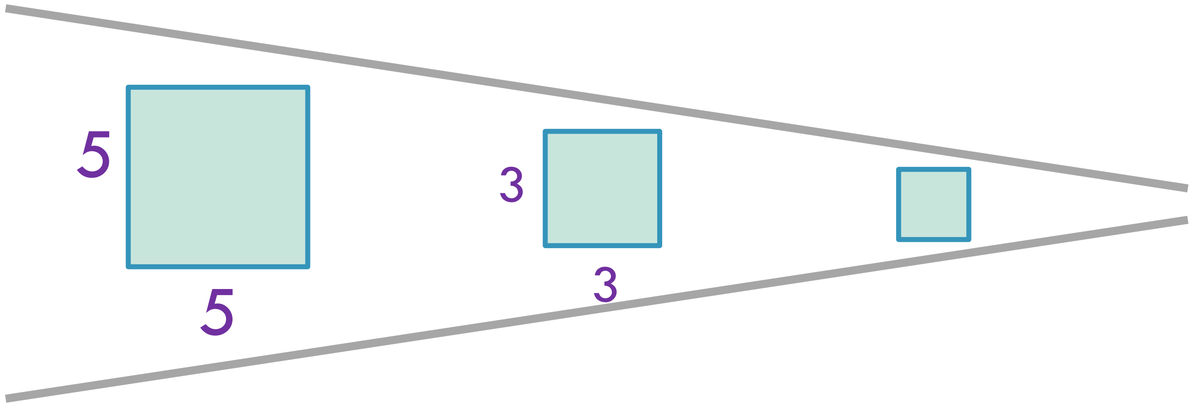
Output size → (N - F) / stride size + 1

e.g. N = 7, F = 3 :

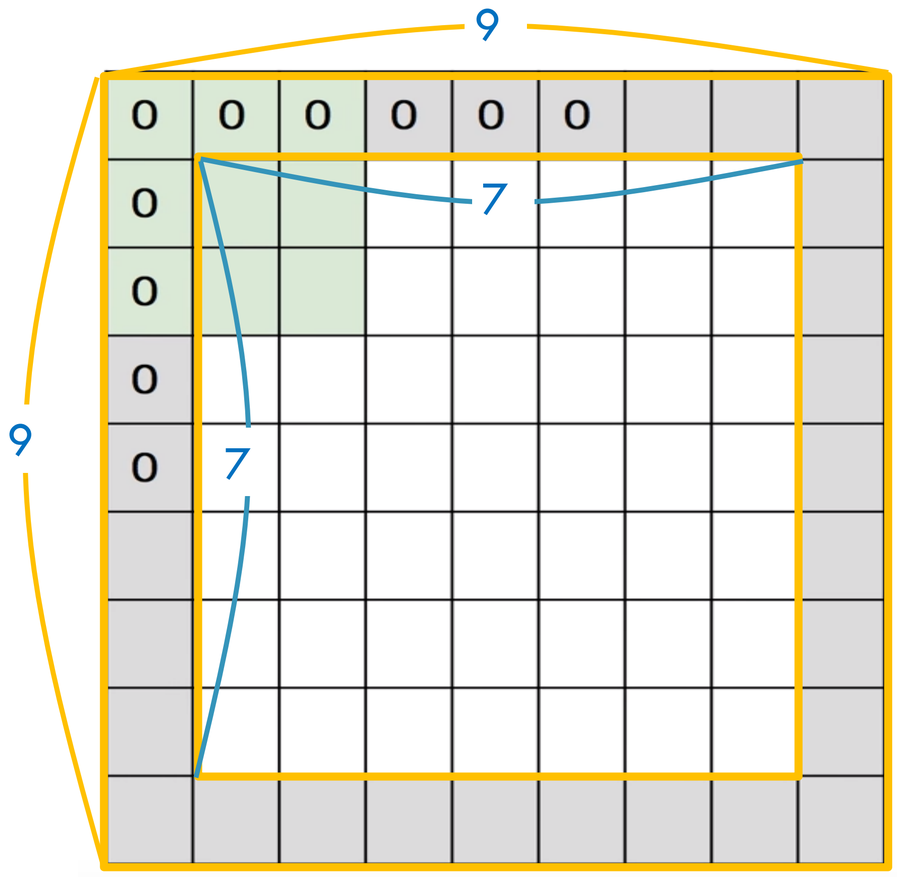
if stride 1 → (7 – 3) / 1 + 1 = 5 개

if stride 2 → (7 – 3) / 2 + 1 = 3 개

~~if stride 3 → (7 – 3) / 3 + 1 = 2.33 개 ?~~



Output, zero pad the border



e.g.

input : 7x7 , filter : 3x3 , stride : 1 , pad : 1

→ **what is the output?**

7x7 → 9x9 (expansion)

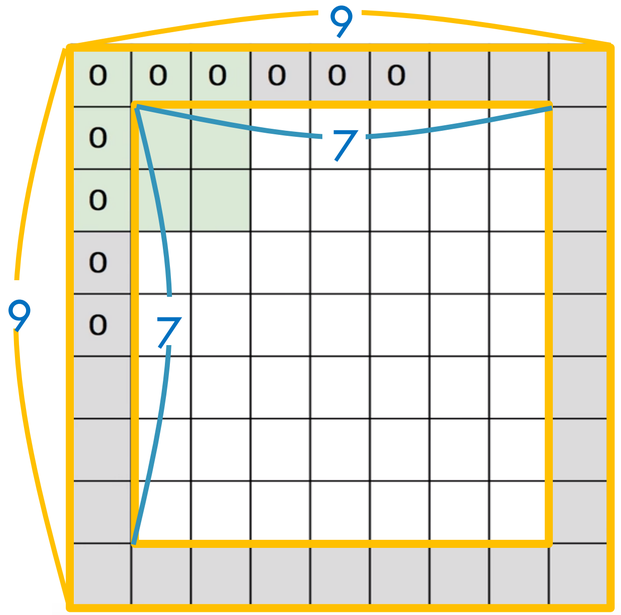
수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

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

output : 7x7

Padding size



e.g.

input : 7x7 , filter : 3x3 , stride : 1 , pad : 1

→ what is the output?

=

output : 7x7

in general, common to see CONV layers with stride 1,

filters of size FxF, and zero-padding with **(Filters of size – 1) / 2**

(will preserve size spatially).

e.g.

padding size

(3 - 1) / 2 → 1 → if filter size = 3 then zero pad : **1**

(5 - 1) / 2 → 2 → if filter size = 5 then zero pad : **2**

(7 - 1) / 2 → 3 → if filter size = 7 then zero pad : **3**

Swiping the entire image

filter size = 5

padding size = (Filters of size – 1) / 2

(5 – 1) / 2 = 2

input : 32 → **36 으로 확장**

(32 + **2 * 2**) = 36

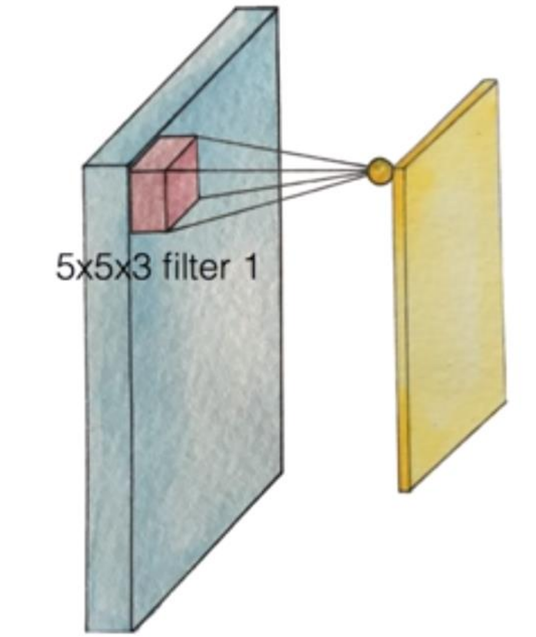
(padding : 수평 방향으로 좌, 우 1개씩 → 2개)

(padding : 수직 방향으로 좌, 우 1개씩 → 2개)

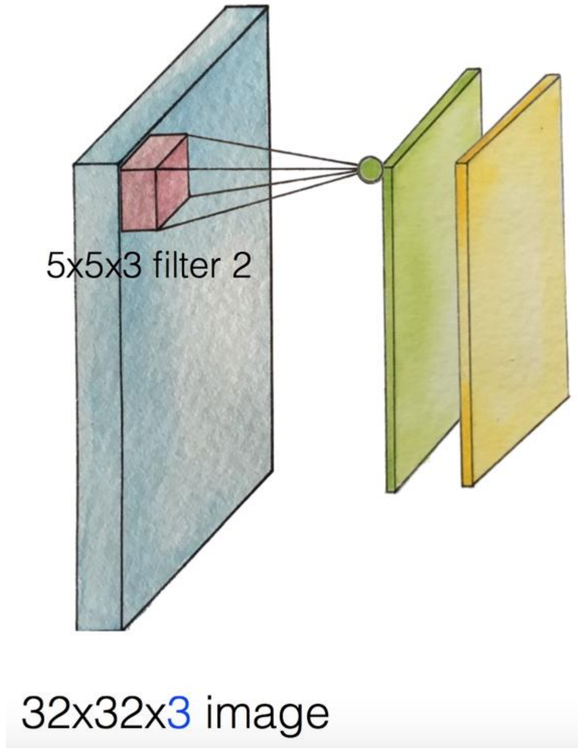
Output size → **(N – F) / stride size** + 1

