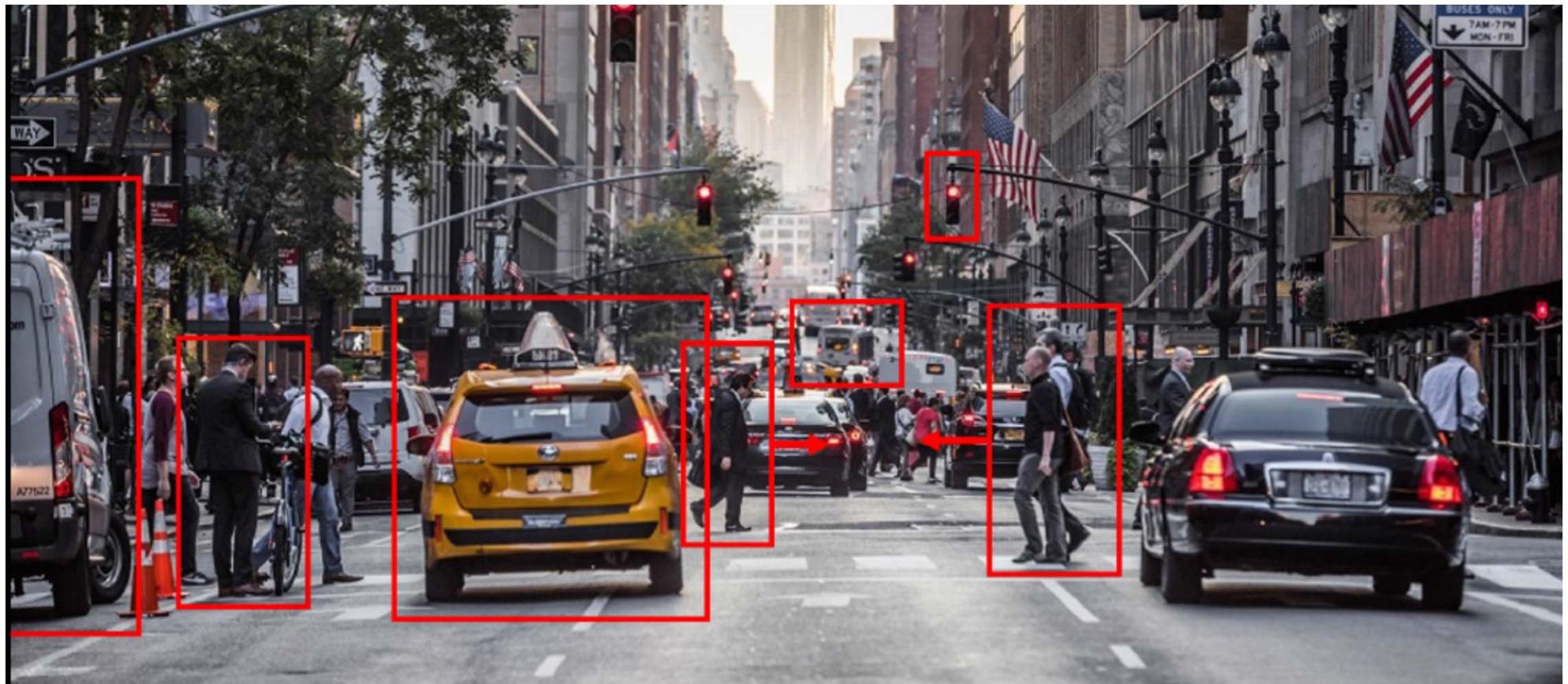


02. 딥러닝과 컴퓨터 비전

컴퓨터 비전

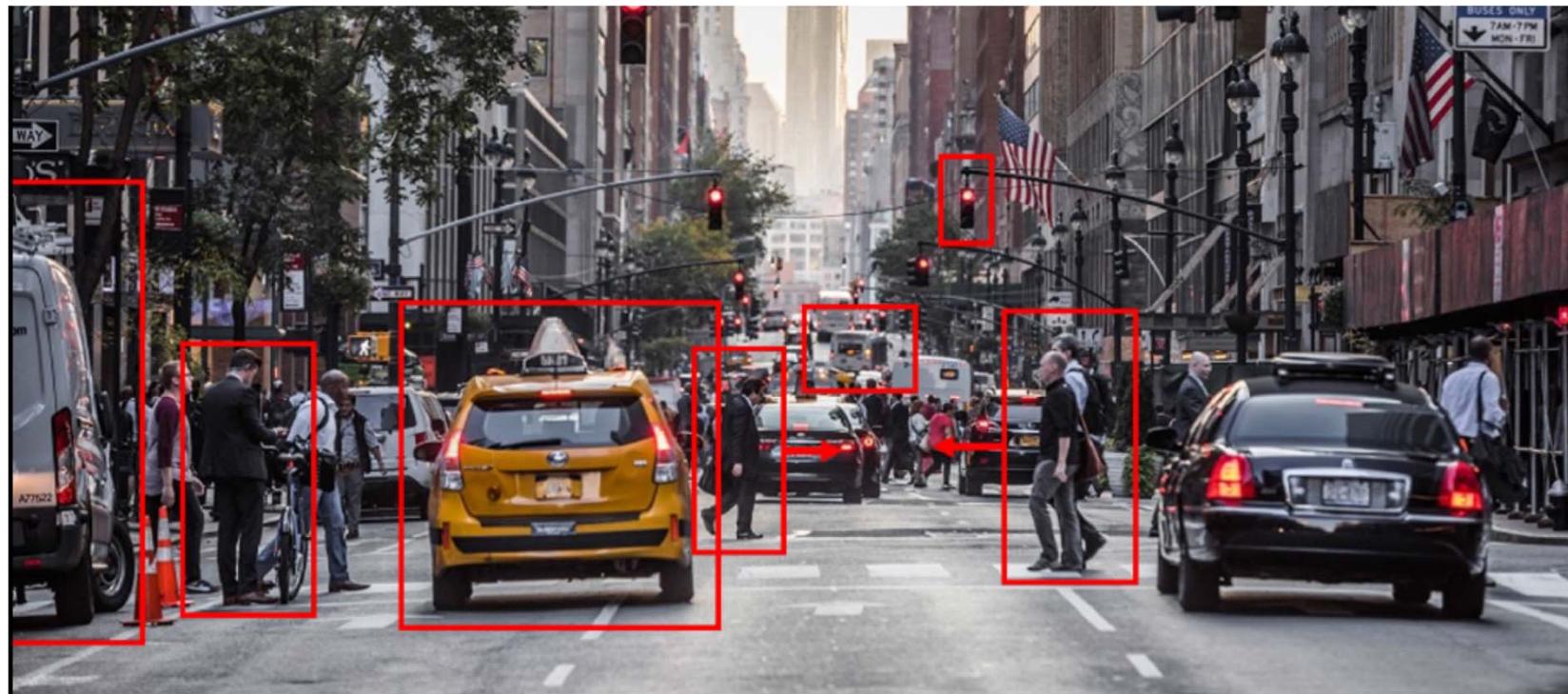
컴퓨터 비전

- 이미지에서 추출할 수 있는 정보
 - 사물의 종류, 위치, 색깔...



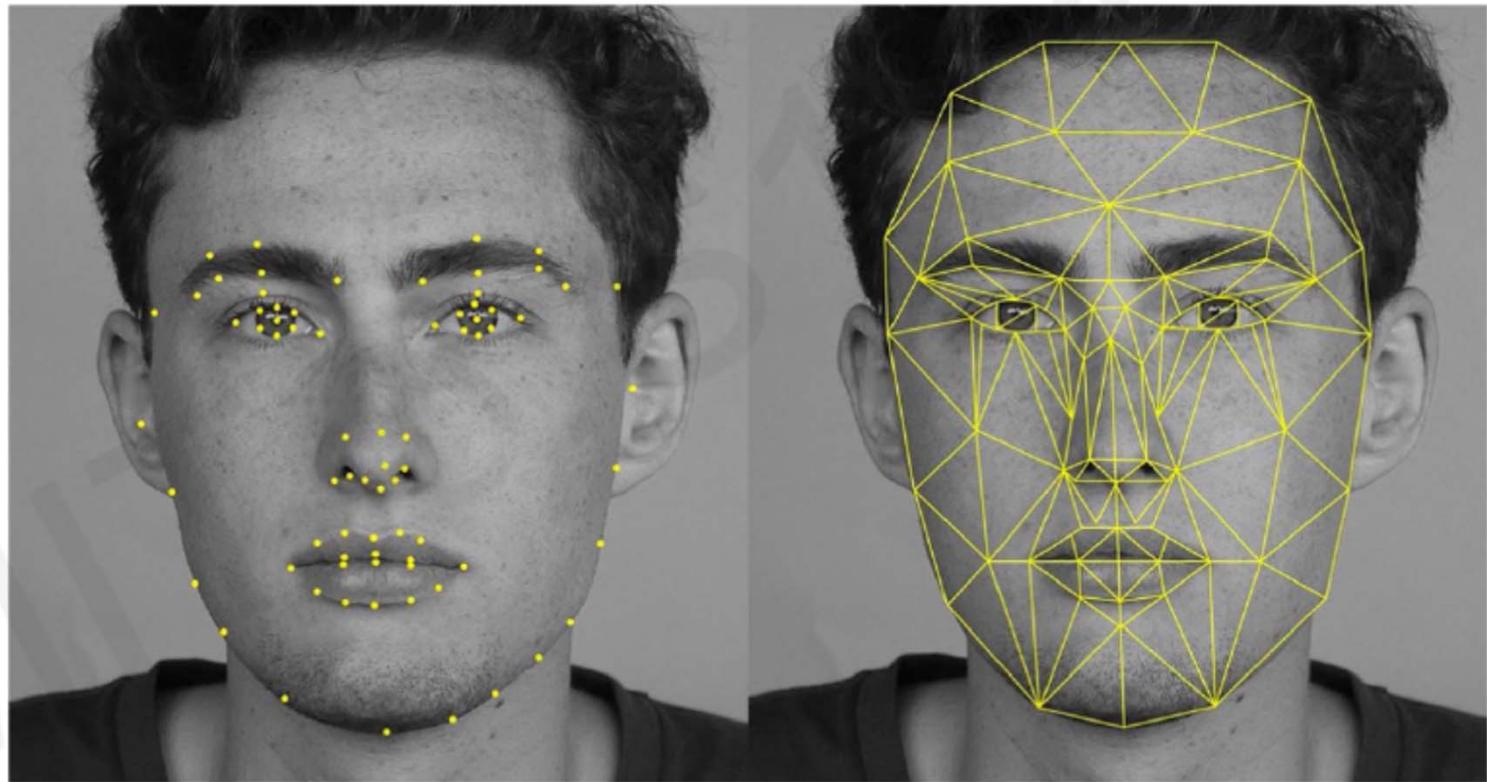
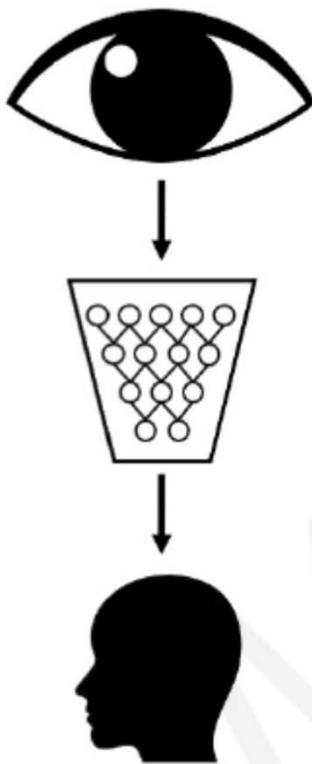
컴퓨터 비전

- 컴퓨터비전: 이미지로부터 유용한 정보를 추출하여 의사결정 및 예측에 사용하는 분야
- 이미지를 처리하는 사람의 작업을 대체할 수 있음 할 수 있음



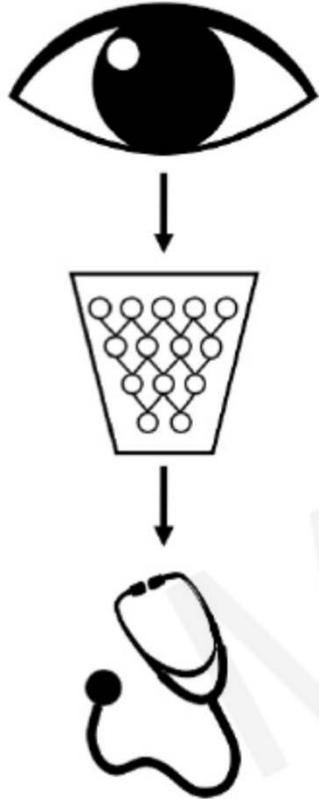
컴퓨터 비전의 응용 - 안면인식

- 경비
- 얼굴을 이용한 보안시스템

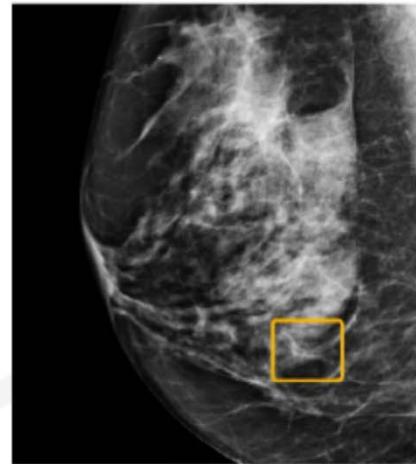


컴퓨터 비전의 응용 - 질병 진단

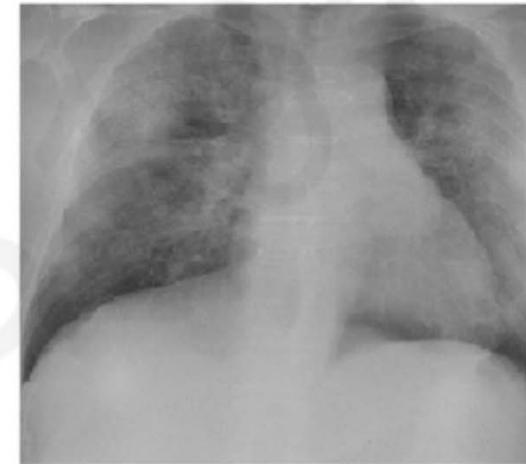
- 의사
- X-ray 사진으로부터 병명 진단



Breast cancer



COVID-19

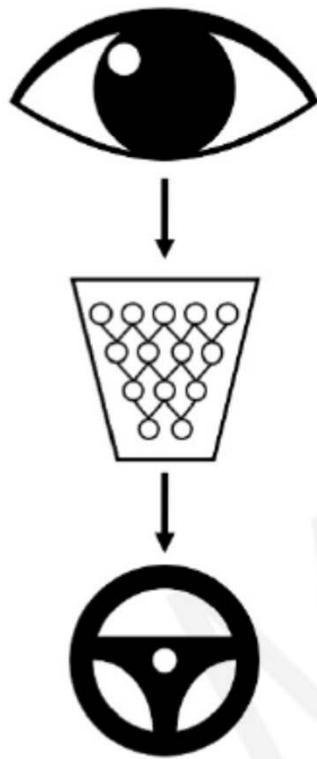


Skin cancer



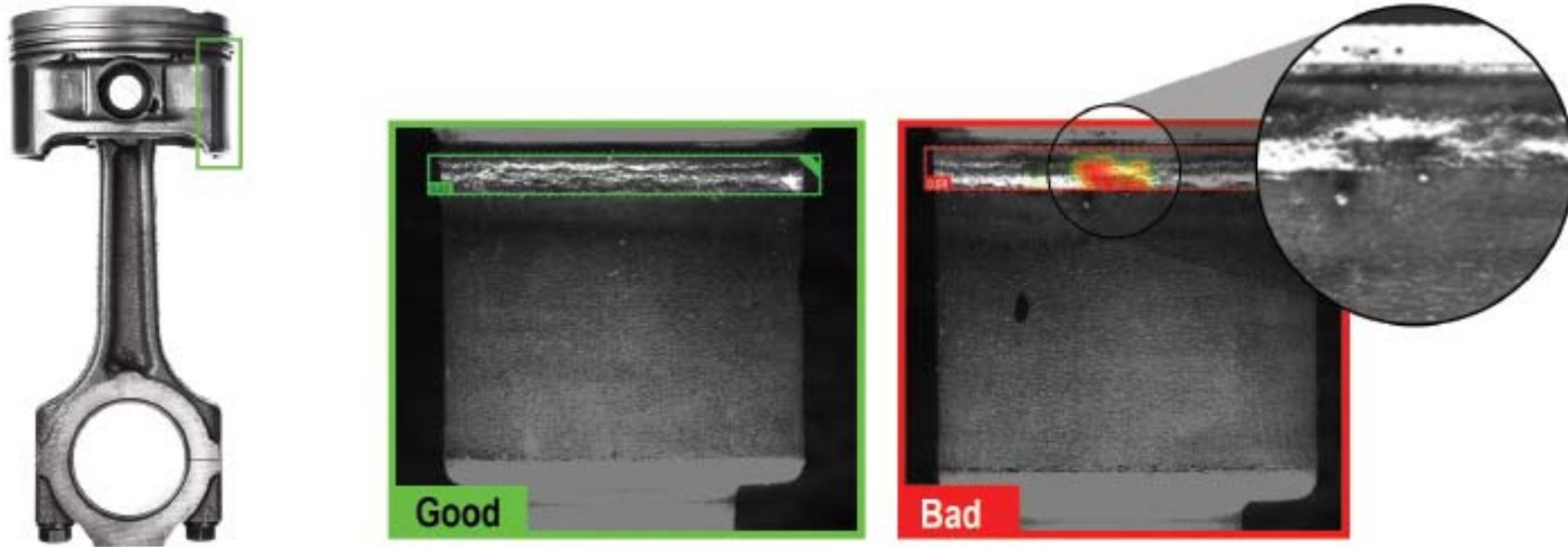
컴퓨터 비전의 응용 - 자율주행

- 운전자
- 실시간으로 들어오는 이미지를 처리하여 자율주행



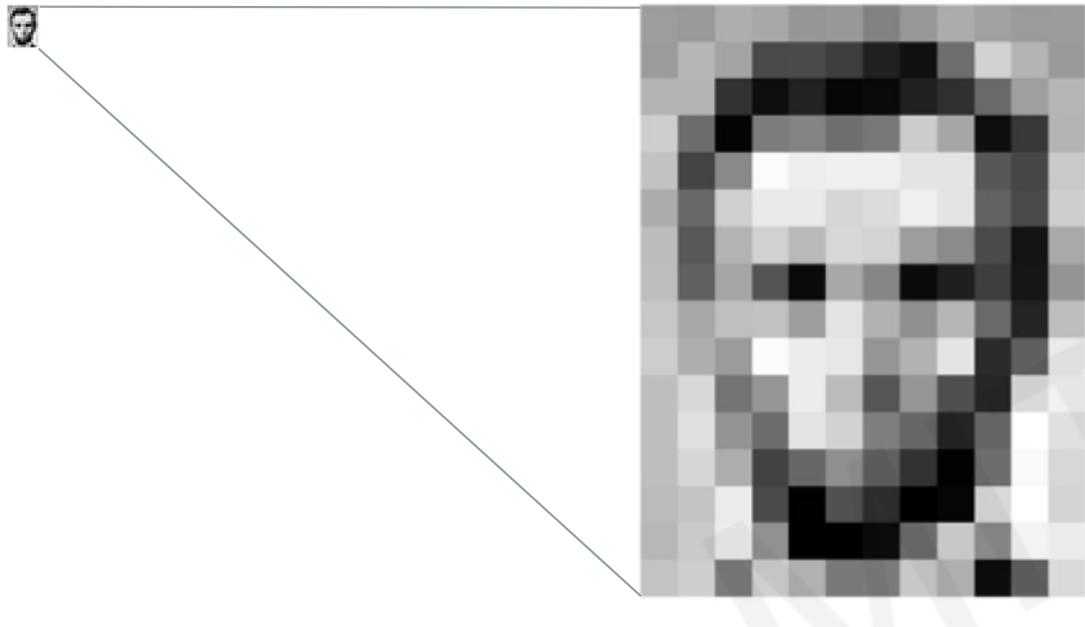
컴퓨터 비전의 응용 - 불량 진단

- 공장 품질검사팀
- 육안검사를 자동화 검사로 대체 가능



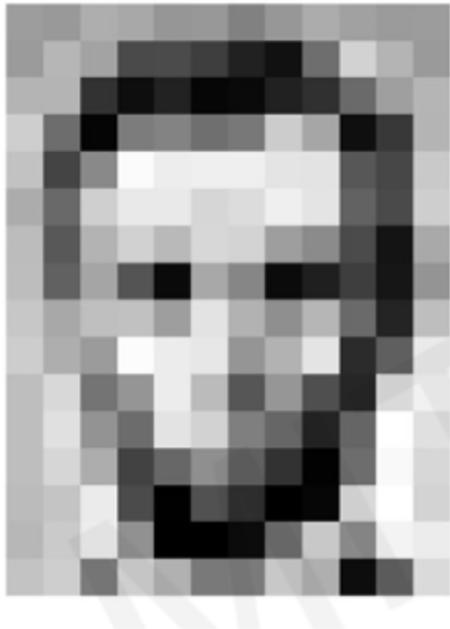
컴퓨터가 인식하는 이미지

- 컴퓨터의 비트맵 이미지를 크게 확대해 보면 다음과 같다
- 모든 이미지는 밝기가 다른 격자들로 구성되어 있음



컴퓨터가 인식하는 이미지

- 격자 = 픽셀 (pixel)
- 이미지를 구성하는 기본단위



컴퓨터가 인식하는 이미지

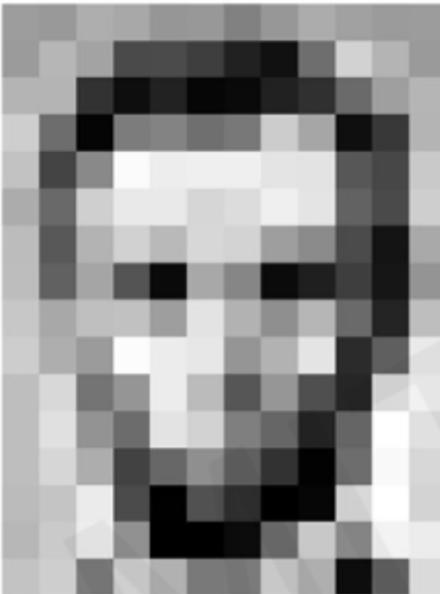
- 픽셀은 0~255 사이의 값을 가질 수 있음
- 255: 백색, 0: 흑색



157	153	174	168	150	152	129	151	172	163	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	6	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	38	218	241
190	224	147	108	227	210	127	102	36	105	255	224
190	214	173	66	103	143	96	90	2	109	249	215
187	196	236	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	176	13	96	218

컴퓨터가 인식하는 이미지

- 흑백이미지: pixel값으로 구성되어 있는 2차원 매트릭스로 표현됨
- 왜 이차원? (가로 pixel 수, 세로 pixel 수)



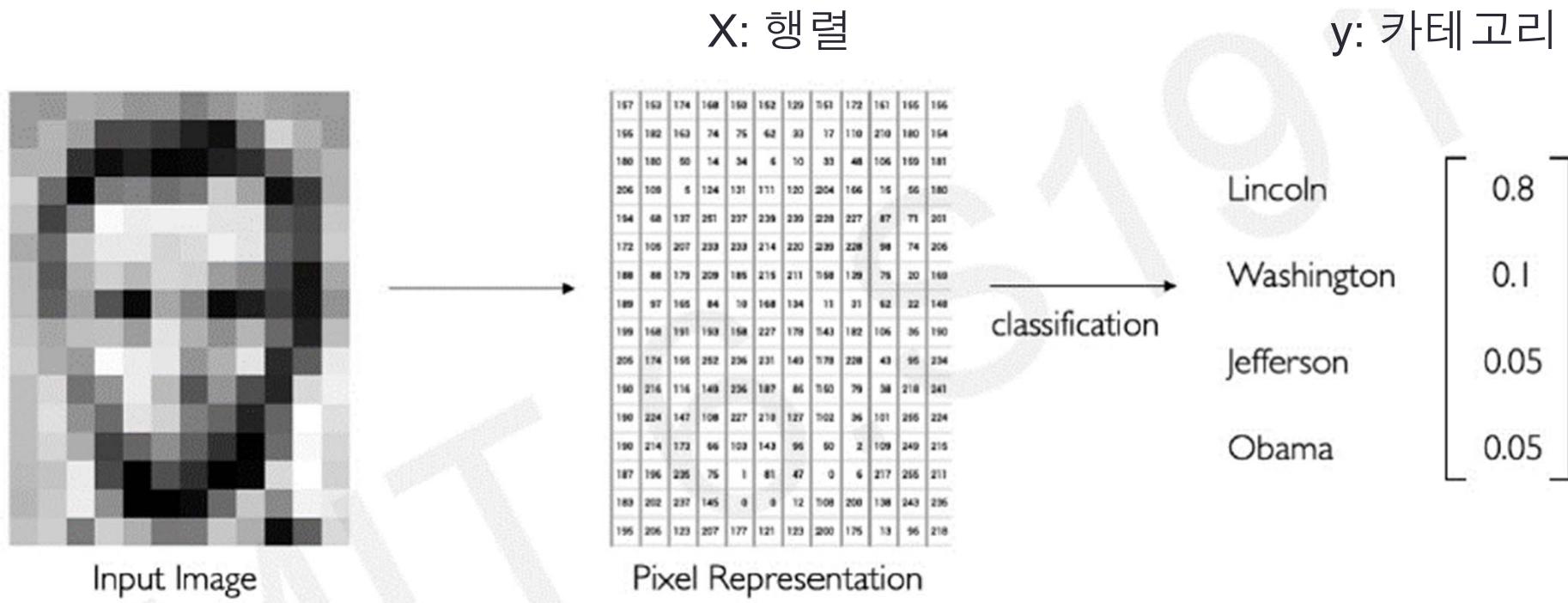
157	153	174	168	160	162	129	151	172	161	165	166
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	6	124	131	111	120	204	166	15	56	180
194	58	137	251	237	239	239	228	227	17	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	159	79	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	176	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	6	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	159	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	176	13	96	218

