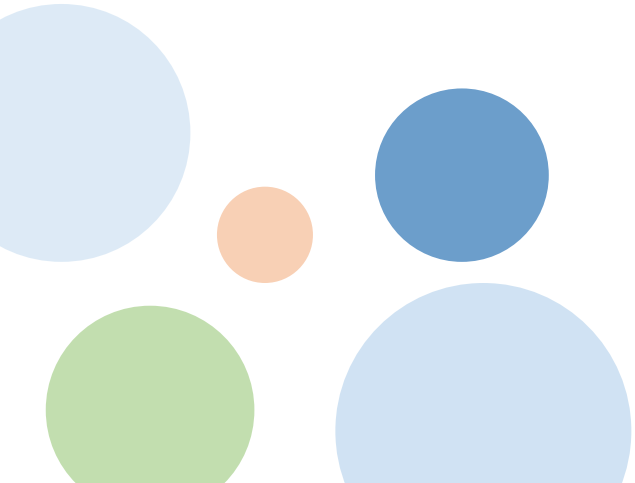
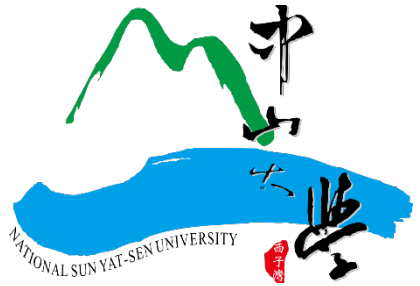
CNN Benchmark Models





Outlines

- LeNet (1998)
- AlexNet (2012)
- ZFNet (2013)
- VGG (2014)
- GoogLeNet (2014)
- ResNet (2015)
- ResNeXT (2016)
- MobileNet (2017)

ImageNet Dataset

- ~14 million labeled images, 20k classes
- Images gathered from Internet
- Human labels via Amazon MTurk
- ImageNet Large-Scale Visual Recognition Challenge (ILSVRC): 1.2 million training images, 1000 classes

www.image-net.org/challenges/LSVRC/

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners

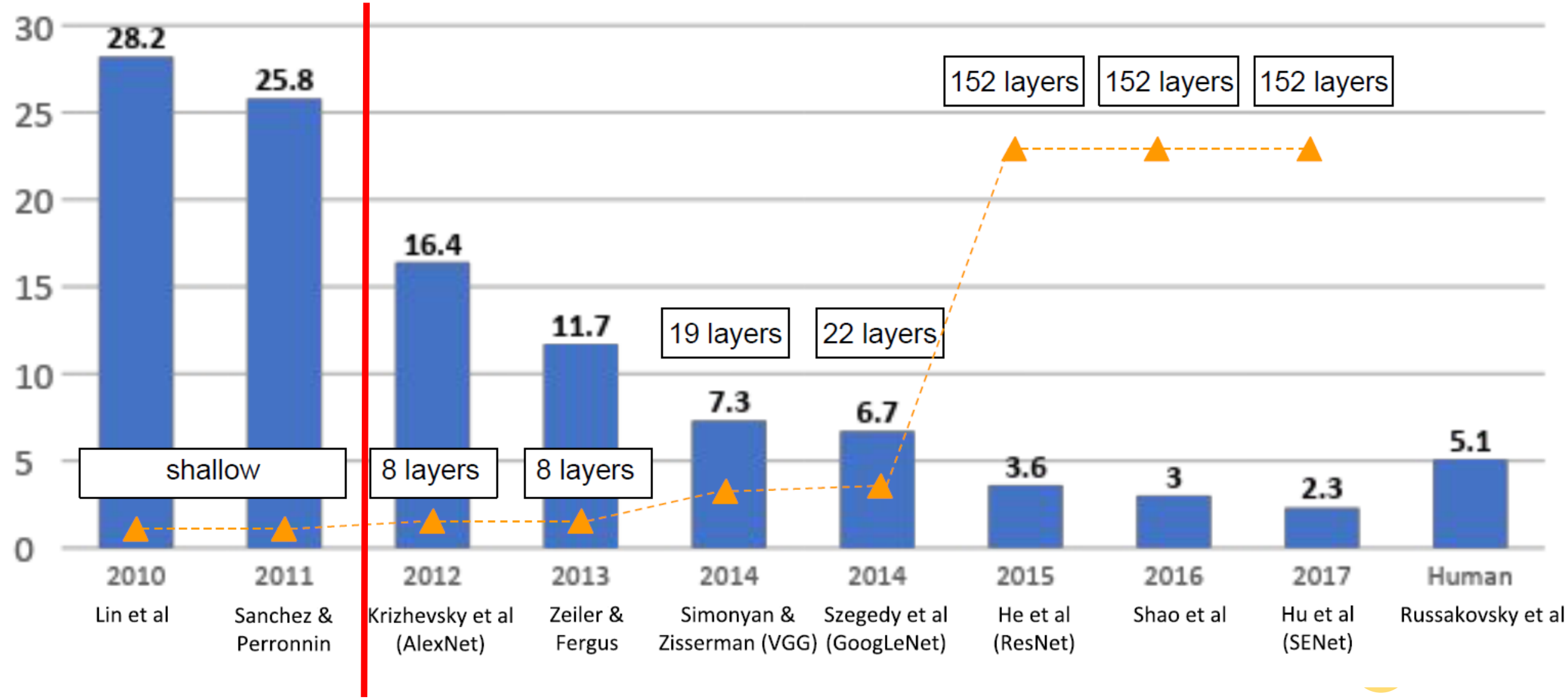


Image classification

Easiest classes

'ed fox (100) hen-of-the-woods (100) ibex (100) goldfinch (100) flat-coated retriever (100)





tiger (100) hamster (100) porcupine (100) stingray (100) Blenheim spaniel (100)





Hardest classes

muzzle (71) hatchet (68) water bottle (68) velvet (68) loupe (66)





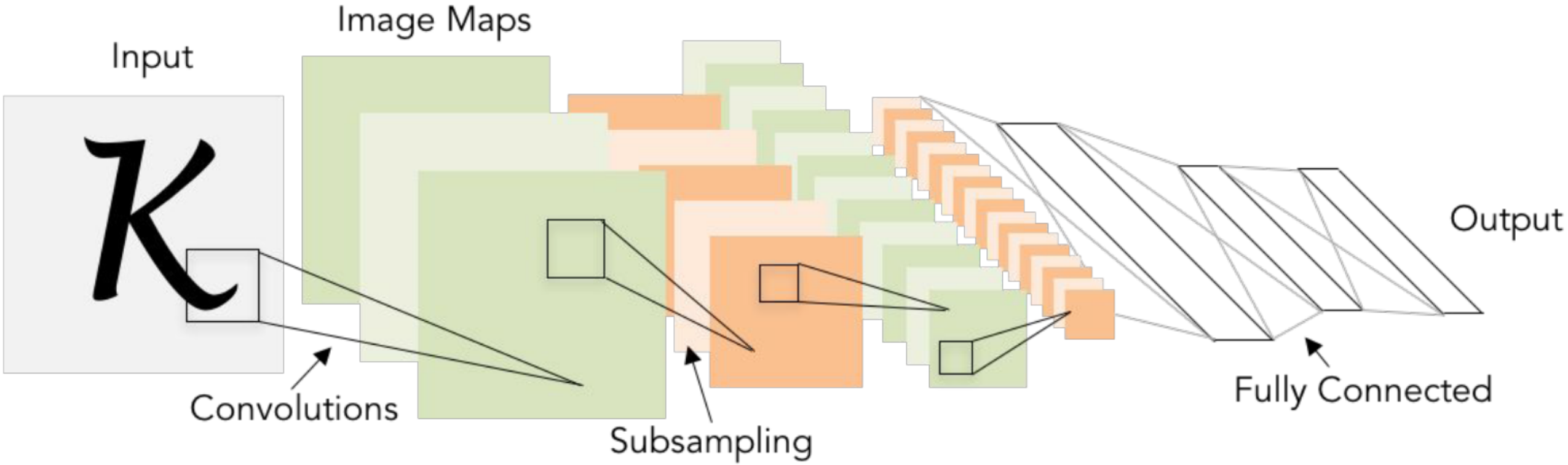
hook (66) spotlight (66) ladle (65) restaurant (64) letter opener (59)





LeNet* (1/3)

first CNN model, published in 1998
Training dataset: MNIST, recognition of 0~9 digits



Cony 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]

[LeCun et al., 1998]

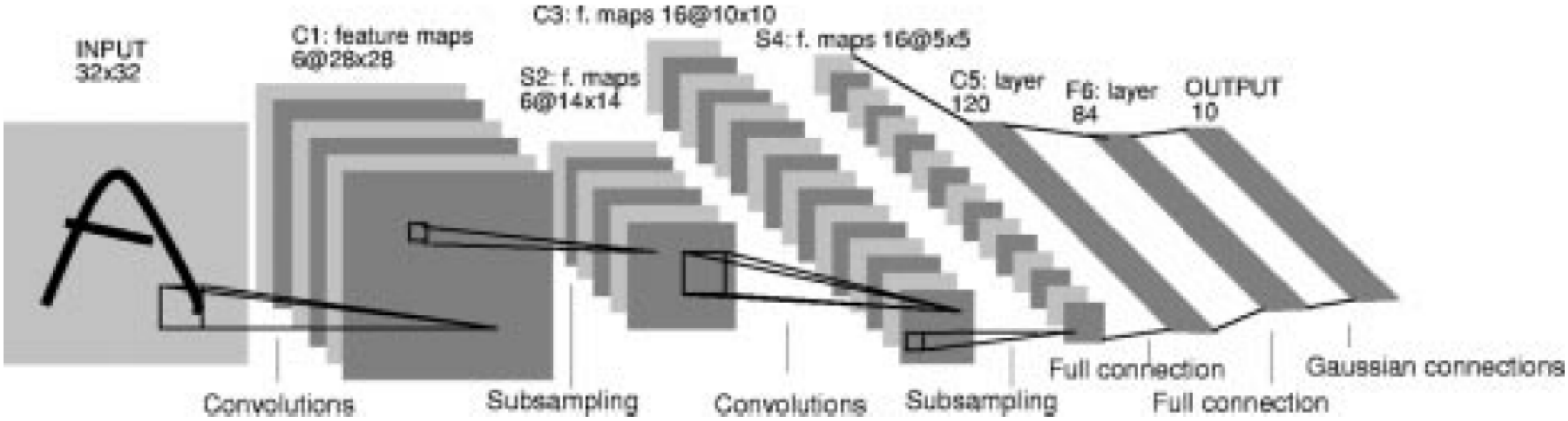
*Y. LeCun, et al., "Gradient-Based Learning Applied to Document Recognition," Proc. IEEE, 1998. (AT&T Bell Lab)

LeNet (2/3)

- C1: 5x5x1x6 convolution (32x32 -> 28x28)
- S2: 2x2 avg. pooling with stride=2 (28x28 -> 14x14)
- C3: 5x5x**{3,4,6}**x16 convolution (14x14 -> 10x10)
- S4: 2x2 avg. pooling with stride=2 (10x10 -> 5x5)
- C5: 5x5x16x120 convolution (5x5 -> 1x1)
- F6: 120x1 fully-connected (120 -> 84)

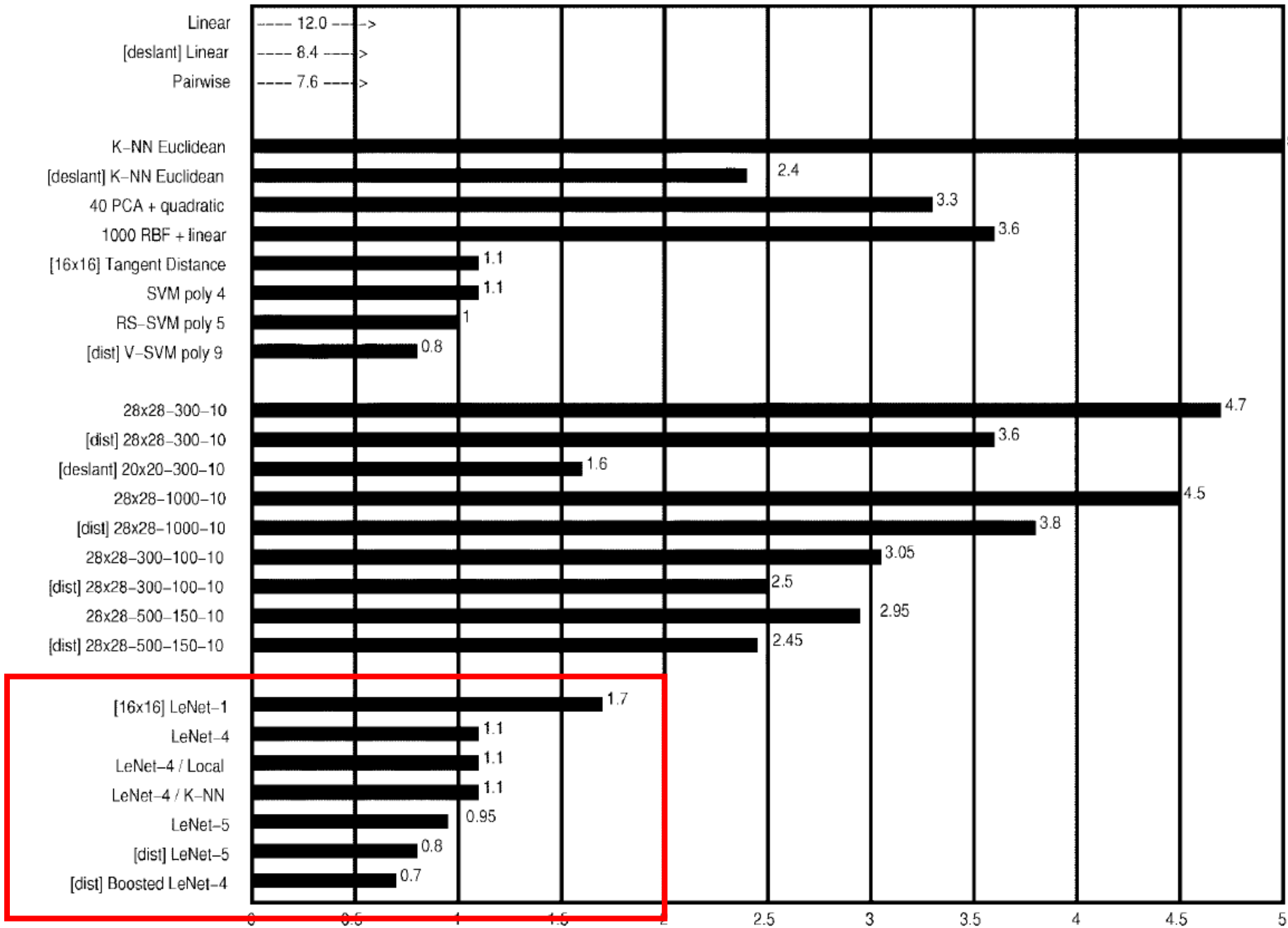
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	X				X	X	X			X	X	X	X		X	X
1	X	X				X	X	X			X	X	X	X		X
2	X	X	X				X	X	X			X		X	X	X
3		X	X	X			X	X	X	X			X		X	X
4			X	X	X			X	X	X	X		X	X		X
5				X	X	X			X	X	X	X		X	X	X

combination of input feature maps in C3



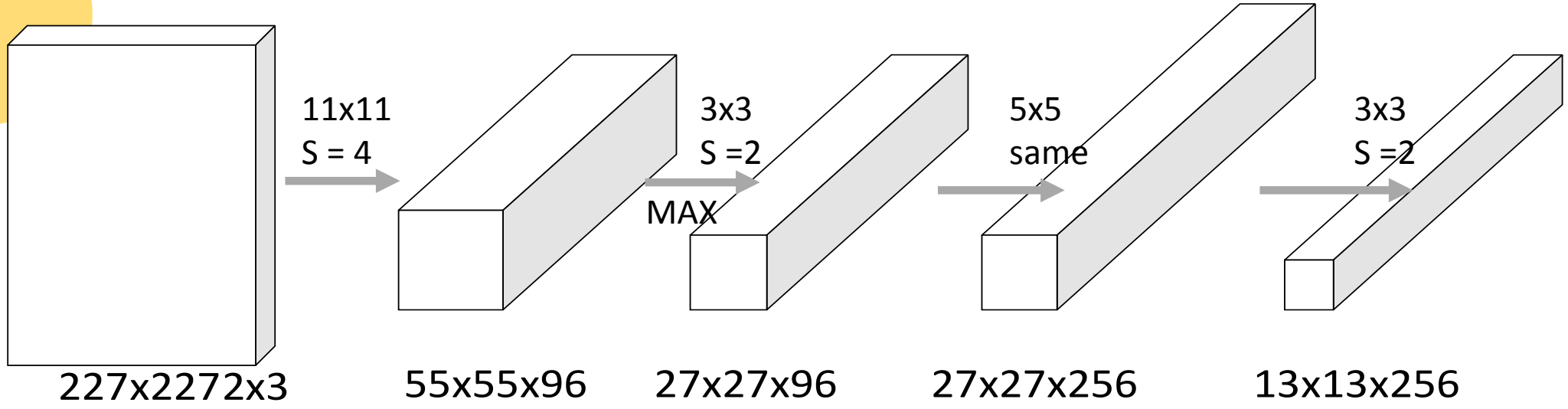
LeNet (3/3)

