CNN Architectures

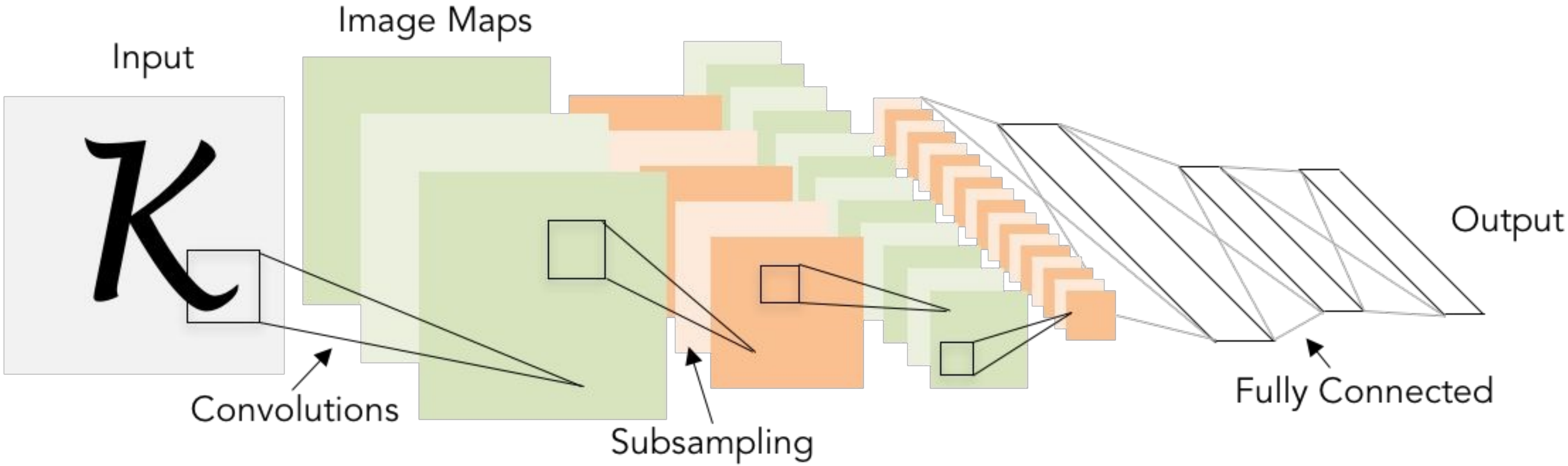
CNN Architectures

- VGG
- GoogLeNet
- ResNet

- Depthwise Convolution

Review: LeNet-5

[LeCun et al., 1998]



Conv filters were 5x5, applied at stride 1
Subsampling (Pooling) layers were 2x2 applied at stride 2
i.e. architecture is [CONV-POOL-CONV-POOL-FC-FC]

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

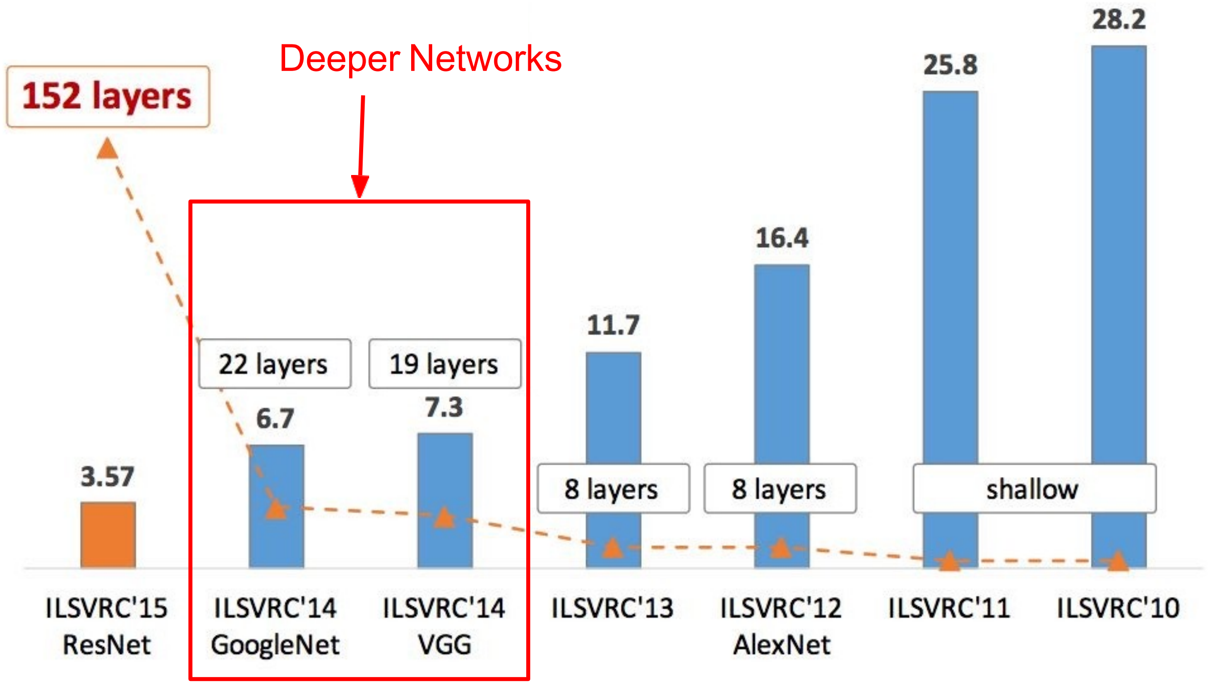


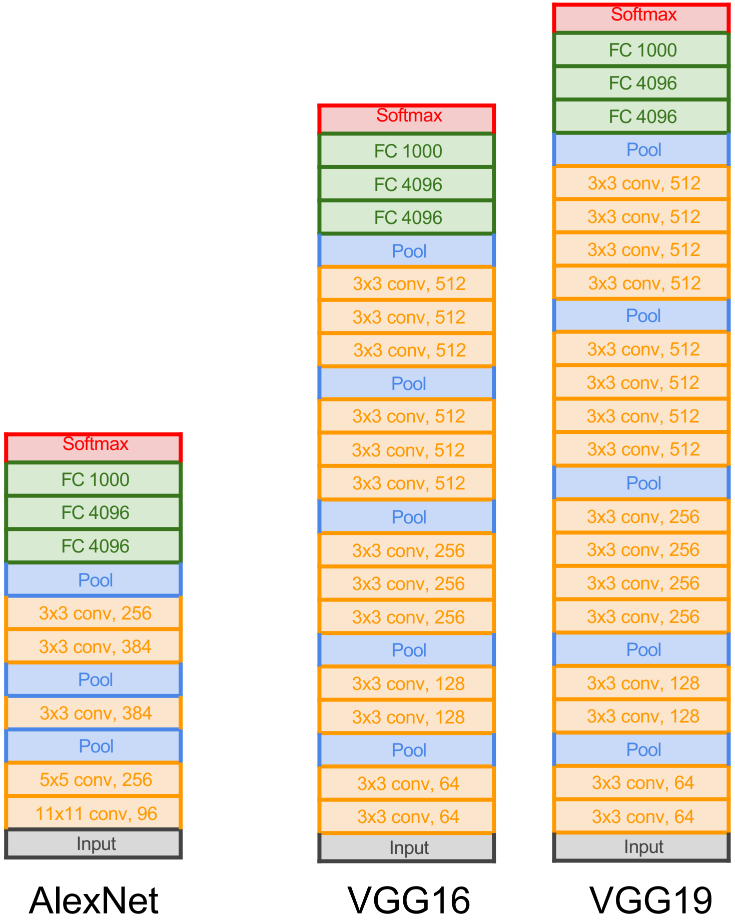
Figure copyright Kaiming He, 2016. Reproduced with permission.

VGGNet

Small filters, Deeper networks

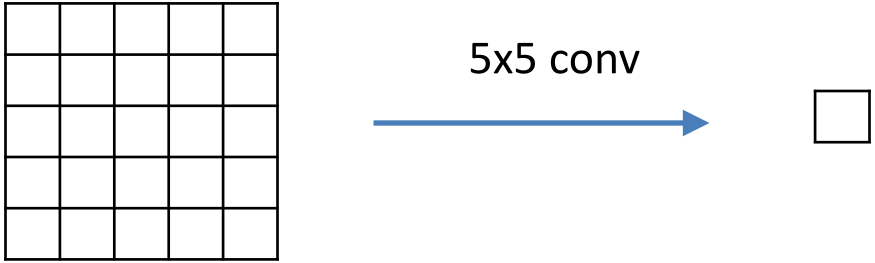
8 layers (AlexNet)
-> 16 - 19 layers (VGG16Net)

Only 3x3 CONV stride 1, pad 1
and 2x2 MAX POOL stride 2



VGGNet

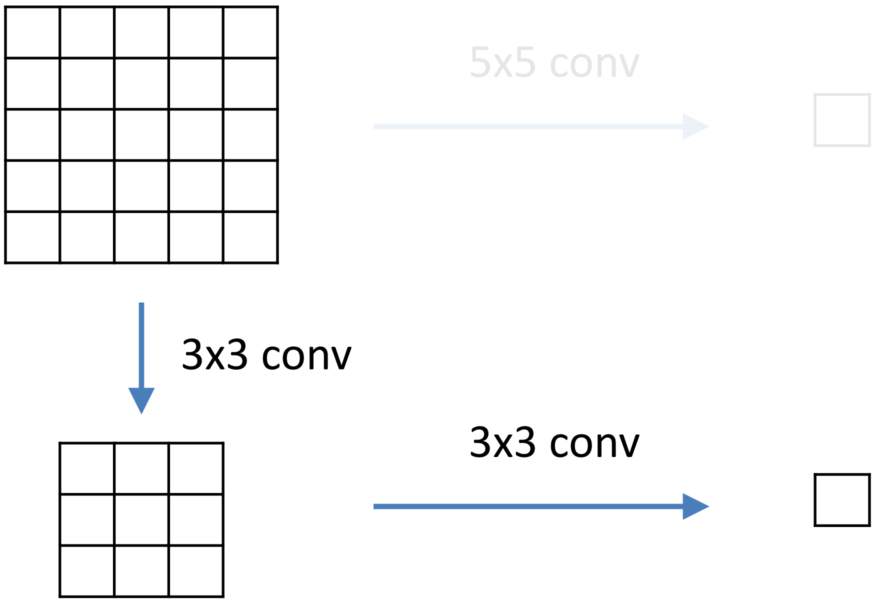
Large Filters vs Small Filters



25 params

VGGNet

Large Filters vs Small Filters



5x5 conv

25 params

3x3 conv

3x3 conv

9+9 prams
More non-linearity

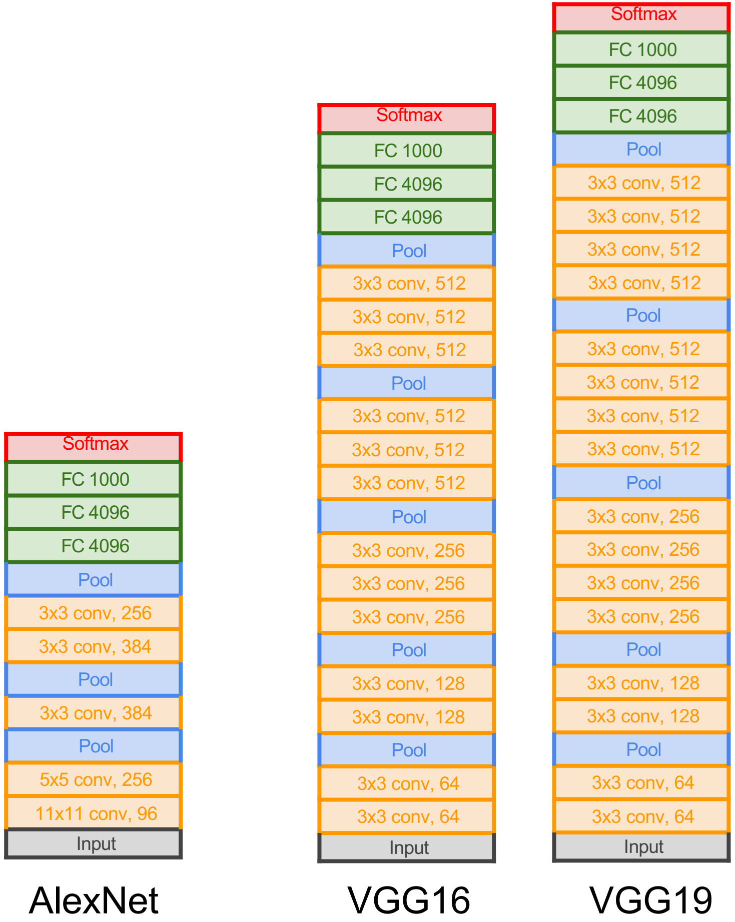
VGGNet

Q: Why use smaller filters? (3x3 conv)