Output size → **(36 – 5) / 1** + 1 = 32

output : 32

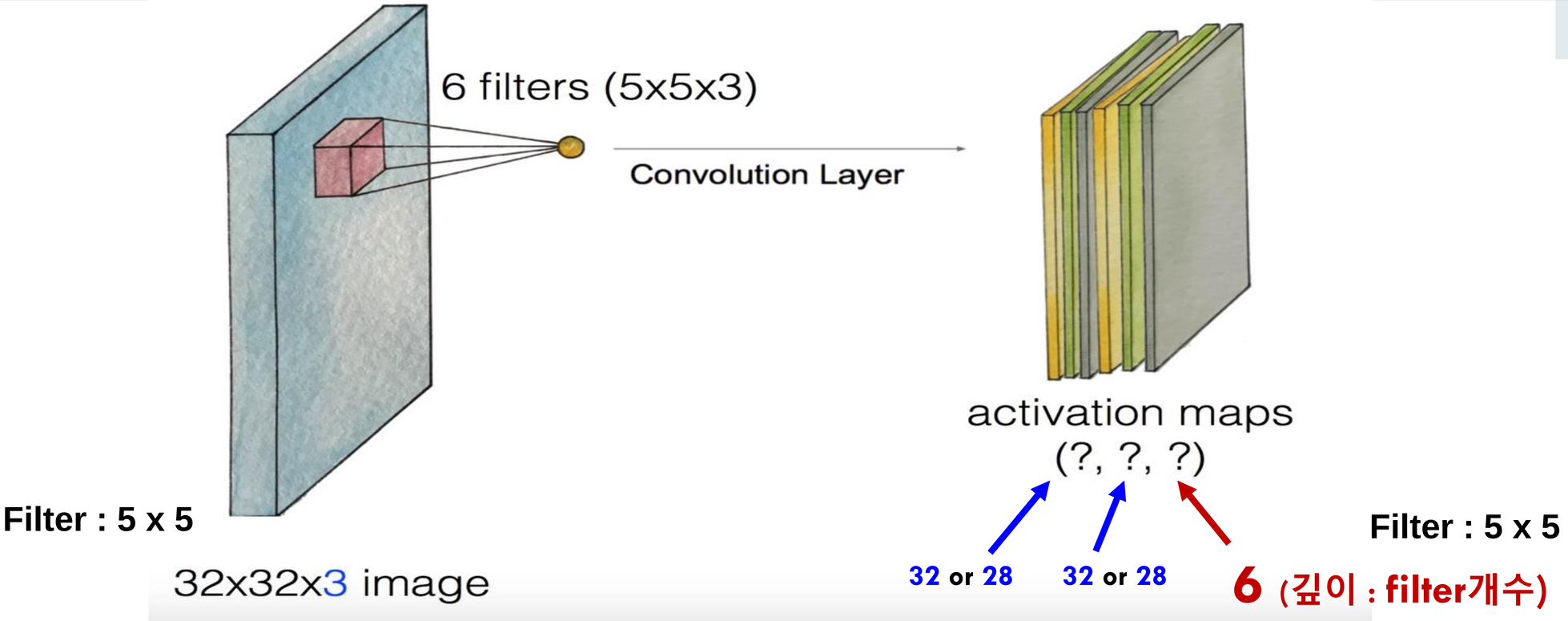


Filter : 5 x 5



Filter : 5 x 5

Swiping the entire image



if) Padding 사용했다면….

Output size → **(36 – 5) / 1** + 1 = **32**

Output size → **(N – F) / stride size** + 1

Output size : **32 x 32 x 6**

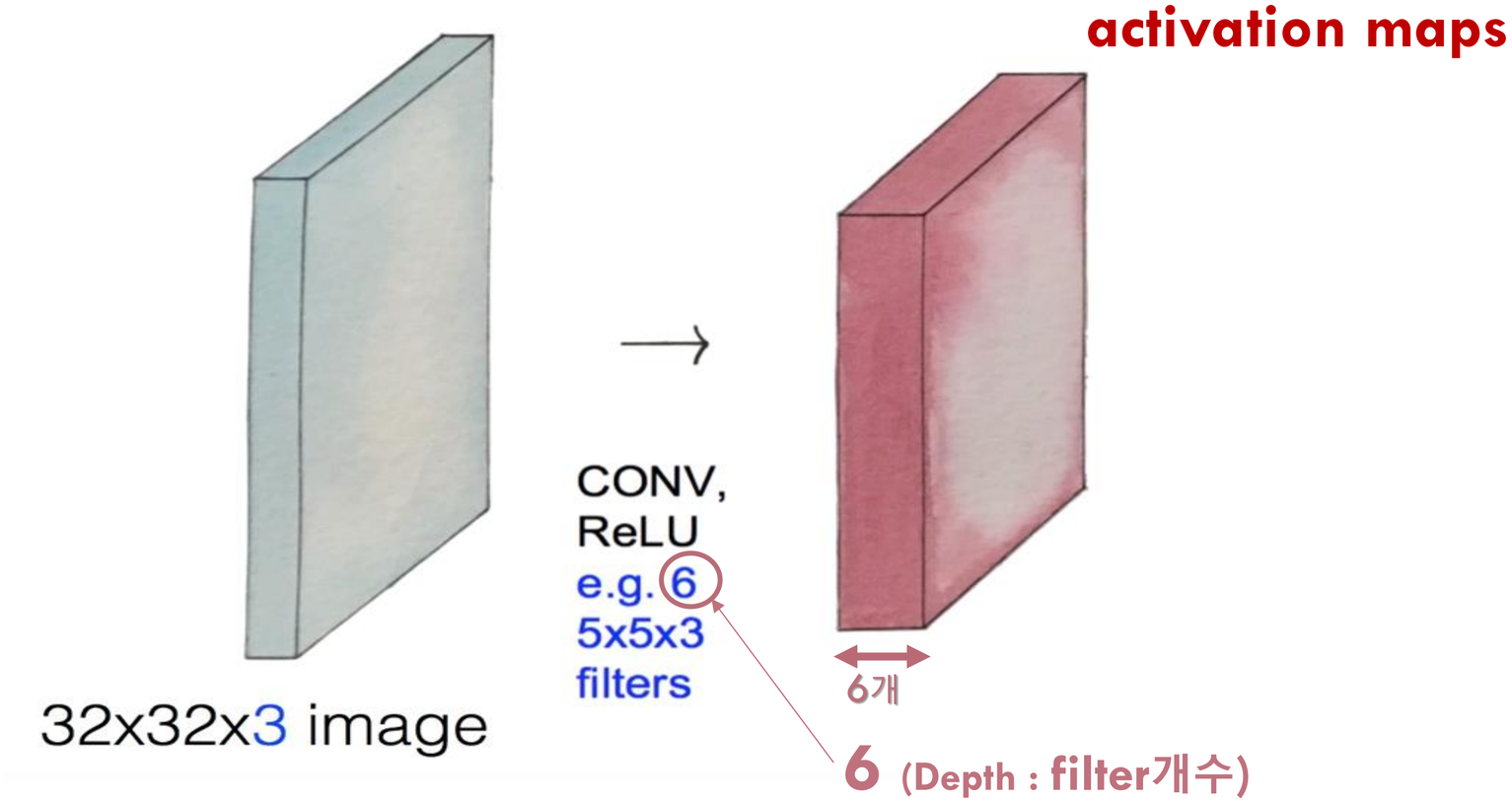
if) Padding 사용하지 않았다면….

