Popular DNNs and Inference

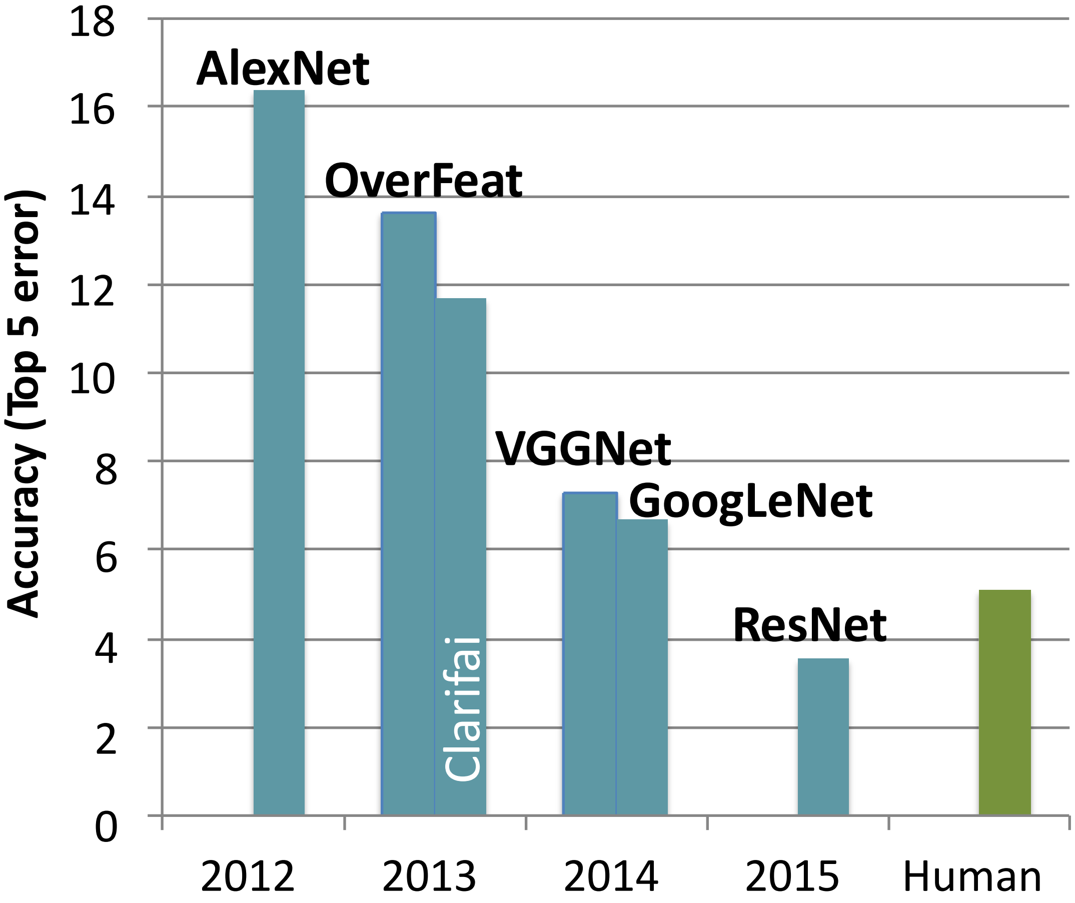
Kaisheng Ma

Ref: http://eyeriss.mit.edu/tutorial.html

Popular DNNs

- **LeNet (1998)**
- **AlexNet (2012)**
- **OverFeat (2013)**
- **VGGNet (2014)**
- **GoogleNet (2014)**
- **ResNet (2015)**

ImageNet: Large Scale Visual Recognition Challenge (ILSVRC)



[O. Russakovsky et al., IJCV 2015]

MNIST

Digit Classification
28x28 pixels (B&W)
10 Classes
60,000 Training
10,000 Testing

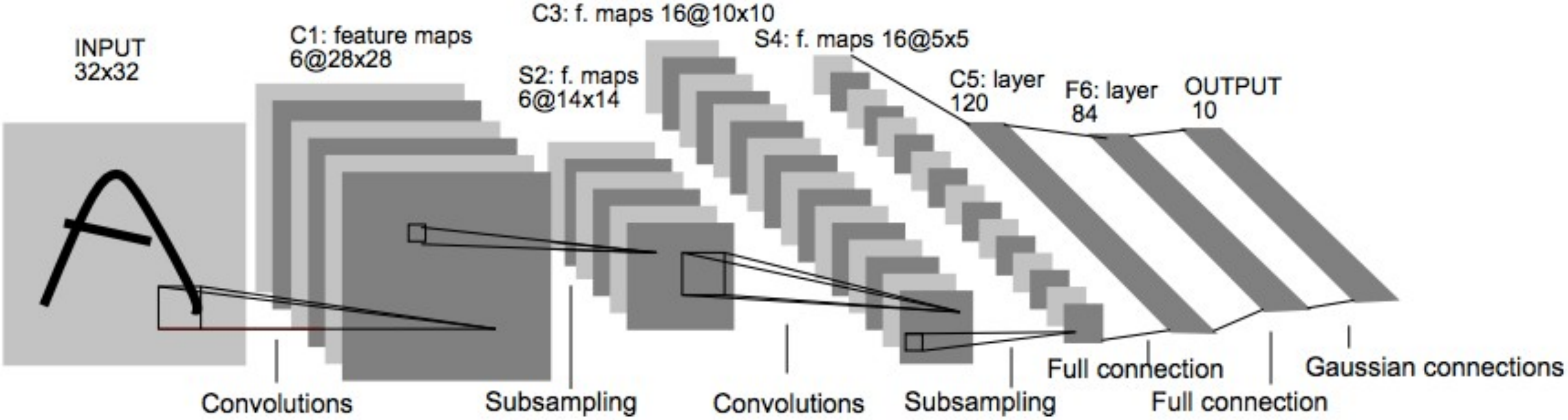
http://yann.lecun.com/exdb/mnist/

LeNet-5

CONV Layers: 2
Fully Connected Layers: 2
Weights: 60k
MACs: 341k

Sigmoid used for non-linearity

Digit Classification!
(MNIST Dataset)



six 5x5 filters

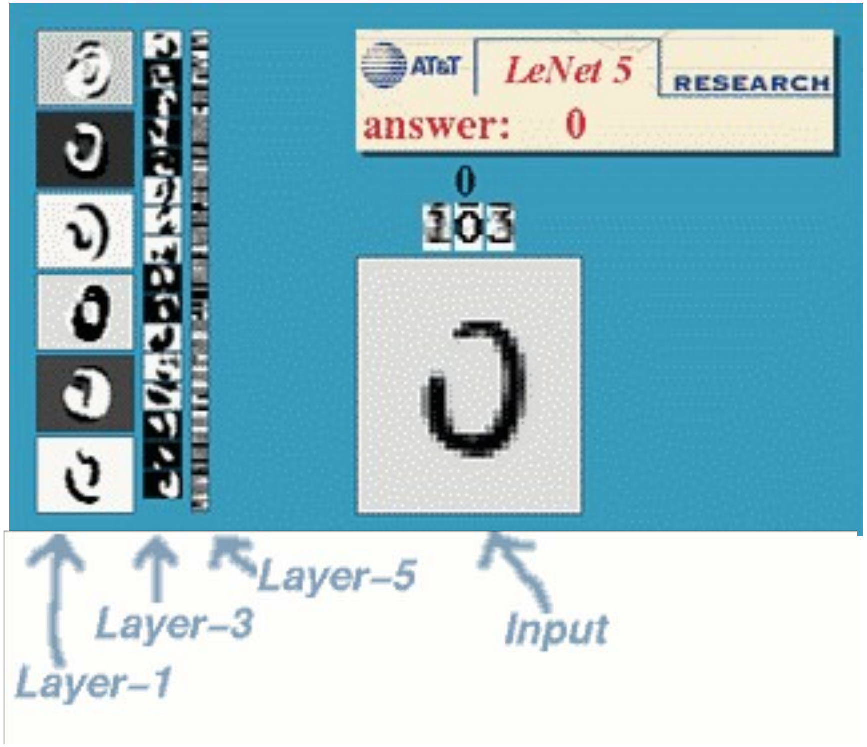
2x2 average pooling

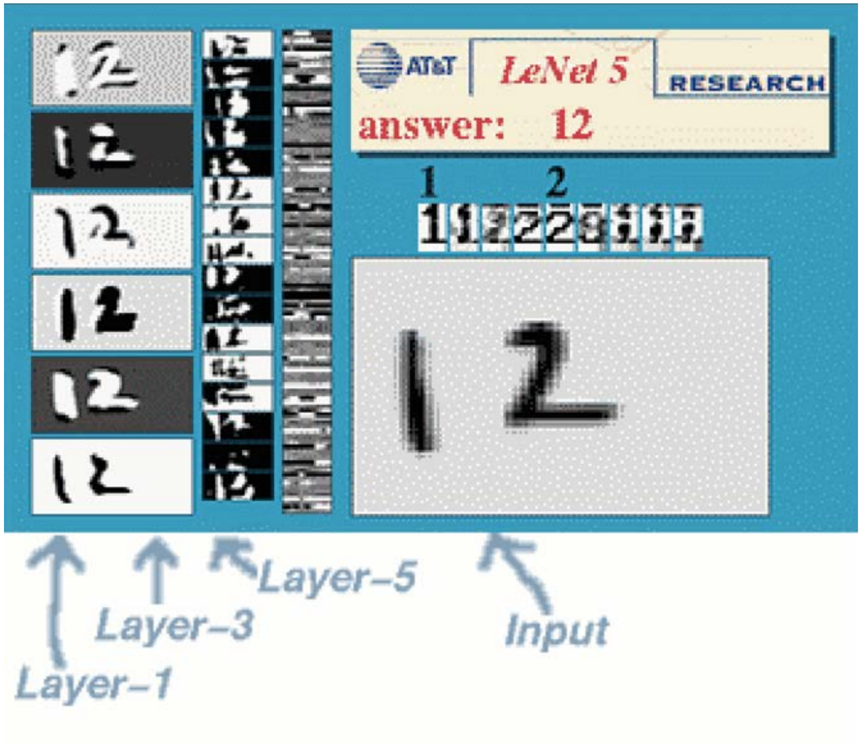
sixteen 5x5 filters

2x2 average pooling

[Lecun et al., Proceedings of the IEEE, 1998]

LeNet-5





http://yann.lecun.com/exdb/lenet/

IMAGENET

Image Classification
~256x256 pixels (color)
1000 Classes
1.3M Training
100,000 Testing (50,000 Validation)

For **ImageNet Large Scale Visual Recognition Challenge (ILSVRC)** accuracy of classification task reported based on top-1 and top-5 error

Image Source: http://karpathy.github.io/

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

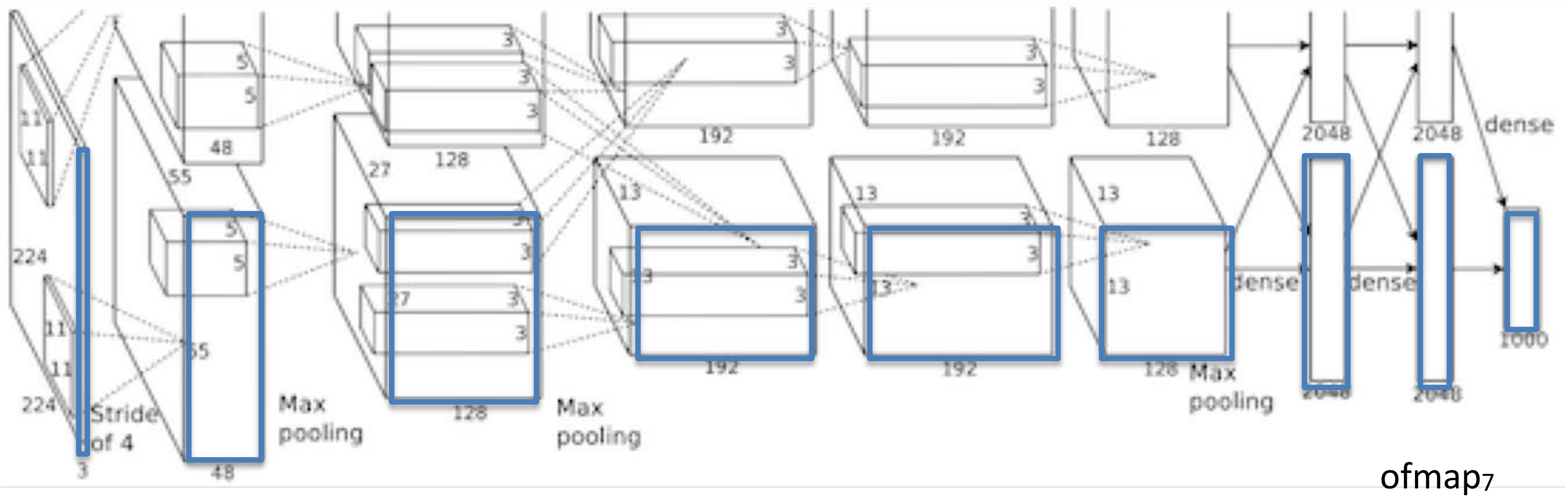
AlexNet

CONV Layers: 5
Fully Connected Layers: 3
Weights: 61M
MACs: 724M
ReLU used for non-linearity

ILSCVR12 Winner

Uses Local Response Normalization (LRN)

[Krizhevsky et al., NeurIPS 2012]



input ofmap$_1$ ofmap$_2$ ofmap$_3$ ofmap$_4$ ofmap$_5$ ofmap$_6$ ofmap$_7$ ofmap$_8$

