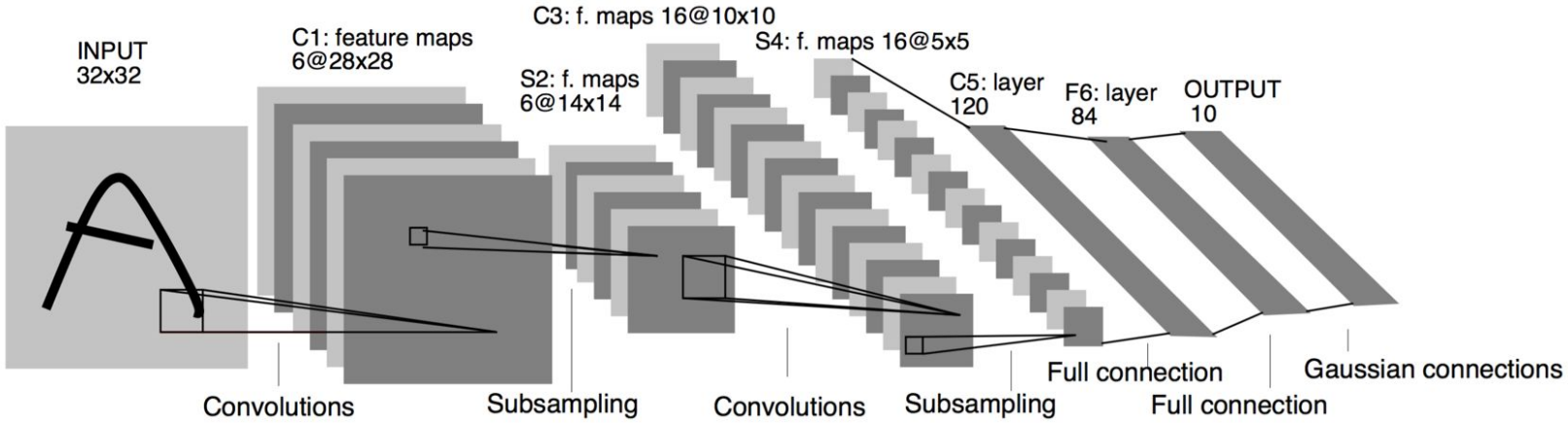
CNN Architecture

모두의연구소 Rubato Lab.
소준섭

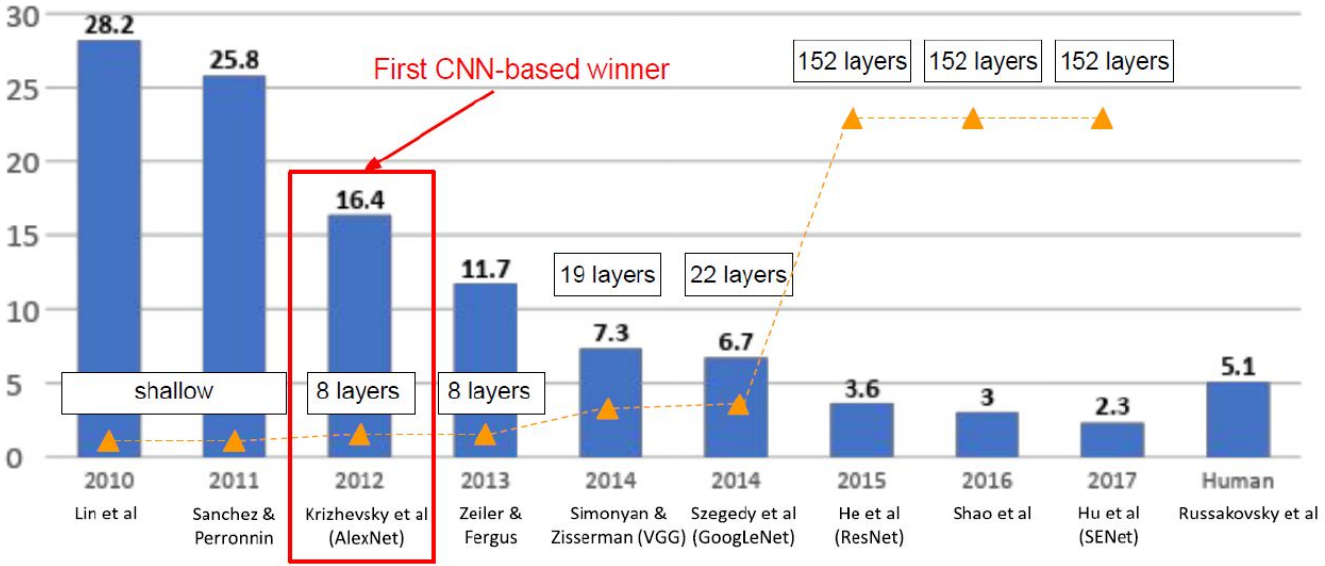
CNN Architecture - LeNet 5



- 7 Layer : [CONV-POOL-CONV-POOL-FC-FC-FC]
- Conv Layer : 5x5 filter, stride 1
- Pooling Layer : 2x2 average pooling, stride 2
- Sigmoid/tanh activation function
- 60k parameters

CNN Architecture - AlexNet

ImageNet ILSVRC(Large Scale Visual Recognition Challenge) winners

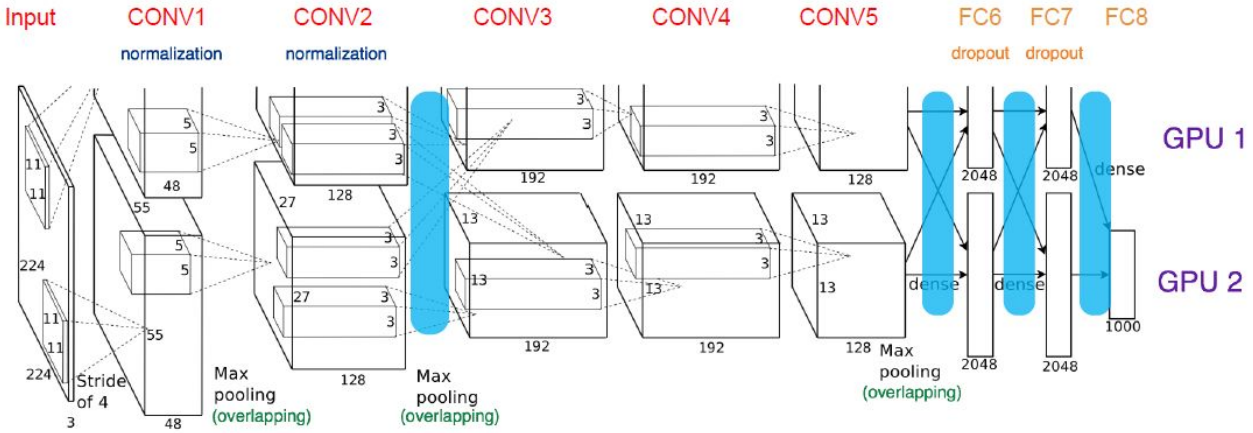


CNN Architecture - AlexNet

GPU 2개로 병렬 연산을 수행하기 위해 병렬 구조로 설계

227x227x3 이미지

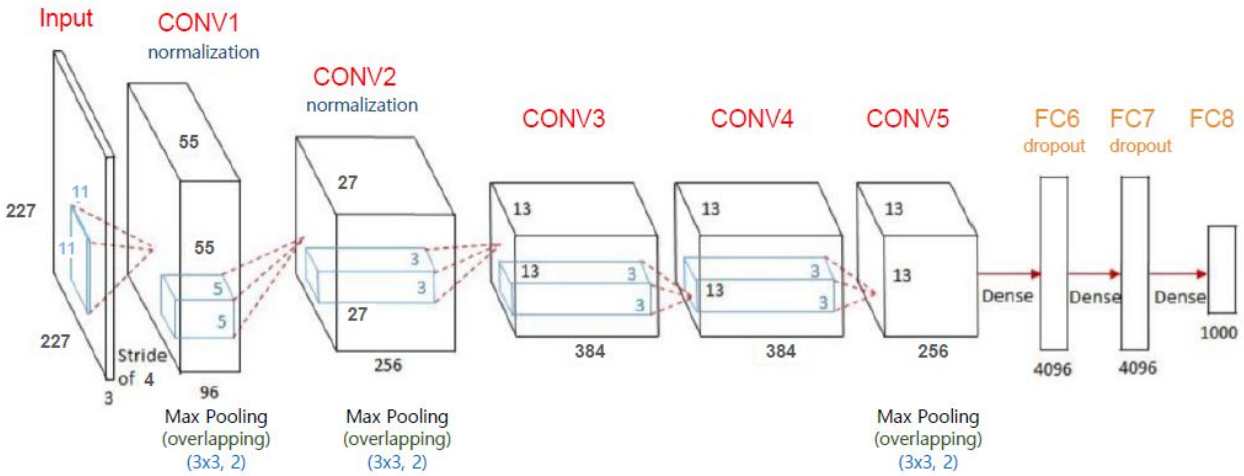
[참고]
- Input 크기 : 227x227x3
- 논문에 224x224x3으로 되어있는데 오타라고 함



