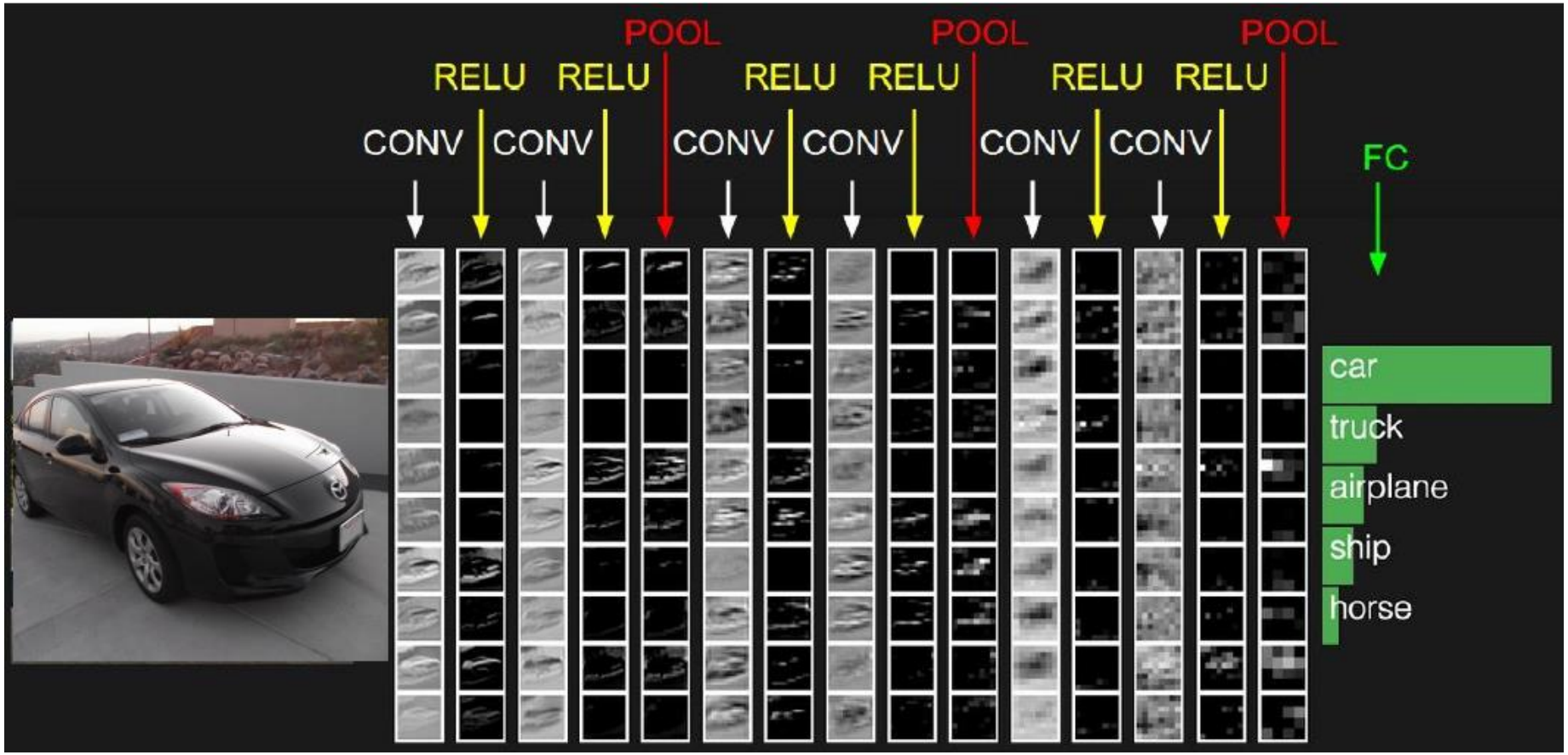
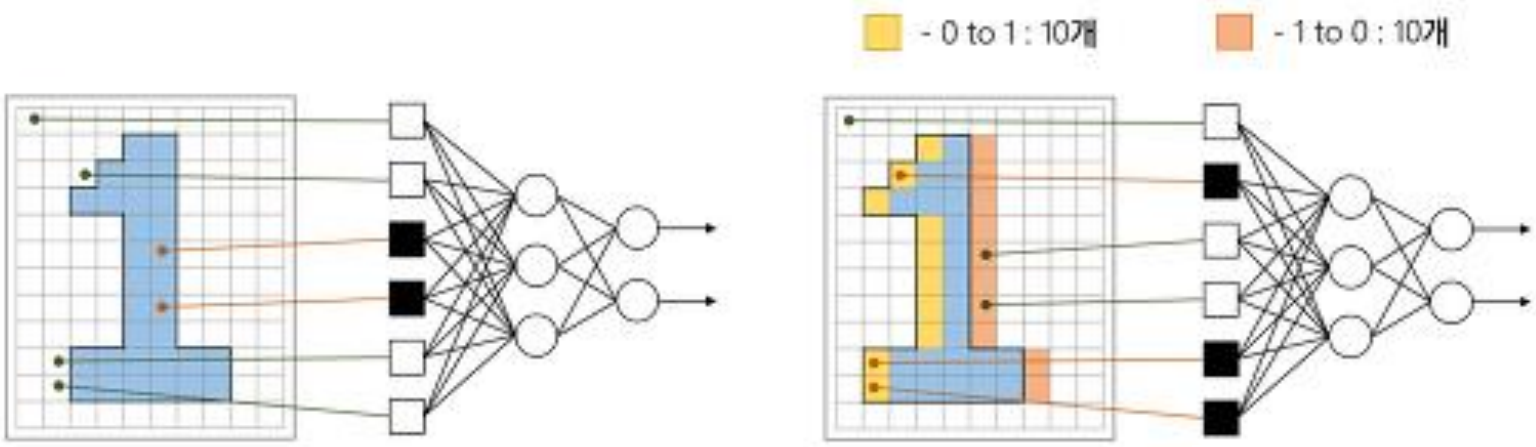
CNN 알고리즘