컴퓨터가 인식하는 이미지
12x18 pixel image

컴퓨터 비전

- 이미지가 숫자로 이루어진 데이터로 표현 가능하므로, 이를 입력데이터로 활용하여 회귀 혹은 분류를 위한 모델을 만들 수 있음



참고: 색깔 있는 이미지의 표현

- 모든 색깔은 R,G,B 3원색의 조합으로 표현 가능
- 칼라 이미지의 경우 3차원 픽셀 매트릭스로 표현 가능

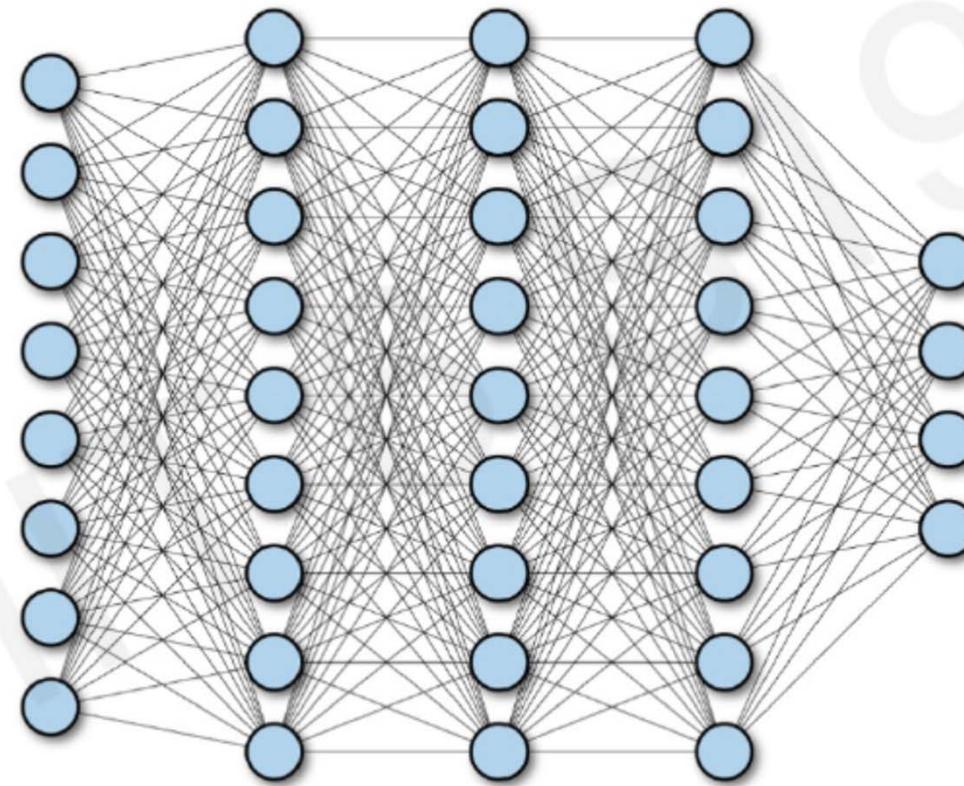


02. 딥러닝과 컴퓨터 비전

기존 방식의 문제점

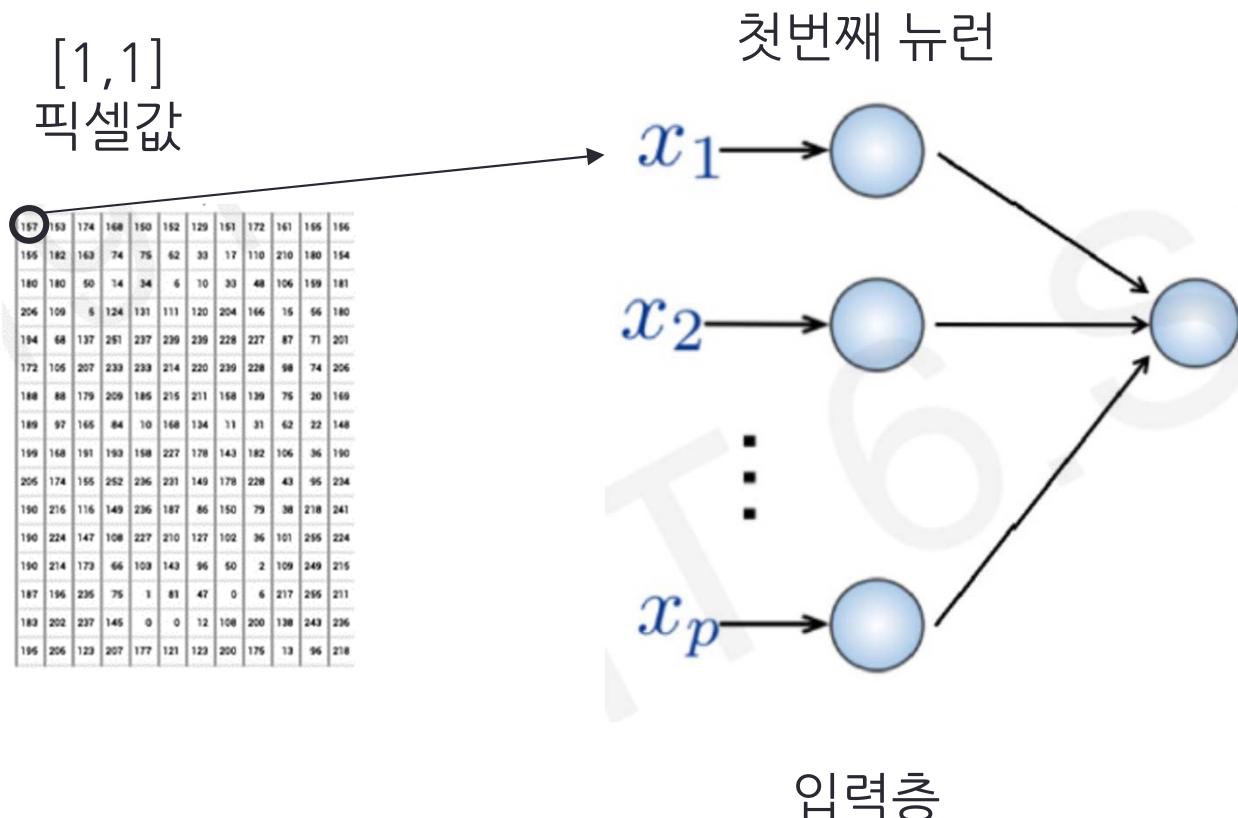
(지금까지 배운) 인공신경망 구조

- Fully Connected Neural Network
 - 기존의 딥뉴럴네트워크를 사용하여 이미지 분류하기



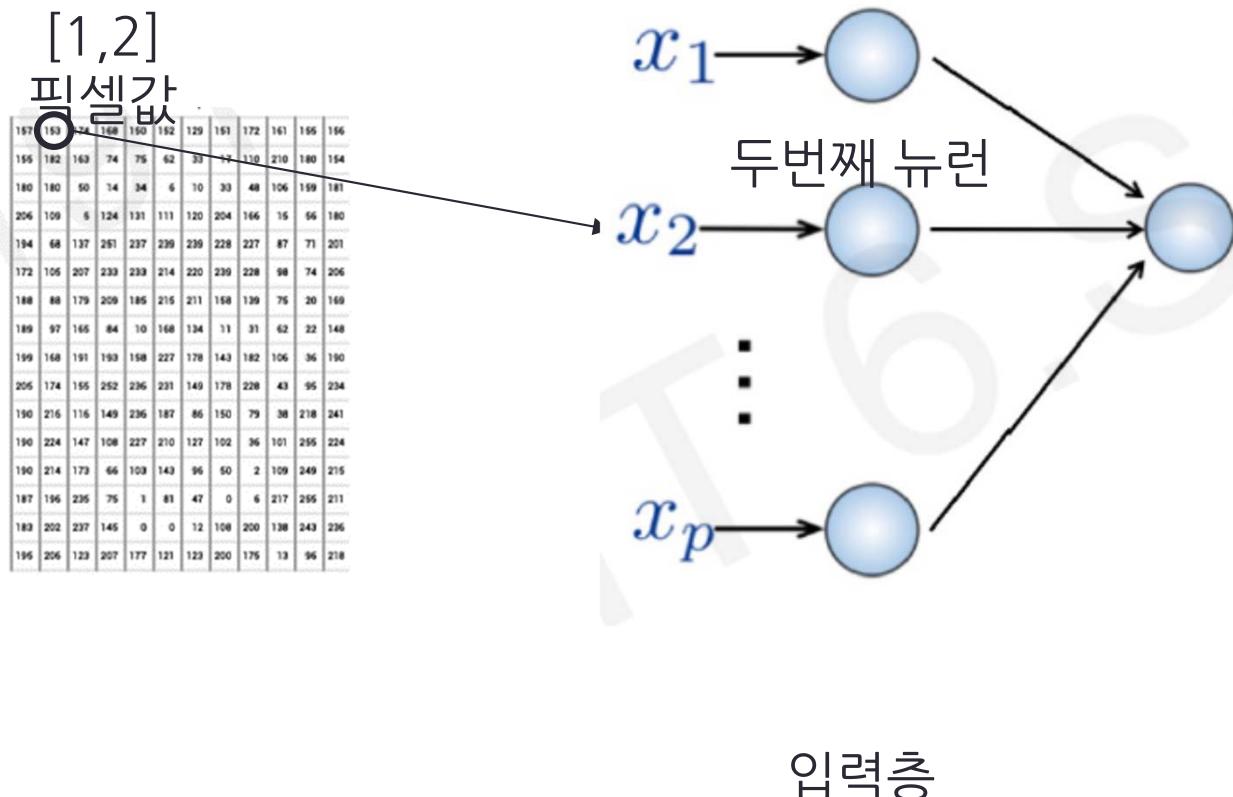
일반적 인공신경망 구조를 사용하는 경우

- Fully Connected Neural Network
- 입력노드
 - 행렬을 1차원으로 바꾸어 (flatten) 일반적 인공신경망에 입력시킬 수 있다.



일반적 인공신경망 구조를 사용하는 경우

- Fully Connected Neural Network
- 행렬을 1차원으로 바꾸어 (flatten) 일반적 인공신경망에 입력시킬 수 있다.



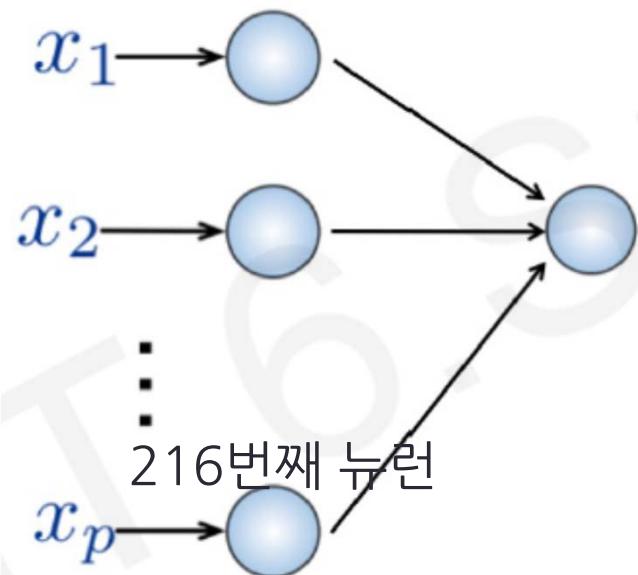
일반적 인공신경망 구조를 사용하는 경우

- Fully Connected Neural Network
- 행렬을 1차원으로 바꾸어 (flatten) 일반적 인공신경망에 입력시킬 수 있다.

157	153	174	168	150	152	129	151	172	161	195	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	30	48	106	199	181
206	109	6	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	230	228	227	87	71	201
172	105	207	232	233	214	220	230	228	98	74	206
188	88	179	209	186	215	211	158	139	75	20	169
189	97	166	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	106	227	210	127	102	36	101	295	224
190	214	173	66	109	143	96	50	2	109	249	215
187	156	235	75	1	81	47	0	6	217	295	211
183	202	237	145	9	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

[18, 12]
픽셀값

입력층

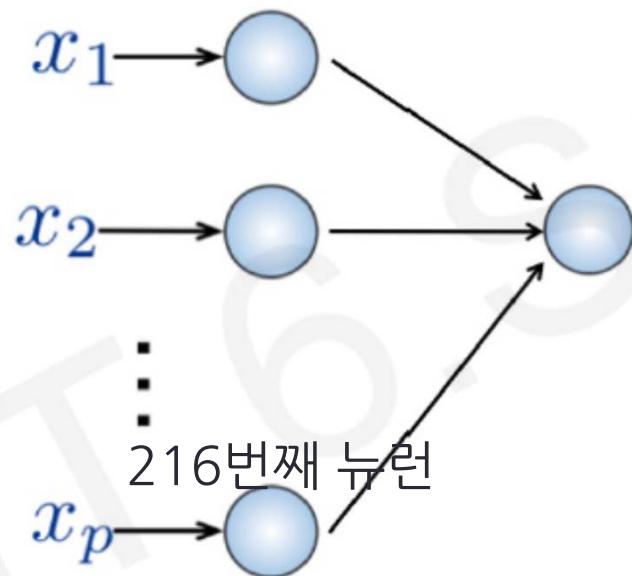


문제점

- Fully Connected Neural Network
- 행렬을 1차원으로 바꾸어 (flatten) 일반적 인공신경망에 입력시킬 수 있다.

157	153	174	168	150	152	129	151	172	161	195	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	199	181
206	109	6	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	230	228	227	87	71	201
172	105	207	232	233	214	220	230	228	98	74	206
188	88	179	209	186	215	211	158	139	75	20	169
189	97	166	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	36	218	241
190	224	147	106	227	210	127	102	36	101	295	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	295	211
183	202	237	145	9	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

[18, 12]
픽셀값



입력층

- 1) 이차원 데이터가 일차원으로 바뀌면서 공간적 정보가 손실
- 2) Fully Connected 구조에서 입력 feature의 수가 매우 많음 → 학습에 필요한 패러미터 수가 많아짐

인간이 이미지를 인식하는 과정

- 사람은 이미지를 어떻게 분류할까?



인간이 이미지를 인식하는 과정

- 사람은 이미지를 어떻게 분류할까?
- 사람은 특정 카테고리가 갖고 있는 **핵심 특성**들을 알고 있다.
- 해당 이미지에 **핵심 특성**들이 있는지 비교한다.



눈
코
입

번호판
바퀴
헤드램프

문
창문
계단

인간이 이미지를 인식하는 과정

사전 지식

특성 정의

분류를 위한
특성 비교



눈
코
입

번호판
바퀴
헤드램프

문
창문
계단

인간이 이미지를 인식하는 과정

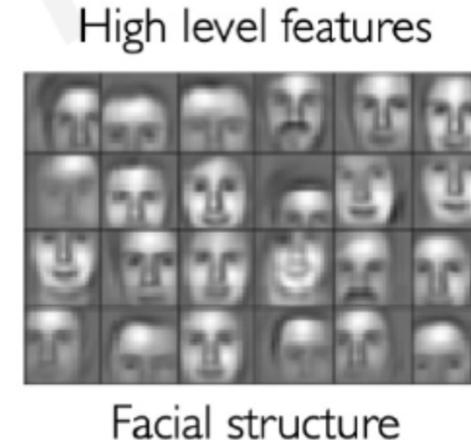
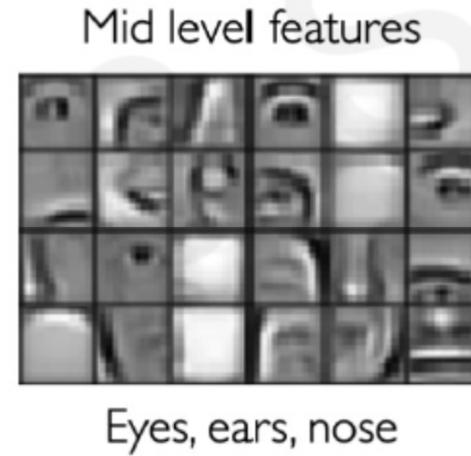
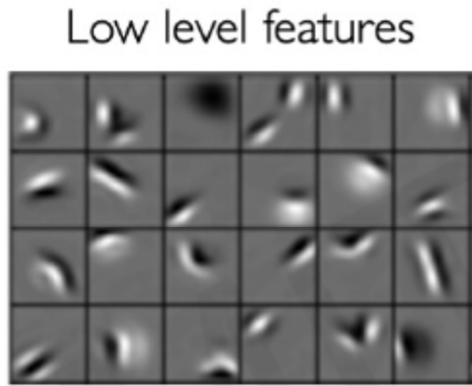


컴퓨터 비전이 해결해야하는 질문

- 사람의 개입 없이 자동으로 이미지의 핵심 특성을 추출할 수 있을까?

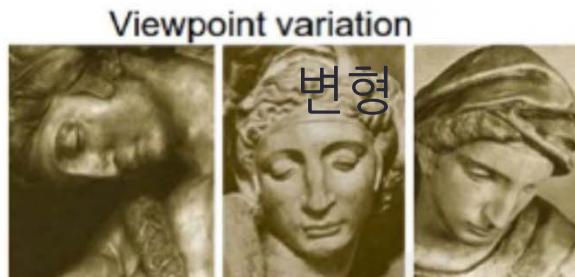
컴퓨터 비전이 해결해야 하는 질문

- 사람의 개입 없이 자동으로 이미지의 핵심 특성을 추출할 수 있을까?



특성 파악을 어렵게 하는 요인들

- 직관적인 사람과 달리 컴퓨터가 특성을 자동적으로 파악하는데에는 다음과 같은 어려움이 존재



Scale variation



Deformation



Background clutter



Occlusion



Intra-class variation



명암, 각도의 차이

크기

변형/배경

02. 딥러닝과 컴퓨터 비전

콘볼루션 연산

입력값: 이미지

- 입력값
 - 2차원 매트릭스의 형태로 픽셀의 데이터를 표현

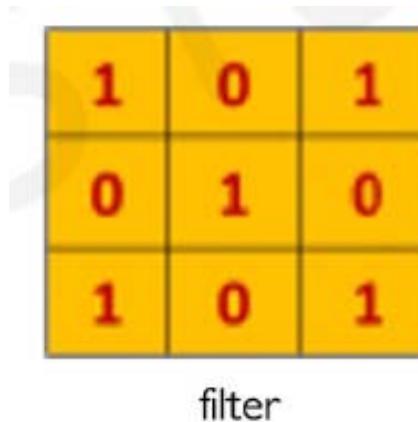
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

image

5x5 이미지

커널 (Kernel)

- 커널 (kernel)
 - 이미지 내에 존재하는 특성을 포착
 - 커널의 크기: (가로 x 세로) 형태의 픽셀로 표현



3x3 커널

콘볼루션 연산(Convolutional Operation)

- 커널로 이미지의 왼쪽위부터 오른쪽 아래 방향으로 훑으면서 스칼라 곱 연산을 진행

이미지

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

image

커널

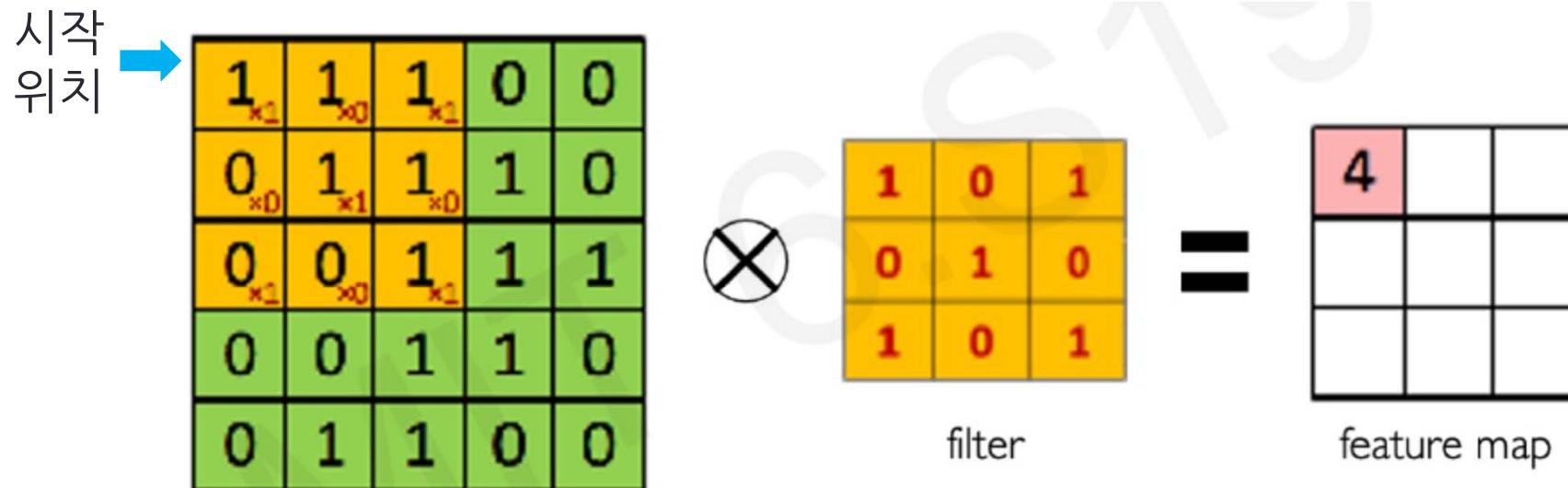


1	0	1
0	1	0
1	0	1

filter

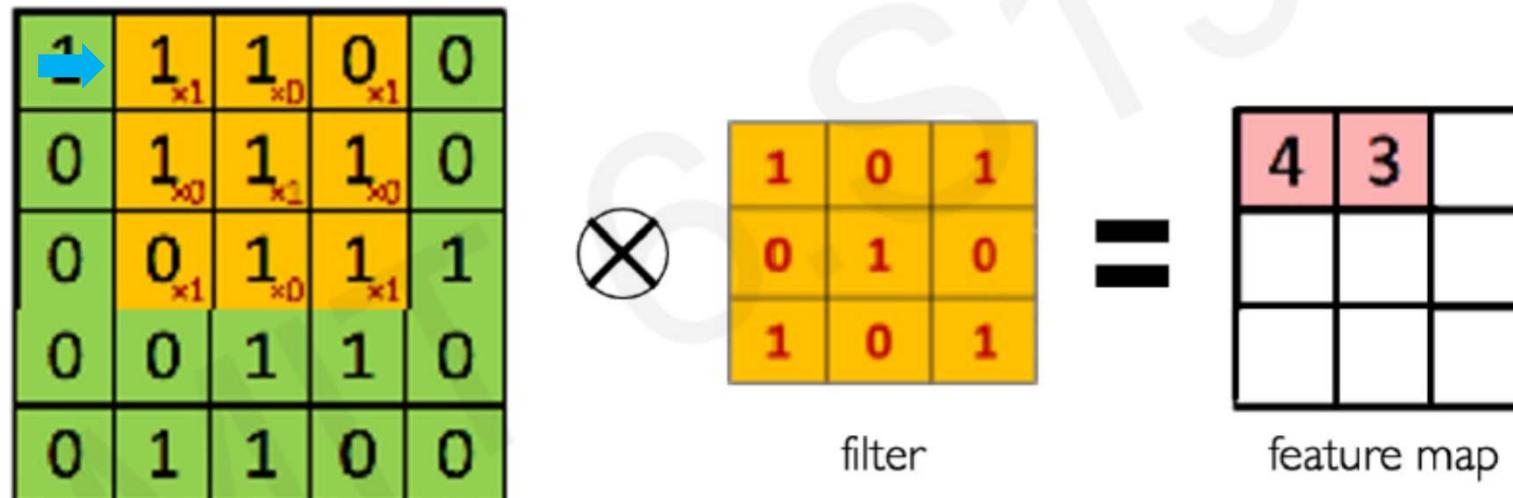
콘볼루션 연산(Convolutional Operation)

- 커널로 이미지의 왼쪽위부터 오른쪽 아래 방향으로 훑으면서 스칼라 곱 연산을 진행
- 이미지 왼쪽 위부분과 커널사이의 곱연산 예시



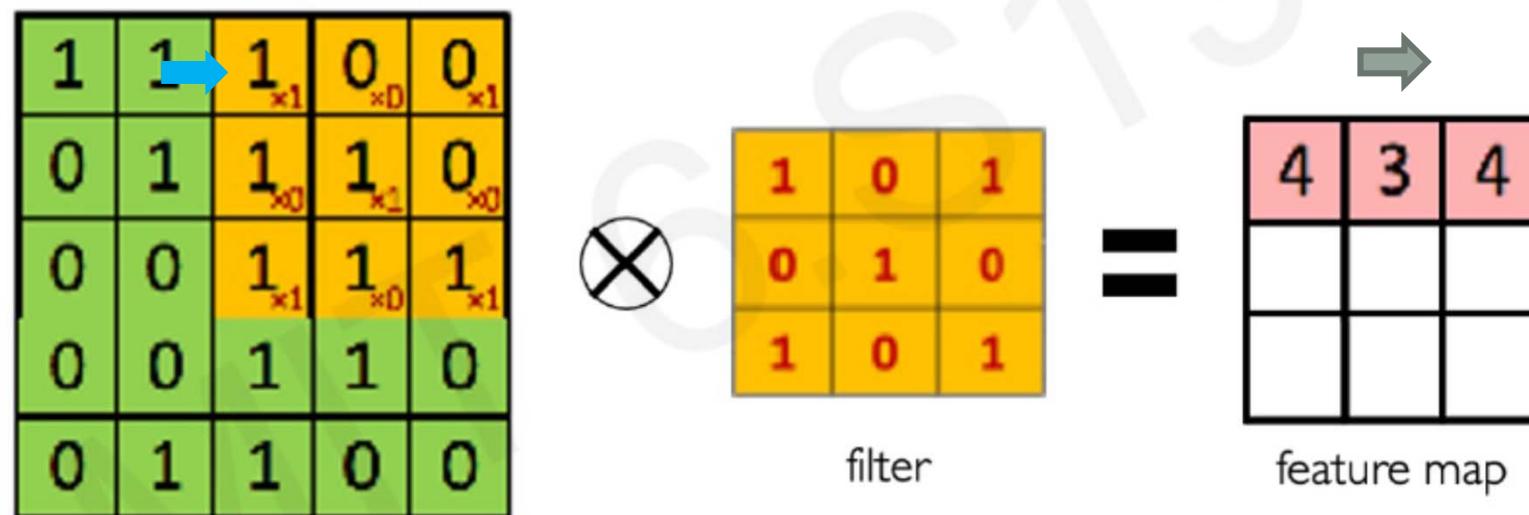
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



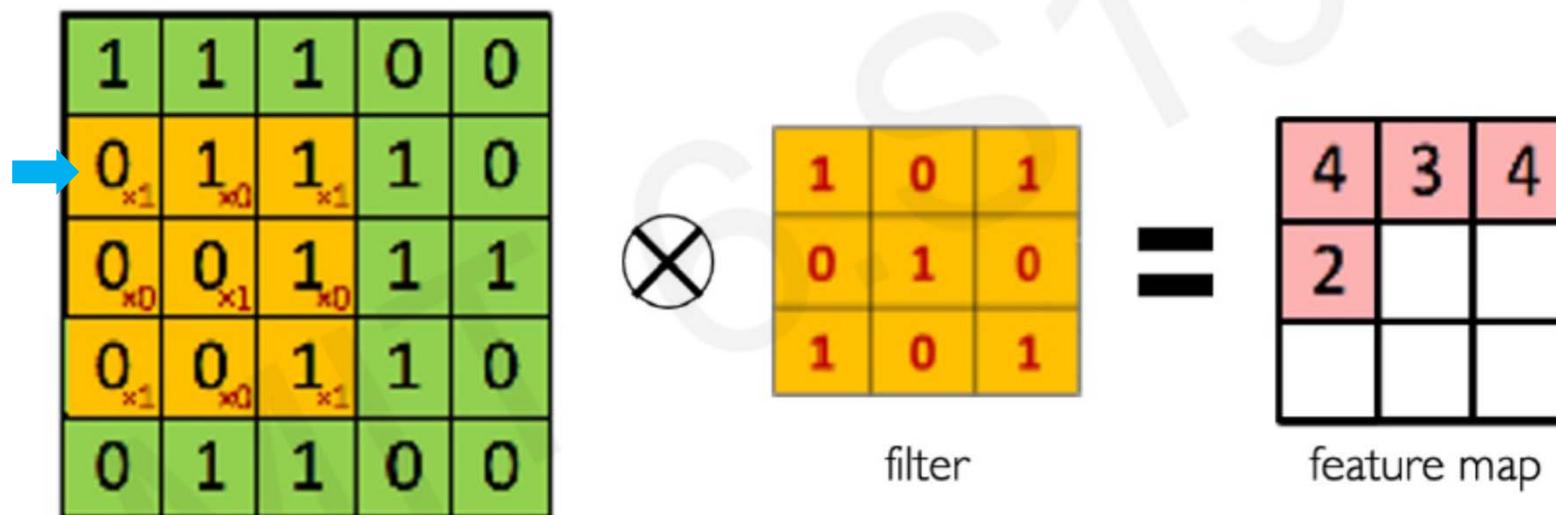
대안 - Convolutional operation

- 입력: 이미지 (5x5)
- 패러미터: 필터 (3x3)
- 출력: feature map



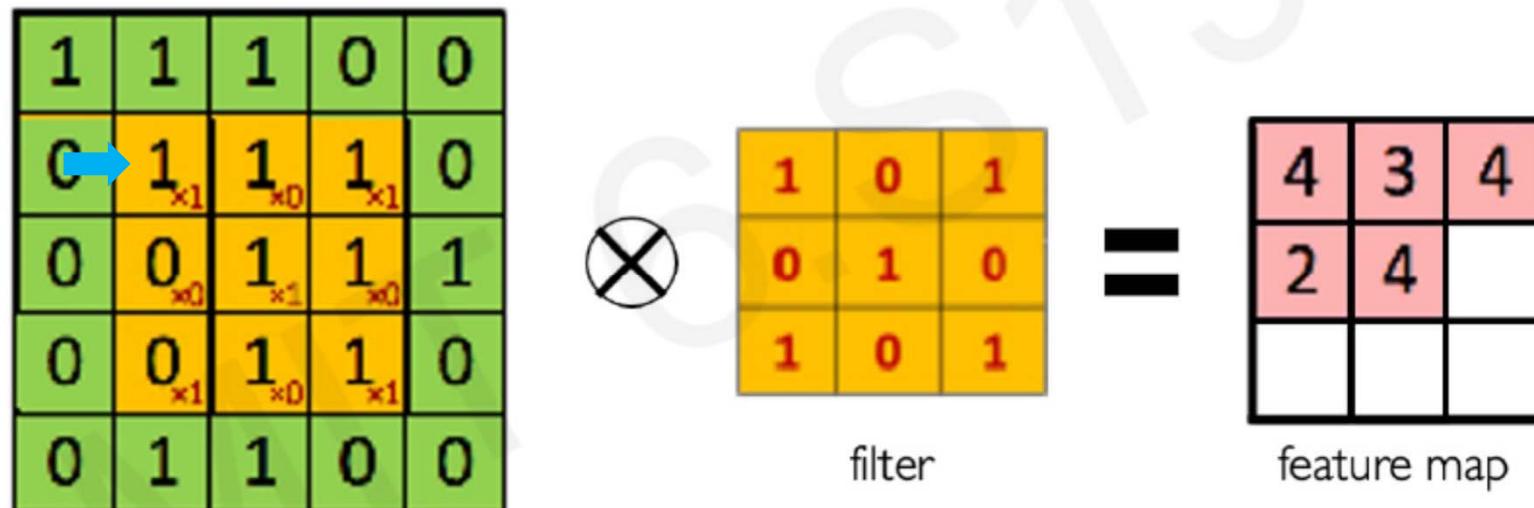
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행
- 첫번째 행에 대한 연산이 모두 이루어졌으면 두번째 행으로 넘어가 가장 왼쪽부터 연산 시작



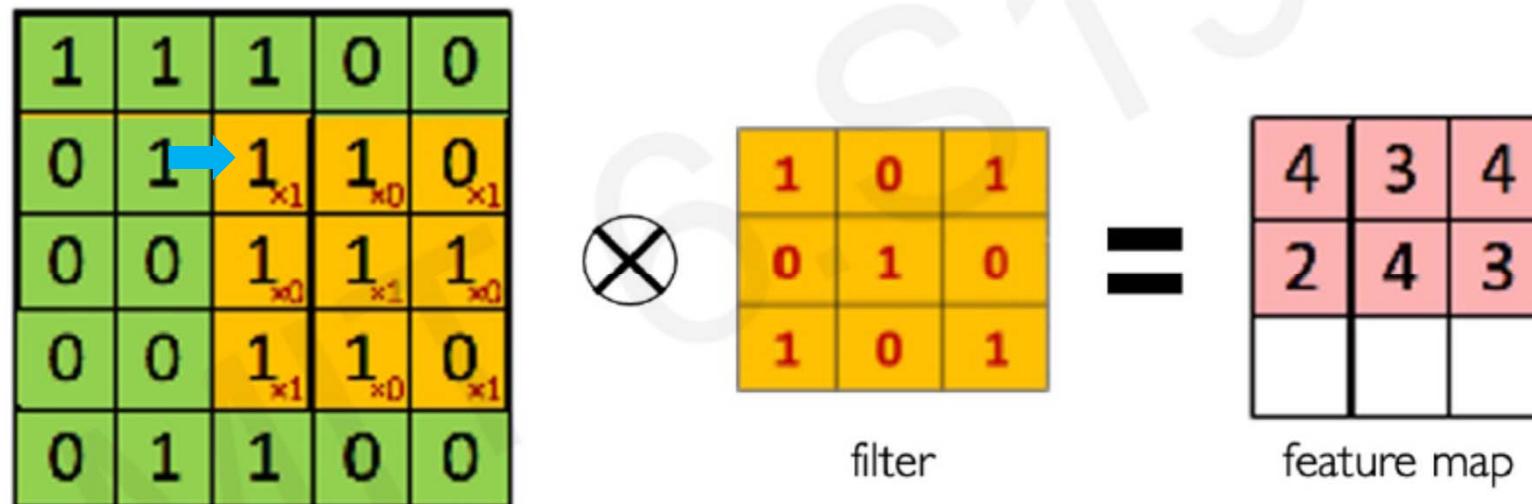
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