Stack of three 3x3 conv (stride 1) layers has same **effective receptive field** as one 7x7 conv layer

And fewer parameters: 3 * (3^2C^2) vs. 7^2C^2 for C channels per layer

But deeper, more non-linearities



INPUT: [224x224x3] memory: 224*224*3=150K params: 0 (not counting biases)

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 C

ONV3-256: [56x56x256] memory: 56*56*256=800K params: (3*3*256)*256 = 589,824 CO

NV3-256: [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 C

ONV3-512: [28x28x512] memory: 28*28*512=400K params: (3*3*512)*512 = 2,359,296 CO

NV3-512: [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 C

ONV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296 CO

NV3-512: [14x14x512] memory: 14*14*512=100K params: (3*3*512)*512 = 2,359,296

POOL2: [7x7x512] memory: 7*7*512=25K params: 0

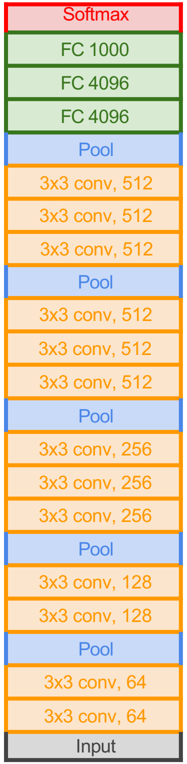
FC: [1x1x4096] memory: 4096 params: 7*7*512*4096 = 102,760,448 F

C: [1x1x4096] memory: 4096 params: 4096*4096 = 16,777,216

FC: [1x1x1000] memory: 1000 params: 4096*1000 = 4,096,000

TOTAL memory: 24M * 4 bytes ~= 96MB / image (only forward! ~*2 for bwd)

TOTAL params: 138M parameters



Softmax

FC 1000

FC 4096

FC 4096

Pool

3x3 conv, 512

3x3 conv, 512

3x3 conv, 512

Pool

3x3 conv, 512

3x3 conv, 512

3x3 conv, 512

Pool

3x3 conv, 256

3x3 conv, 256

3x3 conv, 256

Pool

3x3 conv, 128

3x3 conv, 128

Pool

3x3 conv, 64

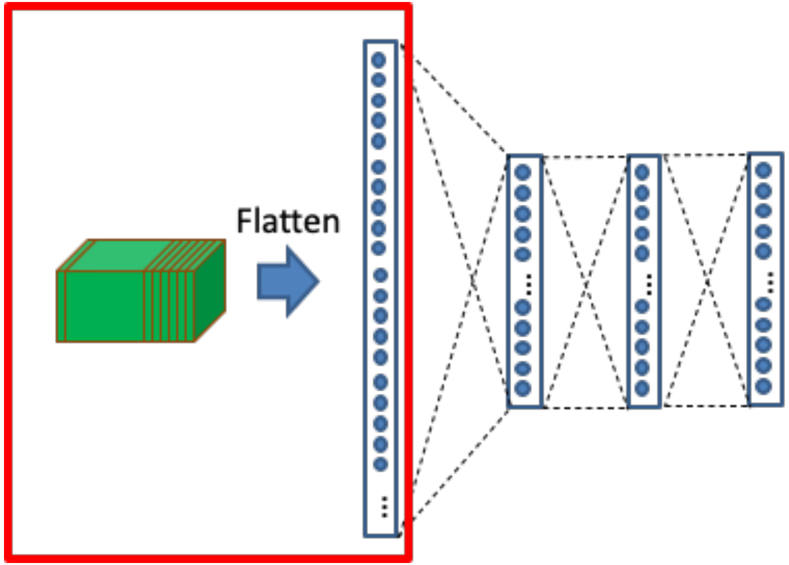
3x3 conv, 64

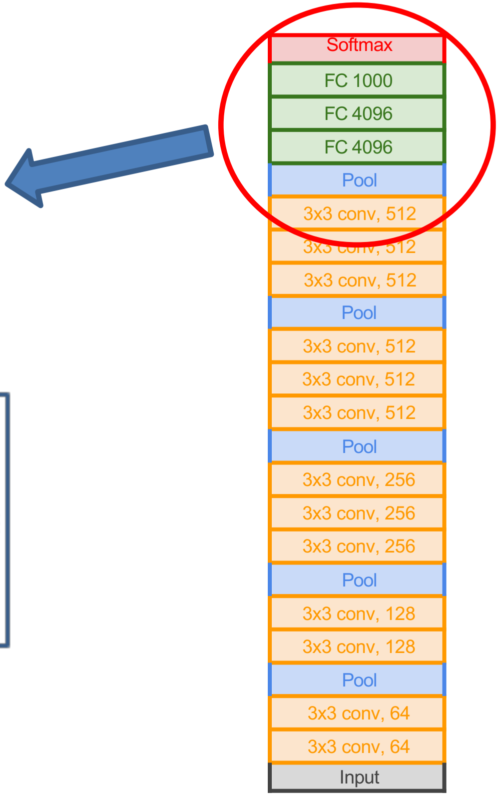
Input

VGG16

VGGNet

Too many parameters. Especially in FC layers



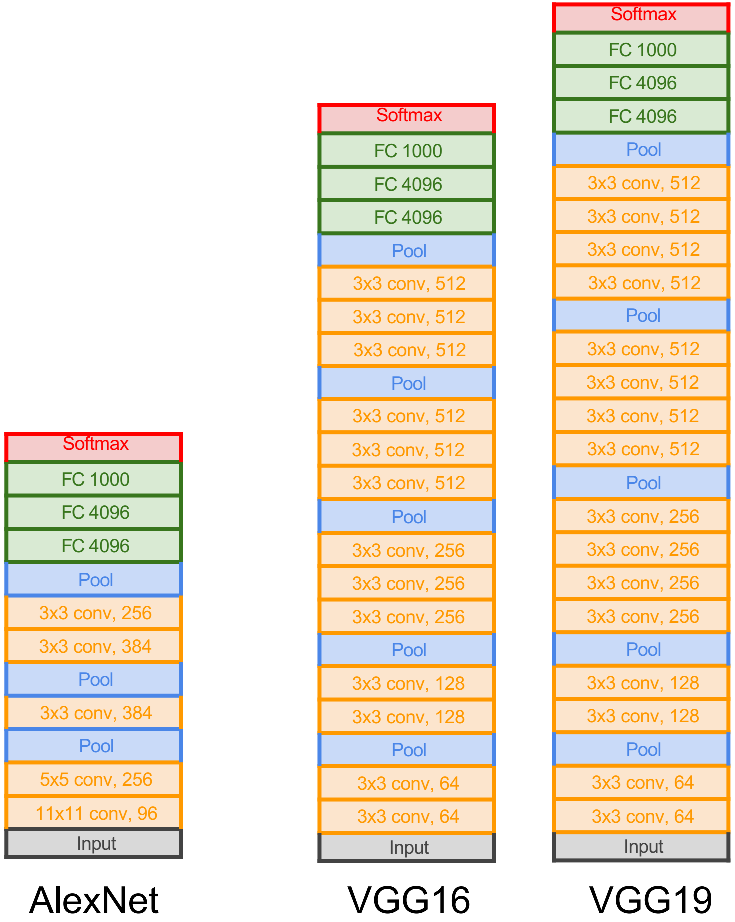

Softmax
FC 1000
FC 4096
FC 4096
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 512
3x3 conv, 512
3x3 conv, 512
Pool
3x3 conv, 256
3x3 conv, 256
3x3 conv, 256
Pool
3x3 conv, 128
3x3 conv, 128
Pool
3x3 conv, 64
3x3 conv, 64
Input

VGG16

VGGNet

Summary:

- Only 3x3 filters
- Deeper Structure
- Huge # of parameters



ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

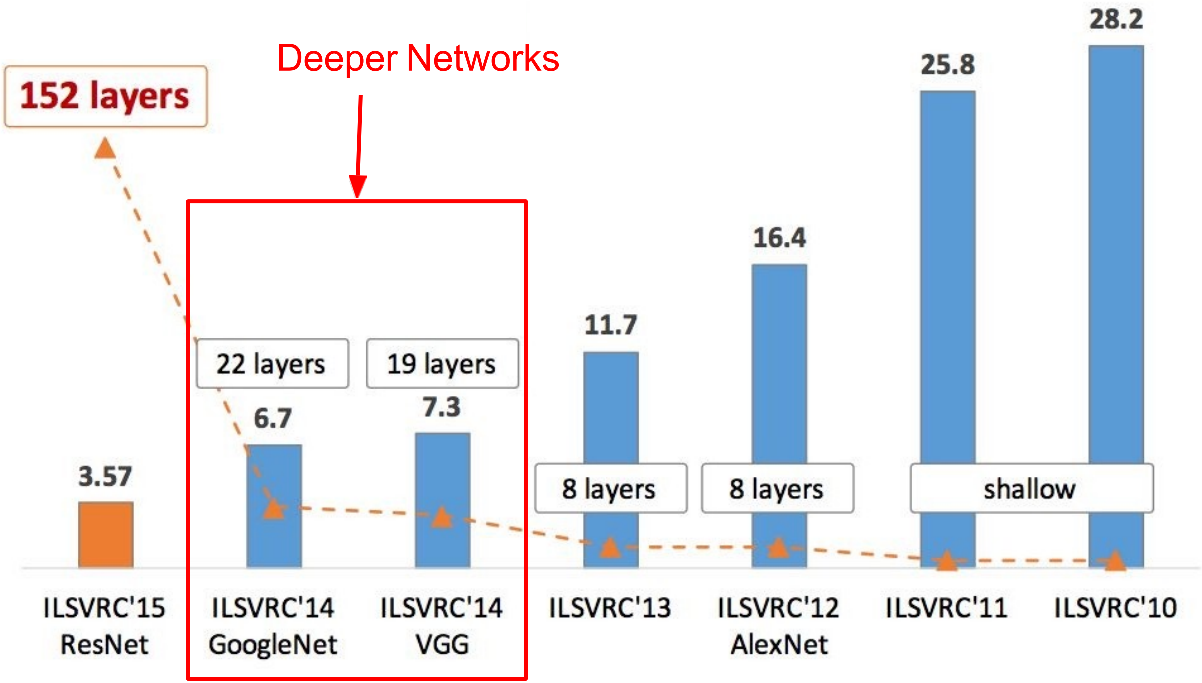
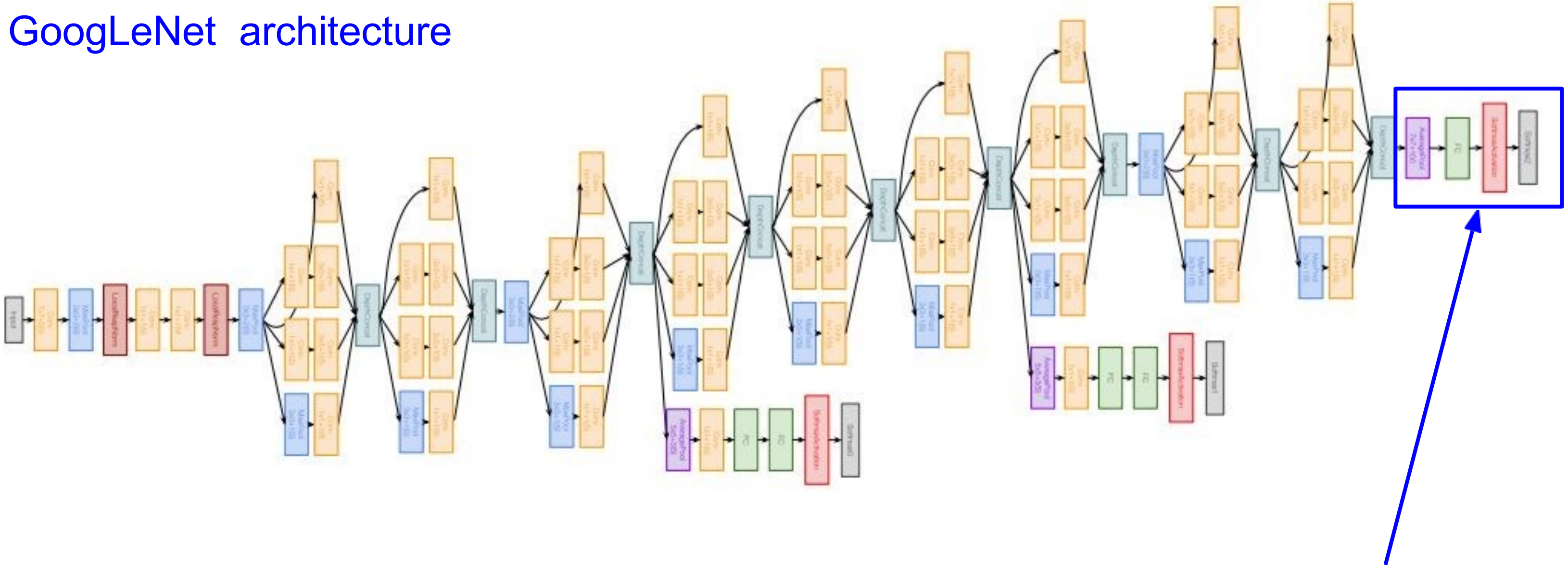