Output size → **(32 – 5) / 1** + 1 = **28**

Output size → **(N – F) / stride size** + 1

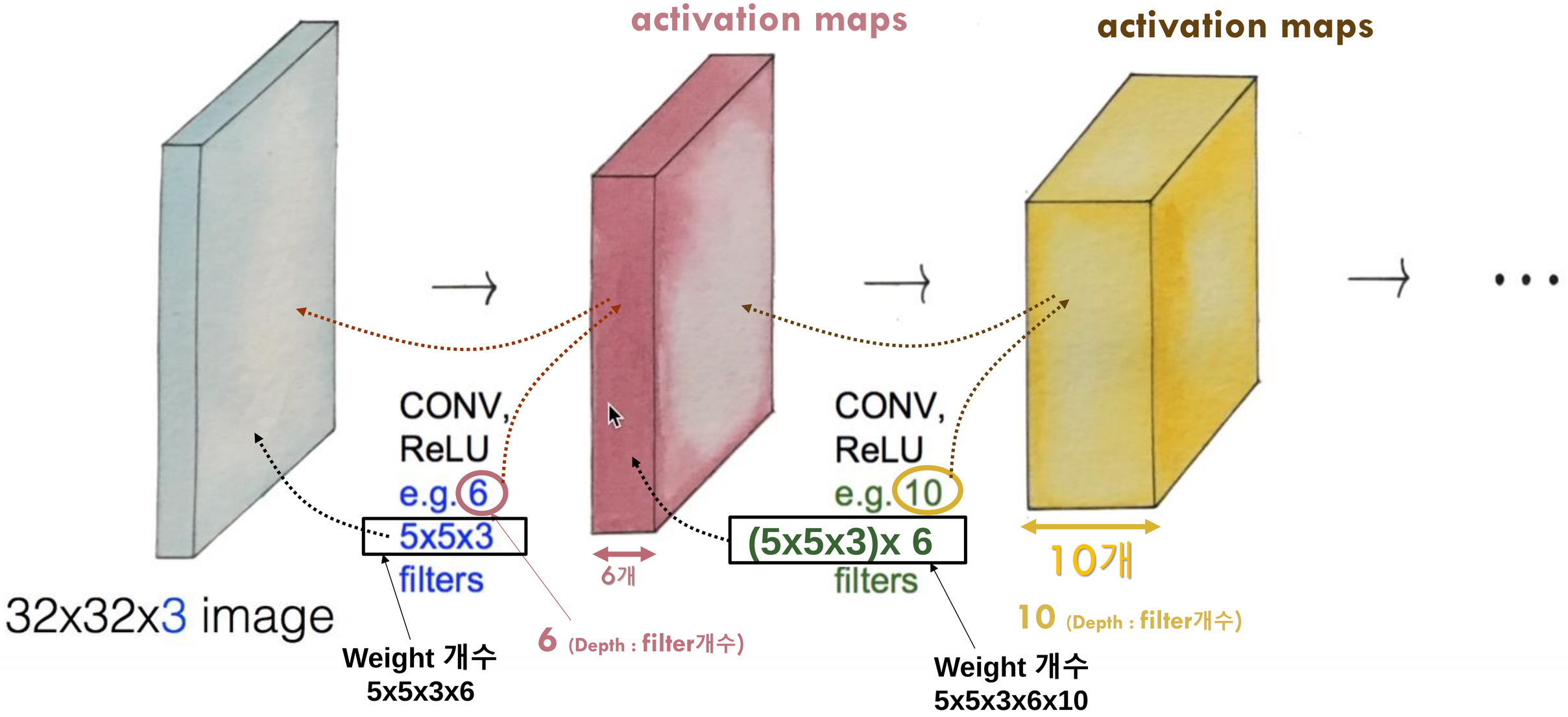
Output size : **28 x 28 x 6**

Convolution layers



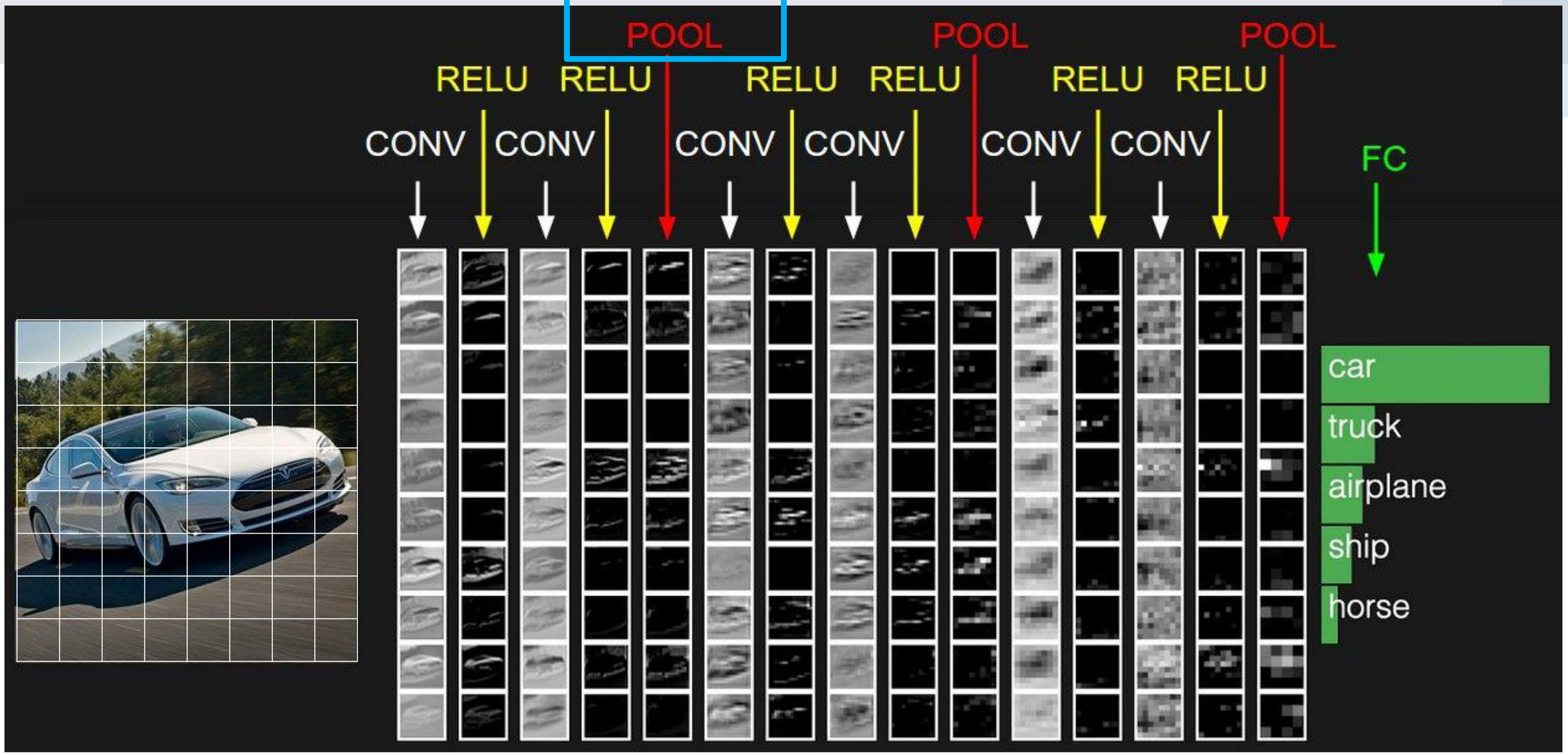
Convolution layers

How many weight variables? How to set them?