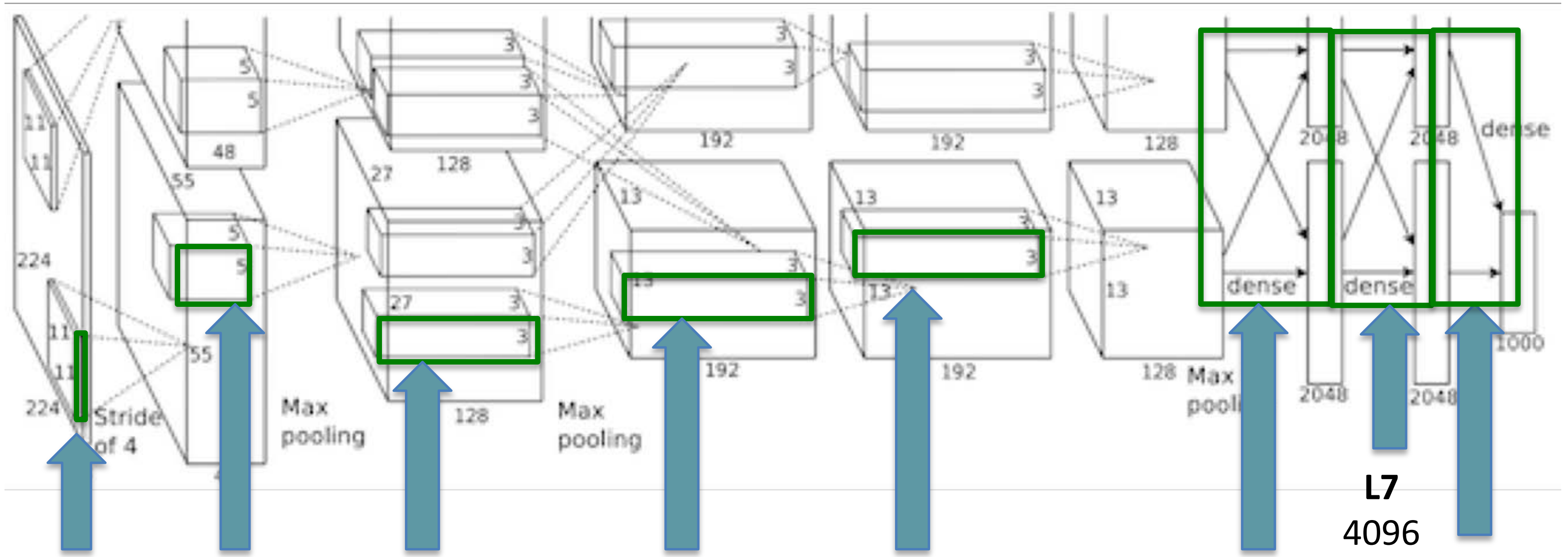
AlexNet

CONV Layers: 5
Fully Connected Layers: 3
Weights: 61M
MACs: 724M
ReLU used for non-linearity

ILSCVR12 Winner

Uses Local Response Normalization (LRN)

[Krizhevsky et al., NeurIPS 2012]



L1
96 filters
(11x11x3)

L2
256 filters
(5x5x48)

L3
384 filters
(3x3x256)

L4
384 filters
(3x3x192)

L5
384 filters
(3x3x192)

L6
4096 filters
(13x13x256)

L7
4096
(4096x1)

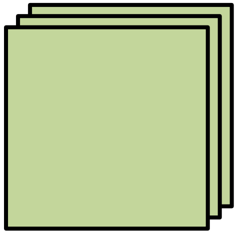
L8
1000
(4096x1)

Large Sizes with Varying Shapes

AlexNet Convolutional Layer Configurations

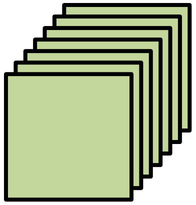
Layer	Filter Size (RxS)	# Filters (M)	# Channels (C)	Stride
1	11x11	96	3	4
2	5x5	256	48	1
3	3x3	384	256	1
4	3x3	384	192	1
5	3x3	256	192	1

Layer 1



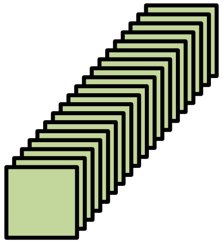
34k Params
105M MACs

Layer 2



307k Params
224M MACs

Layer 3



885k Params
150M MACs

[Krizhevsky et al., NIPS 2012]

AlexNet

CONV Layers: 5

Fully Connected Layers: 3

Weights: 61M

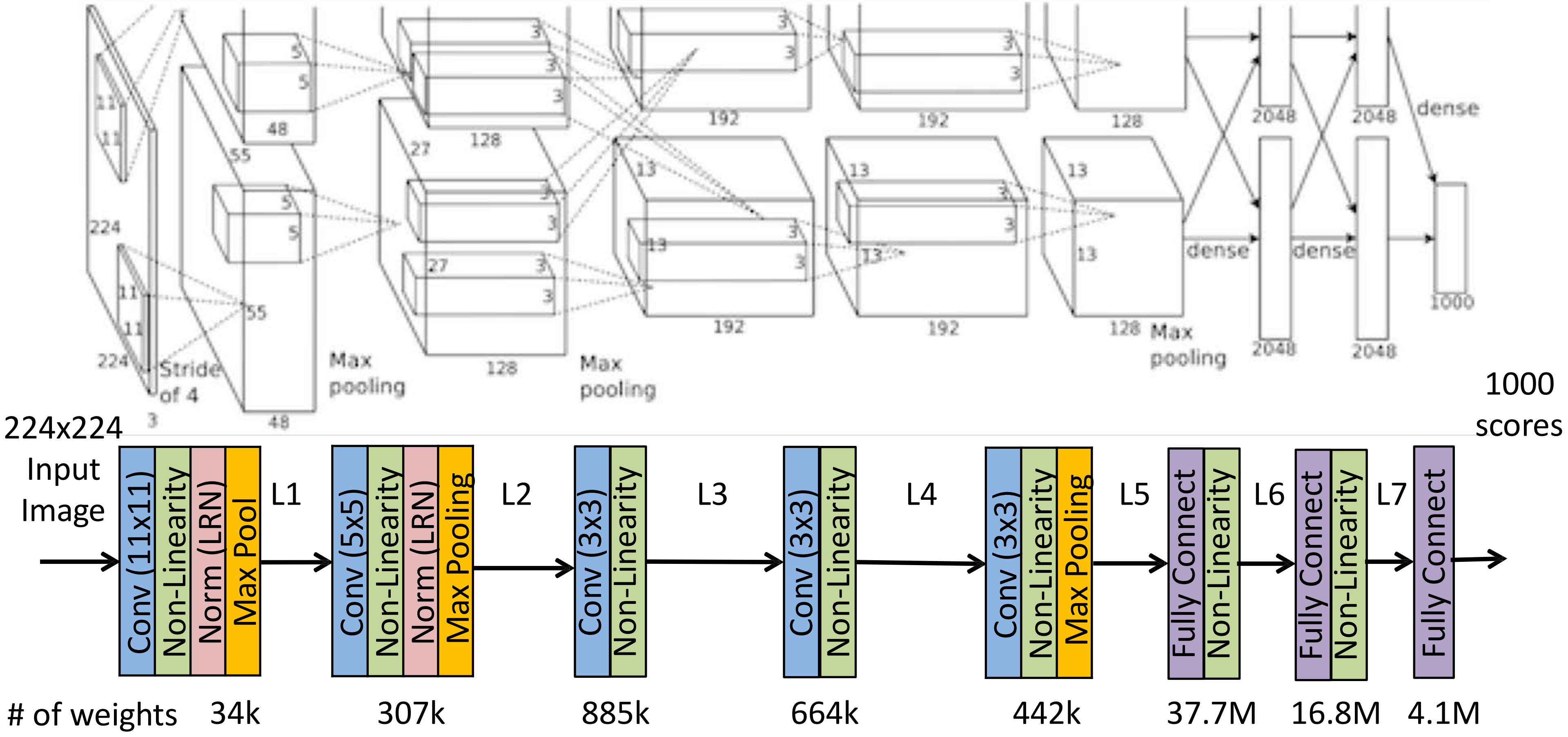
MACs: 724M

ReLU used for non-linearity

ILSCVR12 Winner

Uses Local Response Normalization (LRN)

[Krizhevsky et al., NeurIPS 2012]



224x224 Input Image

Conv (11x11) | Non-Linearity | Norm (LRN) | Max Pool

L1

Conv (5x5) | Non-Linearity | Norm (LRN) | Max Pooling

L2

Conv (3x3) | Non-Linearity

L3

Conv (3x3) | Non-Linearity

L4

Conv (3x3) | Non-Linearity | Max Pooling

L5

Fully Connect | Non-Linearity

L6

Fully Connect | Non-Linearity

L7

Fully Connect

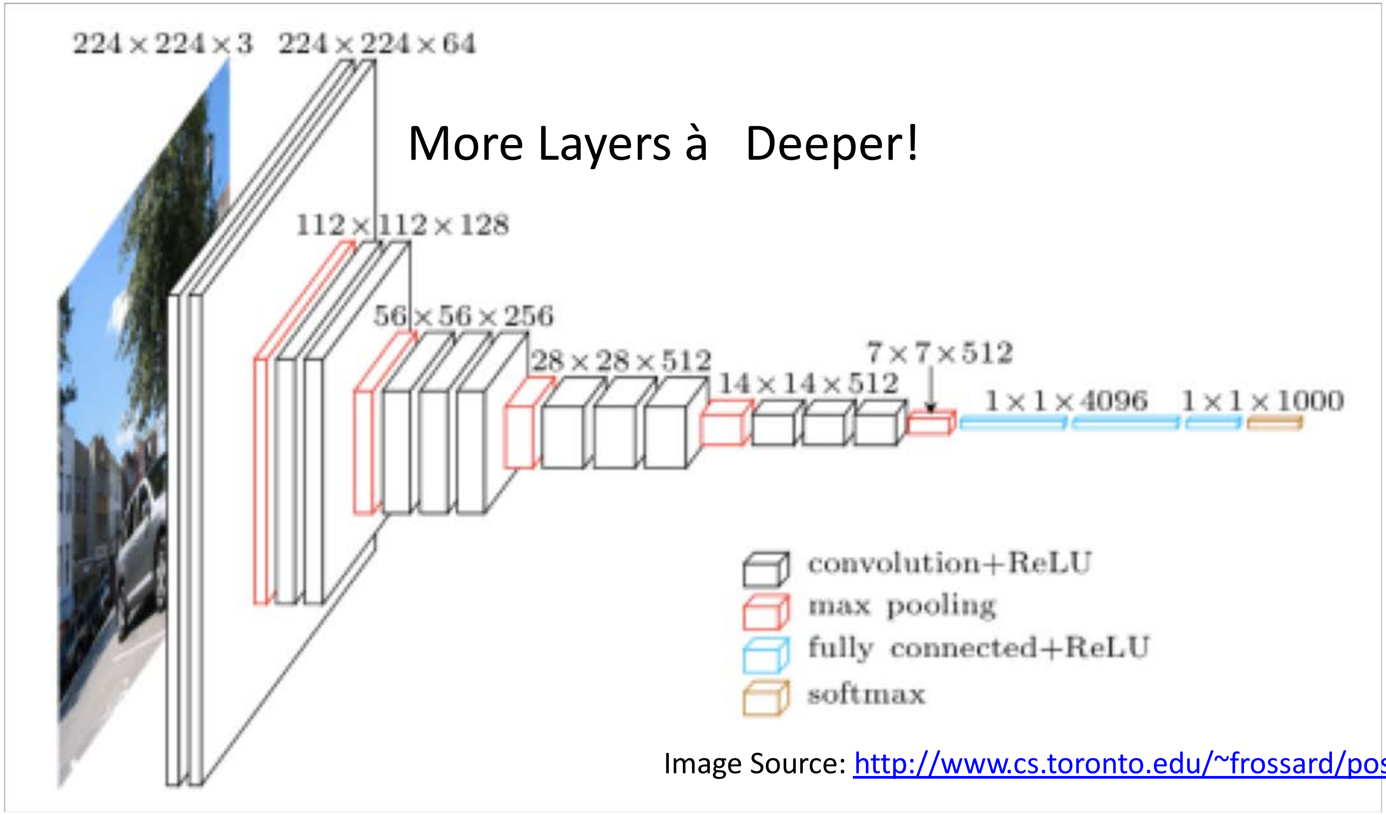
1000 scores

# of weights	34k	307k	885k	664k	442k	37.7M	16.8M	4.1M

VGG-16

CONV Layers: 13
Fully Connected Layers: 3
Weights: 138M
MACs: 15.5G

Also, 19 layer version



More Layers à Deeper!

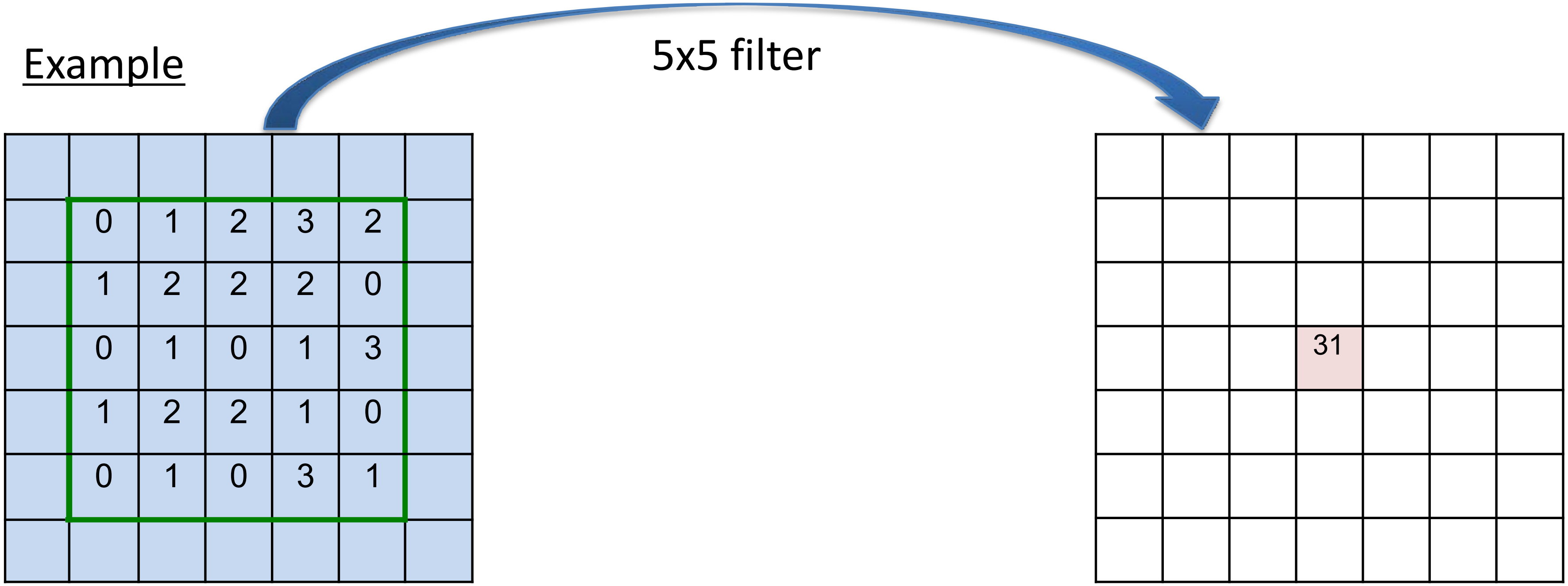
Image Source: http://www.cs.toronto.edu/~frossard/post/vgg16/

[Simonyan et al., arXiv 2014, ICLR 2015]

Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

Example



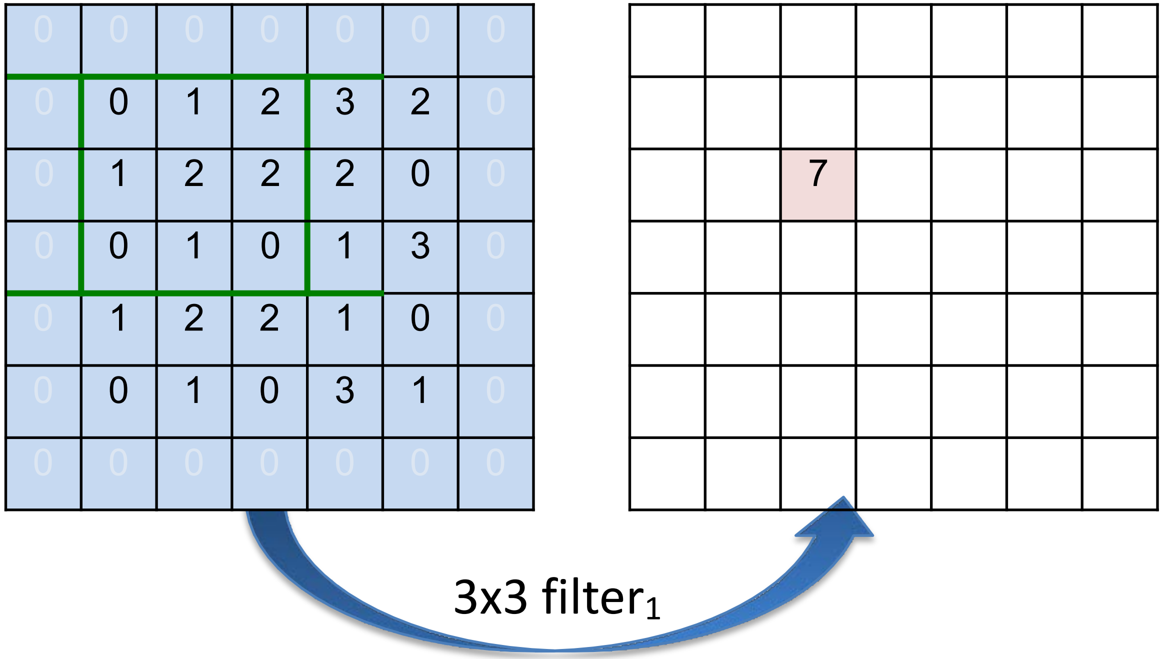
	0	1	2	3	2	
	1	2	2	2	0	
	0	1	0	1	3	
	1	2	2	1	0	
	0	1	0	3	1	

			31			

Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

Example



0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	2	2	2	0	0
0	0	1	0	1	3	0
0	1	2	2	1	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

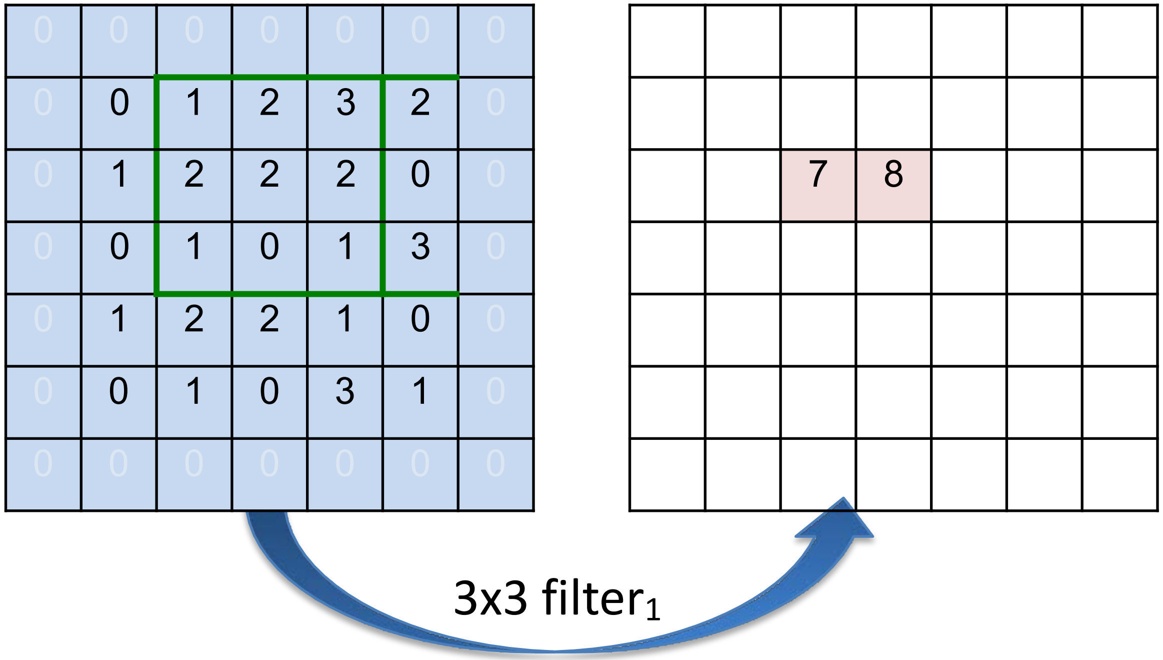
		7				

3x3 filter$_1$

Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

Example



0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	2	2	2	0	0
0	0	1	0	1	3	0
0	1	2	2	1	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

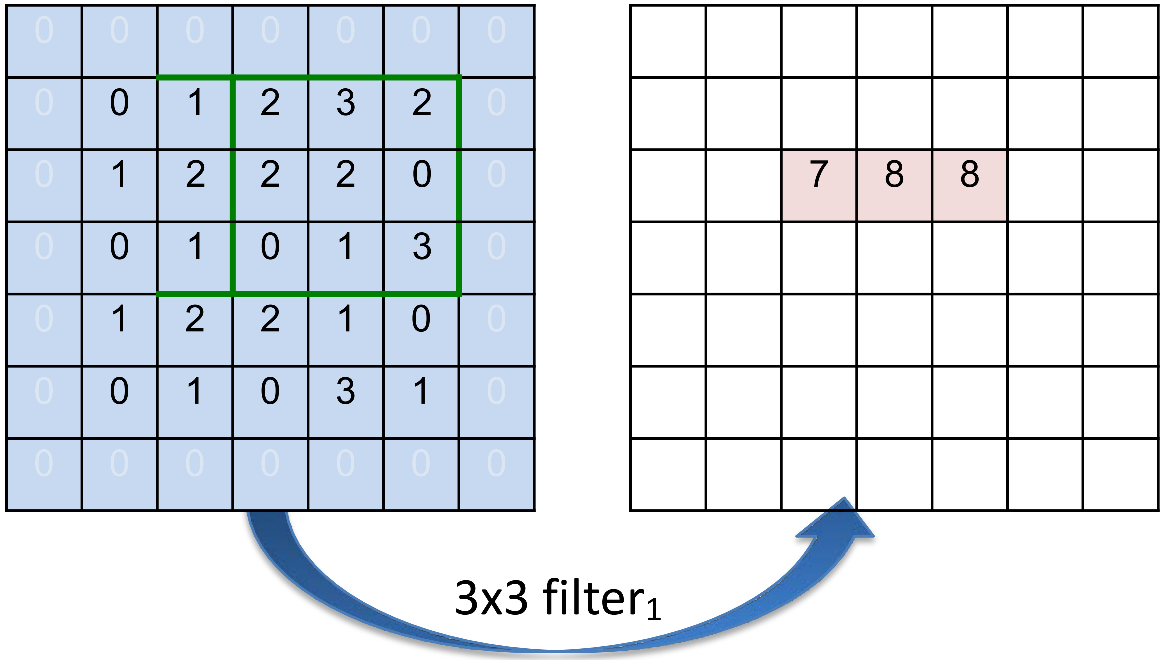
		7	8			

3x3 filter$_1$

Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

<u>Example</u>



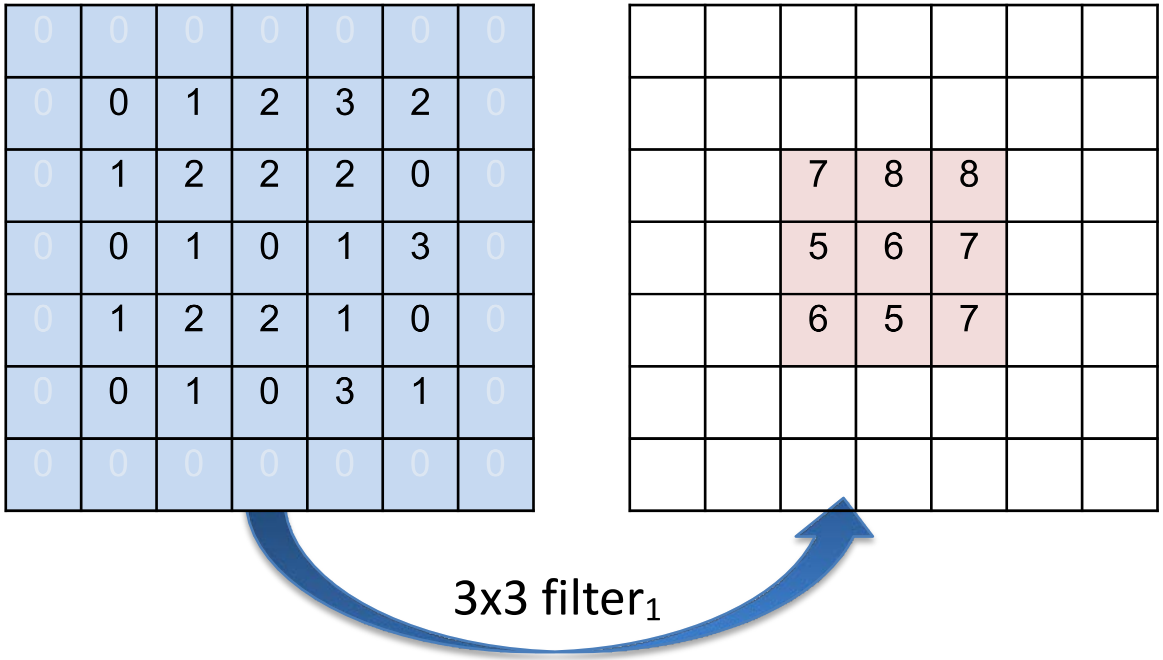
0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	2	2	2	0	0
0	0	1	0	1	3	0
0	1	2	2	1	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

		7	8	8		

Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

Example



0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	2	2	2	0	0
0	0	1	0	1	3	0
0	1	2	2	1	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

		7	8	8		
		5	6	7		
		6	5	7		

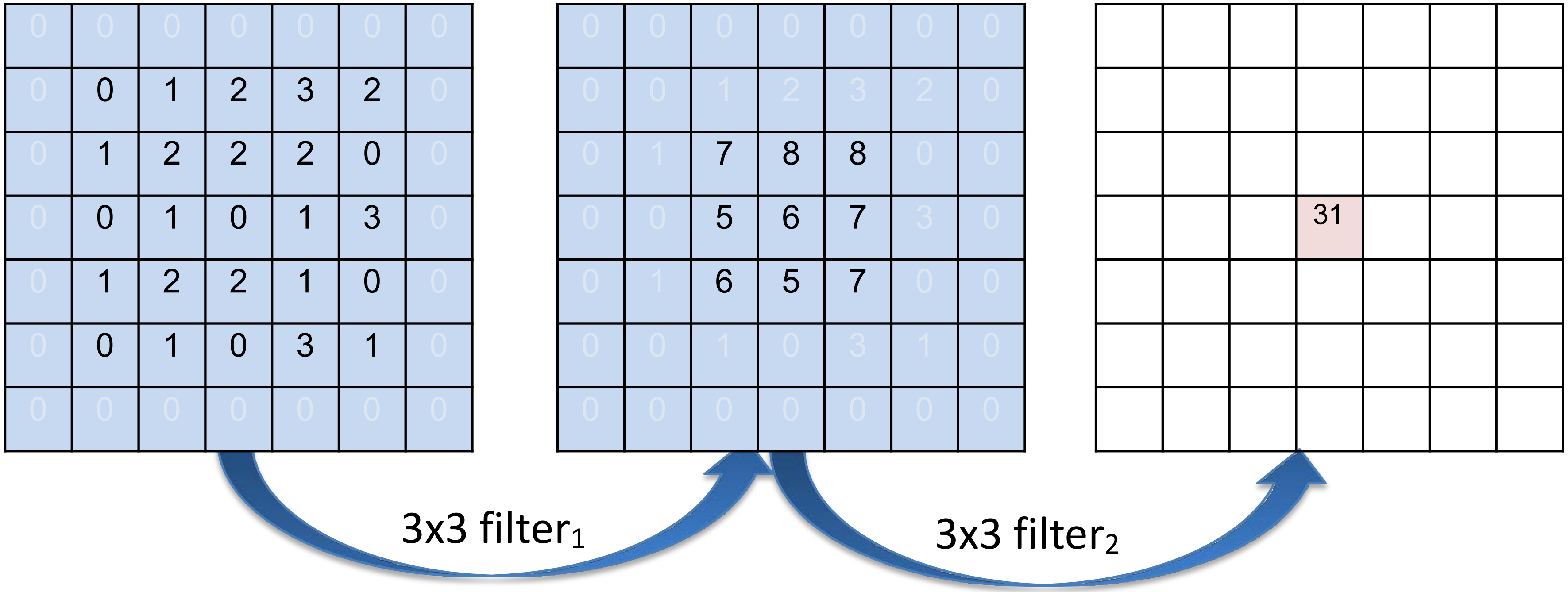
3x3 filter$_1$

VGGNet: Stacked Filters

- Deeper network means more weights
- Use stack of smaller filters (3x3) to cover the same receptive field with fewer filter weights

•Non-linear activation inserted between each filter <u>Example</u>: 5x5

filter (25 weights) à two 3x3 filters (18 weights)



0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	2	2	2	0	0
0	0	1	0	1	3	0
0	1	2	2	1	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