Figure copyright Kaiming He, 2016. Reproduced with permission.

GoogLeNet

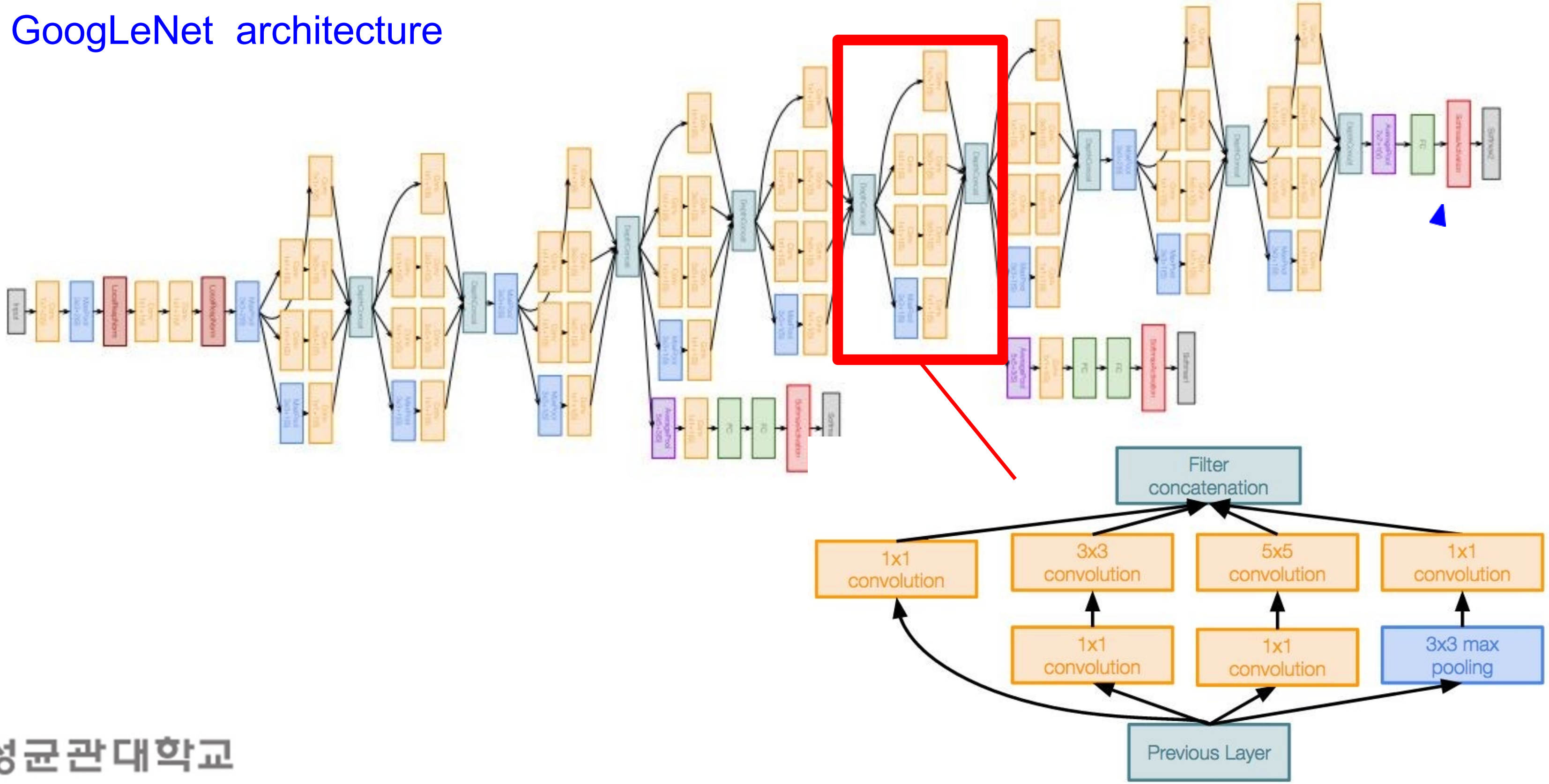
Full GoogLeNet architecture



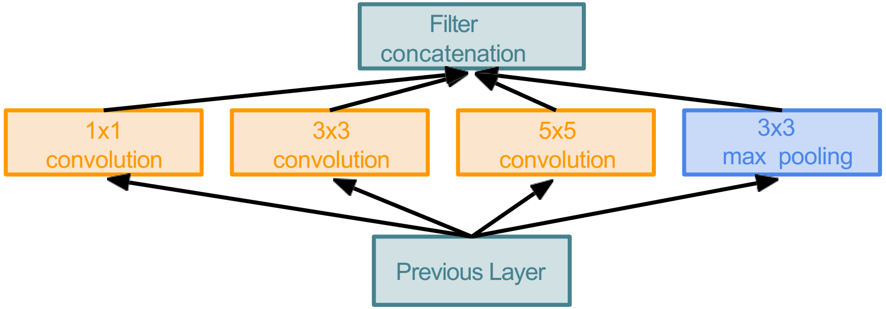
Classifier output

GoogLeNet

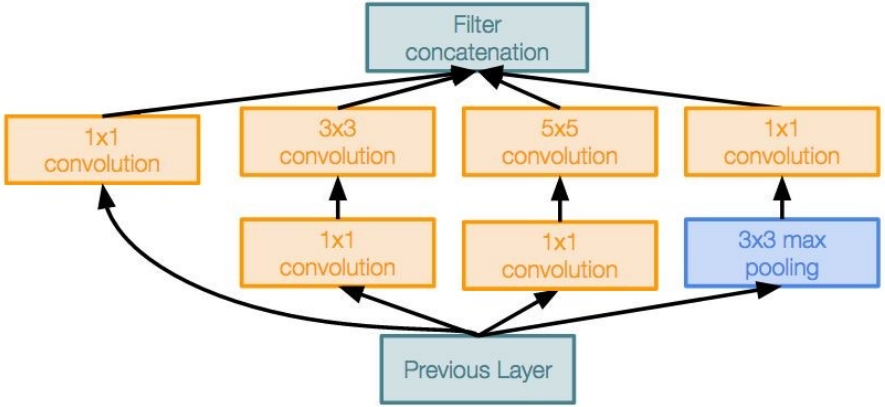
Full GoogLeNet architecture



GoogLeNet: Inception Module

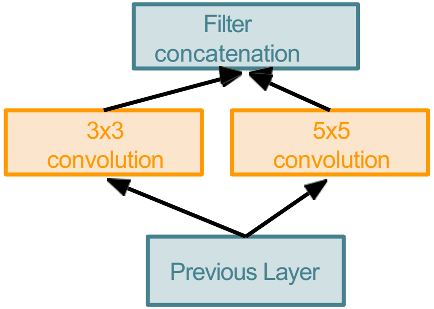


Naive Inception module

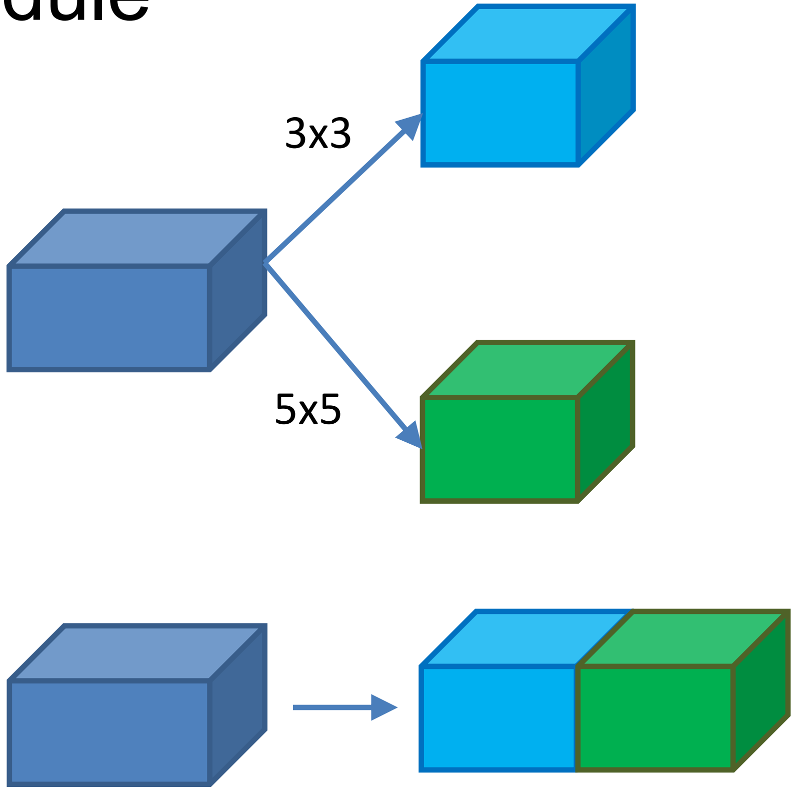


Inception module with dimension reduction

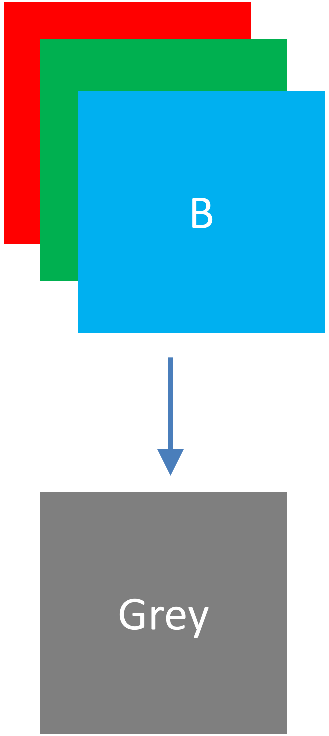
GoogLeNet: Inception Module

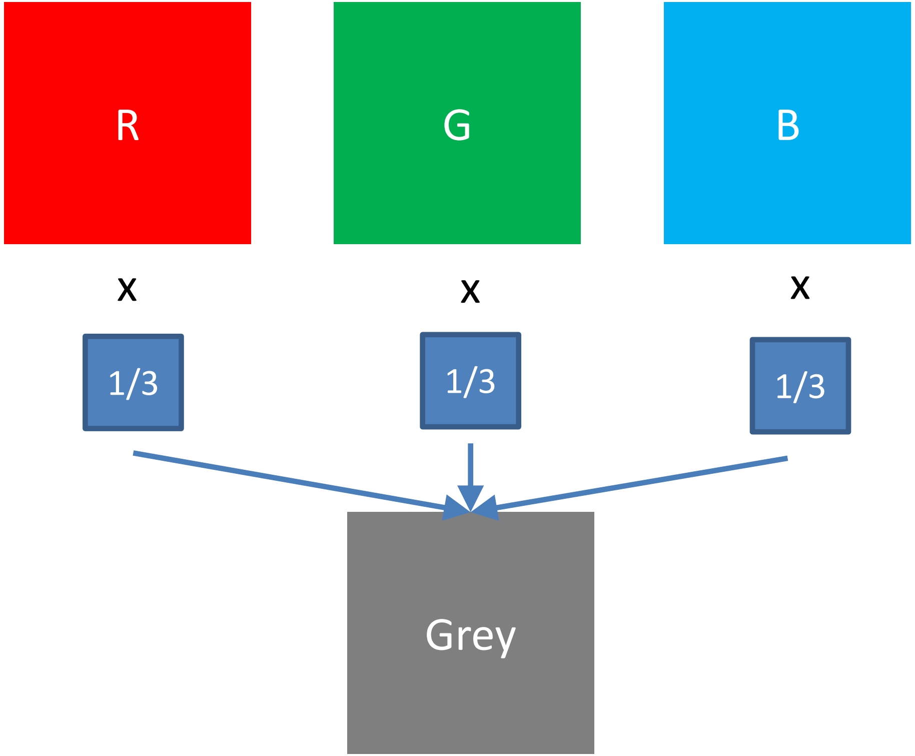


Naive Inception module

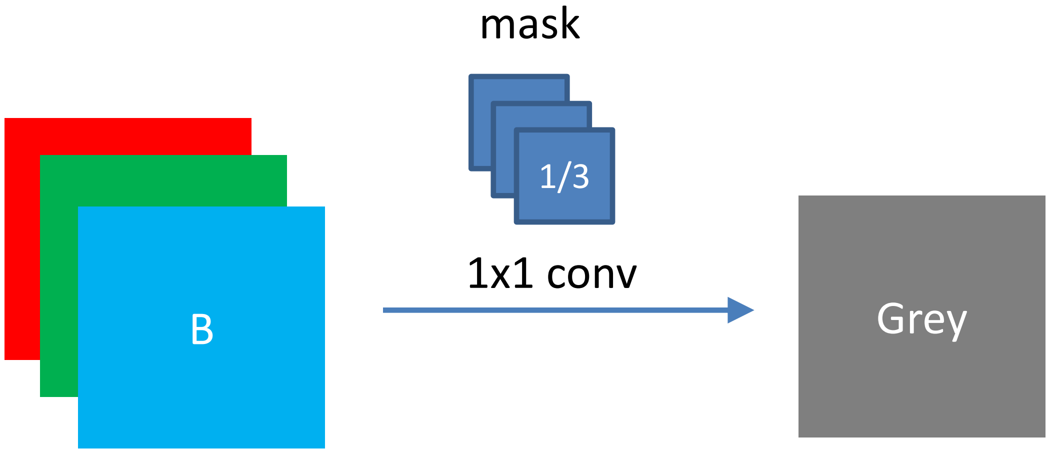


GoogLeNet : 1x1 convolutions

