CNN Architectures

AlexNet (2012)

conv1 → max pool 1 → norm 1 → conv2 → max pool 2 → norm 2
→ conv3 → conv4 → conv5 → max pool 3 → FC1 → FC2 → FC3

Input: 227 × 227 × 3

- First layer (conv1): 96 11×11 filters applied at stride 4

Output volume: (227 − 11)/4 + 1 = 55
55 × 55 × 96
Parameters: (11×11×3) × 96 = 35K

- Second layer (pool 1): 3×3 filters applied at stride 2

Output volume: (55−3)/2 + 1 = 27
27×27×96
~~Parameters~~

- first use of ReLU
- used Norm layers (not common now)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- lr 1e-2, reduced by 10 manually
- L2 weight decay
- 7 CNN ensemble

ZFNet

AlexNet finetuning 정확도

VGGNet

AlexNet (8 layers) → VGGNet (16, 19 layers)

Smaller filter 적용

∴ deeper, more non-linearities, fewer parameters

GoogleNet

Deeper networks with computational efficiency

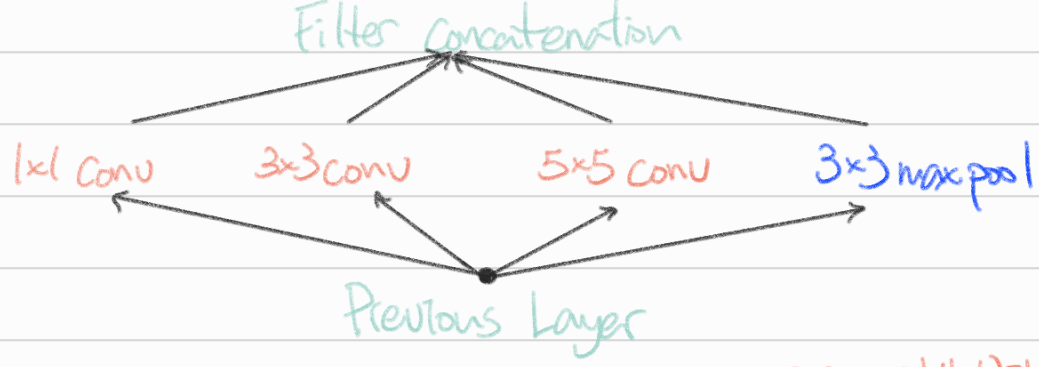
Inception module used

Apply parallel filter operations on the input from previous layer:

- Multiple receptive field sizes for convolution
- Pooling operation

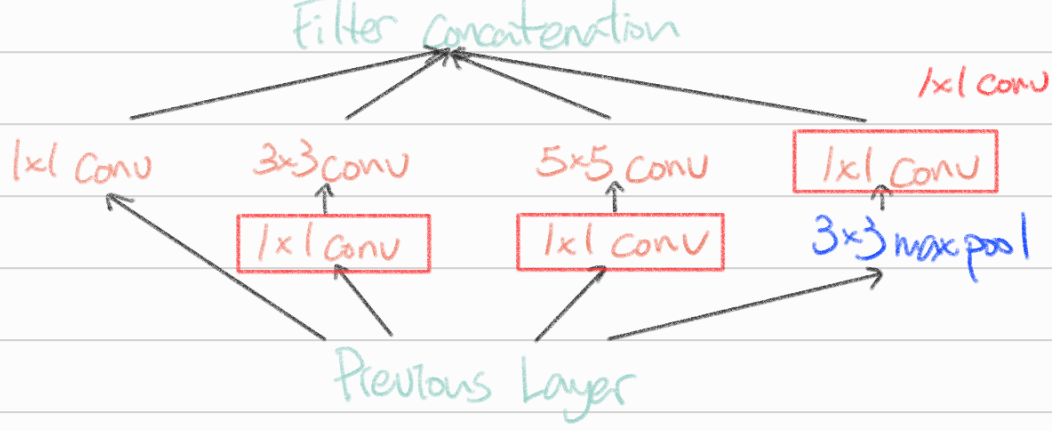
Concatenate all filter outputs together depth-wise

[Naive]



BUT 계산 복잡성 문제

[Inception module with dimension reduction]



1x1 conv "bottleneck" layers

ResNet (2015)

very deep networks using residual connections

152-layer

* Conv layer를 쌓으면 쌓을수록 성능이 좋은가? X

56-layer이 20-layer보다 train · test 모두에서 성능 안좋음.

deeper model performs worse, but it's not caused by overfitting

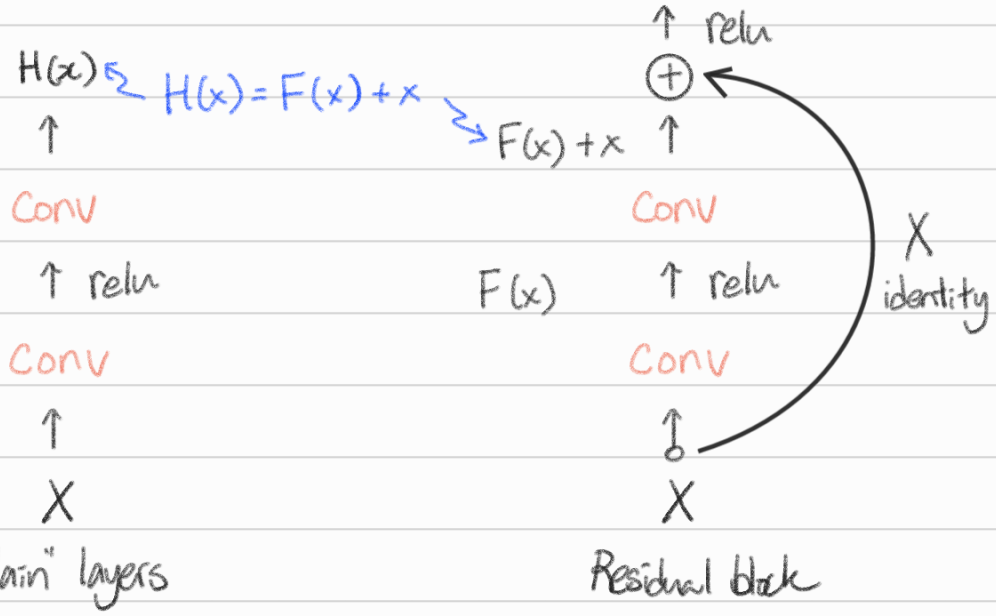
So, ResNet developers' 가설:

"the problem is optimization problem,

deeper models are harder to optimize"

Solution: Use network layers to fit a residual mapping 잔차 대응

instead of directly trying to fit a desired underlying map



Use layers to fit residual

$F(x) = H(x) - x$

instead of $H(x)$ directly

- Stack residual blocks
- Every residual blocks has two 3×3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2
- Additional conv layer at the beginning