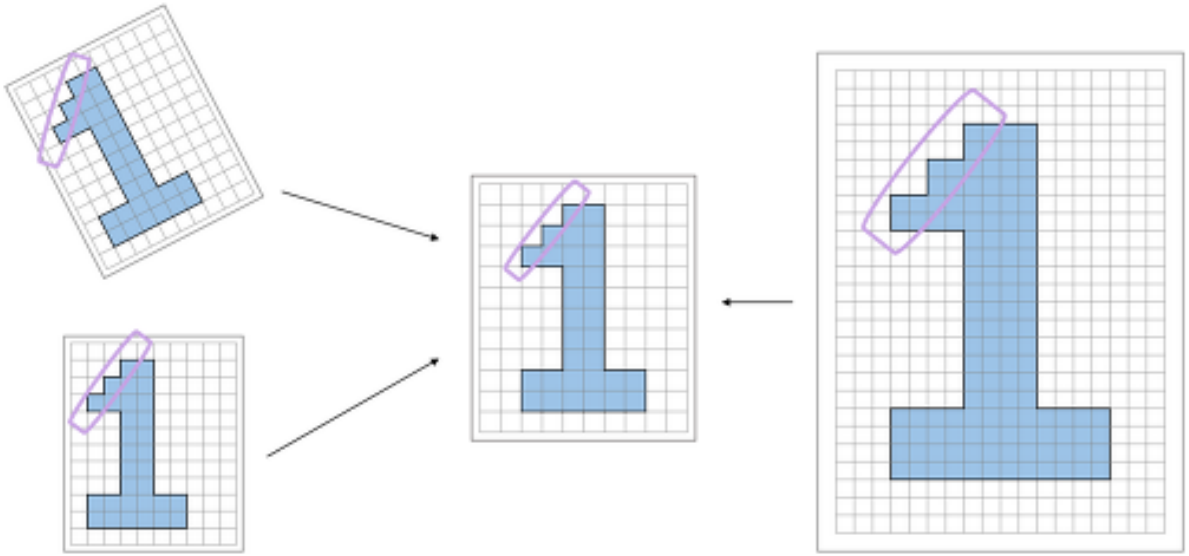

POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse

박경규

MLP의 문제점



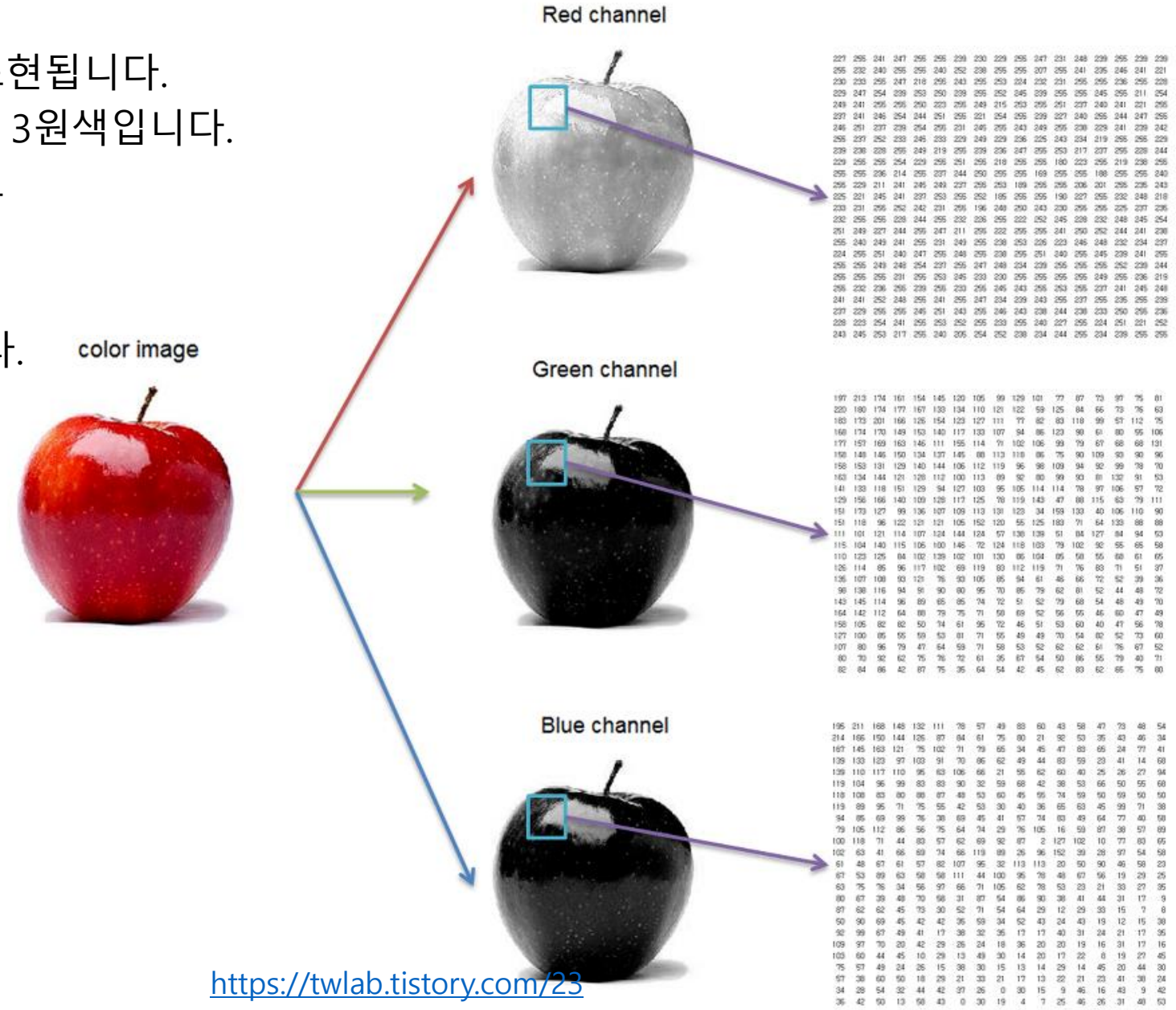
한 칸씩만 움직였는데
변화하는 인풋값이 20개



https://gomguard.tistory.com/188?category=712467

이미지 데이터

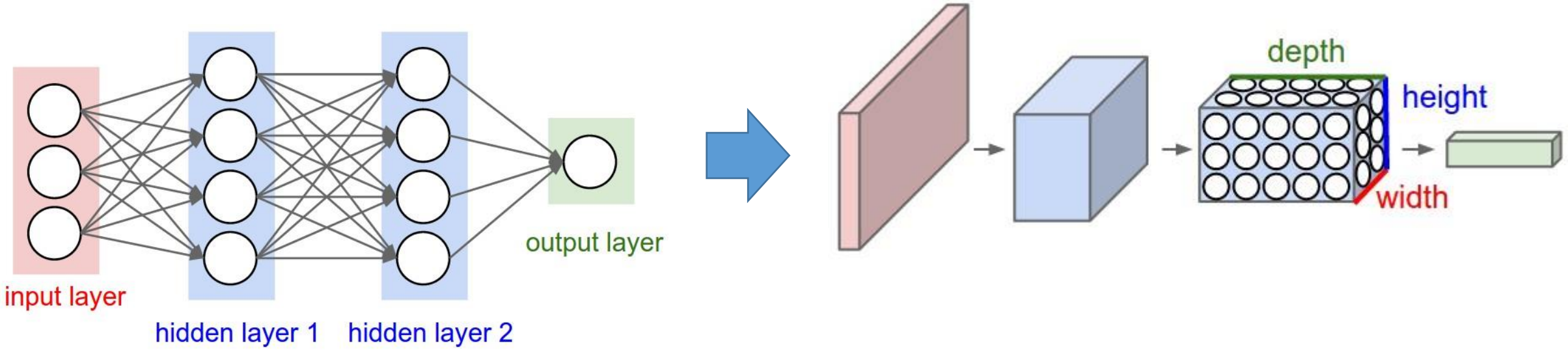
- 컬러 이미지는 3개의 채널로 표현됩니다.
- 3개의 채널은 Red, Green, Blue 3원색입니다.
- 각 채널은 0~255사이의 값으로
  빨강의 정도,
  녹색의 정도,
  파랑의 정도를 각각 나타냅니다.



https://twlab.tistory.com/23

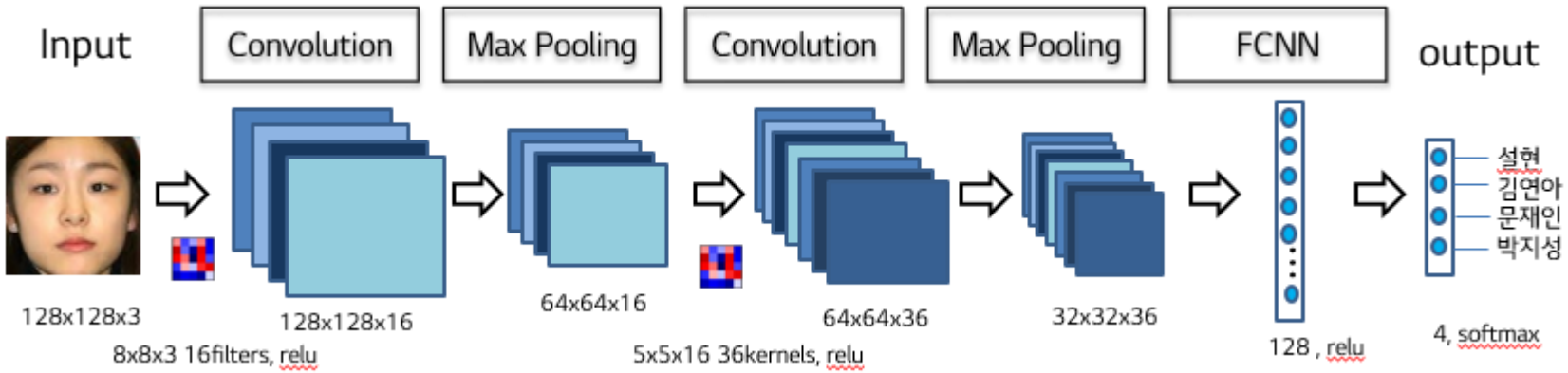
이미지 데이터 처리

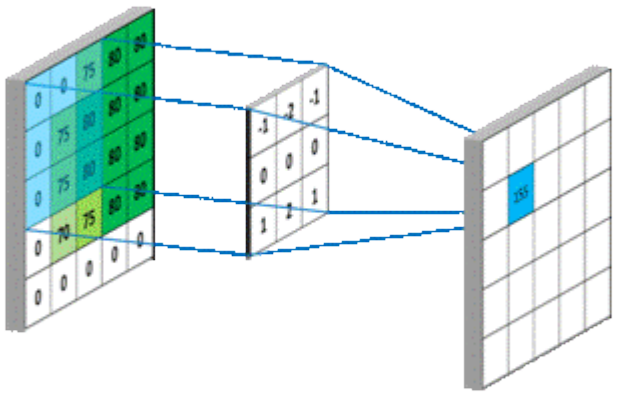
- 200x200x3 크기 이미지는 첫 번째 hidden layer의 1개의 뉴런에 대해 200x200x3=120,000개의가중치를 필요로 하여, 일반 신경망은 이미지를 다루기에 적합하지 않습니다.
- ConvNet은 입력이 이미지로 이뤄져 있다는 특징을 살려 좀 더 합리적인 방향으로 아키텍처를 구성하였습니다.
- CNN과 MLP의 가장 큰 차이점은 이미지 Feature 추출방법입니다. MLP는 이미지의 픽셀 값을 Input으로 사용하는 것이고, CNN은 이미지의 Region Feature를 Convolution Layer와 Pooling Layer를 이용해 추출하고 그 Feature를 MLP의 Input으로 사용하는 것입니다.
- CNN이 Computer Vision에서 성능이 좋은 이유는 Region Feature를 추출할 수 있기 때문입니다.



CNN(Convolutional Neural Network)

사람의 시각 피질 메커니즘에 영감을 받아 설계된 이미지, 영상등을 인식하는 신경망 모델
Convolution Layer(합성곱층)에서는 각 filter가 입력 이미지의 픽셀 전체를 차례로 훑고 지나가며
linear combination을 진행하고 Feature Map을 구성합니다.



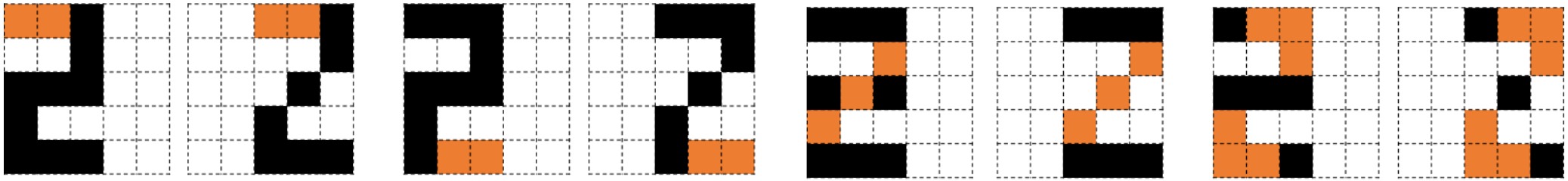


출처 : https://wonwooddo.tistory.com/47

CNN 모델 개념

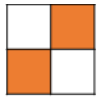
CNN은 뇌가 사물을 구별하듯 생김새 정보로 사물을 학습하고 구별해 낸다.

■ 숫자 2에서 공통적으로 얻을 수 있는 생김새 정보

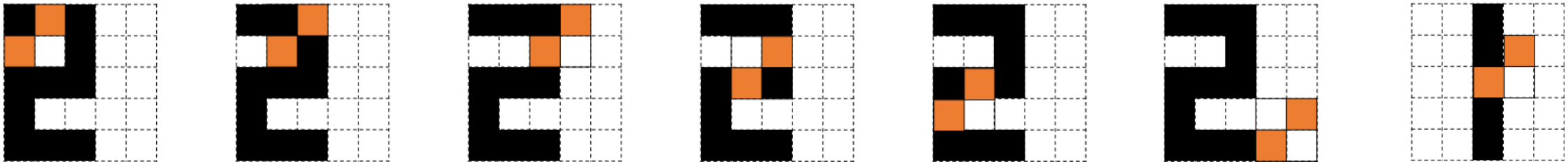


머리 꼬리 이음새 모서리

■ CNN은 어떻게 특징을 찾아 내는가?



필터(커널)



대각선 필터는 숫자 2로부터 두 곳의 대각선 특징을 감지하지만, 숫자 1에서는 대각선 특징을 발견하지 못한다.