- GTX 580 GPU (3 GB 메모리) 2개로 훈련 (5-6 days)
- 각 GPU 별로 feature map을 반반씩 처리
- CONV1, CONV2, CONV4, CONV5: 같은 GPU의 feature map만 연결
- CONV3, FC6, FC7, FC8: 전체 feature map 연결

CNN Architecture - AlexNet

아래 네트워크를 쪼개서 GPU로 병렬처리



CNN Architecture - AlexNet

Layer Name	Tensor Size	Weights	Biases	Parameters
Input Image	227x227x3	0	0	0
Conv-1	55x55x96	34,848	96	34,944
MaxPool-1	27x27x96	0	0	0
Conv-2	27x27x256	614,400	256	614,656
MaxPool-2	13x13x256	0	0	0
Conv-3	13x13x384	884,736	384	885,120
Conv-4	13x13x384	1,327,104	384	1,327,488
Conv-5	13x13x256	884,736	256	884,992
MaxPool-3	6x6x256	0	0	0
FC-1	4096×1	37,748,736	4,096	37,752,832
FC-2	4096×1	16,777,216	4,096	16,781,312
FC-3	1000×1	4,096,000	1,000	4,097,000
Output	1000×1	0	0	0
Total				**62,378,344**

62 MByte

https://www.learnopencv.com/number-of-parameters-and-tensor-sizes-in-convolutional-neural-network/

CNN Architecture - AlexNet

- 최초로 ReLU 사용

- Normalization 사용 (Local Response Normalization)
- Dropout 0.5
- Data Augmentation 사용
- L2 weight decay 5e-4
- 7개 모델 앙상블 : 18.2% -> 15.4%

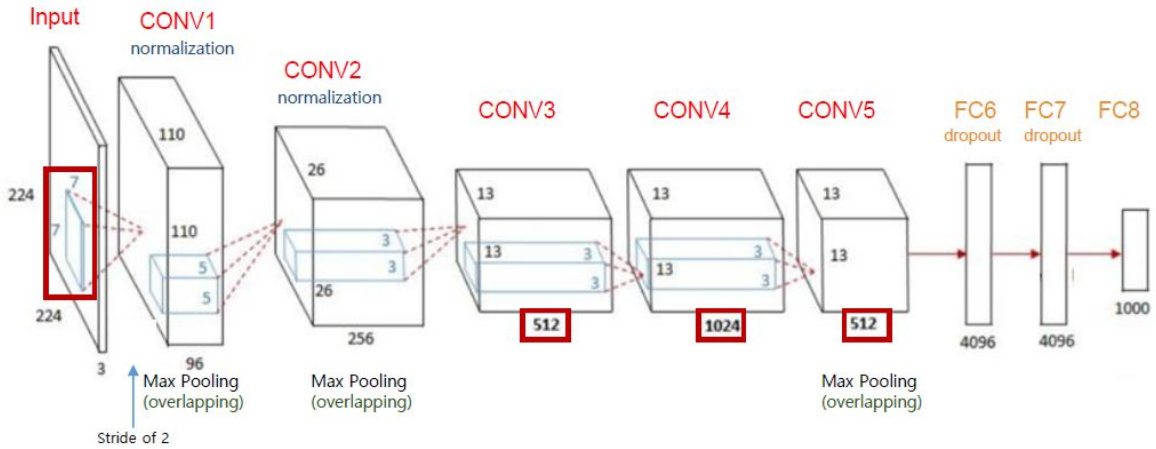
- 배치 크기 128
- SGD Momentum 0.9
- 학습률 1e-2, Validation Accuracy가 감소되지 않으면 수작업으로 10씩 감소

https://www.learnopencv.com/number-of-parameters-and-tensor-sizes-in-convolutional-neural-network/

CNN Architecture - ZFNet

학습 원리를 정확히 진단하고 Hyperparameter를 최적화하자!

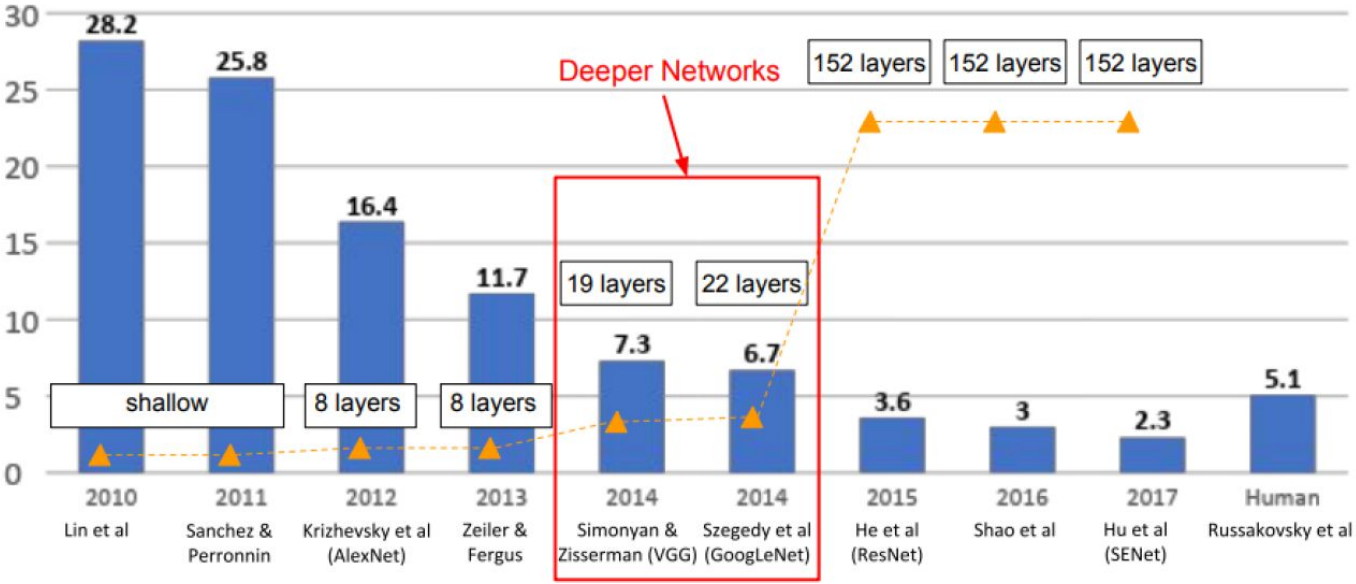
11.7% Top 5 error in ILSVRC'13



- AlexNet과 아키텍처는 동일하고 Hyperparameter를 조정해서 오류율을 개선
  - CONV1: (11x11 stride 4)를 (7x7 stride 2)로 변경
  - CONV3,4,5 : 384, 384, 256 filters 대신 512, 1024, 512 사용
- GTX 580 GPU 1개로 훈련 (12 days)
  - 단, AlexNet은 15 million images로 훈련한 반면 ZFNet 1.3 million images만 사용

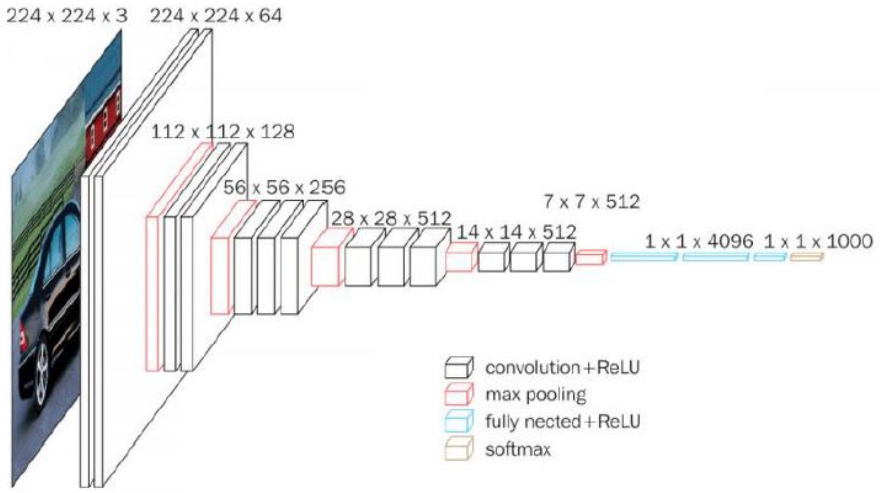
CNN Architecture - VGGNet

ImageNet ILSVRC(Large Scale Visual Recognition Challenge) winners



CNN Architecture - VGGNet

작은 filter를 사용해서 네트워크를 더 깊게 만들자!



