CNN - Convolutional Neural Network

MLP의 Fully Connected Layer는 1차원배열 ⇒ 공간정보유실 → 정보부족, 비효율, 정호소실

if image $(?,?,3)$ → $(n, ?×?×3)$

· CNN은 공간정보 유지한 상태로 학습 ⟶ 마지막에 Fully Connected
                                               why? 클래스분류 해줘야해

  - Fully Connected와 다른 점
                    · 입출력 데이터 형상유지
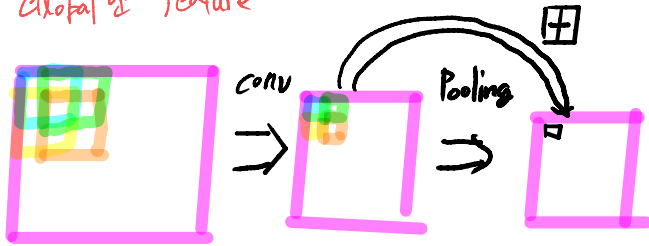   · 공간정보 유지, 인접에지의특징인식
                    · 복수필터 · Pooling Layer
                    · 학습 파라미터 적음

Conv layer n개 filter면
      출력은 n개 채널

· filter는 입력채널에 맞춰서
      1개의 필터맵반환

· Padding - 사이즈 조절, 외곽인식
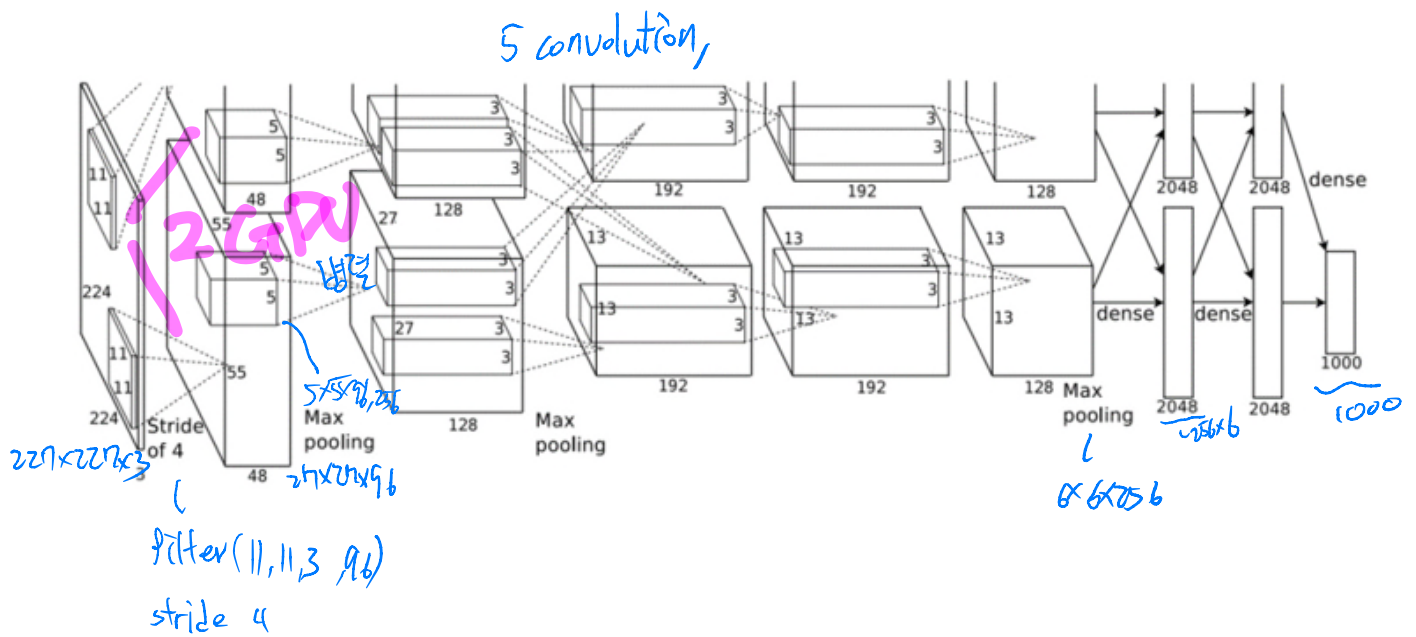· Pooling - 출력사이즈감소, 대표값강조
      · 학습파라미터 X, 크기감소, 채널2개로

∴ Global한 feature

conv → Pooling

· more deep, low parameter

ALEX Net

5 convolution,

227×227×3

filter(11,11,3,96)
stride 4

Stride of 4

5×5×48,256

Max pooling

128

27

128

Max pooling

192

192

192

192

13

13

13

13

6×6×256

256×6

1000

2개 GPU, 성능, 속도

1. ReLU - 빠른학습   sigmoid, tanh

2. Normalizatin

3. Overlapping Pooling   3×3, 2strid

4. Data Augmentation - overfitting 억제 위해

    1. 256 → 227 crop 2. RGB변환

5. Dropout

GoogLeNet Inception — Layer 추가 〈 무계정 Overfitting
연산량 증폭

○ NIN (Network In Network)