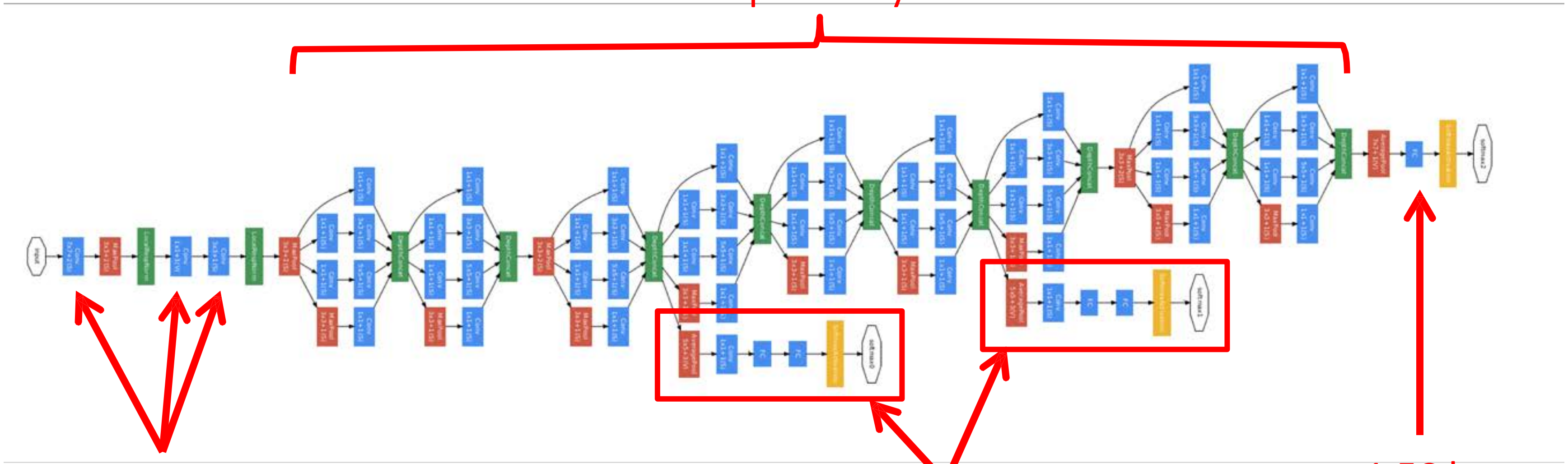
0	0	0	0	0	0	0
0	0	1	2	3	2	0
0	1	7	8	8	0	0
0	0	5	6	7	3	0
0	1	6	5	7	0	0
0	0	1	0	3	1	0
0	0	0	0	0	0	0

			31			

GoogLeNet/Inception (v1)

CONV Layers: 21 (depth), 57 (total)
Fully Connected Layers: 1 Weights: 7.0M
MACs: 1.43G

Also, v2, v3 and v4
ILSVRC14 Winner



9 Inception Layers

3 CONV layers

Auxiliary Classifiers
(helps with training, not used during inference)

1 FC layer
(reduced from 3)

[Szegedy et al., arXiv 2014, CVPR 2015]

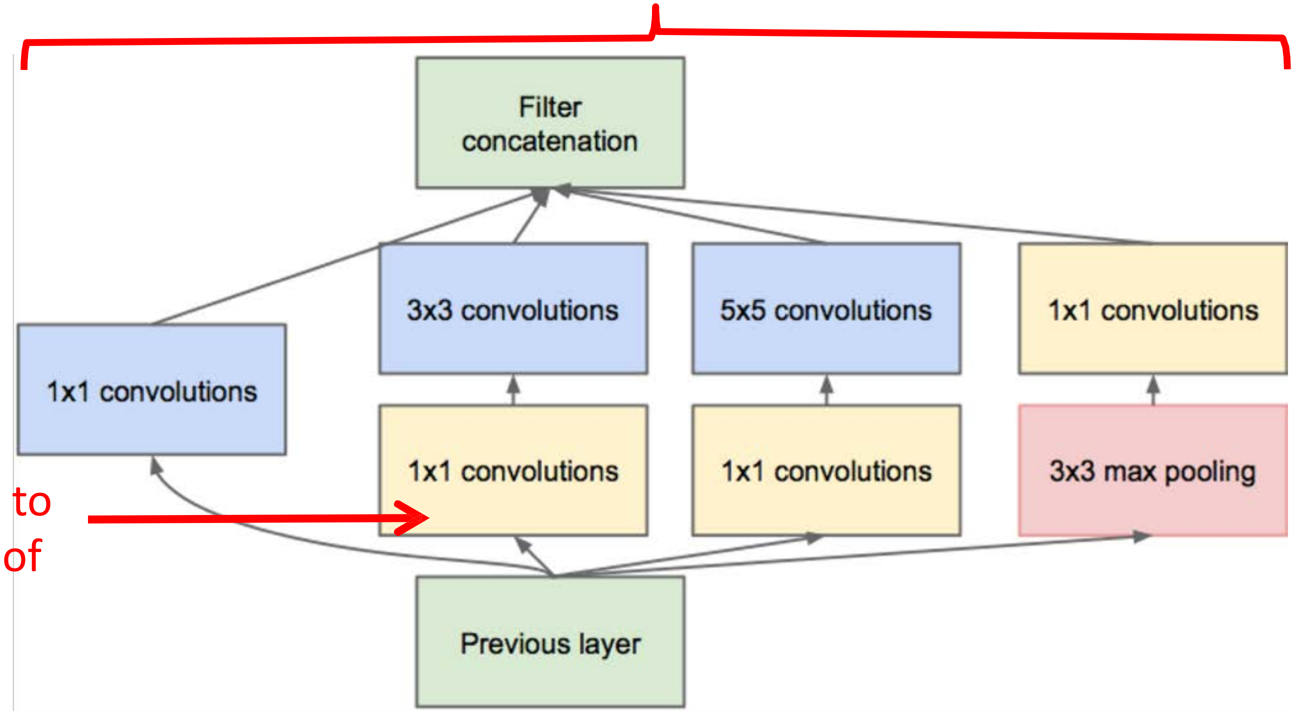
GoogLeNet/Inception (v1)

CONV Layers: 21 (depth), 57 (total)
Fully Connected Layers: 1 Weights: 7.0M
MACs: 1.43G

Also, v2, v3 and v4
ILSVRC14 Winner

parallel filters of different size have the effect of processing image at different scales

Inception Module

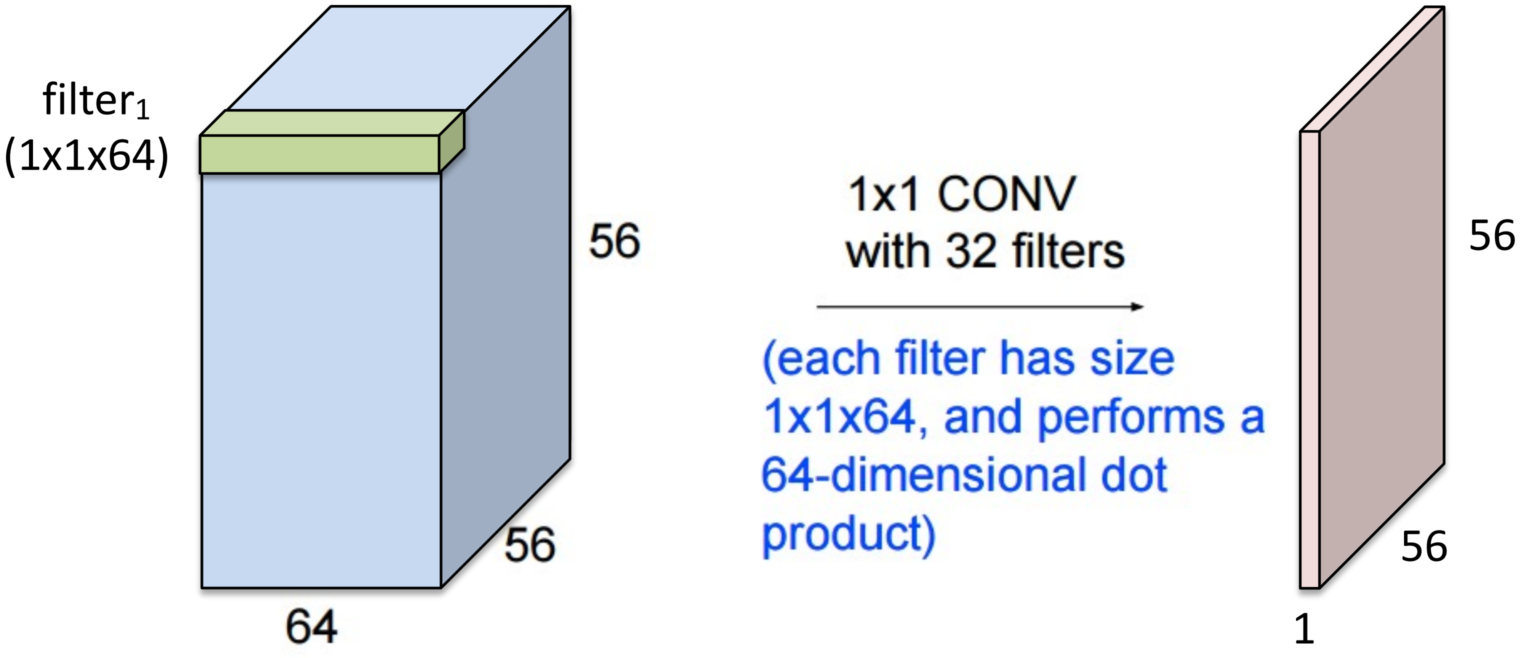


1x1 ‘bottleneck’ to reduce number of weights and multiplications

[Szegedy et al., arXiv 2014, CVPR 2015]

1x1 Bottleneck

Use **1x1 filter** to capture cross-channel correlation, but no spatial correlation.
Can be used to reduce the number of channels in next layer (**bottleneck**)



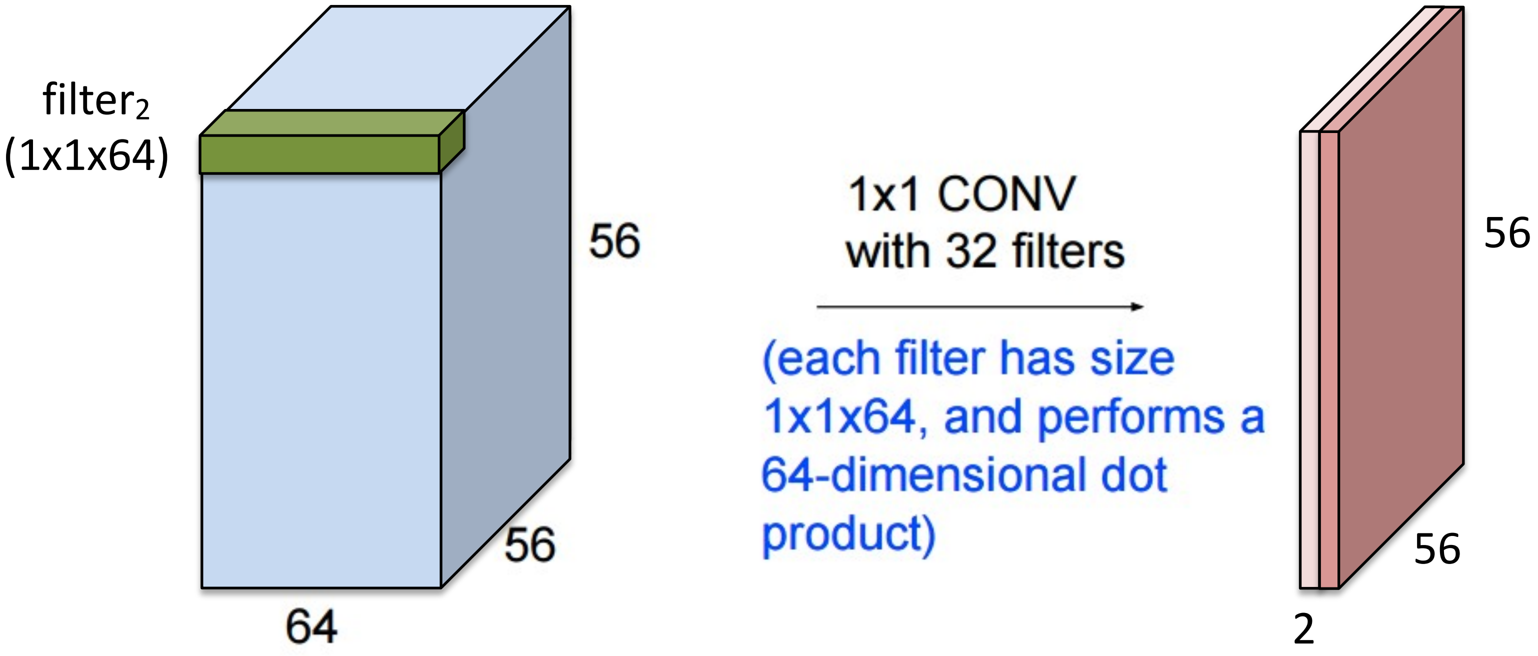
(each filter has size 1x1x64, and performs a 64-dimensional dot product)

Modified image from source:
Stanford cs231n

[Lin et al., Network in Network, arXiv 2013, ICLR 2014]

1x1 Bottleneck

Use **1x1 filter** to capture cross-channel correlation, but no spatial correlation.
Can be used to reduce the number of channels in next layer (**bottleneck**)



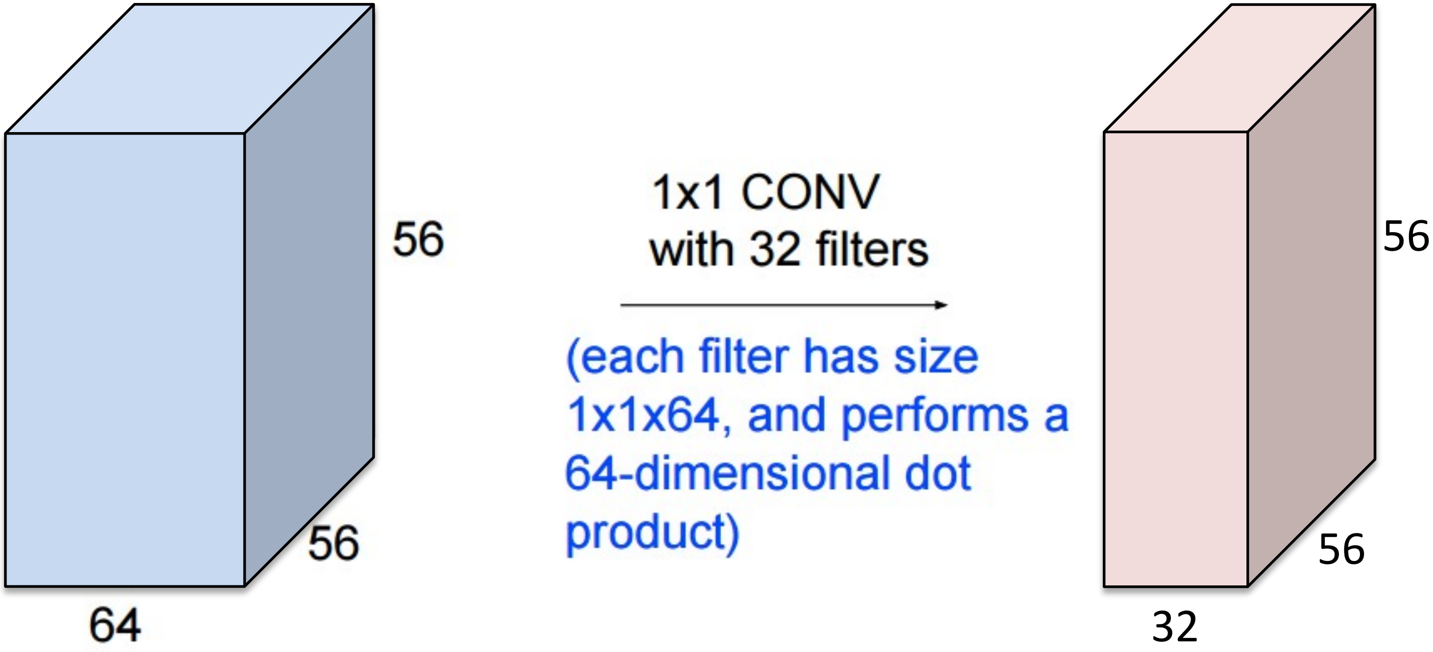
(each filter has size 1x1x64, and performs a 64-dimensional dot product)