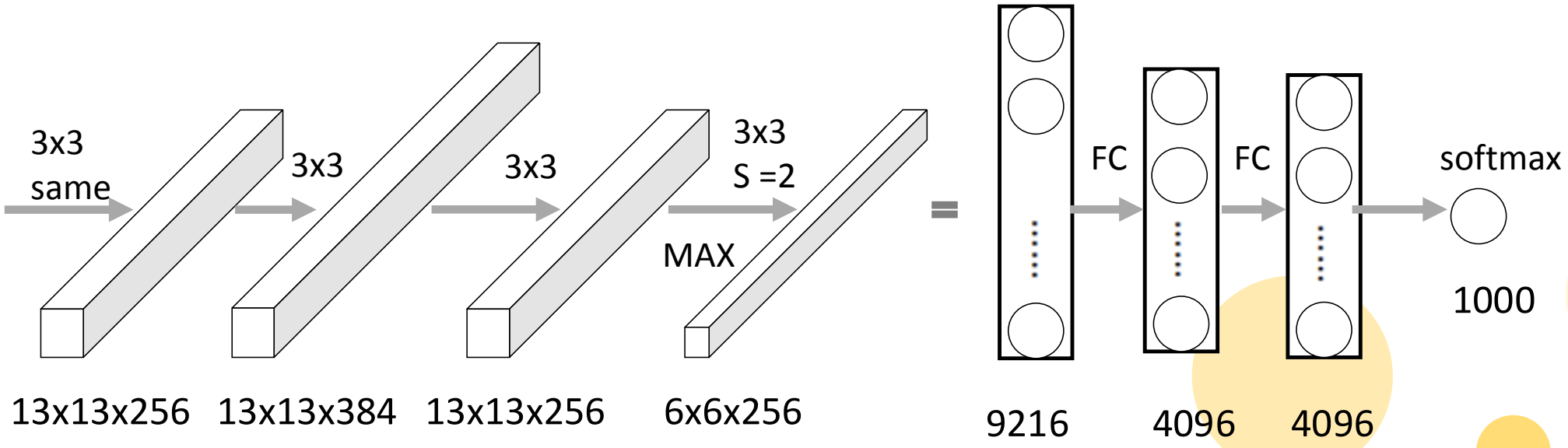
- compared with
  - linear classifier
  - kNN
  - PCA
  - SVM
  - ANN (with various hidden layers)
  - LeNet-5 and its variants



AlexNet (1/3)

2012 ILSVRC winner, training dataset: ImageNet

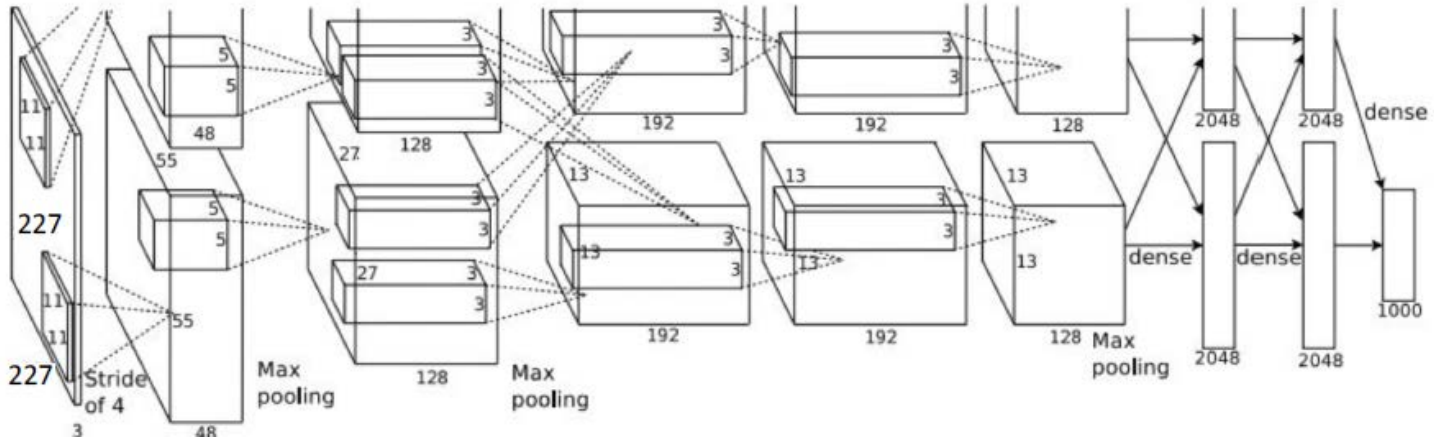




* Alex Krizhevsky, ea tal., “ImageNet Classification with Deep Convolutional Neural Networks,” NIPS 2012. (Univ. Toronto)

AlexNet (2/3)

- first CNN-based ILSVRC winner
  [Krizhevsky et al. 2012]

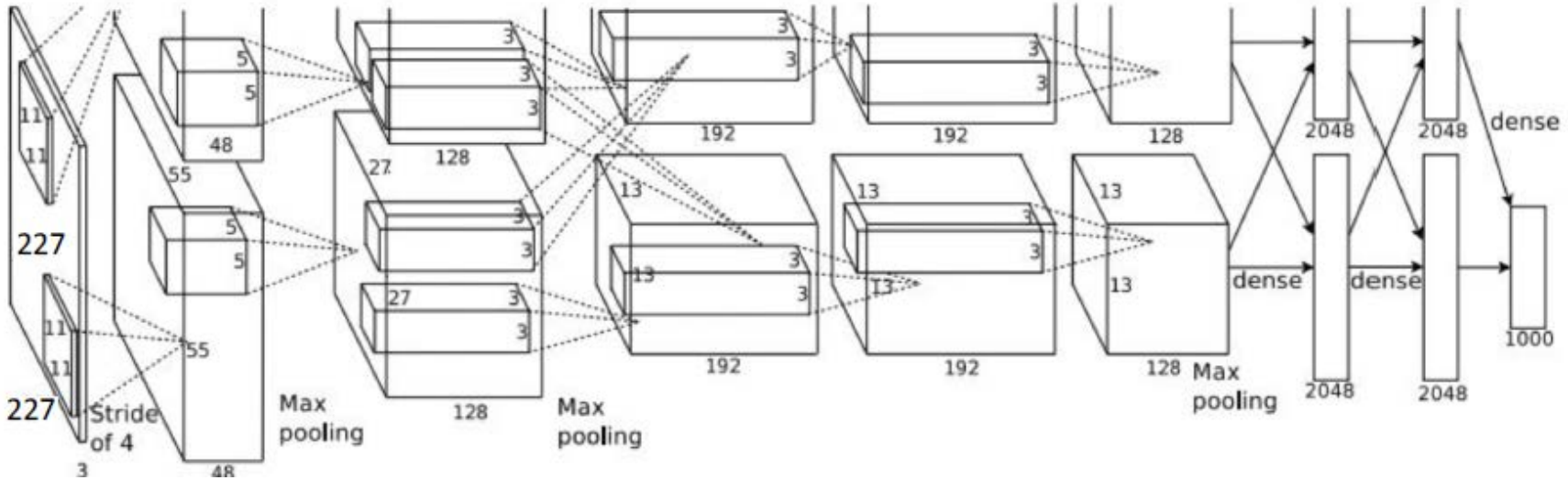


Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

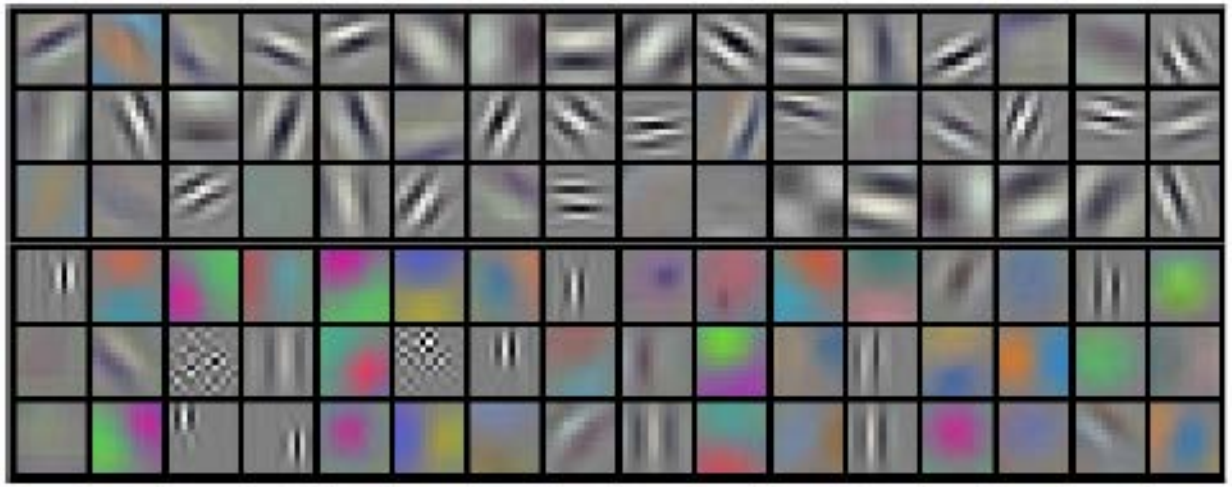
Details/Retrospectives:

- first use of ReLU
- used Local Response Normalization (LRN) layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- mini-batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2. reduced by 10 manually
- L2 weight decay 5e-4

AlexNet (3/3)

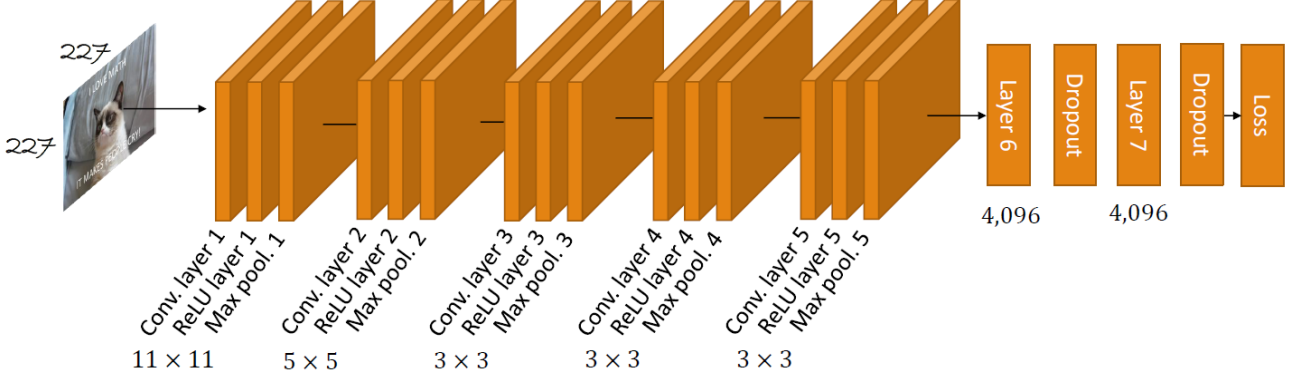


- 5 Conv + 3 FC
  - 60M parameters
  - filter sizes: 11x11, 5x5, 3x3
- trained on two GPUs
- first to use
  - deep (8-layer) CNN in ILSVRC
  - ReLU for non-linear activation function
  - dropout for regularization
  - local response normalization (LRN)
- data augmentation
- 1st place of ILSVRC 2012 with **15.3%** top-5 error rate
  - 2nd place has 26.2% top-5 error rate

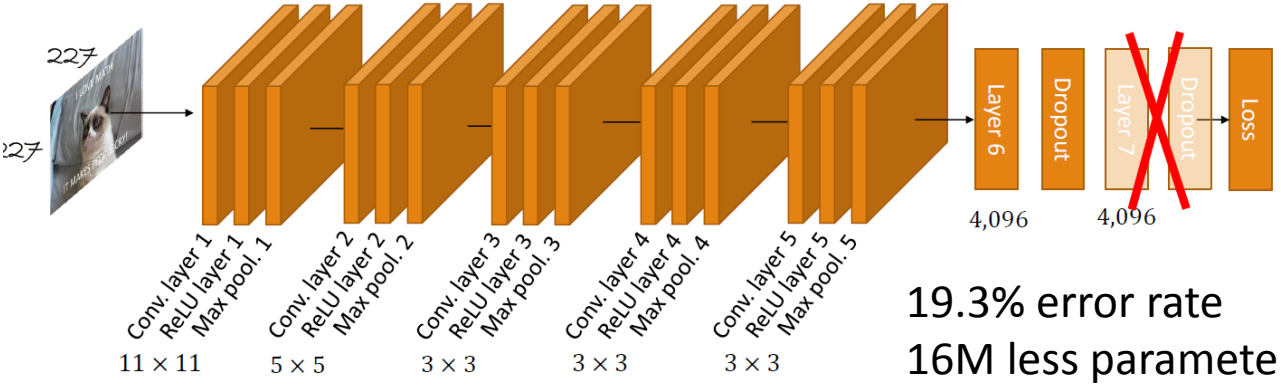


- 96 convolutional kernels of size 11x11x3 in the first Conv. layer
- learned a variety of frequency- and oriented-selected kernels, and colored blobs

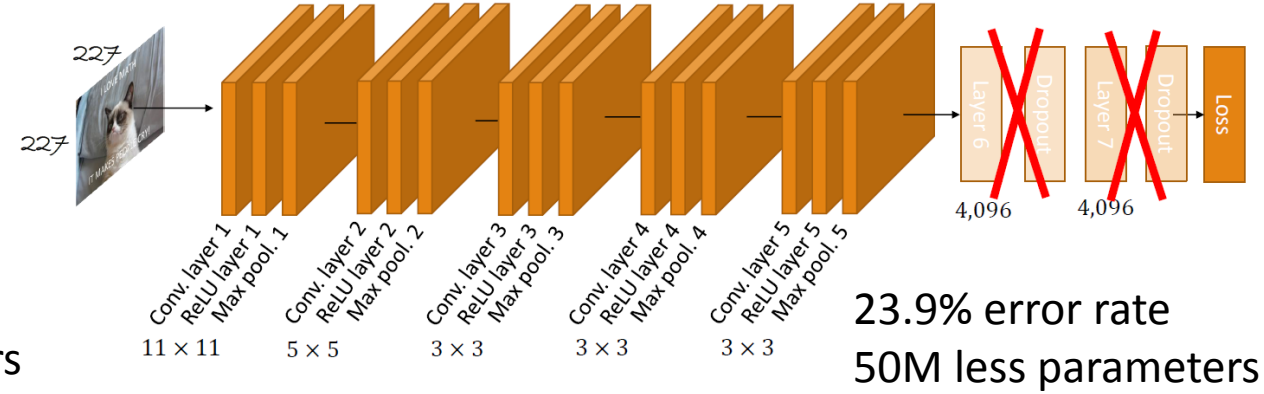
AlexNet variants



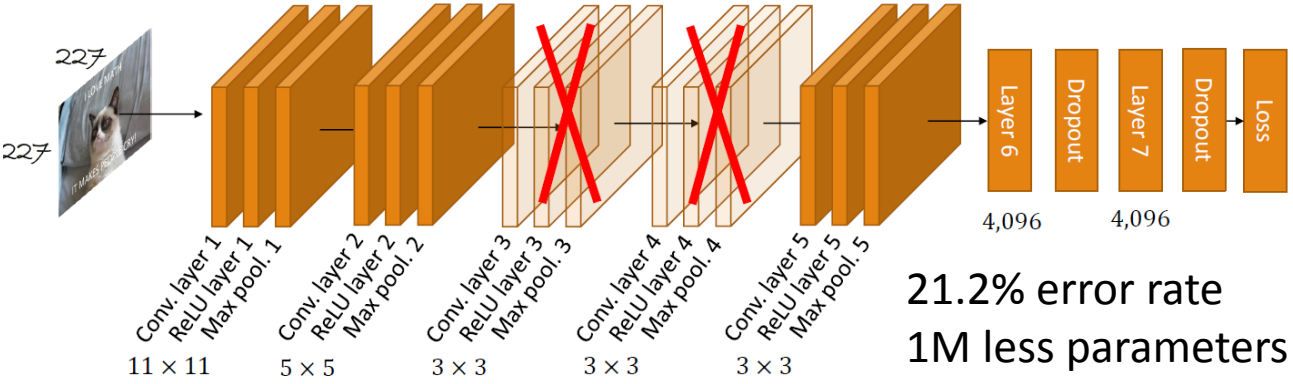
18.2% error rate



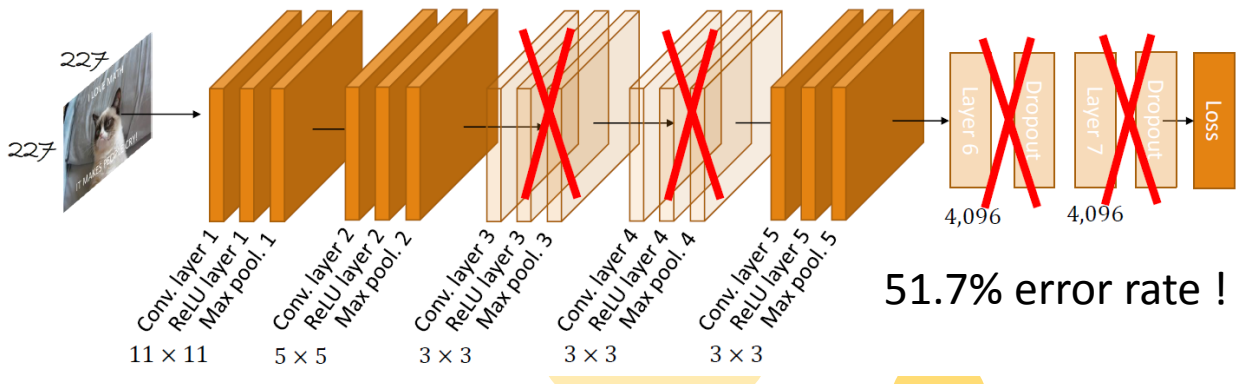
19.3% error rate
16M less parameters



23.9% error rate
50M less parameters



21.2% error rate
1M less parameters

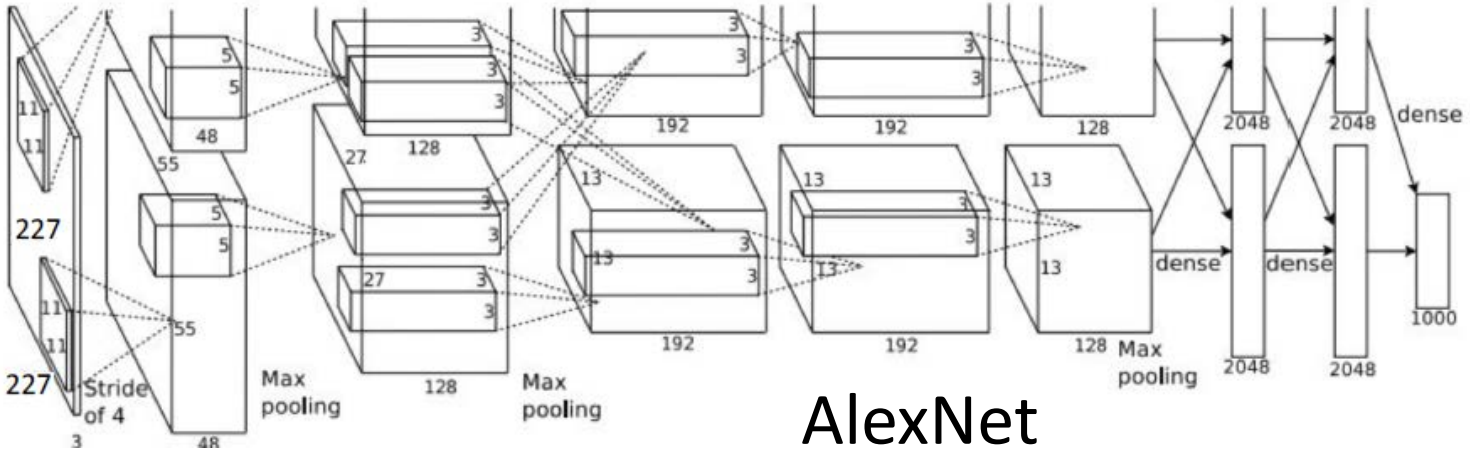


51.7% error rate !