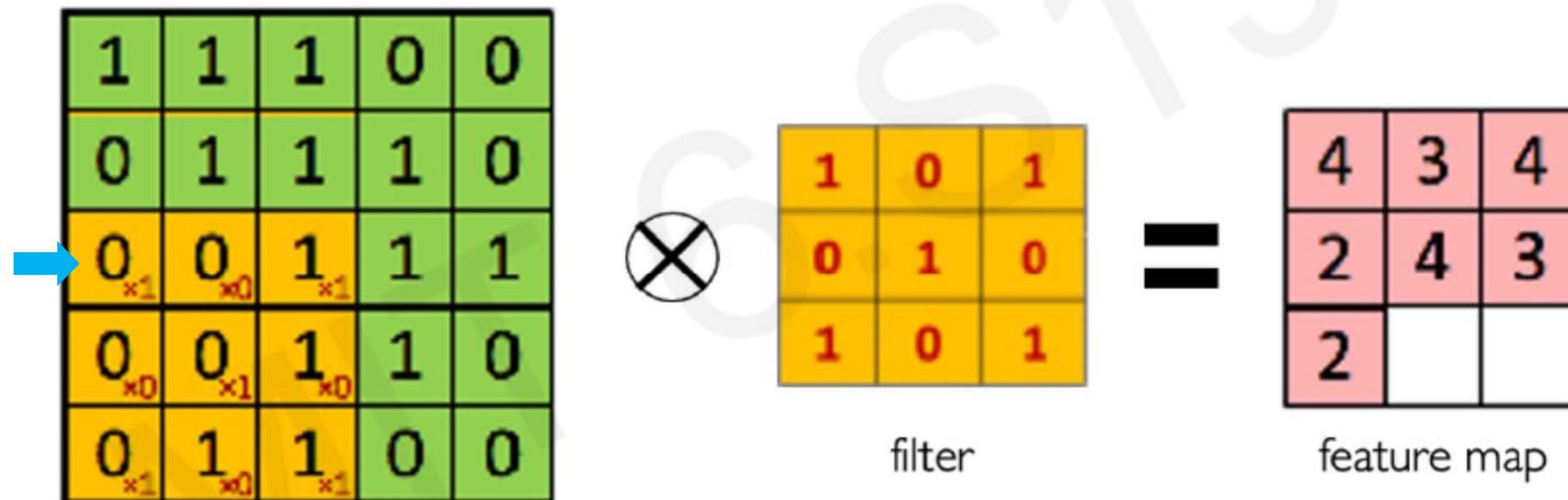
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



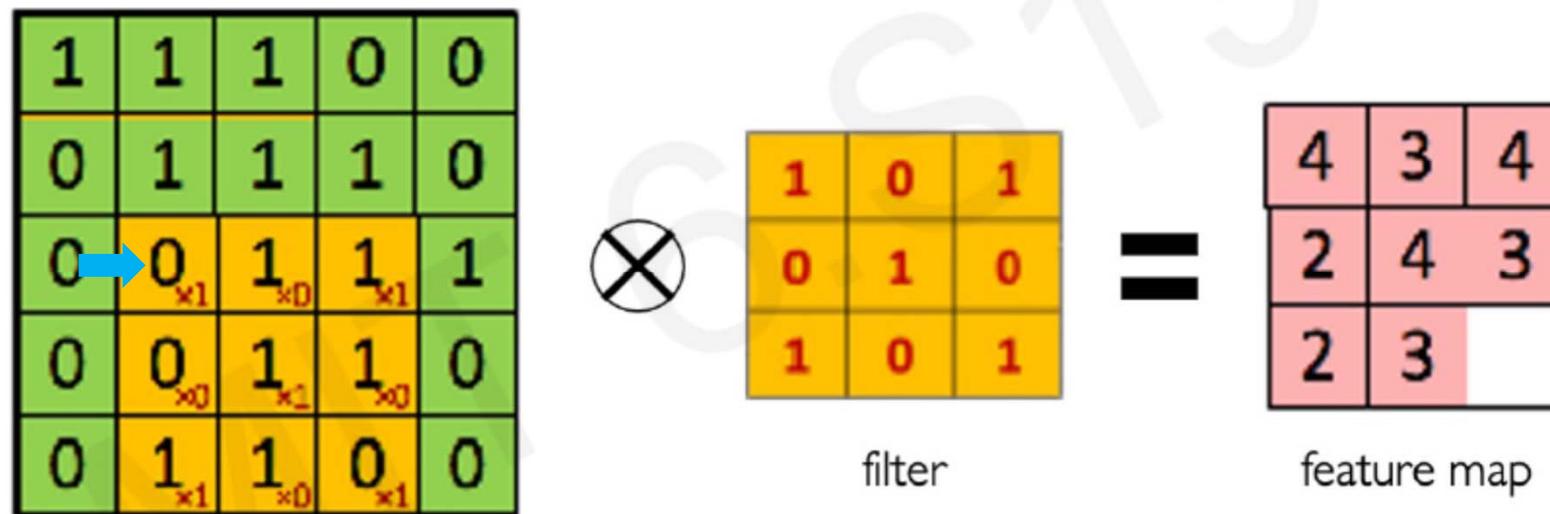
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



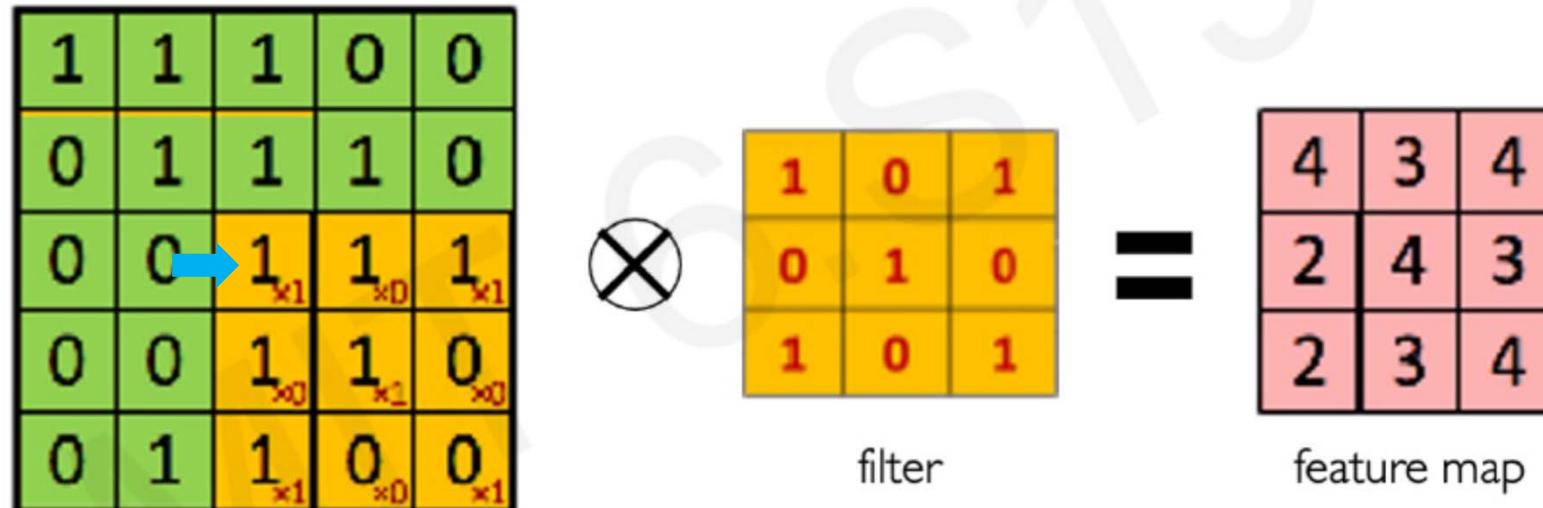
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



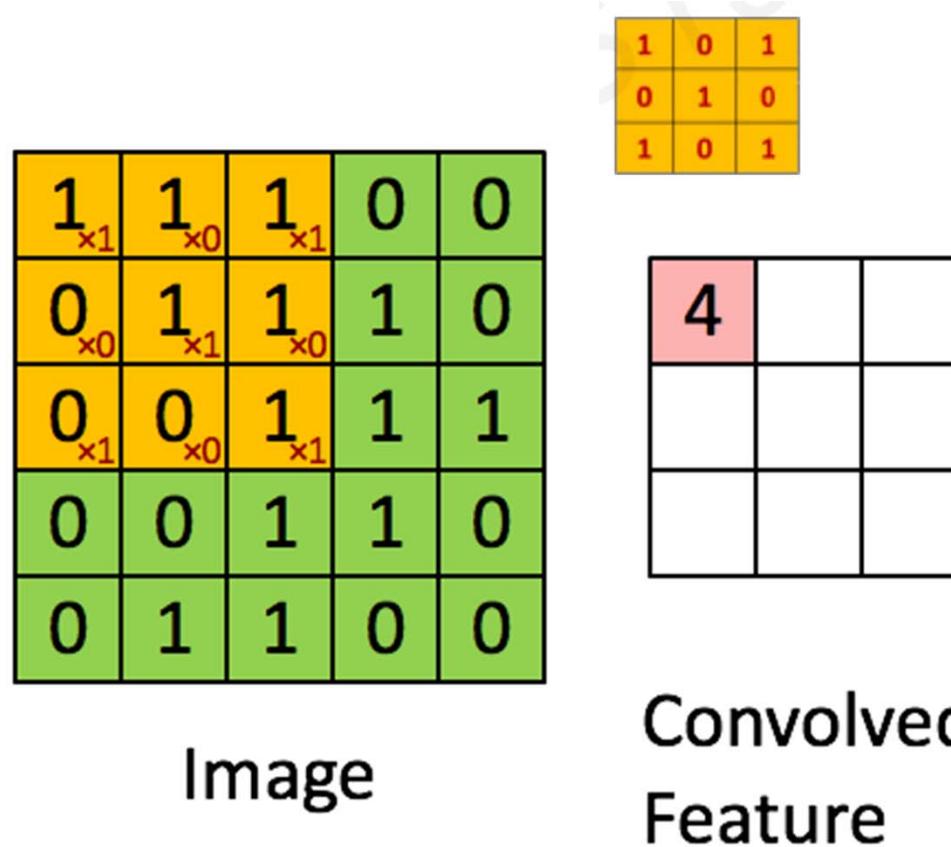
대안 - Convolutional operation

- 커널을 이용해 이미지 전부분을 훑고 지나가면서 (convolve over the input) 스칼라 곱 연산 수행



대안 - Convolutional Operation

- 커널로 이미지의 왼쪽위부터 오른쪽 아래 방향으로 훑으면서 스칼라 곱 연산을 진행



Convolutional Operation 연산의 결과

- Feature Map
 - 이미지와 커널의 콘볼루션 연산결과 얻어진 결과값이 기록된 2차원 형태의 데이터

1	1	1	0	0
0	1	1	1	0
0	0	1 _{x1}	1 _{x0}	1 _{x1}
0	0	1 _{x0}	1 _{x1}	0 _{x0}
0	1	1 _{x1}	0 _{x0}	0 _{x1}



1	0	1
0	1	0
1	0	1

=

4	3	4
2	4	3
2	3	4

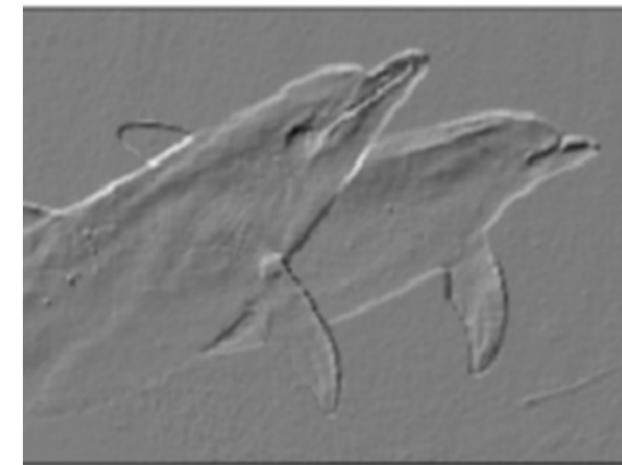
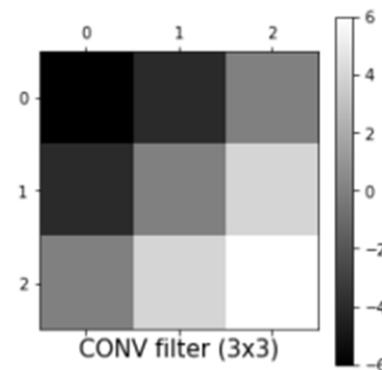
5x5 이미지

3x3 이미지

3x3
feature map

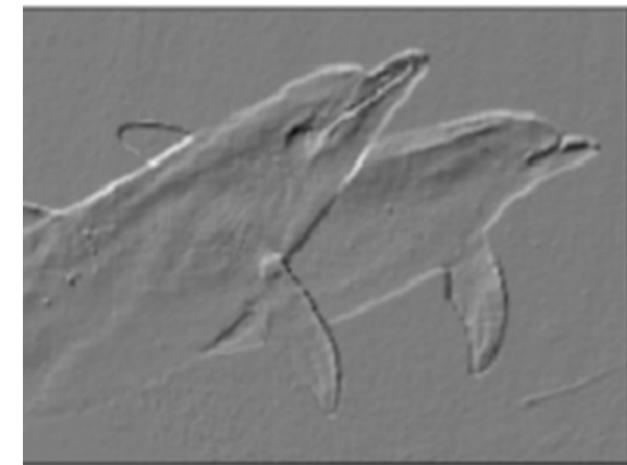
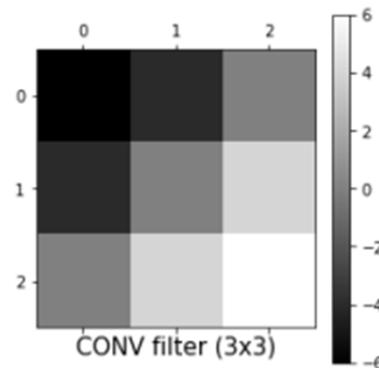
Feature Map의 의미

- 커널과 이미지의 콘볼루션 연산을 거쳐 얻어진 또 하나의 이미지라고 생각해 볼 수 있음
- Feature Map이 큰 값을 가질 수록 이미지의 해당 부분이 특성과 일치



Feature Map의 의미

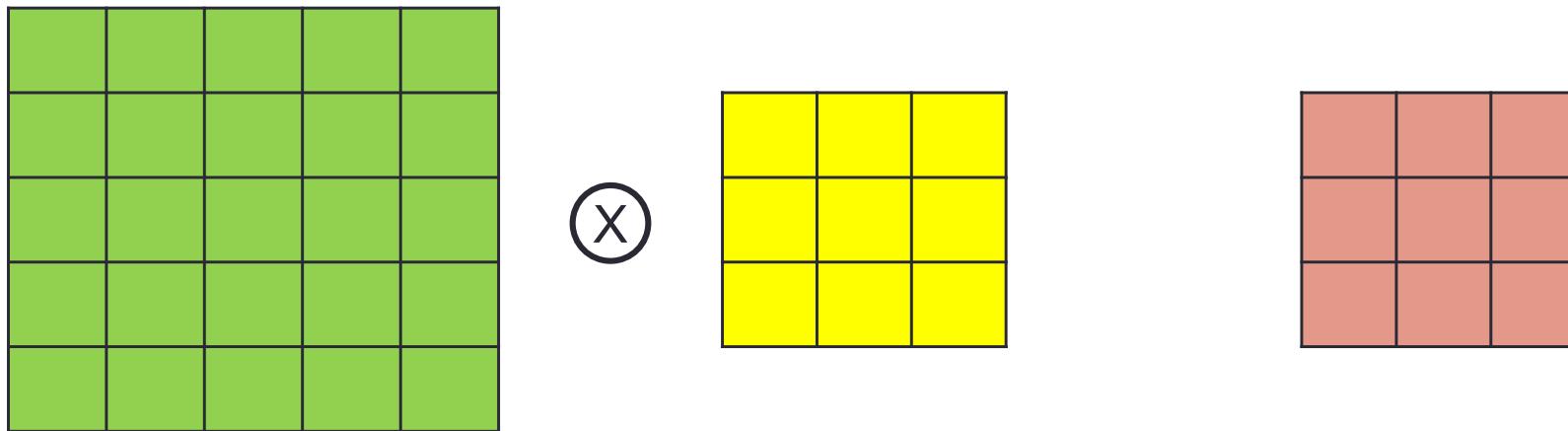
- 커널과 이미지의 콘볼루션 연산을 거쳐 얻어진 또 하나의 이미지라고 생각해 볼 수 있음
- Feature Map이 큰 값을 가질 수록 이미지의 해당 부분이 특성과 일치



- 커널의 패턴: 왼쪽 위가 어둡고, 오른쪽 아래가 밝은 이미지
- Feature map: 일치부분의 값이 밝은 픽셀로 표시됨

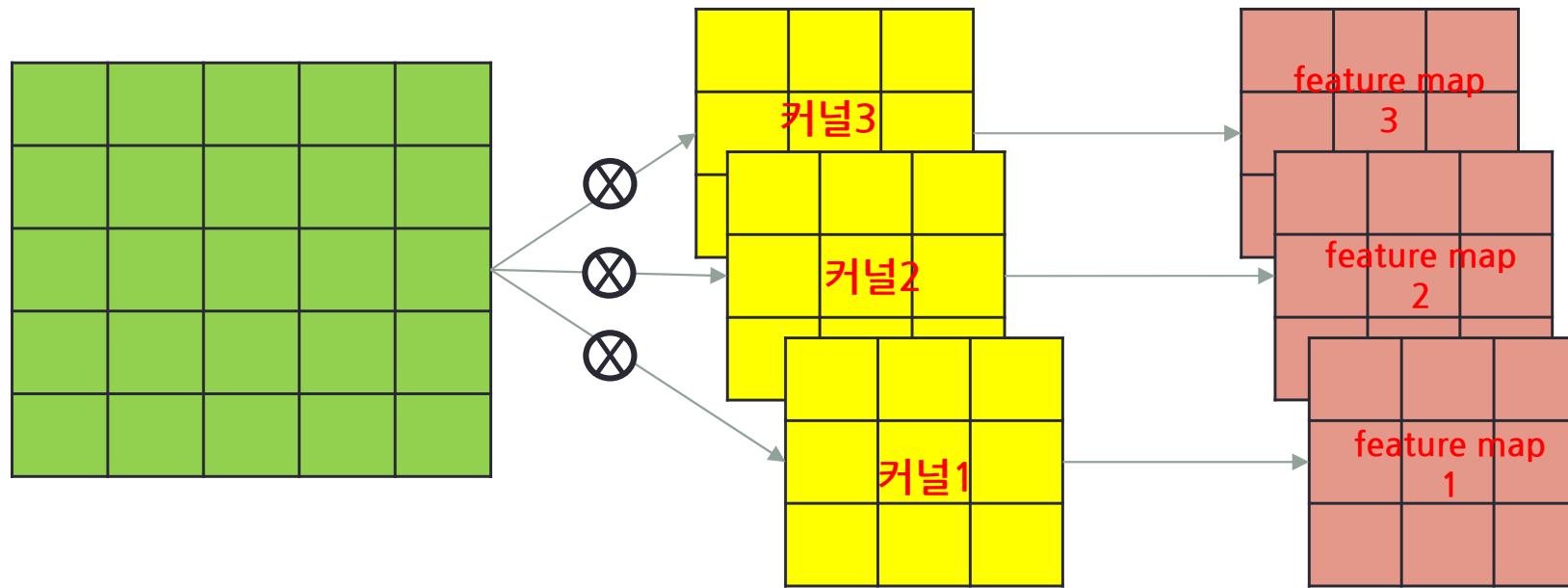
Convolutional Operation with Multiple Kernel

- 이미지: 주어진 것
- 커널: 이미지에서 포착하고자 하는 특성



Convolutional Operation with Multiple Kernel

- 여러 개의 커널을 동일한 이미지에 적용
- 이미지에서 서로 다른 여러 개의 특성을 포착하고 싶을 때 사용



Convolutional Operation with Multiple Kernel

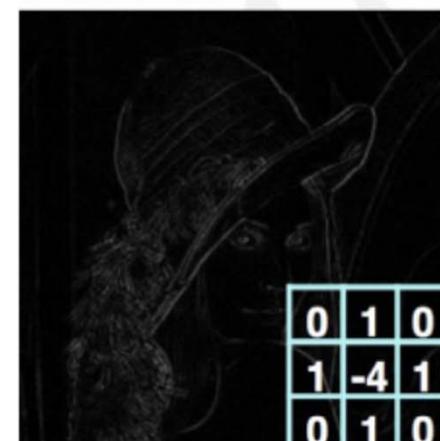
- 여러 개의 커널을 동일한 이미지에 적용
- 이미지에서 서로 다른 여러 개의 특성을 포착하고 싶을 때 사용



Original



Sharpen



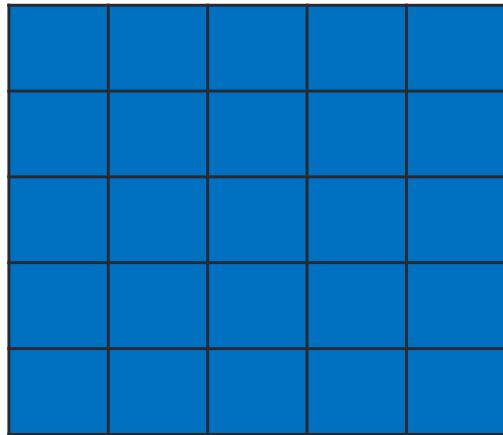
Edge Detect



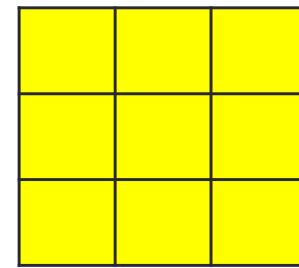
"Strong" Edge
Detect

콘볼루션 연산 패러미터: Kernel Size (커널 크기)

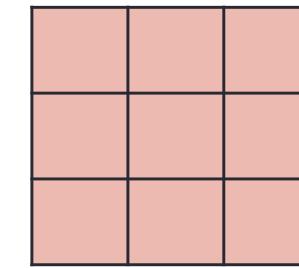
- 커널이 패턴을 얼마나 크게 표현할 것인가?
- 이미지 처리에서 보통 정사각형(가로, 세로 크기 동일)으로 커널 표현
- 커널크기: 3



5x5
이미지



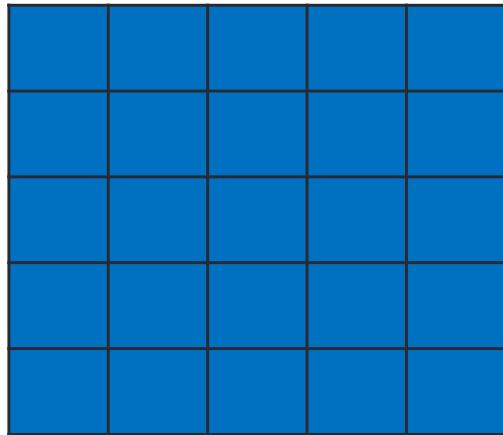
3x3
커널



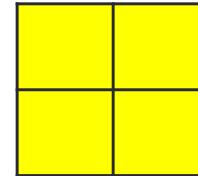
3x3
Feature Map

콘볼루션 연산 패러미터: Kernel Size (커널 크기)

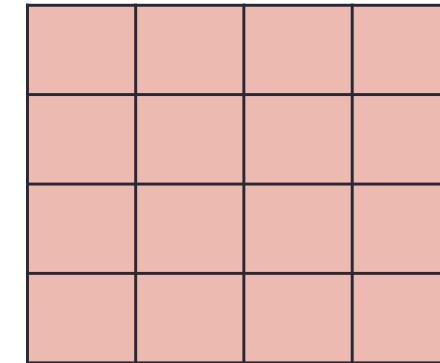
- 커널이 패턴을 얼마나 크게 표현할 것인가?
- 이미지 처리에서 보통 정사각형(가로, 세로 크기 동일)으로 커널 표현
- 커널크기: 2



5x5
이미지



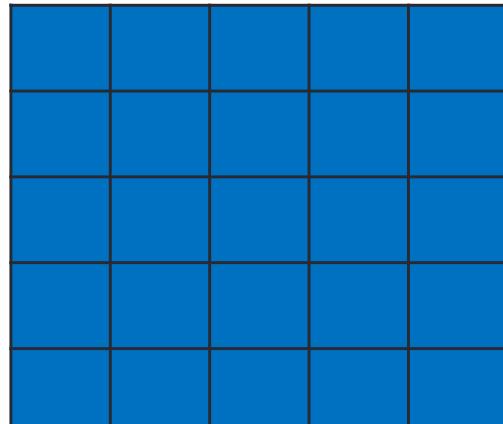
2x2
커널



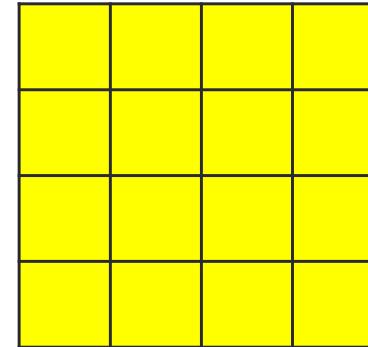
4x4
Feature Map

콘볼루션 연산 패러미터: Kernel Size (커널 크기)

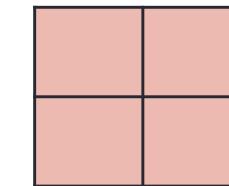
- 커널이 패턴을 얼마나 크게 표현할 것인가?
- 이미지 처리에서 보통 정사각형(가로, 세로 크기 동일)으로 커널 표현
- 커널크기: 4



5x5
이미지



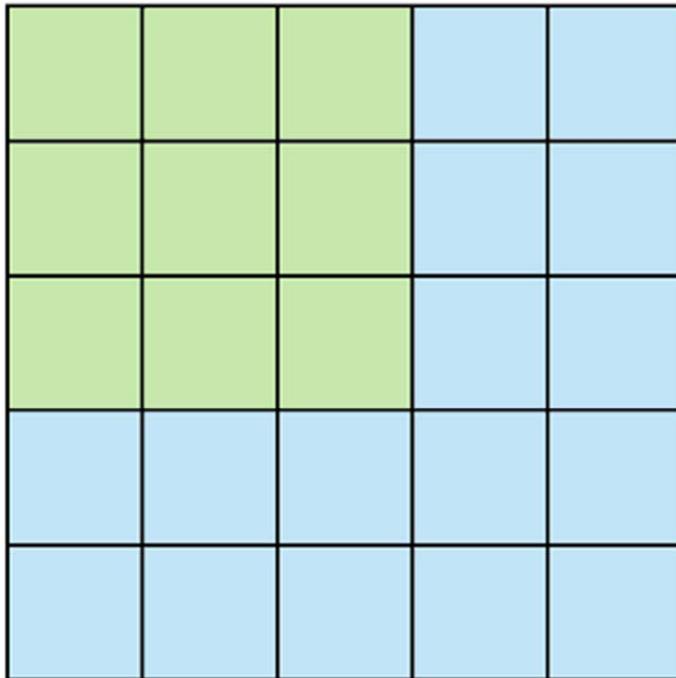
4x4
커널



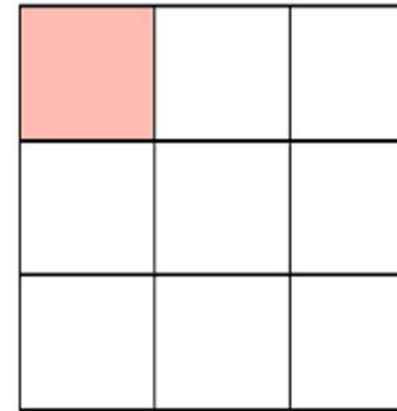
2x2
Feature Map

콘볼루션 연산 패러미터: Stride(보폭)

- 커널이 이미지를 얼마나 자세히 훑으면서 지나갈지 결정하는 수
- Stride 1 → 가로, 세로 이동시 한 픽셀씩 이동



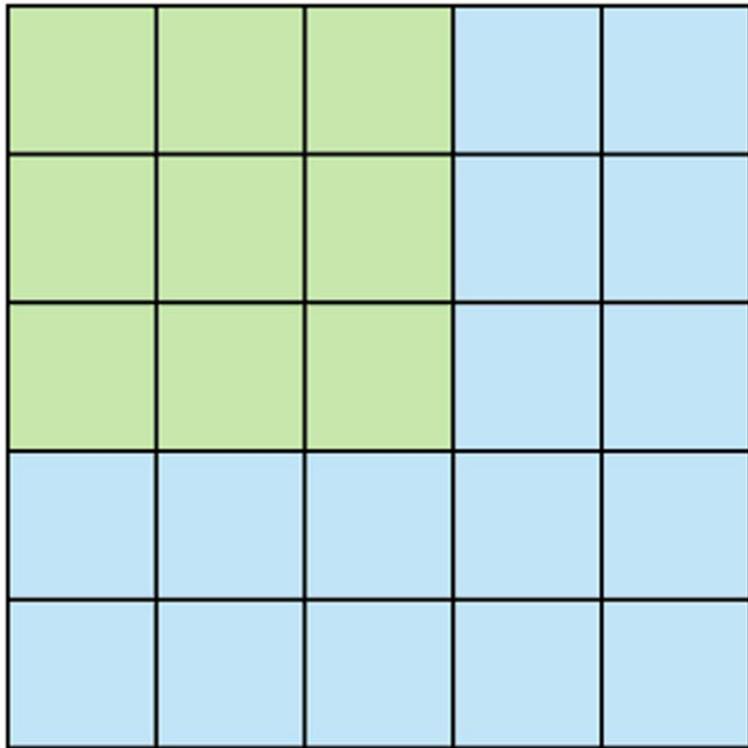
Stride 1



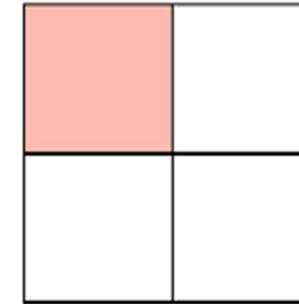
Feature Map

콘볼루션 연산 패러미터: Stride(보폭)