Modified image from source:
Stanford cs231n

[Lin et al., Network in Network, arXiv 2013, ICLR 2014]

1x1 Bottleneck

Use **1x1 filter** to capture cross-channel correlation, but no spatial correlation.
Can be used to reduce the number of channels in next layer (**bottleneck**)



1x1 CONV
with 32 filters

(each filter has size 1x1x64, and performs a 64-dimensional dot product)

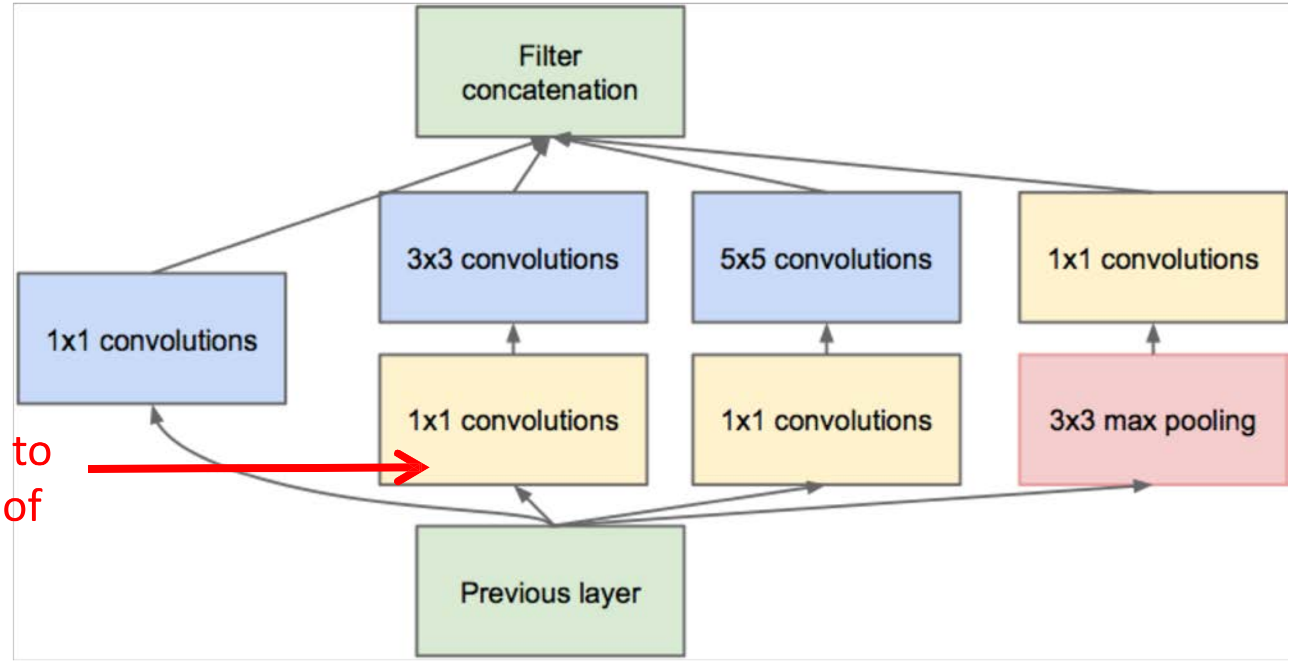
Modified image from source:
Stanford cs231n

[Lin et al., Network in Network, arXiv 2013, ICLR 2014]

GoogLeNet:1x1 Bottleneck

Apply bottleneck before ‘large’ convolution filters.
Reduce weights such that **entire CNN can be trained on one GPU.**
Number of multiplications reduced from 854M à 358M

Inception Module



1x1 ‘bottleneck’ to reduce number of weights and multiplications

[Szegedy et al., arXiv 2014, CVPR 2015]

ResNet

ILSVRC15 Winner (better than human level accuracy!)

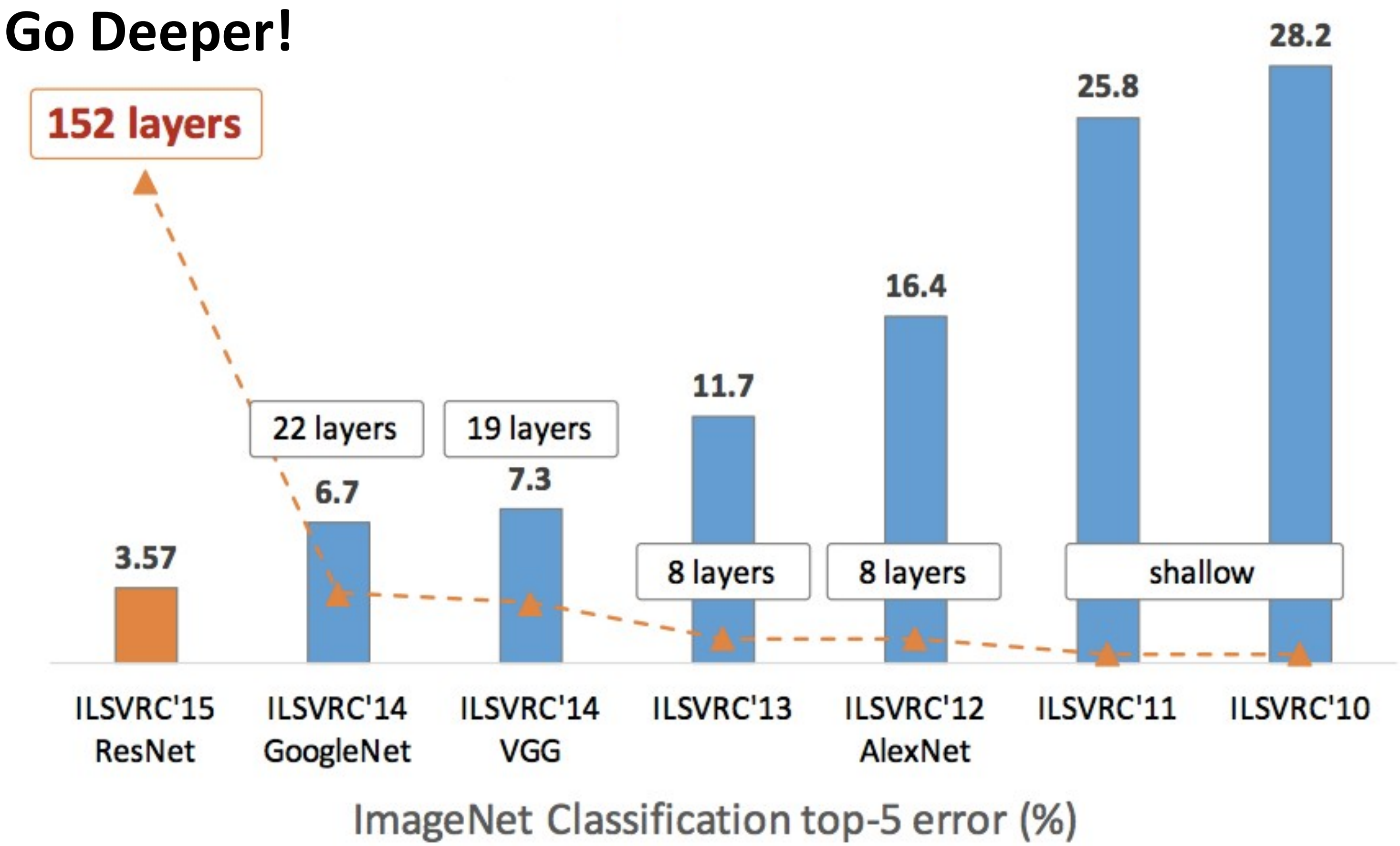
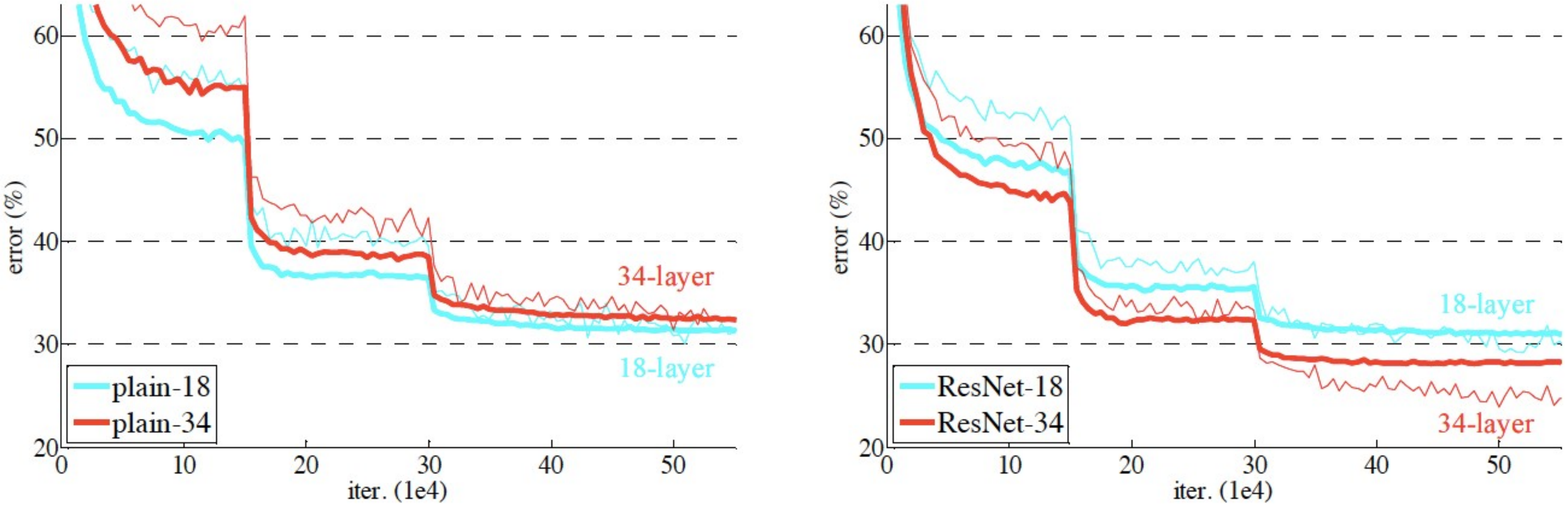


Image Source: http://icml.cc/2016/tutorials/icml2016_tutorial_deep_residual_networks_kaiminghe.pdf

ResNet: Training

Training and validation error **increases** with more layers; this is due to vanishing gradient, no overfitting.
Introduce **short cut module** to address this!



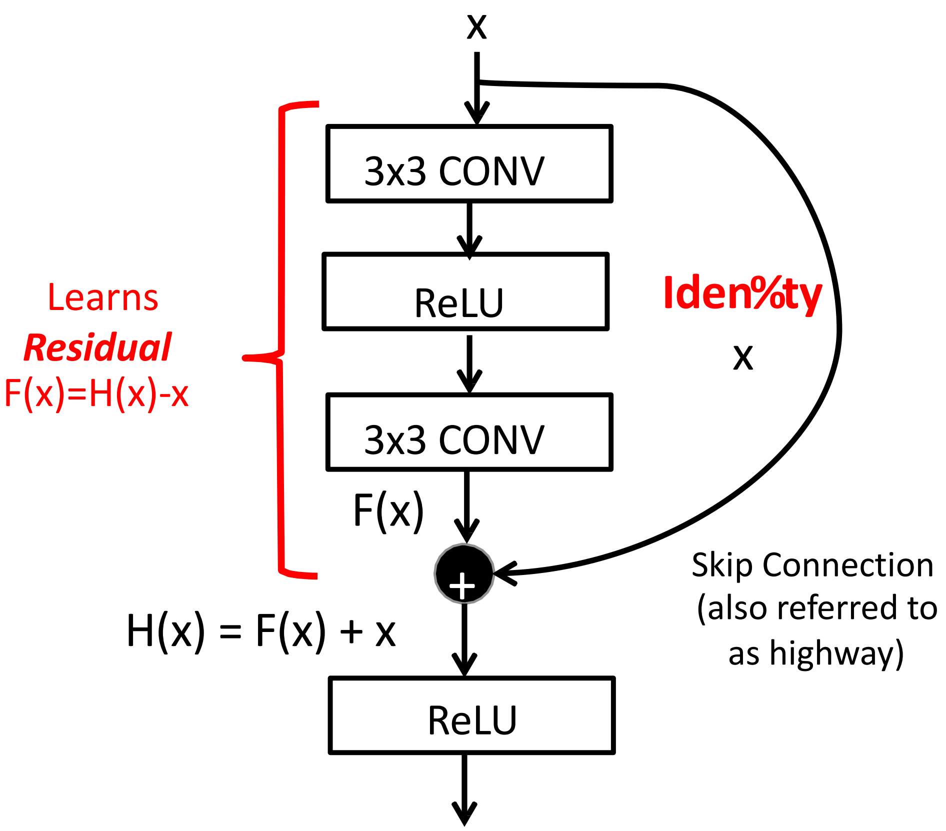
Without shortcut

With shortcut

Thin curves denote training error, and bold curves denote validation error.