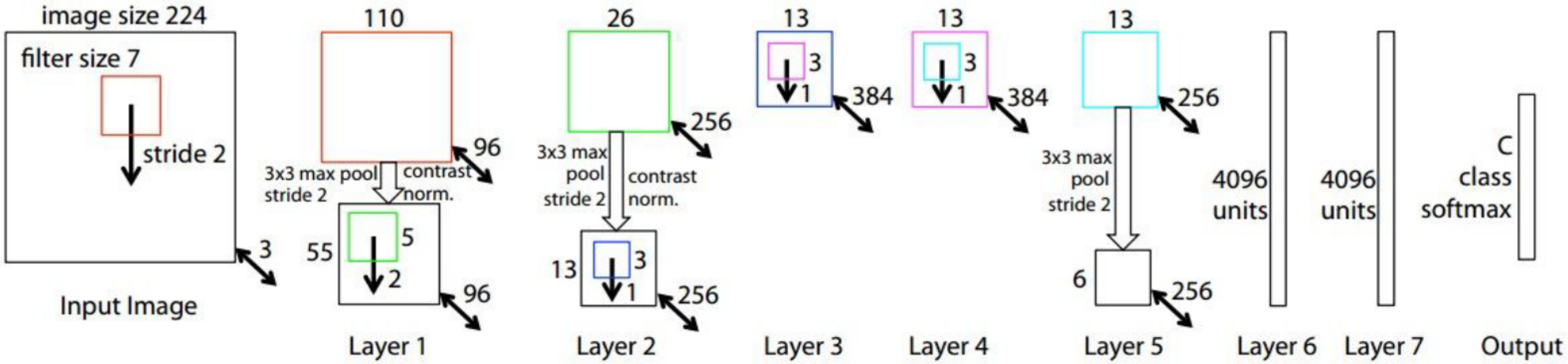
* UVA Deep Learning Course, by Efstratios Gavves

ZFNet* (1/3)

- improve hyper-parameters over AlexNet
- Visualization of CNN



ZFNet



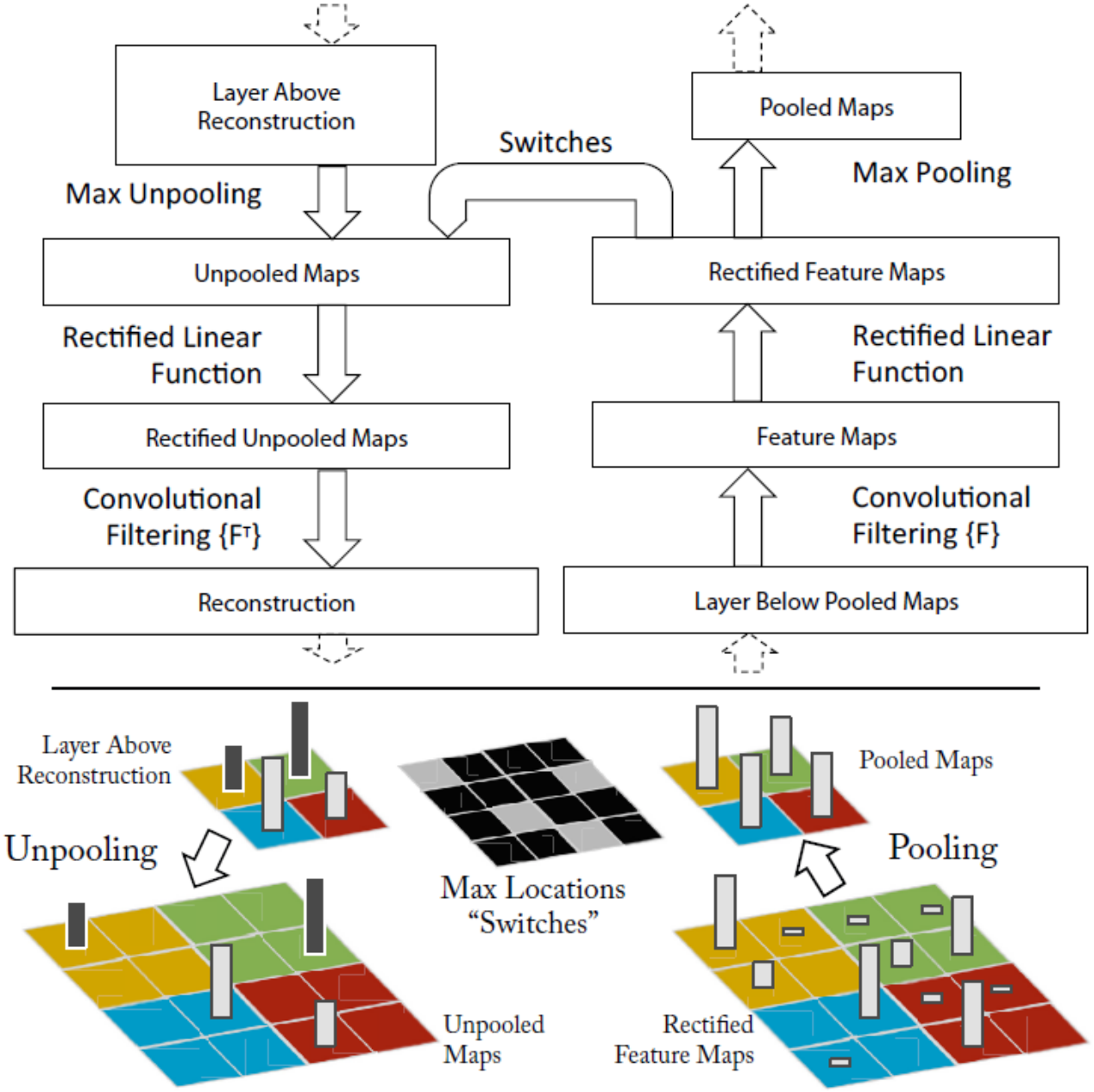
Similar to AlexNet, but:
CONV1: change from (11 x11 stride 4) to (7x7 stride 2)
CONV3,4,5: instead of 384. 384, 256 filters use 512, 1024, 512

ImageNet top 5 error: 16.4% -> 11.7%

*M. D. Zeiler and R. Fergus, "Visualizing and Understanding Convolutional Networks," ECCV, 2014, (NYU)

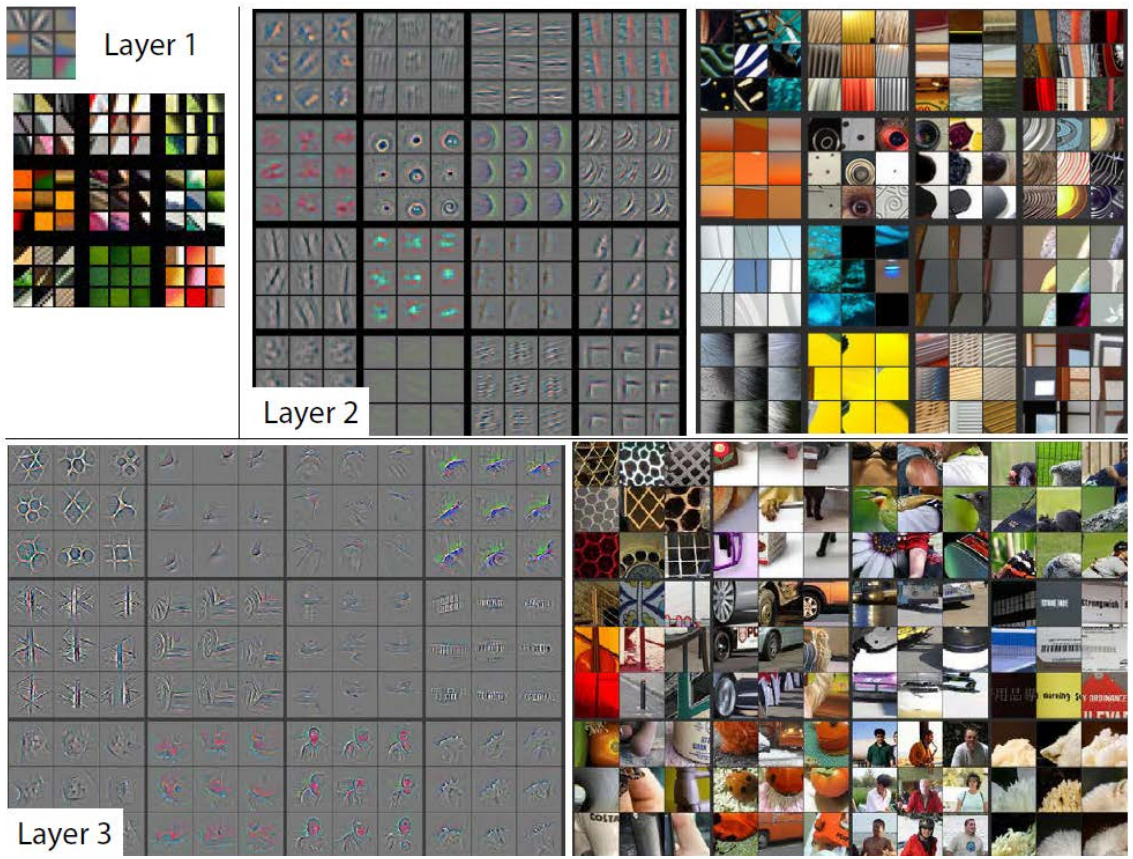
Visualization

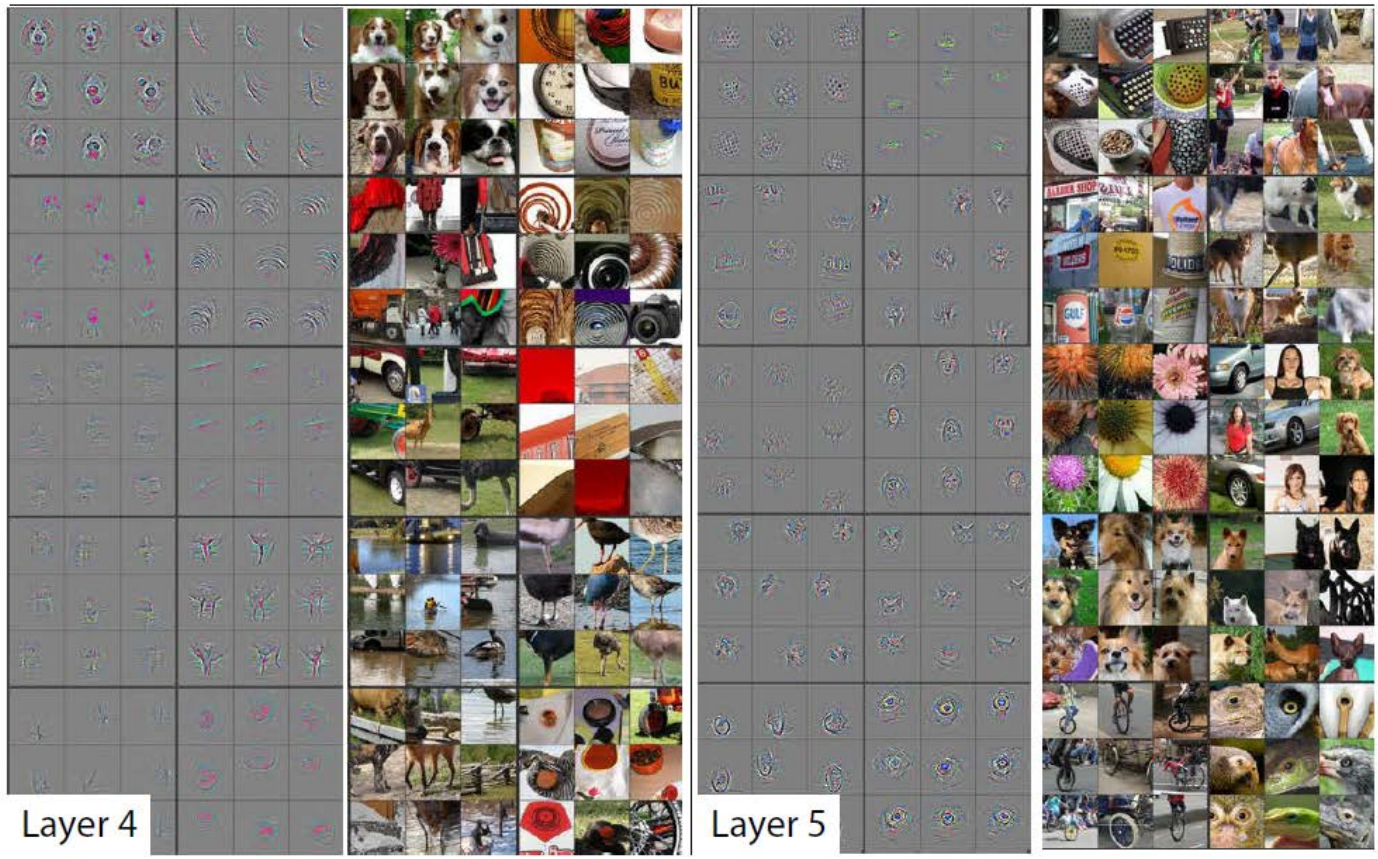
- map activations back to the input pixel space, showing what input patterns originally caused a given activation in the feature maps
- a deconvnet is attached to each layers, providing a continuous path back to image pixel
- to examine a given convent activation, set all other activations in the layer to zero and pass the feature maps as input to the deconvnet, successfully (1) unpool, (2) rectify, (3) filter, to reconstruct the activity in the layer beneath that gave rise to the chosen activations



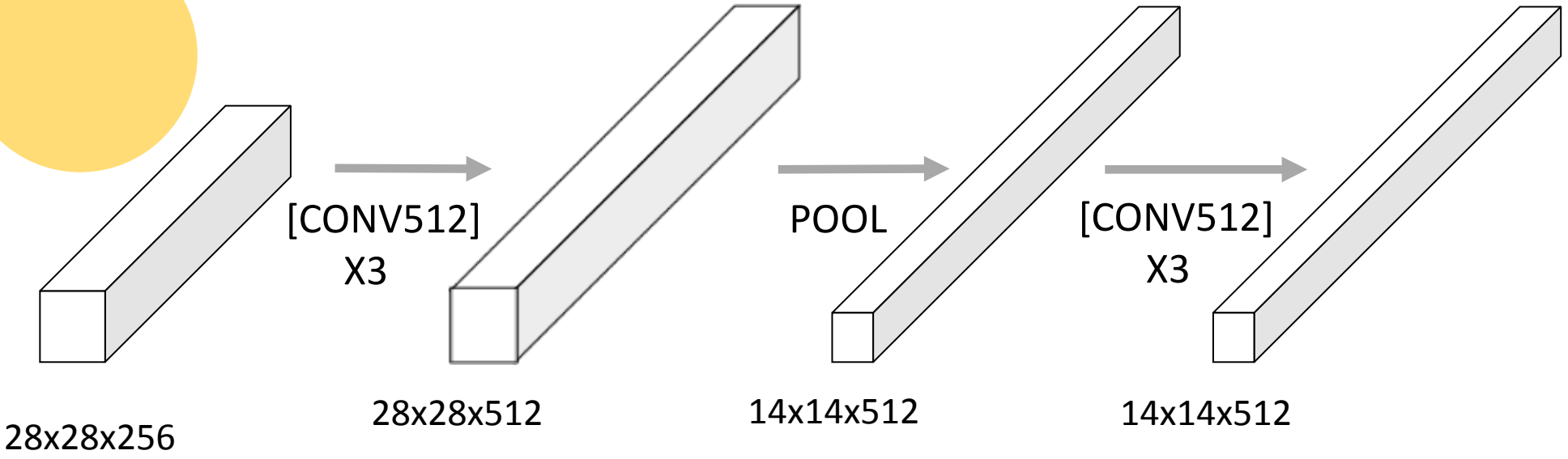
ZF-Net (3/3)