- 같은 크기의 필터 사용
  - 3x3 CONV stride 1, pad 1
  - 2x2 MAX POOL stride 2

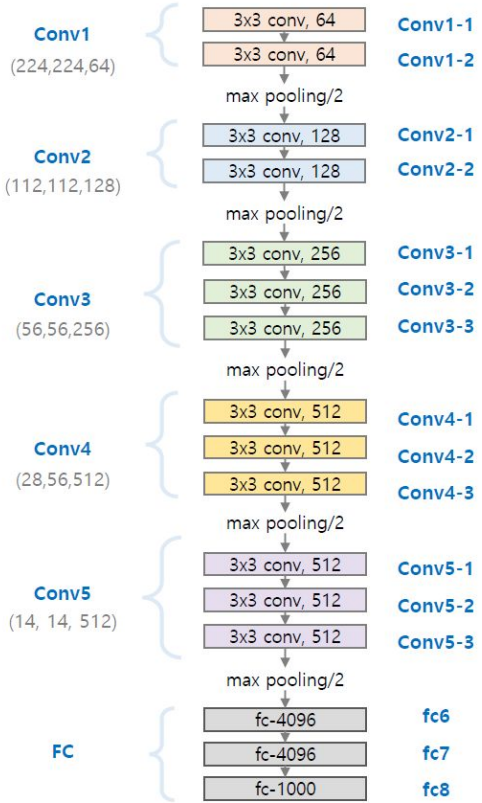
7.3% Top 5 error in ILSVRC'14
ILSVRC'14 classification 부분 2위, localization 부분 1위

"Very Deep Convolutional Networks for Large-Scale Image Recognition"

CNN Architecture - VGGNet

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

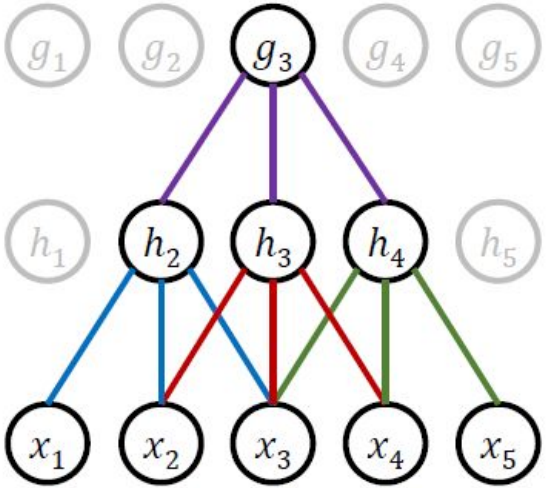
VGG16



"Very Deep Convolutional Networks for Large-Scale Image Recognition"

CNN Architecture - VGGNet

3 Convolution Filter의 Receptive Field (1D)



Receptive Field = 5

Receptive Field = 3

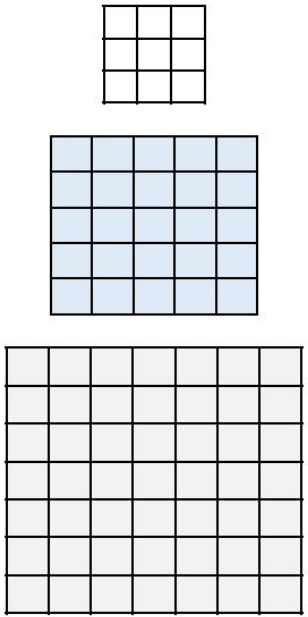
filter size =3
stride =1

계층을 깊이 쌓으면 Receptive Field를 키울 수 있다.

"Very Deep Convolutional Networks for Large-Scale Image Recognition"

CNN Architecture - VGGNet

Receptive Field 크기

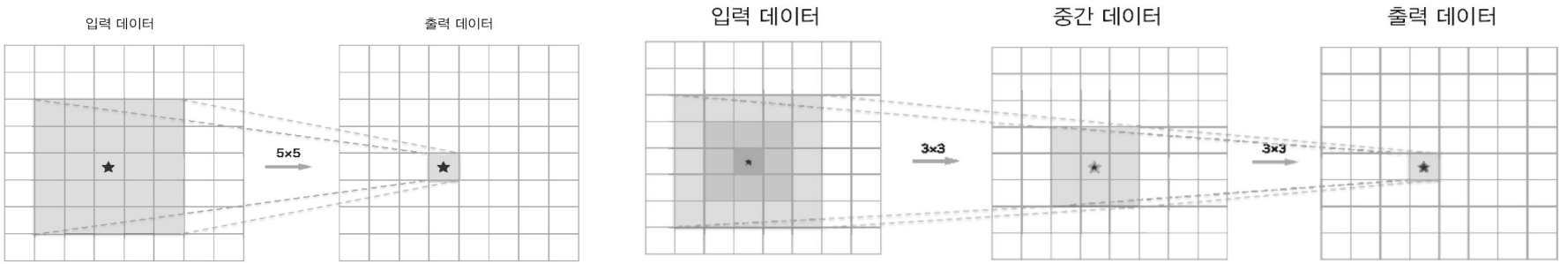


	3x3 필터	3x3 필터 x 깊이 1
파라미터 수 :	9 개	1x9 = 9 개
	5x5 필터	**3x3 필터 x 깊이 2**
파라미터 수 :	25 개	2x9 = 18 개
	7x7 필터	**3x3 필터 x 깊이 3**
파라미터 수 :	49 개	3x9 = 27 개

작은 filter를 사용하면 파라미터를 줄일 수 있다!

"Very Deep Convolutional Networks for Large-Scale Image Recognition"

CNN Architecture - VGGNet



- 5x5 와 (3x3)x2 필터는 같은 영역(Receptive Field)을 처리
- 층이 깊어져 ReLU 같은 Activation function(비선형성) 추가

CNN Architecture - VGGNet

INPUT: [224x224x3] memory: 224*224*3=150K params: 0
CONV3-64: [224x224x64] memory: 224*224*64=3.2M params: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64] **memory: 224*224*64=3.2M** params: (3*3*64)*64 = 36,864
POOL2: [112x112x64] memory: 112*112*64=800K params: 0
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128] memory: 112*112*128=1.6M params: (3*3*128)*128 = 147,456
POOL2: [56x56x128] memory: 56*56*128=400K params: 0
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824
POOL2: [28x28x256] memory: 28*28*256=200K params: 0
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512] memory: 14*14*512=100K params: 0
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512] memory: 7*7*512=25K params: 0
FC: [1x1x4096] memory: 4096 **params: 7*7*512*4096 = 102,760,448**
FC: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216
FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