CNN 모델 개념

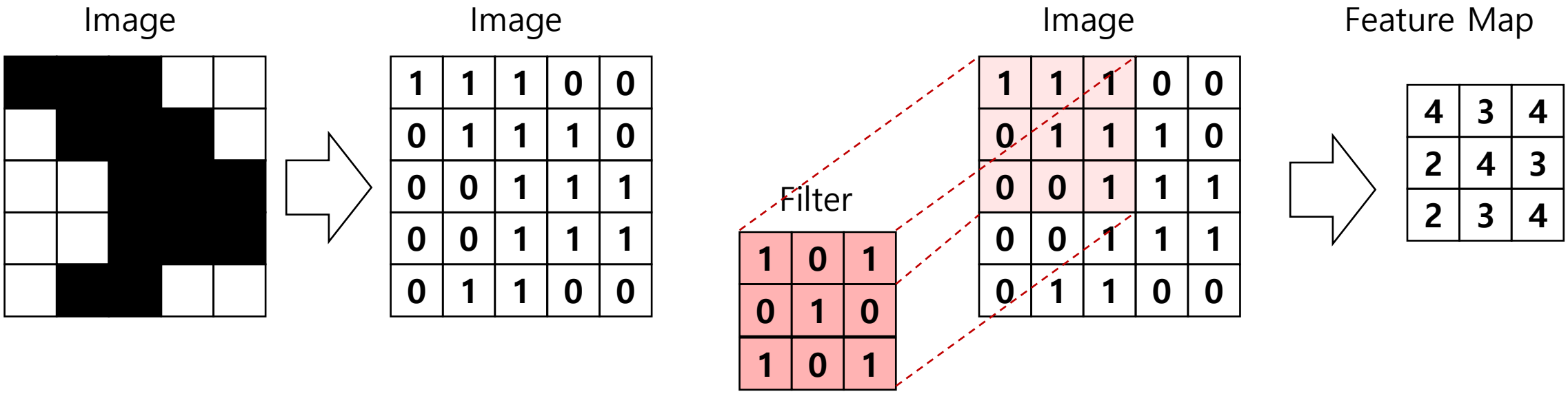
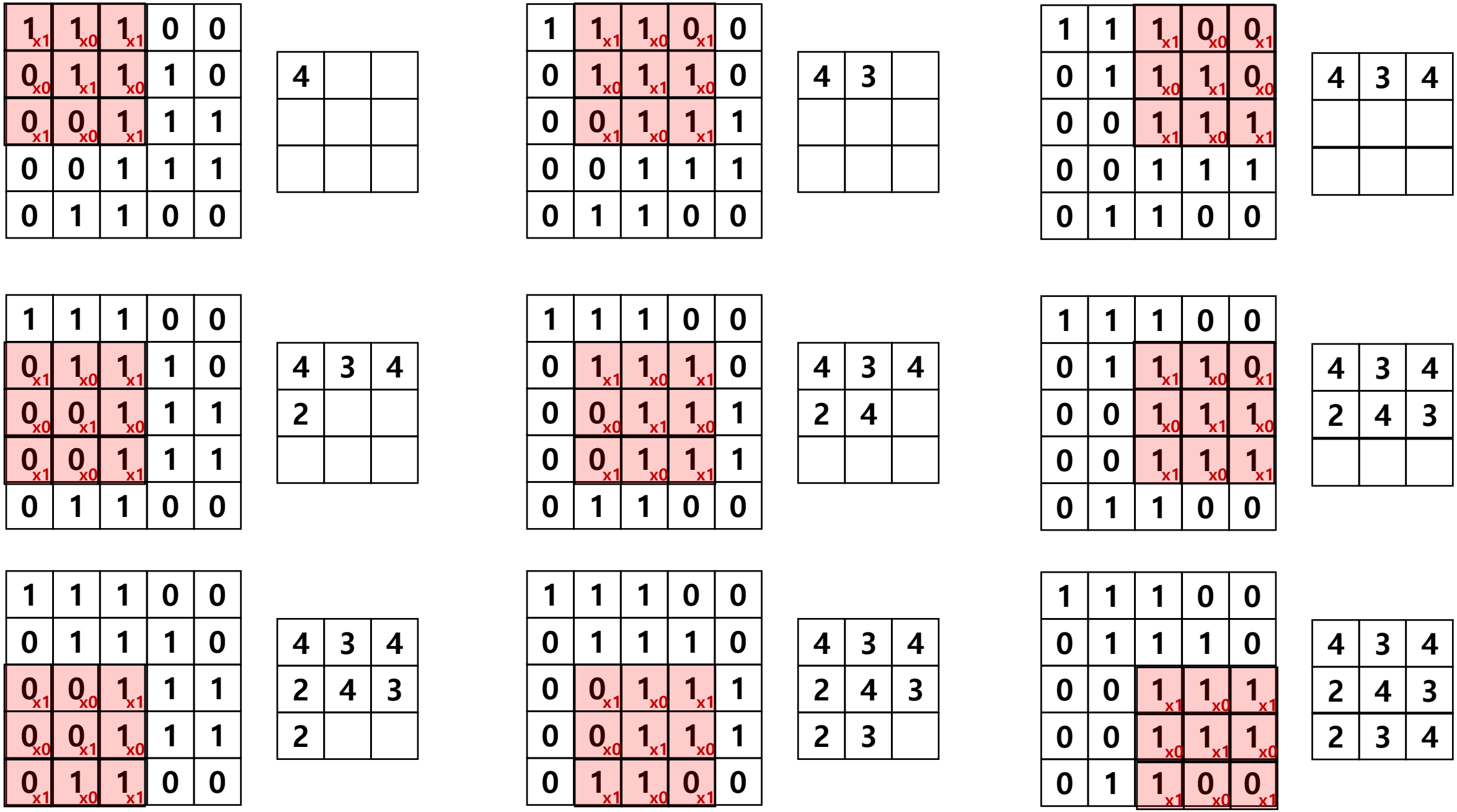


Image
Image
Image
Feature Map
Filter
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
1 0 1
0 1 0
1 0 1
4 3 4
2 4 3
2 3 4


1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2 4
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2 4 3
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2 4 3
2
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2 4 3
2 3
1 1 1 0 0
0 1 1 1 0
0 0 1 1 1
0 0 1 1 1
0 1 1 0 0
4 3 4
2 4 3
2 3 4

Padding

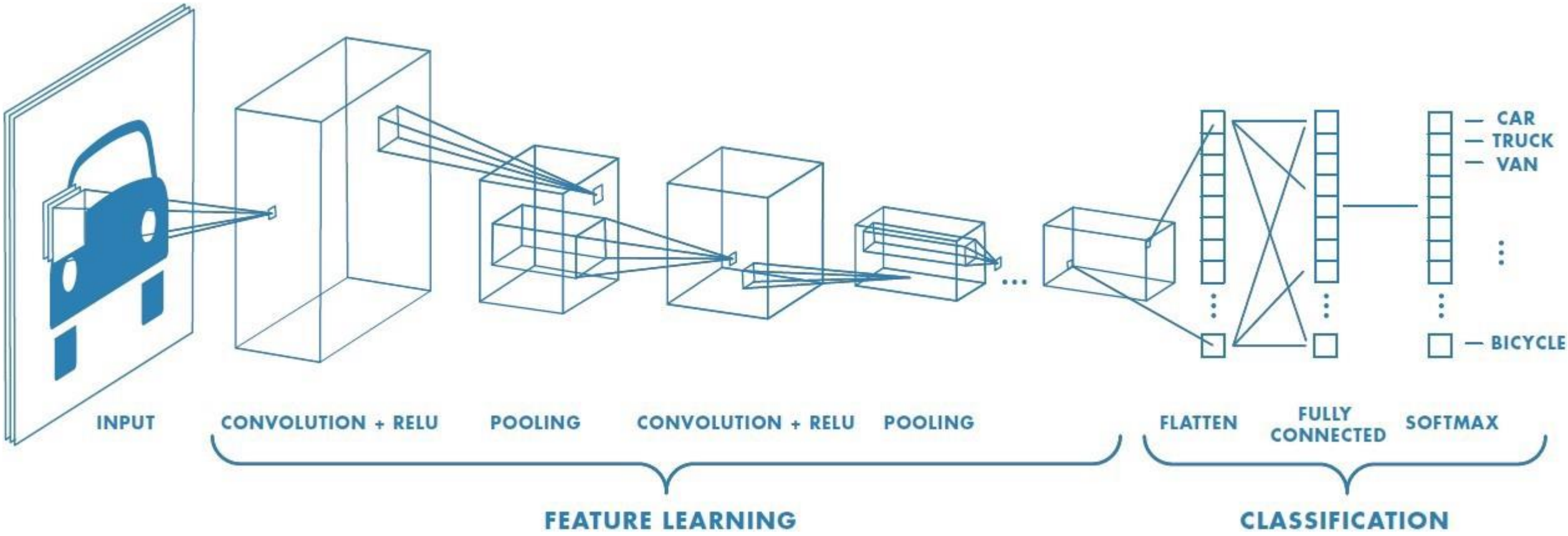
패딩(padding)을 1만큼 적용한 이미지 데이터 행렬

0	0	0	0	0	0	0
0	1	1	1	0	0	0
0	0	1	1	1	0	0
0	0	0	1	1	1	0
0	0	0	1	1	1	0
0	0	1	1	0	0	0
0	0	0	0	0	0	0