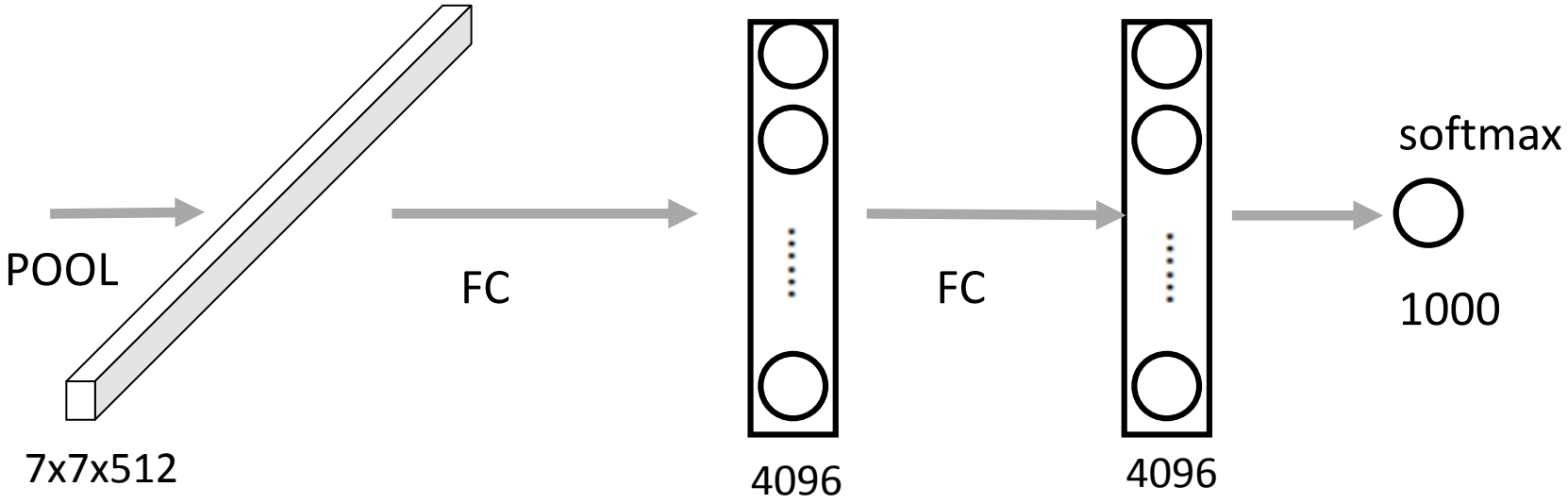
- top 9 activations in a random subset of feature maps across validation data projected down to pixel space
  - reconstructed patterns from validation set that cause high activations in a given feature map





VGG* (1/4)





VGG16

*K. Simonyan and A. Zisserman, "Very Deep Convolution Networks for Large-Scale Image Recognition," ICLR 2015.
(Visual Geometry Group, Oxford)

VGG (2/4)

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
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

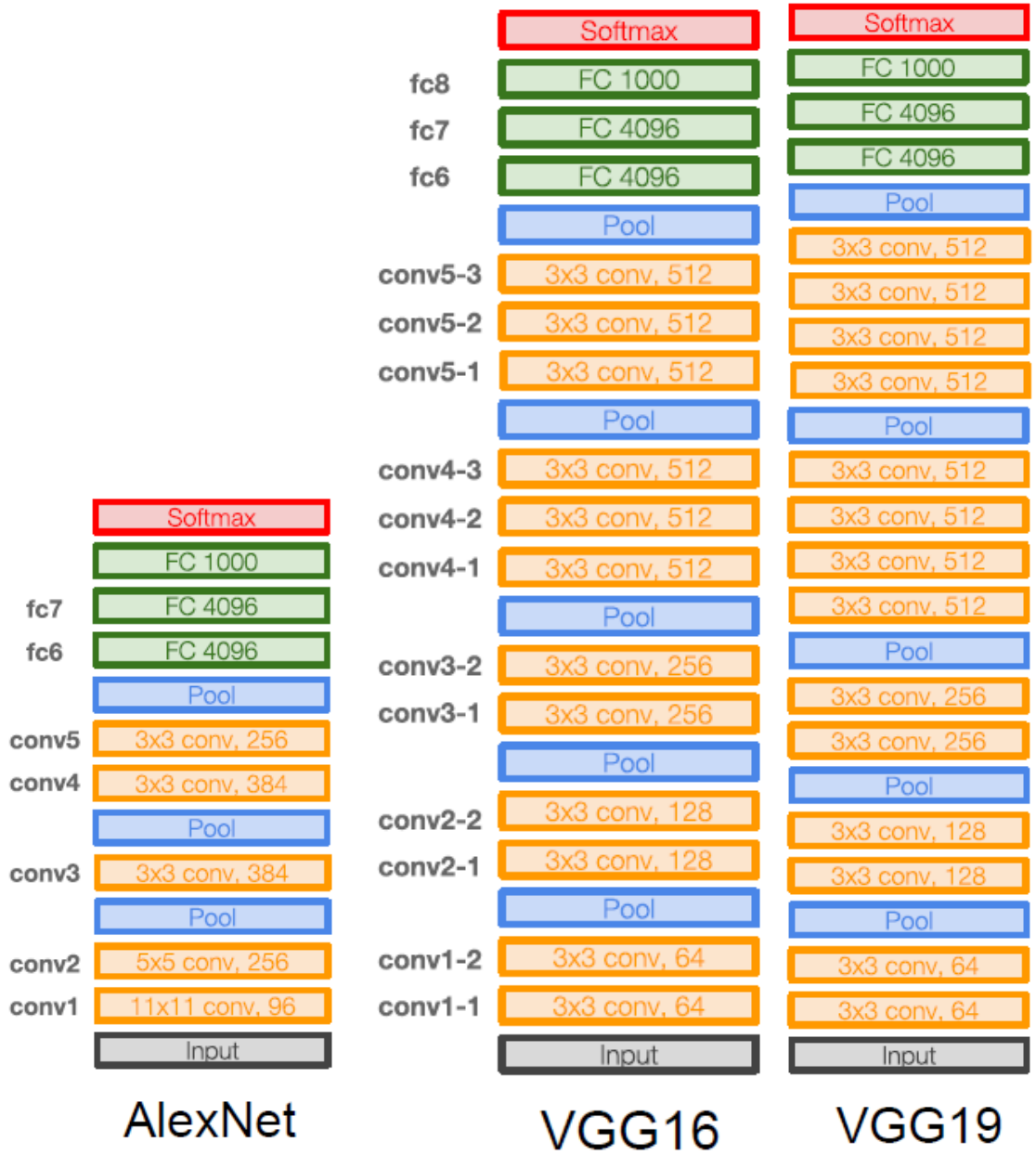
TOTAL memory: 24M * 4 bytes ~= 96MB / image (for a forward pass)
TOTAL parameters: 138M parameters

VGG16

VGG (3/4)

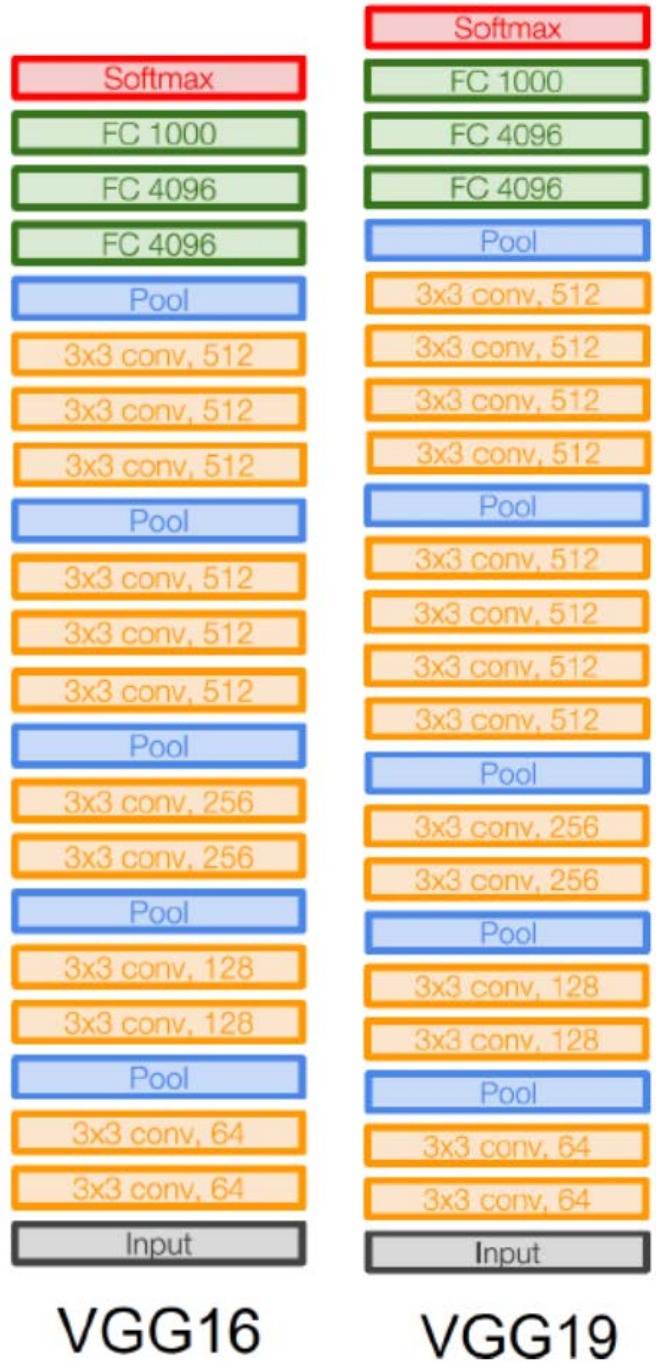
[Simonyan and Zisserman, 2014]

- ILSVRC'14 2nd in classification, 1st in localization
  - Similar training procedure as AlexNet
- No Local Response Normalization (LRN)
  - Use VGG16 or VGG19 (VGG19 only slightly better, more memory)
  - Use ensembles for best results
  - FC7 features generalize well to other tasks



VGG (4/4)

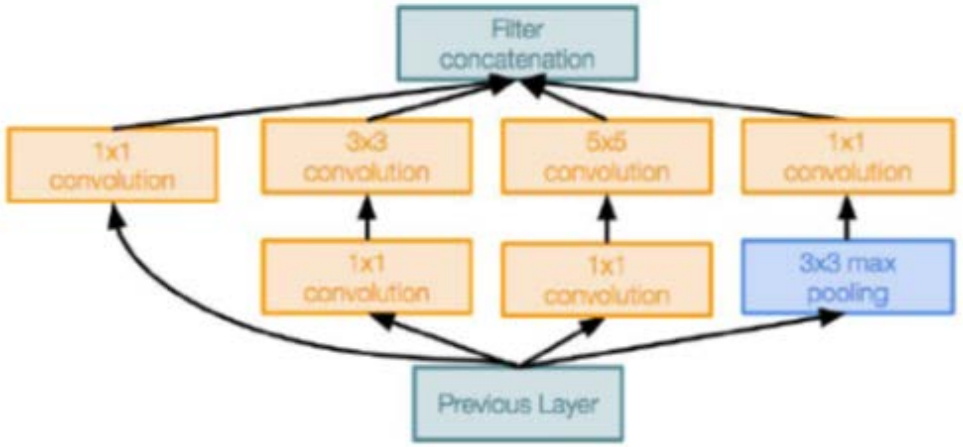
- 13~16 Conv + 3 FC
  - 138 M parameters (cp. 60M in AlexNet)
  - filter sizes: 3x3
- three 3x3 Conv layers has same receptive field as one 7x7 Conv layer
- no local response normalization (LRN)
- ILSVRC 2014 with top-5 error rate 7.3%
  - cp. 16.4% in AlexNet
- VGG Conv layers are used for feature extraction in many applications such as object detection



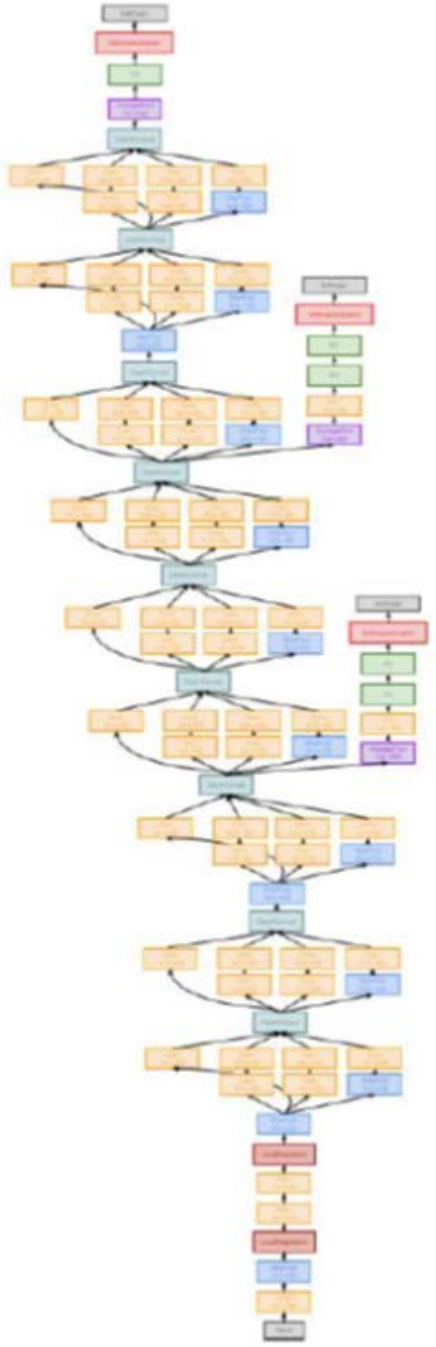
GoogLeNet*

- inception + 1x1 convolution
  - multiple kernel filters of different sizes (1x1, 3x3, 5x5)
  - objects have great variation in size
    - receptive field should vary in size accordingly

- 22 layers (+ 5 pooling 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

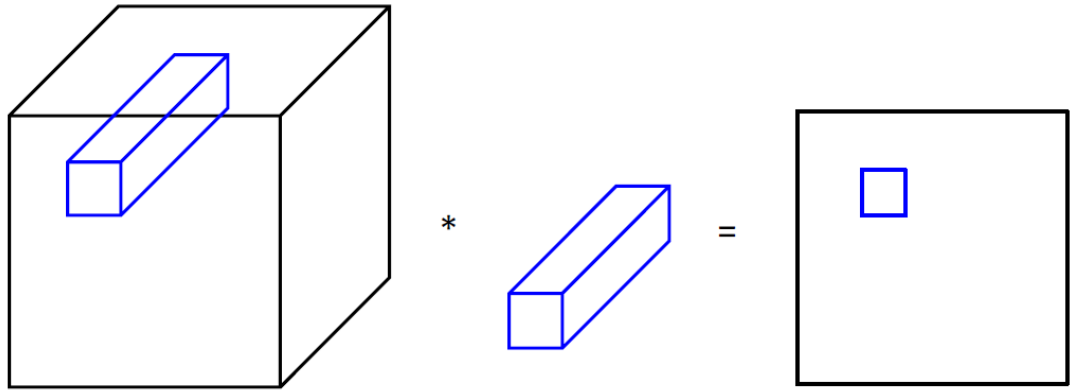


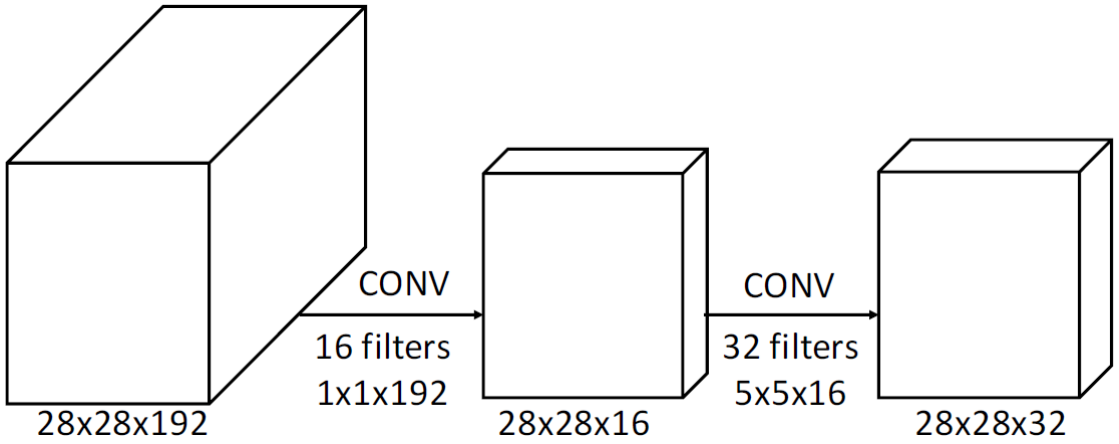
Deeper networks, with computational efficiency

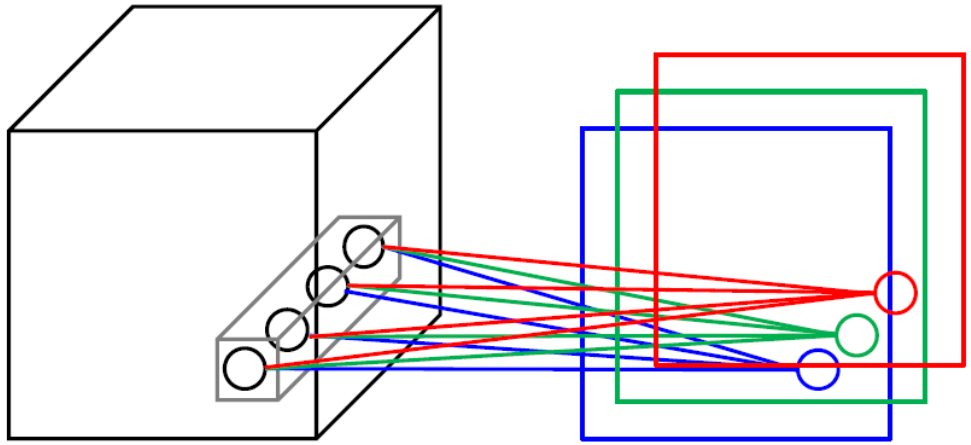
* C. Szegedy, et al., “Going Deeper with Convolutions,” CVPR, 2015. (Google)