초기 Conv 계층에서 메모리 사용이 집중됨

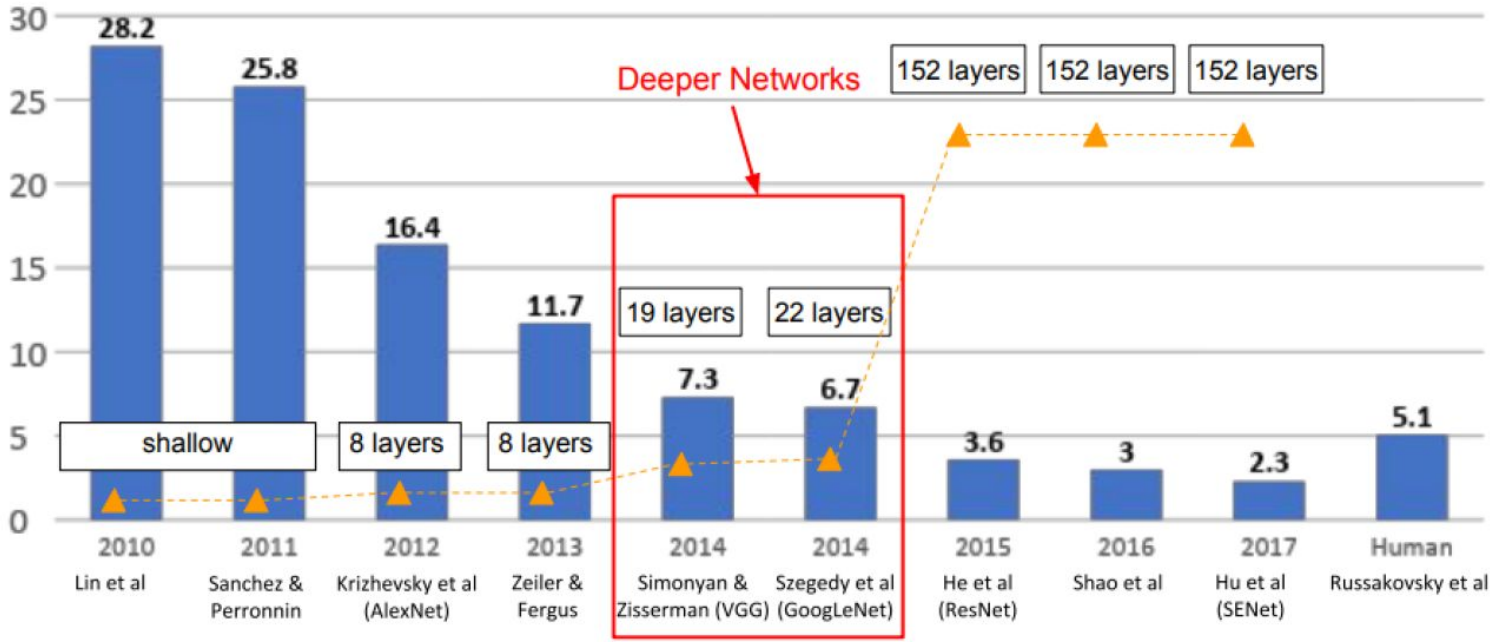
마지막 FC 계층에 파라미터 사용이 집중됨

- VGG 16 보다 VGG 19 가 메모리가 사용이 많지만, 성능은 조금 더 좋다.

TOTAL memory: 24M * 4 bytes ~= 96MB / image (for a forward pass)
TOTAL params: 138M parameters

CNN Architecture - GoogLeNet

ImageNet ILSVRC(Large Scale Visual Recognition Challenge) winners

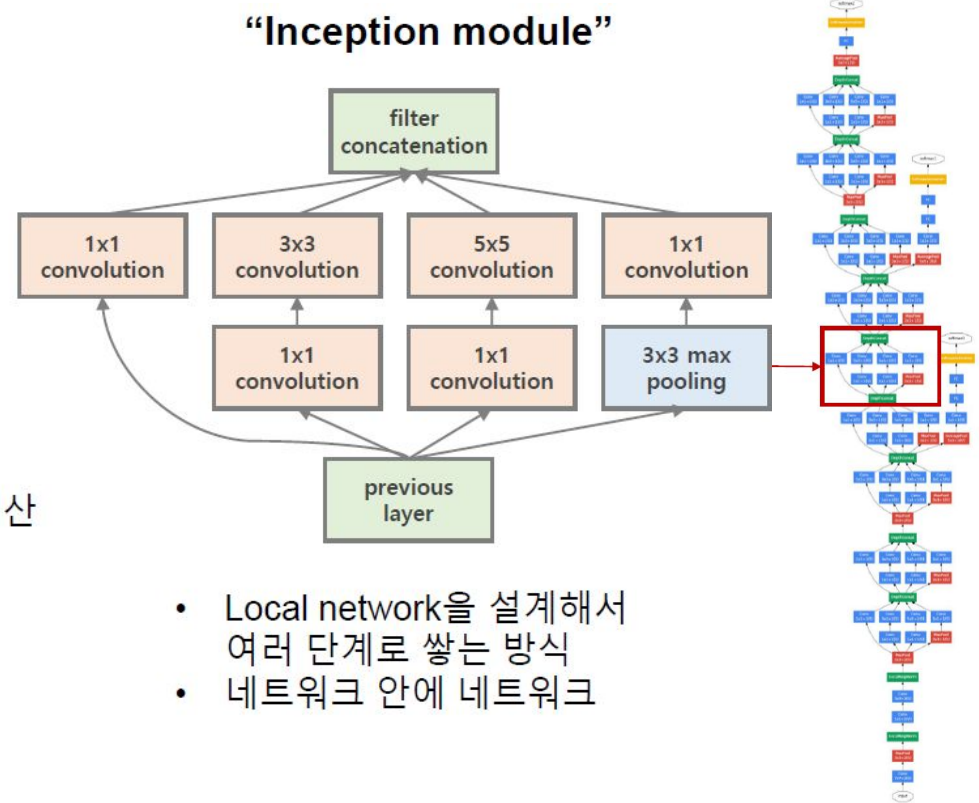


CNN Architecture - GoogLeNet

계산 효율성을 갖춘 깊은 네트워크

6.7% Top 5 error in ILSVRC'14

- 전체 22 계층
- Inception 모듈을 도입해서 효율적으로 계산
- FC layers 제거
- 5M 파라미터 (AlexNet보다 12배 작음)



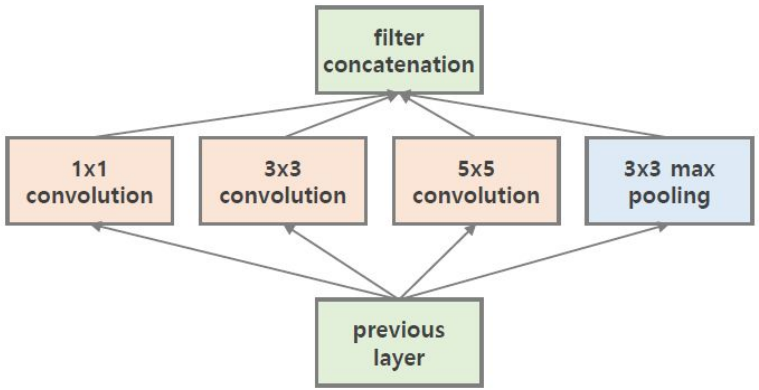
- Local network을 설계해서 여러 단계로 쌓는 방식
- 네트워크 안에 네트워크

"Going Deeper with Convolutions"

CNN Architecture - GoogLeNet

모듈 내에서 역할을 여러 개로 분리하고 역할 별로 연결을 분리하는 방식으로 설계

Naive Inception module



역할의 분리
- Convolution 연산 : (1x1, 3x3, 5x5)
- Pooling 연산 (3x3, Stride1)

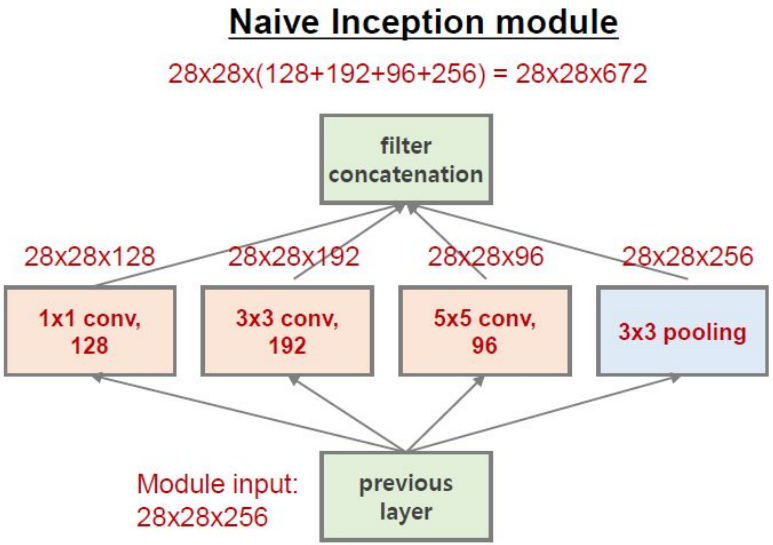
희소 연결
- 필요한 역할에 따라 연결이 되므로 희소 연결을 갖게 됨

조밀 연산
- 연산 효율을 높이기 위해 역할 별로 분리된 출력을 다시 합침

계산량이 많다는 문제점이 있음!