- 커널이 이미지를 얼마나 자세히 훑으면서 지나갈지 결정하는 수
- Stride 2 → 커널이 이미지를 한번에 두 픽셀씩 이동하면서 지나감



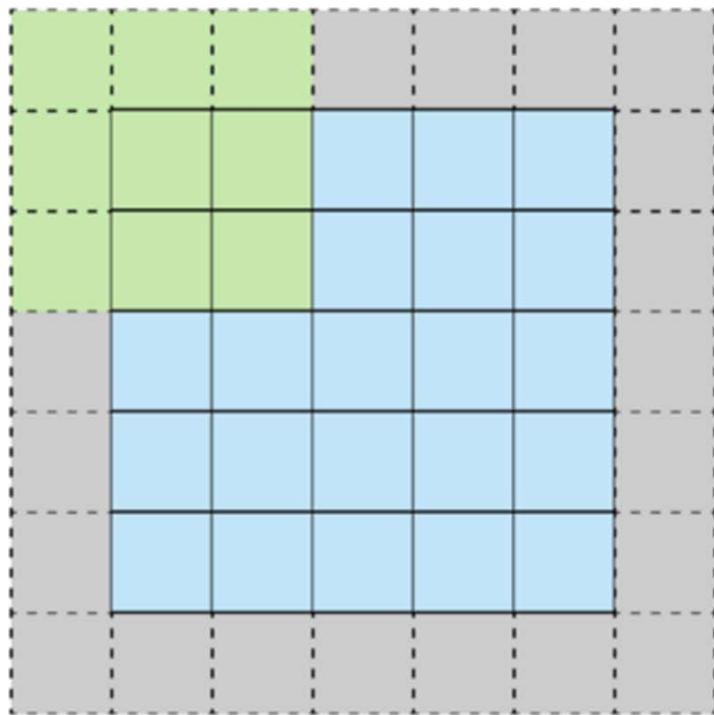
Stride 2



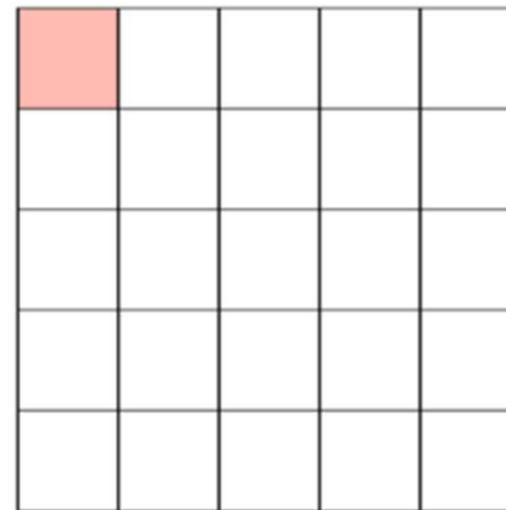
Feature Map

콘볼루션 연산 패러미터: Padding (패딩사이즈)

- 이미지의 테두리를 얼만큼의 두께로 칠 것인가?
- 이미지의 테두리에 의미없는 값(0)을 주고, stride를 1로 설정하면, 원본이미지와 같은 크기의 feature map을 얻을 수 있음



Stride 1 with Padding

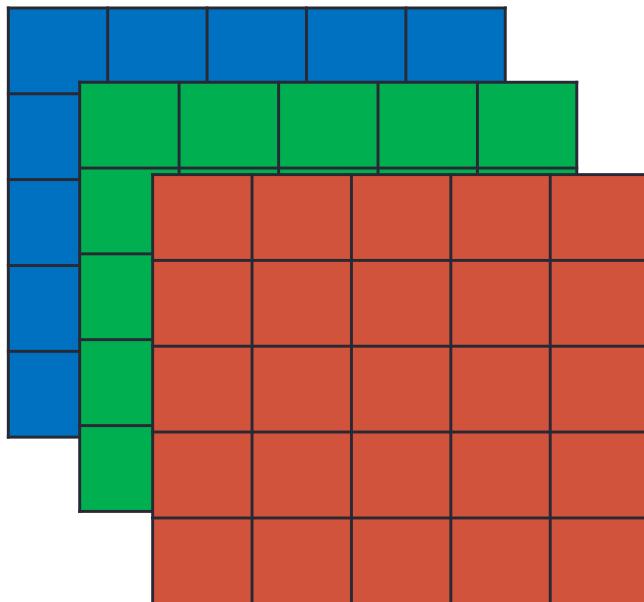


Feature Map

3차원 이미지의 콘볼루션 연산

- 3차원 이미지: 서로 다른 값을 가진 이미지를 겹쳐 하나의 이미지로 표현
- 이때 서로 다른 이미지를 채널(chanel)이라고 함
- 예시: 칼라 사진의 경우 R,G,B로 3개 채널로 구성된 3차원 이미지 (가로x세로x채널)

이미지

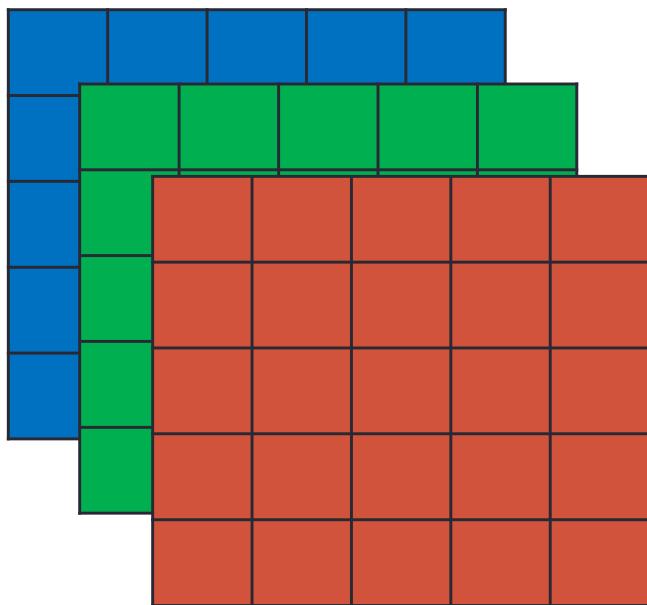


5x5x3

3차원 이미지의 콘볼루션 연산

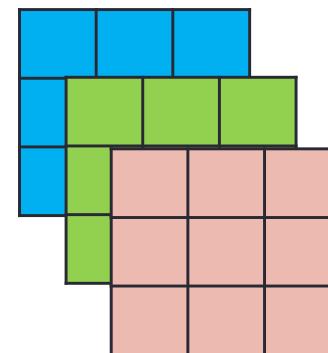
- 채널의 수와 동일하게 커널도 다수의 채널을 갖도록 세팅함 (3차원)
- 예시: 칼라 사진의 경우 R,G,B로 3개 채널로 구성된 3차원 이미지 (가로x세로x채널)
- 커널도 R,G,B 각각에 대응하는 채널로 구성됨

이미지



5x5x3

커널

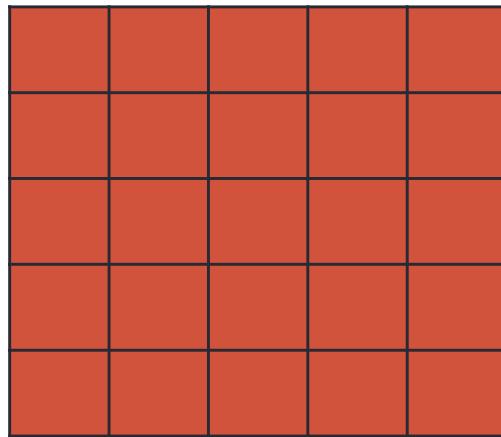


3x3x3

3차원 이미지의 콘볼루션 연산

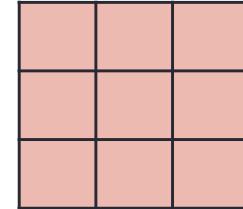
- 각 채널에 해당하는 콘볼루션 연산 진행
- R 채널에 해당하는 콘볼루션 연산

이미지

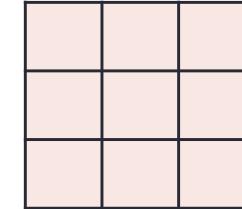


5x5x1
(Red)

커널



3x3x1



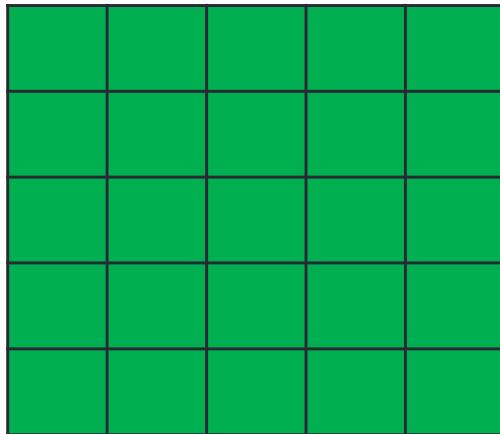
3x3x1



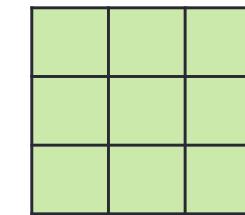
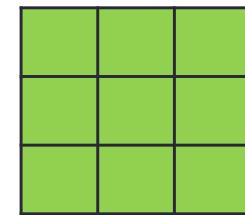
3차원 이미지의 콘볼루션 연산

- 각 채널에 해당하는 콘볼루션 연산 진행
- G 채널에 해당하는 콘볼루션 연산

이미지



커널



$5 \times 5 \times 1$
(Green)

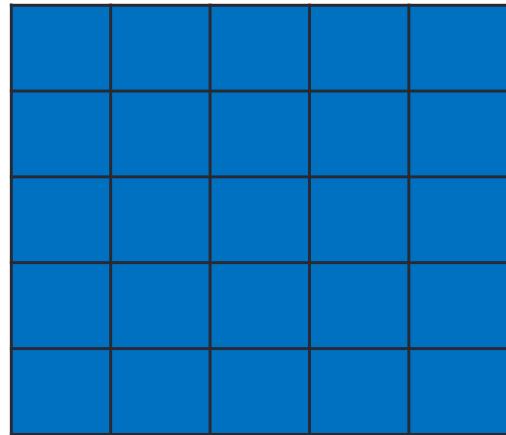
$3 \times 3 \times 1$

$3 \times 3 \times 1$

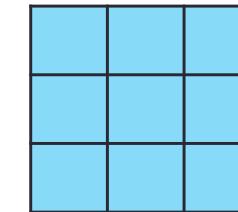
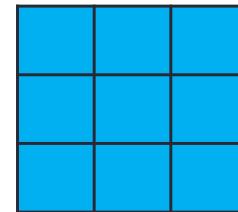
3차원 이미지의 콘볼루션 연산

- 각 채널에 해당하는 콘볼루션 연산 진행
- B 채널에 해당하는 콘볼루션 연산

이미지



커널



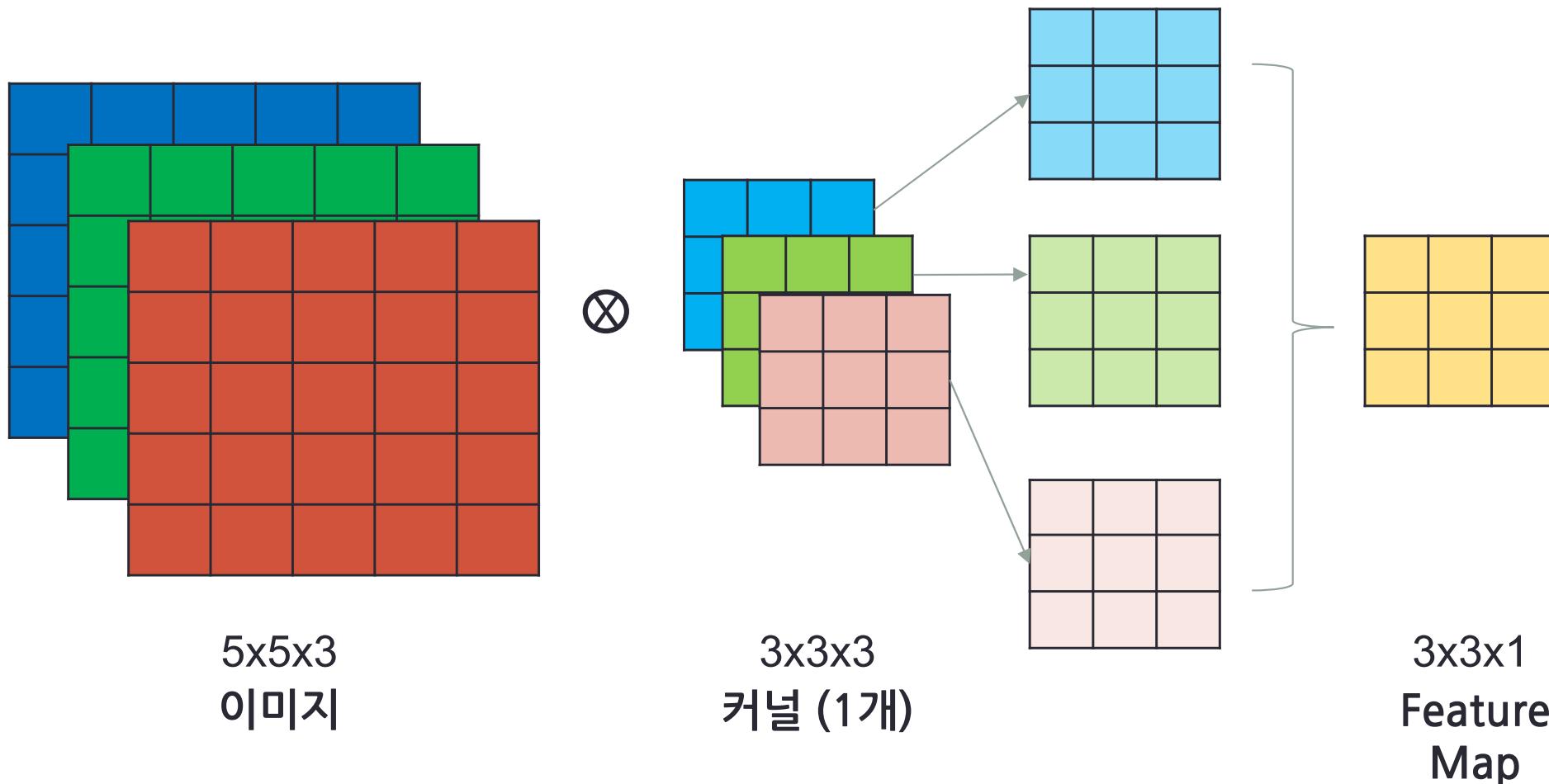
5x5x1
(Blue)

3x3x1

3x3x1

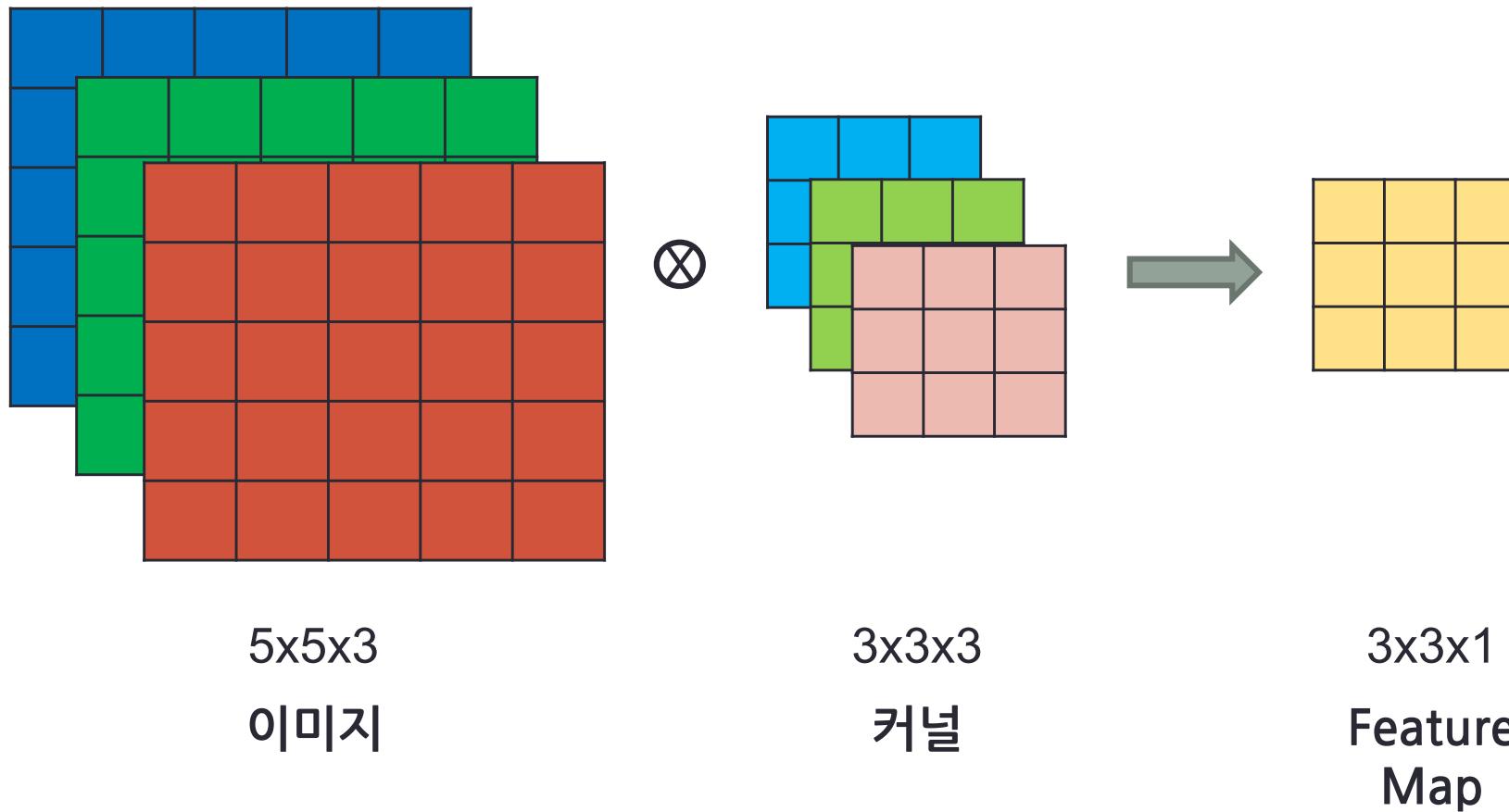
3차원 이미지의 콘볼루션 연산

- 각 채널에서 얻어진 feature map을 모두 합하여 하나의 feature map 계산



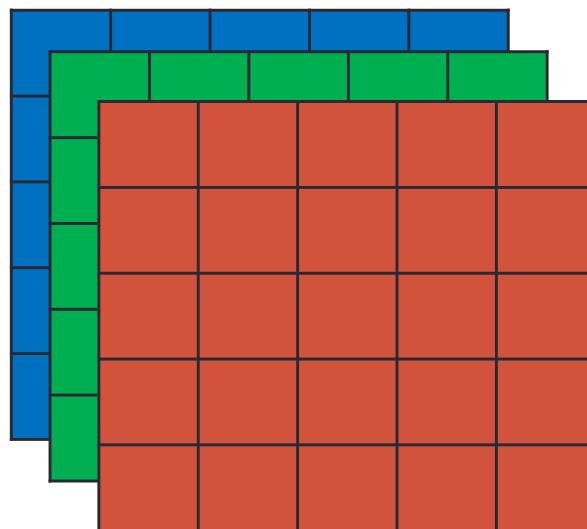
색깔있는 이미지의 콘볼루션 연산

- 각 채널에서 얻어진 feature map을 모두 합하여 하나의 feature map 계산

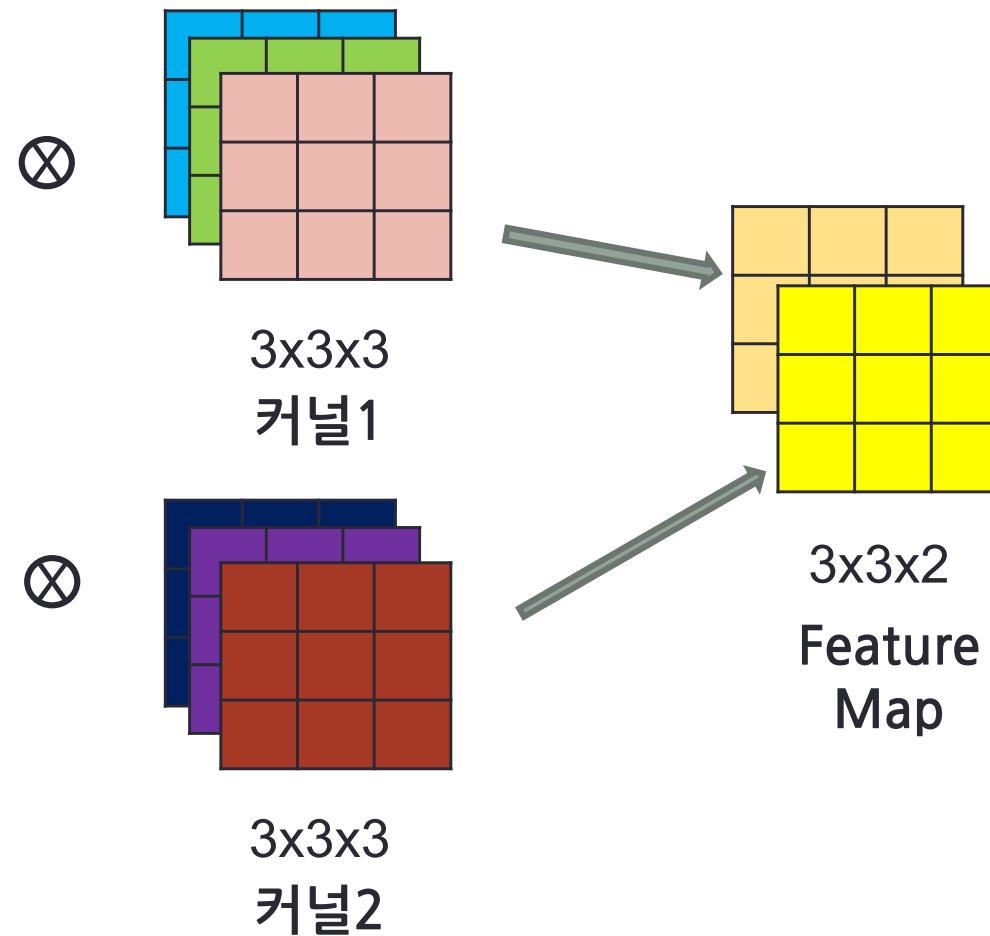


색깔있는 이미지의 콘볼루션 연산

- 여러 개의 커널에서도 동일한 연산



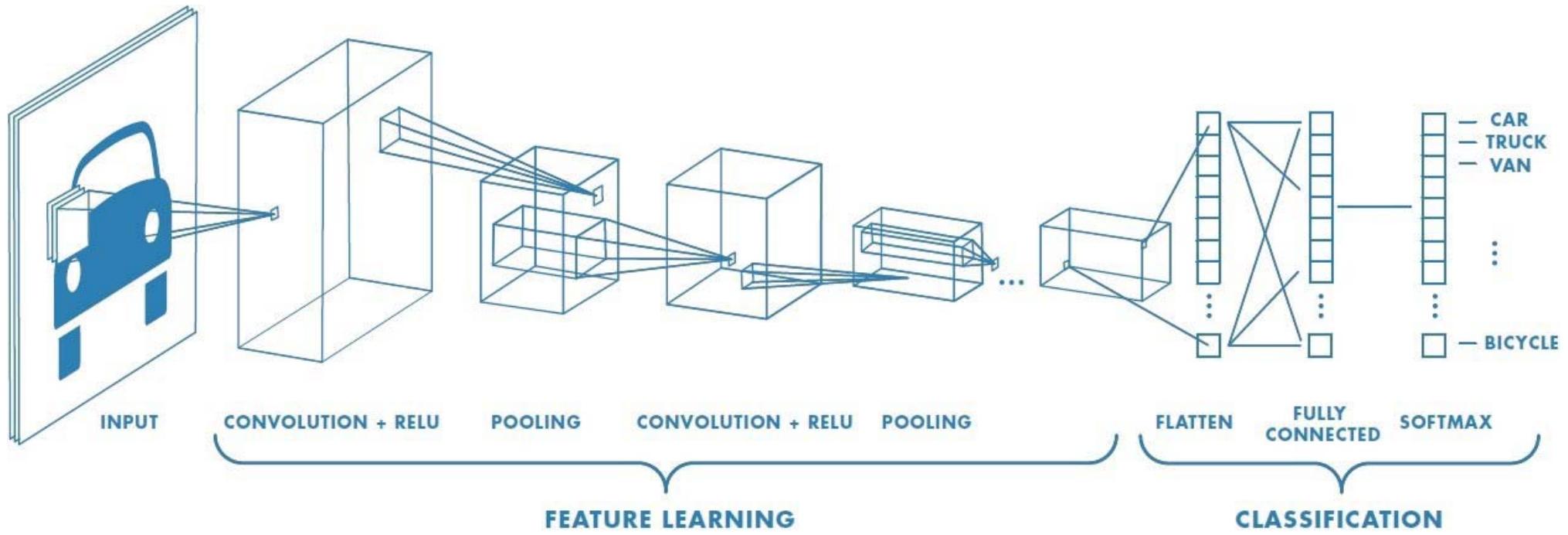
5x5x3
이미지



02. 딥러닝과 컴퓨터 비전

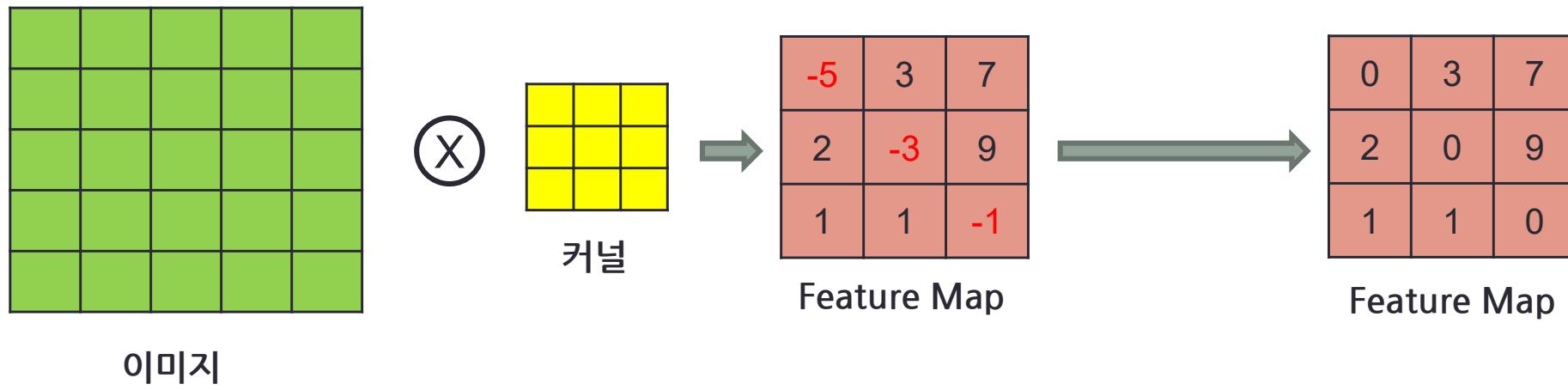
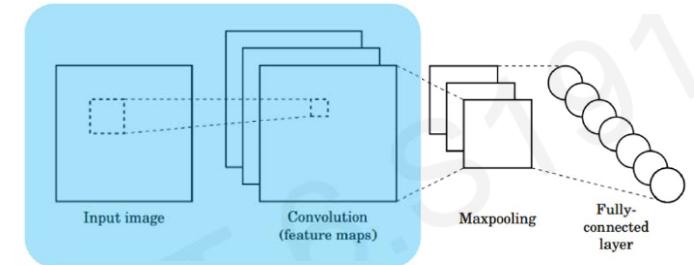
콘볼루션 인공신경망
Convolutional Neural Network

