이 미 지　딥 러 닝

ToBig's　9기　김수지

Deep Learning

Convolutional Neural Networks &
Generative Adversarial Networks

Contents

이미지 딥러닝??

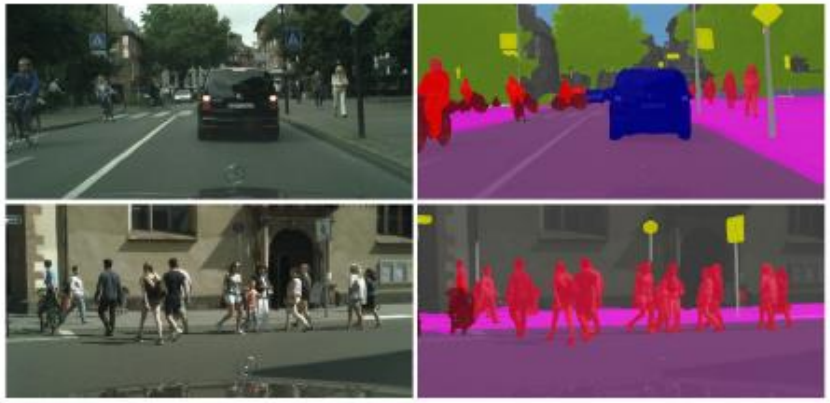
기계가 이미지를 인식해서 그 이미지를 분류/검출/분할/생성 하는 것!

- Classification: 이미지 안에 어떤 사물이 포함되어 있는지 분류
- Detection: 이미지 안에 찾고자 하는 사물이 있다면 어느 위치에 포함되어 있는지 박스 형태로 검출하는 것
- Segmentation: 이미지 안에 찾고자 하는 사물이 있다면 어느 위치에 포함되어 있는지 픽셀 단위로 분할하는 것
  Detection보다 더 자세하게 위치를 표시함
- Generation: 이미지 생성, 이미지 보정

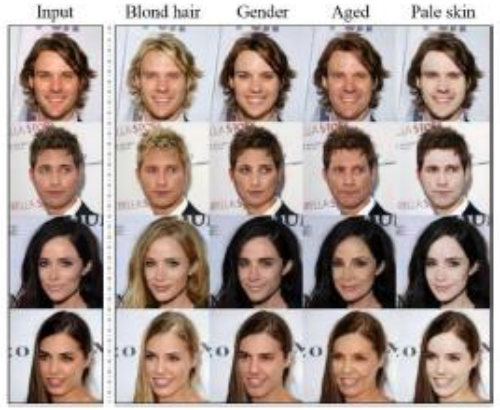
Unit 01 | CNN

Classification

Detection



Segmentation



Generation

Unit 01 | CNN

Question: 오… 이런 걸 어떻게 하나요?

Answer: CNN 알고리즘을 이용해 가능합니다!!

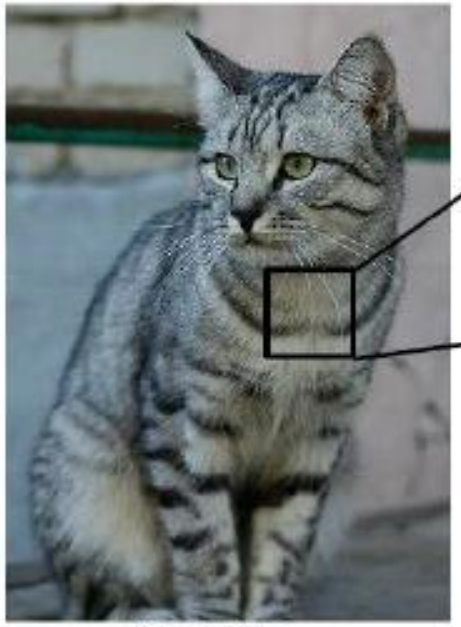
Cf) 이미지 딥러닝의 유효성이 크게 알려진 계기는 2012년 9월에 열린 일반물체 인식 콘테스트 ILSVRC(ImageNet Large Scale Visual Recognition Challenge)에서 크리제프스키(Alex Krizhevsky)가 제안한 AlexNet이 1위를 차지한 것이다.

Unit 01 | CNN

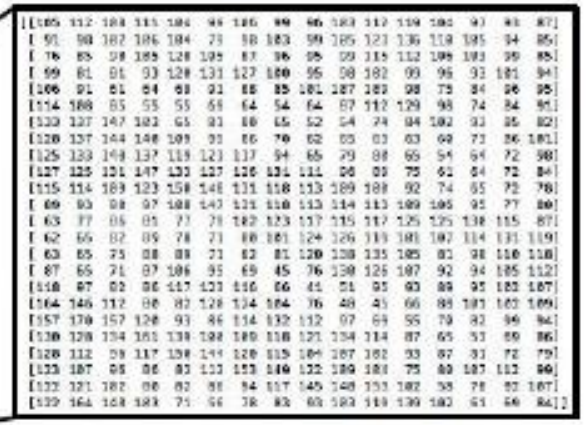
CNN을 배우기 전에, 우리가 인식하고자 하는 이미지에 대해서 먼저 알아봅시다!!

Unit 01 | CNN

Image



This image by Nikita is licensed under CC-BY 2.0

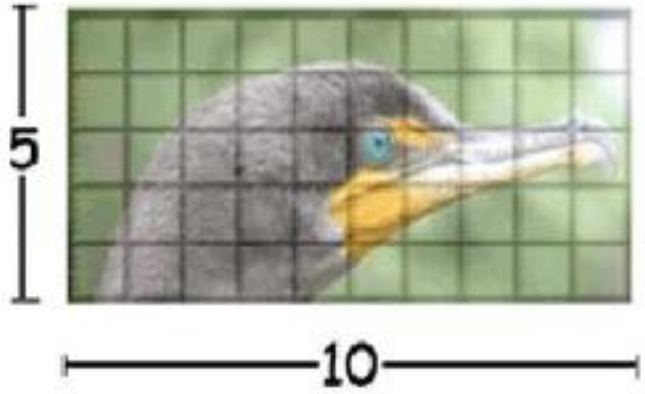


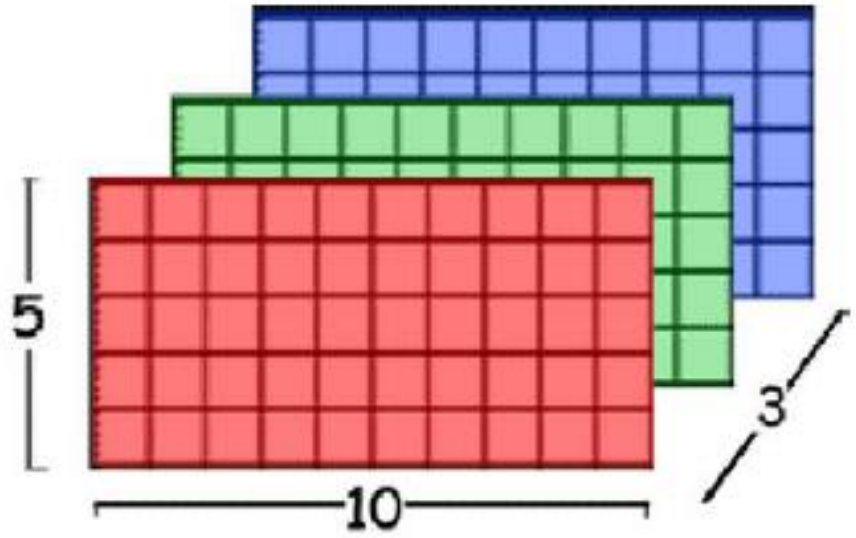
What the computer sees

An image is just a big grid of numbers between [0, 255]:

e.g. 800 x 600 x 3
(3 channels RGB)