1x1 Convolution

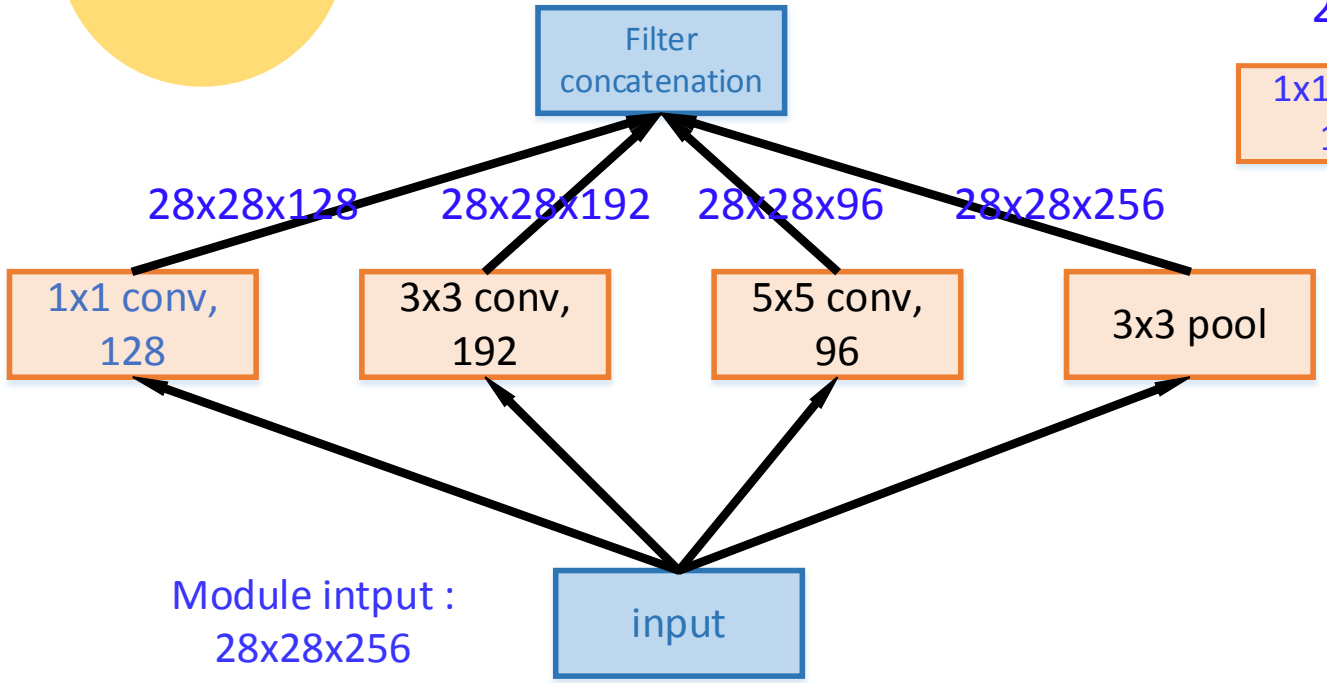
- weighted average across input channels (1x1 filter)
- different 1x1 convolution (for different output channels) apply different weights
  - different 1x1 filters generate different features
- 1x1 convolution can be used to shrink/expand number of channels
- more 1x1 convolution layers provide extra non-linearity
- similar idea of NiN (Network in Network)



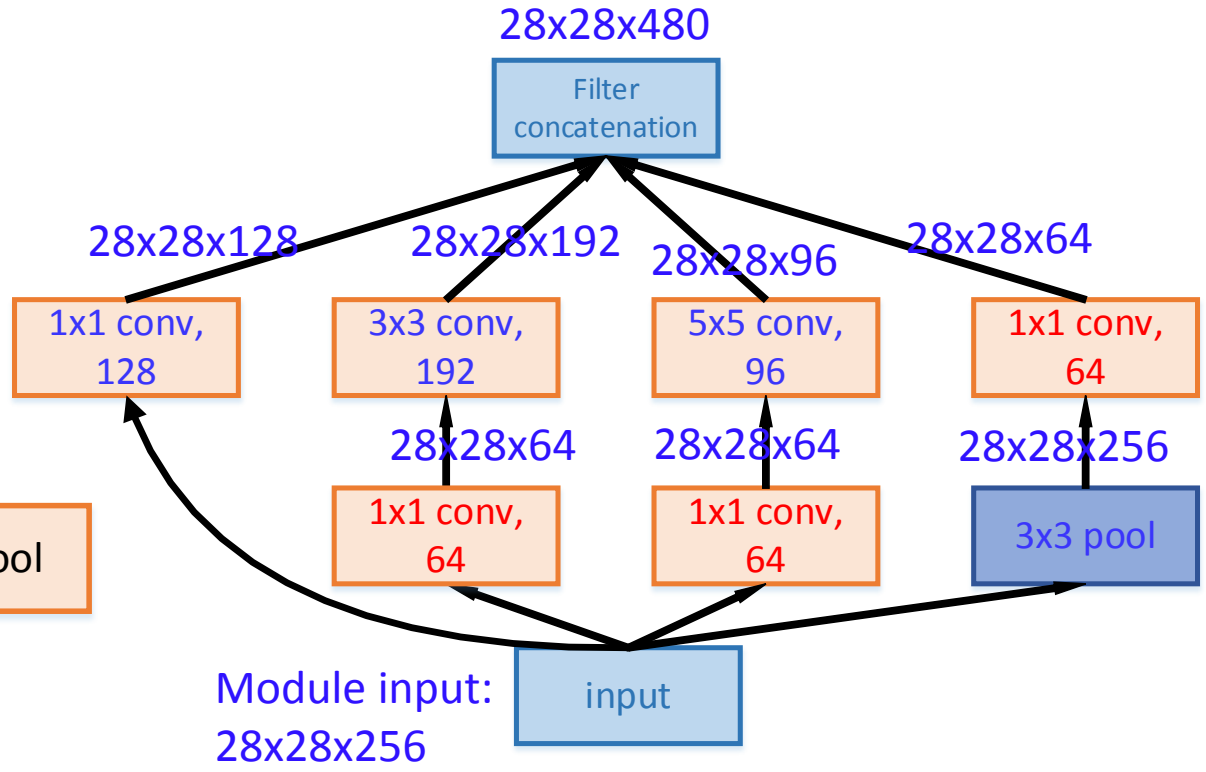




Inception



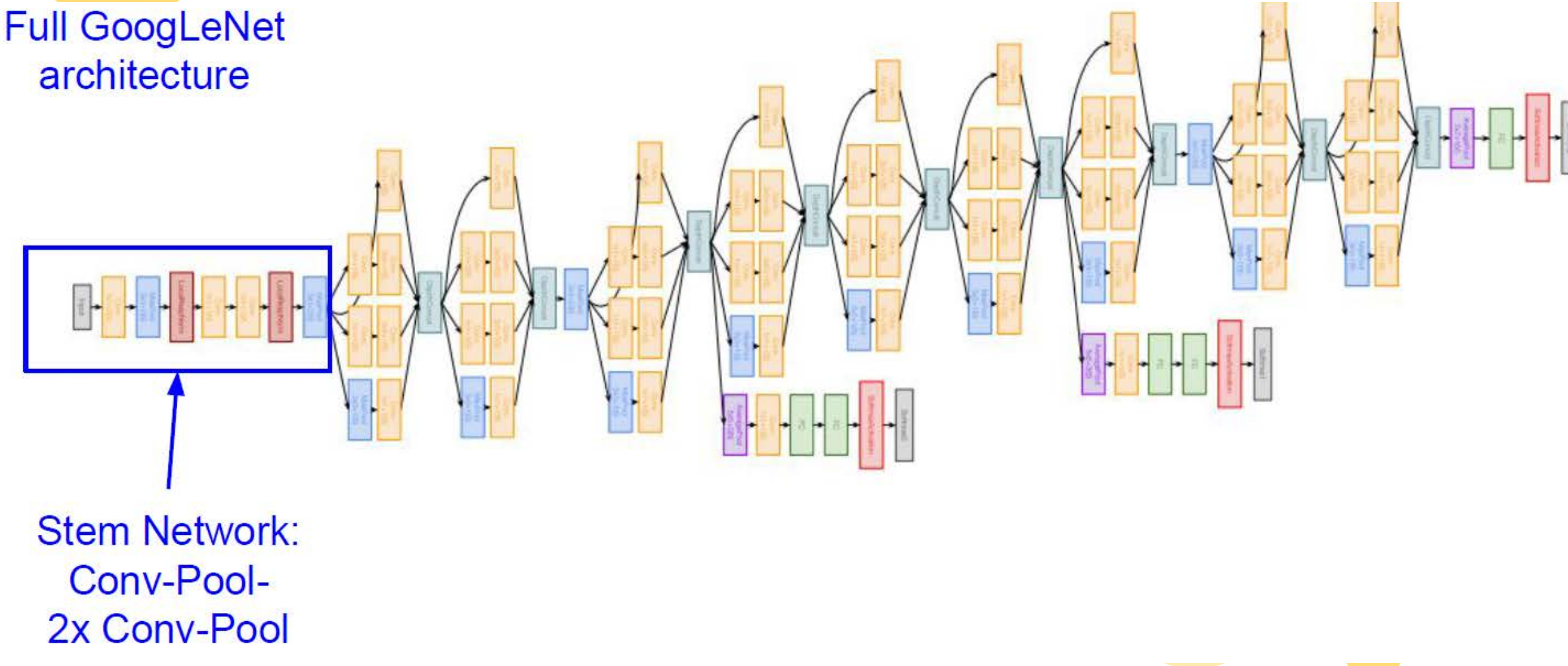
Conv Ops:
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x256
[5x5 conv, 96] 28x28x96x5x5x256
Total: 854M ops



Conv Ops:
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 128] 28x28x128x1x1x25
[3x3 conv, 192] 28x28x192x3x3x64
[5x5 conv, 96] 28x28x96x5x5x64
[1x1 conv, 64] 28x28x64x1x1x256
Total: 358M ops

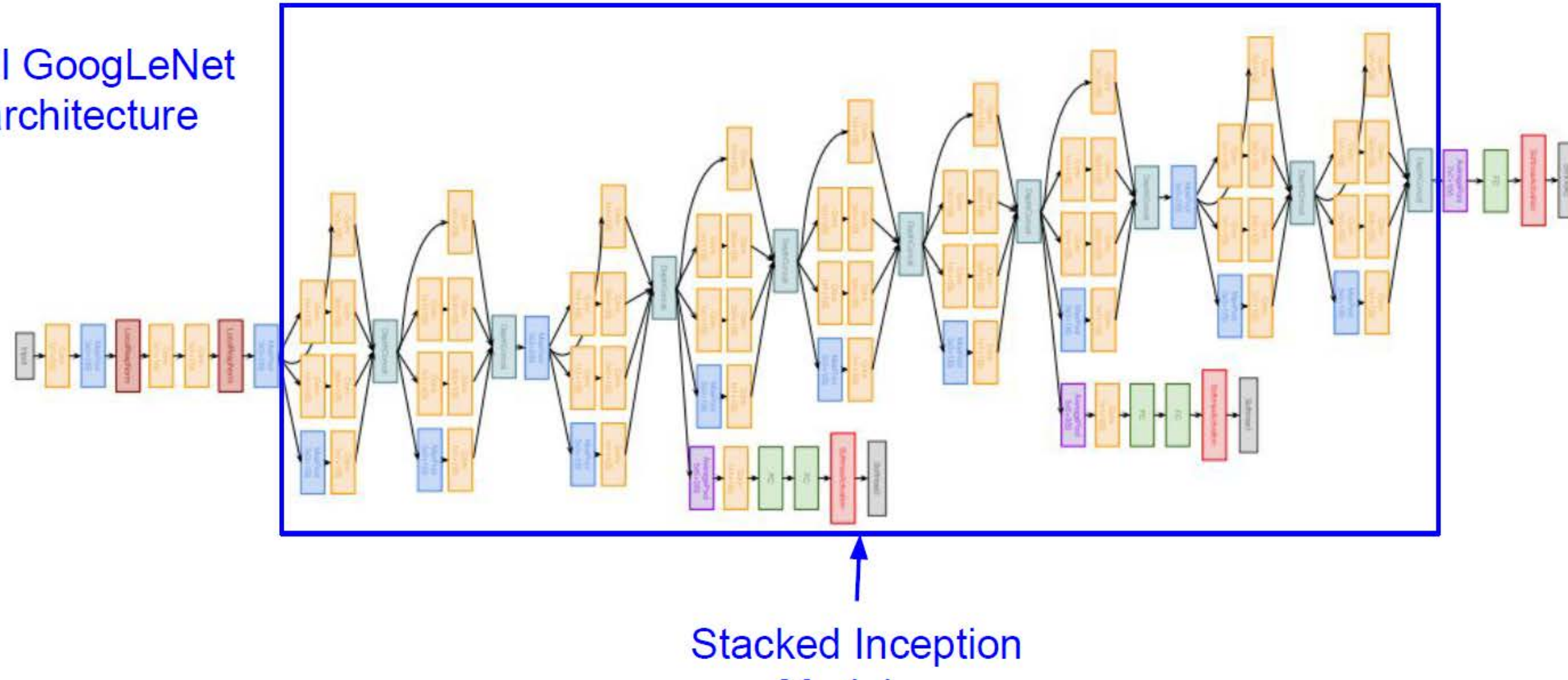
GoogLeNet Stem Network

- stem network of several Conv and Pool layers at the beginning



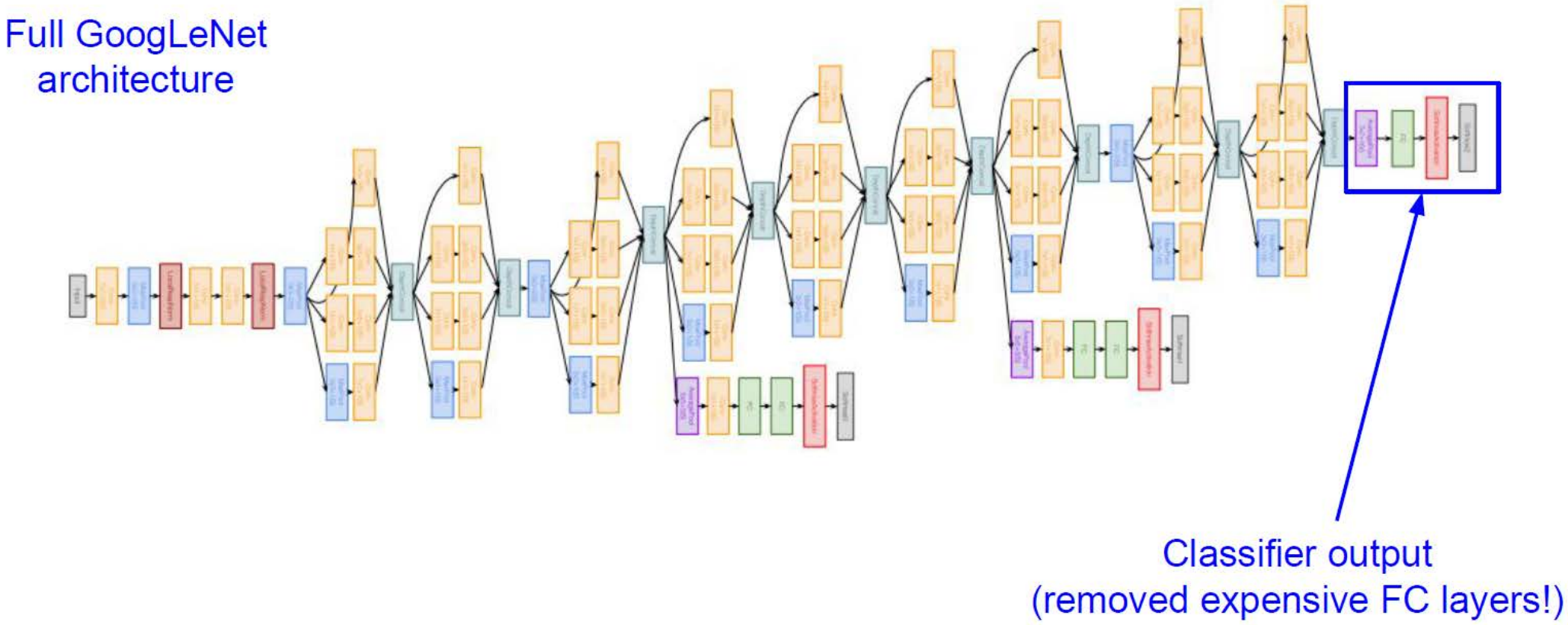
GoogLeNet Stacked Inception Modules

- 9 inception modules



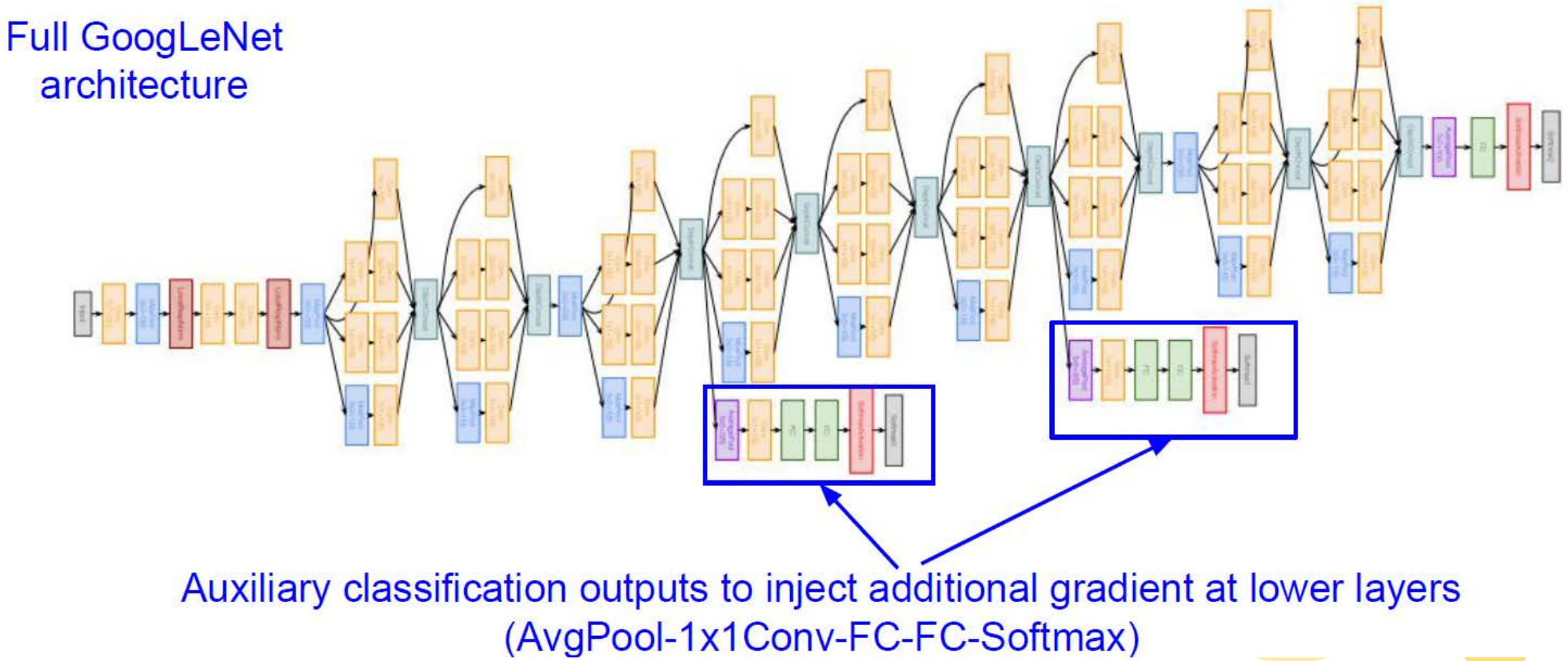
GoogLeNet Classifier Output

- Global Average Pooling at the last inception module



GoogLeNet Auxiliary Classification Outputs

- add intermediate classifiers to solve vanishing gradient problems
  - removed after training