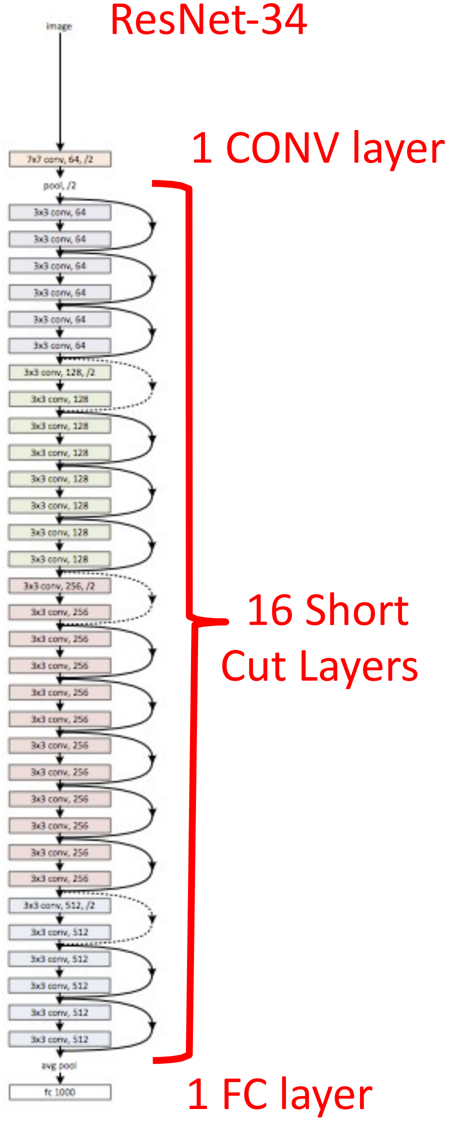
[He et al., arXiv 2015, CVPR 2016]

ResNet: Short Cut Module



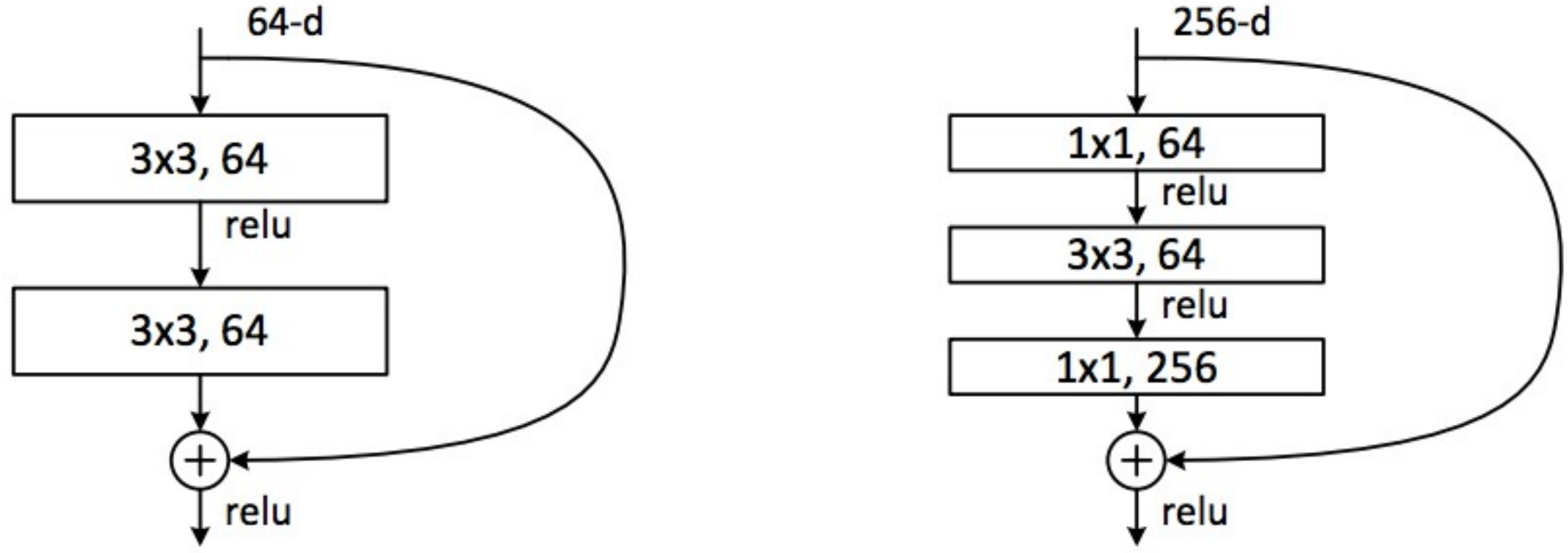
Helps address the vanishing gradient challenge for training very deep networks

[He et al., arXiv 2015, CVPR 2016]



ResNet: Bottleneck

Apply 1x1 bottleneck to reduce computation and size Also makes network deeper (ResNet-34 à   ResNet-50)



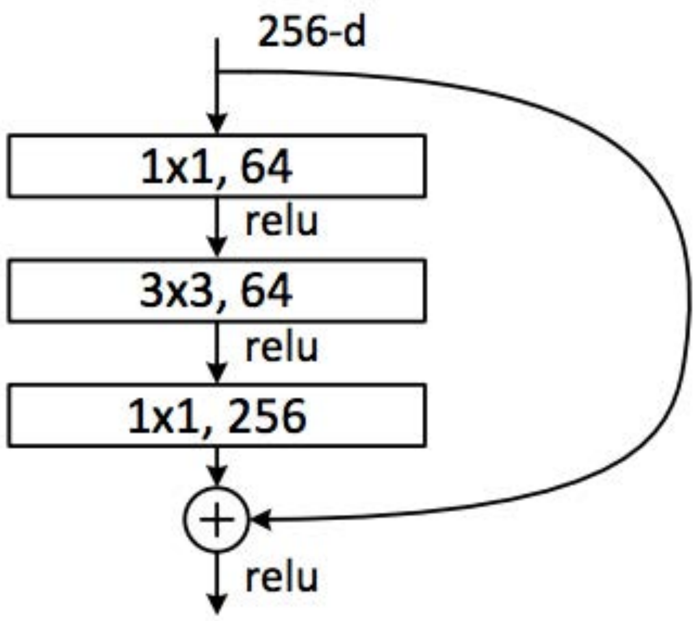
[He et al., arXiv 2015, CVPR 2016]

ResNet-50

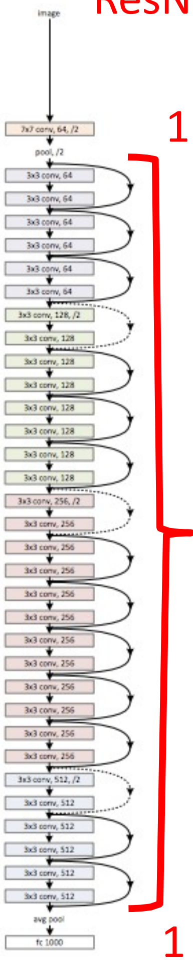
CONV Layers: 49
Fully Connected Layers: 1
Weights: 25.5M
MACs: 3.9G

Also, 34,**152** and 1202 layer versions
ILSVRC15 Winner

Short Cut Module



ResNet-34



1 CONV layer

16 Short Cut Layers

1 FC layer

[He et al., arXiv 2015, CVPR 2016]

Summary of Popular DNNs

Metrics	LeNet-5	AlexNet	VGG-16	GoogLeNet (v1)	ResNet-50
Top-5 error	n/a	16.4	7.4	6.7	5.3
Input Size	28x28	227x227	224x224	224x224	224x224
# of CONV Layers	**2**	**5**	**16**	**21 (depth)**	**49**
Filter Sizes	5	3, 5,11	3	1, 3 , 5, 7	1, 3, 7
# of Channels	1, 6	3 - 256	3 - 512	3 - 1024	3 - 2048
# of Filters	6, 16	96 - 384	64 - 512	64 - 384	64 - 2048
Stride	1	1, 4	1	1, 2	1, 2
# of Weights	2.6k	2.3M	14.7M	6.0M	23.5M
# of MACs	283k	666M	15.3G	1.43G	3.86G
# of FC layers	**2**	**3**	**3**	**1**	**1**
# of Weights	58k	58.6M	124M	1M	2M
# of MACs	58k	58.6M	124M	1M	2M
Total Weights	**60k**	**61M**	**138M**	**7M**	**25.5M**
Total MACs	**341k**	**724M**	**15.5G**	**1.43G**	**3.9G**

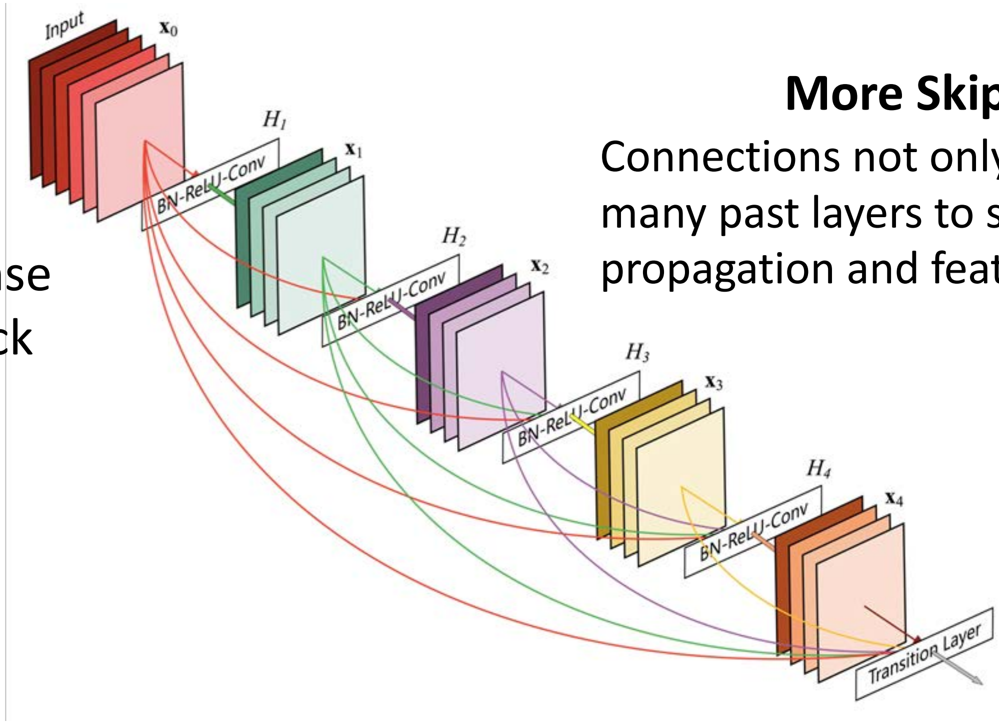
CONV Layers increasingly important!

Summary of Popular DNNs

- **AlexNet**
  - First CNN Winner of ILSVRC
  - Uses LRN (deprecated after this)
- **VGG-16**
  - Goes Deeper (16+ layers)
  - Uses only 3x3 filters (stack for larger filters)
- **GoogLeNet (v1)**
  - Reduces weights with Inception and only one FC layer
  - Inception: 1x1 and DAG (parallel connections)
  - Batch Normalization
- **ResNet**
  - Goes Deeper (24+ layers)
  - Shortcut connections

DenseNet

Dense Block

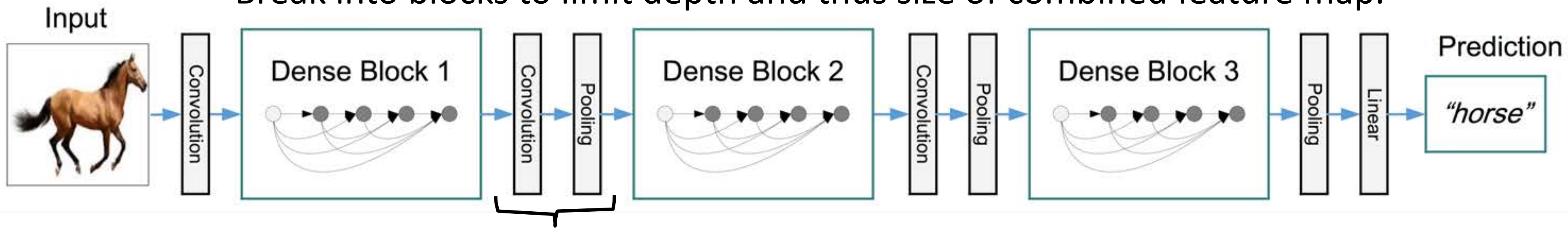


More Skip Connections!

Connections not only from previous layer, but many past layers to strengthen feature map propagation and feature reuse.

Feature maps are concatenated rather than added.

Break into blocks to limit depth and thus size of combined feature map.



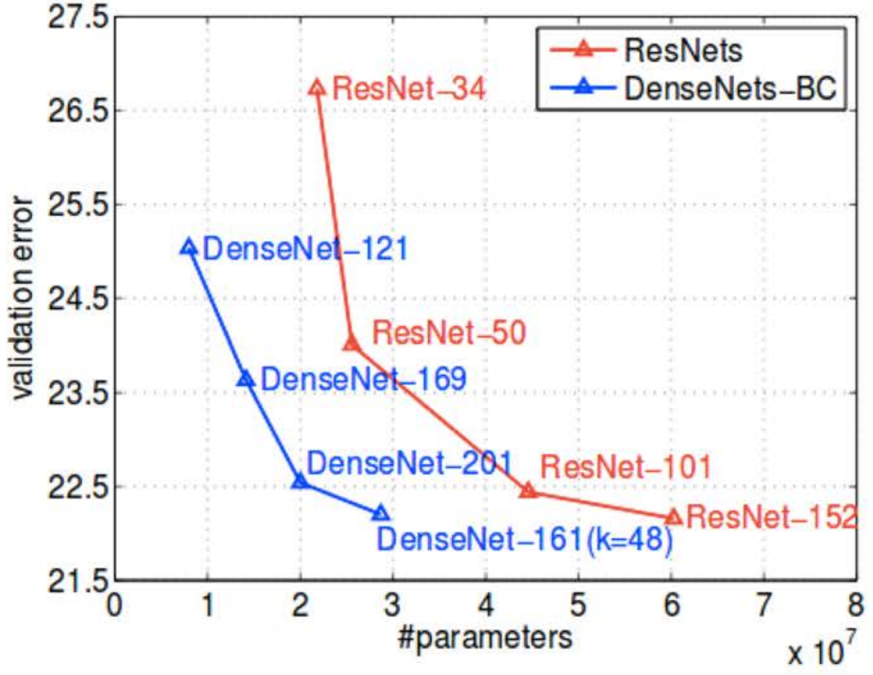
Transition layers

[Huang et al., CVPR 2017]