Image



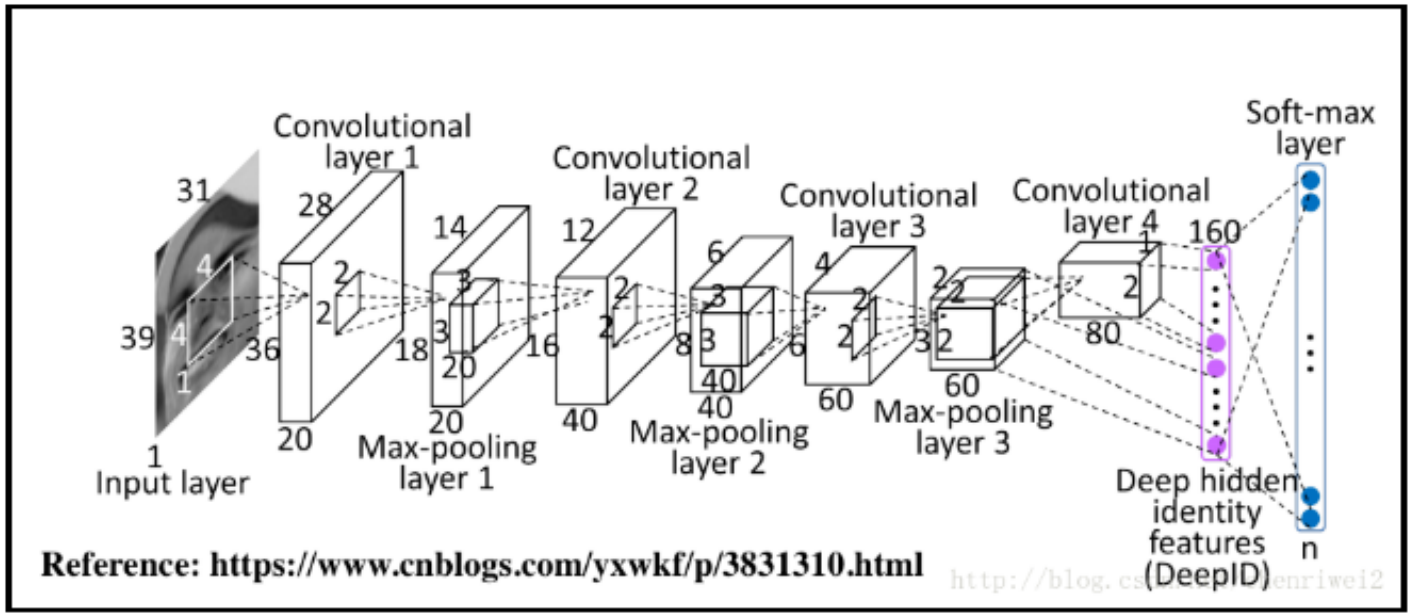


1. CNN

CNN이란?

구조적 정의:

convolutional layer와 pooling layer가 층을 번갈아가며 쌓아 올린 구조를 갖는 feedforward 신경망

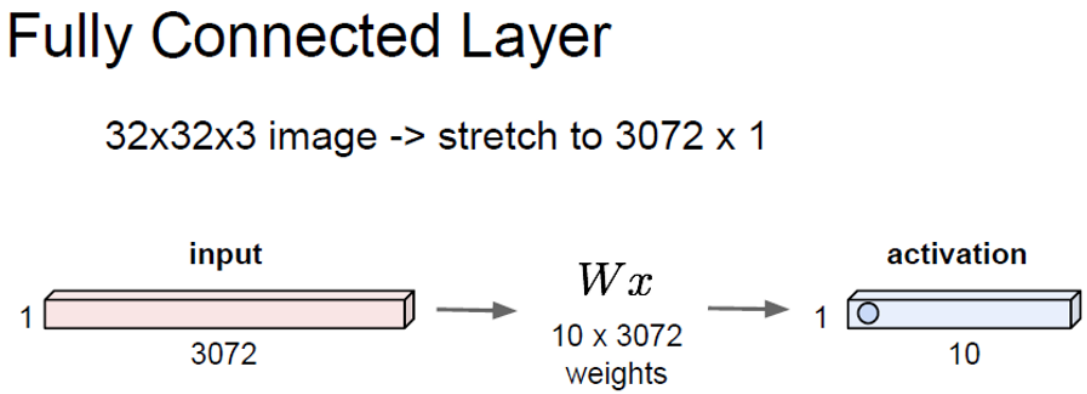


CNN이란?

기능적 정의:

이미지의 공간 정보를 유지한 상태로 학습이 가능한 지도 학습 모델

앞에서 봤듯이 이미지는 3차원 데이터.
Fully Connected Layer로 구성된 인공 신경망에 학습시킬 경우, input data는 1차원이어야 한다
따라서 image로 인공 신경망을 학습시킬 경우, 3차원인 사진을 1차원으로 평면화시켜야 함



CNN이란?

기능적 정의:

이미지의 공간 정보를 유지한 상태로 학습이 가능한 지도 학습 모델

앞에서 봤듯이 이미지는 3차원 데이터.
Fully Connected Layer로 구성된 인공 신경망에 학습시킬 경우, input data는 1차원이어야 한다
따라서 image로 인공 신경망을 학습시킬 경우, 3차원인 사진을 1차원으로 평면화시켜야 함

평면화 시키는 과정에서 공간 정보가 손실될 수 밖에 없다!
→ 이미지 공간 정보 유실로 인한 정보 부족으로 인공 신경망이 특징 추출과 학습을 비효율적으로 함

CNN의 장점

- 각 레이어의 입출력 데이터 형상 유지
- 이미지의 공간 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식
- 복수의 필터로 이미지 특징 추출 및 학습
- 추출한 이미지의 특징을 모으고 강화하는 Pooling Layer
- 필터를 공유 파라미터로 사용하기 때문에, 일반 인공 신경망과 비교하여 학습 파라미터가 매우 적음

CNN의 주요 용어

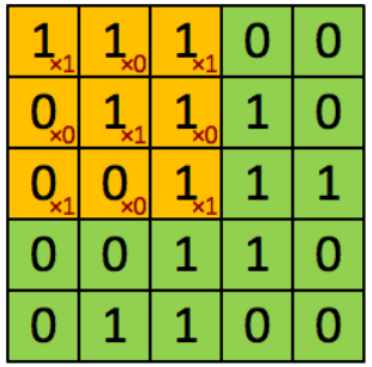
① Convolution Layer
② Feature Map, Activation Map
③ Channel(Depth)
④ Filter
⑤ Stride
⑥ Padding
⑦ Pooling Layer

1. Convolution Layer (합성곱층)

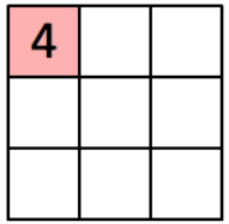
Convolution Layer에서는 입력과 필터를 합성곱 연산합니다.
Convolution은 이미지 처리에서 일반적으로 입력과 필터의 합성곱 연산을 말하는 것 입니다.

다음 그림은 2차원 입력데이터(shape (5,5))를 1개의 필터로 convolution 연산 수행 과정이며,
연산의 결과는 feature map입니다.