Inception v2~v5

- Inception v2 (2015)
  - batch normalization
- Inception v3 (2016)
  - filter factorization
- Inception v4 (2017)
  - ResNet + Inception
- v5: Xception (2017)
  - depthwise separable convolution

(Inception v1): C. Szegedy, et al., “Going Deeper with Convolutions,” CVPR, 2015.
(Inception v2): S. Ioffe and C. Szegedy, “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift,” ICML 2015.
(Inception v3): C. Szegedy et al., “Rethinking the Inception Architecture for Computer Visiion,” CVPR, 2016.
(Inception v4): C. Szegedy, et al, “Inception-v4, Inception-ResNet and the Impact of Residual Connections in Learning,” AAAI, 2017.
(Inception v5): F. Chollet, “Xception: Deep Learning with Datawise Separable Convolutions,” CVPR, 2017.

Inception v2 (Batch Normalization)

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β

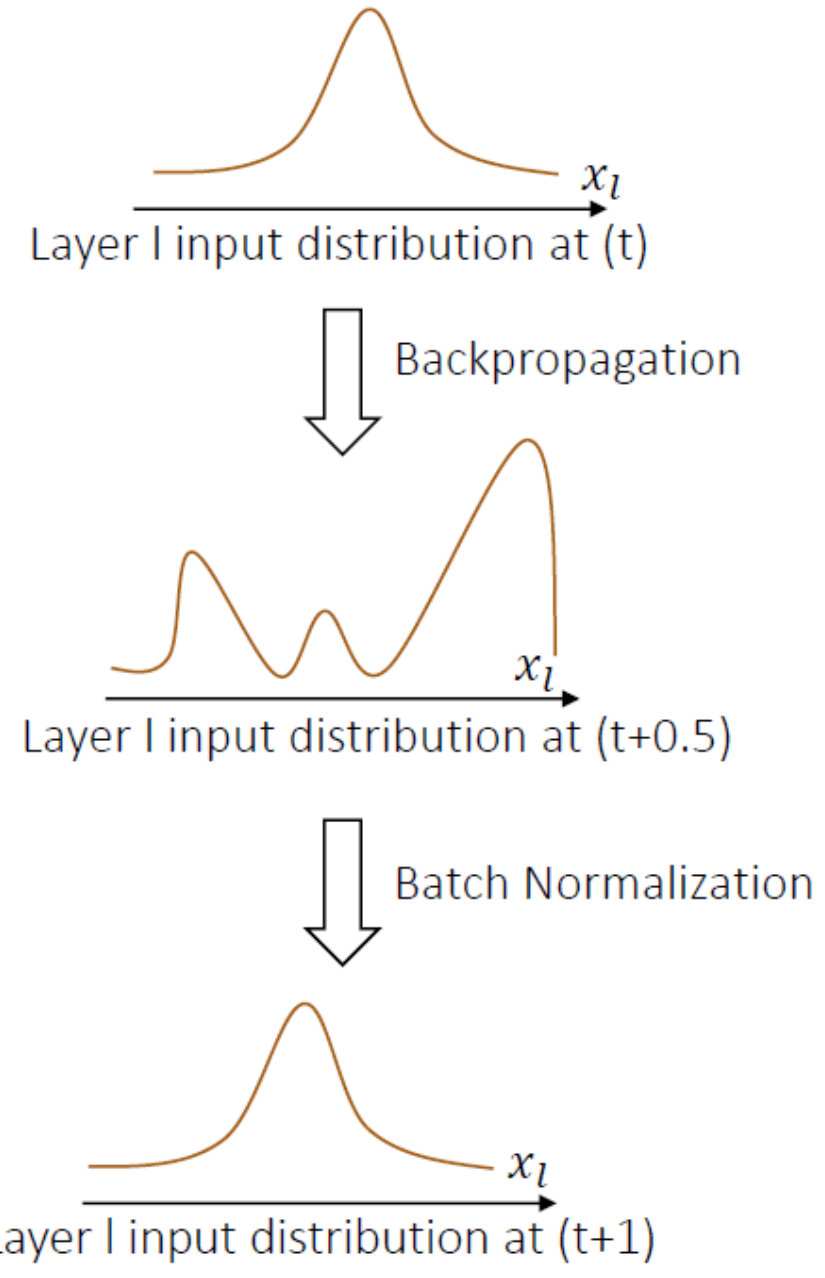
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

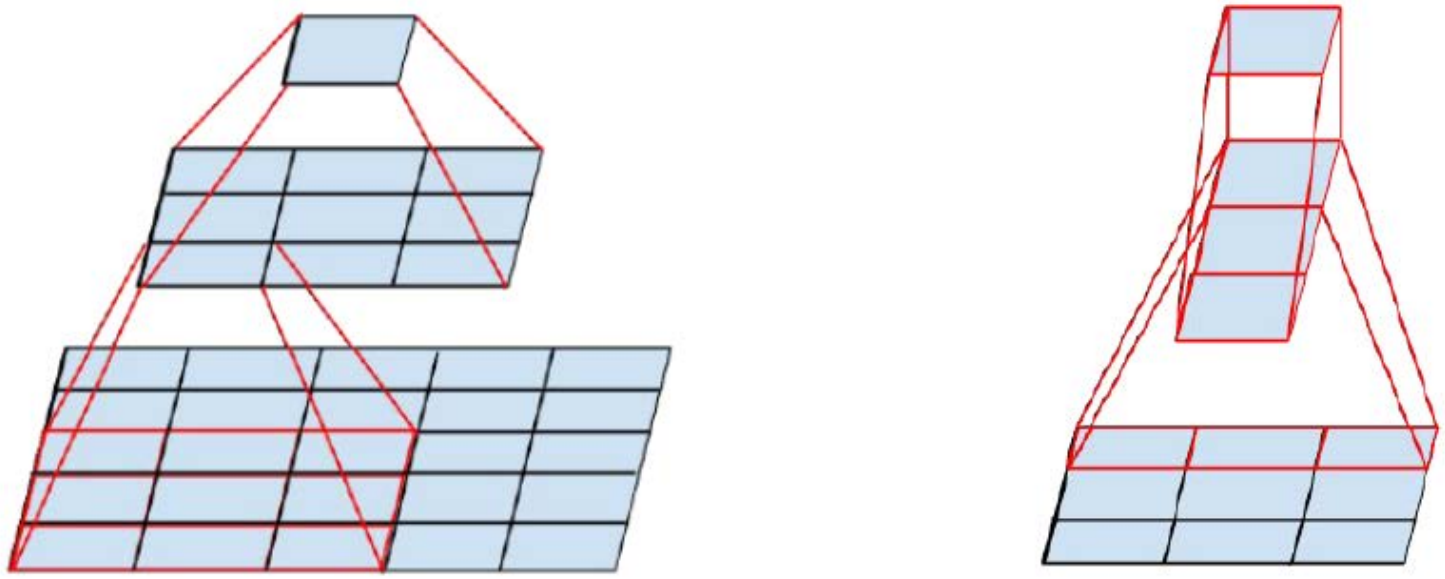
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$



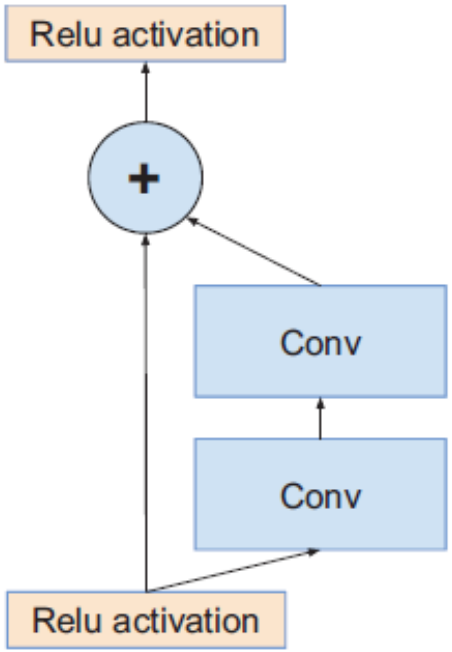
Inception v3 (factorization)

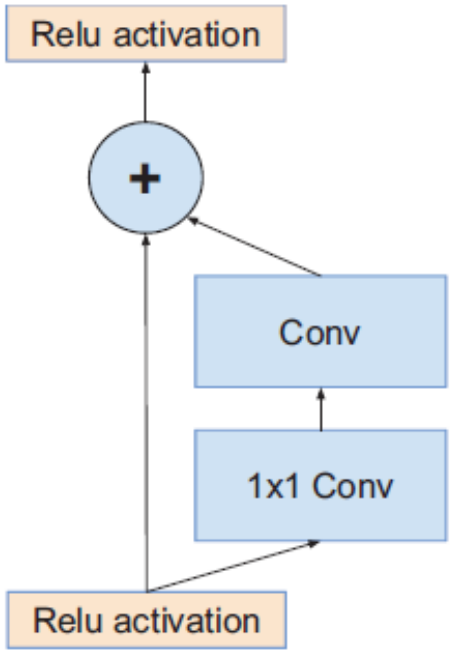
- factorize a large filter (e.g., 5x5) into several smaller filters (e.g., two 3x3)
- factorize a 3x3 filter into 3x1 and then 1x3
-

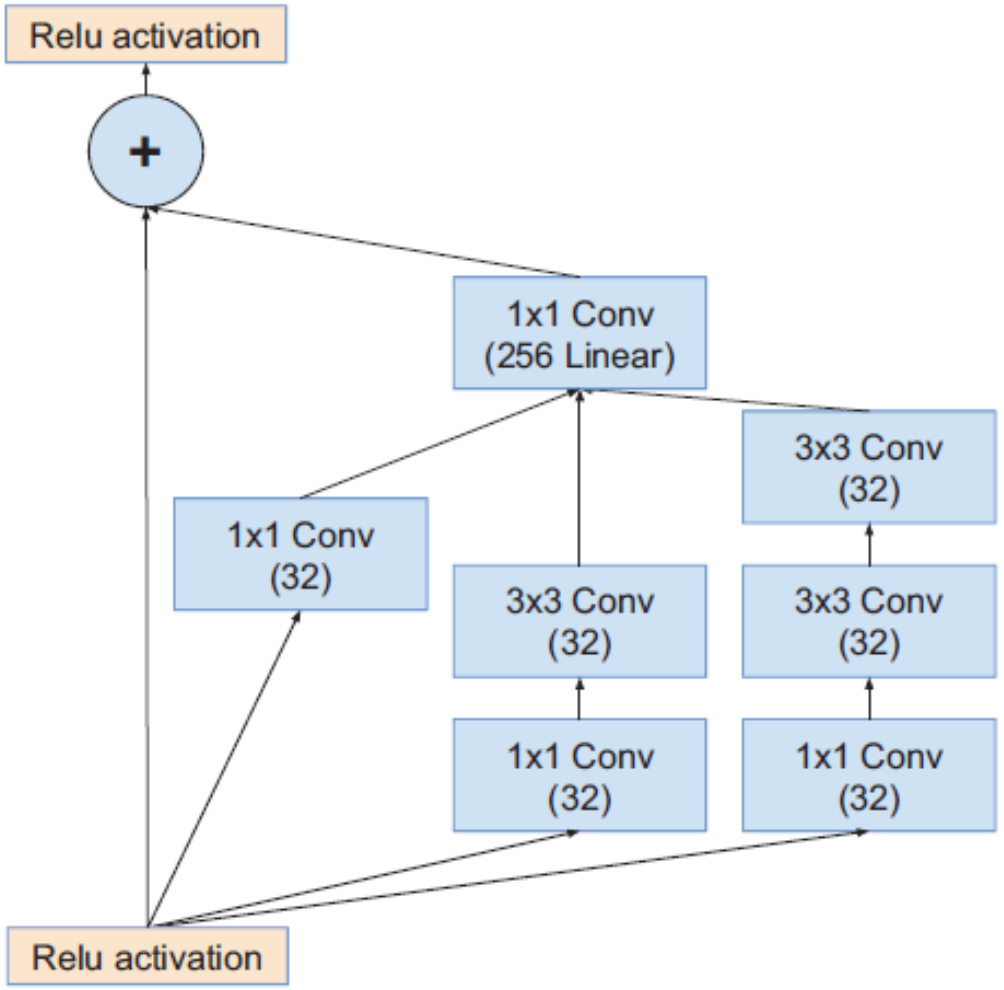


Inception v4

- Inception + ResNet

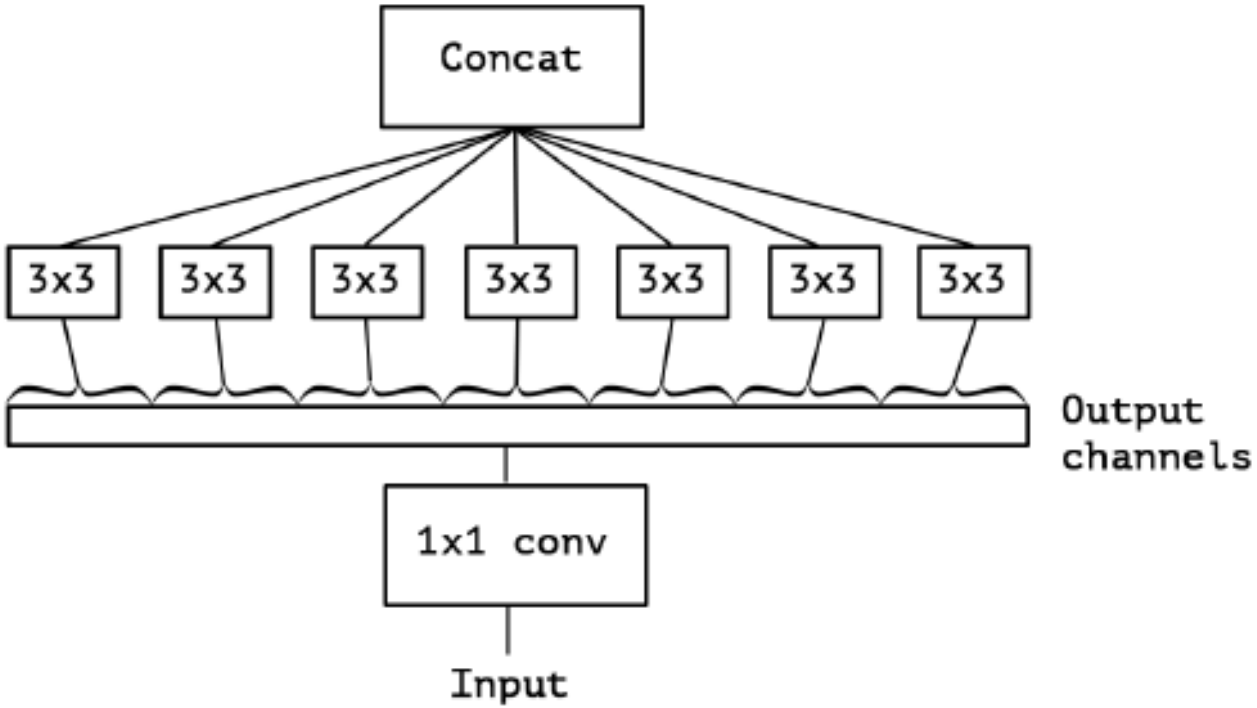






Xception (Inception v5)

- 1x1 convolution + depthwise convolution
  - cp. depthwise convolution + 1x1 convolution