"Going Deeper with Convolutions"

CNN Architecture - GoogLeNet



Conv Ops:
[1x1 conv, 128] [28x28x256]x[1x1]x128
[3x3 conv, 192] [28x28x256]x[3x3]x192
[5x5 conv, 96] [28x28x256]x[5x5]x96
Total: 854M ops

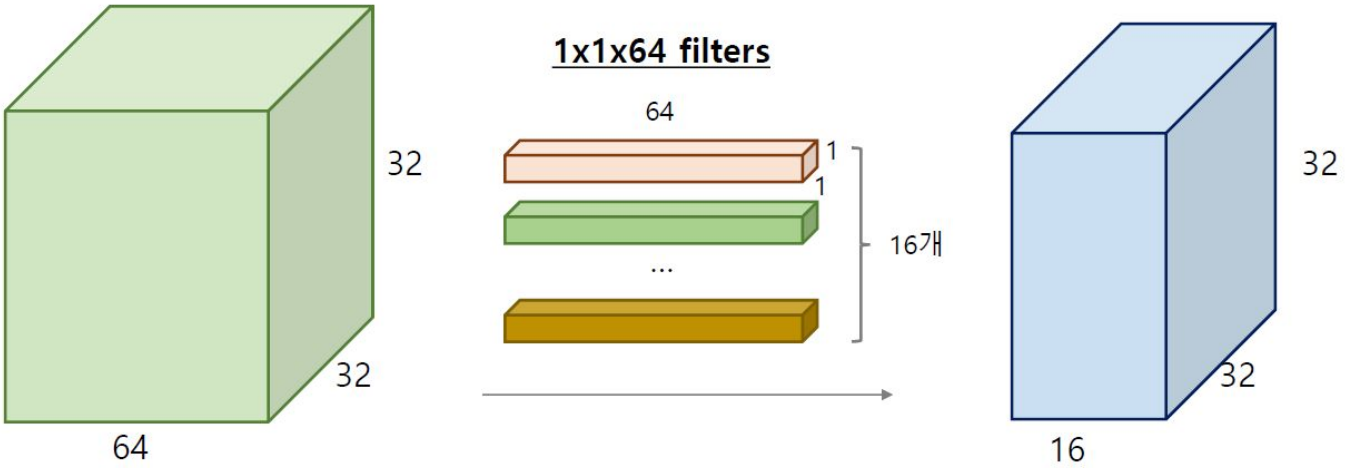
Issue
- 계산양이 매우 많음
- Feature map depth가 점점 증가

Pooling layer는 feature depth를 유지하기 때문에 filter concat 후 총 depth가 점점 증가하게 됨

"Going Deeper with Convolutions"

CNN Architecture - GoogLeNet

“bottleneck” layer를 사용해서 feature depth를 줄이자!

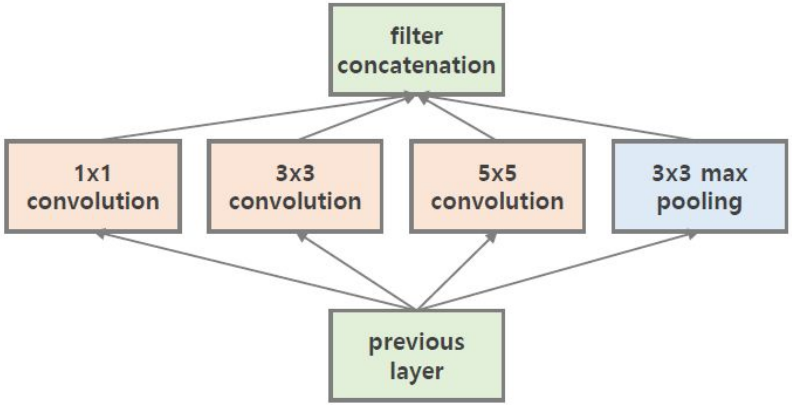


- 차원 축소를 하면서 각 채널의 가중치를 학습하는 방식
- Activation map의 depth를 줄여서 계산의 효율성을 높일 수 있음

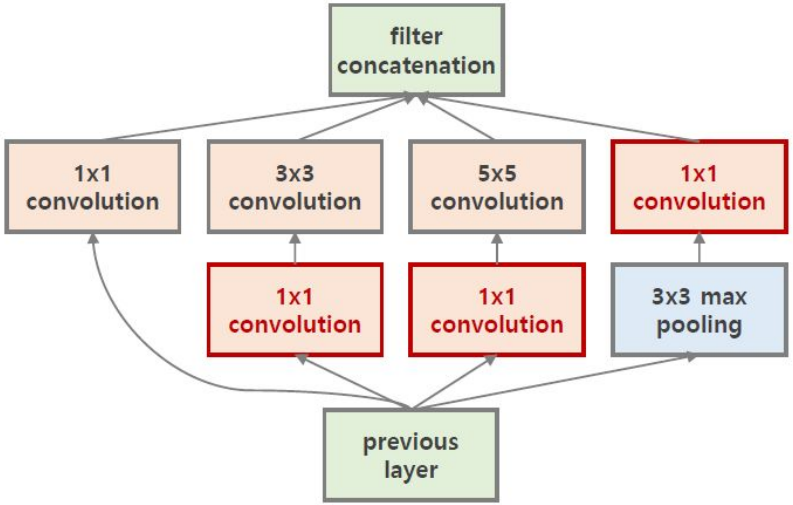
“Going Deeper with Convolutions”

CNN Architecture - GoogLeNet

Naive Inception module



Inception module with dimension reduction

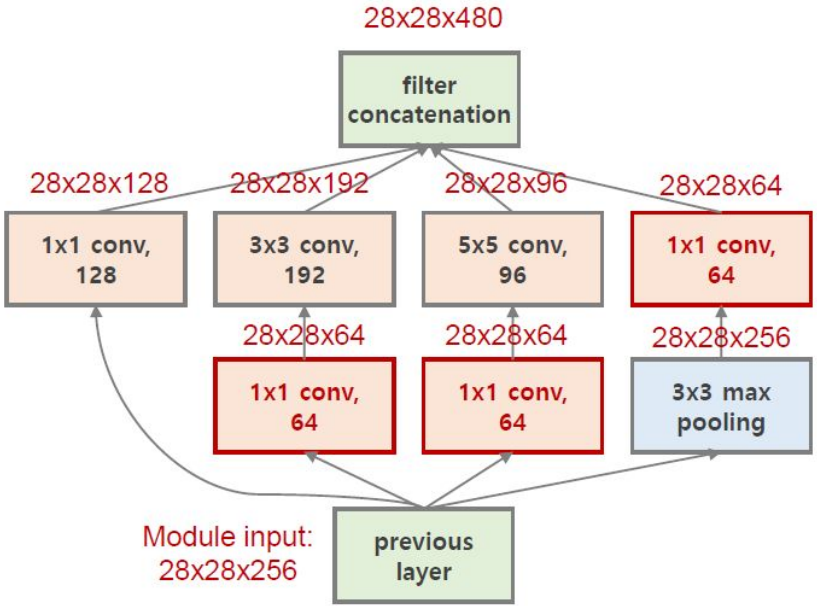


1x1 conv “bottleneck” layers

“Going Deeper with Convolutions”