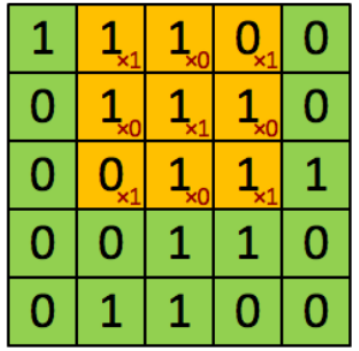
Unit 01 | CNN



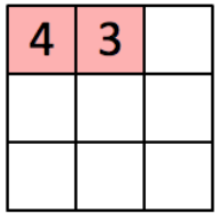
Image



Convolved Feature

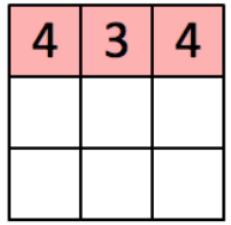


Image

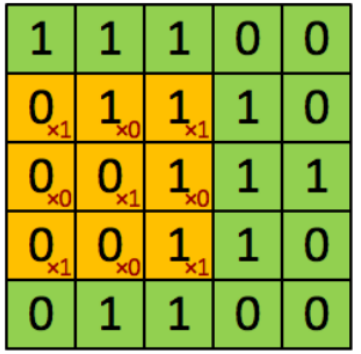


Convolved Feature

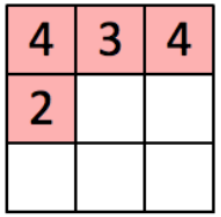
Image



Convolved Feature



Image



Convolved Feature

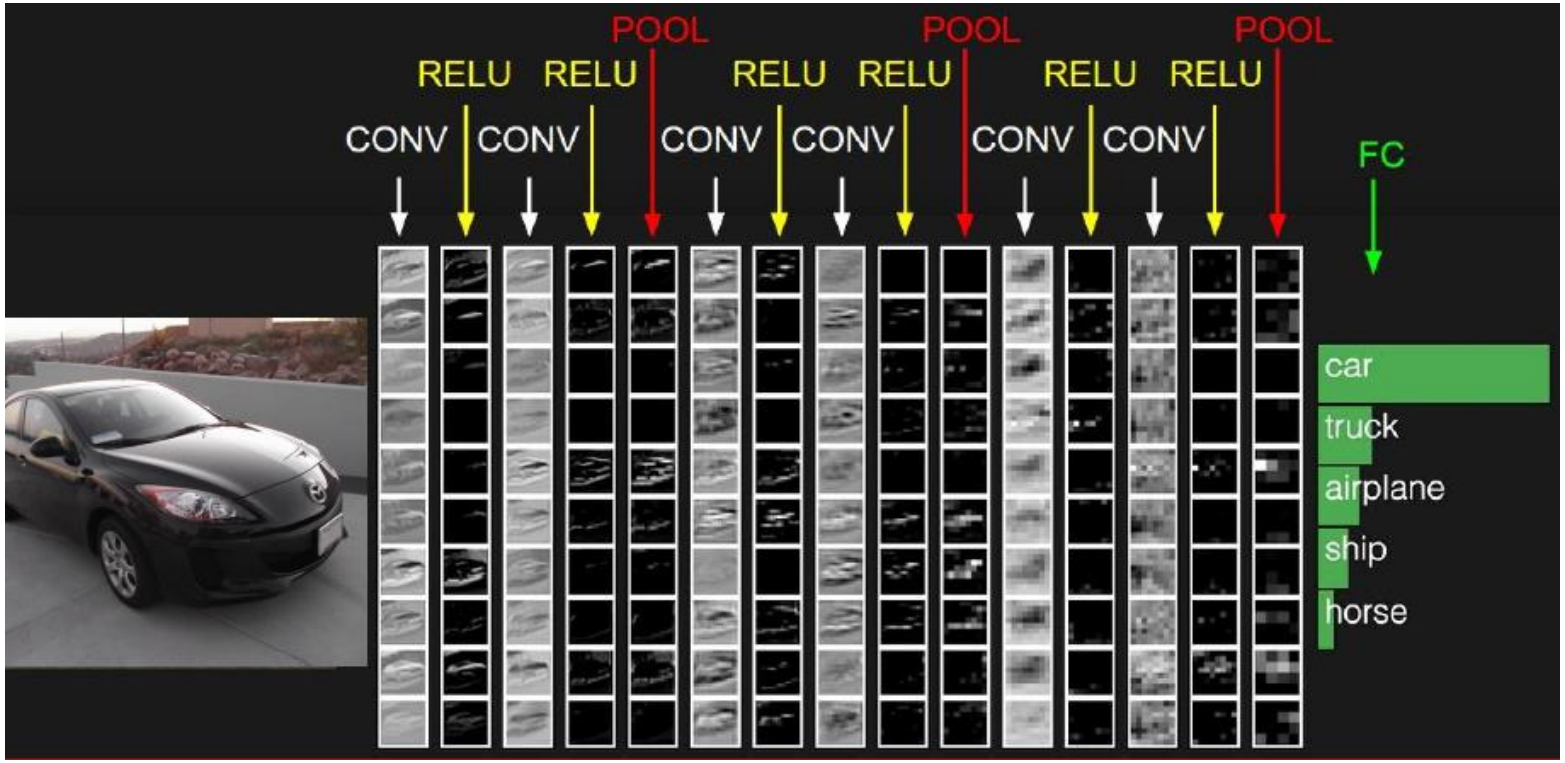
2. Feature Map, Activation Map

Convolution Layer이 입력 데이터를 필터가 순회하면서 합성곱을 통해서 만든 출력을 Feature map이라고 합니다. Feature map은 합성곱 계산으로 만들어진 행렬이고, Activation map은 feature map 행렬에 활성 함수(주로 Relu)를 적용한 결과입니다.

즉, convolution layer의 최종 출력 결과는 activation map입니다.
다만, 보통 feature map과 activation map을 혼용해서 사용하는 것 같습니다.

Unit 01 | CNN

Feature map은 합성곱 계산으로 만들어진 행렬이고,
Activation map은 feature map 행렬에 활성 함수(주로 Relu)를 적용한 결과입니다.



3. Channel (Depth)

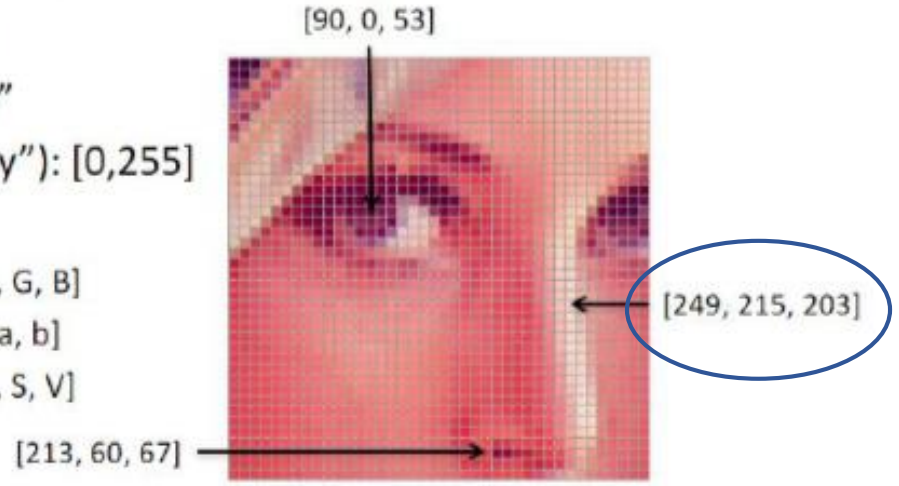
이미지의 픽셀 하나하나는 실수입니다.
컬러 이미지는 각 픽셀을 RGB 3개의 실수로 표현한 3차원 데이터입니다.
즉, 컬러 이미지는 R,G,B 3개의 채널로 구성되어 있다는 말이고, 'depth가 3이다'라는 말과 같은 말입니다.

반면 흑백 사진은 2차원 데이터로, 1개 채널로 구성됩니다.

높이가 39, 폭이 31인 컬러 사진의 shape은 (39, 31, 3)으로 표현하고,
높이가 39, 폭이 31인 흑백 사진의 shape은 (39, 31, 1)입니다.

Image

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale" (or "intensity"): [0,255]
    - "color"
      - RGB: [R, G, B]
      - Lab: [L, a, b]
      - HSV: [H, S, V]



4. Filter

필터는 일반적으로 3*3, 4*4 와 같은 정사각 행렬로 정의합니다.
CNN에서 학습의 대상이죠!

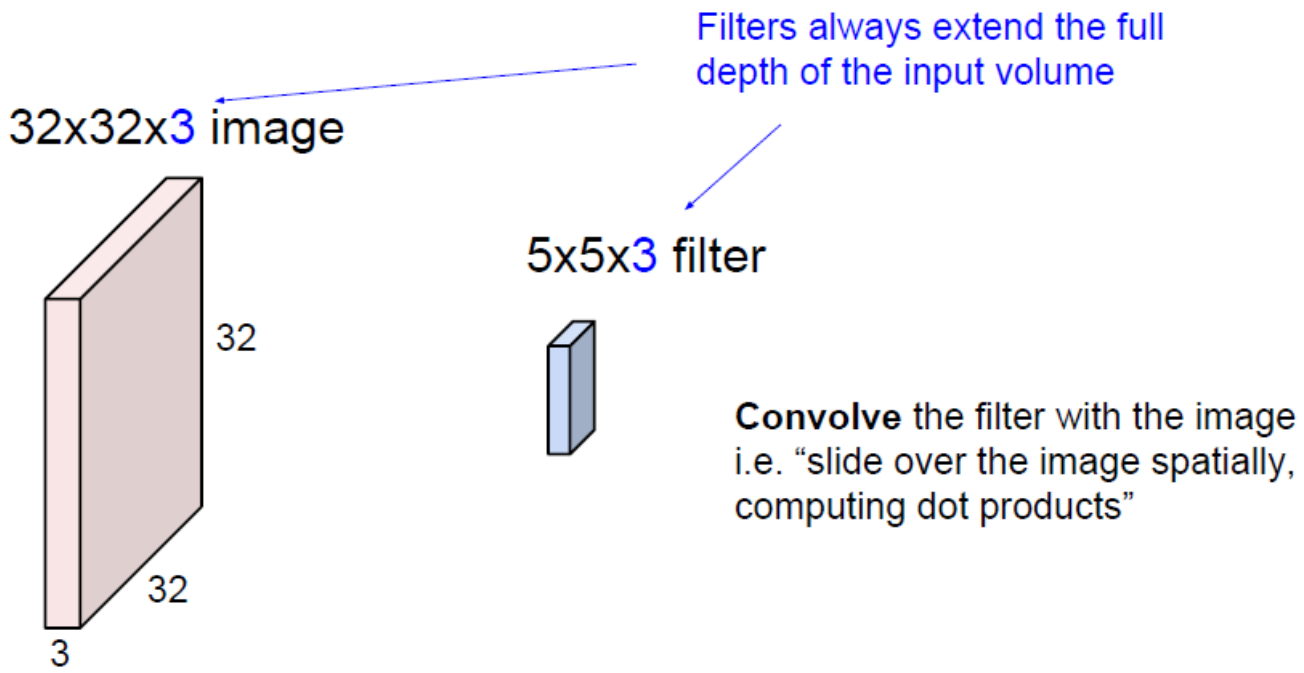
Convolution Layer의 입력 데이터가 channel이 3이라면, Filter의 depth도 3이어야 합니다.