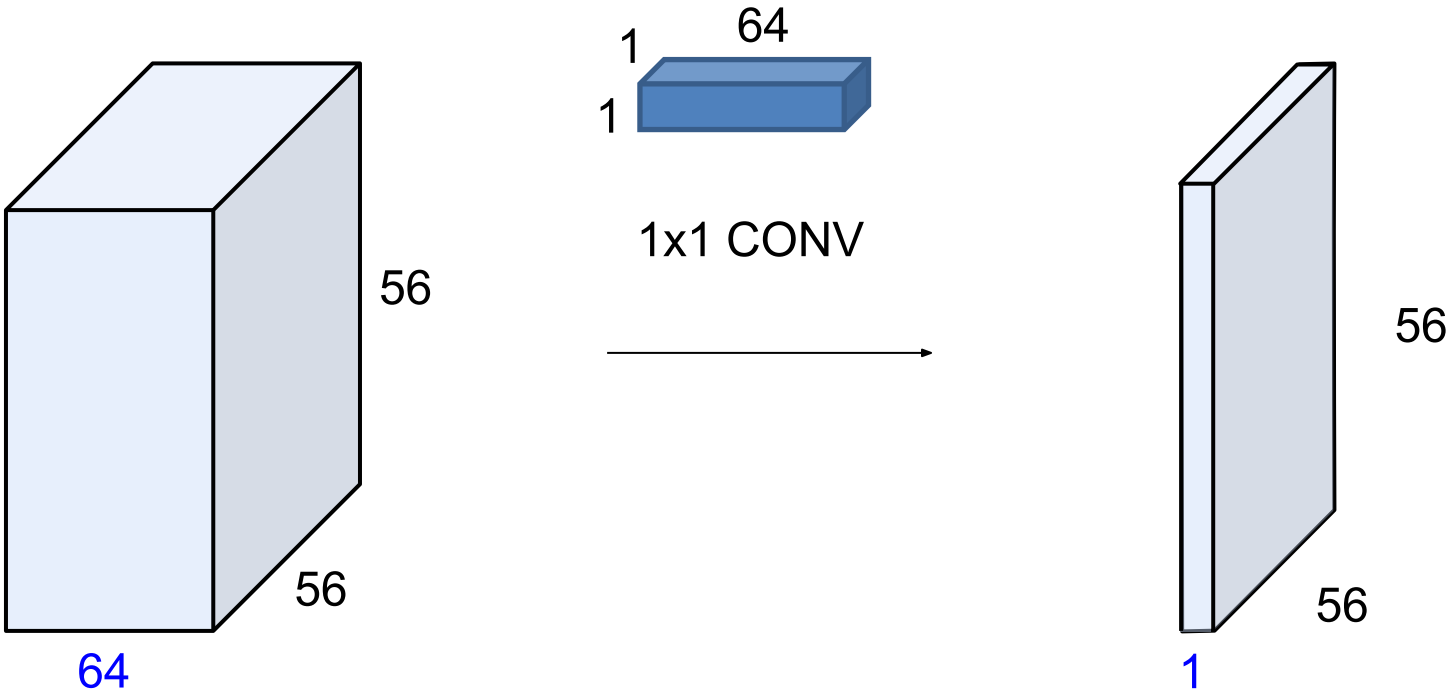




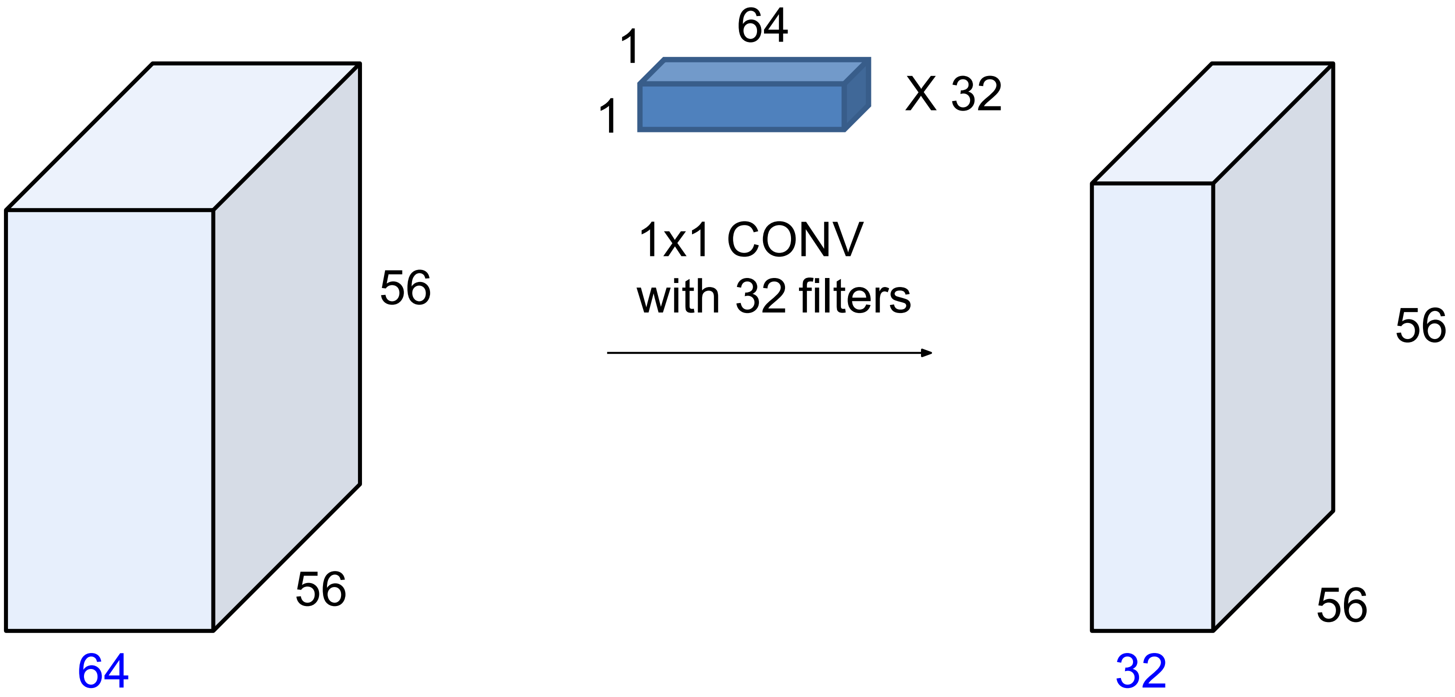
GoogLeNet : 1x1 convolutions



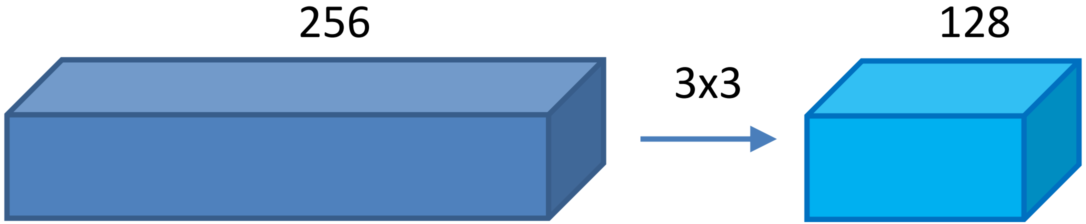
GoogLeNet : 1x1 convolutions



GoogLeNet : 1x1 convolutions

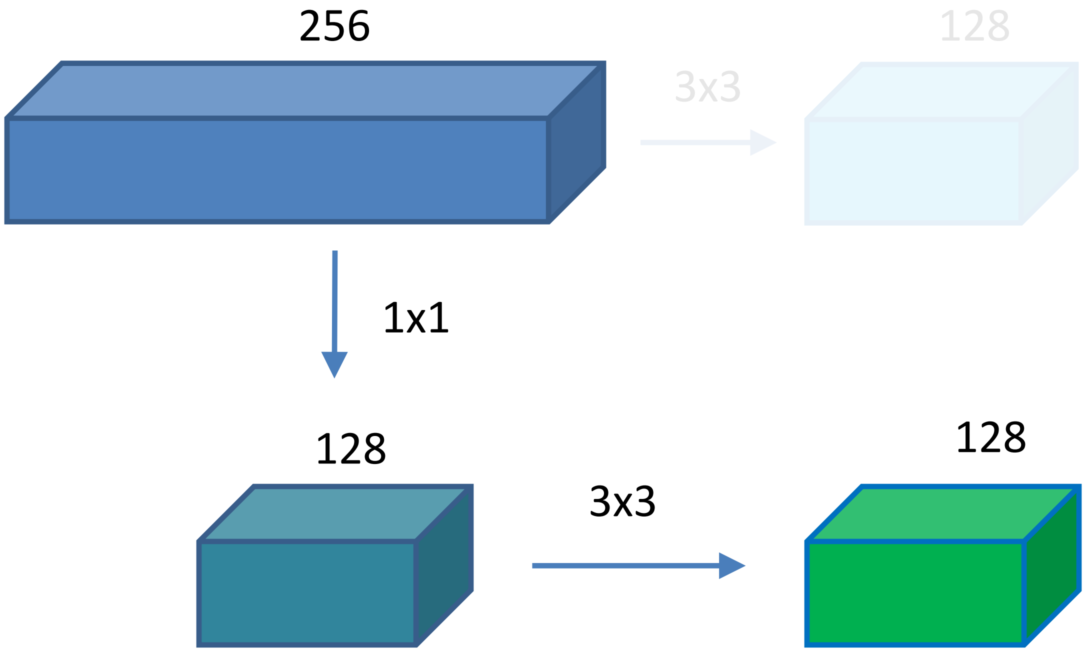


GoogLeNet: Convolution with 1x1 Convolution



256 X 3X3 X 128
= 294,912

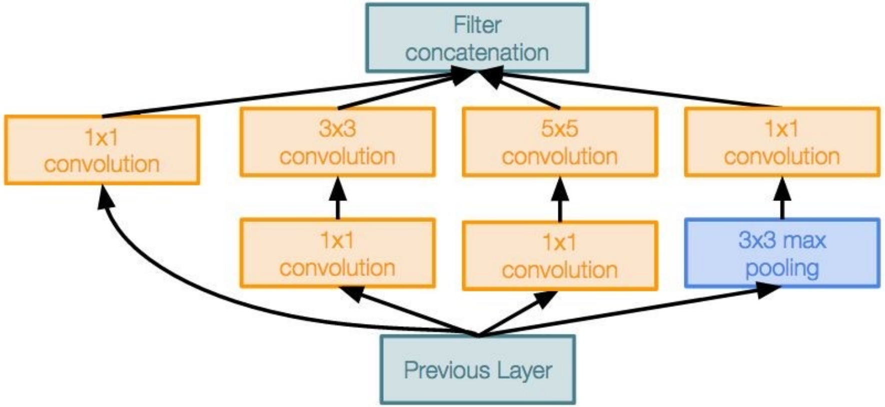
GoogLeNet: Convolution with 1x1 Convolution



256 X 3X3 X 128
= 294,912

256 X 1X1 X 128
+ 128 X 3X3 X 128
= 189,224

GoogLeNet: Inception Module



Inception module with dimension reduction

3x3 max pooling, stride=1

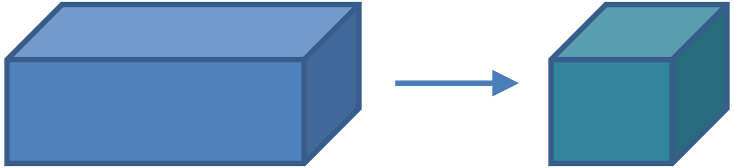