정리: CNN의 일반적인 구조



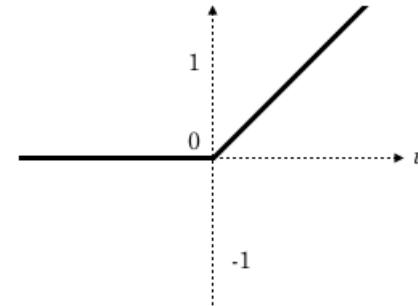
RELU Activation

- 콘볼루션 연산으로 추출한 feature map에서 양의 값만 활성화시키는 layer 추가



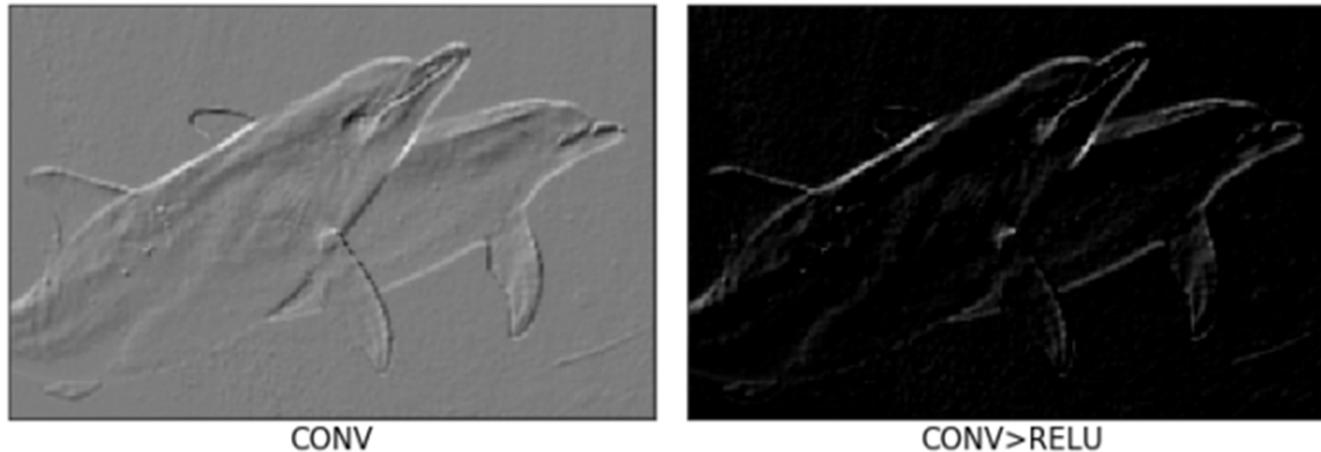
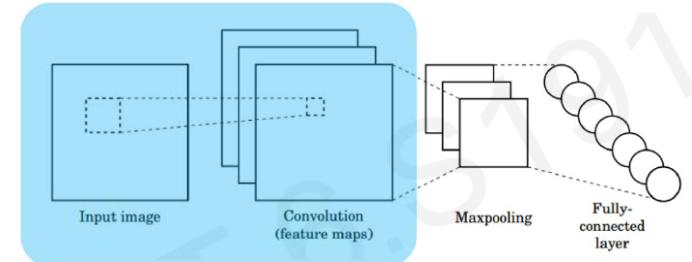
RELU activation

$$f(u) = \max(0, u)$$



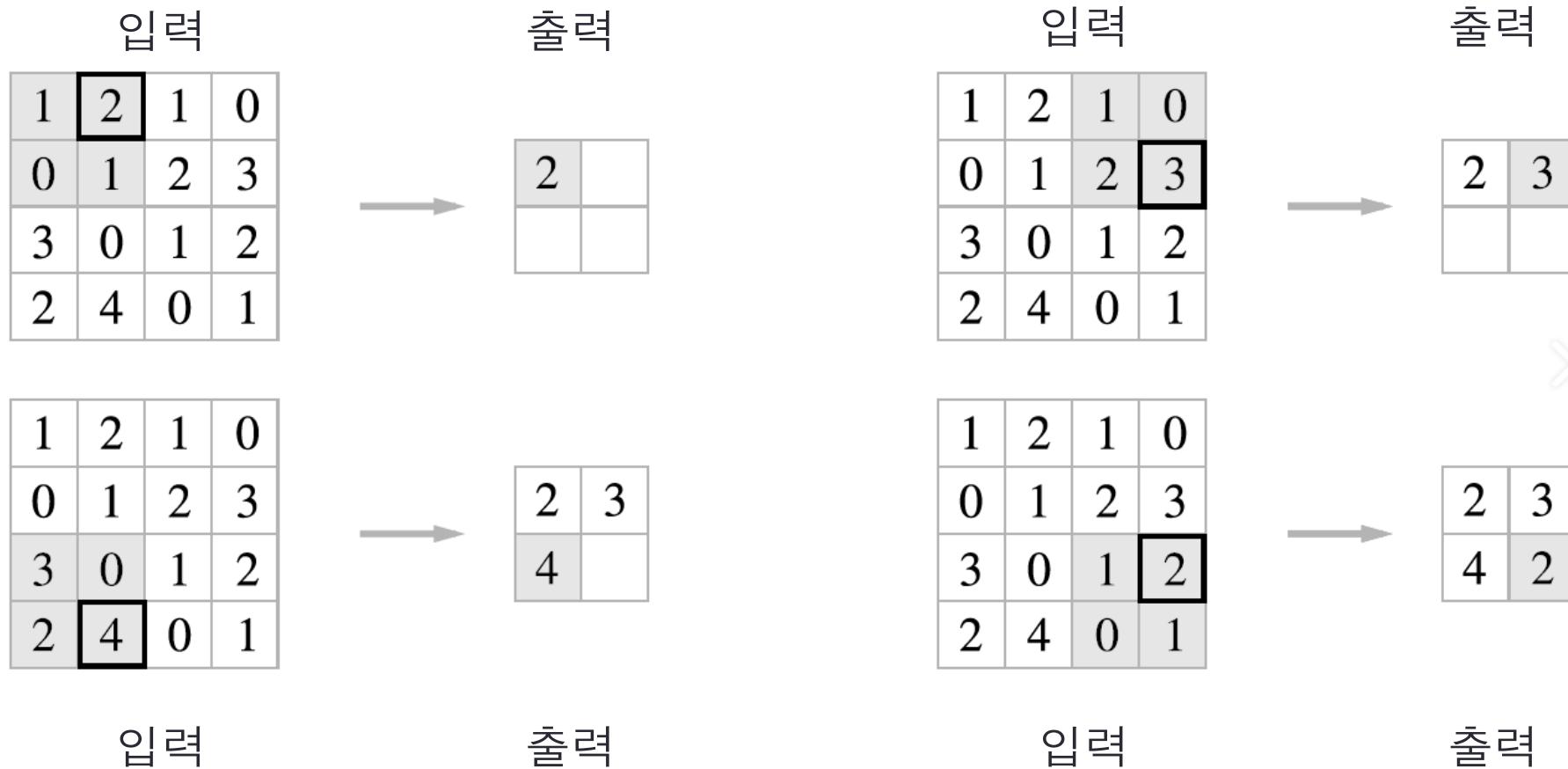
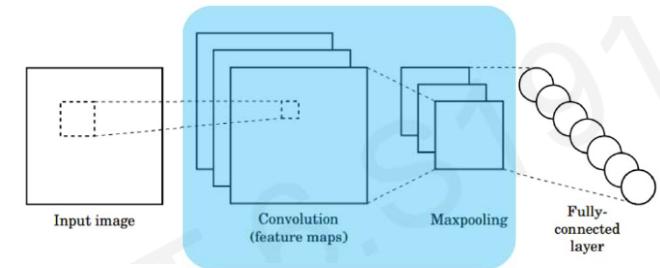
RELU Activation의 의미

- 콘볼루션 연산으로 추출한 feature map에서 양의 값만 활성화시키는 layer 추가
- feature map에서 두드러지는 특징만 더 강조할 수 있게 됨



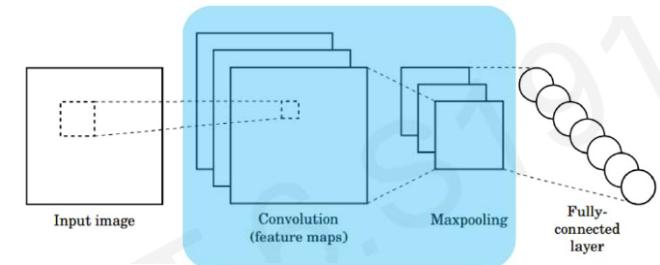
Pooling Layer

- 입력 이미지의 크기를 줄이는 연산 수행



Pooling Layer

- 입력 이미지의 크기를 줄이는 연산 수행
- 패러미터
 - 사이즈 : 2
 - 보폭(Stride) : 2
 - Max or Average : Max (최대값)



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

출력			
2			



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

출력			
2	3		



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

출력			
2	3		



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

출력			
2	3		



입력

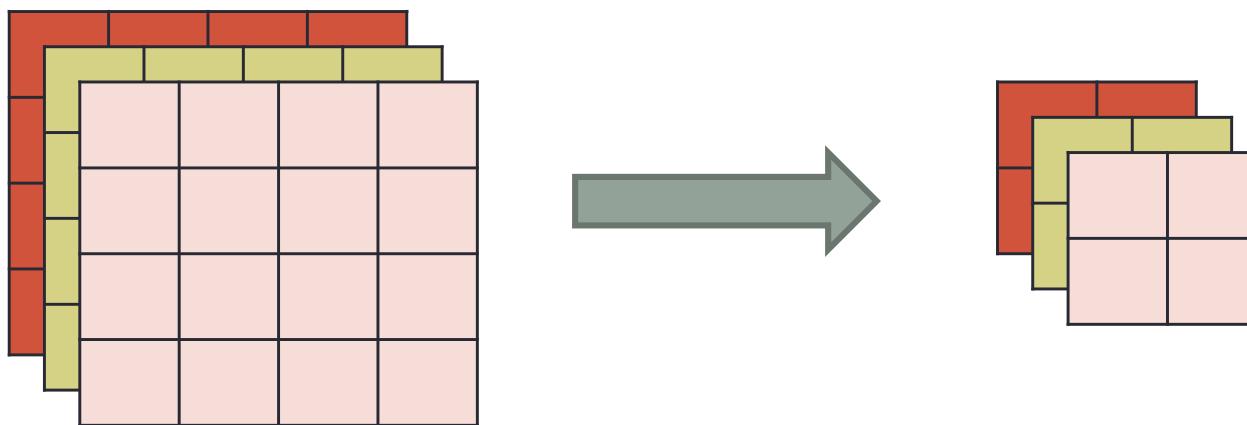
출력

입력

출력

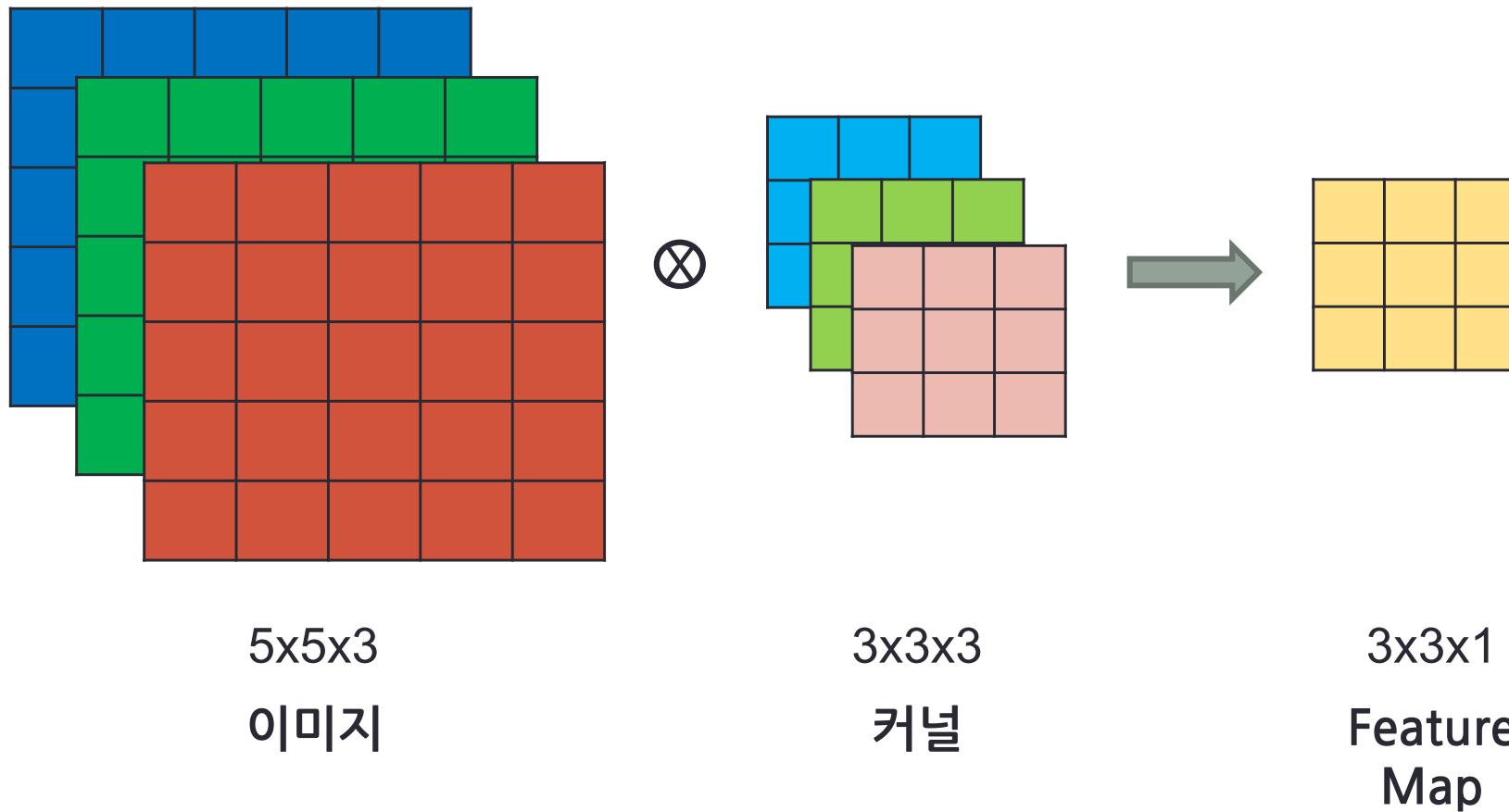
채널이 여러개일때 풀링

- 각 채널을 구성하는 이미지에 대하여 풀링 연산 수행

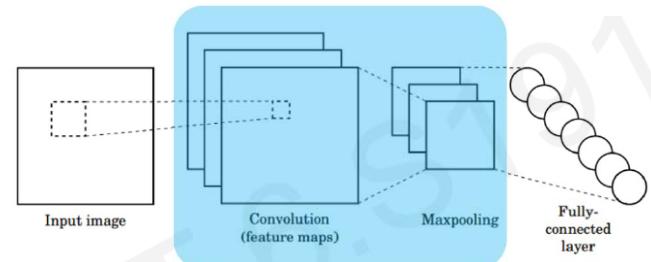


색깔있는 이미지의 콘볼루션 연산

- 각 채널에서 얻어진 feature map을 모두 합하여 하나의 feature map 계산



왜 Pooling을 쓰는가?



- 이미지 사이즈를 줄인다 하더라도 특성은 크게 변하지 않는다
- 연산에 사용되는 이미지를 줄여 연산 효율성을 높일 수 있다
- 축약과정을 통해 Variance를 줄이는 효과



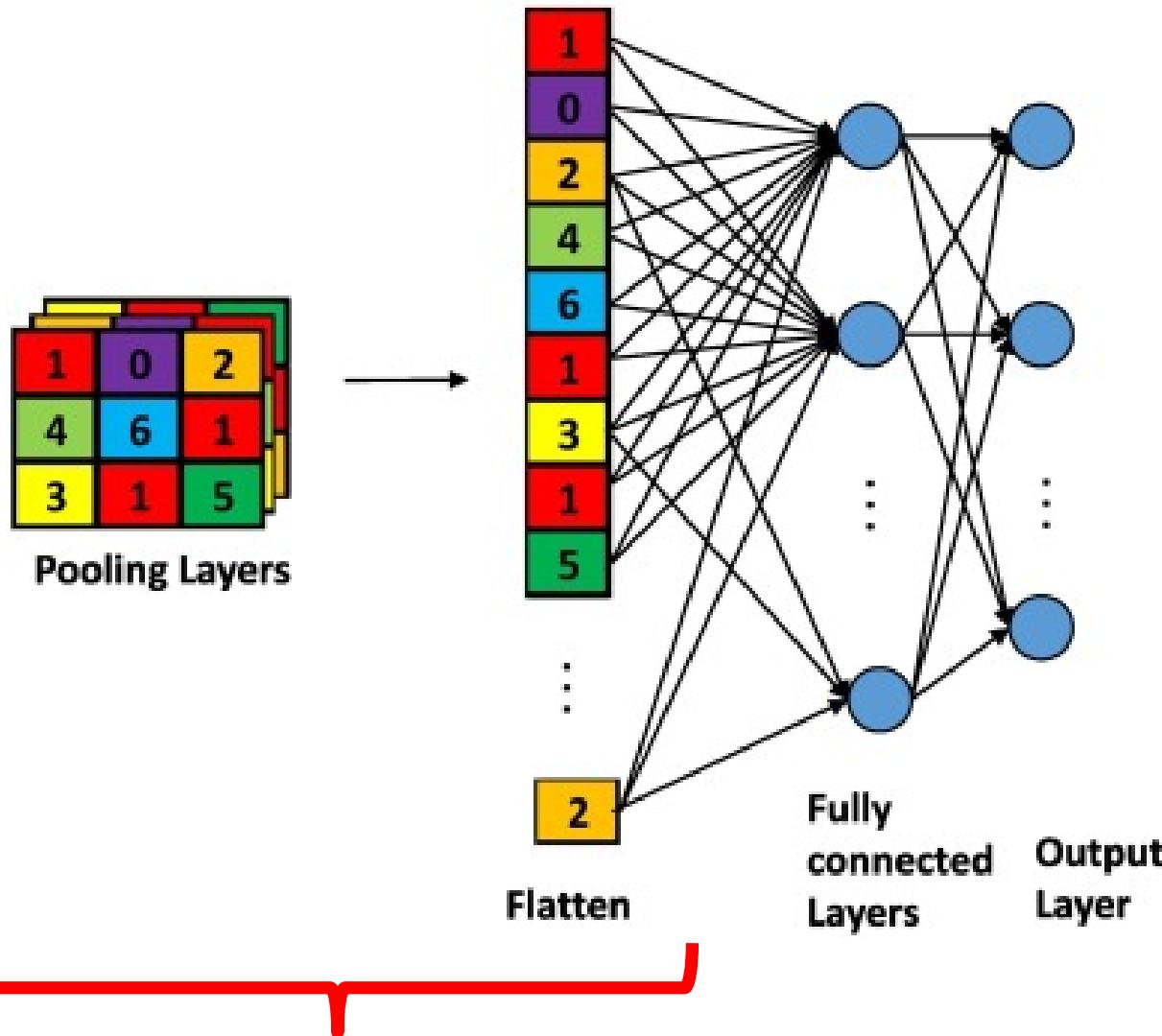
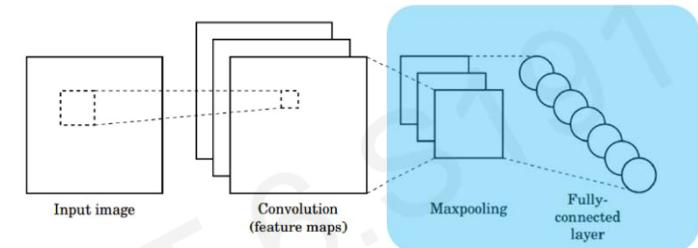
Subsampling



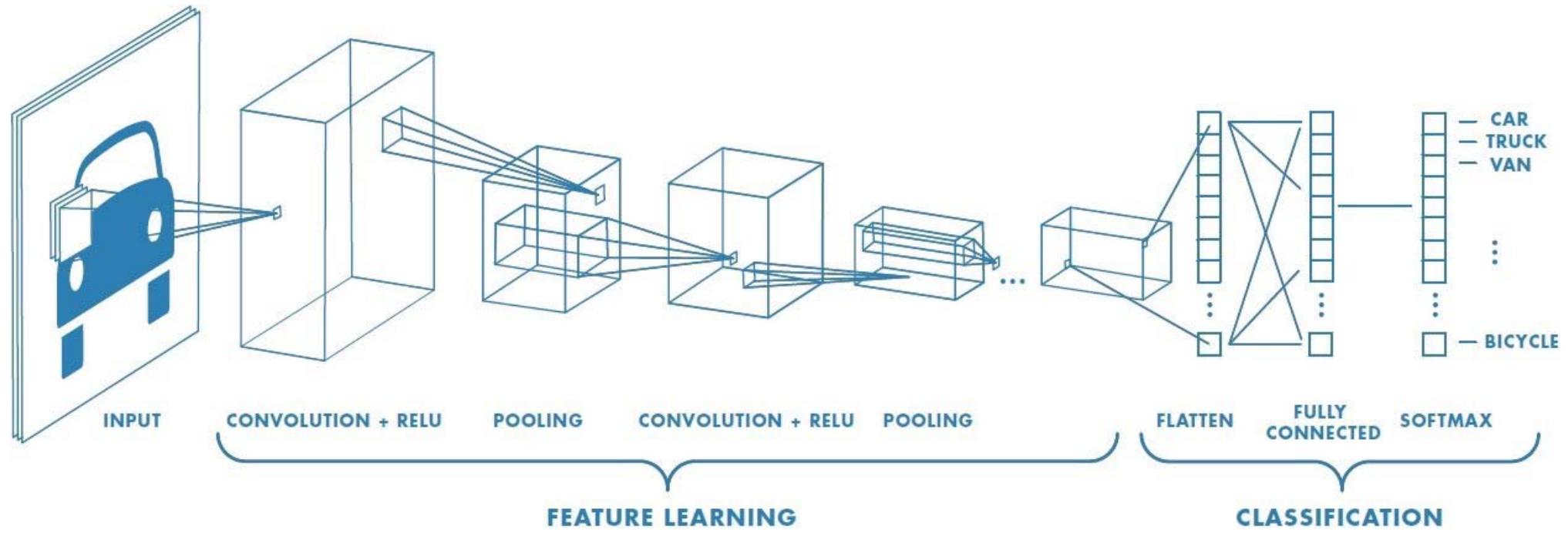
bird

Fully Connected Layer

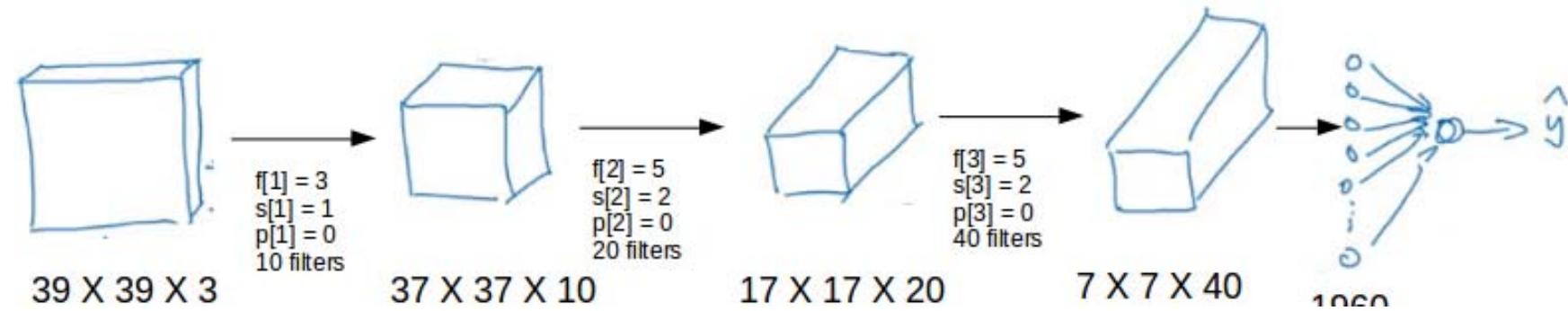
- CNN의 마지막 Layer
- 모든 데이터를 펼쳐서 (Flatten) 1차원으로 만들고 분류