Pooling

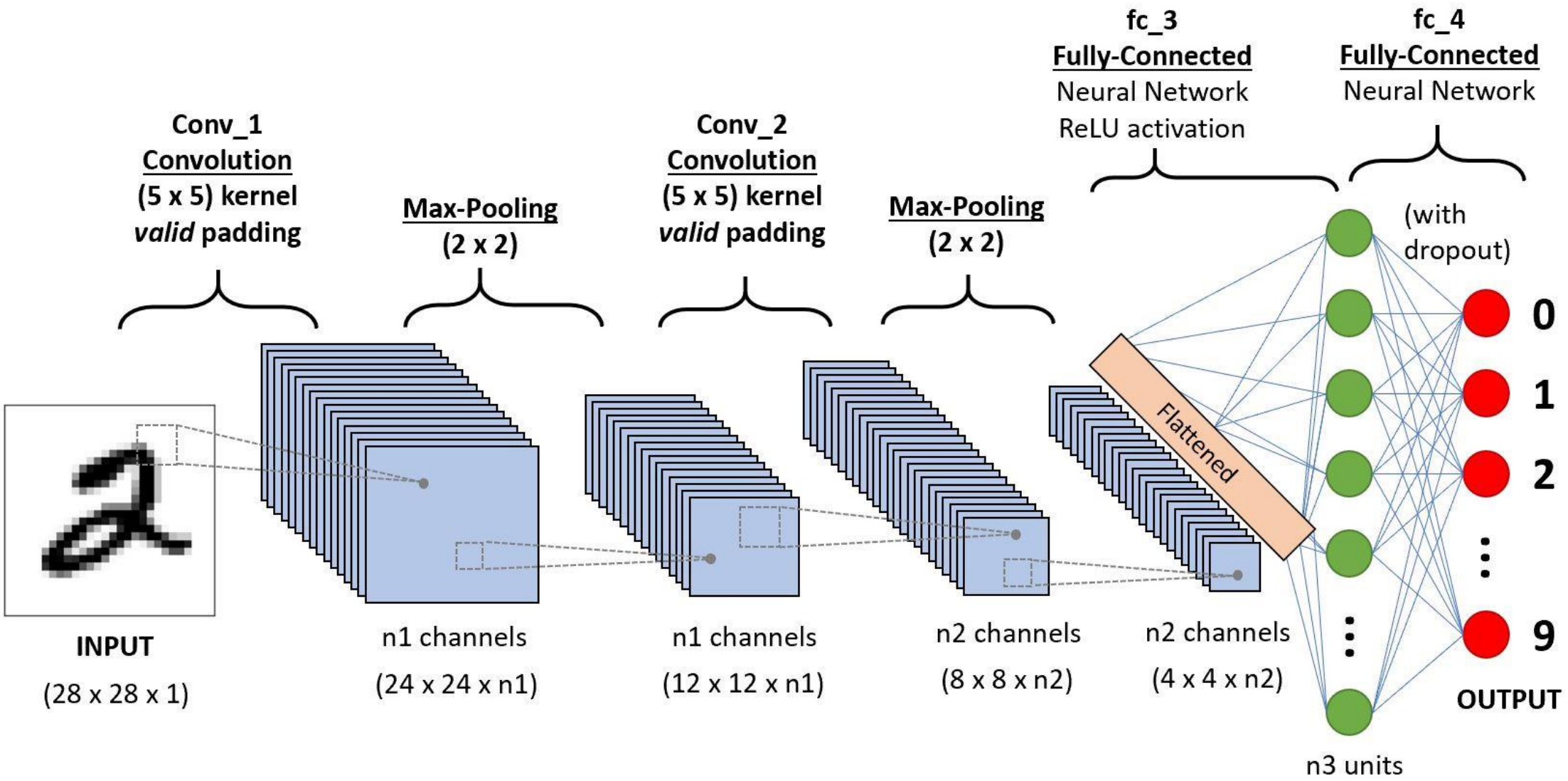
1	4	**8**	3
6	**9**	2	1
11	13	6	7
8	**19**	**8**	2