반면, 입력 데이터의 channel 수와 상관없이 필터 1개당 1개의 feature map이 만들어집니다.

즉, 입력 데이터에 적용한 필터의 개수는 출력 데이터인 feature map의 채널이 됩니다.

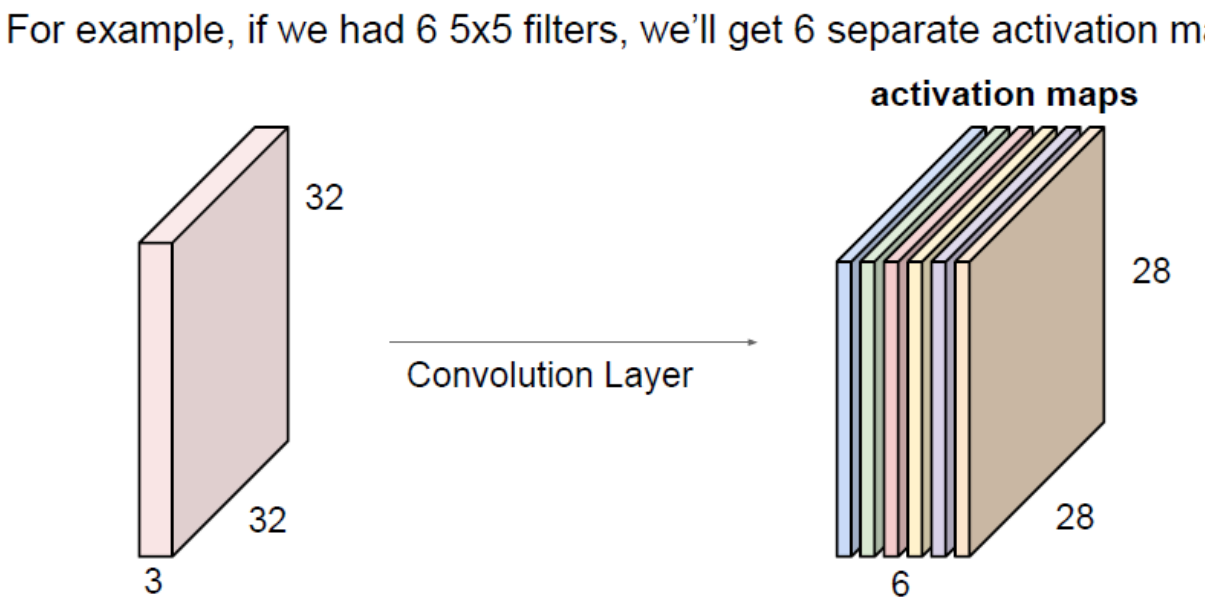
4. Filter

Convolution Layer의 입력 데이터가 channel이 3이라면, Filter의 depth도 3이어야 합니다.

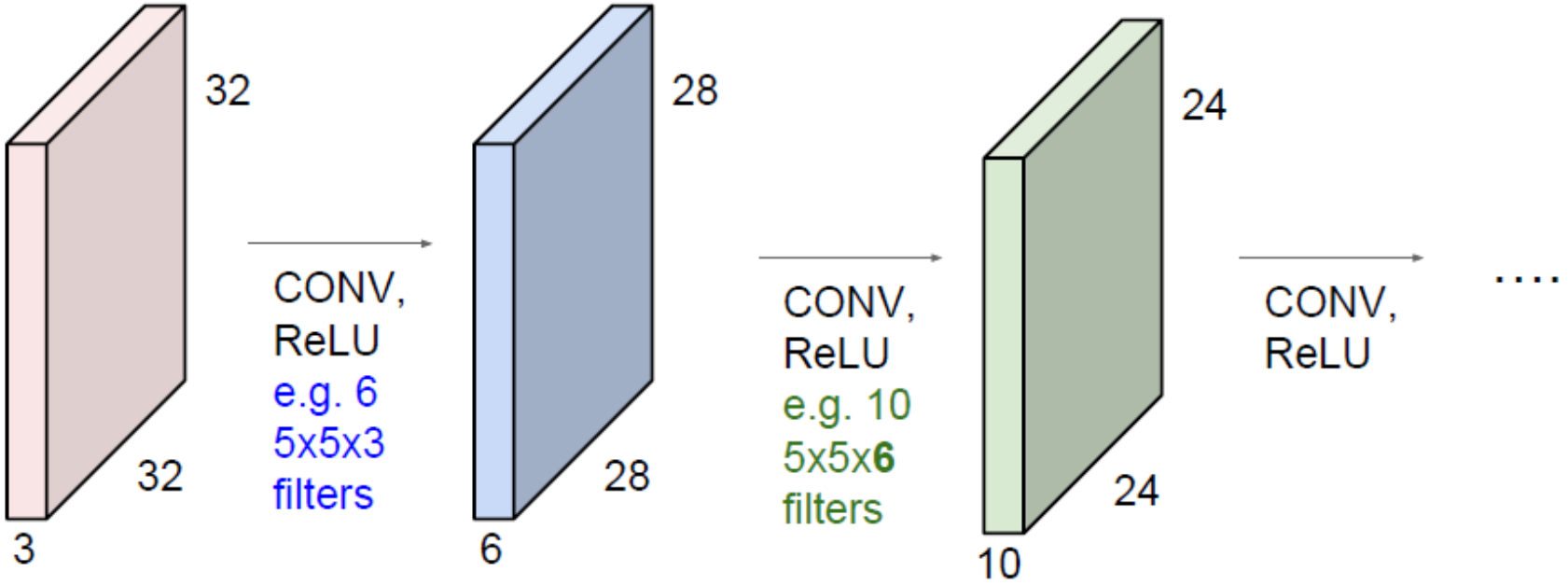


4. Filter

입력 데이터의 channel 수와 상관없이 필터 1개당 1개의 feature map이 만들어집니다.
즉, 입력 데이터에 적용한 필터의 개수는 출력 데이터인 feature map의 채널이 됩니다.



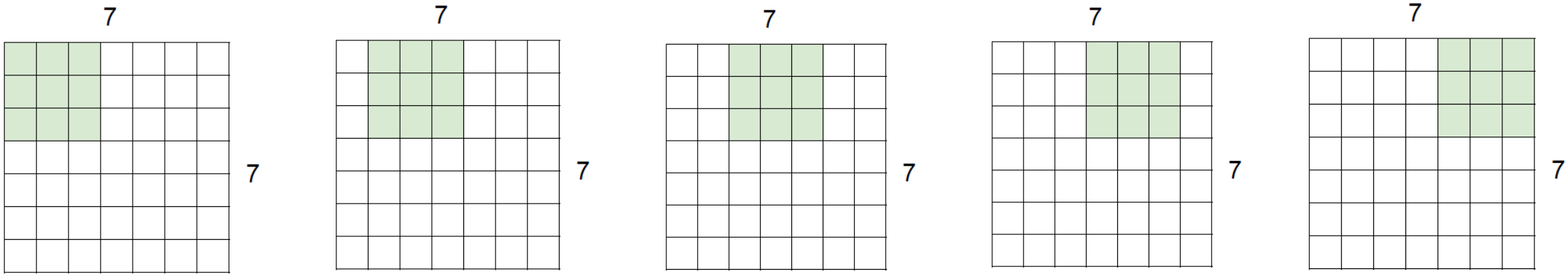
Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions



5. Stride

필터가 입력 데이터를 순회할 간격

7x7 input (spatially)
assume 3x3 filter

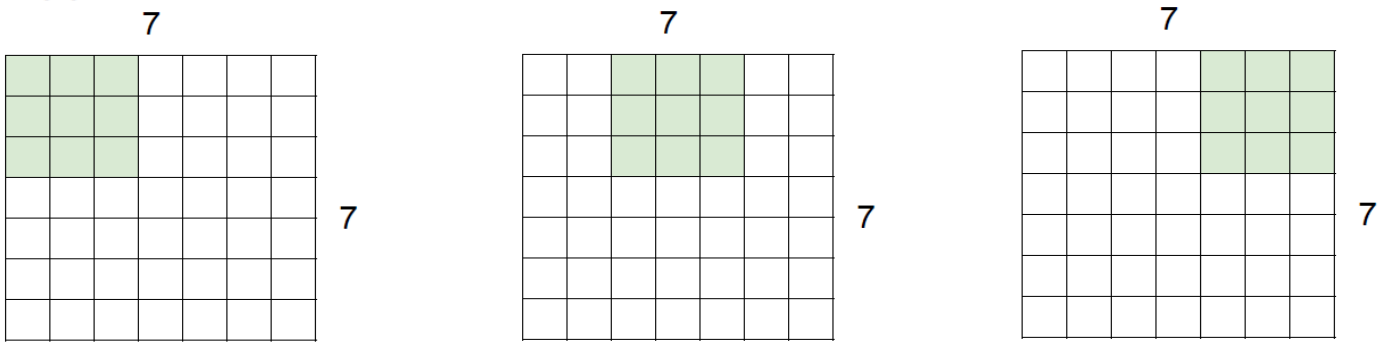


=> 5x5 output

5. Stride

필터가 입력 데이터를 순회할 간격

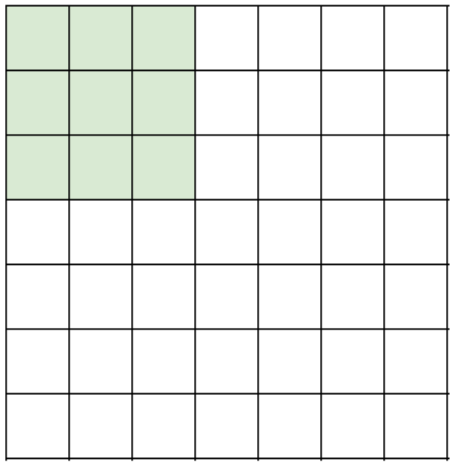
7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**



=> 3x3 output!

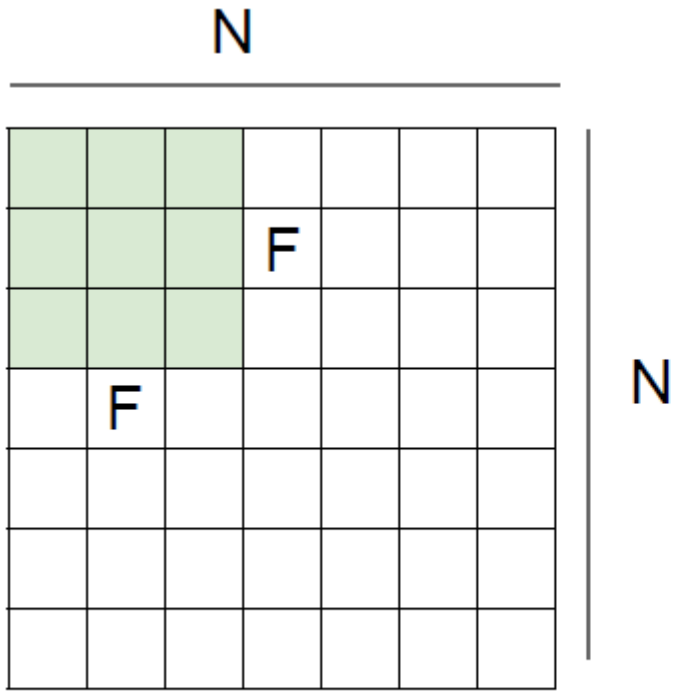
7x7 input (spatially)
assume 3x3 filter
applied **with stride 3?**

7



7

doesn't fit!
cannot apply 3x3 filter on
7x7 input with stride 3.



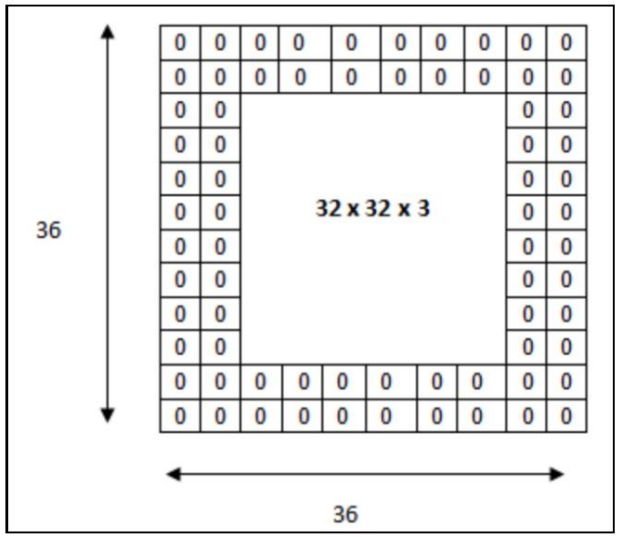
Output size:
(N - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 - 3)/1 + 1 = 5
stride 2 => (7 - 3)/2 + 1 = 3
stride 3 => (7 - 3)/3 + 1 = 2.33

6. Padding

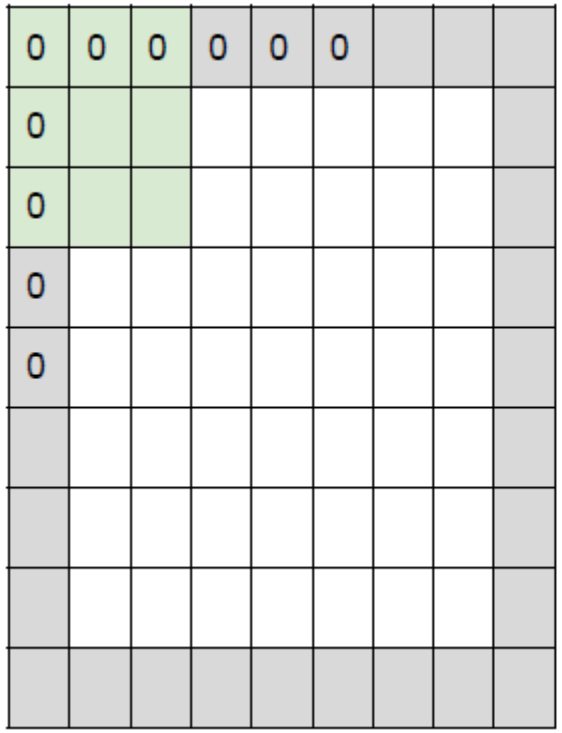
Convolutional Layer에서 filter와 stride의 작용으로 feature map의 크기는 입력데이터보다 작아지게 됩니다.
이를 방지하는 방법이 패딩입니다. 패딩은 입력데이터의 외곽에 지정된 픽셀만큼 특정 값으로 채워 넣는 것을 의미합니다.
보통 패딩 값으로 0을 채워 넣습니다.

패딩을 함으로써 얻을 수 있는 효과는
출력데이터의 사이즈 조절 뿐만 아니라, 인공 신경망이 이미지의 외곽까지 학습할 수 있도록 돕습니다.



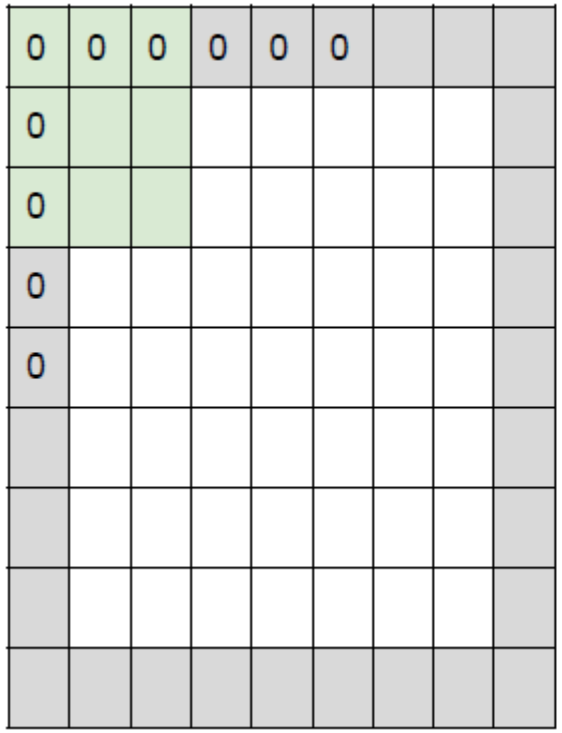
(32, 32, 3) 데이터에 2 pixel padding
→ (36, 36, 3)

Unit 01 | CNN



e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

(recall:)
(N - F) / stride + 1



e.g. input 7x7
3x3 filter, applied with **stride 1**
pad with 1 pixel border => what is the output?