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	3	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Feature map

0	0	0	0	0	0	0
0	3	3	3	0	0	0
0	3	3	3	0	0	0
0	3	3	3	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

Enhanced feature map

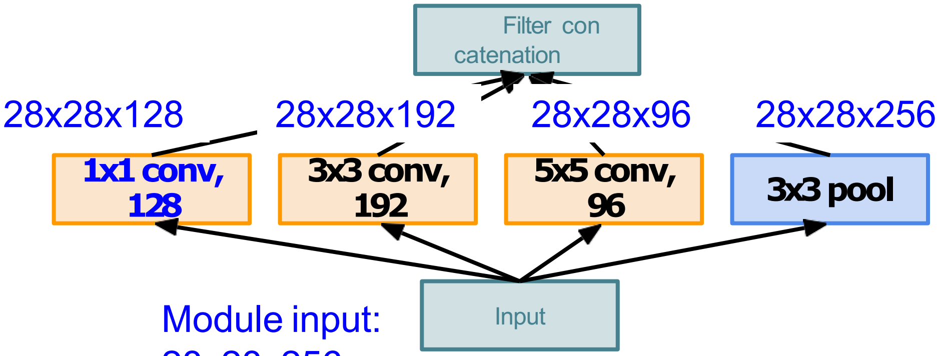
1x1 Convolution



GoogLeNet: Inception Module

Example:

28x28x(128+192+96+256) = 28x28x672



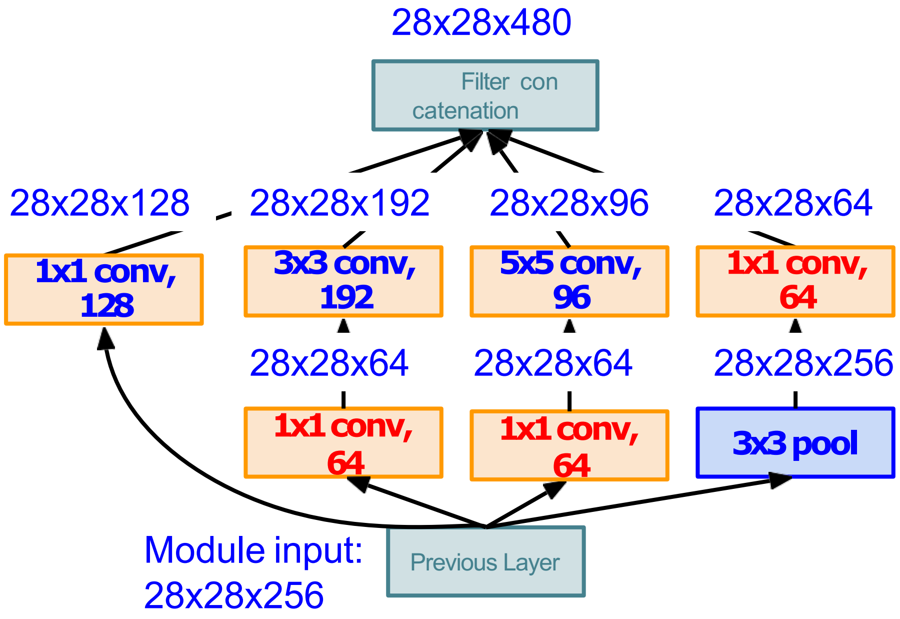
Naive Inception module

Q: What is the problem with this?
[Hint: Computational complexity]

Conv Ops:
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x256
[5x5 conv, 96] 28x28x96x5x5x256
Total: 854M ops

Very expensive compute

GoogLeNet: Inception Module



Inception module with dimension reduction

Conv Ops:

[1x1 conv, 64] 28x28x64x1x1x256

[1x1 conv, 64] 28x28x64x1x1x256

[1x1 conv, 128] 28x28x128x1x1x256

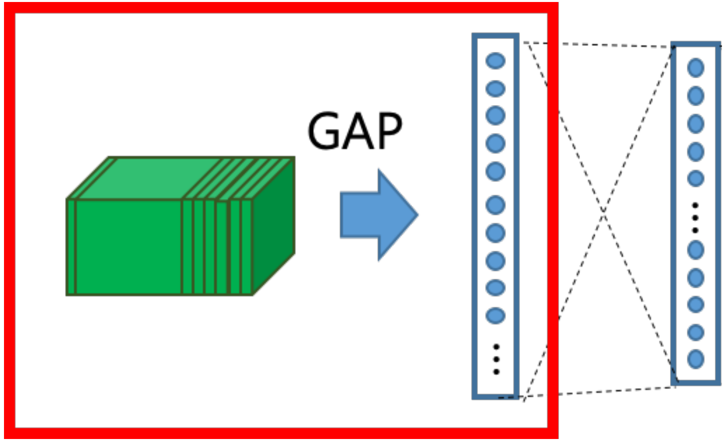
[3x3 conv, 192] 28x28x192x3x3x64

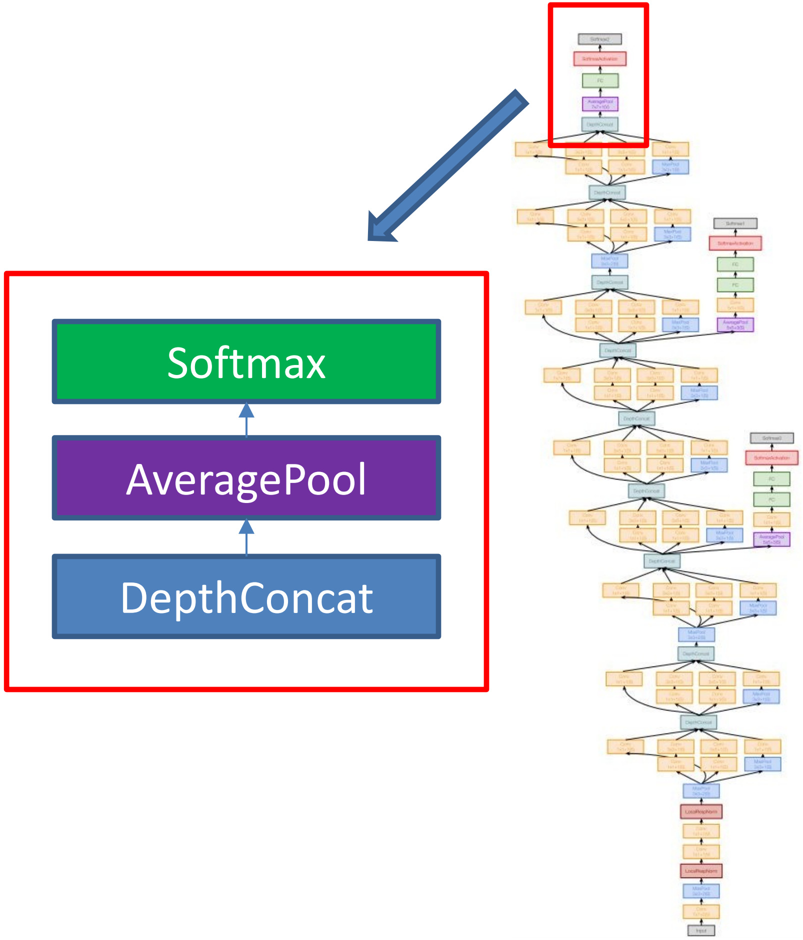
[5x5 conv, 96] 28x28x96x5x5x64

[1x1 conv, 64] 28x28x64x1x1x256

Total: 358M ops

GoogLeNet: FC Layers

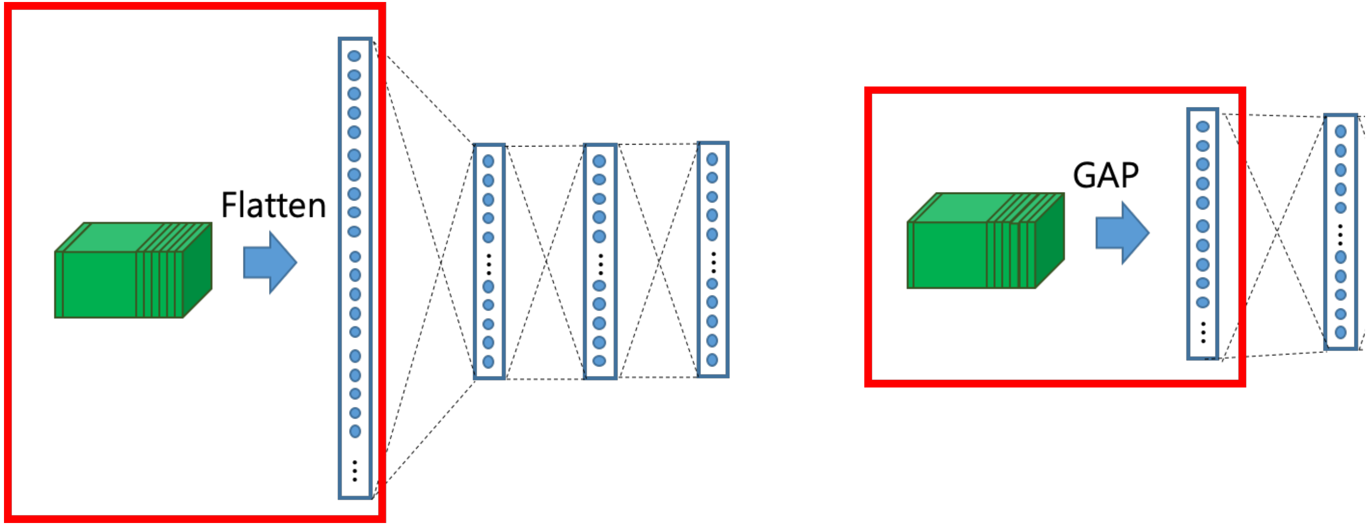



Softmax

AveragePool

DepthConcat

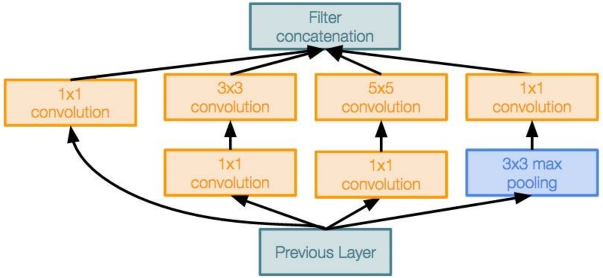
GoogLeNet: FC Layers



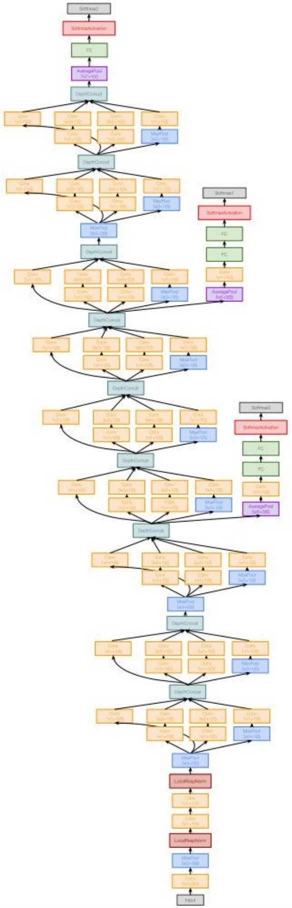
GoogLeNet

Deeper networks, with computational efficiency

- 22 layers
- Efficient “Inception” module
- No FC layers
- Only 5 million parameters!
  12x less than AlexNet
- ILSVRC’14 classification winner
  (6.7% top 5 error)



Inception module



ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

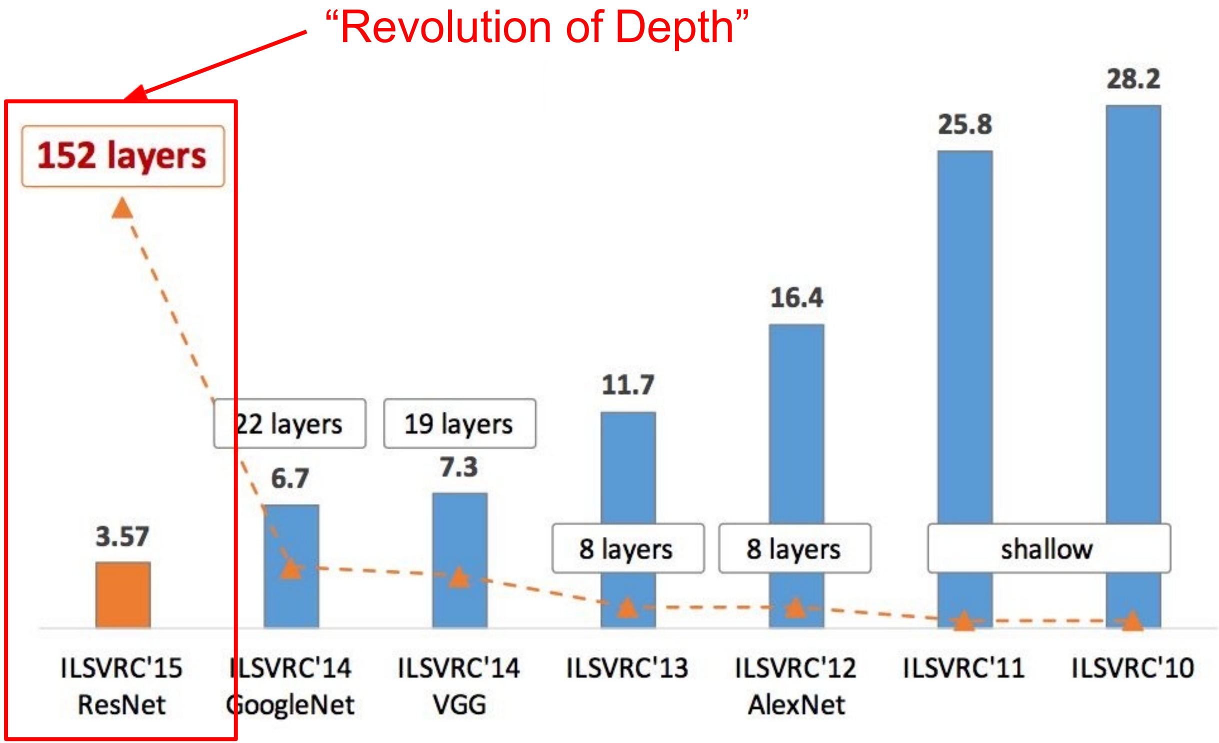
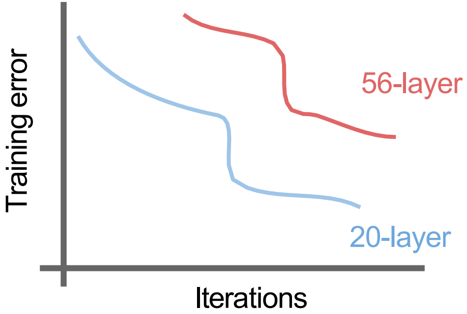
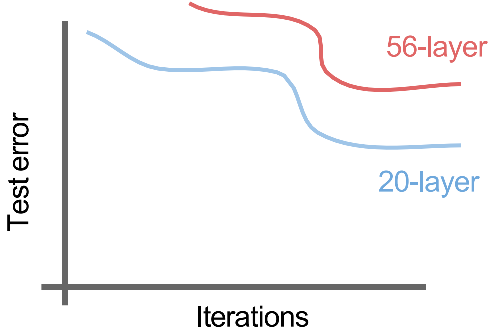


Figure copyright Kaiming He, 2016. Reproduced with permission.

ResNet

What happens with deeper networks?

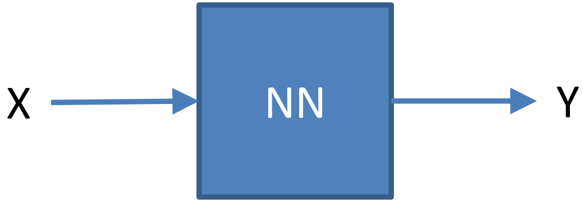




56-layer model performs worse on both training and test error
-> The deeper model performs worse, but it’s not caused by overfitting!