CNN



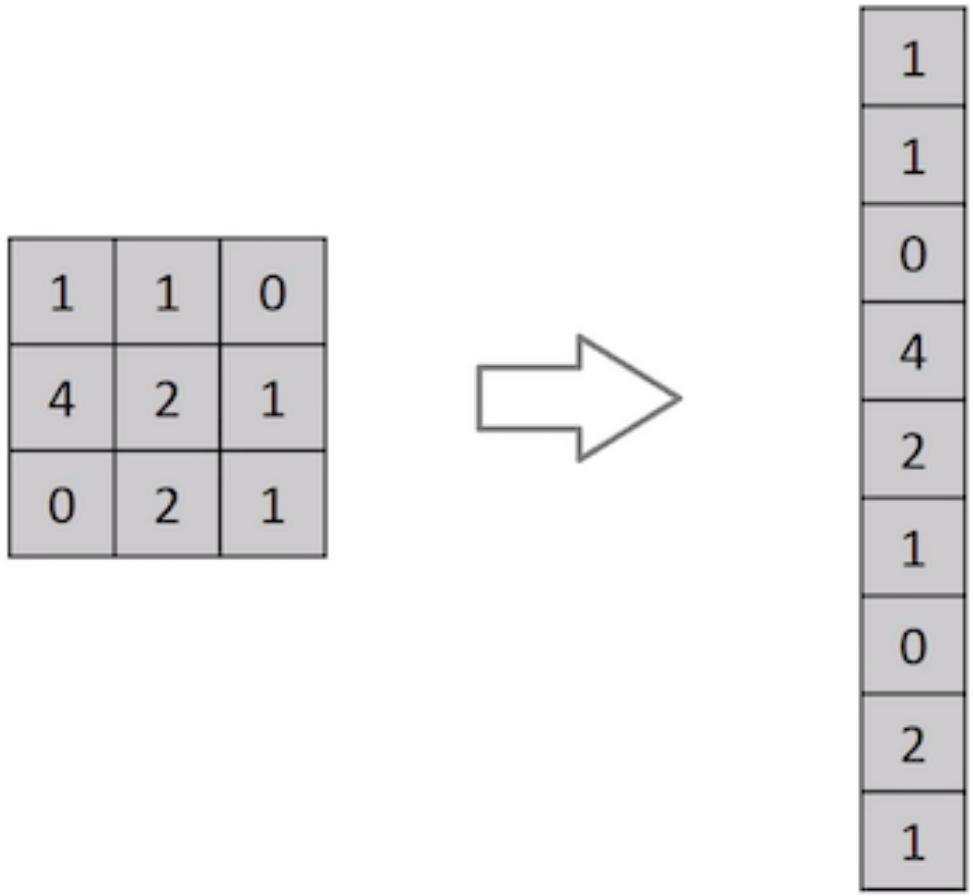
https://bit.ly/2RVDoFI

CNN



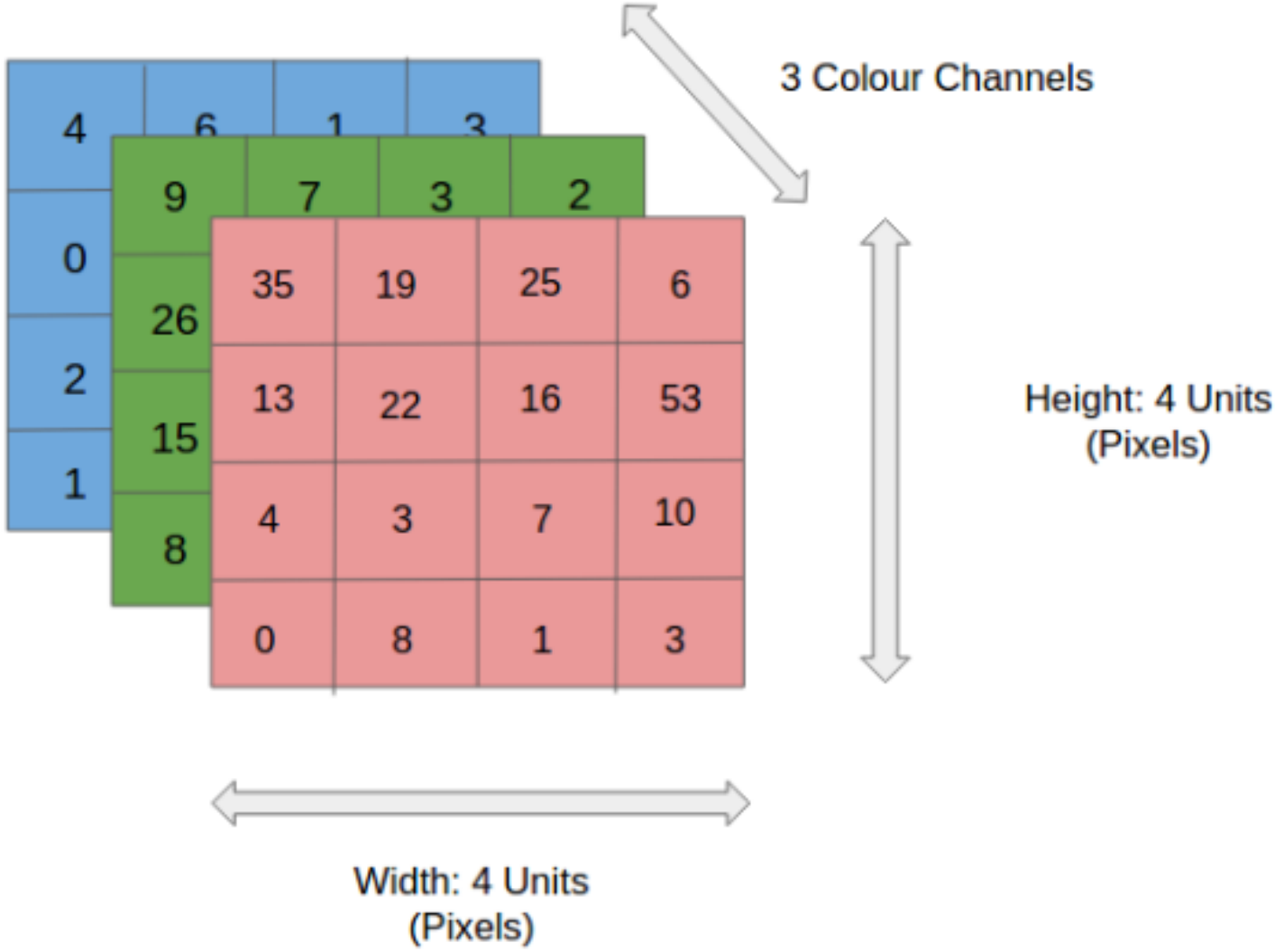
https://bit.ly/2RVDoFI

ConvNet 필요 이유



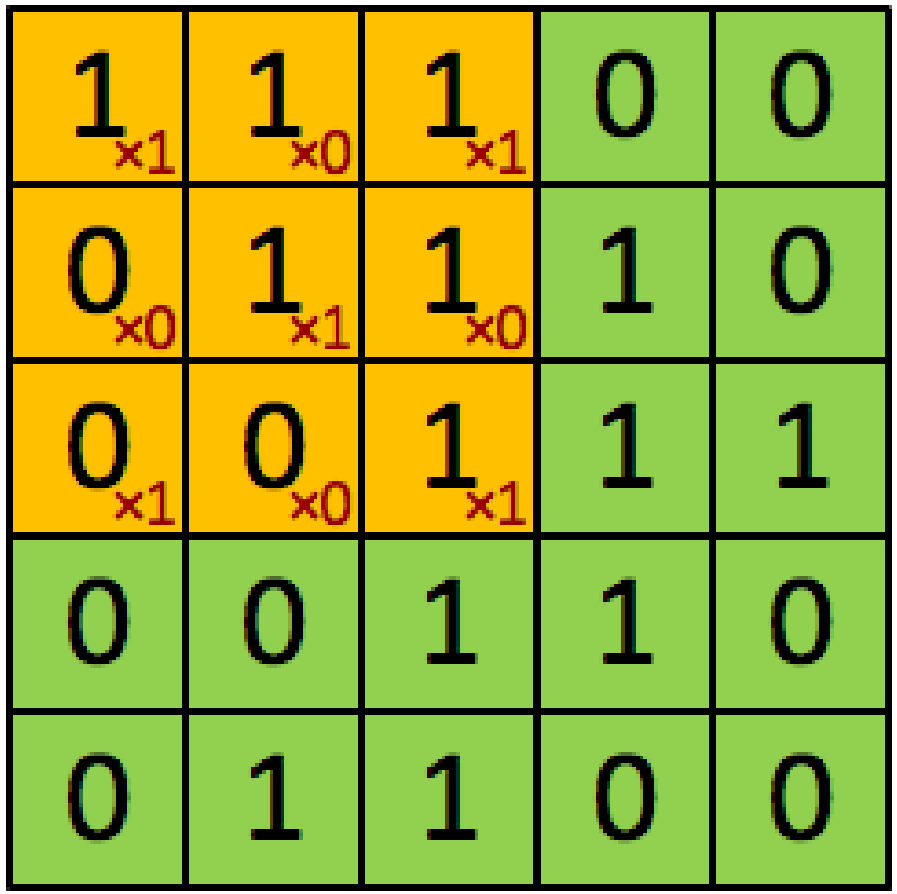
https://bit.ly/2RVDoFI

Input Image

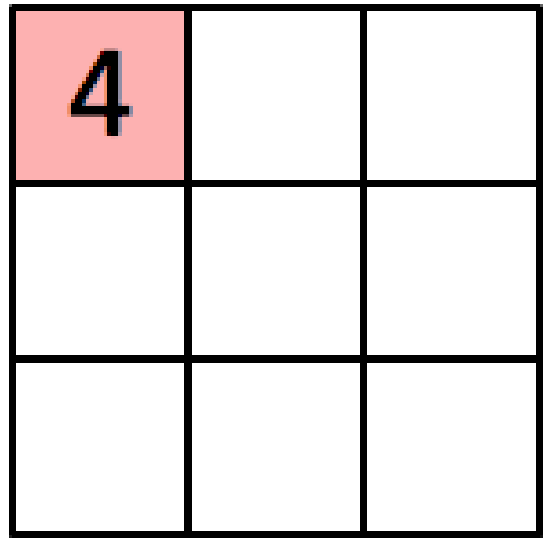


https://bit.ly/2RVDoFI

Convolution Layer - Kernel



Image



Convolved Feature

https://bit.ly/2RVDoFI

Convolution Layer - Kernel

0	0	0	0	0	0	...
0	156	155	156	158	158	...
0	153	154	157	159	159	...
0	149	151	155	158	159	...
0	146	146	149	153	158	...
0	145	143	143	148	158	...
...	...	...	...	...	...	...

Input Channel #1 (Red)

0	0	0	0	0	0	...
0	167	166	167	169	169	...
0	164	165	168	170	170	...
0	160	162	166	169	170	...
0	156	156	159	163	168	...
0	155	153	153	158	168	...
...	...	...	...	...	...	...

Input Channel #2 (Green)

0	0	0	0	0	0	...
0	163	162	163	165	165	...
0	160	161	164	166	166	...
0	156	158	162	165	166	...
0	155	155	158	162	167	...
0	154	152	152	157	167	...
...	...	...	...	...	...	...

Input Channel #3 (Blue)

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

Kernel Channel #1

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

Kernel Channel #2

0	1	1
0	1	0
1	-1	1

Kernel Channel #3

$308 + (-498) + 164 + 1 = -25$

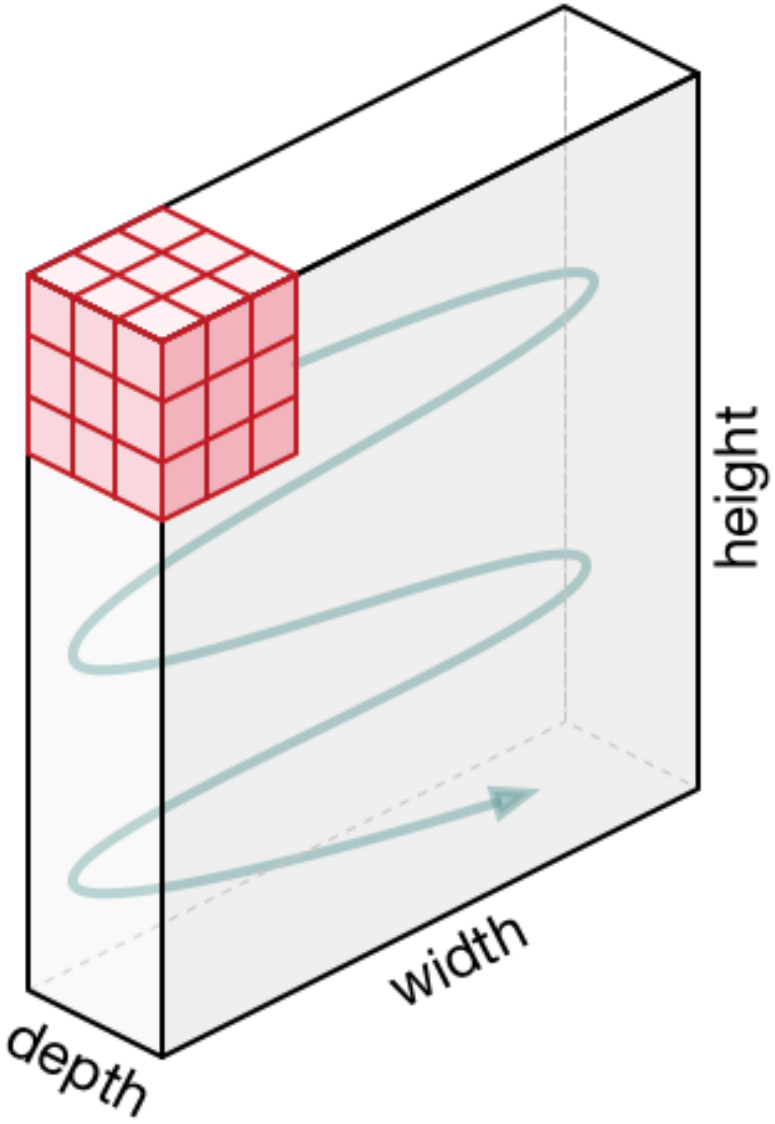
Bias = 1

Output

-25				...
				...
				...
				...
...	...	...	...	...