정리: CNN의 일반적인 구조



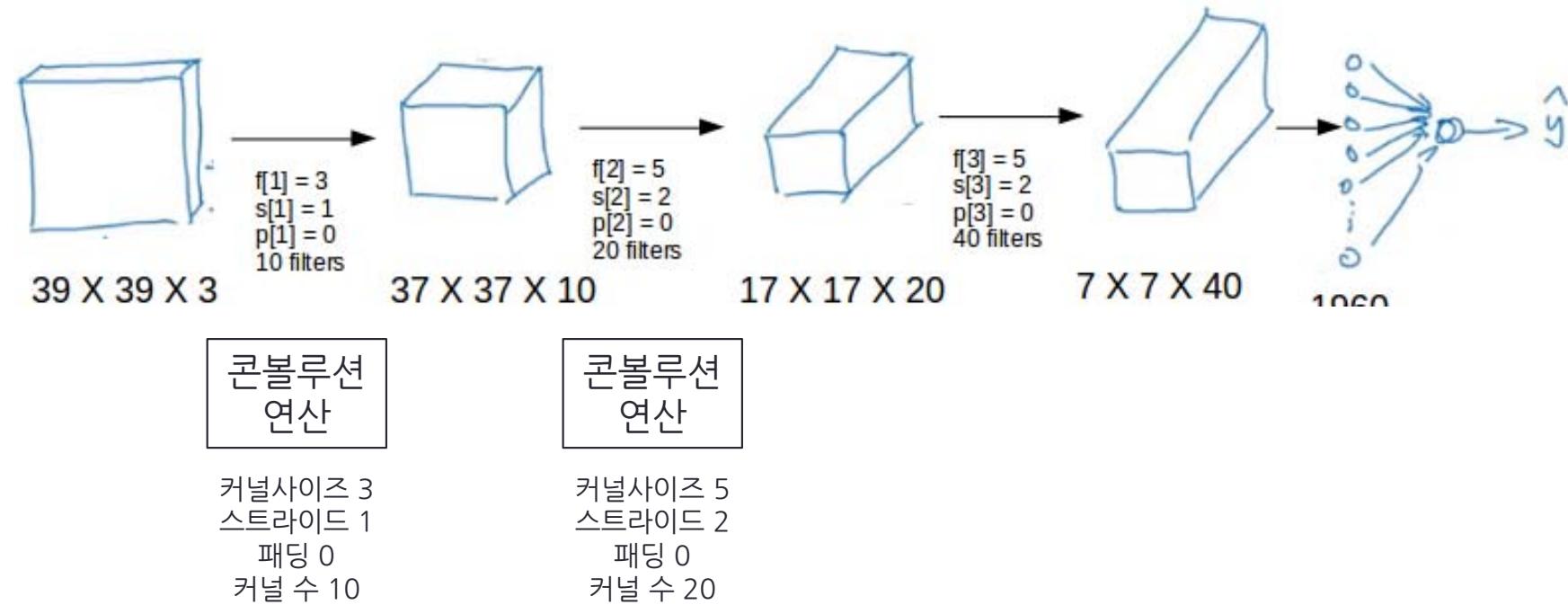
CNN 연산연습



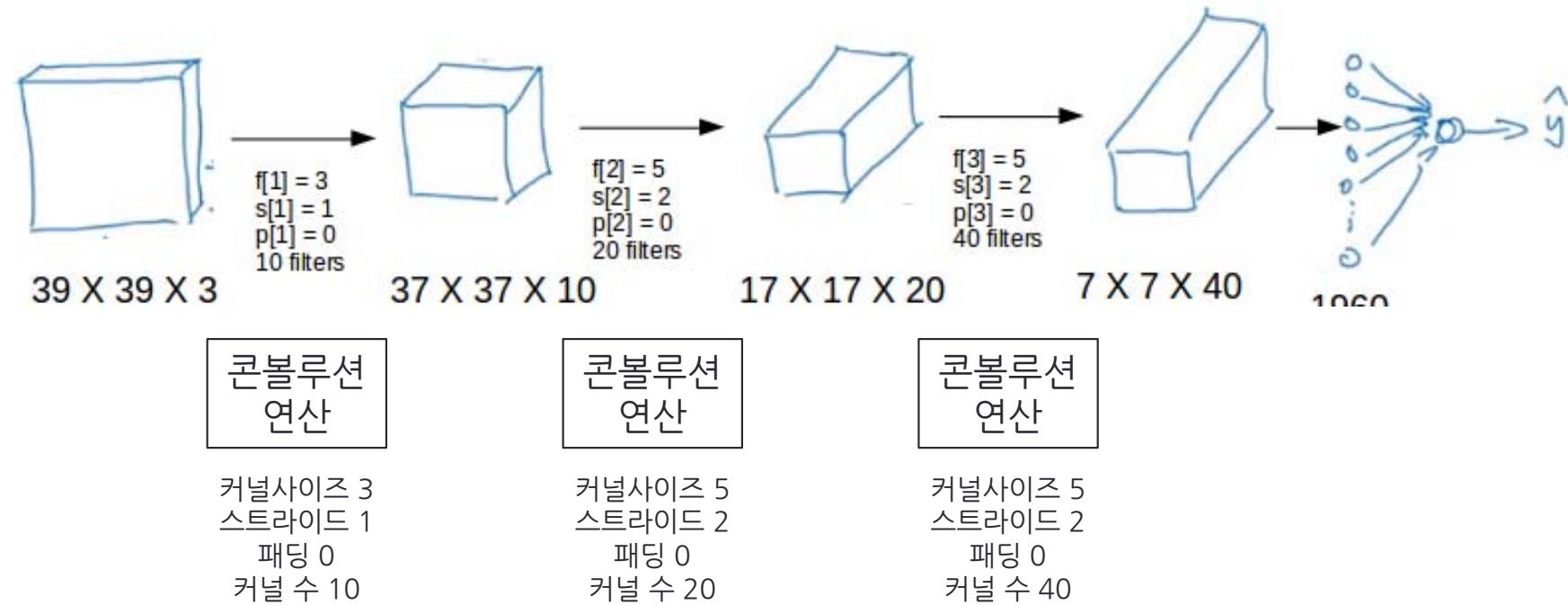
콘볼루션
연산

커널사이즈 3
스트라이드 1
패딩 0
커널 수 10

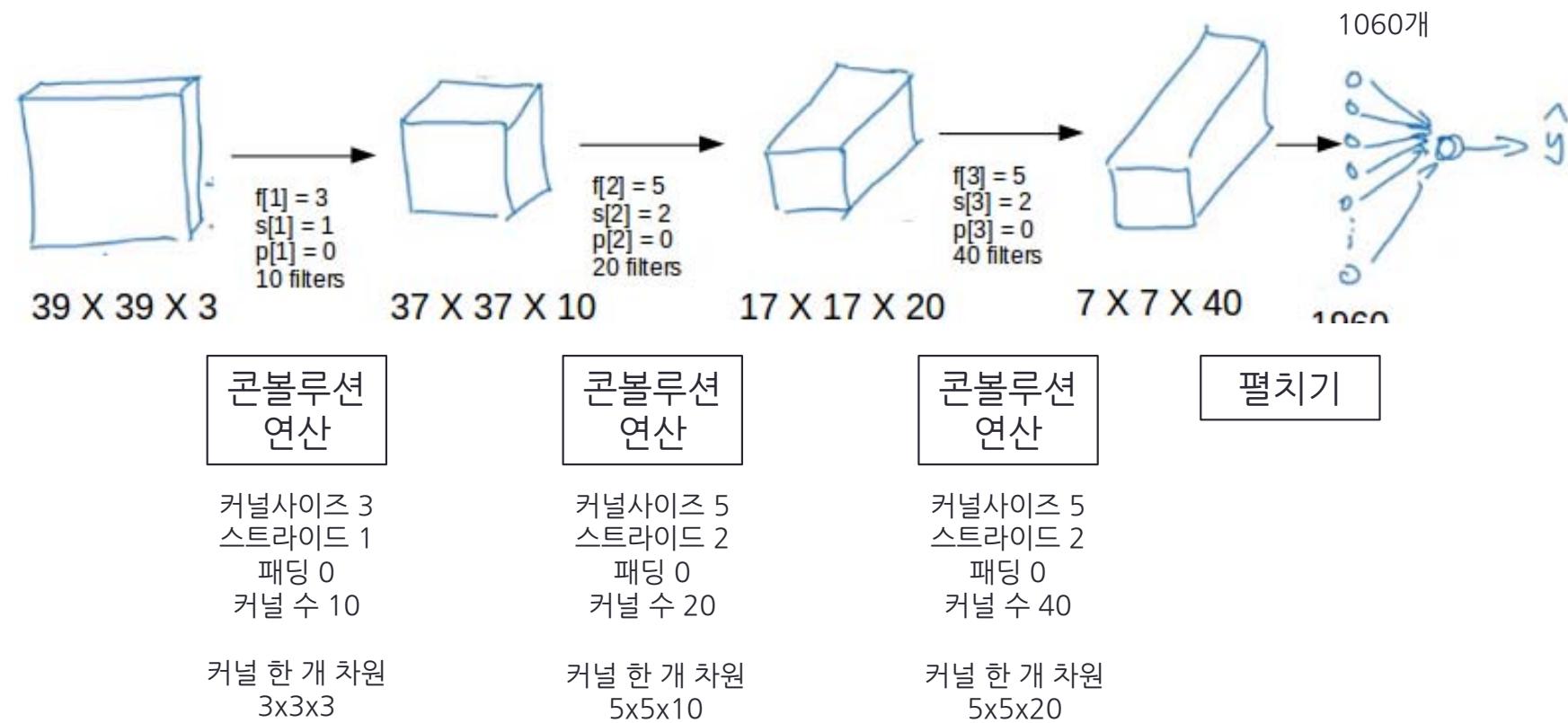
CNN 연산연습



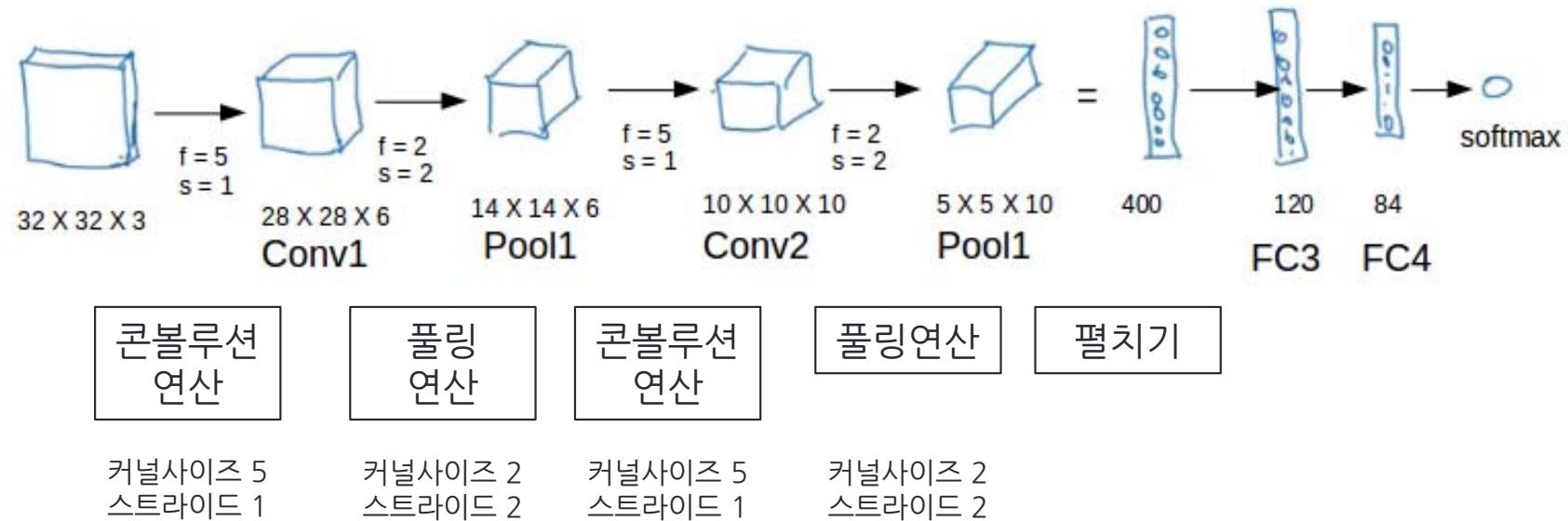
CNN 연산연습



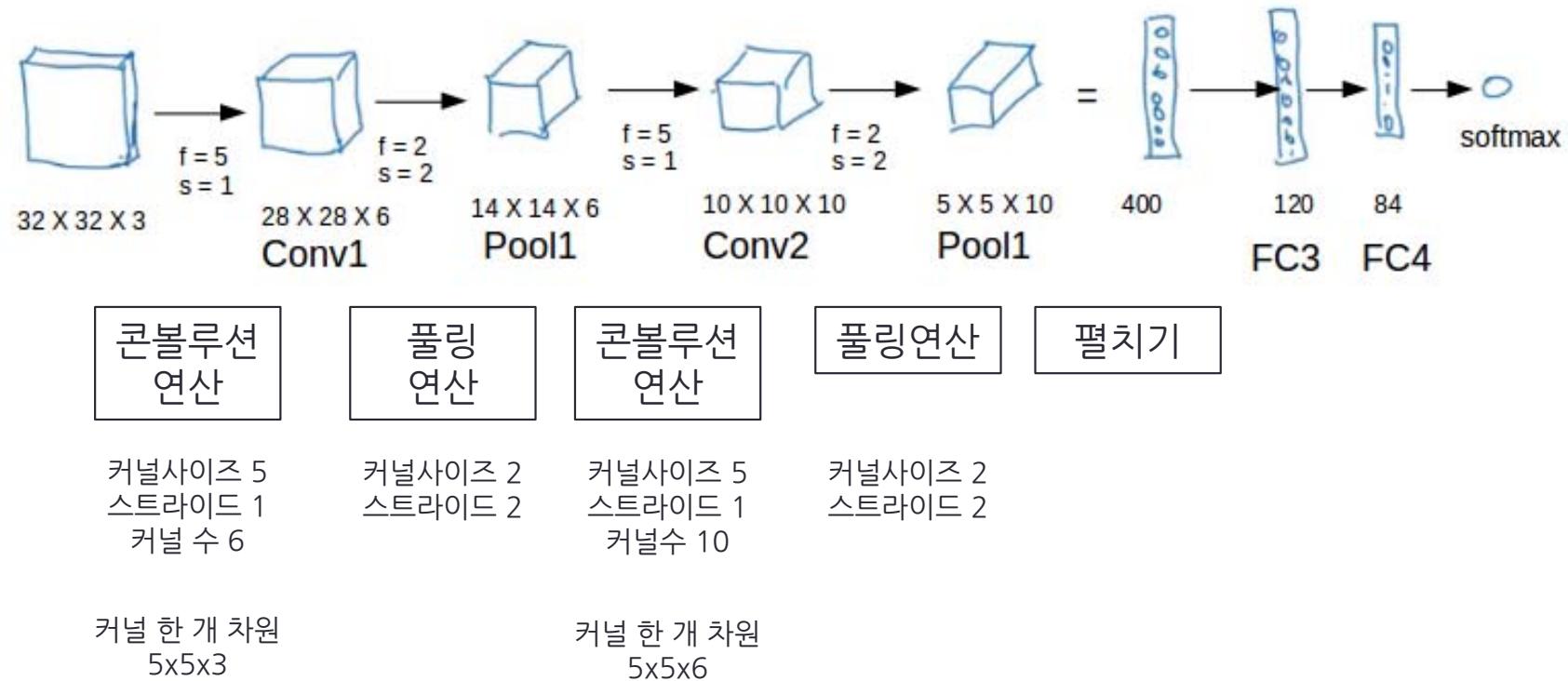
CNN 연산연습



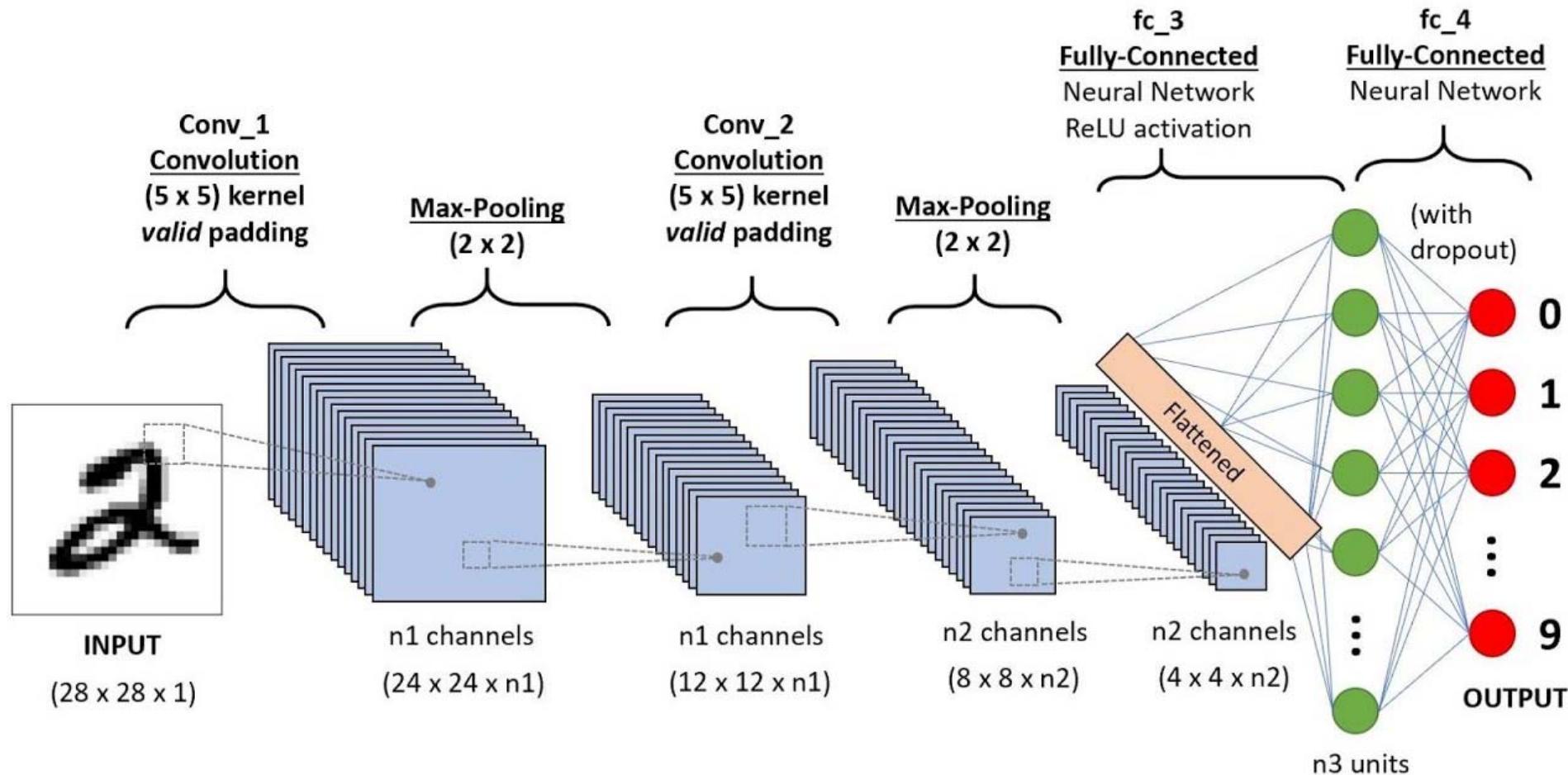
CNN 연산 연습



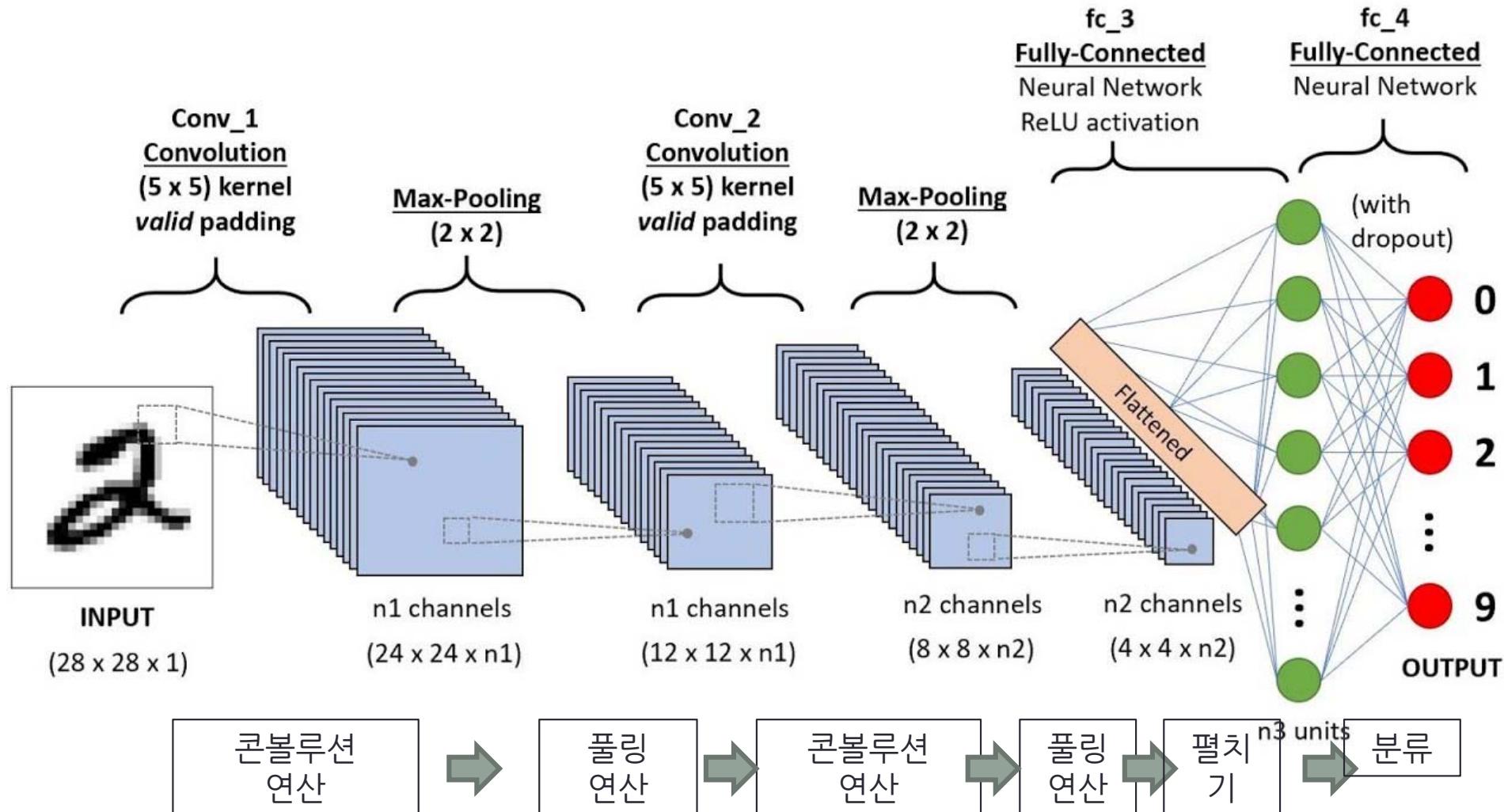
CNN 연산 연습



정리: CNN의 일반적인 구조

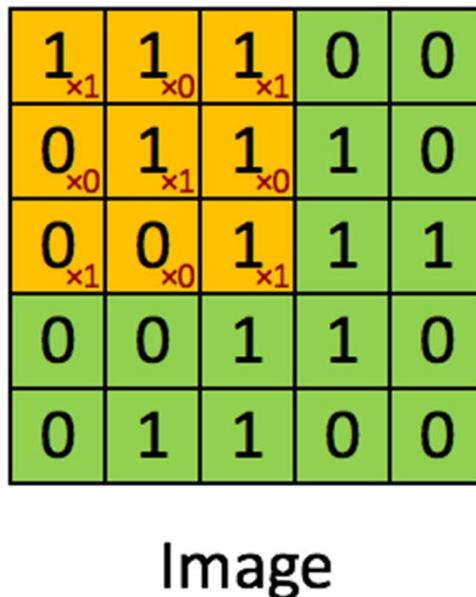


정리: CNN의 일반적인 구조



콘볼루션 연산의 특징: Weight Sharing

- 가중치 공유(Weight Sharing)
 - 같은 가중치를 이미지의 여러부분에 적용
 - 계산에 필요한 패러미터의 수를 크게 줄일수 있음
 - 25 (pixel) → 9 (output)
 - 일반 Dense 뉴럴넷에서 필요한 가중치 수 → $(25+1)*9$
 - 콘볼루션 연산에서 필요한 가중치 수 → $9+1$



1	0	1
0	1	0
1	0	1

4		

Convolved
Feature

콘볼루션 연산의 특징: Shift Invariance

- 이미지가 외곡, 변형되어 있어도 사람은 두 그림이 같은 X를 나타내는 것을 알 수 있다.

-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	1	1	-1	-1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

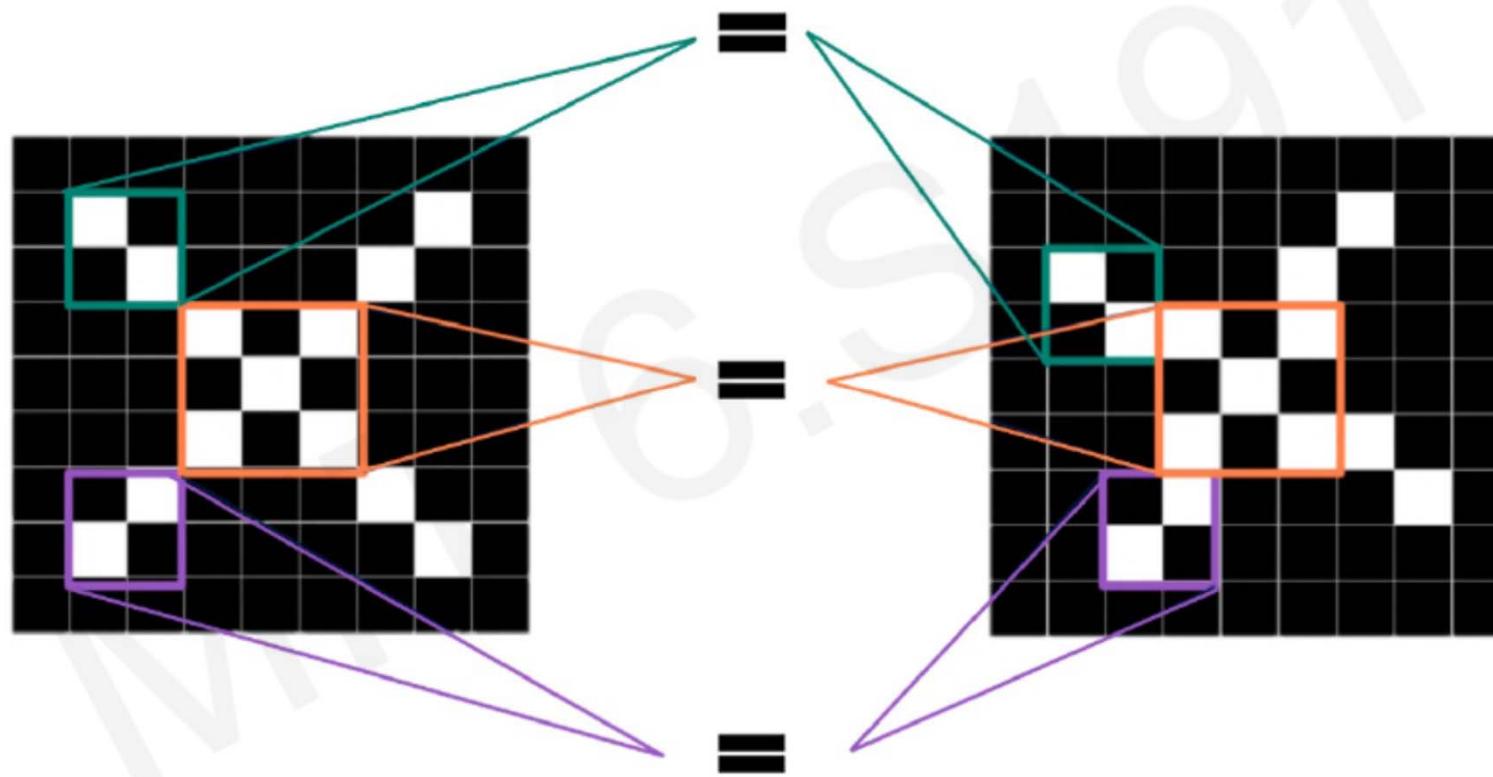
?



-1	-1	-1	-1	-1	-1	-1	-1	-1
-1	-1	-1	-1	-1	-1	-1	1	-1
-1	1	-1	-1	-1	1	-1	-1	-1
-1	-1	1	1	-1	-1	1	-1	-1
-1	-1	-1	-1	1	-1	-1	-1	-1
-1	-1	-1	-1	1	-1	1	-1	-1
-1	-1	-1	1	-1	1	-1	-1	-1
-1	-1	1	-1	-1	-1	1	-1	-1
-1	1	-1	-1	-1	-1	-1	1	-1
-1	-1	-1	-1	-1	-1	-1	-1	-1

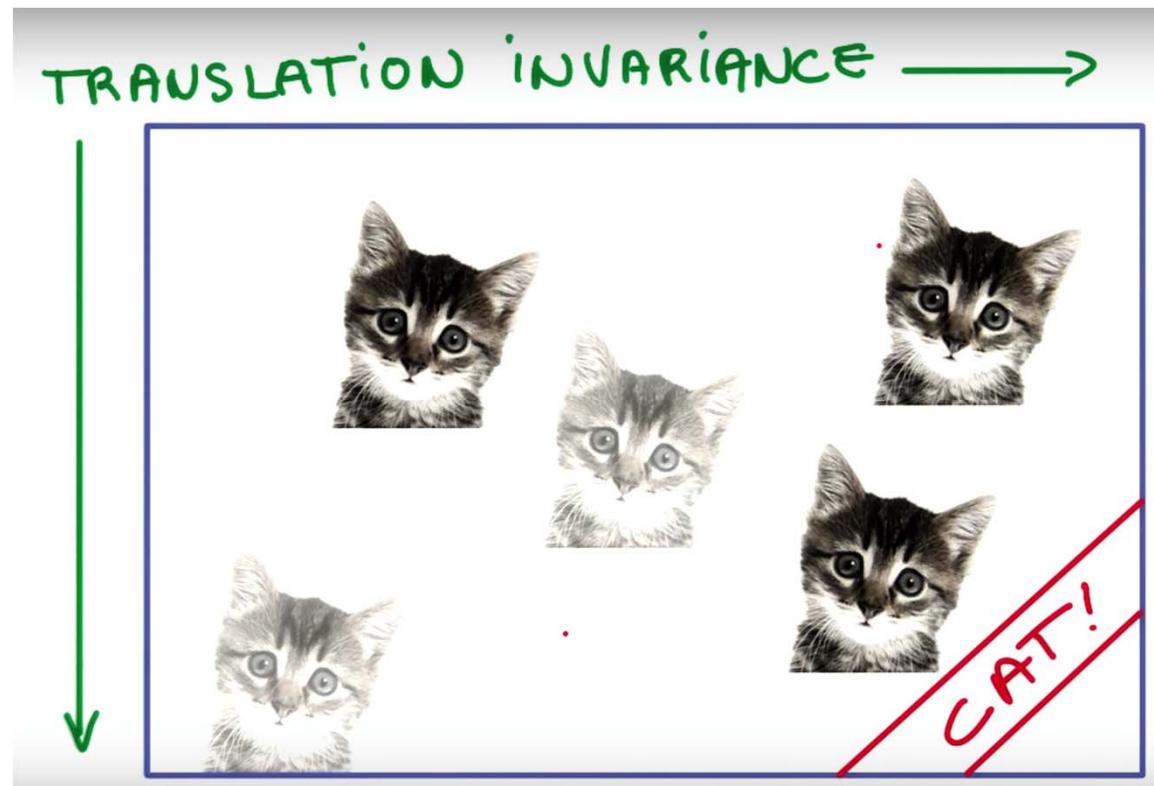
Convolutional Operation의 특성 - Shift Invariant

- 우리가 이미지를 인식할 때 특성의 존재 여부가 중요하지,
특성이 어느 위치에 있는지는 크게 중요하지 않다



Convolutional Operation의 특성 - Shift Invariant

- 우리가 이미지를 인식할때 특성의 존재 여부가 중요하지, 특성이 어느 위치에 있는지는 크게 중요하지 않다.
- (커널이 유용한 특성을 표현한다면) 이미지의 어느 부분에 해당 특성이 존재한다 해도 이를 잘 잡아낼 수 있다.



02. 딥러닝과 컴퓨터 비전

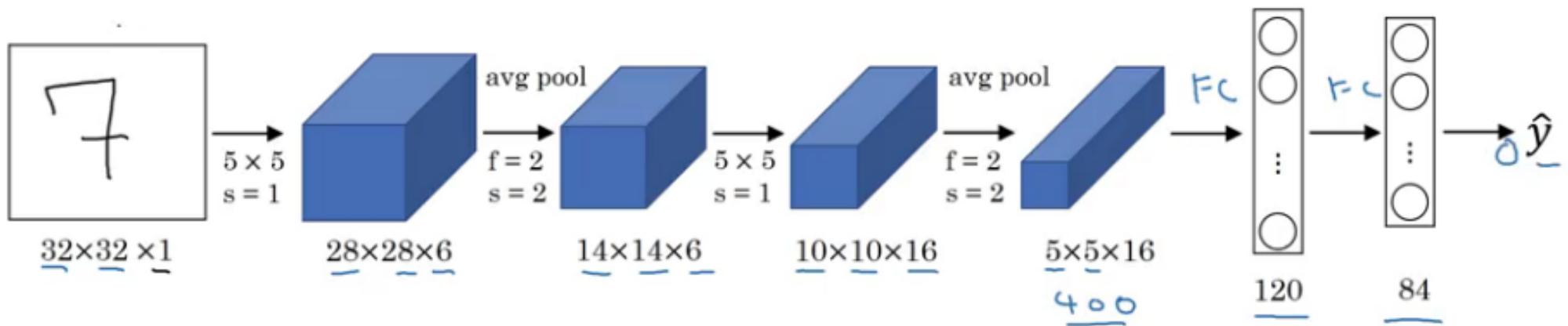
CNN 아키텍처의 발전

딥러닝 아키텍처

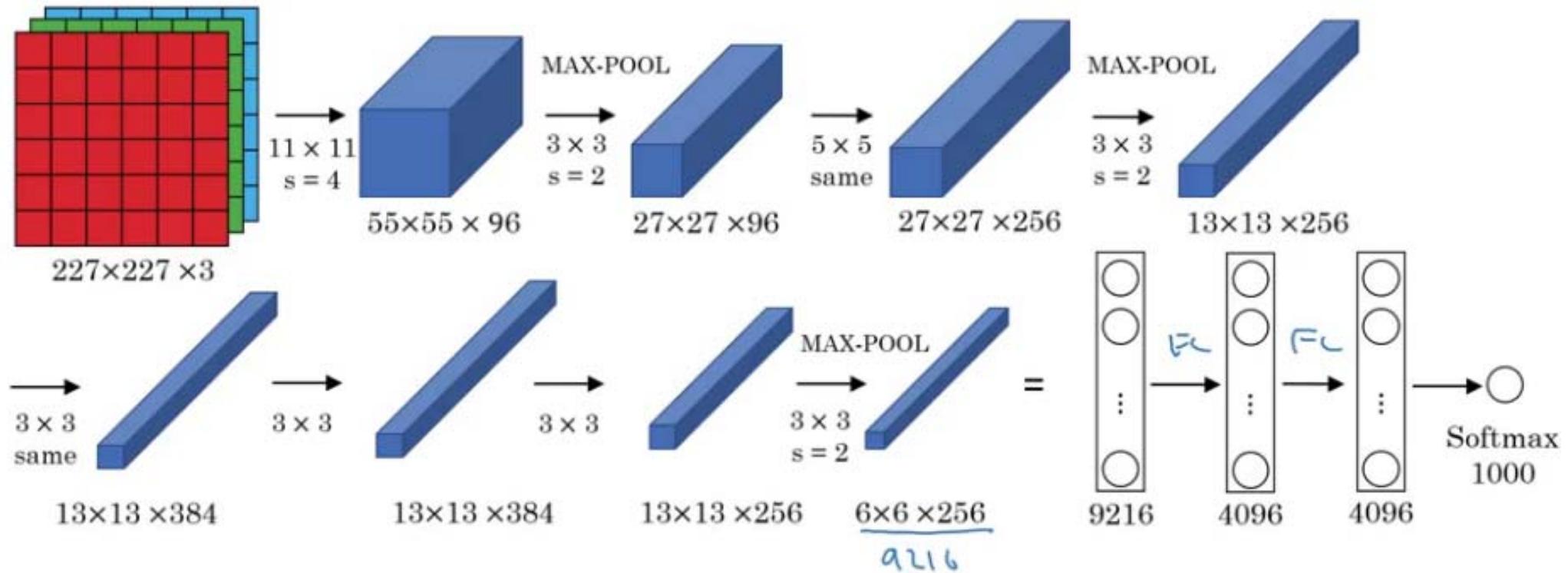
- 딥러닝 아키텍처
 - 딥러닝을 구성하는 특정한 패턴
 - 특정 작업에 보다 효율적인 딥러닝 아키텍처를 찾기 위한 연구가 이루어지고 있음
- 컴퓨터 비전에서 딥러닝 아키텍처의 발전과정
 - LeNet-5
 - AlexNet
 - VGG-16
 - ResNet
 - Inception