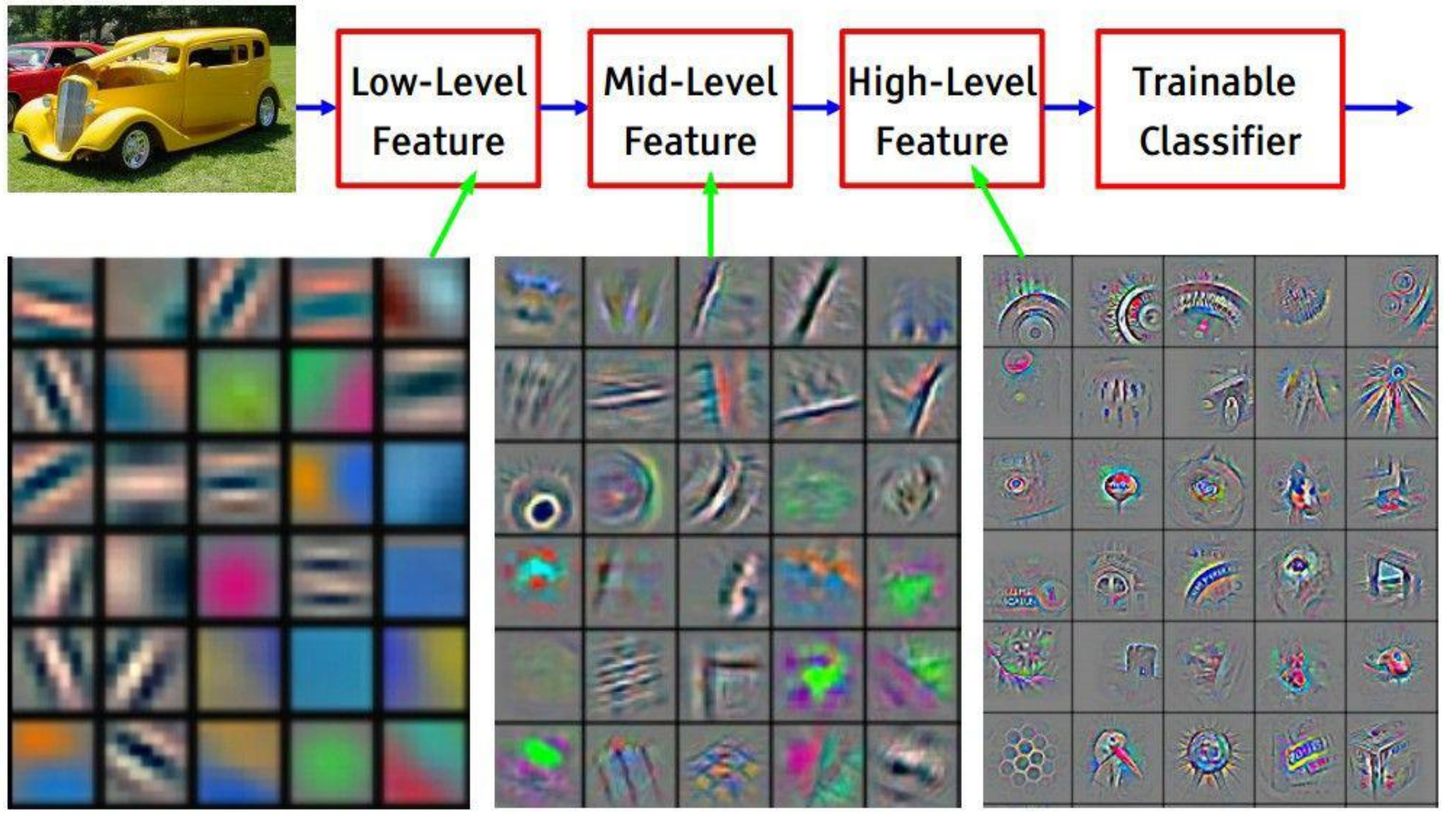
CNN :
Max pooling and others

? sampling


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

Convolution layers

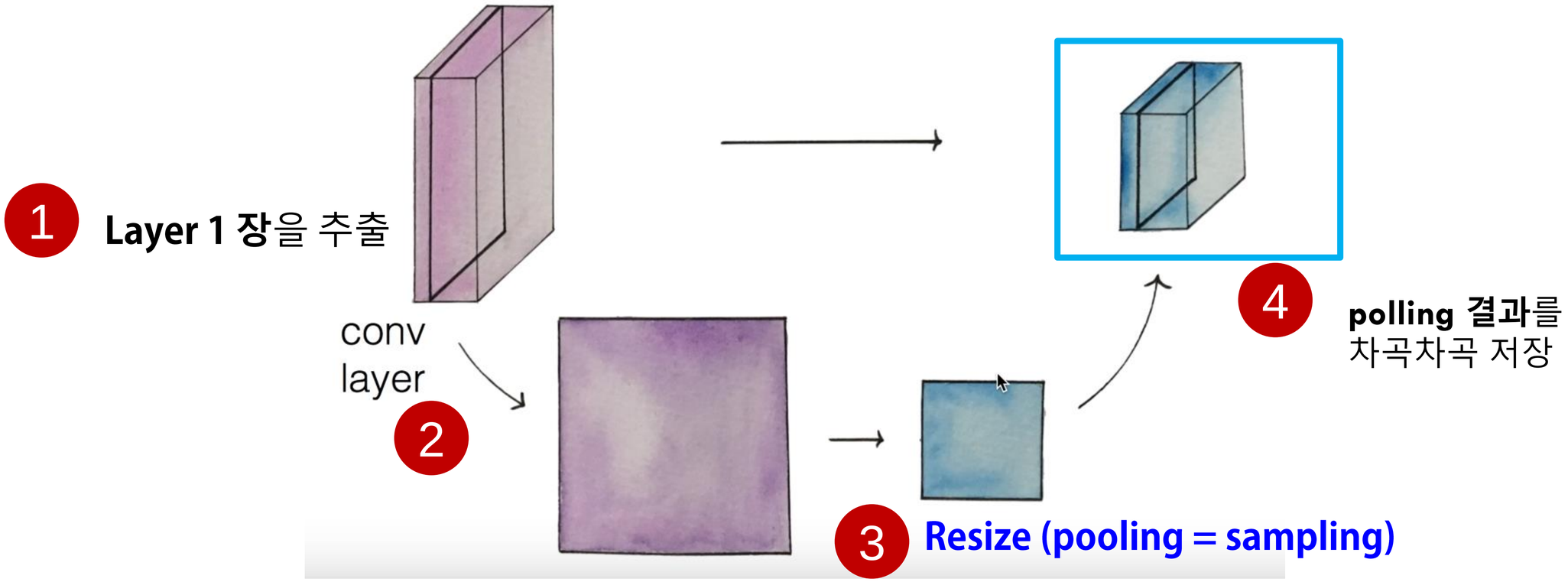
How many weight variables? How to set them?



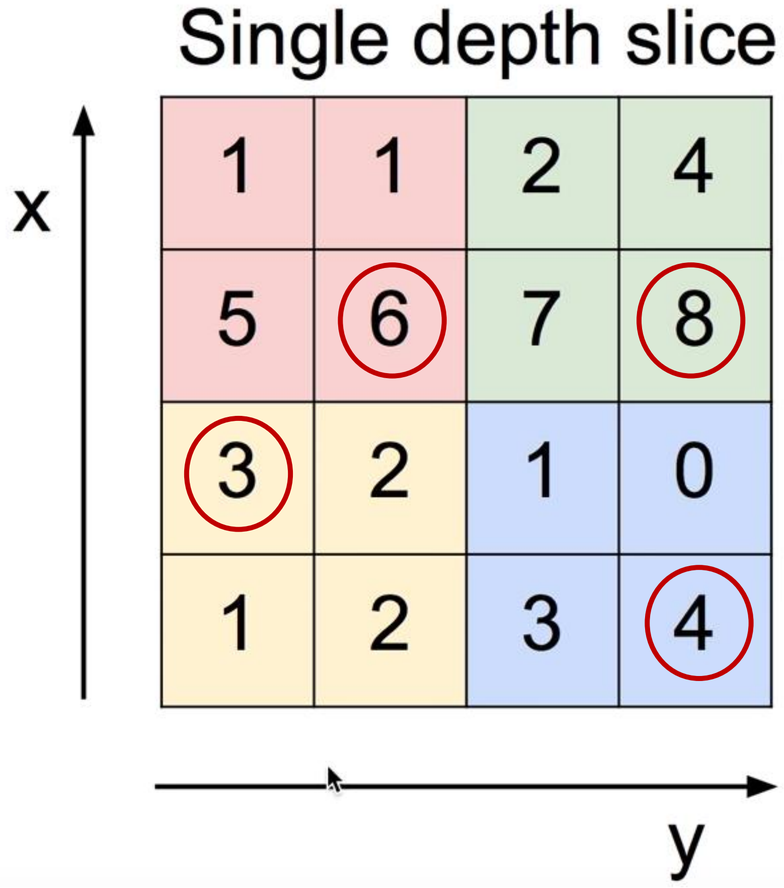
Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]

[From recent Yann LeCun slides]

Polling layer (sampling)



MAX Pooling



max pool with 2x2 filters
and stride 2

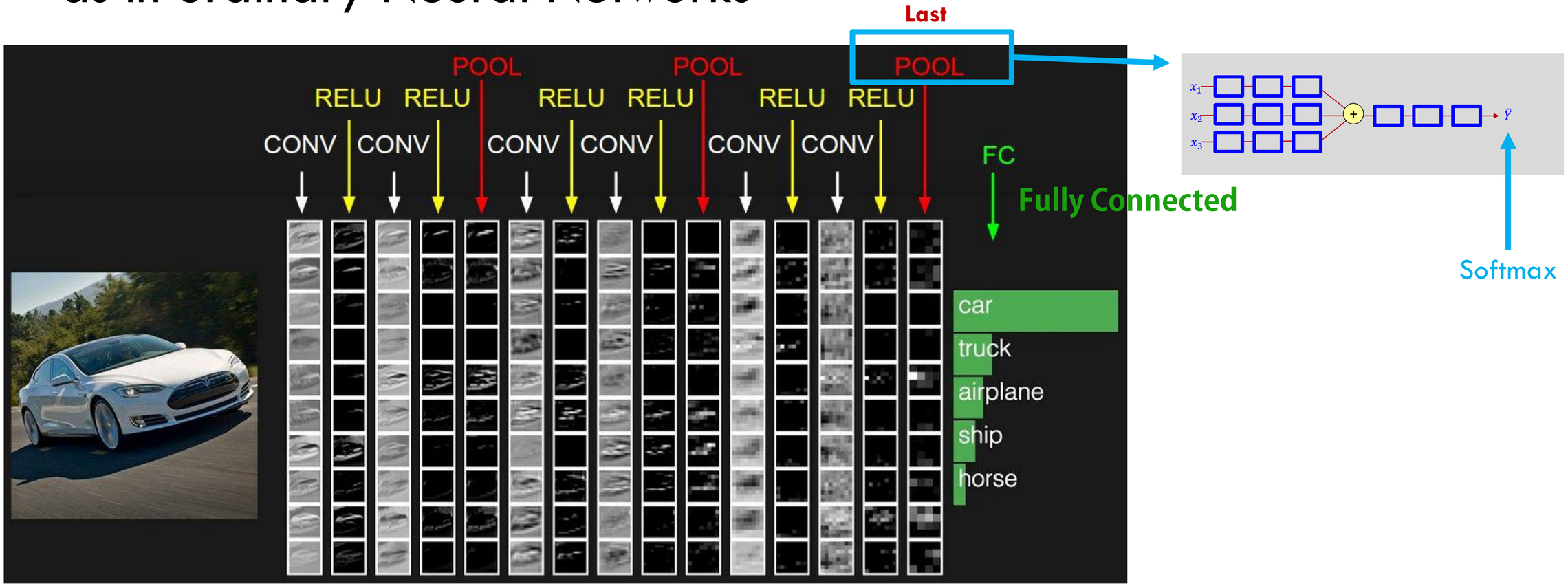
Output

6	8
3	4

2 X 2

Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks



CNN … remember again

e.g.

input : 5x5 , **filter : 3x3** , **stride : 2** , **pad : 1**

→ **what is the output?**

5x5 → 7x7 (expansion)

수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

(7 - 3) / 2 + 1 **= 3**

output : 3x3

Visualization of CNN

- input : 5 x 5 x 3
- stride : 2
- filter : 3 x 3 x 3
- padding size
  = (Filters of size – 1) / 2
  = (3 – 1) / 2
  = **1**