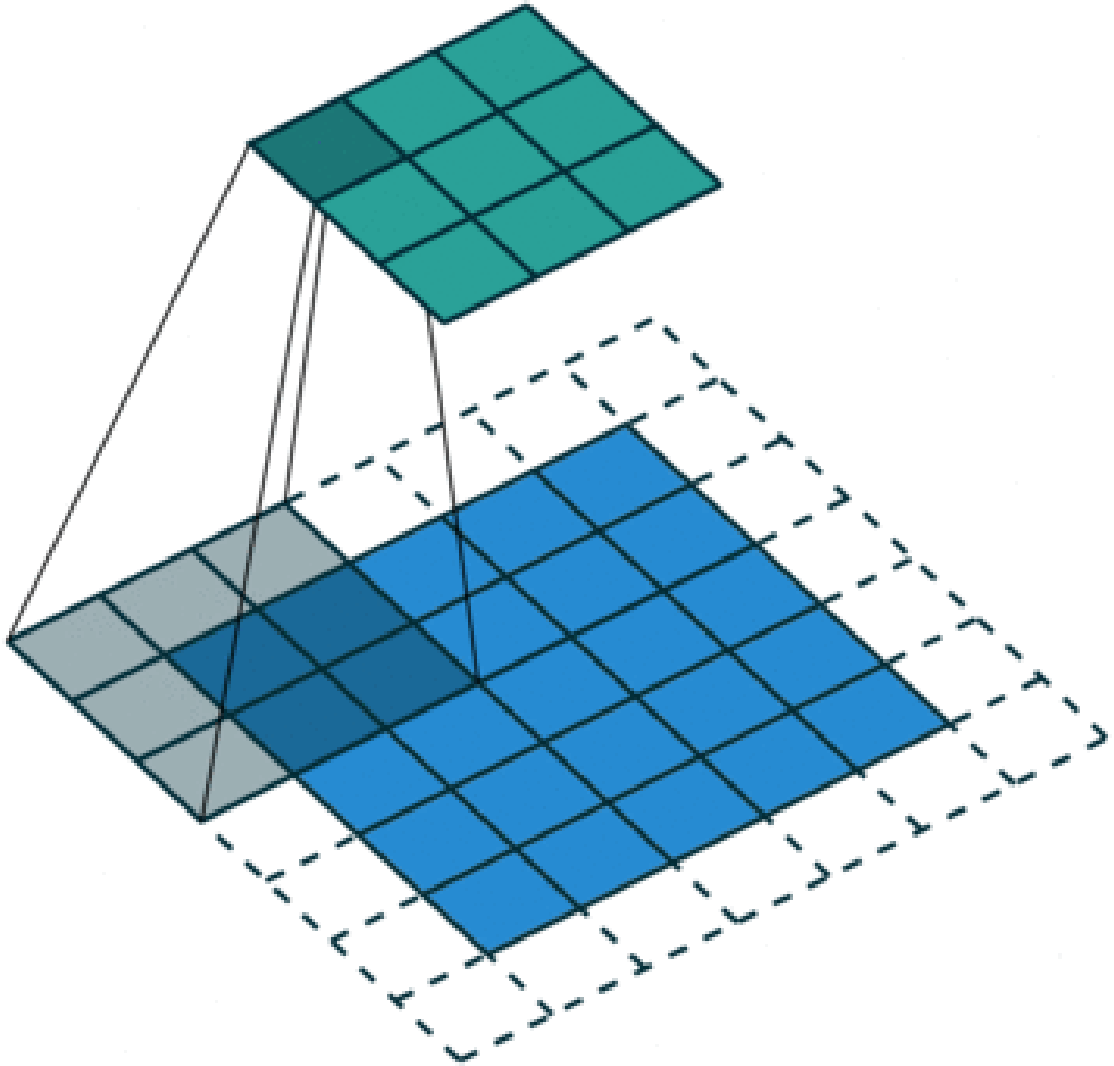
https://bit.ly/2RVDoFI

Convolution Layer - Kernel

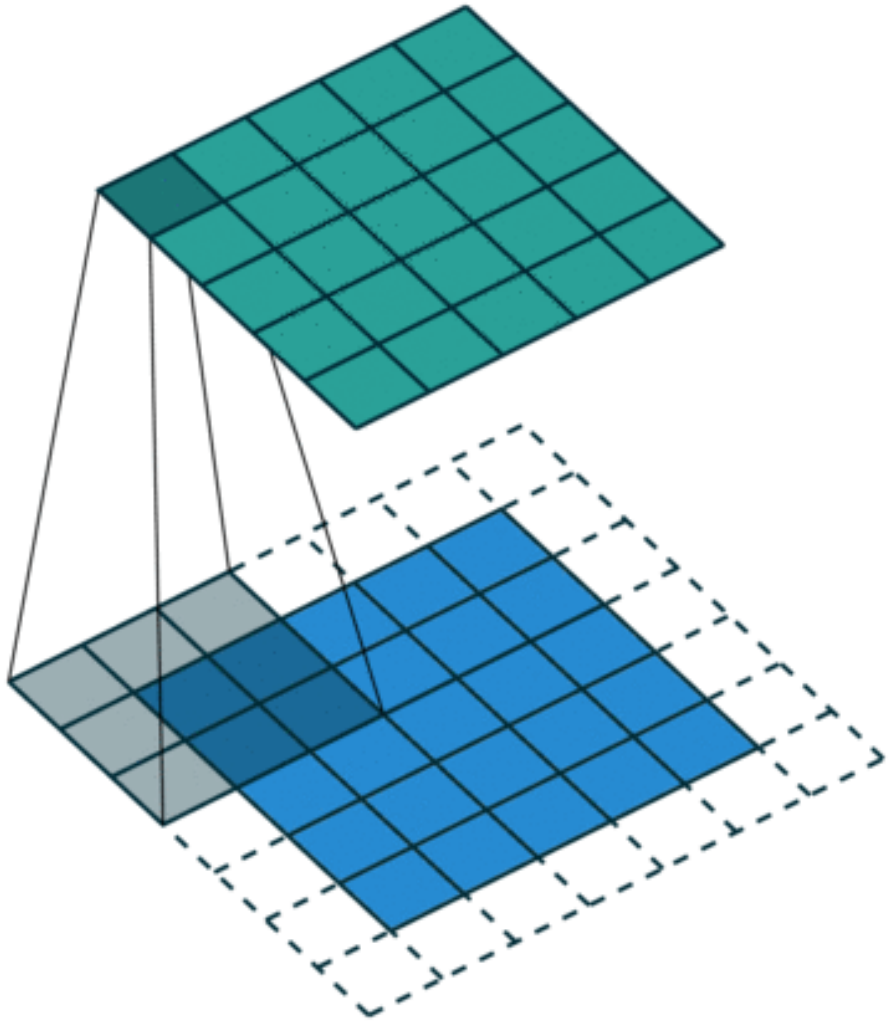


Convolution Layer

Convolution Operation with Stride Length = 2



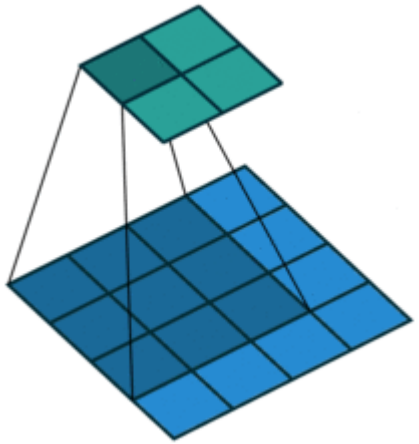
SAME padding



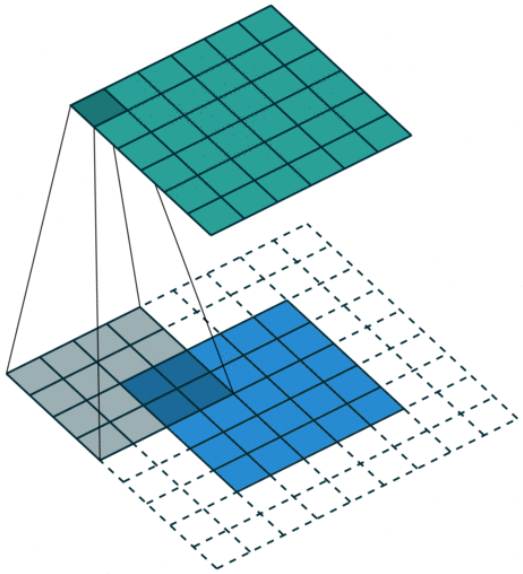
y/2RVDoFI

Convolution Layer - The Kernel

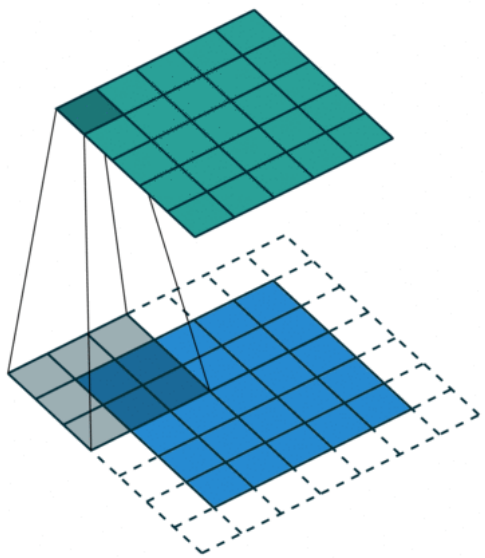
No padding, no strides



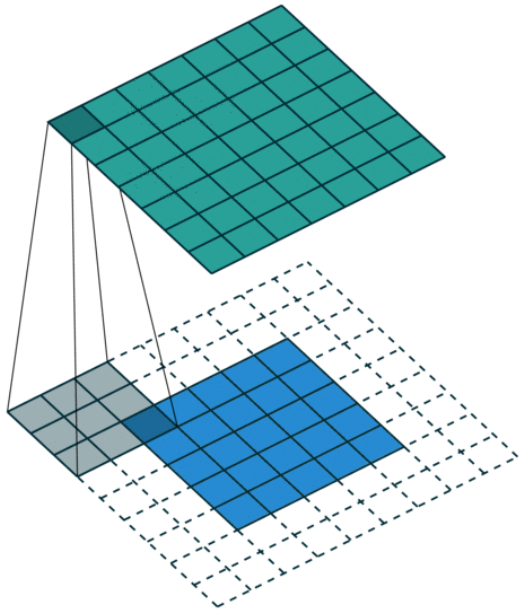
Arbitrary padding, no strides



Half padding, no strides



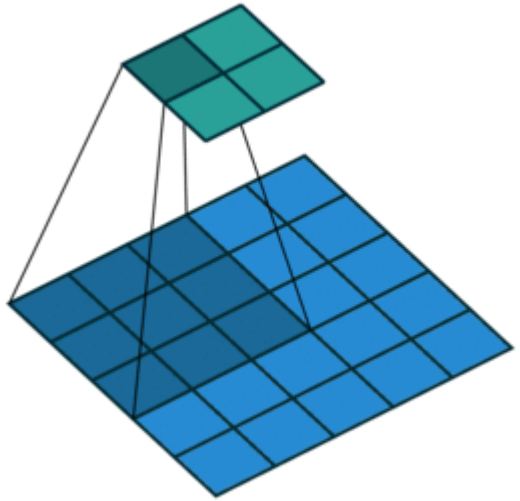
Full padding, no strides



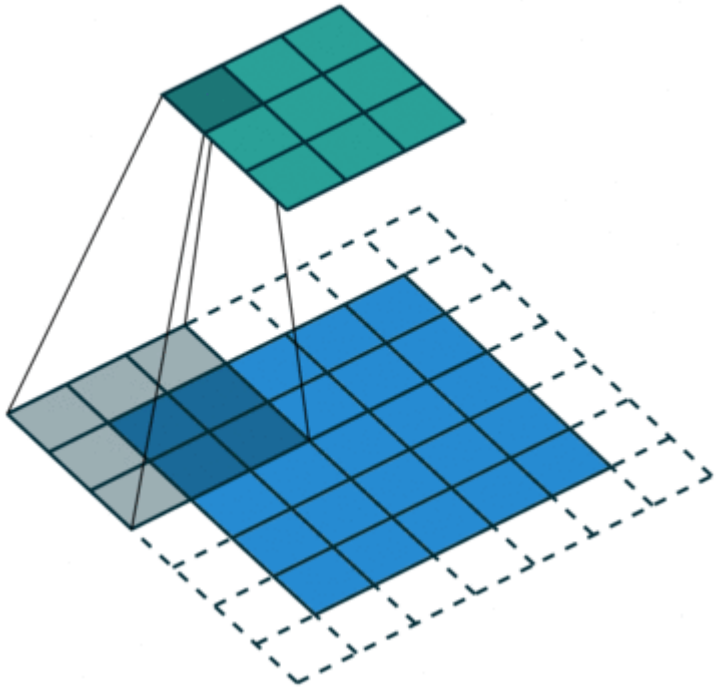
https://github.com/vdumoulin/conv_arithmetic