클래식 CNN 아키텍처: LeNet-5

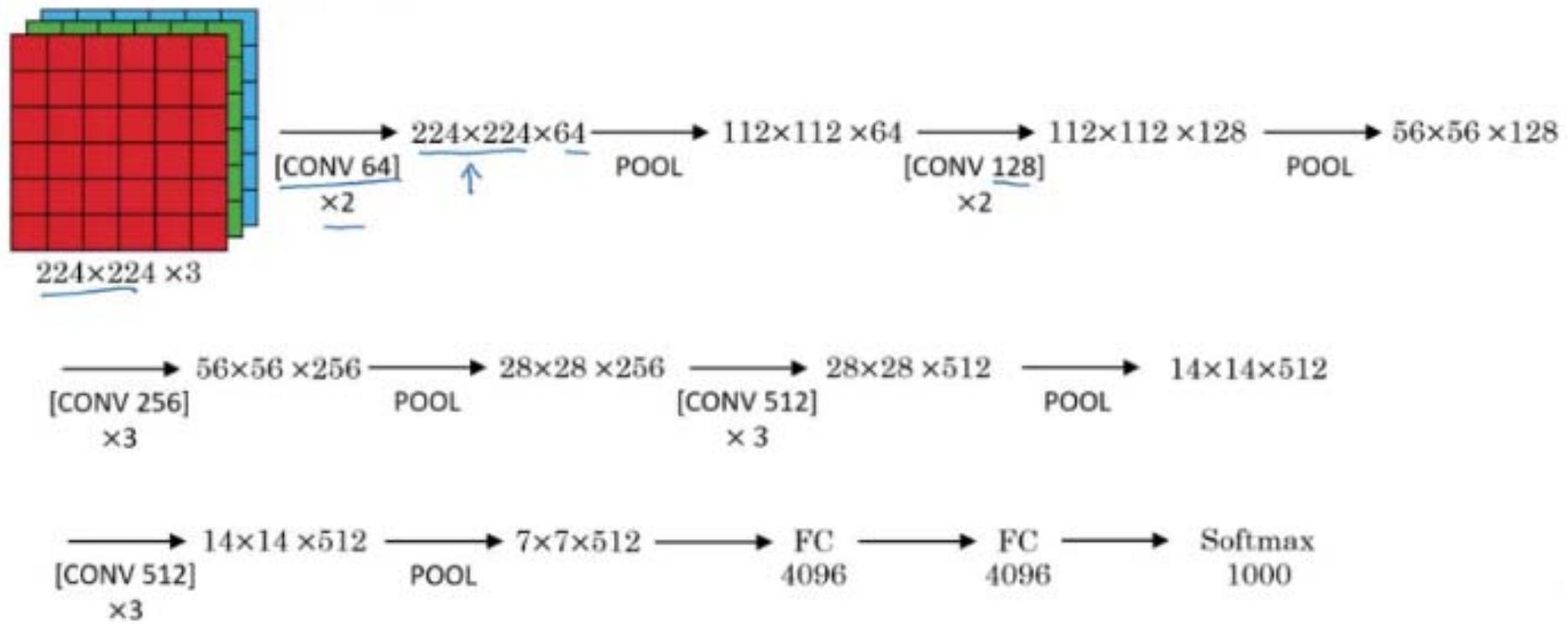
- 우리가 배운 Classical한 CNN 구조



클래식 CNN 아키텍처: AlexNet

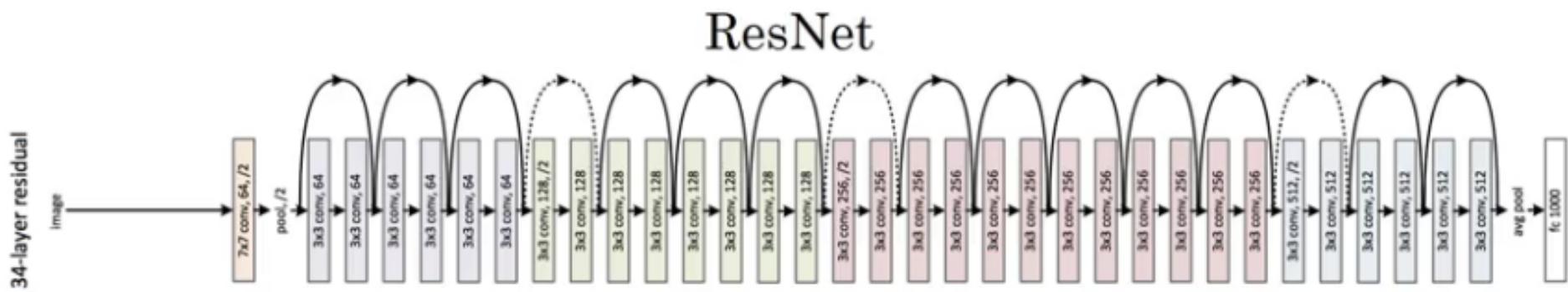


클래식 CNN 아키텍처: VGG-16



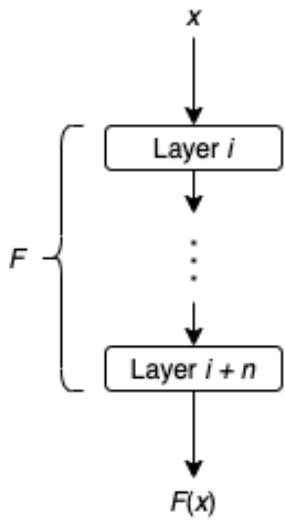
모던 CNN 아키텍처: ResNet

- 클래식 CNN의 문제점
 - 층이 깊어질수록 정확도가 증가
 - 하지만 동시에 학습에 필요한 데이터량이 커지고 수렴속도가 느려지는 단점 발생 (Gradient Vanishing Problem)
- 모던 CNN 아키텍처
 - ResNet
 - Inception

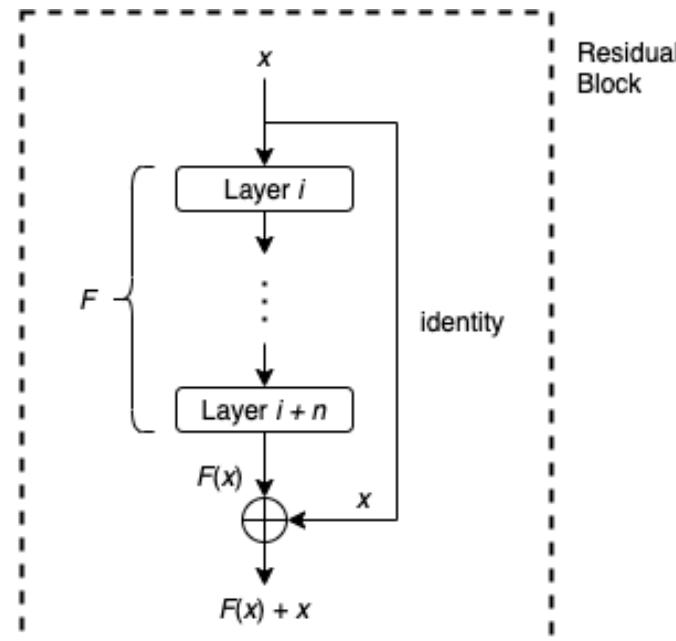


모던 CNN 아키텍처: 잔차연결(Residual Connection)

- X : 입력값
- $F(X)$: X 가 Layer를 통해 출력되는 값
- 잔차연결: $F(X)+X$ 한 값을 다음 layer에 입력값으로 활용



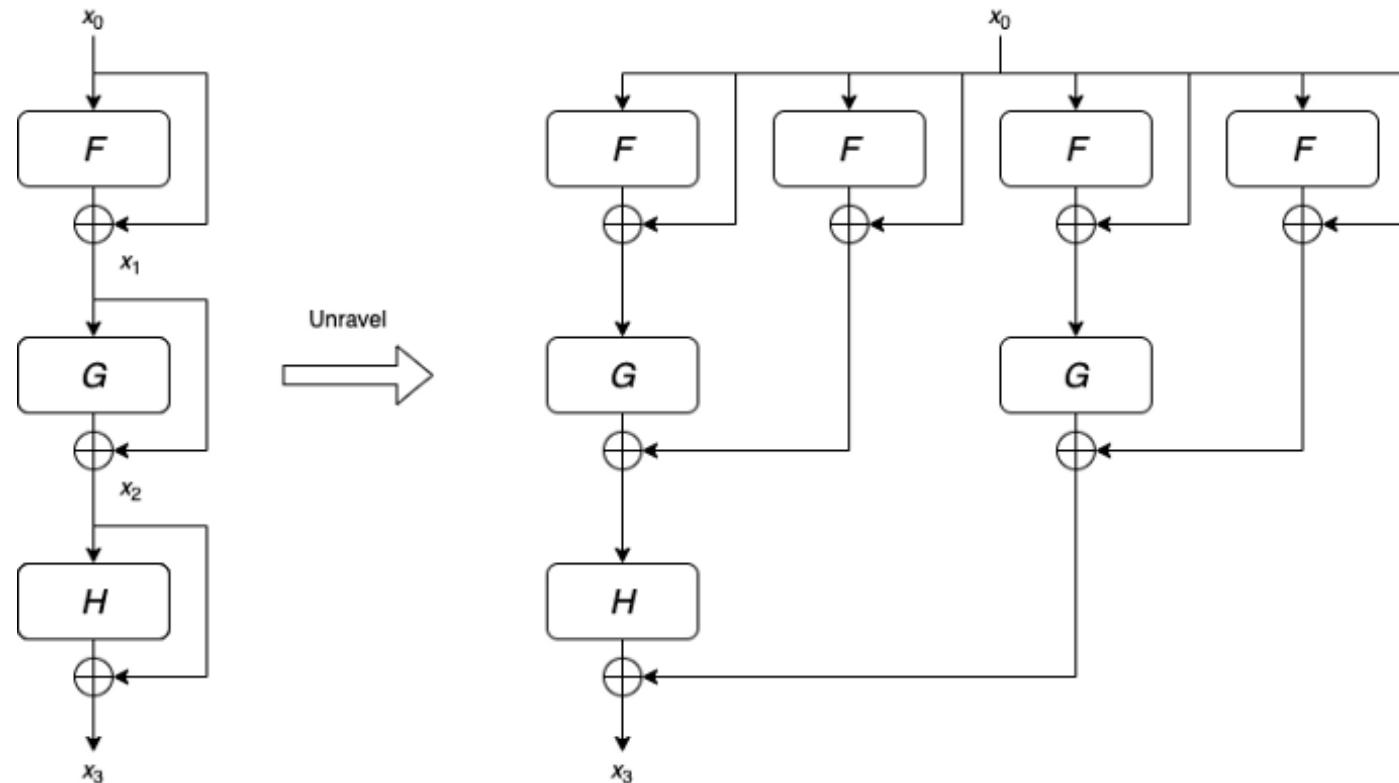
Traditional Feedforward
without Residual Connection



With Residual Connection

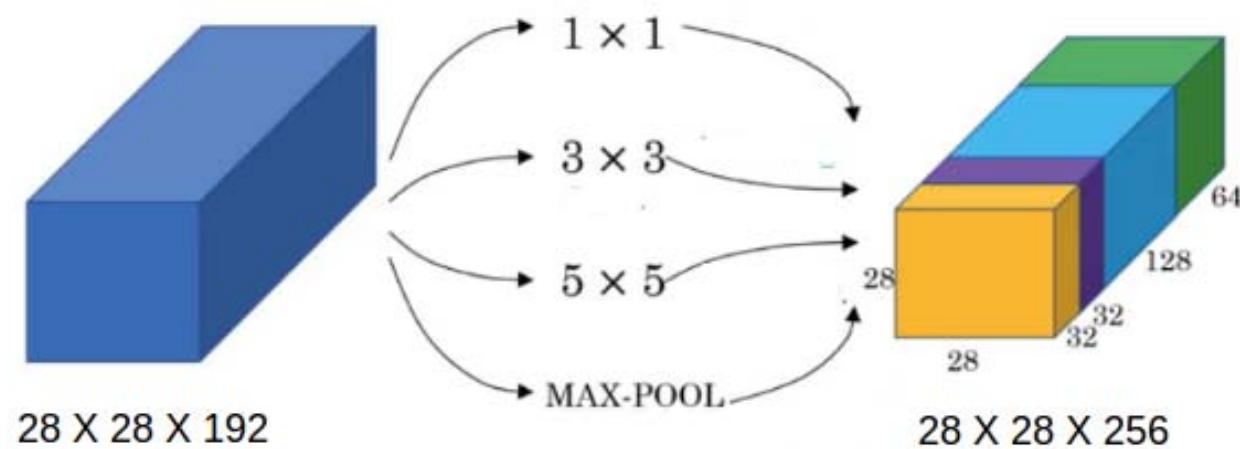
모던 CNN 아키텍처: 잔차연결의 효과

- 기존의 아키텍처: 1개 경로를 따라 데이터 이동
- 잔차연결의 효과: 여러 개의 우회경로가 생겨나는 효과 + 양상블



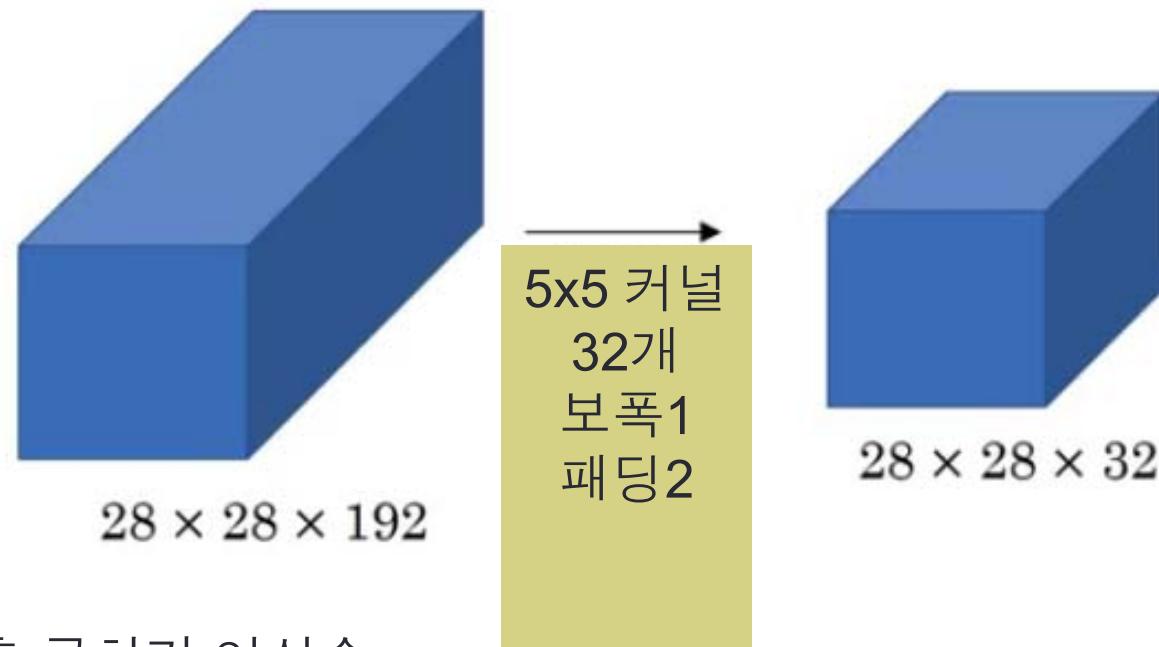
$$\begin{aligned}x_3 &= H(x_2) + x_2 \\&= H(G(x_1) + x_1) + G(x_1) + x_1 \\&= H(G(F(x_0) + x_0) + F(x_0) + x_0) + G(F(x_0) + x_0) + F(x_0) + x_0\end{aligned}$$

모던 CNN 아키텍처: 인셉션



모던 CNN 아키텍처: 인셉션

- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함

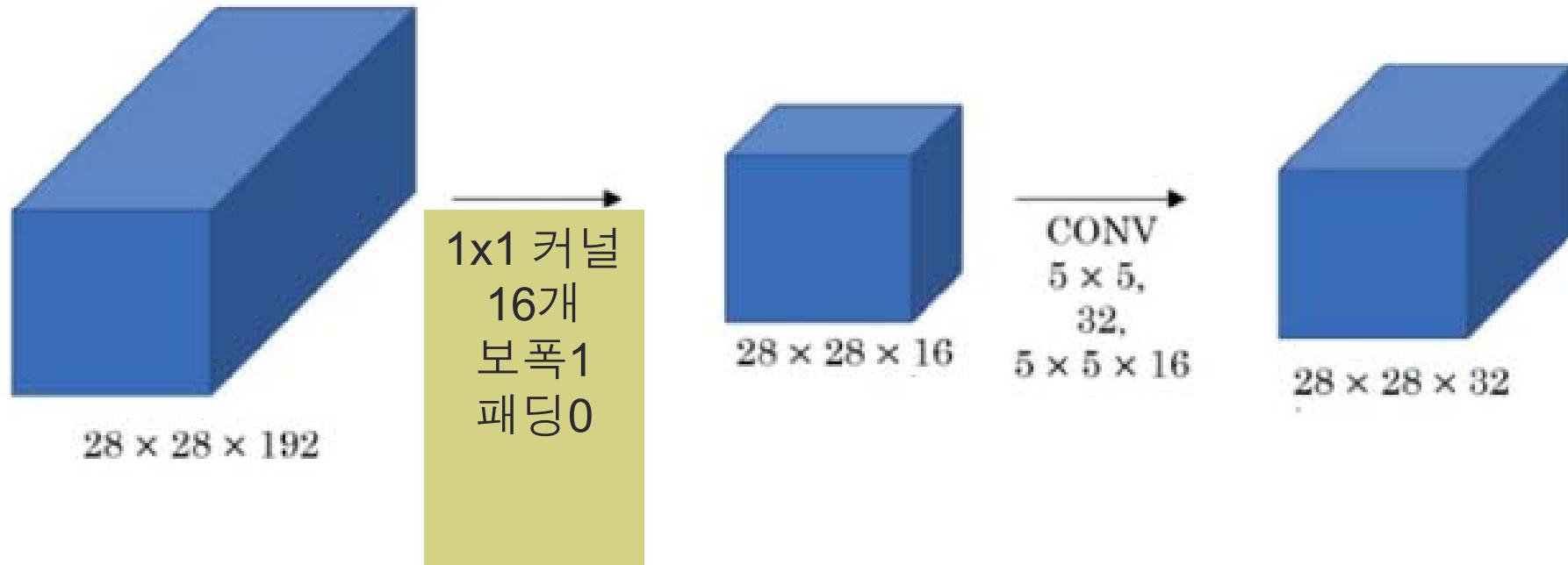


총 곱하기 연산수

- 커널 1개의 차원: $5 \times 5 \times 192$
- 입력 데이터에 대해 수행하는 연산: 28×28
- 커널의 수 32개
- 총 연산: $5 \times 5 \times 192 \times 28 \times 28 \times 32 = 120,422,400$

모던 CNN 아키텍처: 인셉션

- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함

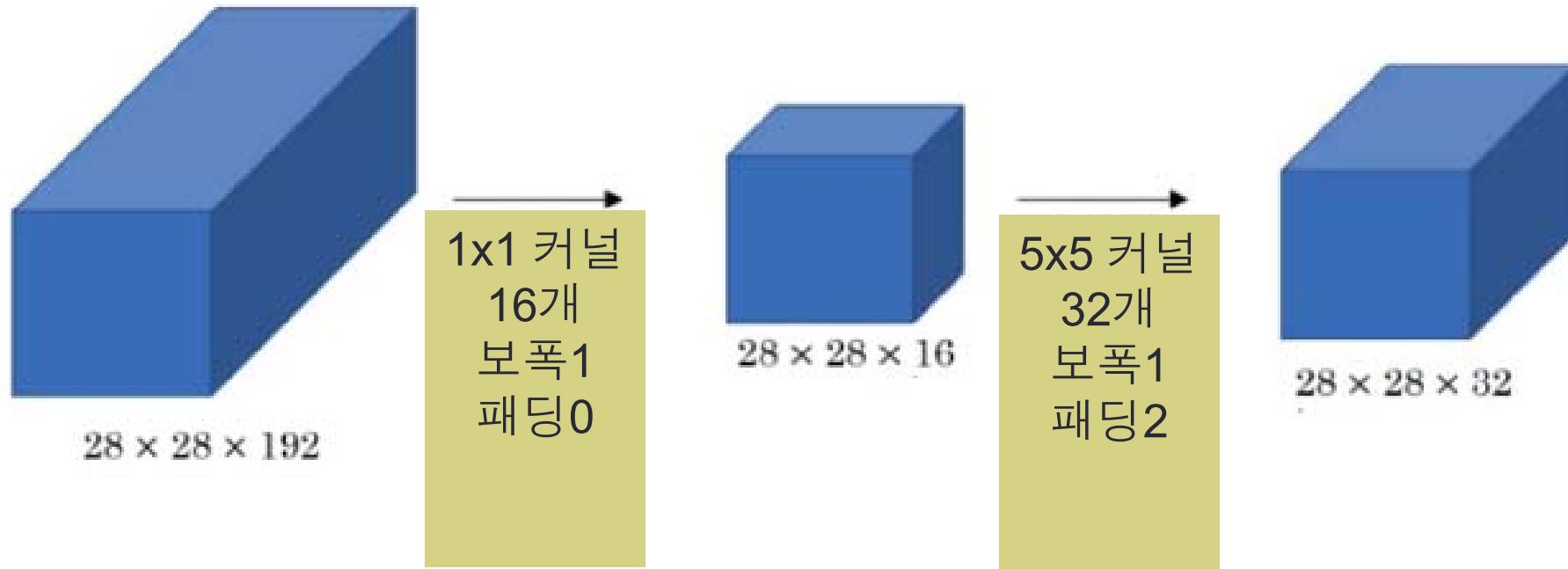


총 곱하기 연산수

- 커널 1개의 차원: $1 \times 1 \times 192$
- 입력 데이터에 대해 수행하는 연산: 28×28
- 커널의 수 16개
- 총 연산: $1 \times 1 \times 192 \times 28 \times 28 \times 16 = 2,408,448$

모던 CNN 아키텍처: 인셉션

- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함

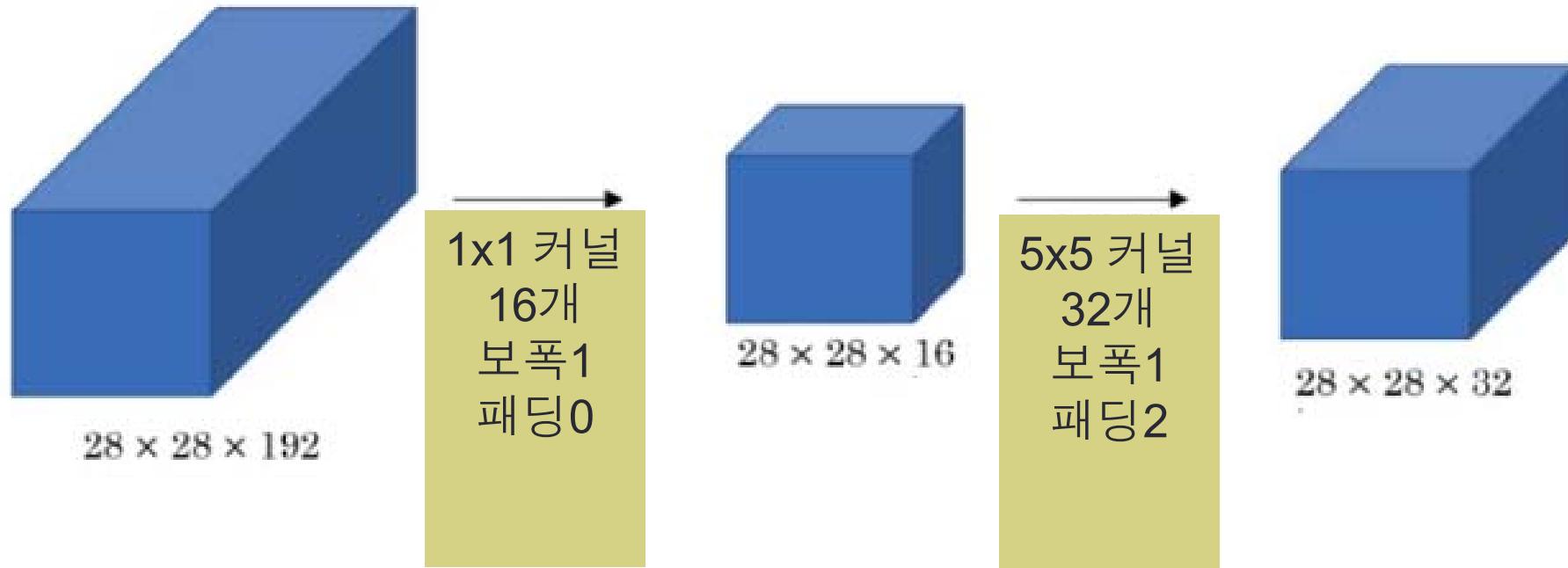


총 곱하기 연산수

- 커널 1개의 차원: 5x5x16
- 입력 데이터에 대해 수행하는 연산: 28x28
- 커널의 수 32개
- 총 연산: $5 \times 5 \times 16 \times 28 \times 28 \times 32 = 10,035,200$

모던 CNN 아키텍처: 인셉션

- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함

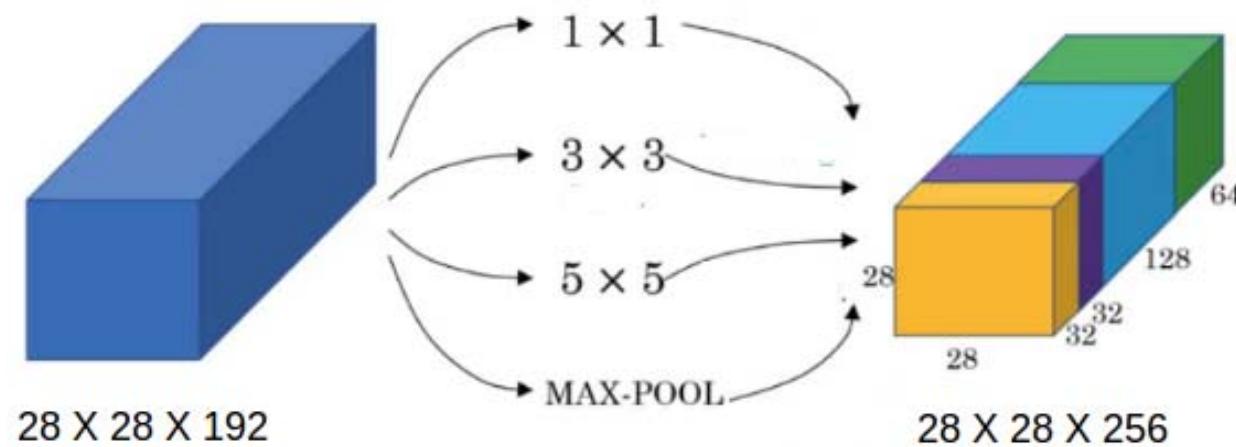


총 곱하기 연산수

- 커널 1개의 차원: 5x5x16
- 입력 데이터에 대해 수행하는 연산: 28x28
- 커널의 수 32개
- 총 연산: $5 \times 5 \times 16 \times 28 \times 28 \times 32 = 10,035,200$
- $\geq 120,422,400$