7x7 output!
in general, common to see CONV layers with stride 1, filters of size FxF, and zero-padding with (F-1)/2. (will preserve size spatially)
e.g. F = 3 => zero pad with 1
F = 5 => zero pad with 2
F = 7 => zero pad with 3

7. Pooling Layer

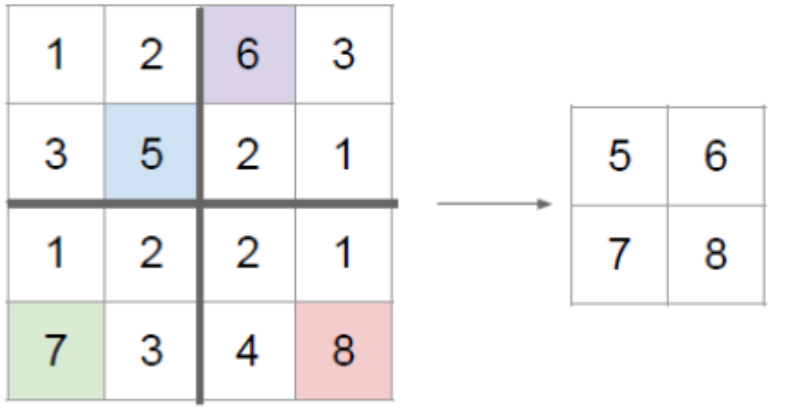
Convolution layer의 출력 데이터(Activation Map)를 입력으로 받아서
그것의 크기를 줄이거나 특정 데이터를 강조하는 용도로 사용됩니다.

Pooling Layer의 방법으로는 Max Pooling, Average Pooling, Min Pooling이 있는데,
CNN에서는 주로 Max Pooling을 사용합니다.

example1.

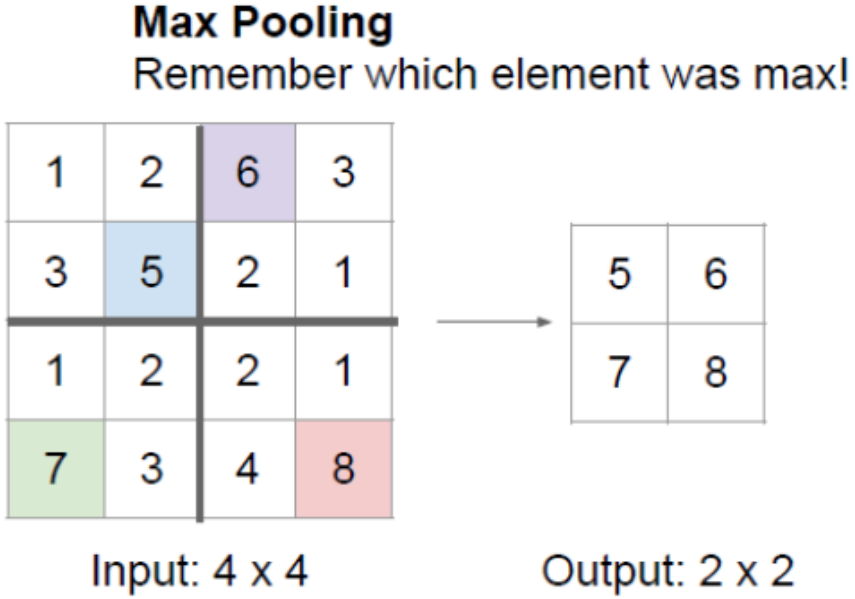
Max Pooling

Remember which element was max!

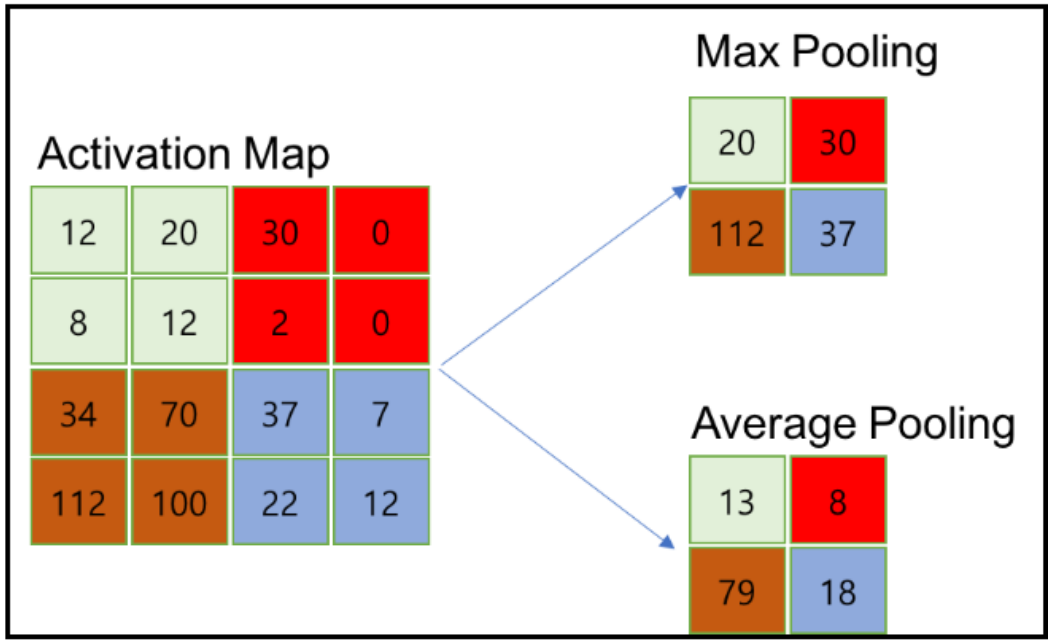


Input: 4 x 4 Output: 2 x 2

example1.



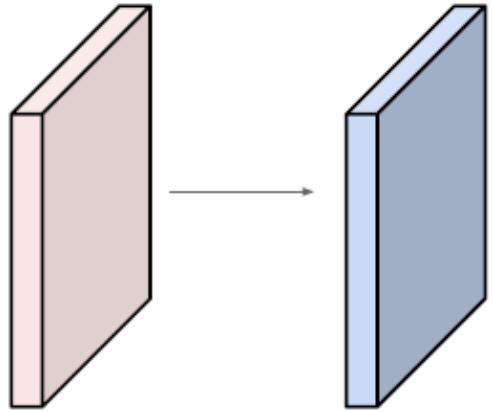
example2.



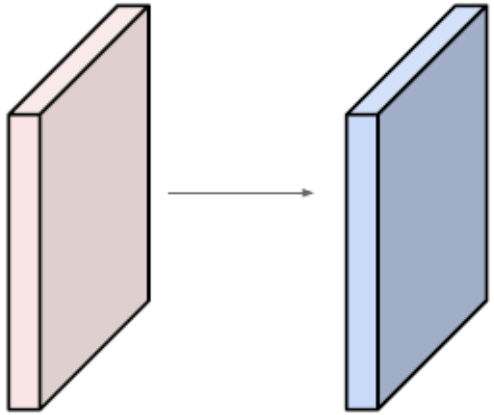
Examples time:

Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size: ?



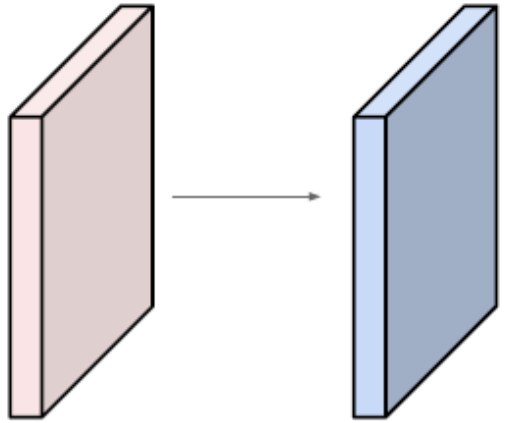
Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Output volume size:
(32+2*2-5)/1+1 = 32 spatially, so
32x32x10

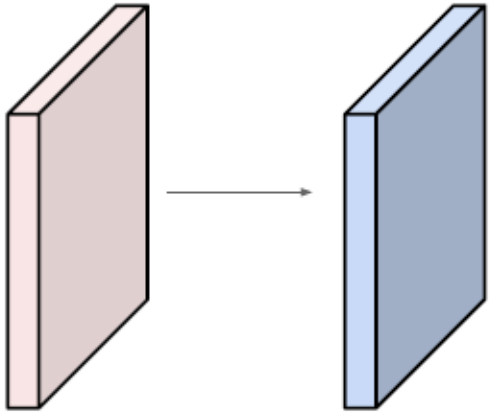
Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?