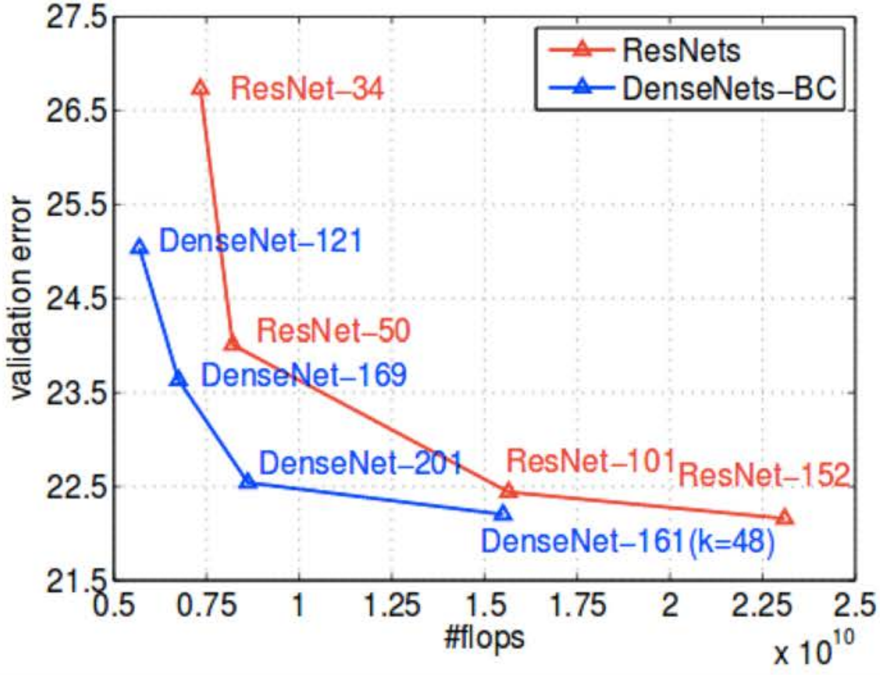
DenseNet

Higher accuracy than ResNet with fewer weights and multiplications

Top-1 error



Top-1 error



Note: 1 MAC = 2 FLOPS

[Huang et al., CVPR 2017]

Wide ResNet

Increase width (# of filters) rather than depth of network

- 50-layer wide ResNet outperforms 152-layer original ResNet
- Increasing width instead of depth is also more parallel-friendly

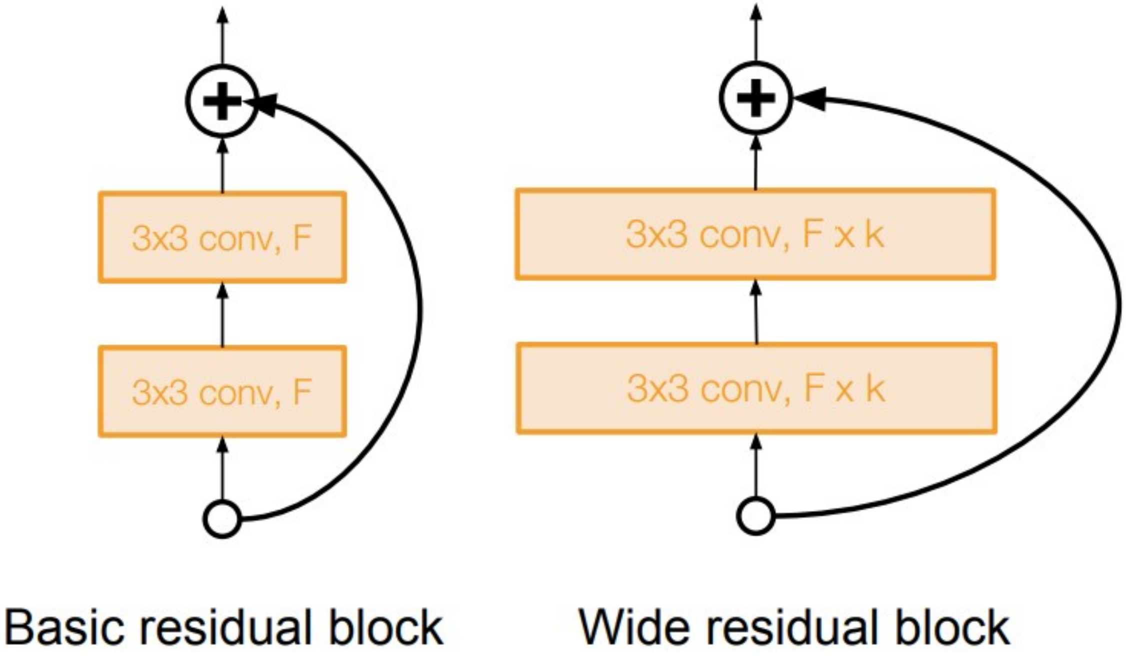
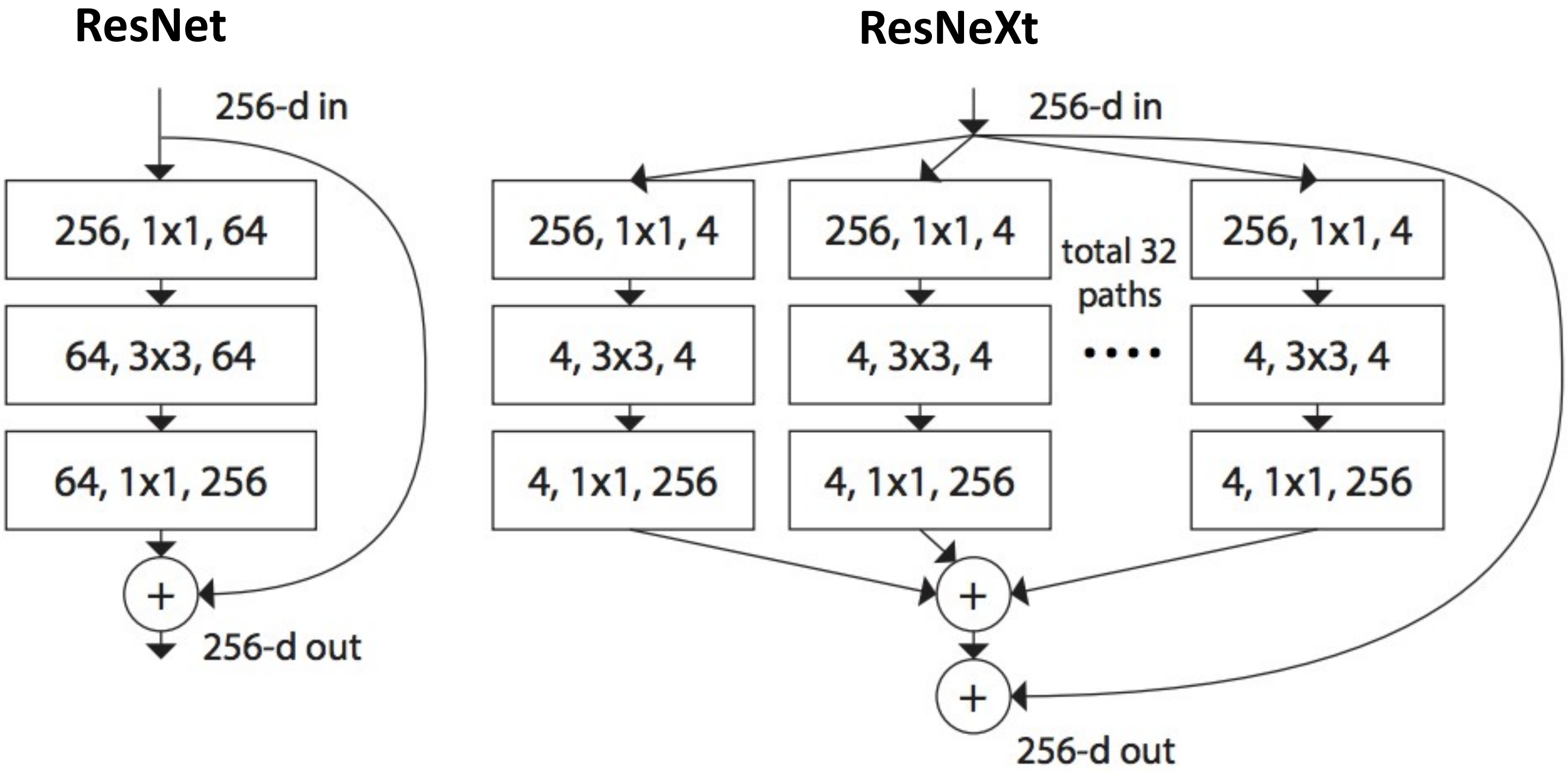


Image Source: Stanford cs231n

[Zagoruyko et al., BMVC 2016]

ResNeXt

Increase number of **convolution groups** (referred to as *cardinality*) instead of depth and width of network



ResNet

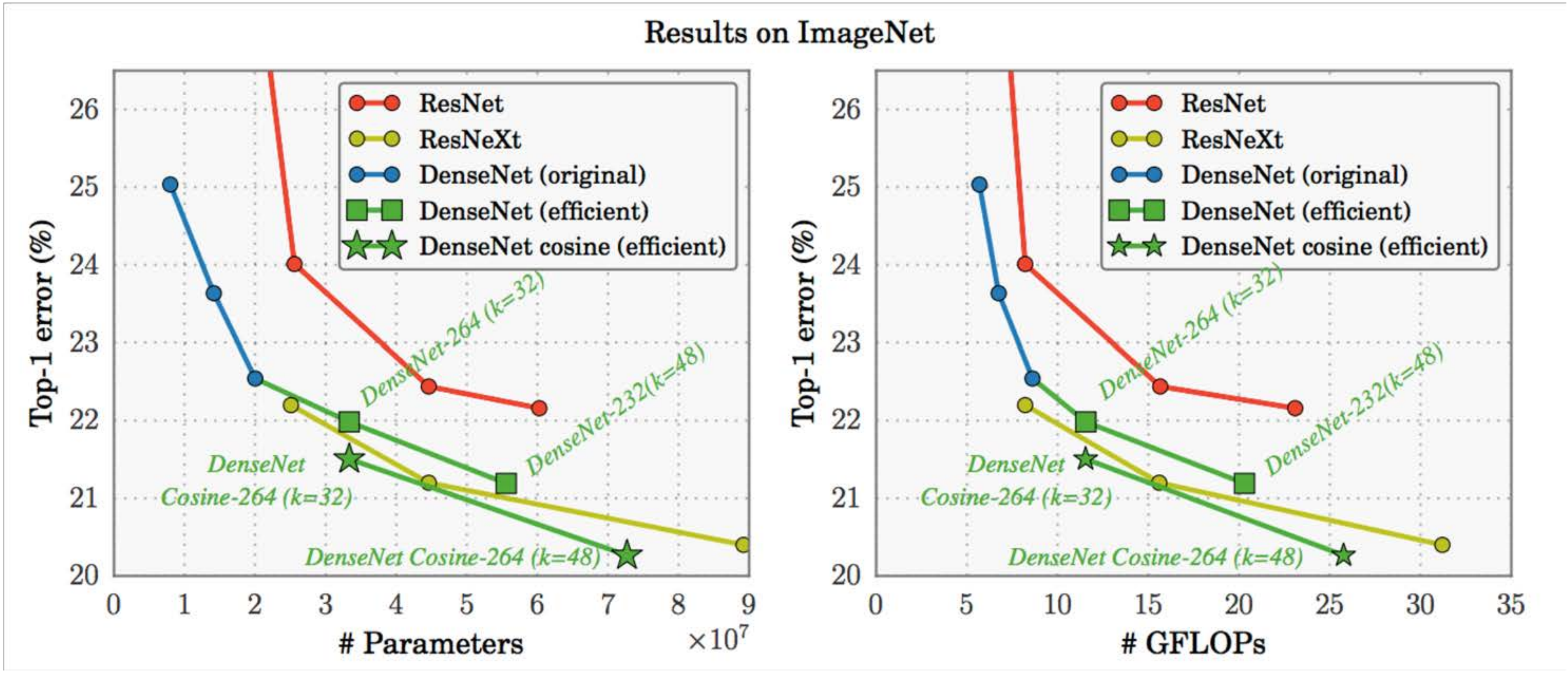
ResNeXt

Used by ILSVRC 2017
Winner WMW

[Xie et al., CVPR 2017]

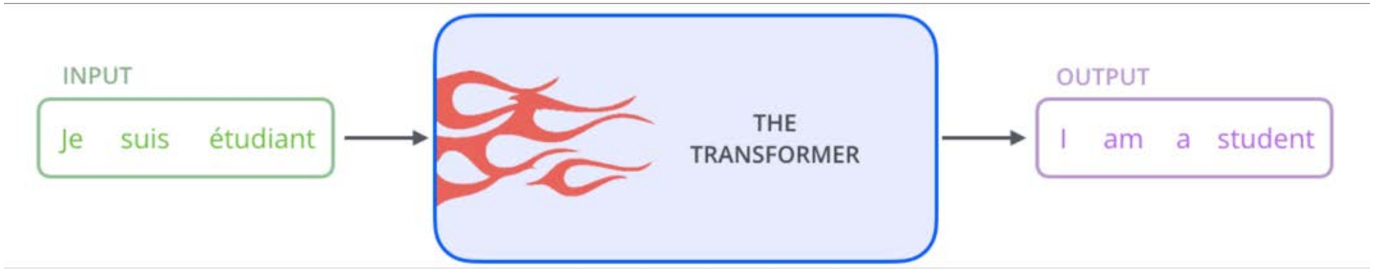
ResNeXt

Improved accuracy vs. 'complexity' tradeoff compared to other ResNet based models

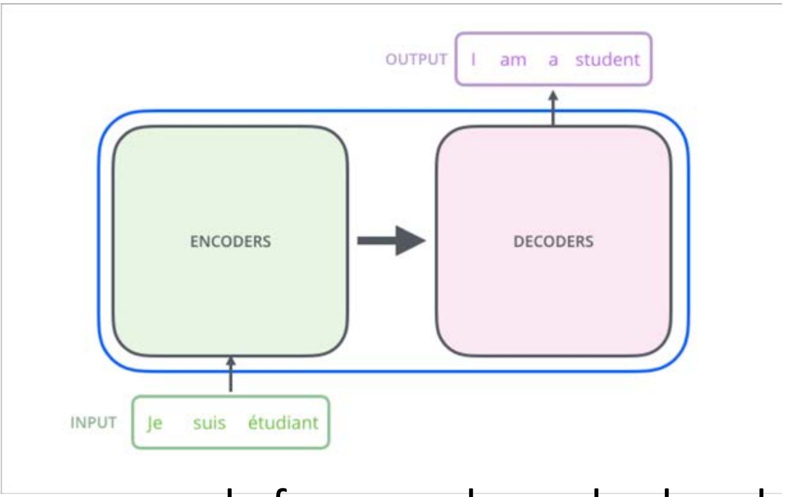


Transformer

A popular model in both natural language processing and computer vision.