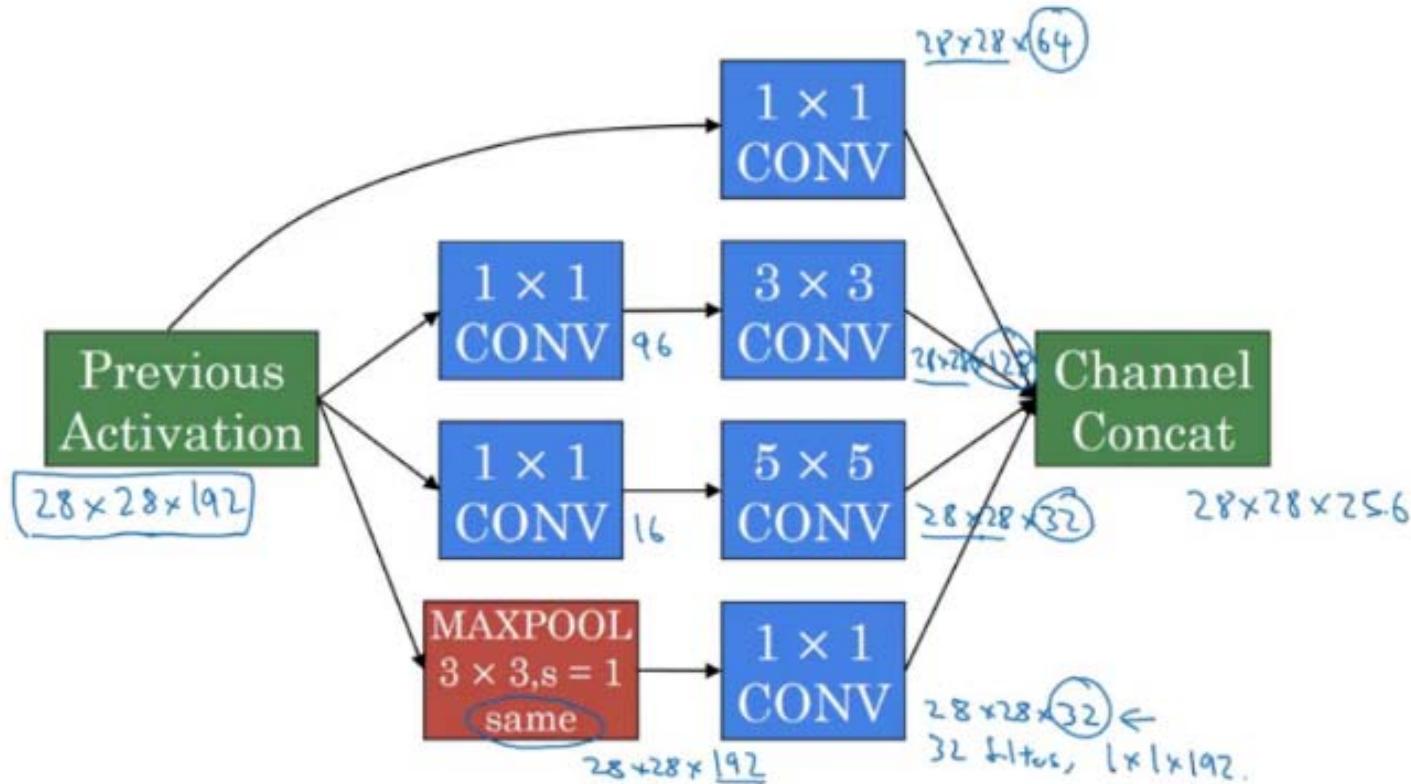
모던 CNN 아키텍처: 인셉션

- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함



모던 CNN 아키텍처: 인셉션

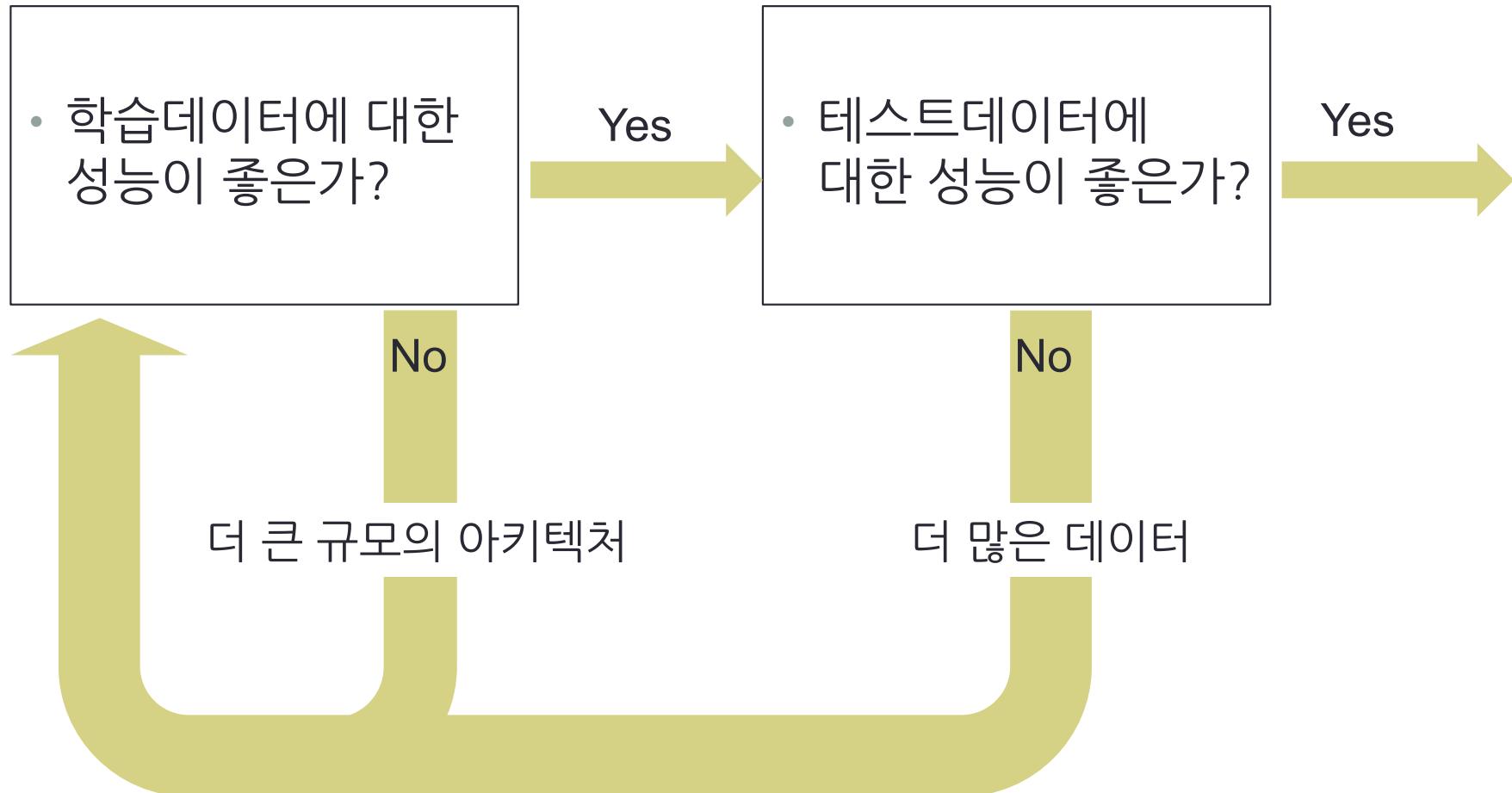
- 콘볼루션의 곱하기 연산의 수를 효율적으로 줄여야 함



02. 딥러닝과 컴퓨터 비전

CNN의 정확도 높이기

뉴럴 네트워크의 학습전략

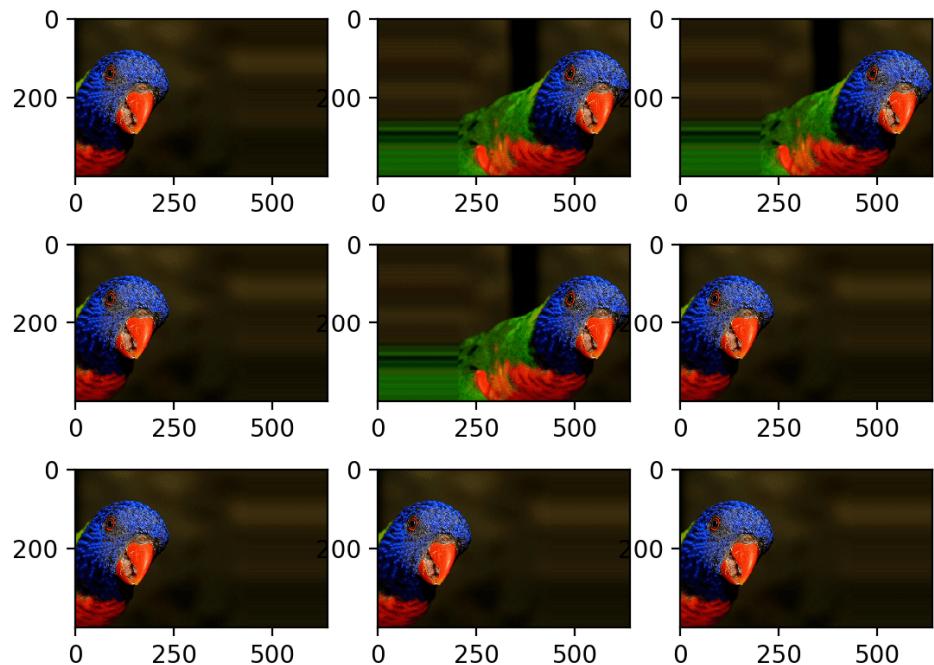


Data Augmentation

- 데이터 증강
 - CNN은 이미지에서 특성을 뽑아내는 방법
 - 같은 이미지에 변형을 가하여 다른 데이터인 것처럼 간주
 - 이미지의 특성을 포함한 데이터가 많아지므로 정확도가 높아질 수 있음

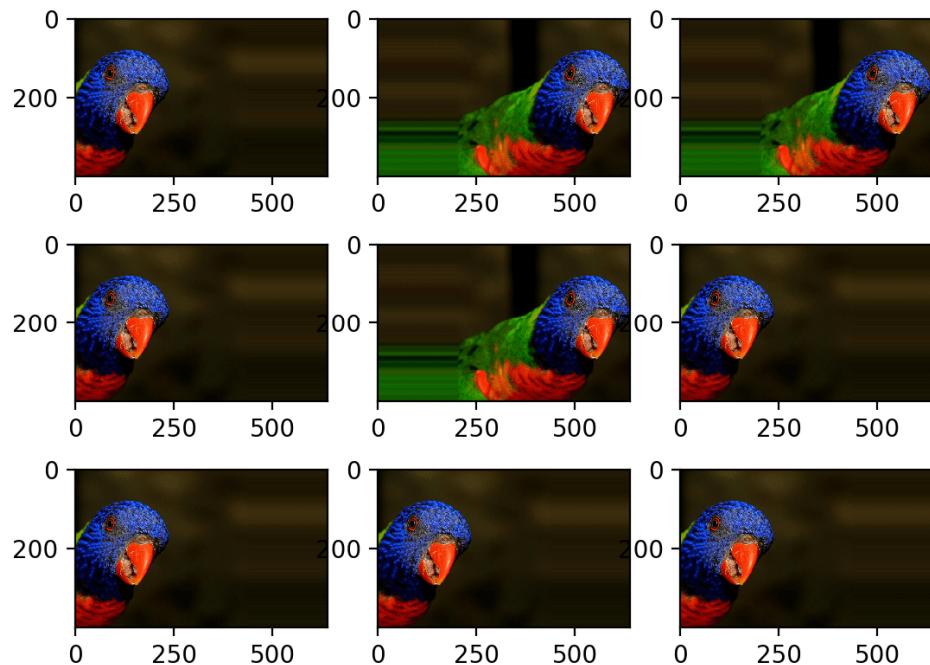
Data Augmentation

- 데이터 증강
 - CNN은 이미지에서 특성을 뽑아내는 방법
 - 같은 이미지에 변형을 가하여 새로운 데이터를 생성
 - 이미지의 특성을 포함한 데이터가 많아지므로 정확도가 높아질 수 있음



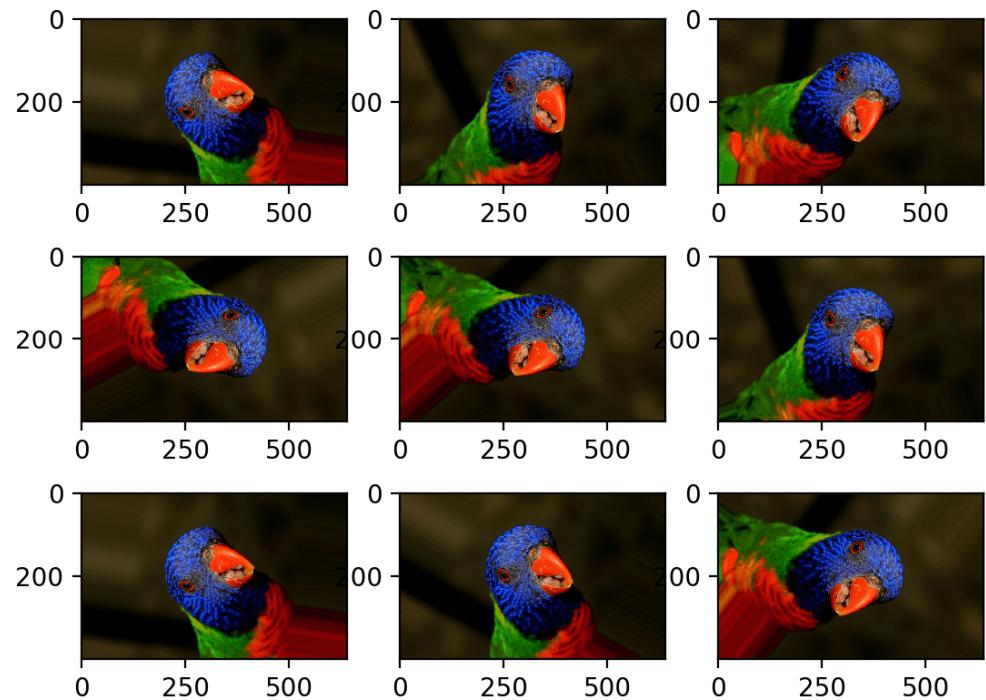
Data Augmentation

- 데이터 증강 방식
 - 위치변경 (좌, 우, 반전)



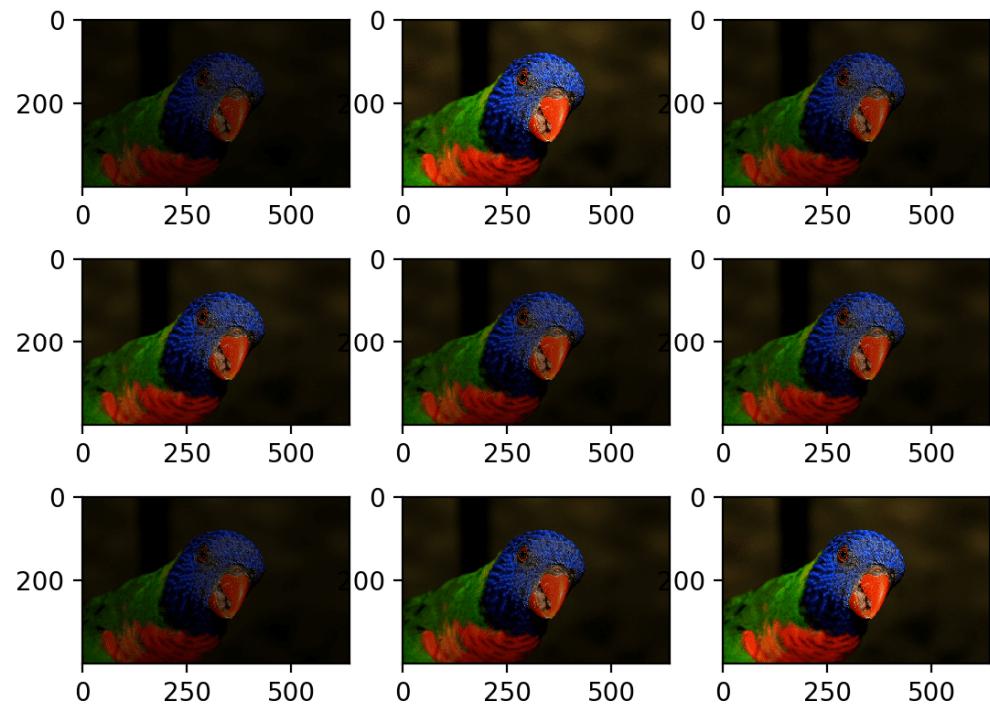
Data Augmentation

- 데이터 증강 방식
 - 위치회전



Data Augmentation

- 데이터 증강 방식
 - 색깔 바꾸기 등



Data Synthesis (데이터 합성)

- 데이터 증강이 기존 데이터에 noise를 발생시켜 변형시키는 것이라면 데이터 합성은 새롭게 합성된 이미지를 이용하여 새로운 데이터를 만드는 것을 의미

Artificial data synthesis for photo OCR



Real data

A
b
c
d
e
f
g

A
b
c
d
e
f
g

*A
b
c
d
e
f
g*

A
b
c
d
e
f
g

A
b
c
d
e
f
g



Real data

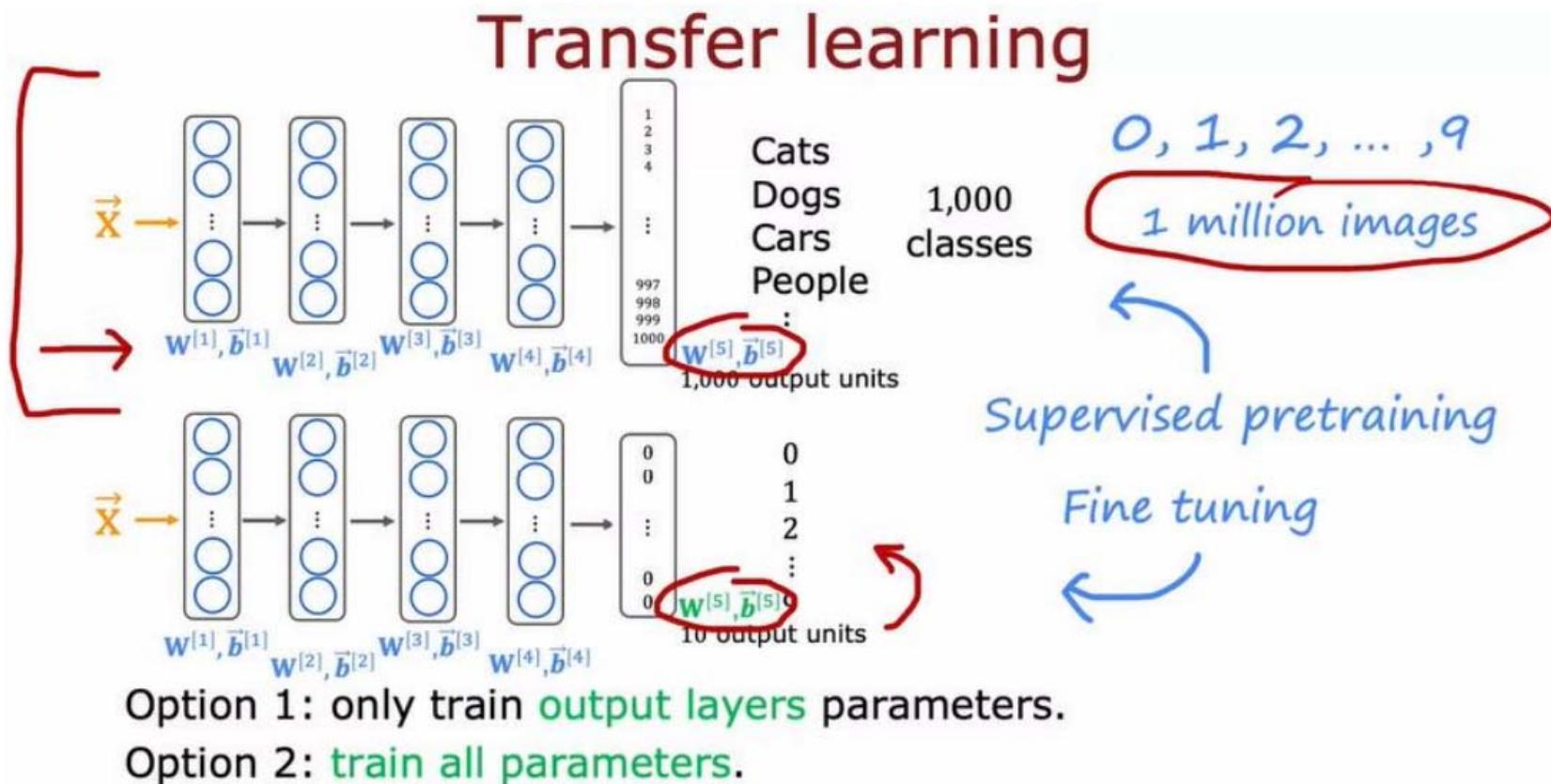
Artificial data synthesis for photo OCR



Synthetic data

Using Pretrained Model

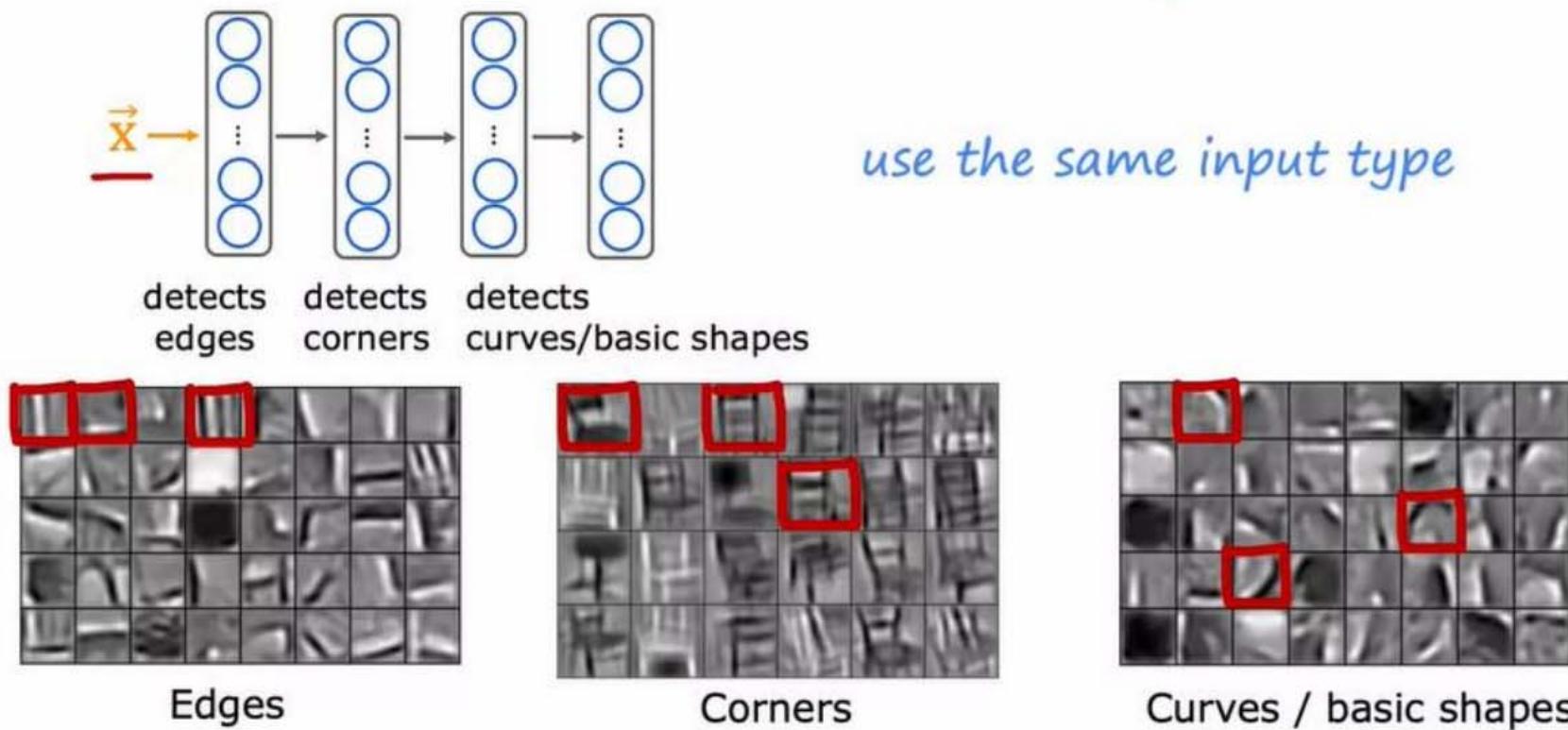
- 사전에 학습된 대형 모델을 가져옴
- 마지막 분류기 부분을 내가 원하는 분류기로 교체
- 내가 가진 (소량의) 데이터를 이용하여 마지막 부분만 재학습



Using Pretrained Model

- 왜 작동하는가?
 - 대량의 데이터를 학습하면서 이미지 전반을 분류하기 위한 공통된 패턴을 충분히 학습하였기 때문

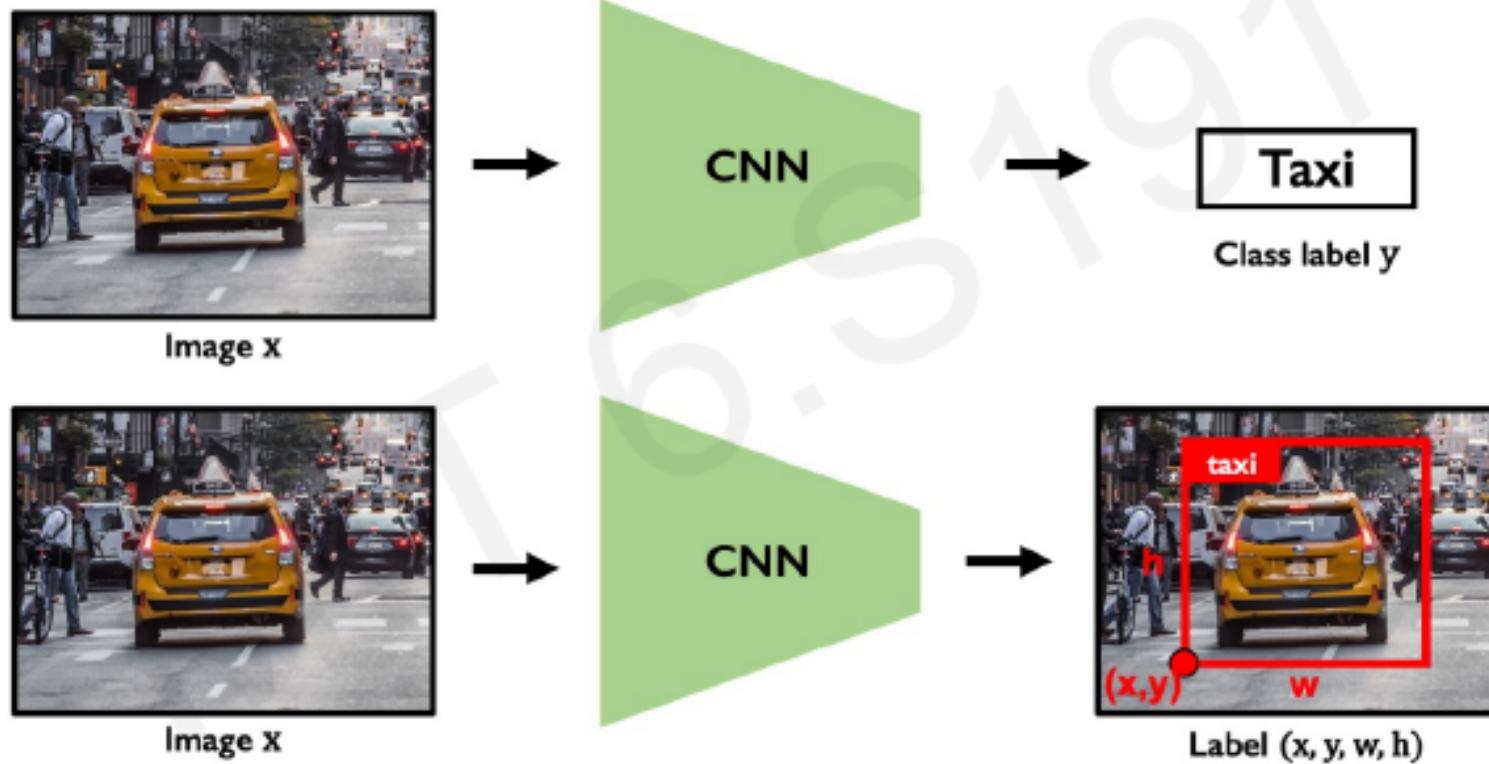
Why does transfer learning work?



02. 딥러닝과 컴퓨터 비전

그 외 컴퓨터 비전 분야 소개

Object Detection



Object Detection



Image X



Output:
taxi: (x₁, y₁, w₁, h₁)

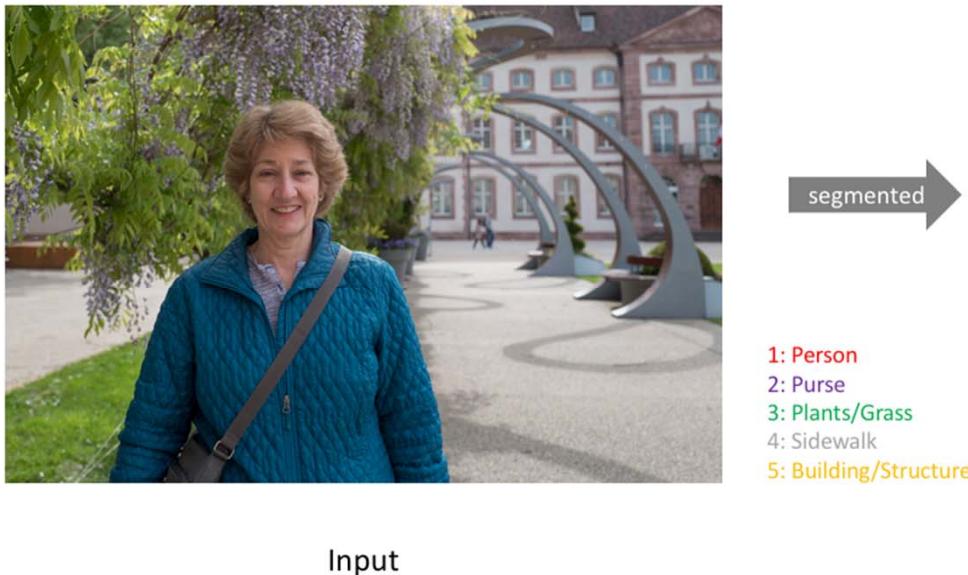
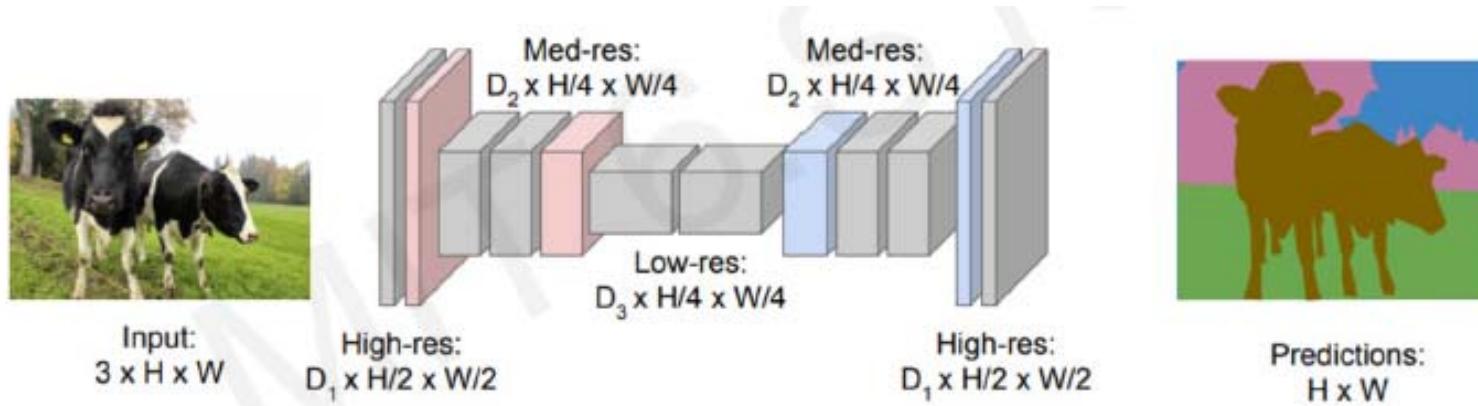


Image Y



Output:
taxi: (x₁, y₁, w₁, h₁)
person: (x₂, y₂, w₂, h₂)
person: (x₃, y₃, w₃, h₃)
....

Semantic Segmentation



3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	5	5	5	5	5
5	5	3	3	3	3	3	3	1	1	1	1	1	1	1	1	1	5	5	5	5	5
4	4	3	4	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	5	5
4	4	3	4	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	5
4	4	4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4
3	3	3	1	1	1	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4
3	3	3	1	2	2	1	1	1	1	1	1	1	1	1	1	1	4	4	4	4	4

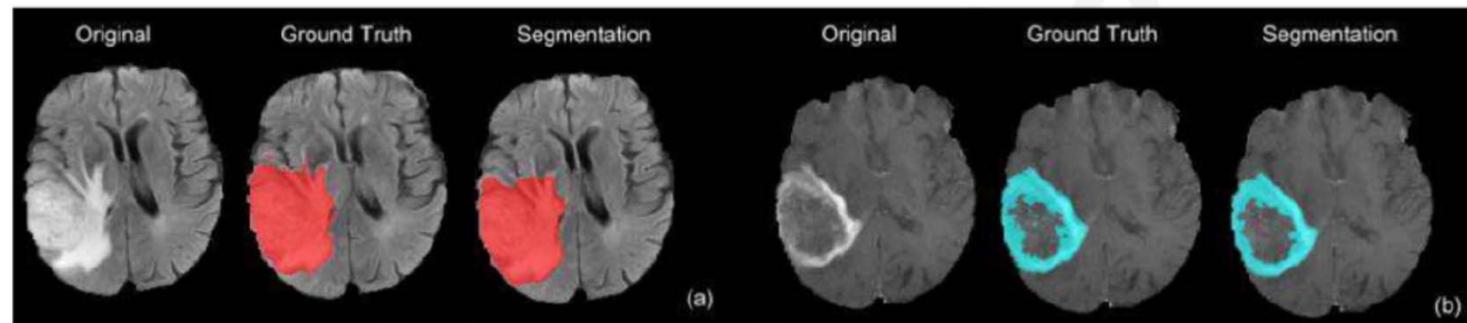
Input

Semantic Labels

Semantic Segmentation 응용

Semantic Segmentation: Biomedical Image Analysis

Brain Tumors
Dong+ MIUA 2017.



Malaria Infection
Soleimany+ arXiv 2019.