→ 2 x 2 = 4
(수평방향 : 1 + 1) x (수직방향 : 1 + 1) = 4

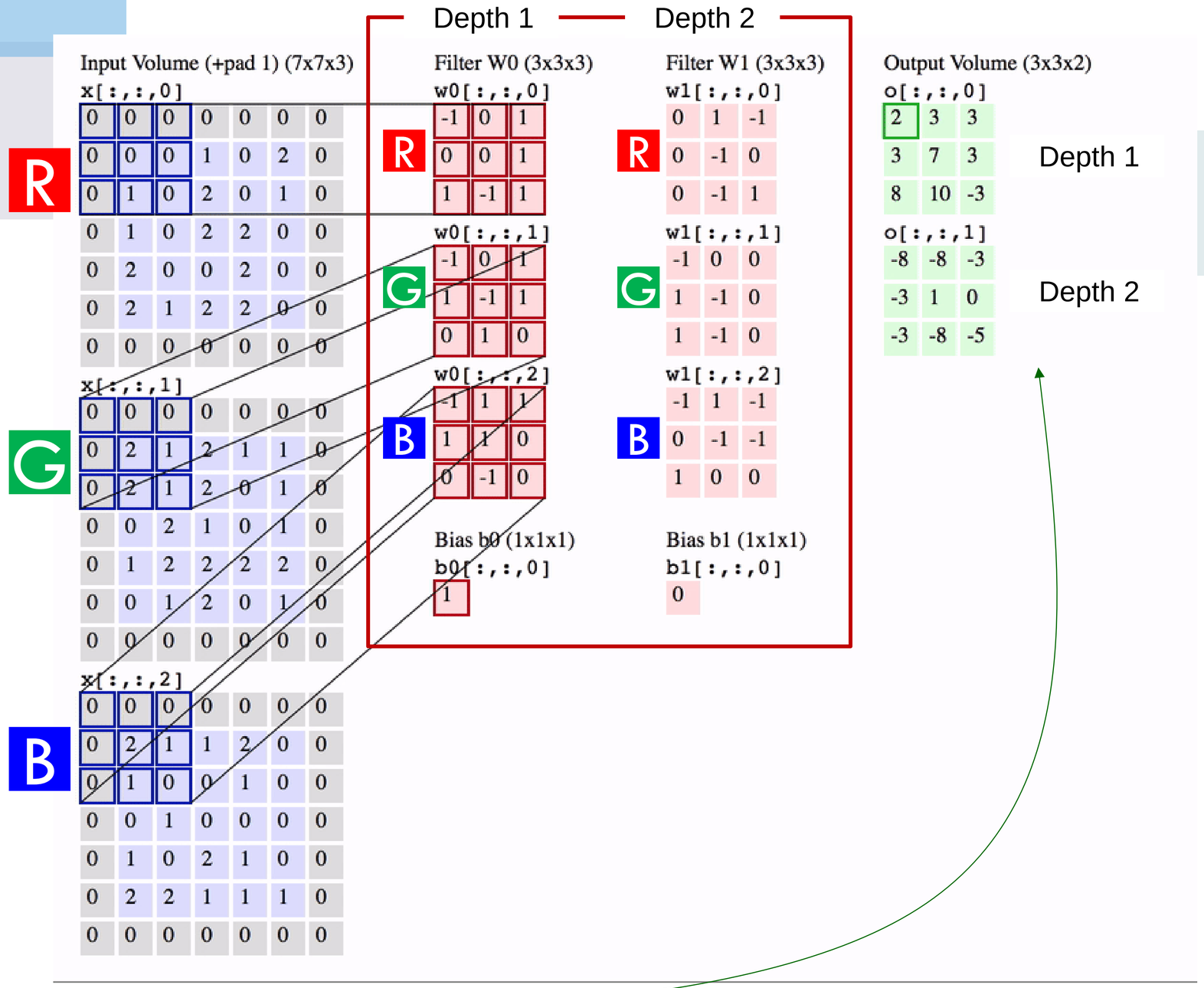
- Input (expansion) :
  수평 방향에서 보면 → 5 + 2 = 7

→ 7 x 7 x 3

- Depth : 2
- Weight (filter) :
  (3 x 3 x 3) x depth
  = (3 x 3 x 3) x 2

- Output : pooling (sampling = output volume)
  filter : 3x3, stride size : 2, depth : 2
  수평 방향에서 보면
  = (N – F) / stride size + 1
  = (7 – 3) / 2 + 1 = 3

→ 3 x 3 x 2



CNN :
CNN case study

Case Study: LeNet-5

[LeCun et al., 1998]

CNN의 고전이라고 부를 수 있는 LeNet-5이다.

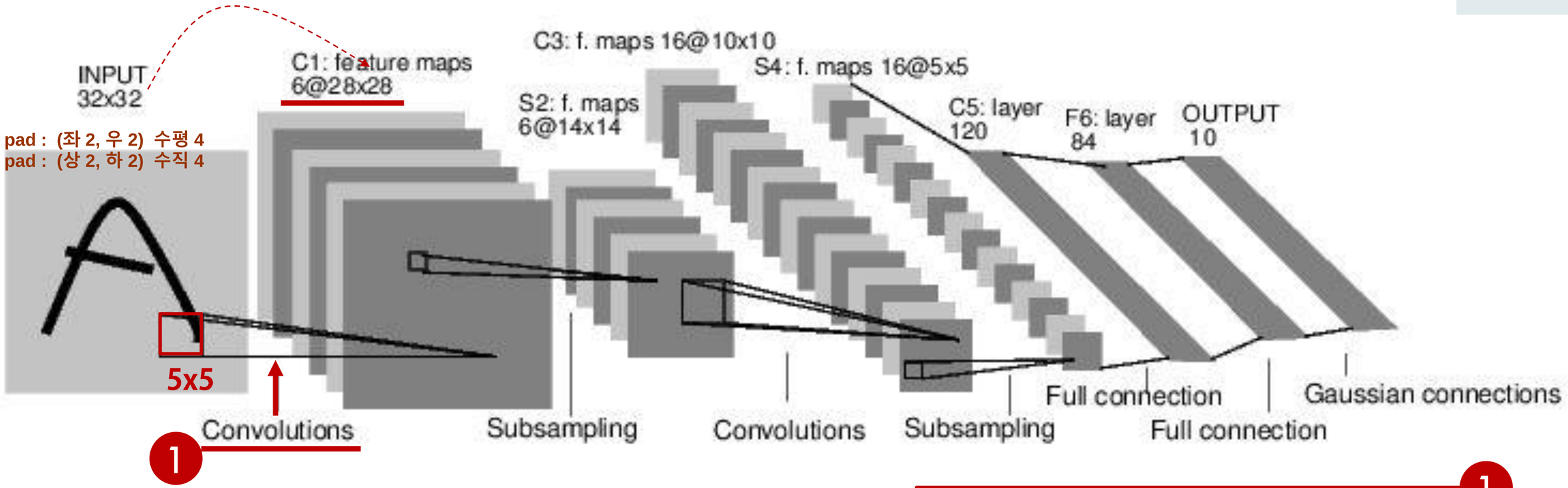
LeCun 교수가 1998년에 만든 모델로 이후에 나오는 모델에 비해 굉장히 단순한 형태이다.

6개의 hidden layer를 사용하고 있다.

1. Input - 크기 32x32x1. 흑백 이미지. (필터 5x5, stride 1)
2. C1 - 크기 28x28x6. feature maps 6개. (필터 2x2, subsampling)
3. S2 - 크기 14x14x6. feature maps 6개. (필터 5x5, stride 1)
4. C3 - 크기 10x10x16. feature maps 16개
5. S4 - 크기 5x5x16. feature maps 16개.
6. C5 - FC(Full Connection)120개
7. F6 - FC(Full Connection) 84개
8. Output - GC(Gaussian Connections) 10개. (최종 분류)

Case Study: LeNet-5

[LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1

1

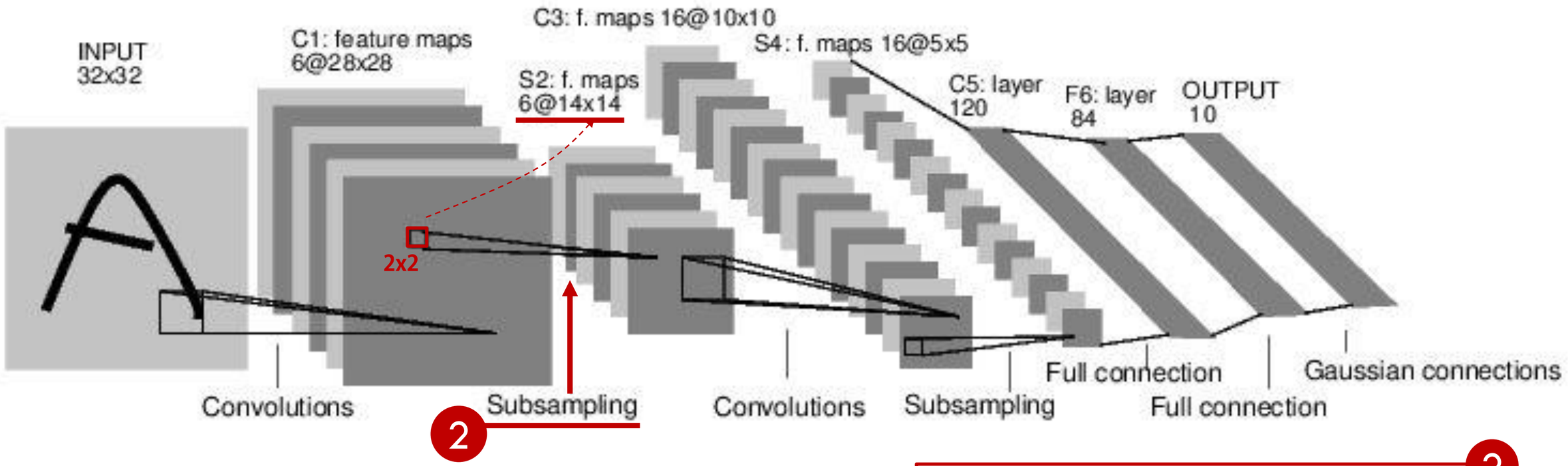
수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