VGGNet

· 단일네트워크에서 Convolution의 개수가
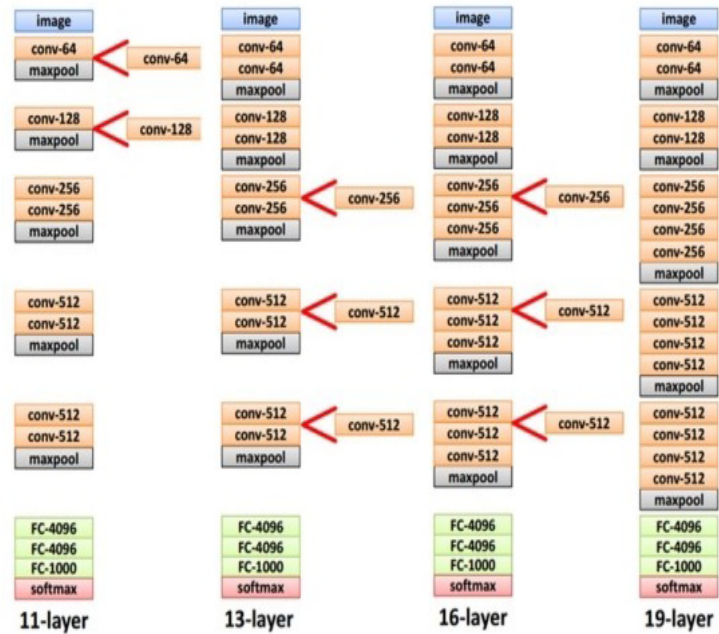 convolu 3×3, padding1, .
  resize는 maxpooling으로한

5×5   3×3 2개 같은size

But Parameter↓,
     more nonlinearity

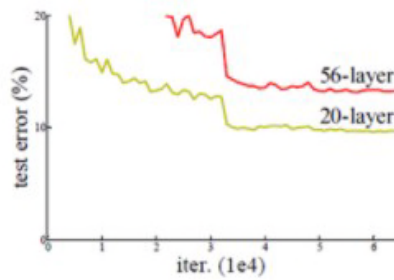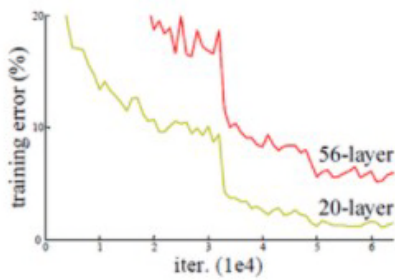| image	image	image	image
conv-64 ← conv-64	conv-64 conv-64	conv-64 conv-64	conv-64 conv-64
maxpool	maxpool	maxpool	maxpool
conv-128 ← conv-128	conv-128 conv-128	conv-128 conv-128	conv-128 conv-128
maxpool	maxpool	maxpool	maxpool
conv-256 conv-256	conv-256 ← conv-256 maxpool	conv-256 conv-256 ← conv-256 conv-256	conv-256 conv-256 conv-256 conv-256
maxpool		maxpool	maxpool
conv-512 conv-512	conv-512 conv-512 ← conv-512	conv-512 conv-512 ← conv-512 conv-512	conv-512 conv-512 conv-512 conv-512
maxpool	maxpool	maxpool	maxpool
conv-512 conv-512	conv-512 conv-512 ← conv-512	conv-512 conv-512 ← conv-512 conv-512	conv-512 conv-512 conv-512 conv-512
maxpool	maxpool	maxpool	maxpool
FC-4096 FC-4096 FC-1000 softmax	FC-4096 FC-4096 FC-1000 softmax	FC-4096 FC-4096 FC-1000 softmax	FC-4096 FC-4096 FC-1000 softmax
11-layer	13-layer	16-layer	19-layer

· 16~19 layer에서 정확도가 내려가

↳ 이를 어느정도 극복한게 Resnet
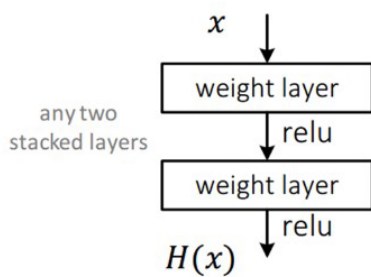
∴ 단순하고 성과가 좋아서 많이 사용됨

ResNet

layer가 깊어져
· gradient vanishing

평범CNN은
$H(x) = conv(conv(x))$

Convolution Layer 2번

거 (x) 얻기 위한 학습
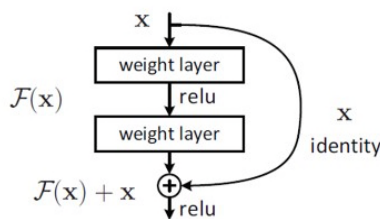
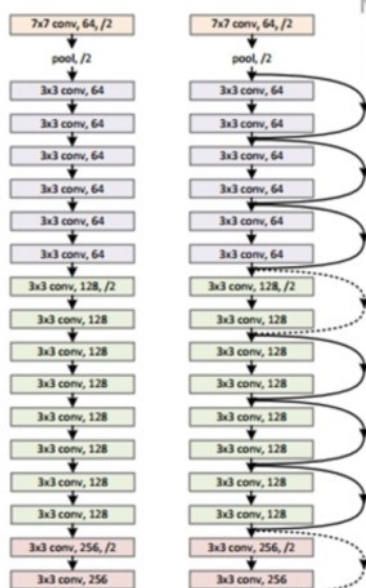