Convolution Layer - The Kernel

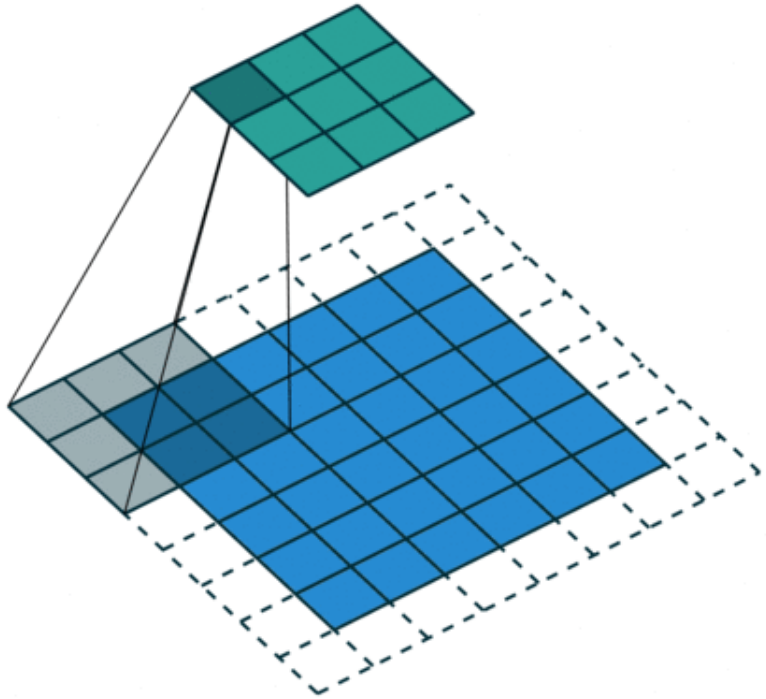
No padding, strides



Padding, strides



Padding, strides (odd)