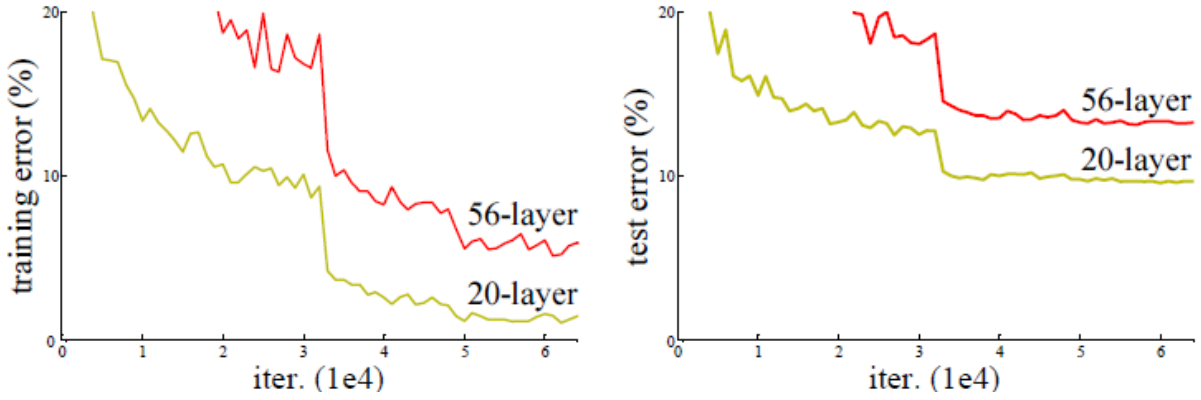
ResNet (1/2)

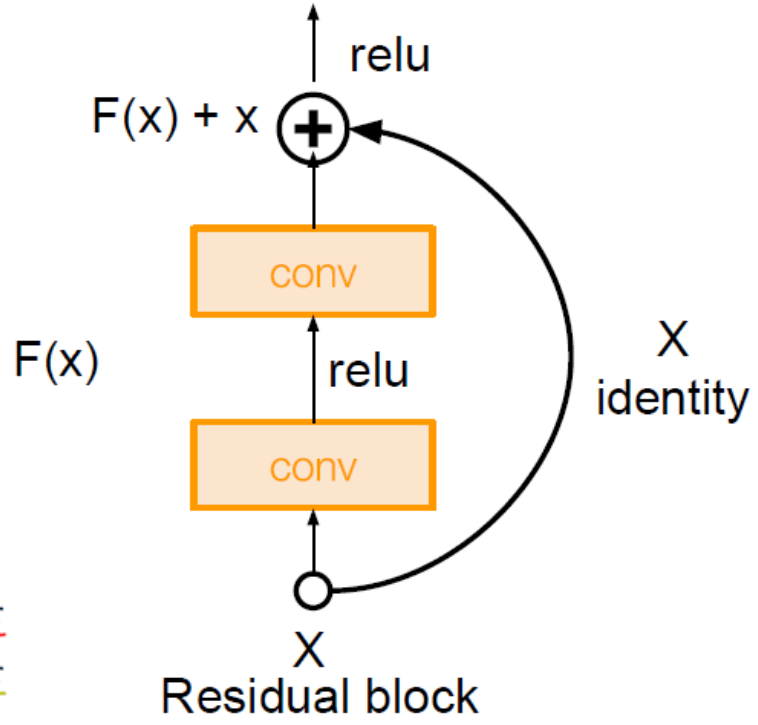
[He et al., 2015]

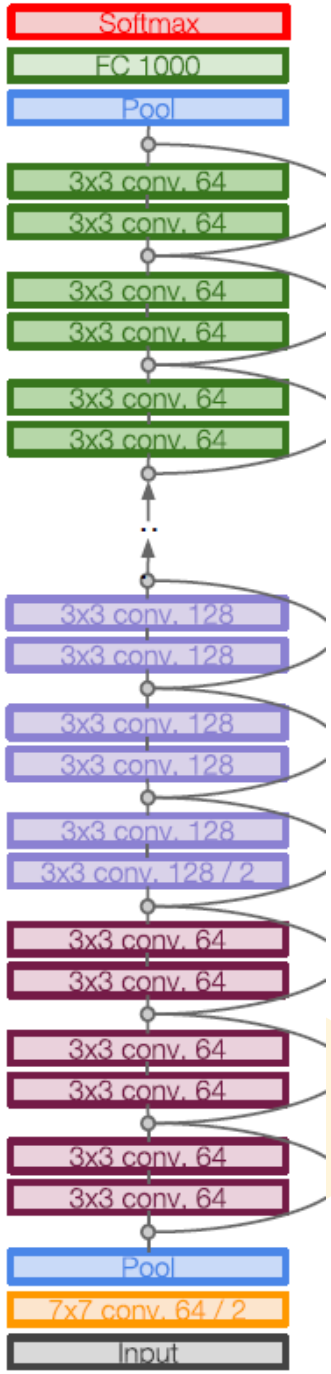
- avoid gradient vanishing problems

Very deep networks using residual connections

- 152-layer model for ImageNet
- ILSVRC'15 classification winner (3.57% top 5 error)
- win over all classification and detection competitions in ILSVRC'15 and COCO'15!



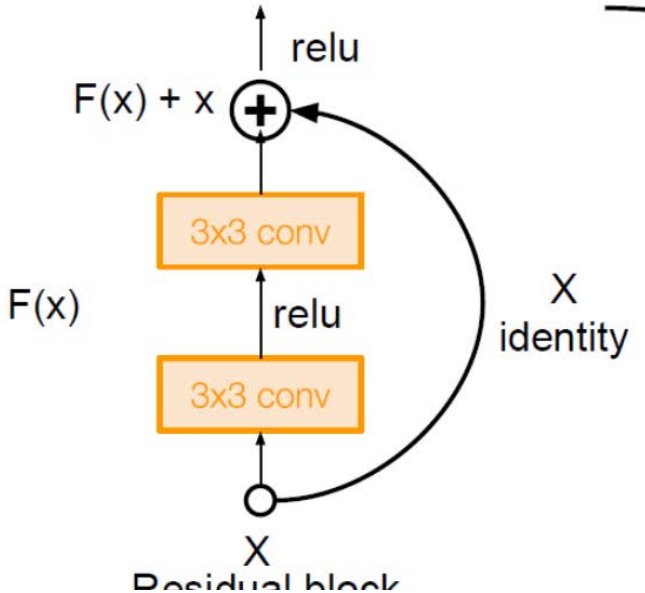


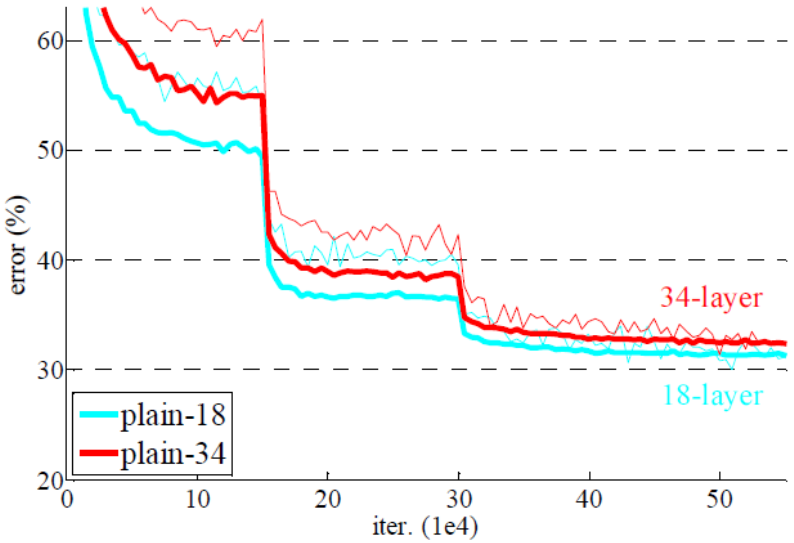


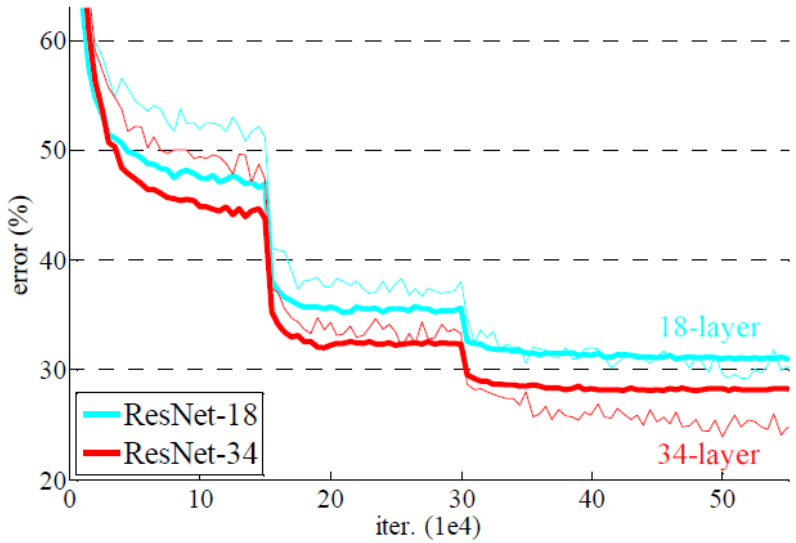
* K. He, et al., "Deep Residual Learning for Image Recognition," CVPR< 2016. (Microsoft)

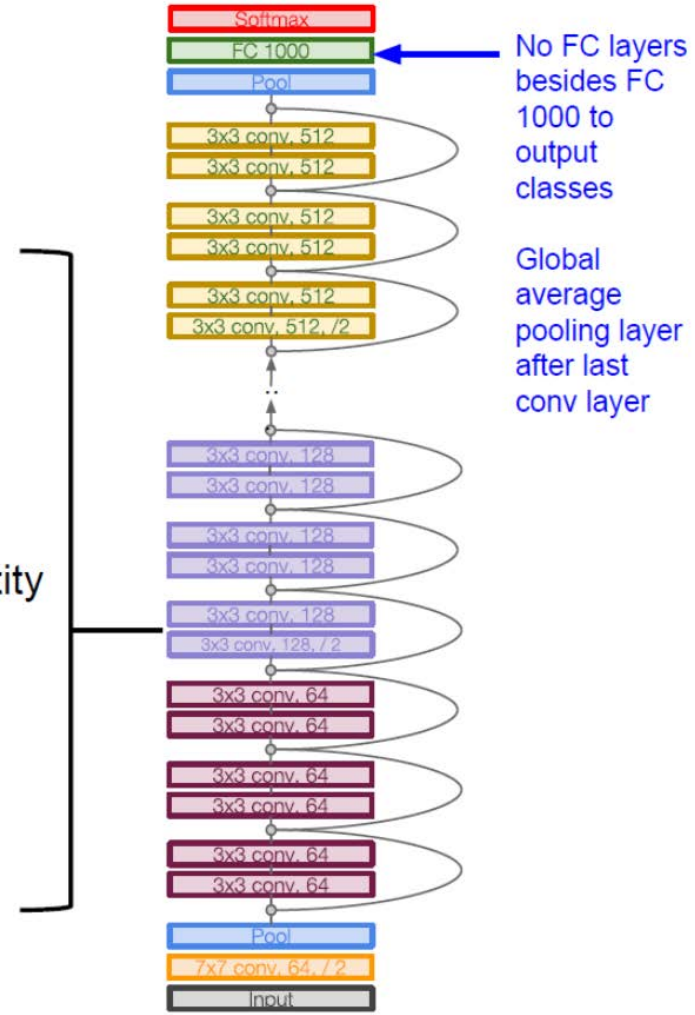
ResNet (2/2)

- 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
- No FC layers at the end (only FC 1000 to output classes)





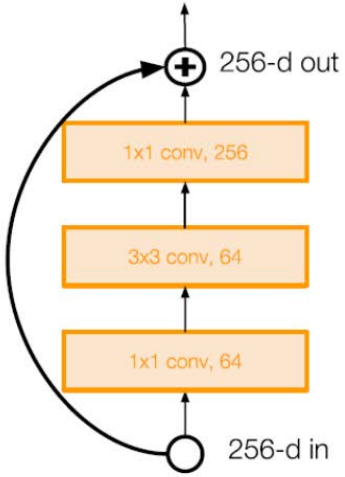


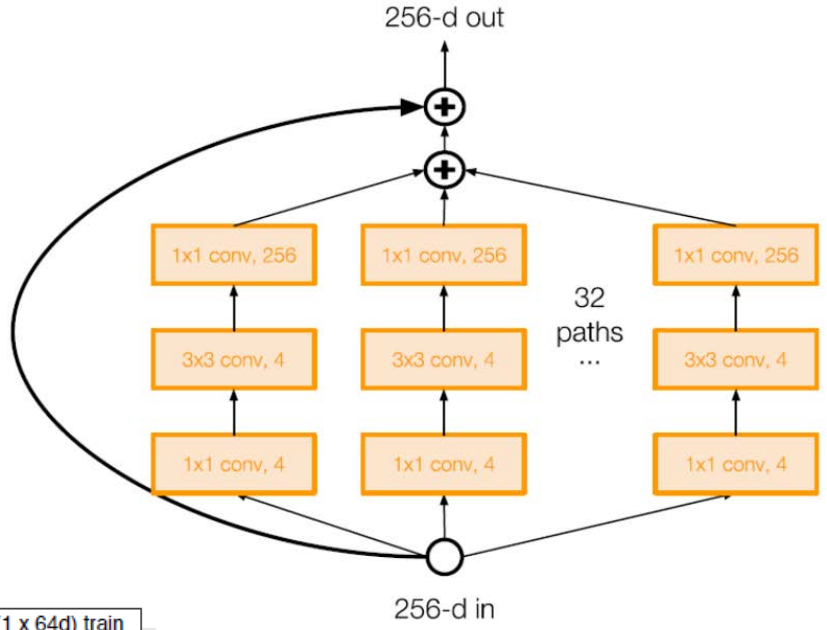


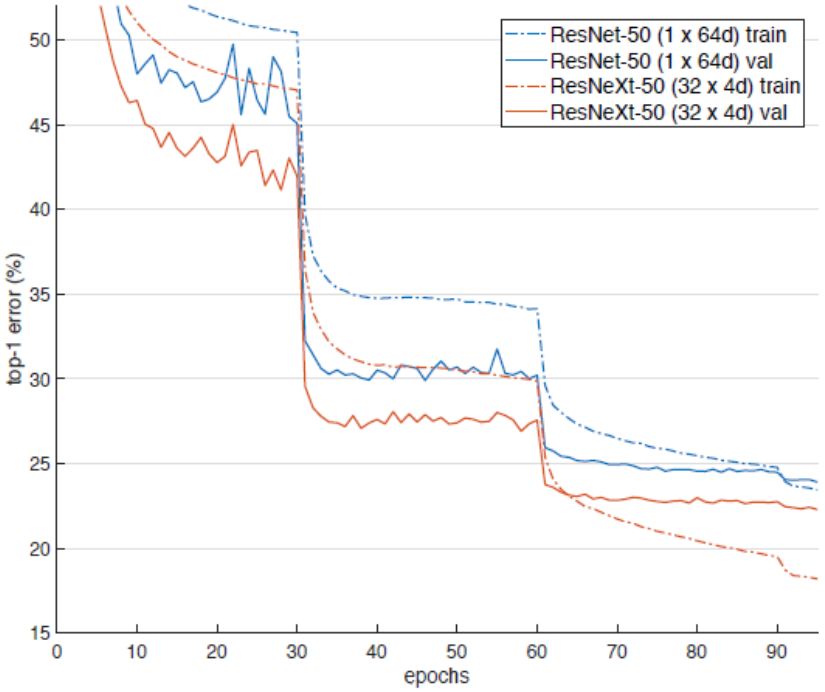
ResNeXt*

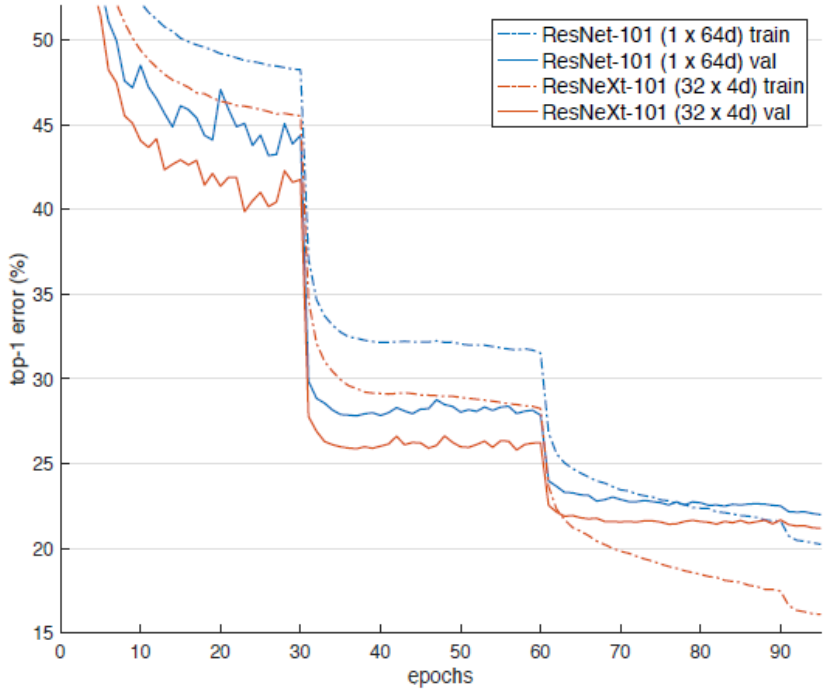
Aggregated Residual Transformations for Deep Neural Networks (ResNeXt)

- Also from creators of ResNet
- Increases width of residual block through multiple parallel pathways
- Parallel pathways similar in spirit to Inception module