Examples time:



Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?
each filter has 5*5*3 + 1 = 76 params (+1 for bias)
=> 76*10 = **760**

요약

CNN은 이미지의 공간 정보를 유지하면서 인접 이미지와의 특징을 효과적으로 인식하고 강조하는 방식으로
이미지의 특징을 추출하는 부분과 이미지를 분류하는 부분으로 구성됩니다.

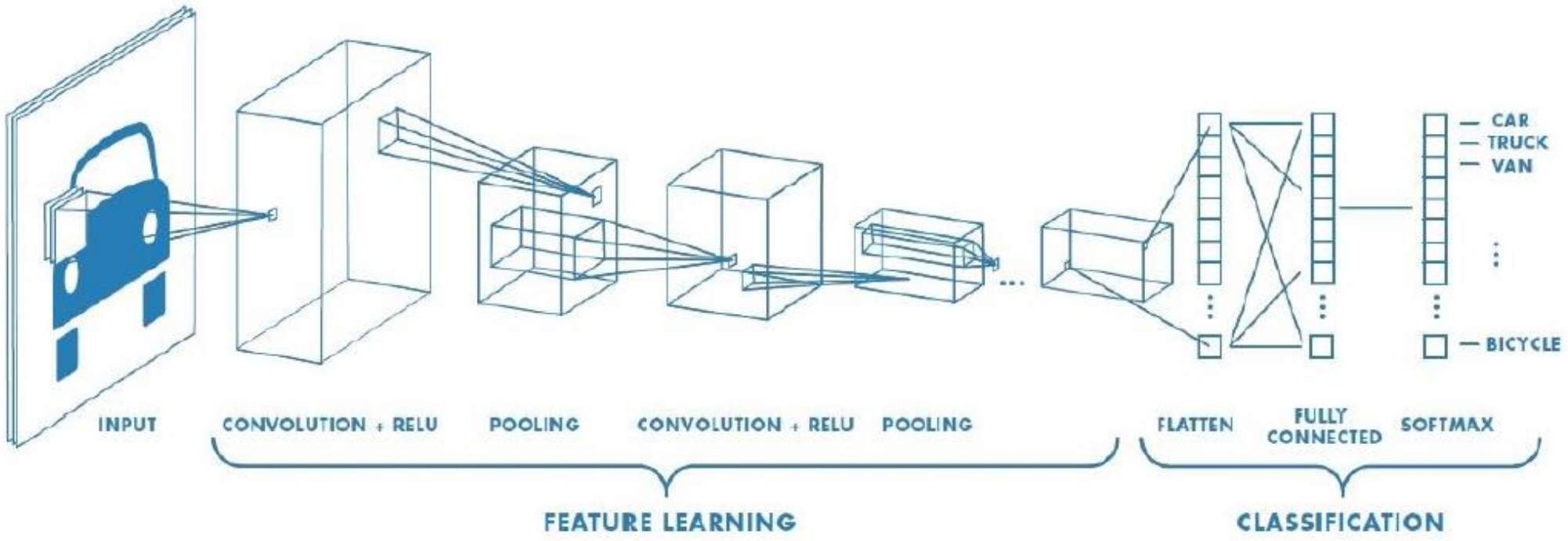
특징 추출 영역은 filter를 사용하여 파라미터를 최소화하면서
이미지의 특징을 찾는 convolution layer와 특징을 강화하고 모으는 pooling layer로 구성됩니다.

CNN은 filter의 크기, stride, padding과 pooling의 크기로 출력 데이터를 조절하고,
필터의 개수로 출력 데이터의 채널을 결정합니다.

학습의 대상은 필터입니다.

CNN은 같은 layer 크기의 fully connected neural network와 비교해 볼 때,
학습 파라미터의 양이 10% 규모입니다. Hidden layer가 깊어질수록 학습 파라미터의 차이는 더 커집니다.
CNN은 더 적은 학습 파라미터로 더 높은 인식률을 보입니다.

Unit 01 | CNN



CNN 구조: 이미지의 특징을 추출하는 부분과 이미지를 분류하는 부분으로 구성

2. GAN

GAN(Generative Adversarial Networks)이란??

직역하면 '생산적 적대 신경망' or '생성대립신경망'
비지도학습에 사용되는 인공지능 알고리즘으로,
데이터의 분포를 학습하는 생성 모델의 일종입니다.
제로섬 게임에서 서로 경쟁하는 두 개의 신경 네트워크 시스템에 의해 구현됩니다.

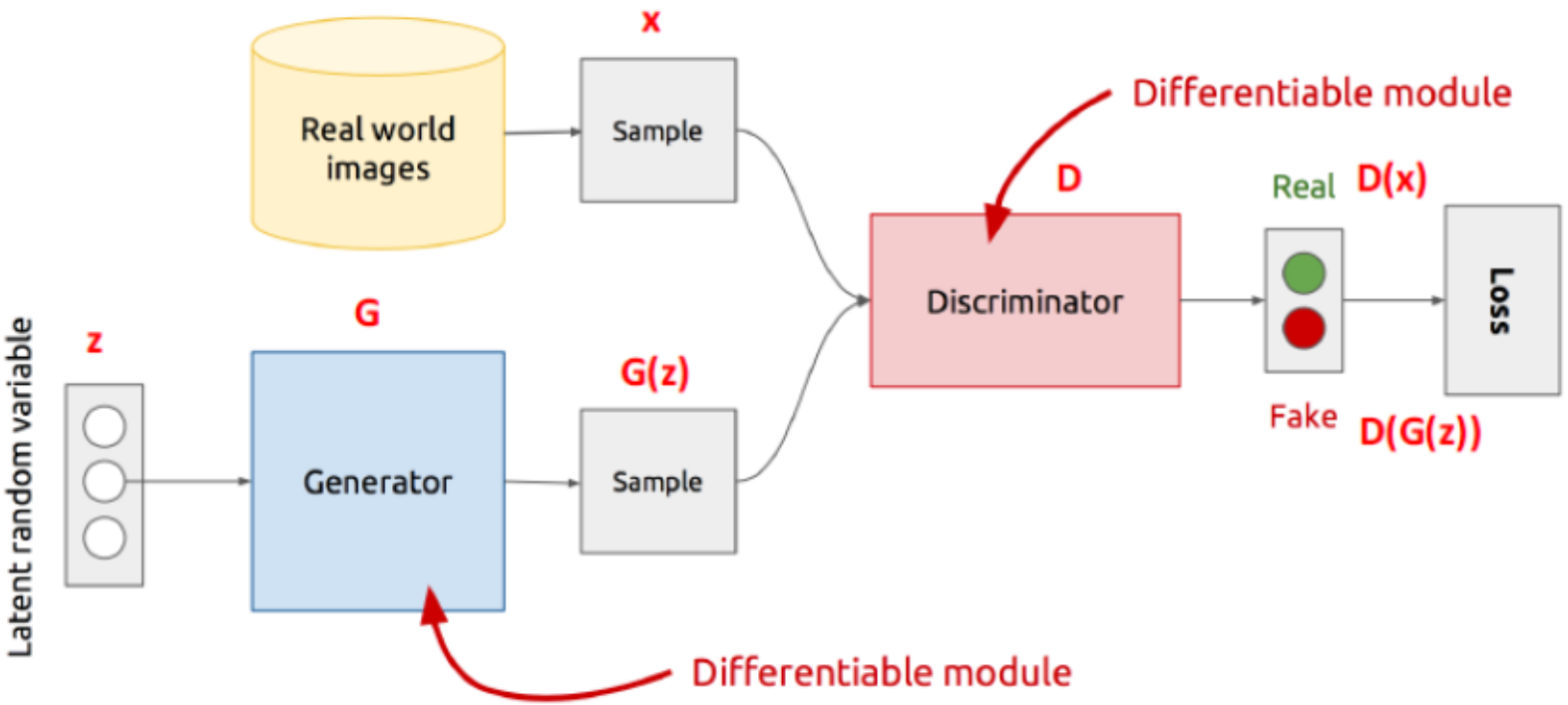
두 개의 신경 네트워크?!
이미지를 만들어내는 Generator(생성넷)와
만들어진 이미지는 평가하는 Discriminator(분류넷)가 서로 대립하면서
서로의 성능을 점차 개선해 나가는 것

Unit 02 | GAN

GAN의 창시자 Lan Goodfellow가 GAN의 개념을 지폐위조범과 경찰로 비유하여 설명했다.