(32 - 5) / 1 + 1 **= 28**

Case Study: LeNet-5

[LeCun et al., 1998]



Subsampling (Pooling) layers were 2x2 applied at stride 2

2

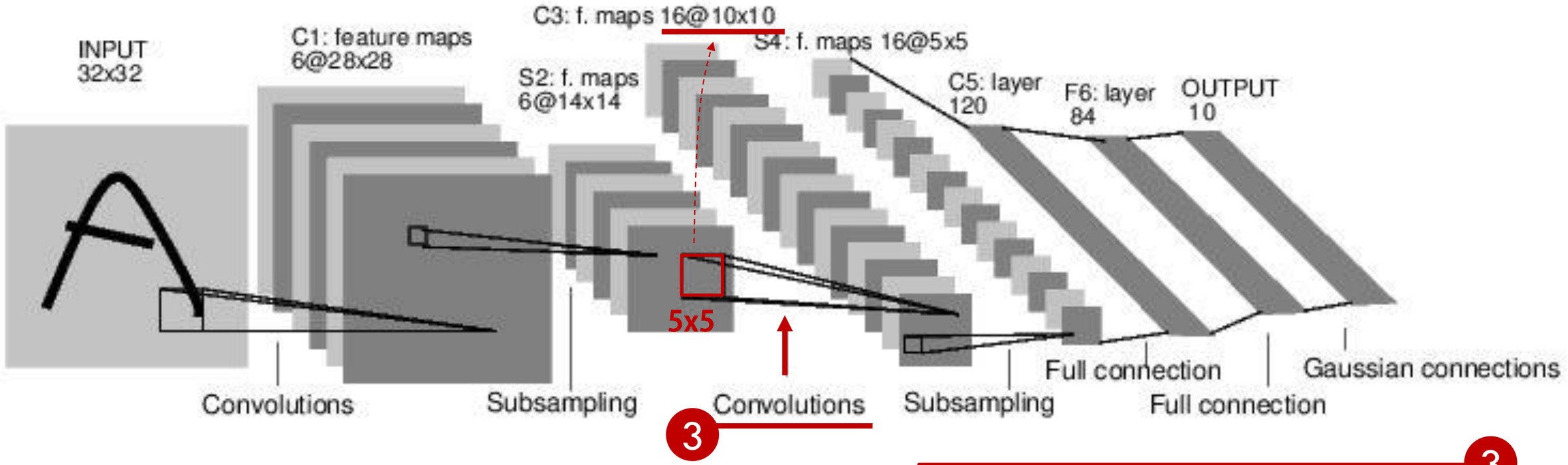
수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

(28 - 2) / 2 + 1 **= 14**

Case Study: LeNet-5

[LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1

3

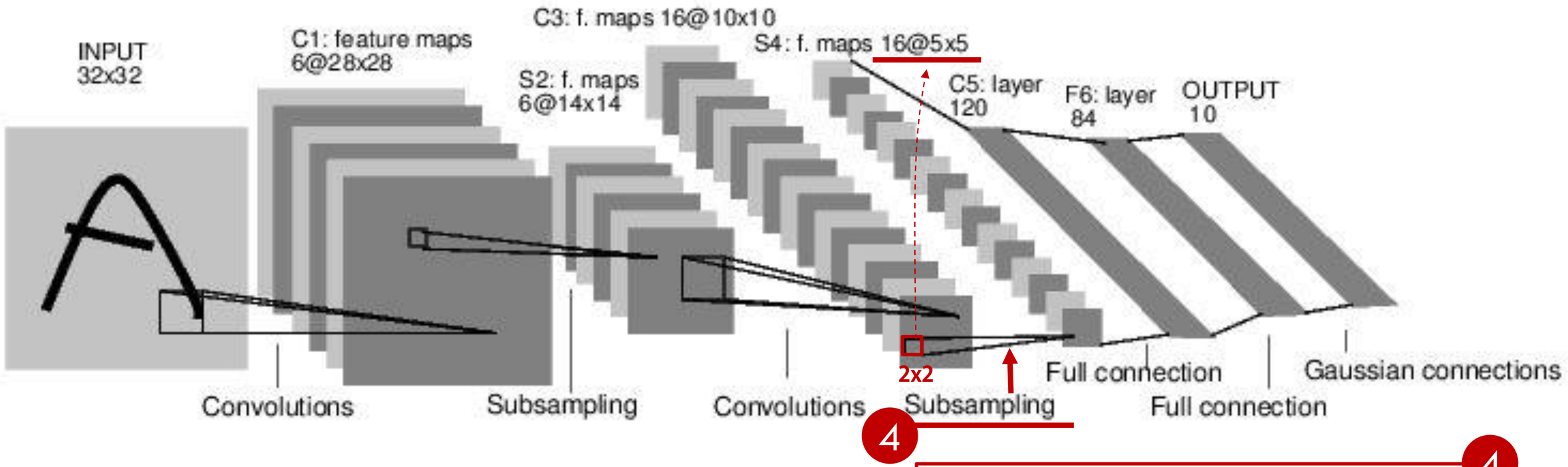
수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

(14 - 5) / 1 + 1 **= 10**

Case Study: LeNet-5

[LeCun et al., 1998]



4

Subsampling (Pooling) layers were 2x2 applied at stride 2

4

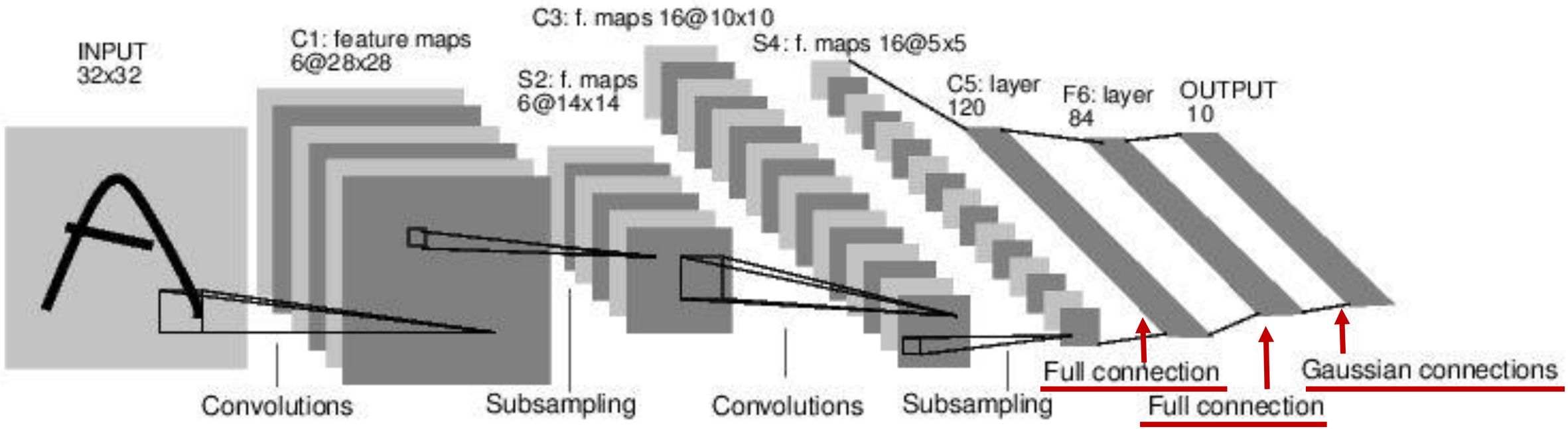
수평 또는 수직의 한 방향에서 보면…

Output size → **(N – F) / stride size** + 1

(10 - 2) / 2 + 1 **= 5**

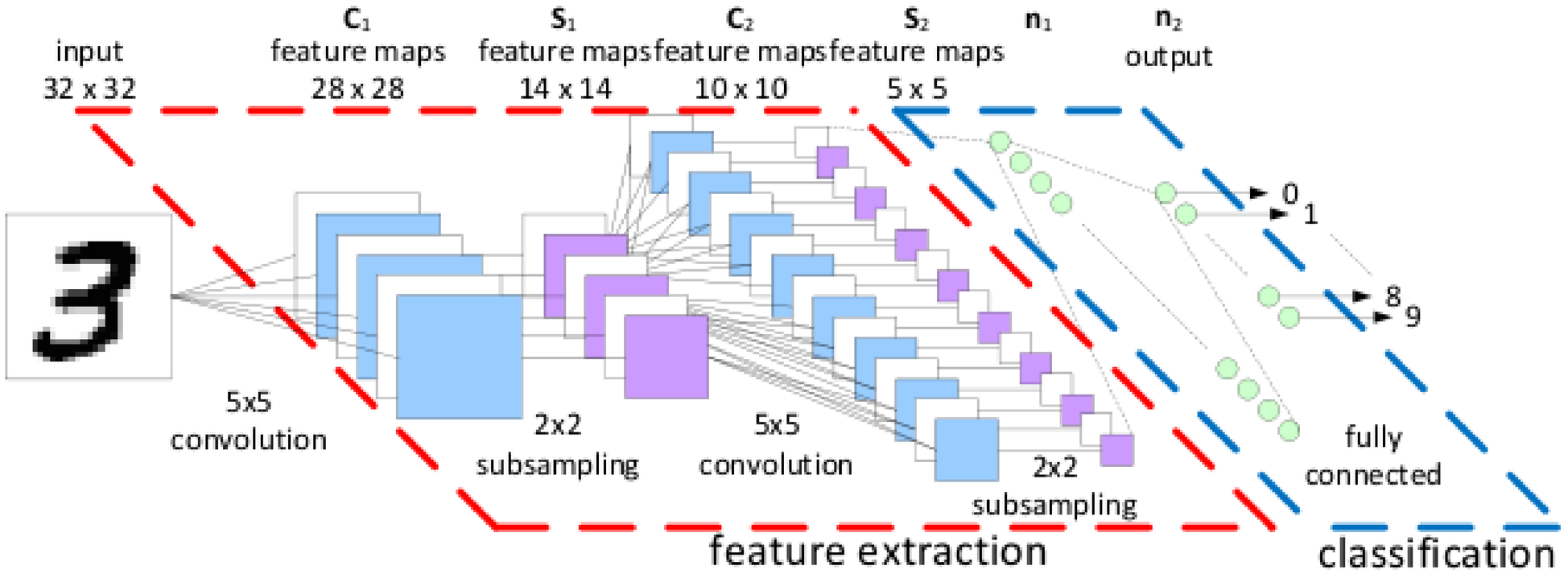
Case Study: LeNet-5

[LeCun et al., 1998]