Resnet have "skip connection"
$H(x) = f(x) + x$
아웃풋에 그 전 결과 더하고 relu

$f(x) \neq 0$ 이 되는 학습
나머지 (residual)

· x 더 하기 ⇒ 연산중가X

plain net

ResNet

특징
1. 3x3 filter
2. 복잡도 외해, max pool
    hidden fc, dropout X
3. 출력 feature map 크기 같으면
    동일한 filter
4. 출력 feature map 절반서, 연산량보존위해
    filter 2배로
5. feature map 줄일떄
    pooling 대신, stride 2

출력결과
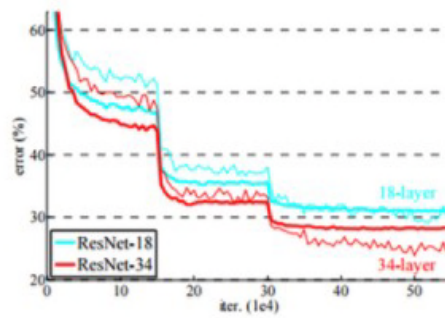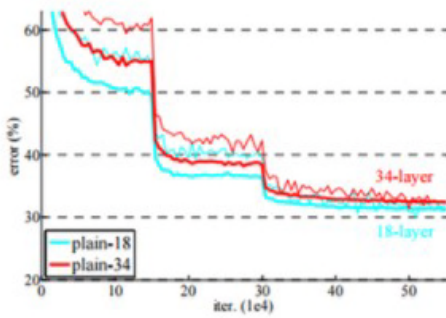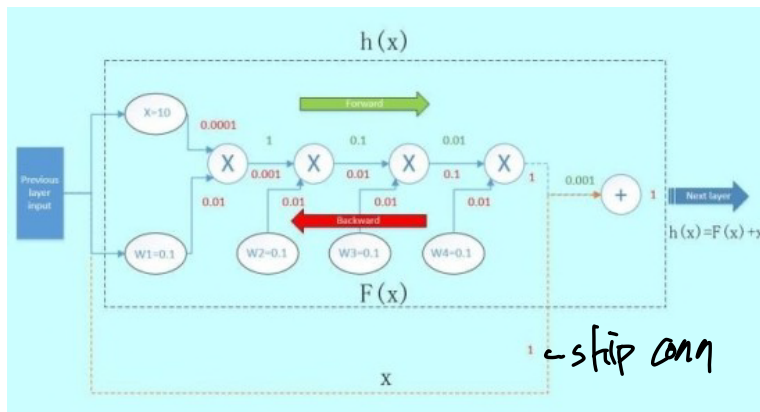

$$x_{L+1} = x_L + F(x_L)$$
$$x_{L+2} = x_{L+1} + F(x_{L+1})$$
$$= x_L + F(x_L) + F(x_{L+1})$$
$$x_L = x_1 + \sum_{i=L}^{L-1} F(x_i)$$
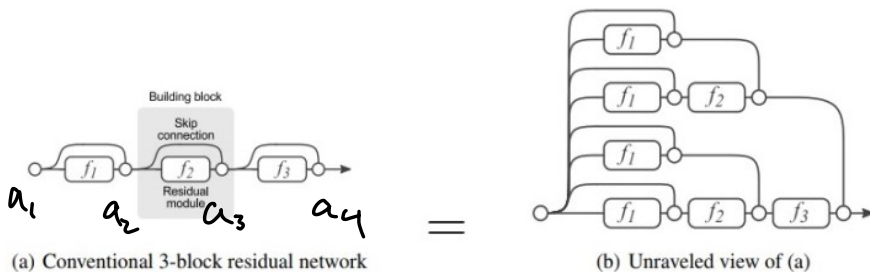

$-$ skip conn

$h(x)$

(How to avoid vanishing)

$[0.0001, 0.01] \to [1, 0.0001, 0.01]$

1이라는 gradient 추가

because relu $dh(x)/dx = 1$

(a) Conventional 3-block residual network = (b) Unraveled view of (a)

$$x_L + \sum_{i=L}^{L-1} f(x_i)$$

$f(1) = Conv(a_1)$
a_1

$a_2 = a_1 + f(1)$

$f(2) = conv(a_2)$
$\quad = conv(a_1 + f_1)$

$a_3 = a_2 + f_2$

$f(3) = conv(a_3)$
$\quad = conv(a_2 + f_2)$
$\quad = conv(a_1 + f_1 + f_2)$

$a_4 = a_3 + conv(a_1 + f_1 + f_2)$
$\quad = a_1 + f_1 + f_2$

$\Rightarrow$ 여러 network 앙상블

layer가 삭제되도 가능