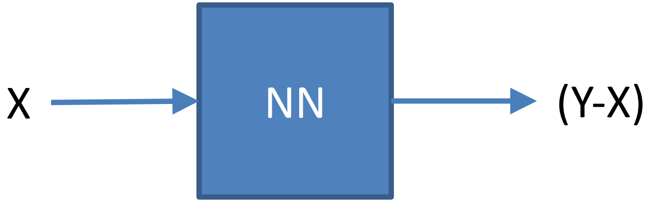
ResNet: Another Form of NN

X	Y
1	0.9
2	2.1
3	3.0
4	4.2



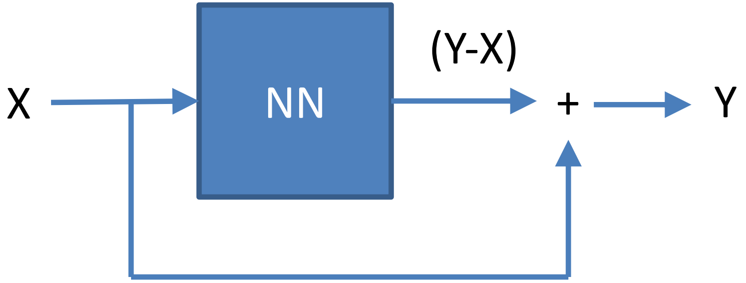
ResNet: Another Form of NN

X	Y	Y-X
1	0.9	-0.1
2	2.1	0.1
3	3.0	0.0
4	4.2	0.2



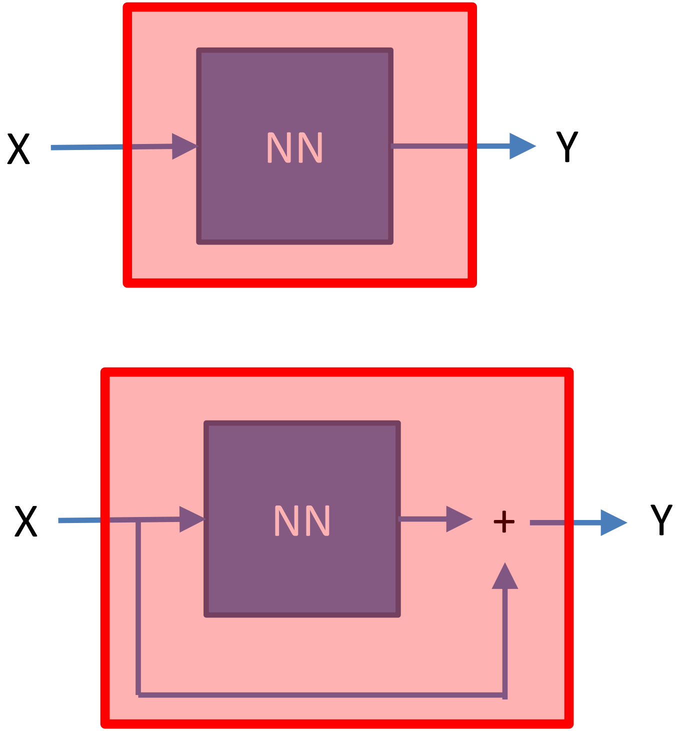
ResNet: Another Form of NN

X	Y	Y-X
1	0.9	-0.1
2	2.1	0.1
3	3.0	0.0
4	4.2	0.2

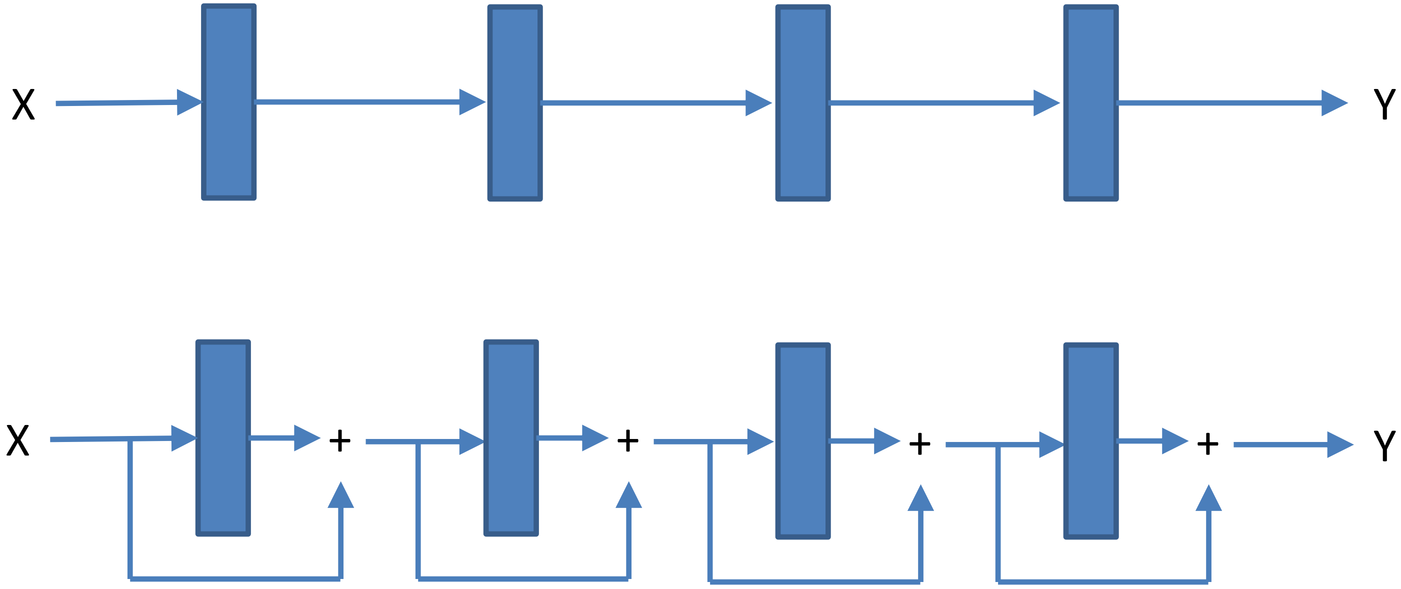


ResNet: Another Form of NN

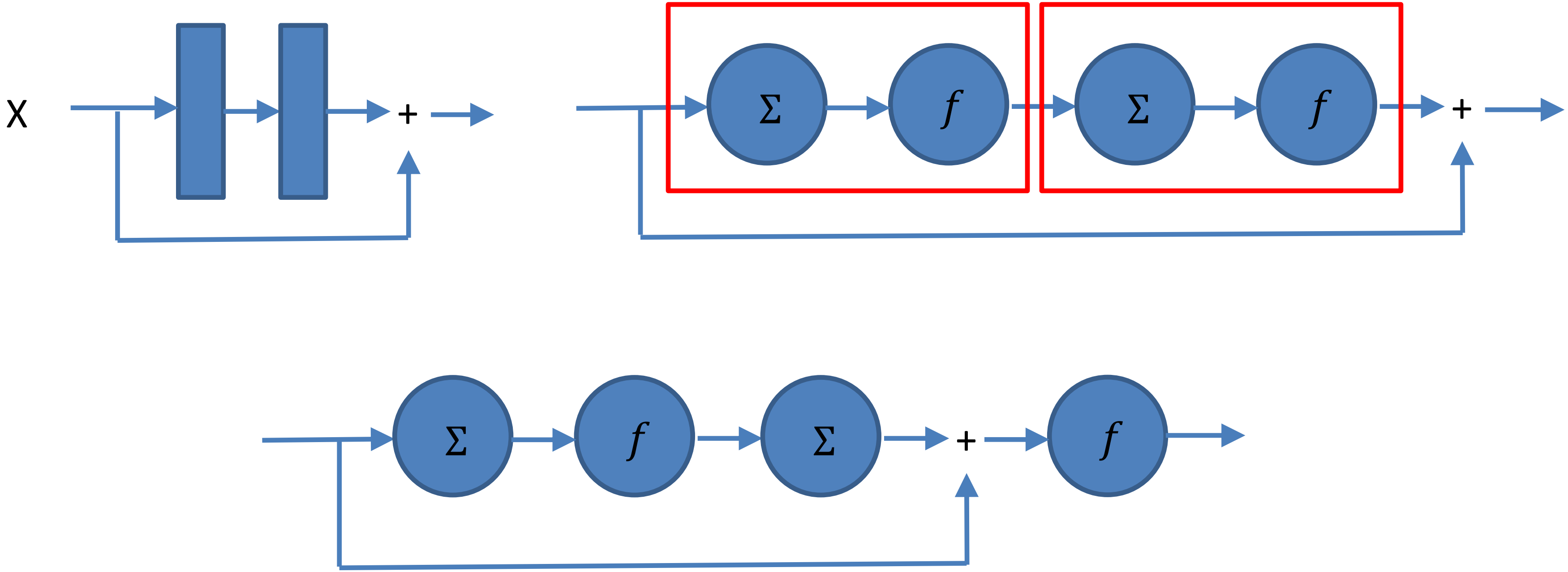
X	Y
1	0.9
2	2.1
3	3.0
4	4.2



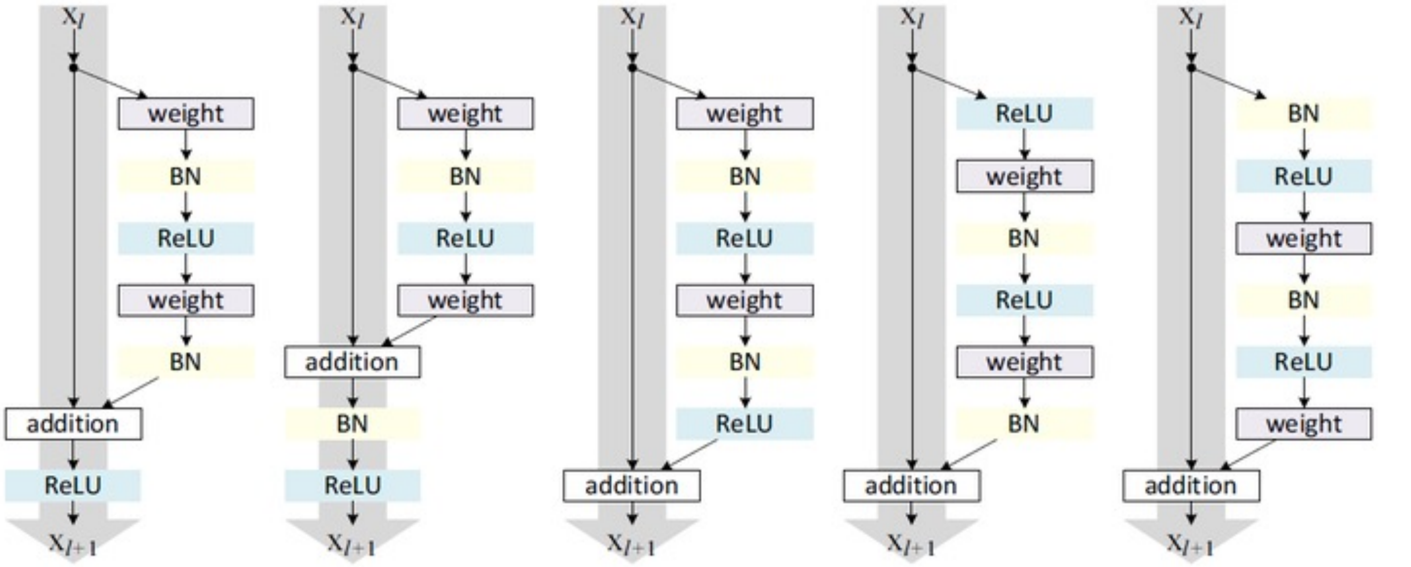
ResNet: Another Form of NN



ResNet



ResNet



(a) original

(b) BN after addition

(c) ReLU before addition

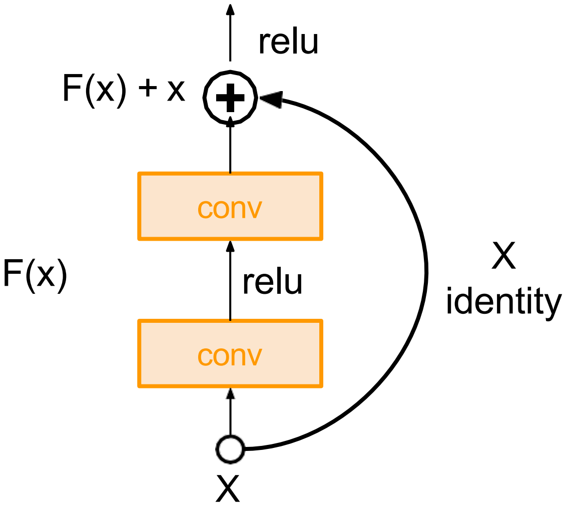
(d) ReLU-only pre-activation

(e) **full pre-activation**

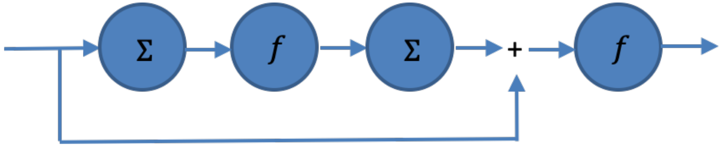
ResNet

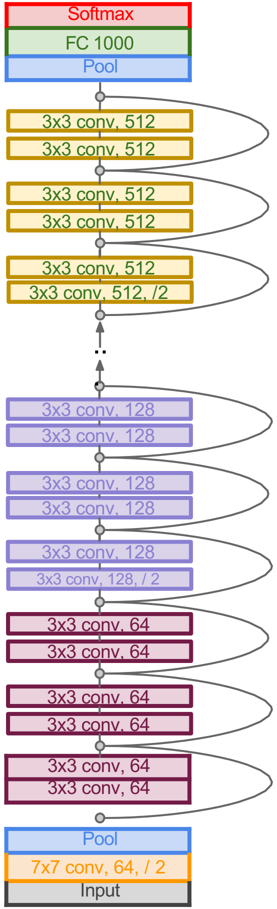
Very deep networks using residual connections

- Stack residual blocks
- Every residual block has two 3x3 conv layers
- Periodically, double # of filters and downsample spatially using stride 2
- Additional conv layer at the beginning
- Global average pooling layer after last conv. layer