CNN Architecture - GoogLeNet

Inception module with dimension reduction



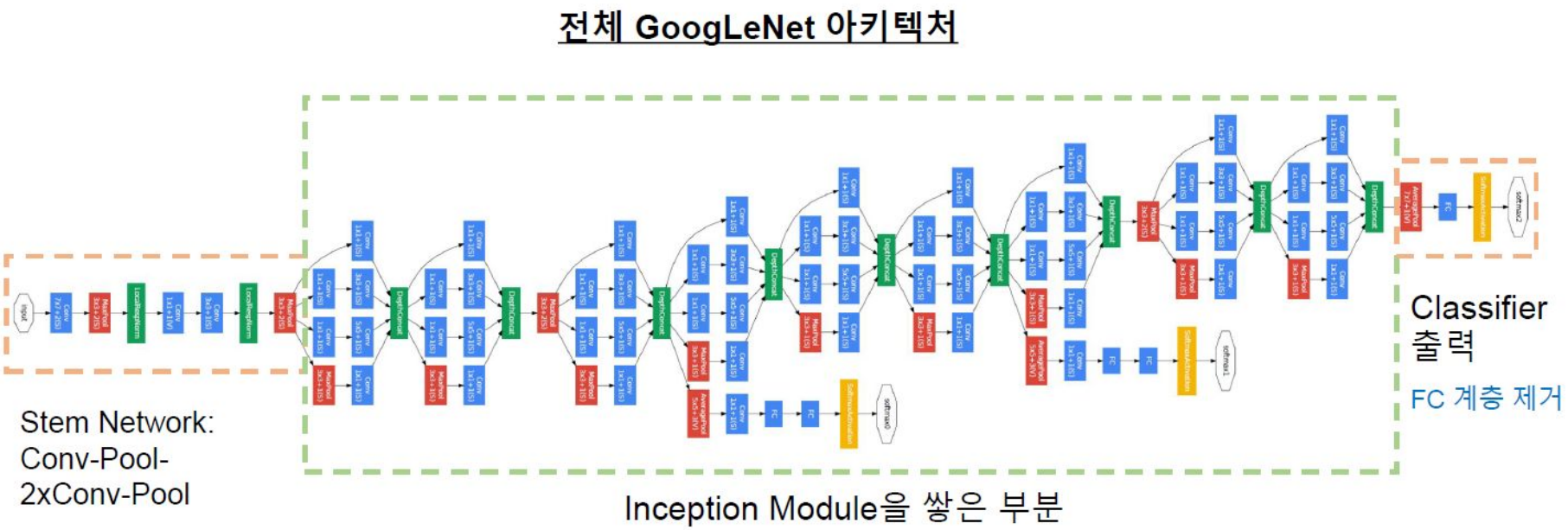
Conv Ops:
[1x1 conv, 64] [28x28x256]x[1x1]x64
[1x1 conv, 64] [28x28x256]x[1x1]x64
[1x1 conv, 128] [28x28x256]x[1x1]x128
[3x3 conv, 192] [28x28x64]x[3x3]x192
[5x5 conv, 96] [28x28x64]x[5x5]x96
[1x1 conv, 64] [28x28x256]x[1x1]x64
Total: 358M ops

- Naive version 854M ops과 비교하면 절반 이하

Pooling layer 다음 bottleneck을 통해 depth 줄임

"Going Deeper with Convolutions"

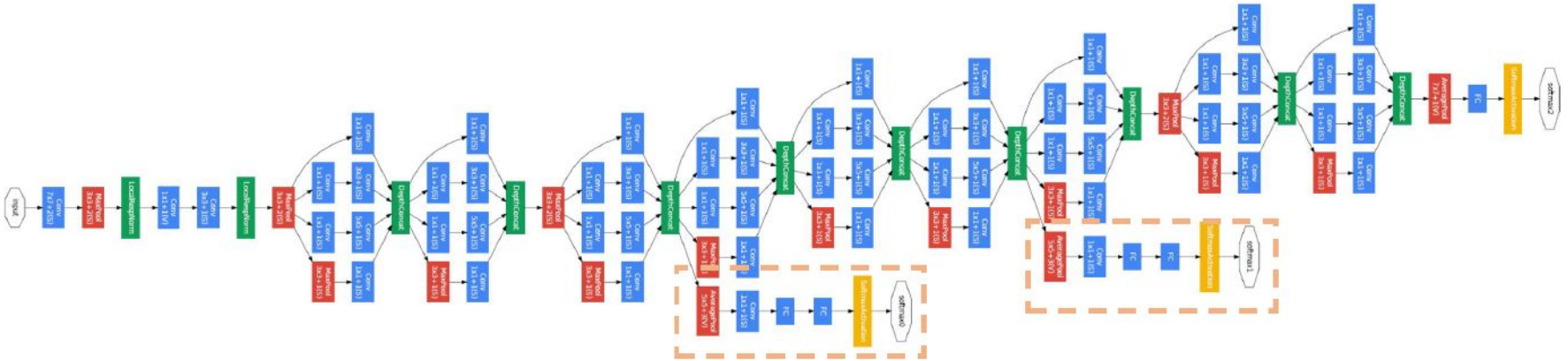
CNN Architecture - GoogLeNet



"Going Deeper with Convolutions"

CNN Architecture - GoogLeNet

전체 GoogLeNet 아키텍처



하위 계층에 Gradient를 원활히 공급하기 위해 보조 classification 출력을 둠
(AvgPool-1x1Conv-FC-FC-Softmax)

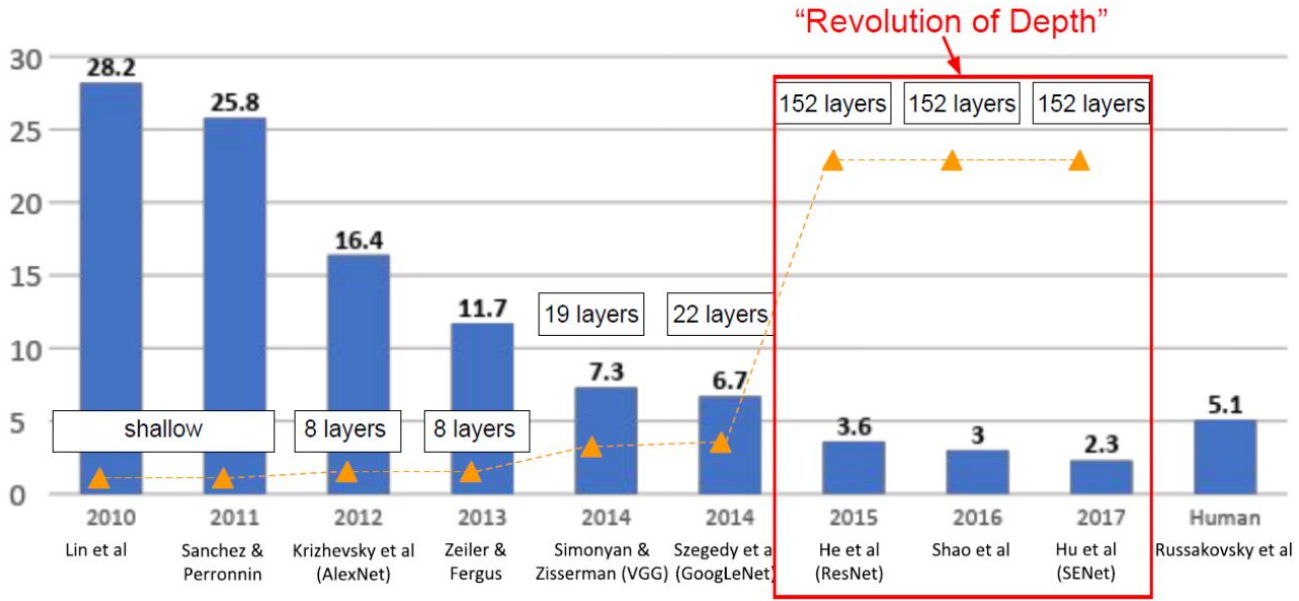
Going Deeper with Convolutions, 2014

전체 22 계층

- Inception module 9개 + conv 4개
- Parallel layer는 1개로 계산
- Inception module 별로 2계층으로 계산