Case Study: LeNet-5

[LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-FC]

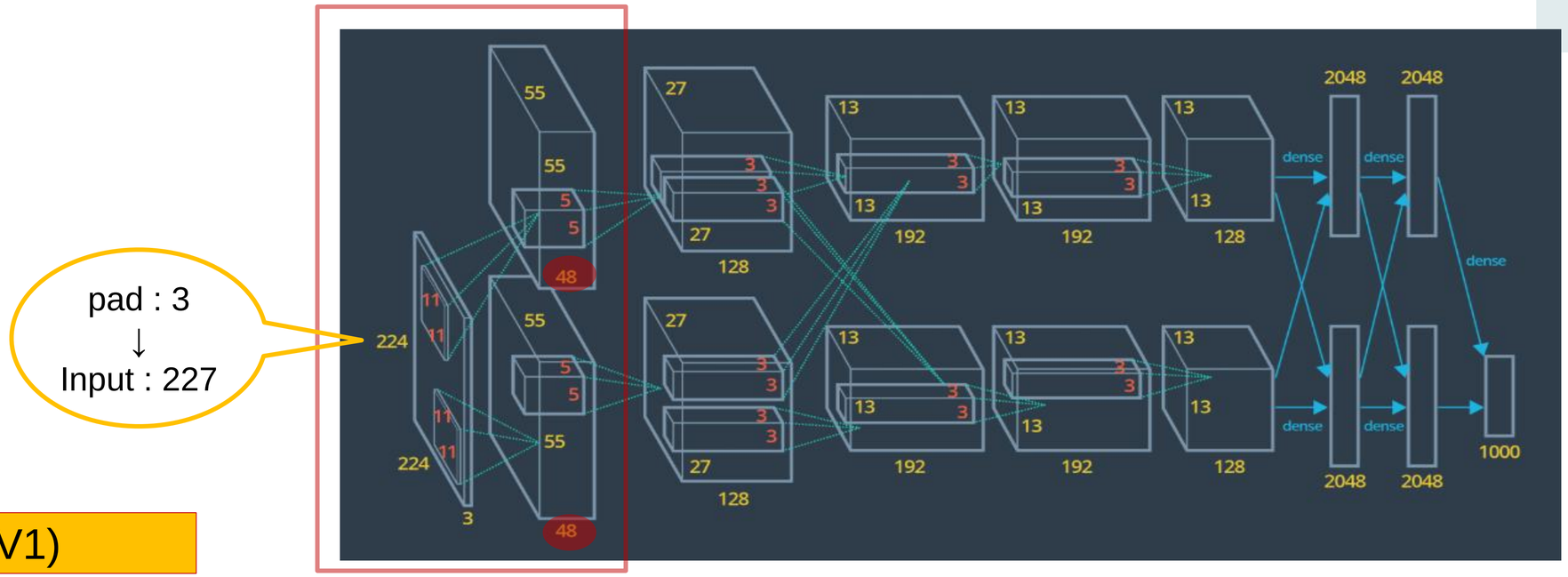
Case Study: AlexNet

[Krizhevsky et al. 2012]

- AlexNet은 딥러닝 혁명을 이끈 초기 딥러닝 모델 중 하나로 컨볼루션 신경망 구조를 갖는다.
  이 모델은 대규모 물체인식 콘테스트 (ImageNet LSVRC-2010)에서 2012 년에 우승하였다.
  LSVRC (Large-Scale Visual Recognition Competition) 대회는 120 만장의 영상을 1000 개의 클래스로 구분하는 대회이다. AlexNet 은 top-5 에러율(모델이 예측한 5 개의 클래스 중에 정답이 포함되지 않은 경우) 17%를 달성하여 기존의 기록을 갱신하였다. 이는 딥러닝으로 기존의 컴퓨터 비전 기법들을 사용한 알고리즘들의 성능을 월등히 능가하였다는 점에서 후속 딥러닝 연구를 촉발한 연구 결과이다.
- AlexNet 은 총 65 만개의 뉴런과 6 천만개의 파라미터로 이루어져 있고, 5 개의 컨볼루션층과 3 개의 완전연결층으로 구성되었으며 최상위층은 1000 개의 소프트맥스 뉴런으로 구성되었다. 학습을 빠르게하기 위해서 ReLU 뉴런을 사용하였으며 GPU 를 사용하여 컨볼루션 연산을 수행하였다.

Case Study: AlexNet

[Krizhevsky et al. 2012]



Layer 1 (CONV1)

- Layer 1 Output :
  55 x 55 x 96(48 x 2)

수평 또는 수직의 한 방향에서 보면…

Output volume → **(N – F) / stride size** + 1

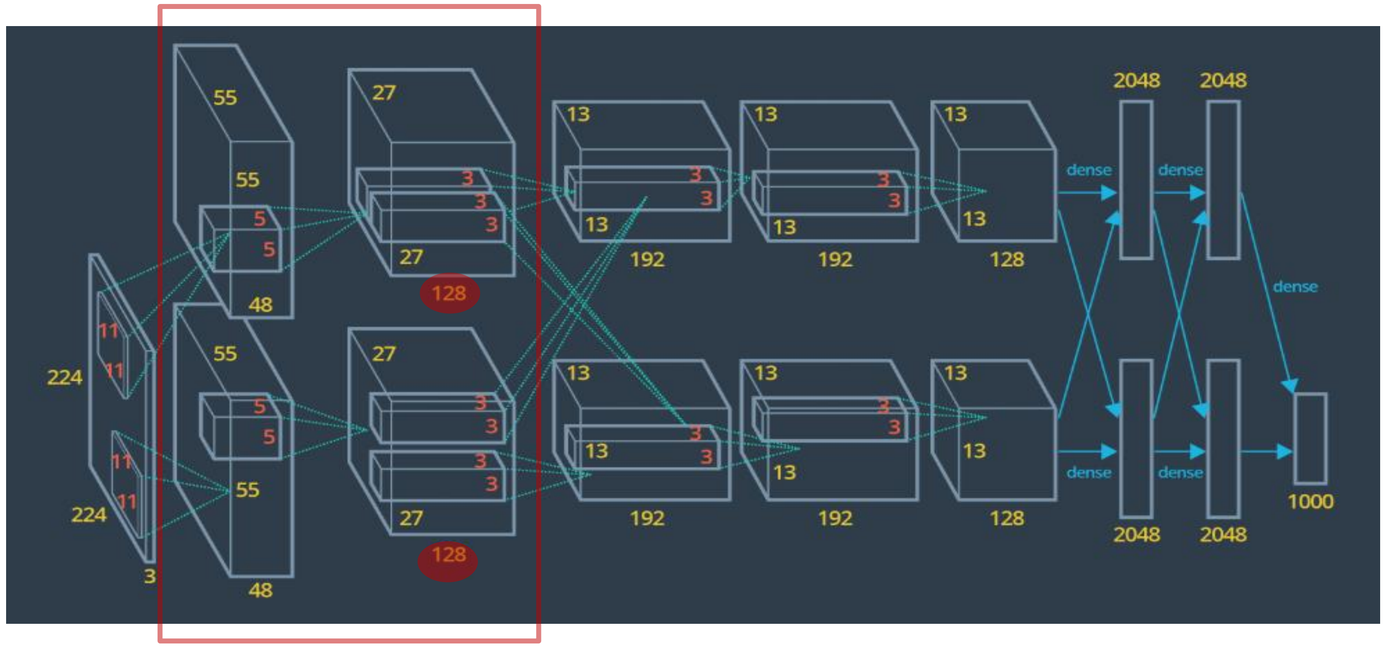
(227 - 11) / 4 + 1 **= 55**

Case Study: AlexNet

[Krizhevsky et al. 2012]

Layer 2 (POOL1)

- Layer 2 Output :
  27 x 27 x 256(128 x 2)