Residual block





Case Study: ResNet

[He et al., 2015]

Training ResNet in practice:

- Batch Normalization after every CONV layer
- Xavier/2 initialization from He et al.
- SGD + Momentum (0.9)
- Learning rate: 0.1, divided by 10 when validation error plateaus
- Mini-batch size 256
- Weight decay of 1e-5
- No dropout used

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

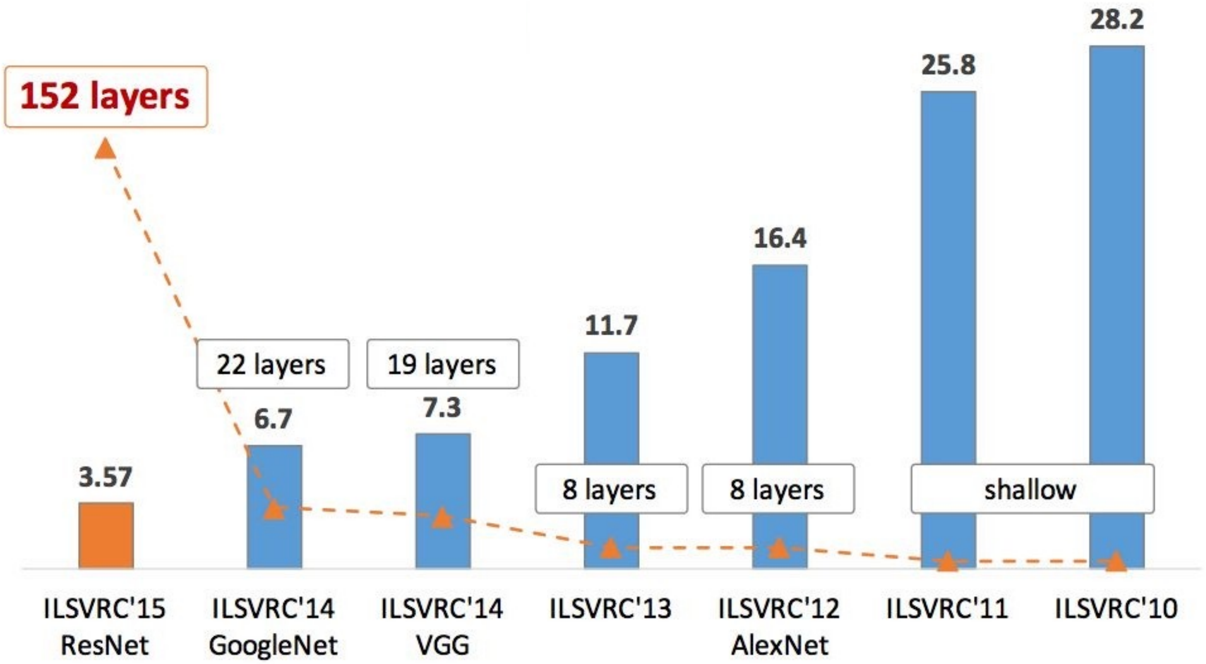
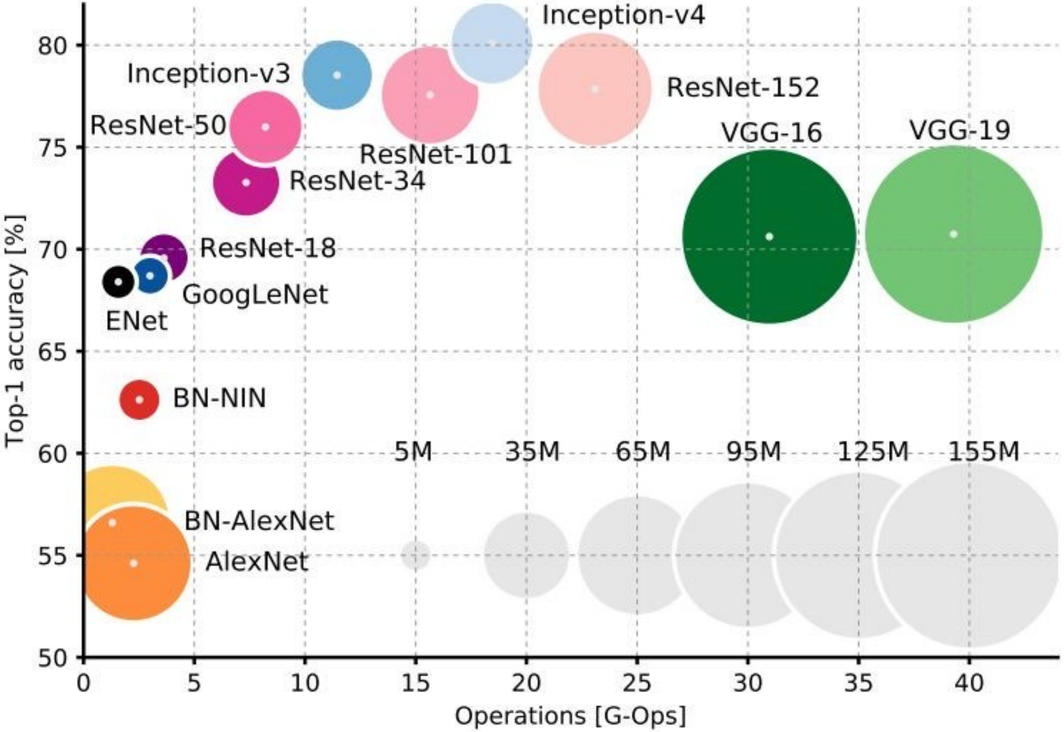


Figure copyright Kaiming He, 2016. Reproduced with permission.

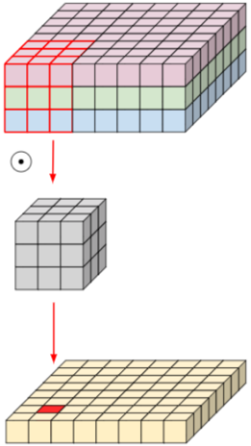
Comparing complexity...



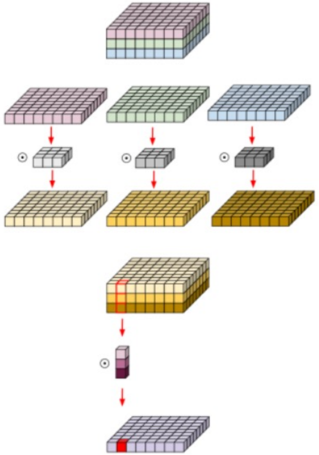
An Analysis of Deep Neural Network Models for Practical Applications, 2017.

Depthwise Separable Convolution: Much Lighter Conv.

Regular Conv

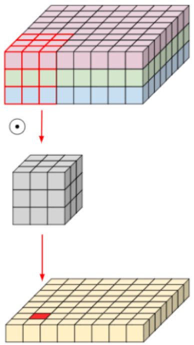


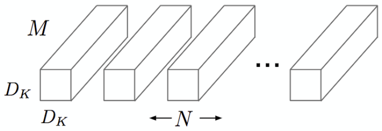
Depthwise Separable Conv



Depthwise Separable Convolution: Much Lighter Conv.

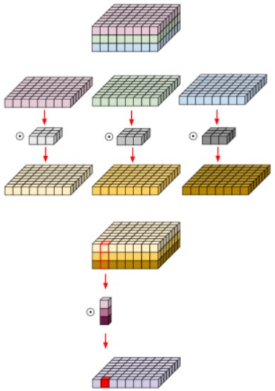
Regular Conv

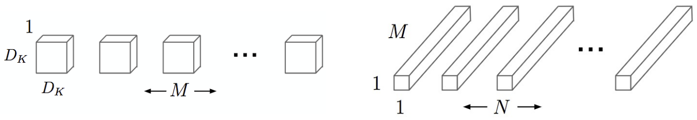




$$D_K \times D_K \times M \times N \times D_F \times D_F$$

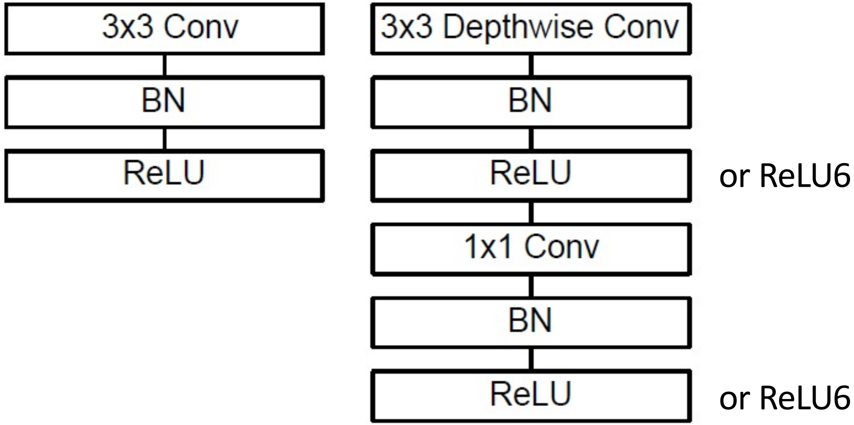
Depthwise Separable Conv





$$D_K \times D_K \times M \times D_F \times D_F + M \times N \times D_F \times D_F$$

Depthwise Separable Convolution: Much Lighter Conv.



or ReLU6

or ReLU6

Figure 3. Left: Standard convolutional layer with batchnorm and ReLU. Right: Depthwise Separable convolutions with Depthwise and Pointwise layers followed by batchnorm and ReLU.

Depthwise Separable Convolution: Much Lighter Conv.

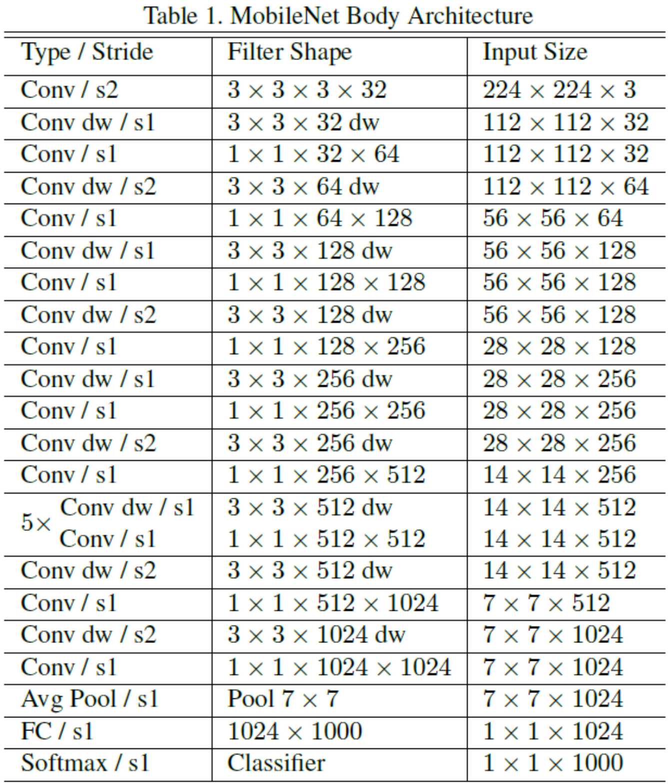


Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
5× Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

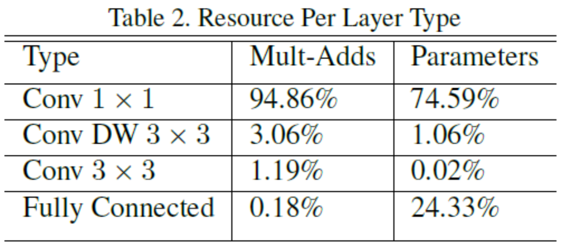


Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%