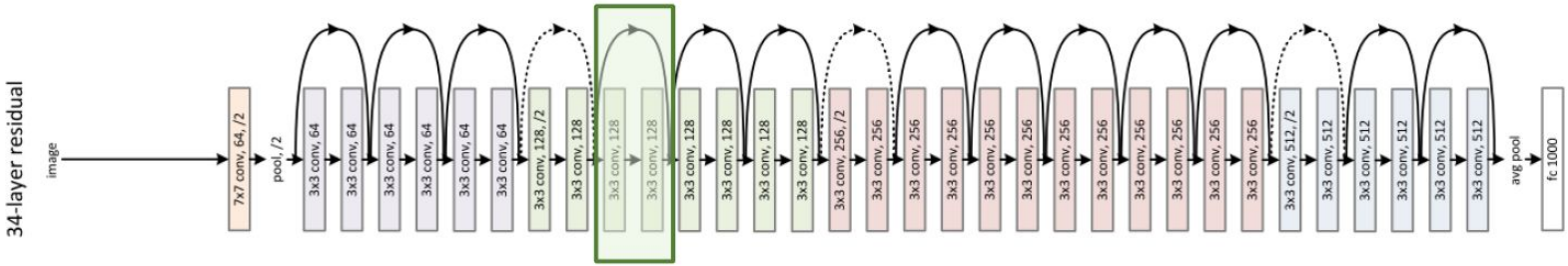
CNN Architecture - ResNet

ImageNet ILSVRC(Large Scale Visual Recognition Challenge) winners

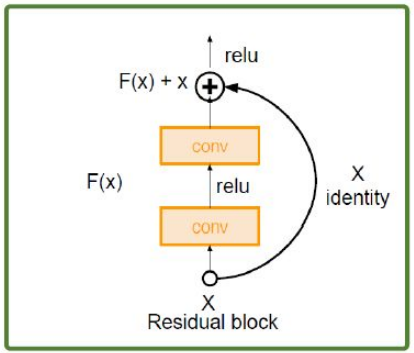


CNN Architecture - ResNet

Residual Connections을 사용한 매우 깊은 네트워크



레즈 블록

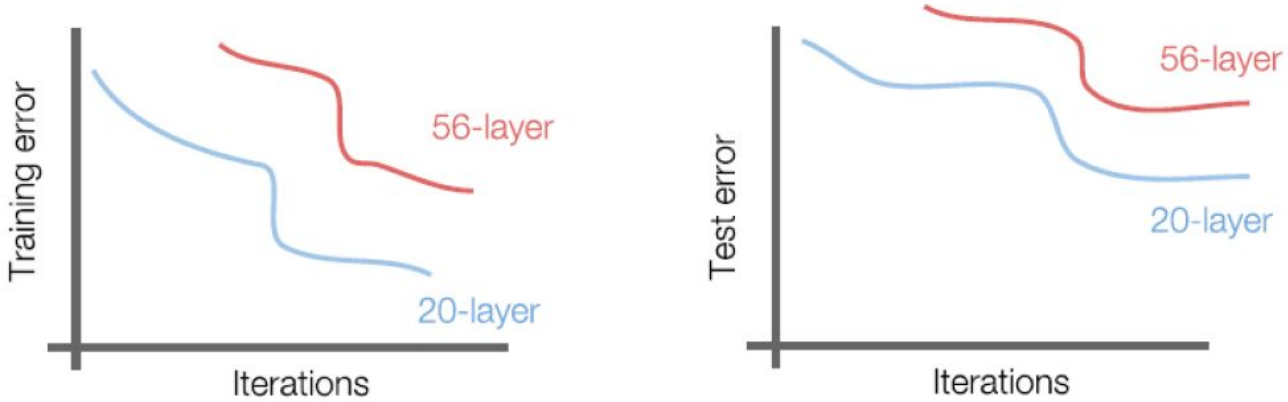


- **152-계층 Model** (ImageNet)
- **3.57%** top 5 error
  (ILSVRC'15 classification)
- ILSVRC'15와 COCO'15에서 모든 classification, detection 부분에서 우승

"Deep Residual Learning for Image Recognition"

CNN Architecture - ResNet

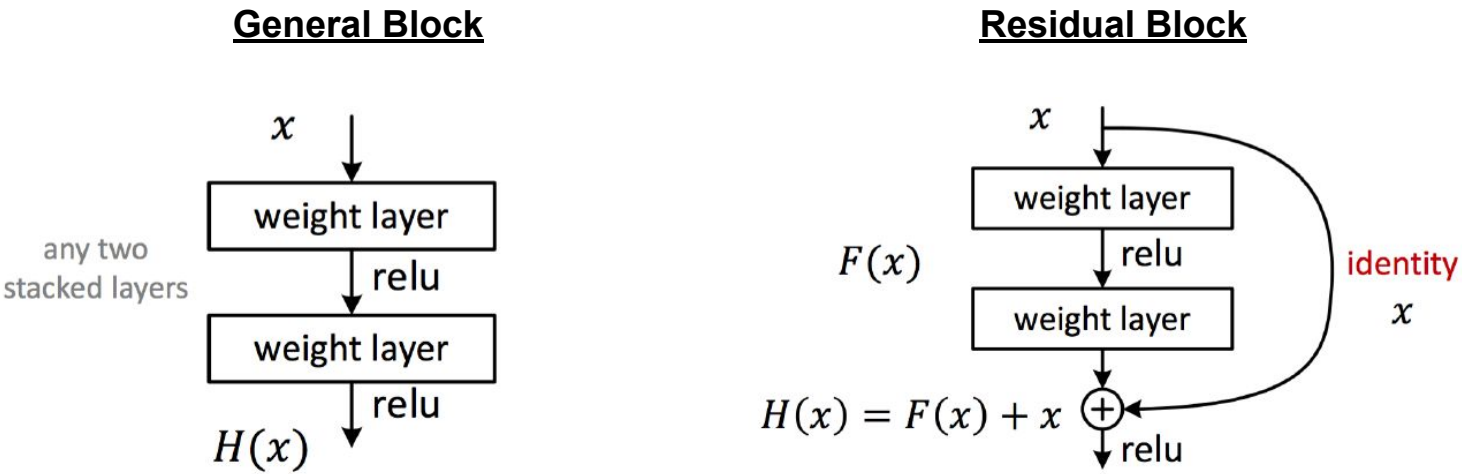
기본 CNN에서 계층을 더욱 깊이 쌓으면 어떻게 될까?



- 56-계층 모델의 훈련, 테스트 오류가 더 커지는 상황이 발생
- 하지만 이 문제는 overfitting 때문에 발생하는 문제가 아니다!
- 층이 깊어질 수록 역전파되는 그래디언트가 중간에 소실되면서 학습이 잘 되지 않는 문제(Gradient Vanishing)가 발생

"Deep Residual Learning for Image Recognition"

CNN Architecture - ResNet



- 중간에 layer를 뛰어넘는 연결을 추가, Skip connection
- 역전파 시 Gradient가 그대로 소실되지않고 전달된다.

"Deep Residual Learning for Image Recognition"

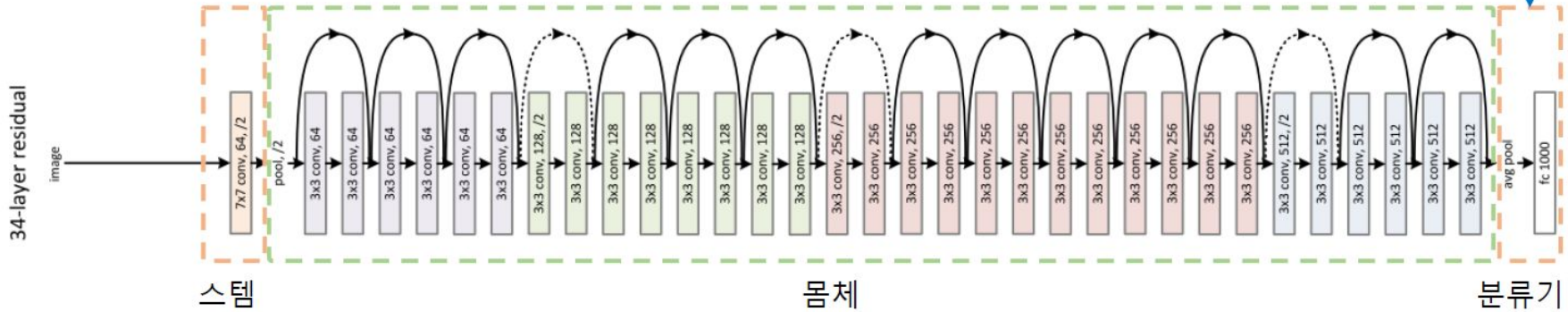
CNN Architecture - ResNet

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

"Deep Residual Learning for Image Recognition"