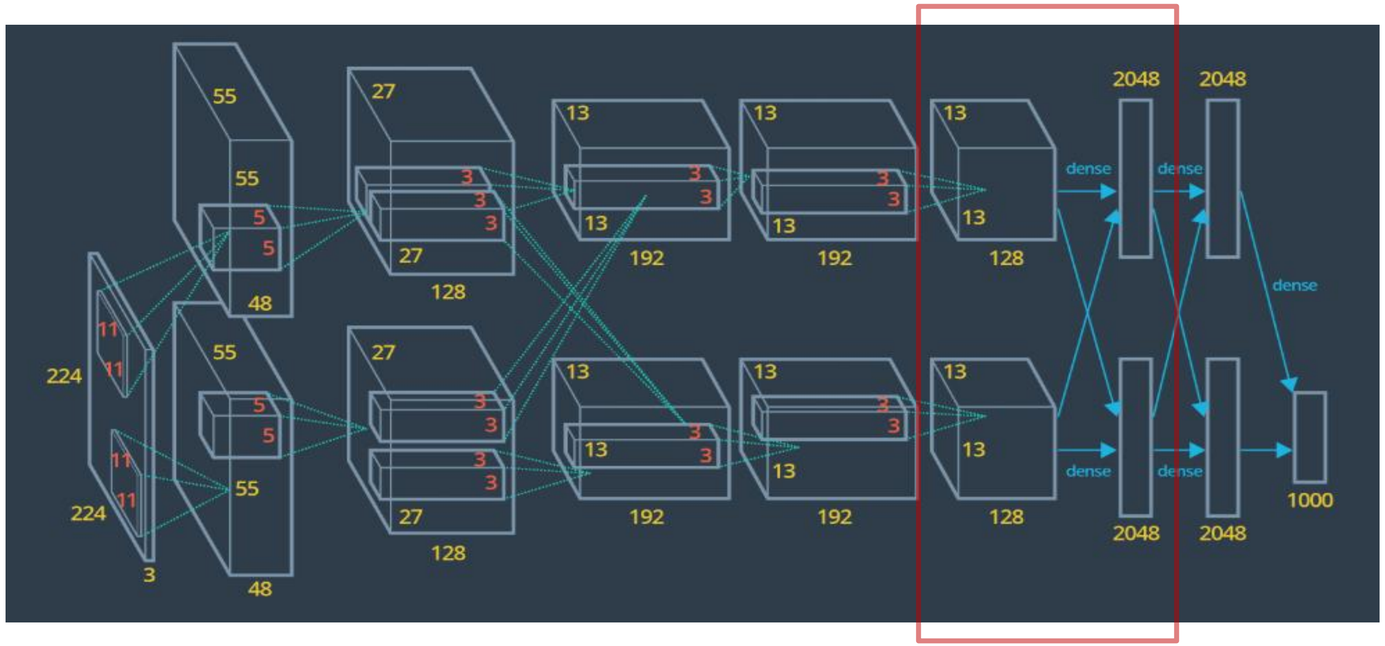


Case Study: AlexNet

[Krizhevsky et al. 2012]

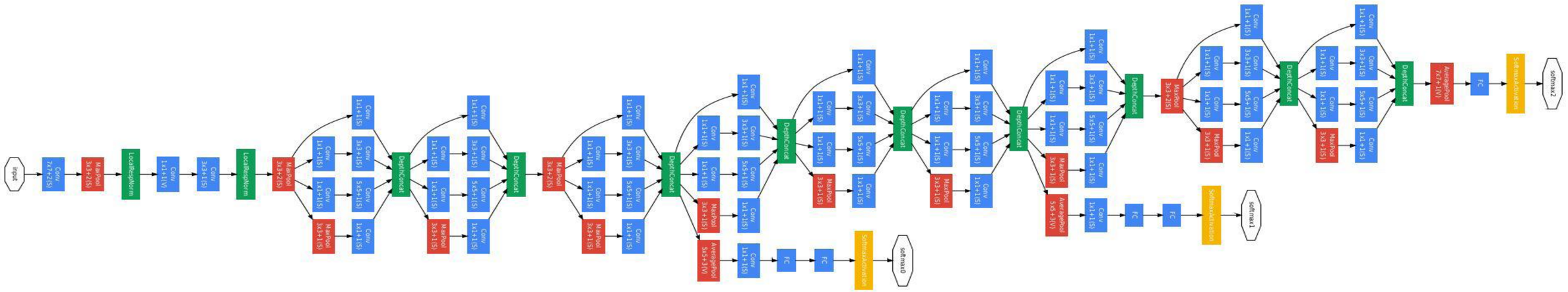
Layer 6 (FC)

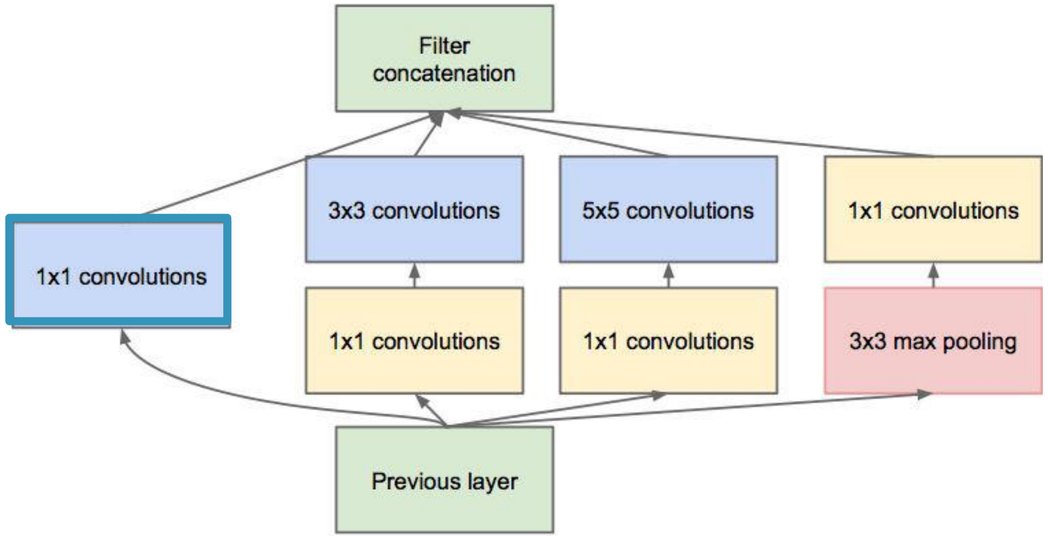
- Input Image size : 13 x 13 x 128 x 2
- Layer 6 Output :
  1 x 2048 x 2



Case Study: GoogLeNet

[Szegedy et al., 2014]



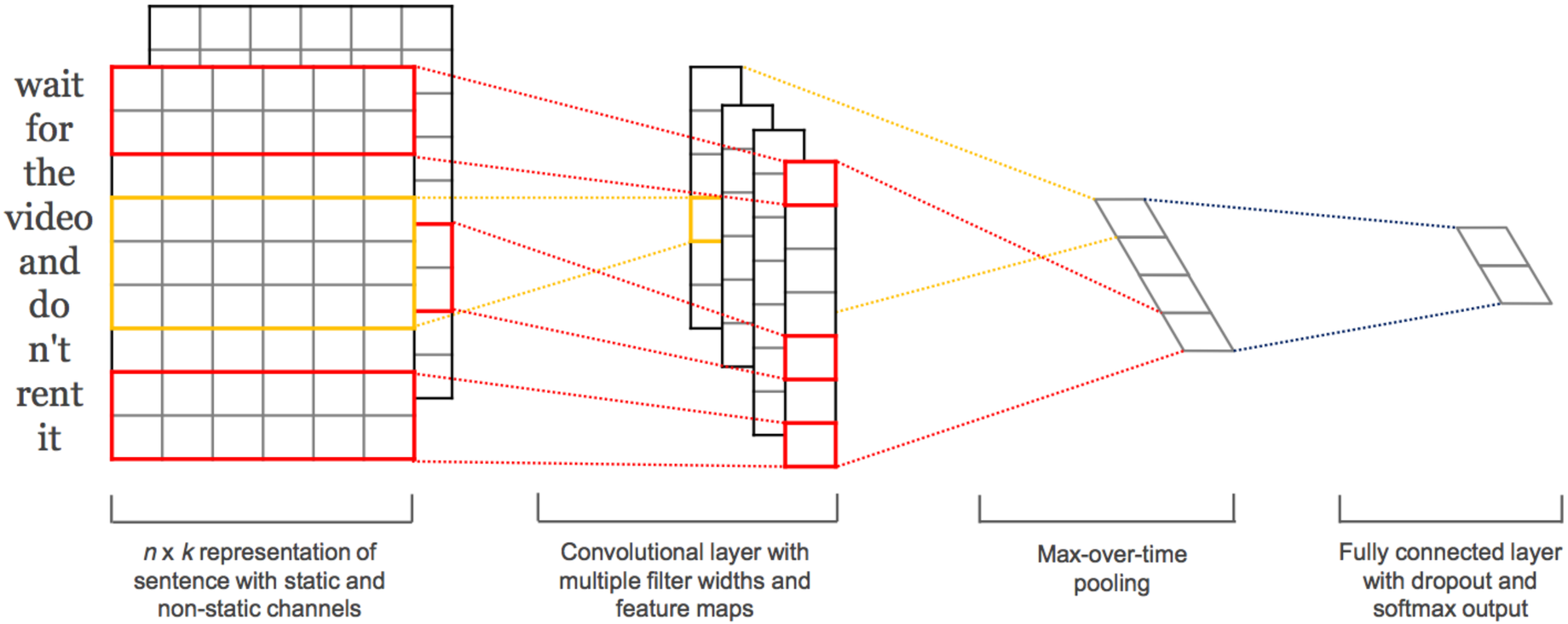


Inception module

ILSVRC 2014 winner (6.7% top 5 error)

Case Study: Convolutional Neural Networks for Sentence Classification

[Yoon Kim, 2014]



Case Study Bonus: DeepMind's AlphaGo