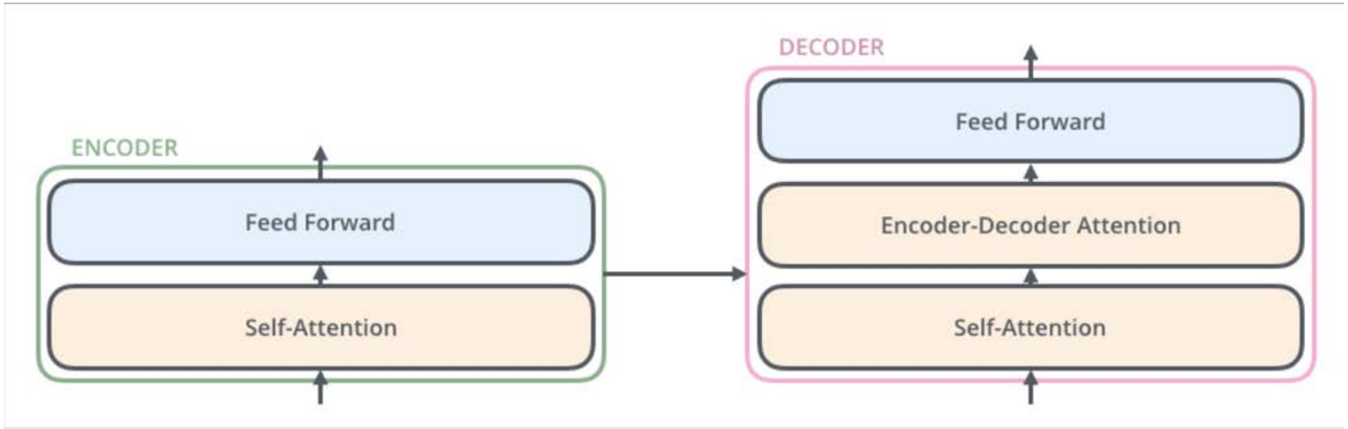
1. Transformer is first proposed for machine translation.



2. Transformer is composed of an encoder and a decoder. The encoder is used to encode the sentence into s hidden space while the decoder is used to decoder the feature from the hidden space to a sentence in another language.

Transformer

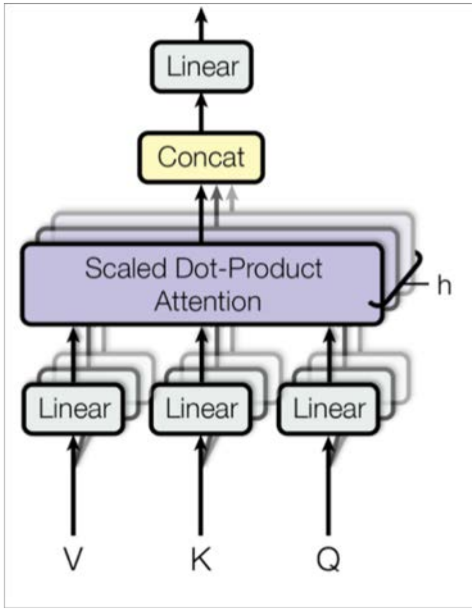
A popular model in both natural language processing and computer vision.



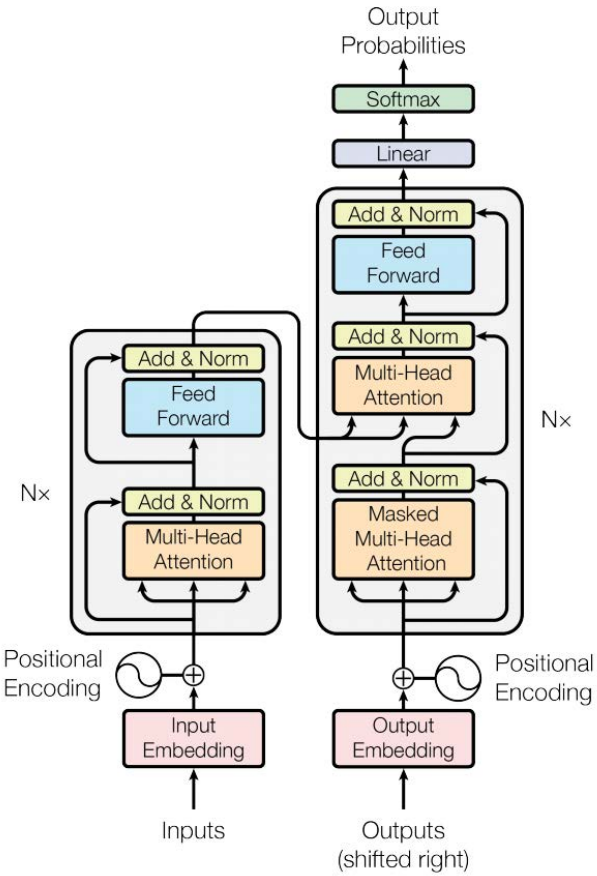
3. Both the encoder and decoder are mainly composed of “Feed Forward” (Fully Connected Layer) and Self- Attention. The outputs of encoders are inputted into the decoder with attention mechanism.

Transformer

A popular model in both natural language processing and computer vision.



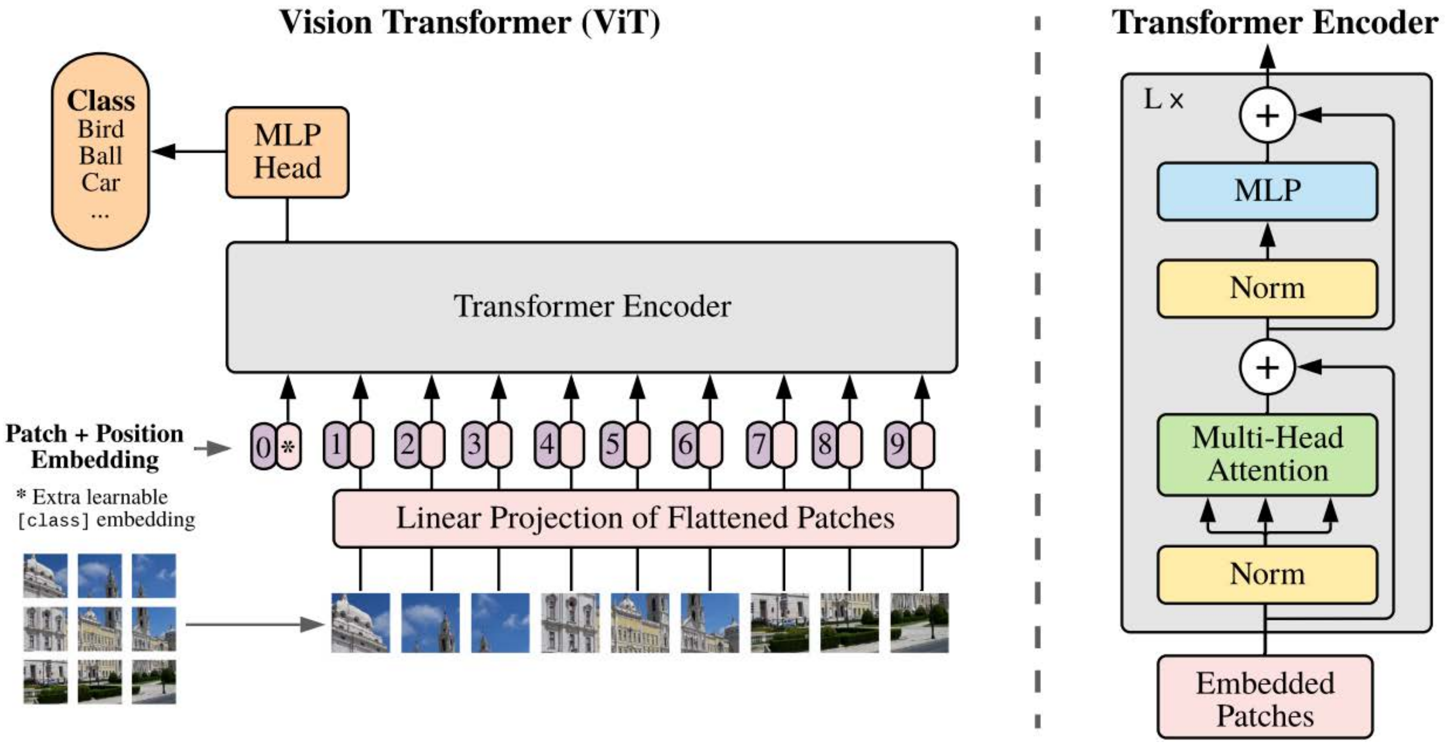
3. Self-Attention. First compute the Value, Key and Query of different words. Then, compute the relation of different words by computing the similarity between their Keys and Queries.



4. The overview of Transformer.

Vision Transformer

Apply Transformers to vision tasks.

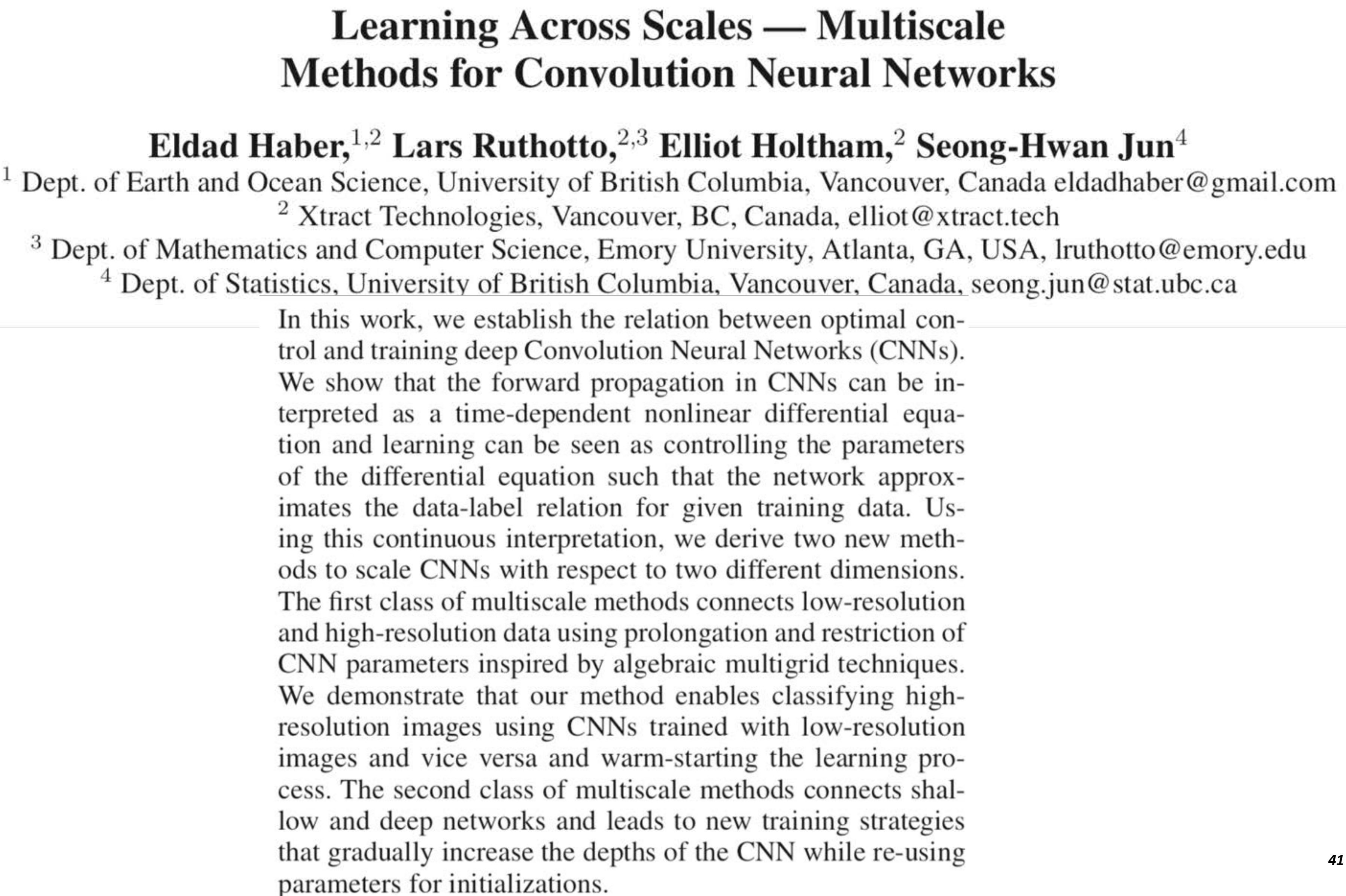


1. Split an image into several patches (usually 16x16). Then, compute the embedding vector of each patch with Linear Projection (a linear convolution).
2. Input the embedding vectors into the Transformer. Each patch is regarded as a word in the sentence.

Summary

- Approaches used to improve accuracy by popular DNN models in the ImageNet Challenge
  - Go deeper (i.e. more layers)
  - Stack smaller filters and apply 1x1 bottlenecks to reduce number of weights such that the deeper models can fit into a GPU (faster training)
  - Use multiple connections across layers (e.g. parallel and short cut)
- Filter shapes vary across layers and models
  - Need flexible hardware!

Theory？



Learning Across Scales — Multiscale Methods for Convolution Neural Networks

Eldad Haber,[1,2] Lars Ruthotto,[2,3] Elliot Holtham,[2] Seong-Hwan Jun[4]
[1] Dept. of Earth and Ocean Science, University of British Columbia, Vancouver, Canada eldadhaber@gmail.com
[2] Xtract Technologies, Vancouver, BC, Canada, elliot@xtract.tech
[3] Dept. of Mathematics and Computer Science, Emory University, Atlanta, GA, USA, lruthotto@emory.edu
[4] Dept. of Statistics, University of British Columbia, Vancouver, Canada, seong.jun@stat.ubc.ca

In this work, we establish the relation between optimal control and training deep Convolution Neural Networks (CNNs). We show that the forward propagation in CNNs can be interpreted as a time-dependent nonlinear differential equation and learning can be seen as controlling the parameters of the differential equation such that the network approximates the data-label relation for given training data. Using this continuous interpretation, we derive two new methods to scale CNNs with respect to two different dimensions. The first class of multiscale methods connects low-resolution and high-resolution data using prolongation and restriction of CNN parameters inspired by algebraic multigrid techniques. We demonstrate that our method enables classifying high-resolution images using CNNs trained with low-resolution images and vice versa and warm-starting the learning process. The second class of multiscale methods connects shallow and deep networks and leads to new training strategies that gradually increase the depths of the CNN while re-using parameters for initializations.

Theory？

- Can you infer the theory for Transformer？
- Two papers for reference：
  - Attenion is all you need
  - Learning Across Scales---Multiscale Methods for Convolution Neural Networks