https://github.com/vdumoulin/conv_arithmetic

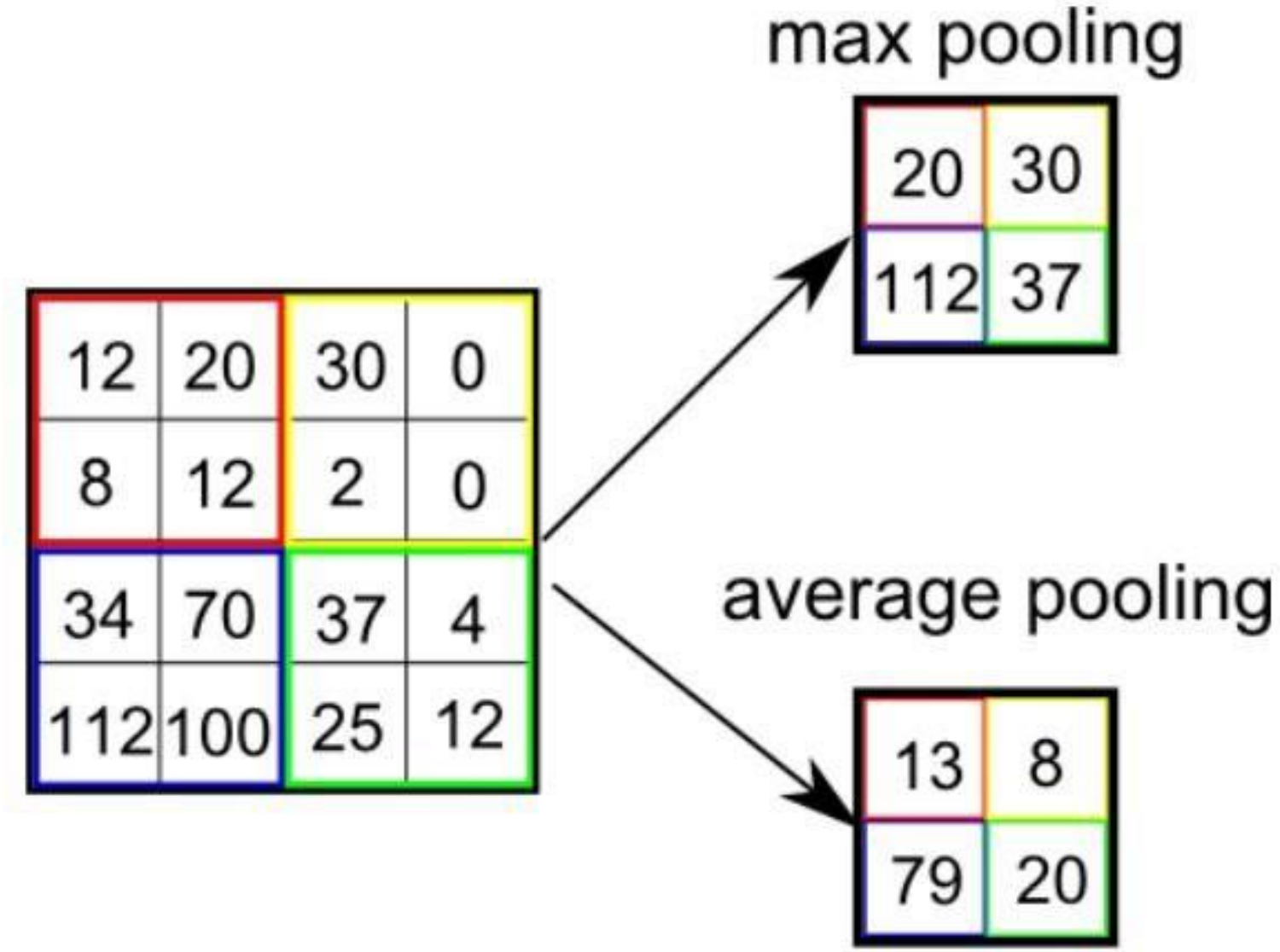
Pooling Layer

3.0	3.0	3.0
3.0	3.0	3.0
3.0	2.0	3.0

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

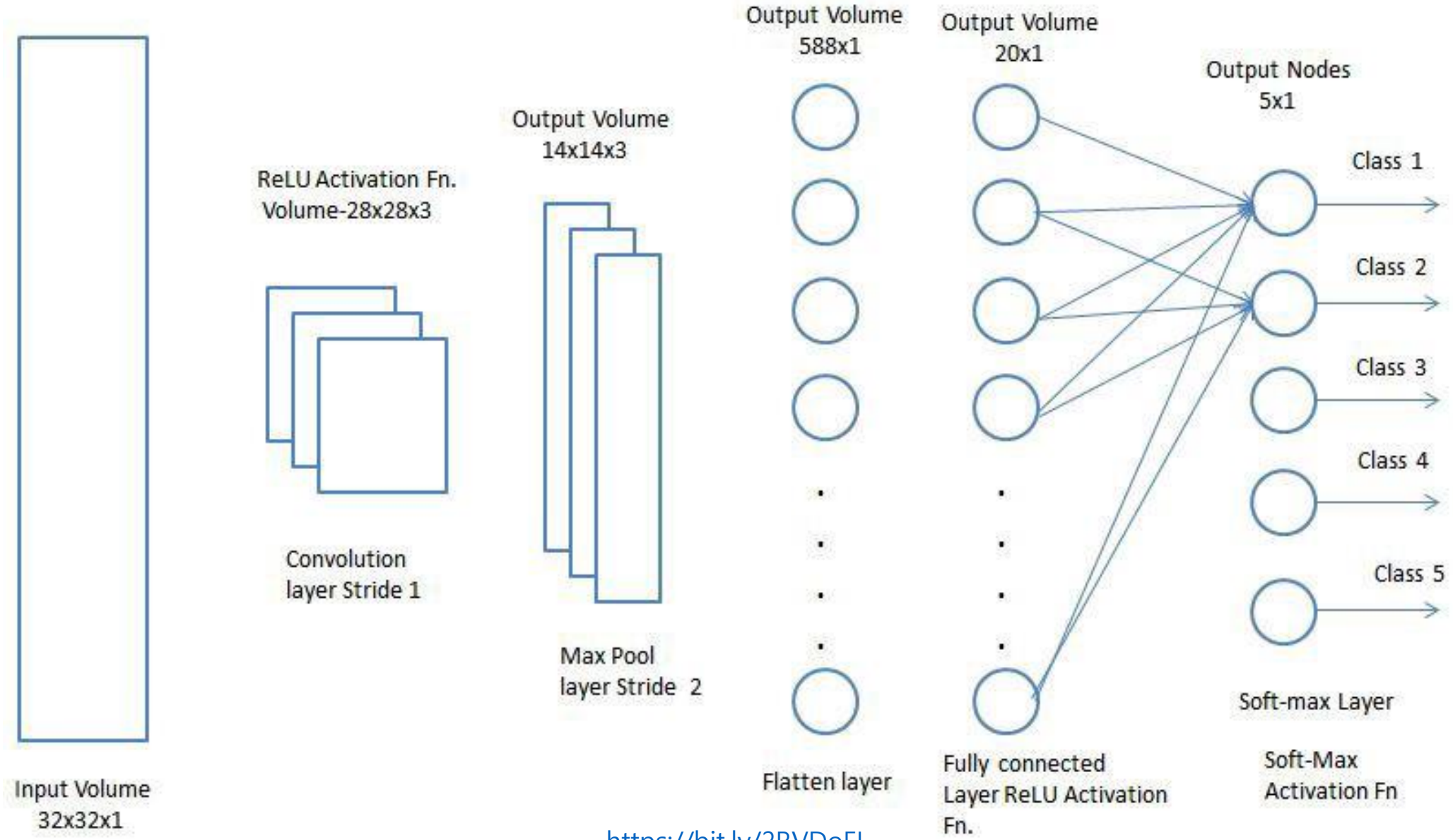
https://bit.ly/2RVDoFI

Pooling Layer



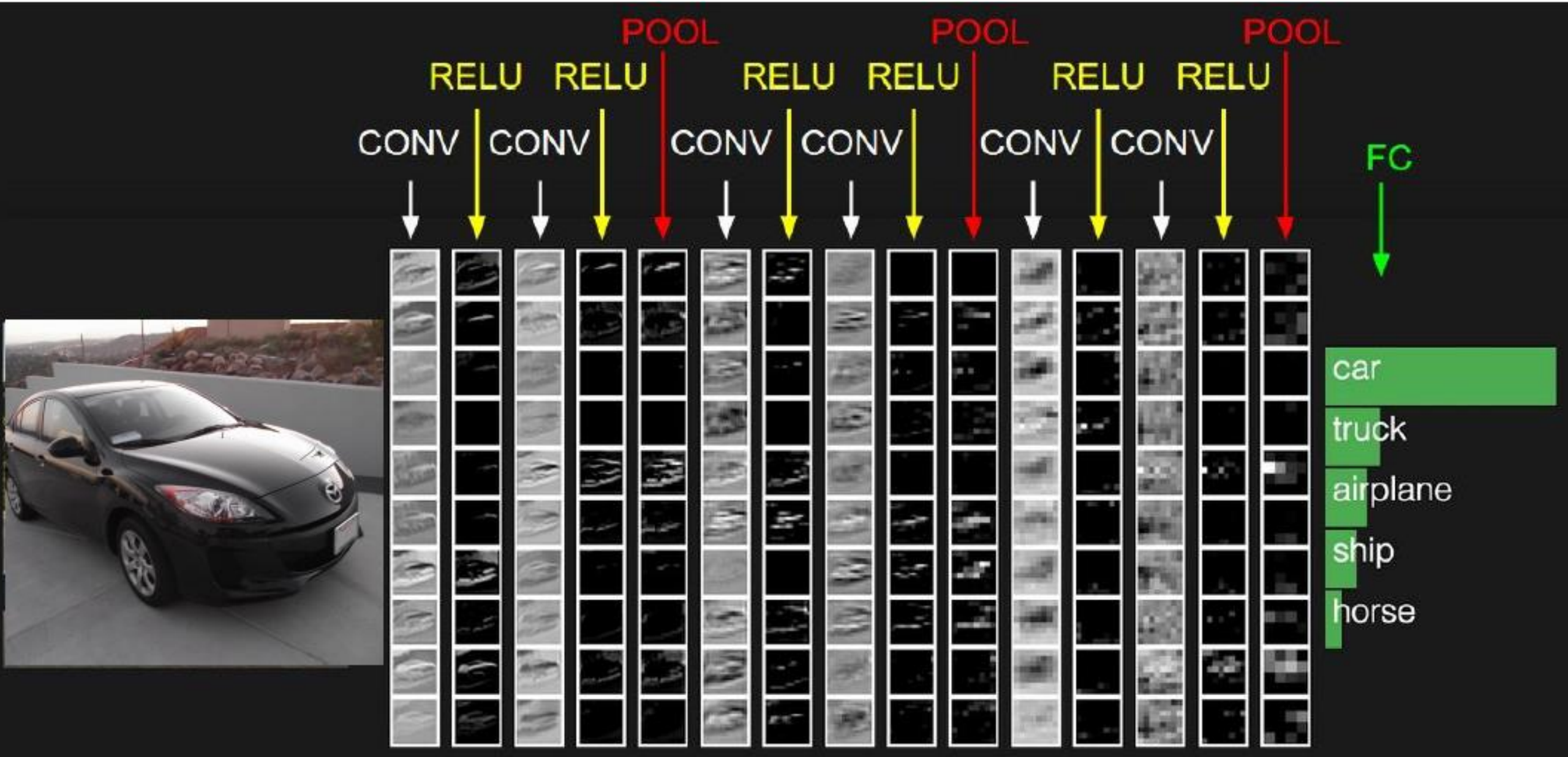
https://bit.ly/2RVDoFI

FC Layer(Fully Connected Layer)

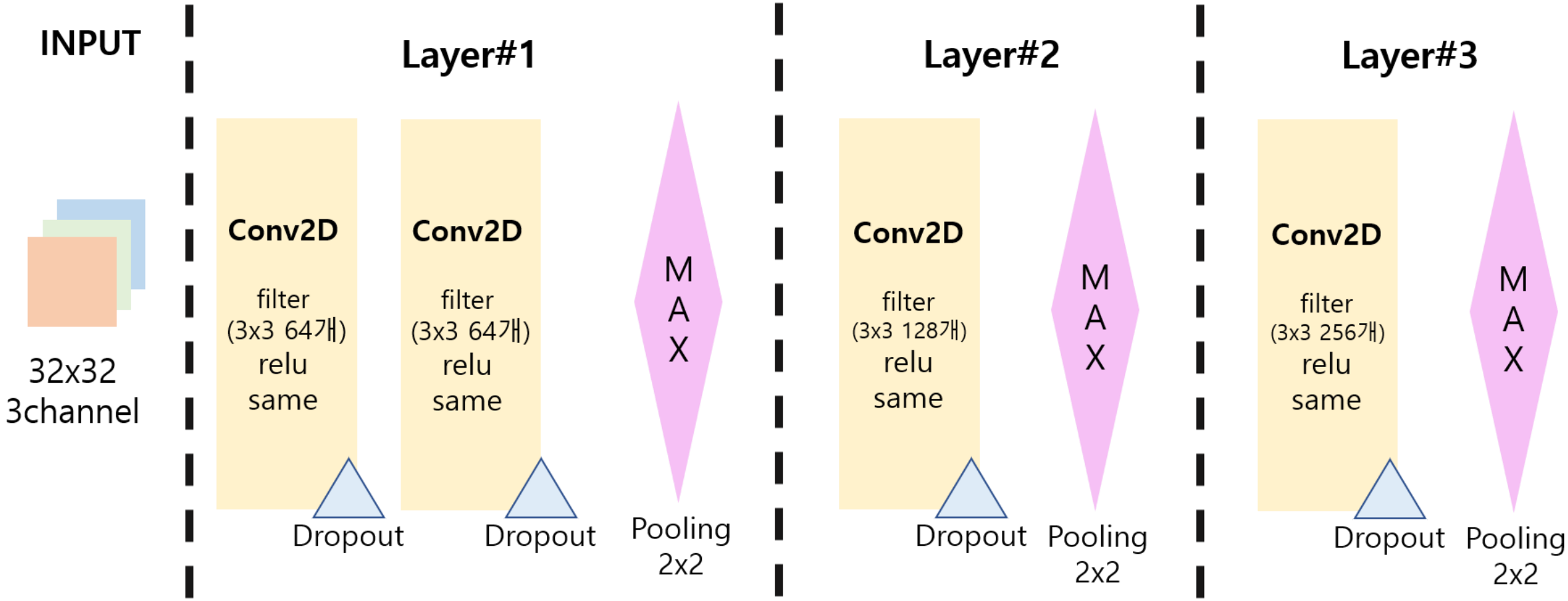


https://bit.ly/2RVDoFI

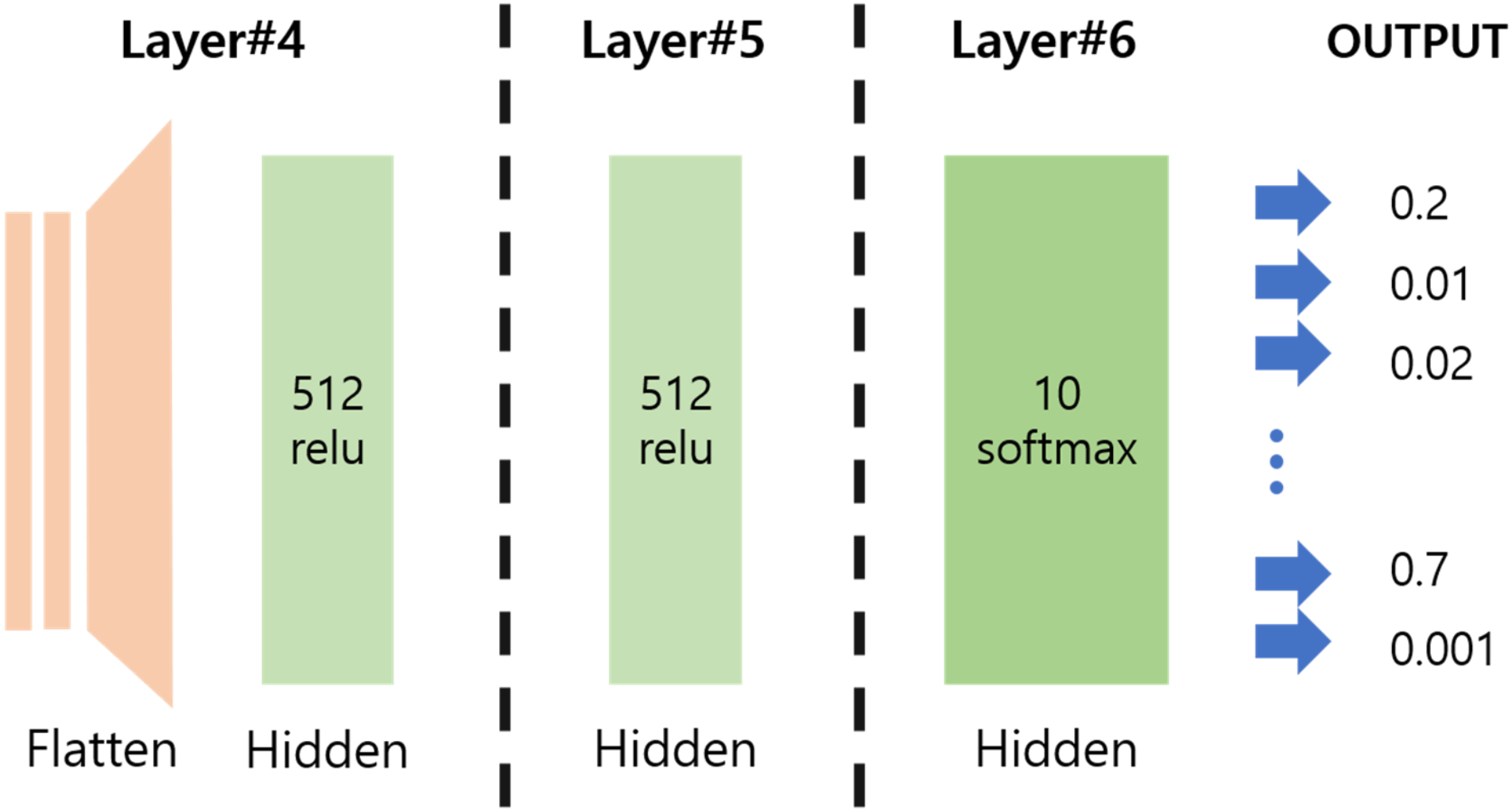
CNN 아키텍처


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

CNN 아키텍처 - Feature Extraction



CNN 아키텍처 - Classification



Thank you

kgpark88@gmail.com