CNN Architecture - ResNet

- 끝 부분에 FC 계층 제거
  (클래스 출력을 위한 FC 1000만 존재)
- Global Average Pooling 사용



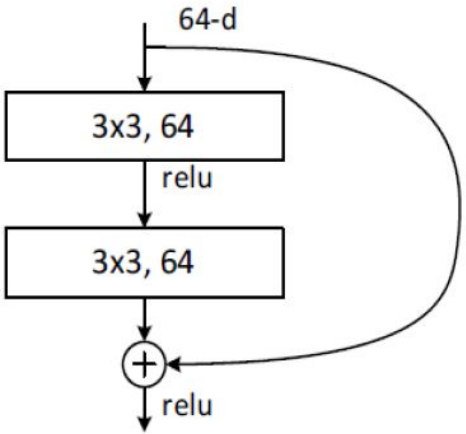
- 주기적으로 filter 개수를 2배로 늘리고
  stride 2를 이용해서 downsampling

"Deep Residual Learning for Image Recognition"

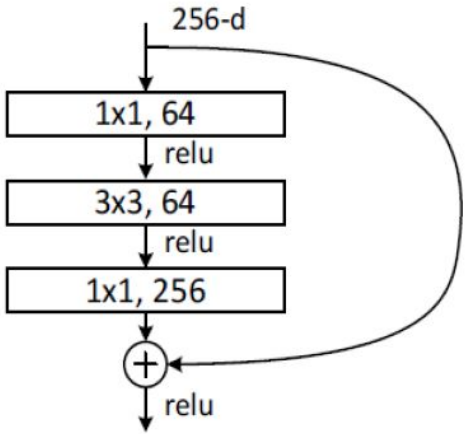
CNN Architecture - ResNet

ResNet-50 이상인 경우 계산 효율을 위해 Bottleneck Layer 추가

Residual Block



Bottleneck Layer가 있는 Residual Block



"Deep Residual Learning for Image Recognition"

CNN Architecture - ResNet

- CONV layer 이후에 Batch Normalization
- Xavier 2/ 초기화
- SGD + Momentum (0.9)
- 학습률 : 0.1, (validation error가 감소되지 않으면 10씩 나눔)
- 미니 배치 크기 : 256
- 가중치 감소 (Weight decay) : 1e-5
- Dropout은 사용하지 않음

ILSVRC과 COCO 2015의 모든 부문에서 1위를 석권

MSRA @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: *"Ultra-deep"* (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

아주 깊은 네트워크에서 실험

- 성능저하 없이 훈련
  - ImageNet에 대해 152계층
  - Cifar에 대해 1202계층
- 낮은 훈련 오류율 달성

- ILSVRC 2015 classification 우승 (**3.6%** top 5 error)
- 사람의 인지 능력보다 뛰어남 (Russakovsky 2014)

"Deep Residual Learning for Image Recognition"

CNN Architecture - ResNet

- CONV layer 이후에 Batch Normalization
- Xavier 2/ 초기화
- SGD + Momentum (0.9)
- 학습률 : 0.1, (validation error가 감소되지 않으면 10씩 나눔)
- 미니 배치 크기 : 256
- 가중치 감소 (Weight decay) : 1e-5
- Dropout은 사용하지 않음

ILSVRC과 COCO 2015의 모든 부문에서 1위를 석권

MSRA @ ILSVRC & COCO 2015 Competitions

- **1st places in all five main tracks**
  - ImageNet Classification: *"Ultra-deep"* (quote Yann) 152-layer nets
  - ImageNet Detection: 16% better than 2nd
  - ImageNet Localization: 27% better than 2nd
  - COCO Detection: 11% better than 2nd
  - COCO Segmentation: 12% better than 2nd

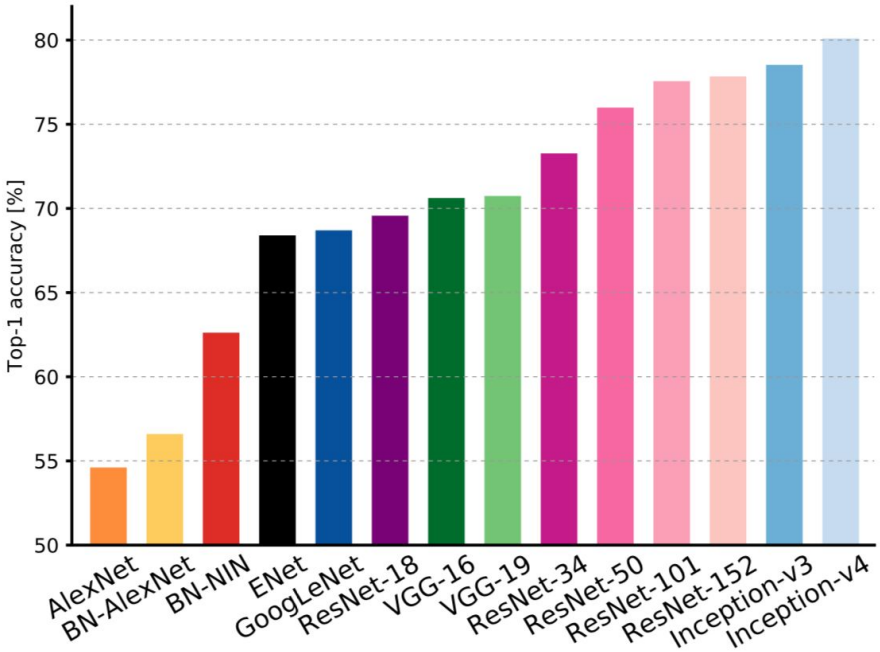
아주 깊은 네트워크에서 실험

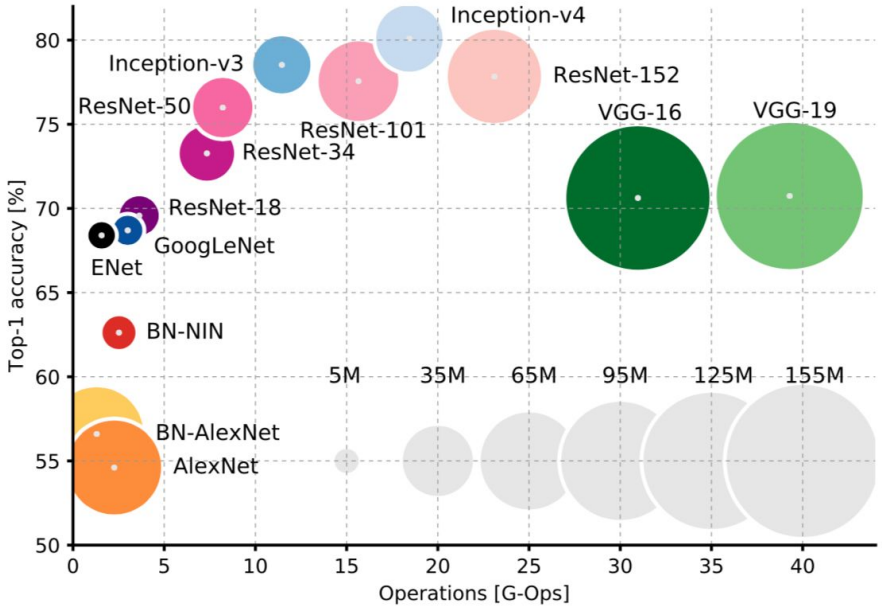
- 성능저하 없이 훈련
  - ImageNet에 대해 152계층
  - Cifar에 대해 1202계층
- 낮은 훈련 오류율 달성

- ILSVRC 2015 classification 우승 (**3.6%** top 5 error)
- 사람의 인지 능력보다 뛰어남 (Russakovsky 2014)

"Deep Residual Learning for Image Recognition"

CNN Architecture





참고자료

- 밑바닥부터 시작하는 딥러닝 1, 2
  http://www.yes24.com/Product/Goods/34970929?Acode=101
  http://www.yes24.com/Product/Goods/72173703

- 모두를 위한 딥러닝 시즌2
  https://www.edwith.org/boostcourse-dl-tensorflow/joinLectures/22150

- 모두의 연구소 이일구, 윤성진님(CRAS Lab) 강의 자료
  https://github.com/ilguyi