위조지폐범(Generator)은 경찰을 최대한 열심히 속이려고 하고
다른 한편에서는 경찰(Discriminator)이 이렇게 위조된 지폐를 진짜와 감별하려고(Classify) 노력한다.
이런 경쟁 속에서 두 그룹 모두 속이고 구별하는 능력이 발전하게 되고,
결과적으로는 진짜 지폐와 위조 지폐를 구별할 수 없을 정도에 이른다는 것!

GAN's Architecture



Unit 02 | GAN

Generative Model, G는 우리가 갖고 있는 data x의 distribution을 알아내려고 합니다.
(위조지폐범이 지폐의 제작 방법을 알아내려고 한다고 보면 됩니다)
만약 G가 정확히 x의 distribution을 알아낸다면,
G가 생성한 fake data z는 real data인 x와 구별할 수 없겠죠!!

Discriminator model, D는 현재 자기가 보고 있는 data x가 real data인지, G가 생성한 것인지
각각의 경우에 대한 확률을 estimate합니다.

- D(x)=1 //x가 real data일 경우 1을 반환
- D(x)=0, D(G(z))=0 // x가 fake data일 경우 0을 반환, G가 생성한 fake data z를 G(z)로 표현

즉, D는 실수할 확률을 낮추기(mini)위해 노력하고 반대로 G는 D가 실수할 확률을 높이기(max)위해
노력하는데, 이 관계를 “minimax two-player game”, ”minimax problem” 이라고 합니다.

수식적으로 생각해보기

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))]$$

분류넷(Discriminator Model)이 출력하는 값 D(x)는 x가 real data일 확률이며 0과 1 사이의 값입니다.
logD(x)는 D(x)가 1일 때가 최대가 되고, 0에 가까워지면 음의 무한대로 발산합니다.

따라서 G가 고정된 상태에서 V(D,G)를 최적화하기 위해서는
Real data에 대해서는 D(x)가 1에 가까워야 하고,
생성넷(Generative Model)이 생성한 데이터 즉, fake data에 대해서는 D(x)가 0에 가까워야 합니다.

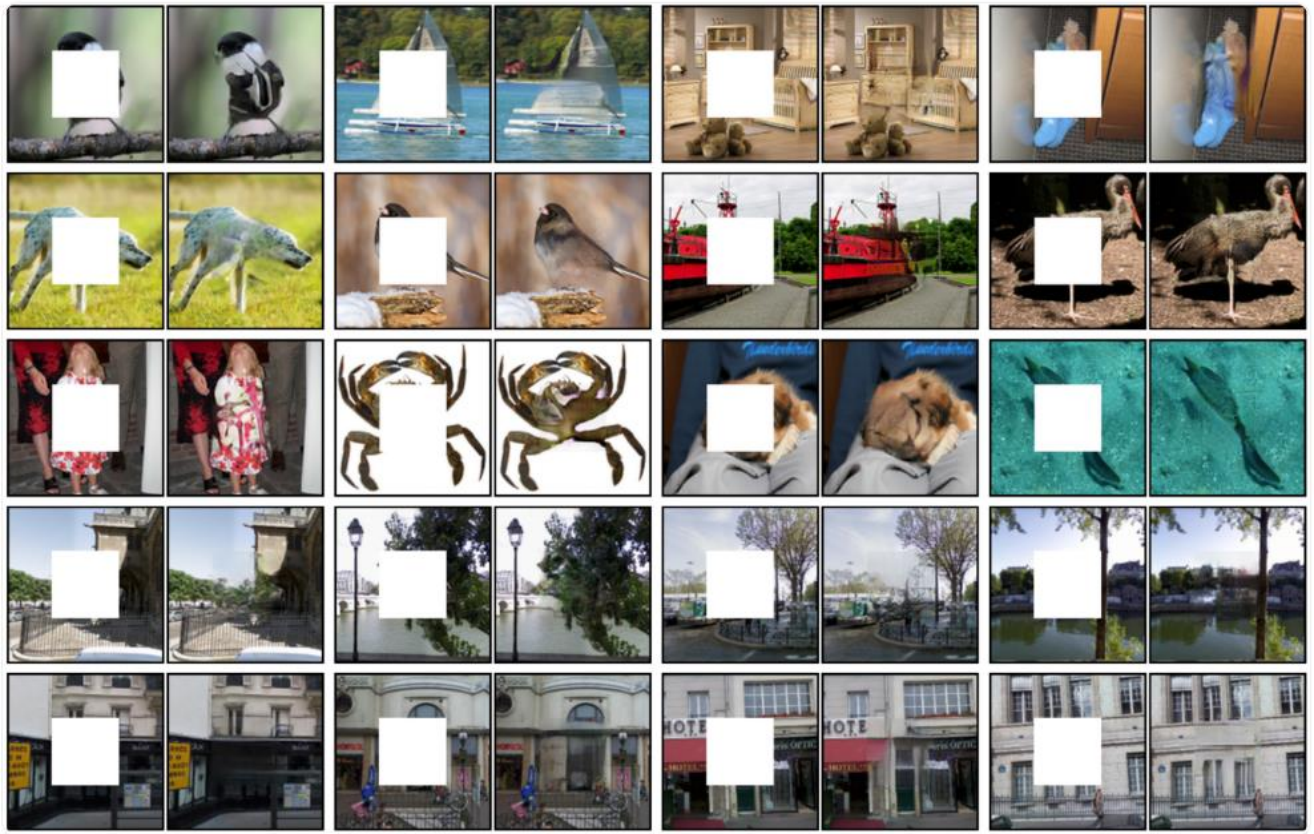
수식적으로 생각해보기

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))]$$

반대로 D가 고정된 상태에서는 생성넷이 만들어내는 fake data를 보고 분류넷이 real로 판별해야 합니다.
즉, log(1-D(G(z))) 의 1-D(G(z))가 0에 가까워져야 합니다.

이렇게 두 넷이 대립적으로 학습되면서 V(G,D)가 수렴하게 되고, D(x)가 항상 ½이 됩니다.
이 순간 분류넷인 D가 더 이상 가짜와 진짜를 구분할 수 없는 상황이 됩니다.

Unit 02 | GAN



GAN으로 이미지의 빈 공간을 채우는 예시

Unit 02 | GAN

요약

GAN은 생성 모델로서, 해상도 보정, 모자이크 제거, 동영상 생성 등에 강세를 보입니다.

분류 모델과 큰 차이가 있는데,
분류 모델은 서로 다른 클래스로 구분하기 위해 이미지에서 구분 경계선을 찾습니다.

이 차이로 생성 모델은 분류 모델로는 풀기 어려운 문제,
앞에서 봤듯이 어떤 이미지에 모자이크를 제거하는 문제를 해결하는데 유용합니다.
이 문제를 해결하기 위해서는 모자이크가 없는 부분을 조건으로 가려진 부분의 일반적인 분포를 이해해야 하기 때문입니다.

GAN의 개선 모델로 DCGAN, BiGAN 등이 있습니다.

Unit 03 | assignment

이번 과제는 CNN 연산을 직접 해보는 것 입니다.

Shape이 (720, 1280, 3)인 image가 input으로 들어왔다고 가정해봅시다.

Q1. Input에 3*3 convolution filter 6개를 돌렸을 때(stride=1, padding=0) activation map의 dimension은? 그리고 이때 filter 6개의 depth는 몇 일까요?

Q2. Input에 3*3 convolution filter 10개를 돌렸을 때(stride=1, padding=1) activation map의 dimension은? 그리고 이때 학습하는 총 weight는 몇 개일까요? (bias 제외)

Q3. Q2의 결과인 activation map에 1*1 convolution filter 5개를 돌렸을 때(stride=1, padding=0) activation map의 dimension은? 그리고 이때 filter 5개의 depth는 몇 일까요?

Q4. Q2의 결과인 activation map에 2*2 max pooling filter을 적용할 때 (stride=2) activation map의 dimension은?

Unit 04 | reference

cs231n강의자료:
http://cs231n.stanford.edu/slides/2018/cs231n_2018_lecture05.pdf

CNN:
http://taewan.kim/post/cnn/#fn:1

GAN:
http://jaejunyoo.blogspot.com/2017/01/generative-adversarial-nets-1.html

들어주셔서 감사합니다.