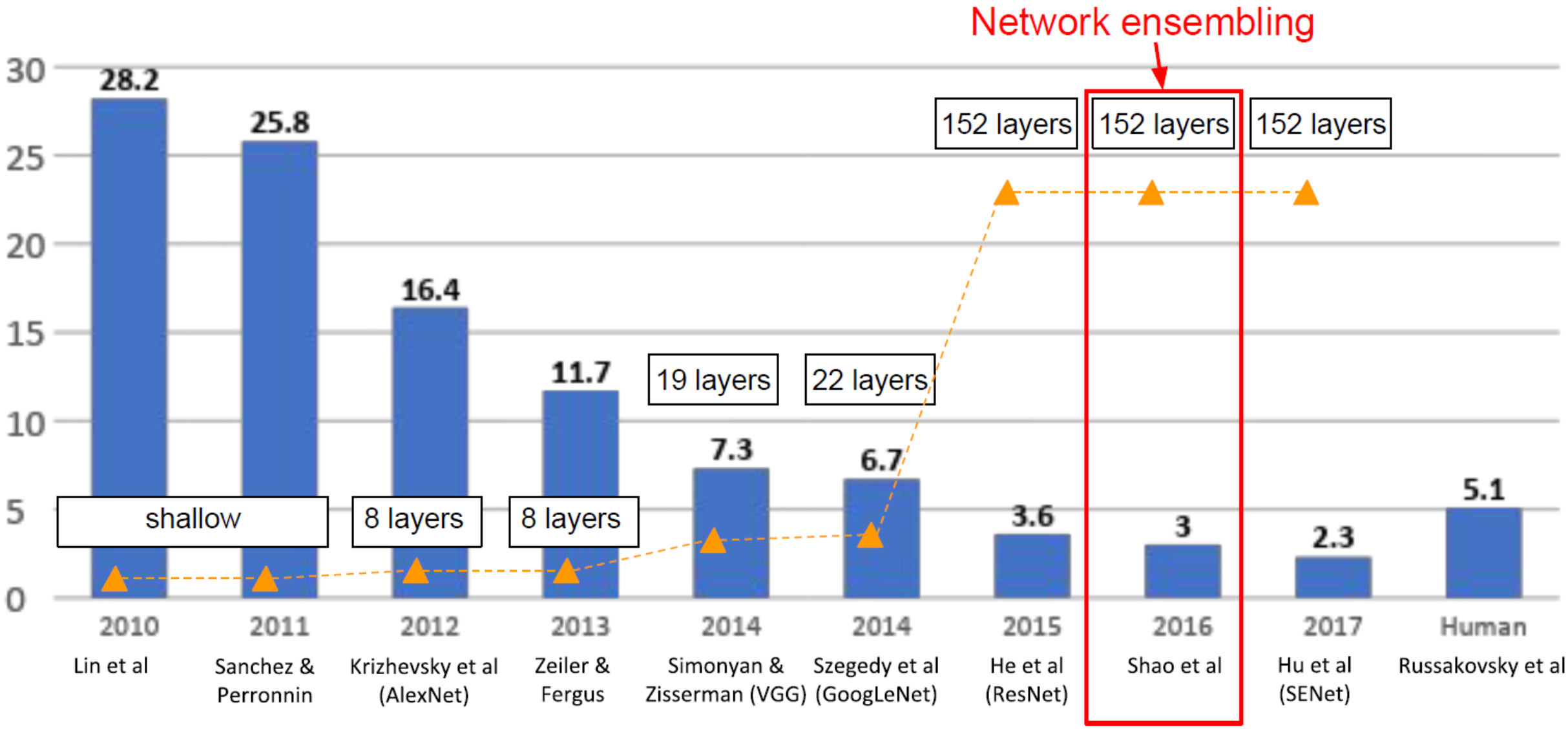






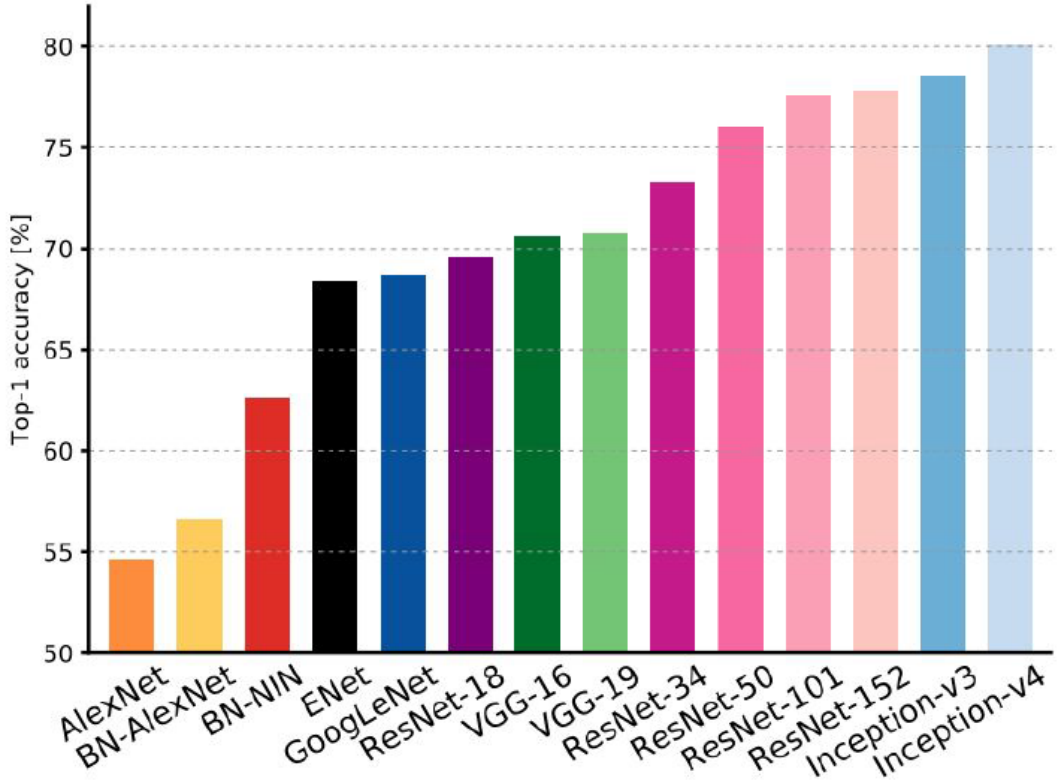
* S. Xie, et al., “Aggregated Residual Transformations for Deep Nerual Networks,” CVPR, 2017. (Facebook)

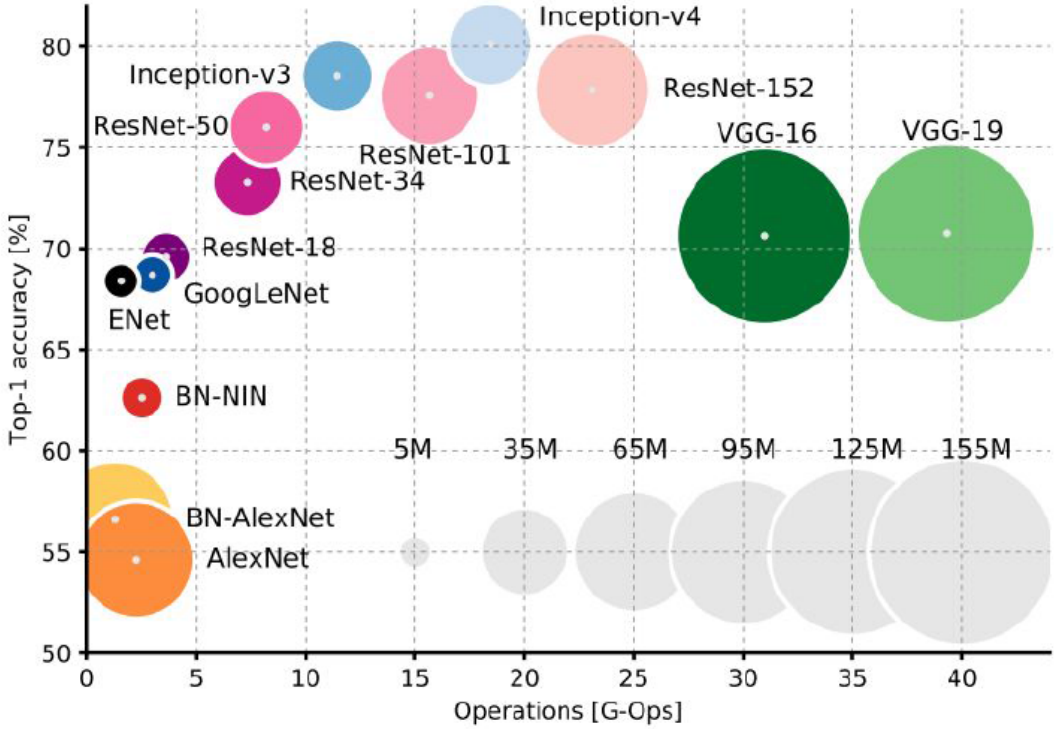
ImageNet Large Scale Visual Recognition Challenge (ILSVRC) winners



Complexity Comparison

- VGG: highest memory and most operations
- GoogLeNet: most efficient
- AlexNet has smaller computation complexity, but still memory heavy, and lower accuracy
- ResNet: moderate efficiency depending on model depth, highest accuracy





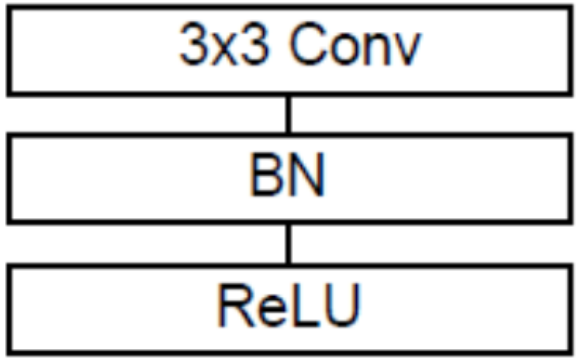
An Analysis of Deep Neural Network Models for Practical Applications, 2017.

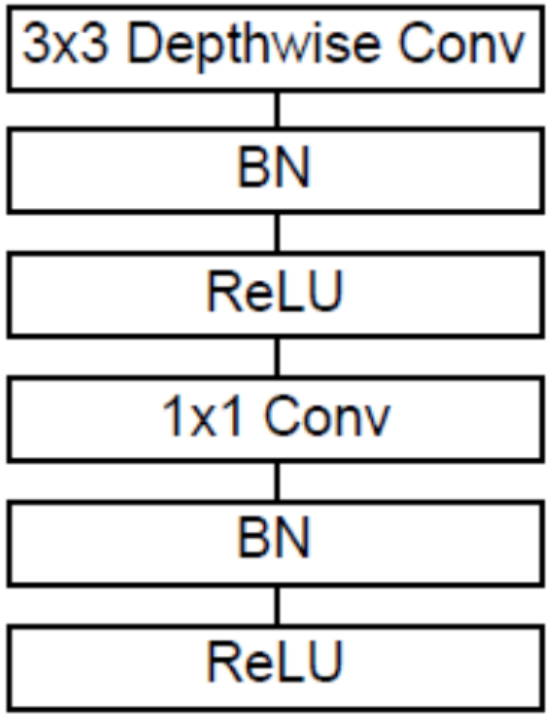
Summary

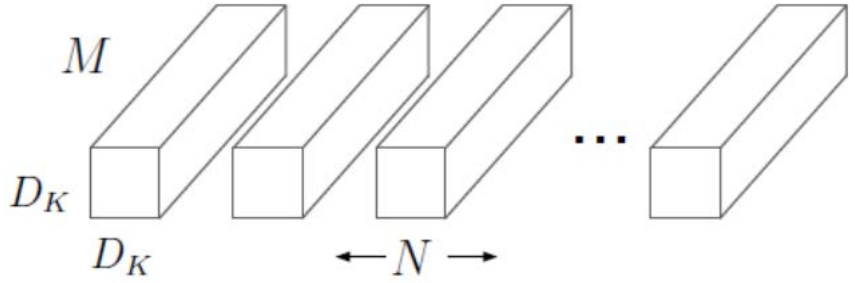
CNN		layers		Kernel size	# of parameters (M)		GOP		Top-1 val. error (%)	Top-5 val. error (%)
		Conv	FC		Conv	FC	Conv	FC		
AlexNet (2016)		5	3	3x3 5x5 11x11	60		0.67	0.05	37.5	17.0
VGG (2015)	11	8	3	3x3	133		6.5	0.12	29.6	10.4
	13	10	3	3x3	133		10.2	0.12	28.7	9.9
	16	13	3	1x1 3x3	138		15.3	0.12	27.0	8.8
	19	16	3	3x3	144		16.8	0.12	27.3	9.0
GoogLeNet (2015)		22		1x1 3x3 5x5 7x7	5.8	1	1.5	0.001	-	9.15
ResNet (2016)	20	19	1	1x1 3x3 7x7	0.27		1.8		-	8.75
	32	31	1		0.46		3.6		-	7.51
	56	55	1		0.85		3.86		-	6.97
	110	109	1		1.7		7.6		-	6.43
	1202	1201	1		19.4		11.3		-	7.93
ResNeXt (2016)	50	49	1	1x1 3x3 7x7	25		4.2		24.4	6.6
	101	100	1		44.3		7.99		22.2	5.7
PyramidNet (2017)	110 α=48	109	0	1x1 3x3	1.7		-		-	4.58
	110 α=84	109	0		3.8		-		-	4.26
	110 α=270	109	0		28.3		-		-	3.73
	164 α=270 bottle neck	163	0		27.0		-		-	3.48

Light CNN Models: MobileNet*

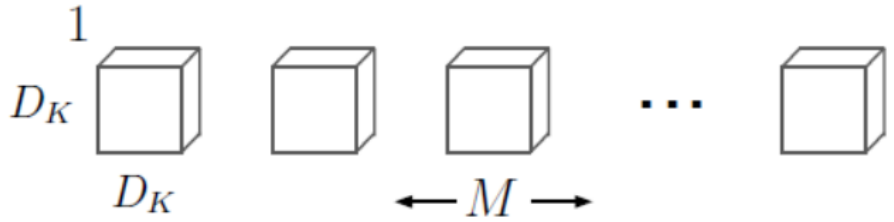
- standard 3D convolution = depthwise convolution + 1x1 convolution



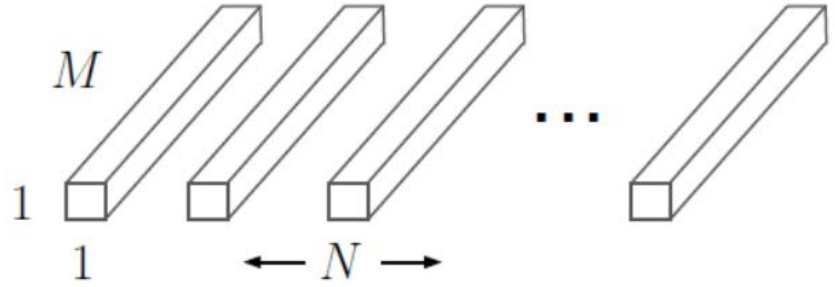




(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



* A. G. Howard, et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” 2017. (Google)

MobileNet

- width multiplier
  - reduced channel number
- resolution multiplier
  - reduced channel size

Table 8. MobileNet Comparison to Popular Models

Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
1.0MobileNet-224	70.6%	569	4.2
GoogleNet	69.8%	1550	6.8
VGG 16	71.5%	15300	138

Table 10. MobileNet for Standford Dogs

Model	Top-1 Accuracy	Million Mult-Adds	Million Parameters
Inception V3	84%	5000	23.2
1.0MobileNet-224	83.3%	569	3.3
0.75MobileNet-224	81.9%	325	1.9
1.0MobileNet-192	81.9%	418	3.3
0.75MobileNet-192	80.5%	239	1.9

Type / Stride	Filter Shape	Input Size
Conv / s2	3 x 3 x 3 x 32	224 x 224 x 3
Conv dw / s1	3 x 3 x 32 dw	112 x 112 x 32
Conv / s1	1 x 1 x 32 x 64	112 x 112 x 32
Conv dw / s2	3 x 3 x 64 dw	112 x 112 x 64
Conv / s1	1 x 1 x 64 x 128	56 x 56 x 64
Conv dw / s1	3 x 3 x 128 dw	56 x 56 x 128
Conv / s1	1 x 1 x 128 x 128	56 x56 x128
Conv dw / s2	3 x 3 x 128 dw	56 x 56 x 128
Conv / s1	1 x 1 x 128 x 256	28 x28 x128
Conv dw / s1	3 x 3 x 256 dw	28 x 28 x 256
Conv / s1	1 x 1 x 256 x 256	28 x 28 x 256
Conv dw / s2	3 x 3 x 256 dw	28 x 28 x 256
Conv / s1	1 x 1 x 256 x 512	14 x 14 x 256
5x **Conv dw / s 1** **Conv / s1**	3 x 3 x 512 dw 1 x 1 x 512 x 512	14 x 14 x 512 14x14x512
Conv dw / s2	3 x 3 x 512 dw	14 x 14 x 512
Conv / s1	1 x 1 x 512 x 1024	7 x 7 x 512
Conv dw / s2	3 x 3 x 1024 dw	7 x 7 x 1024
Conv/ s1	lx 1 x 1024 x1024	7x 7 x 1024
Avg Pool / s1	Pool 7 x 7	7 x 7 x 1024
FC/s1	1024 x 1000	1 x 1 x1024
Softmax / s1	Classifier	1 x 1 x 1000

MobileNet Result

- object detection using MobileNet SSD (Single Shot Detection)
  - COCO dataset

Framework Resolution	Model	mAP	Billion Mult-Adds	Million Parameters
SSD 300	deeplab-VGG	21.1%	34.9	33.1
	Inception V2	22.0%	3.8	13.7
	MobileNet	19.3%	1.2	6.8
Faster-RCNN 300	VGG	22.9%	64.3	138.5
	Inception V2	15.4%	118.2	13.3
	MobileNet	16.4%	25.2	6.1
Faster-RCNN 600	VGG	25.7%	149.6	138.5
	Inception V2	21.9%	129.6	13.3
	Mobilenet	19.8%	30.5	6.1