COMP/EECE 7/8740 Neural Networks

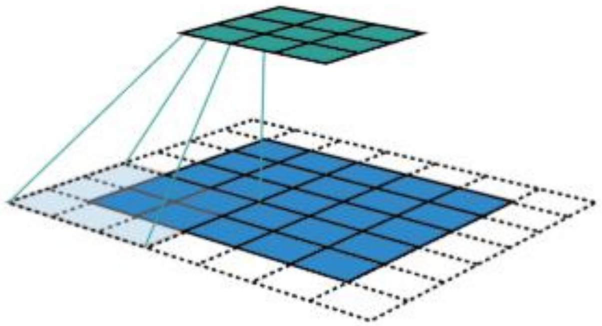
Topics:

- Receptive field for CNN
- CNN architectures or models
  - Classification models
  - Segmentation and models
  - Detection models

Md Zahangir Alom
Department of Computer Science
University of Memphis, TN

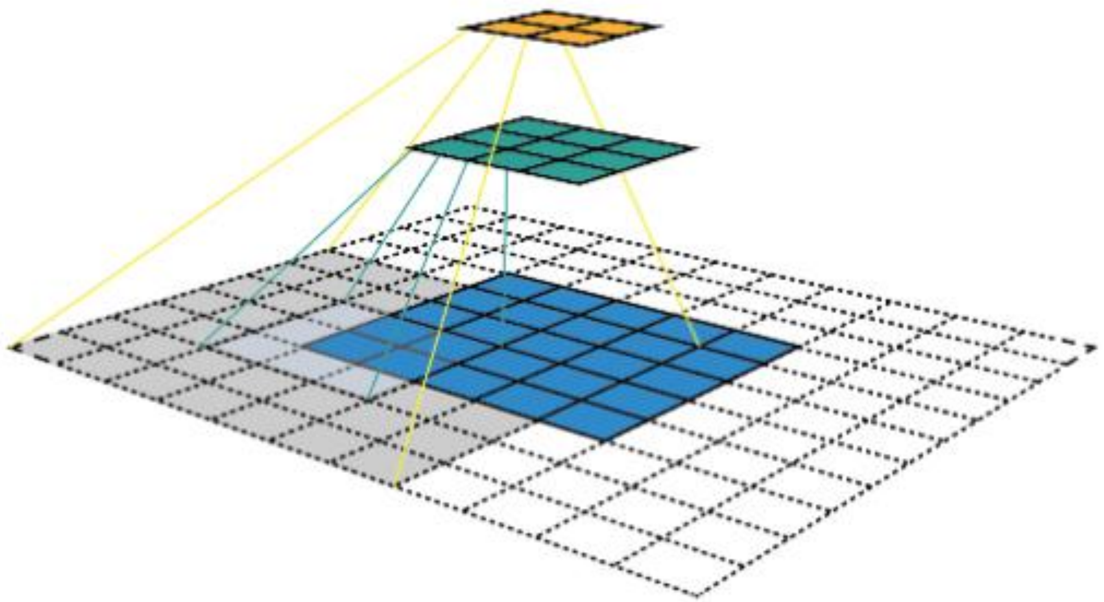
Receptive field in CNNs

- The **receptive field** is defined as the region in the input space that a particular CNN's feature is looking at (i.e. be affected by).
- Not all pixels in a receptive field is equally important to its corresponding CNN's feature
- Closer a pixel to the center of the RF, the more it contributes to the calculation of the output feature (focus exponentially more to the middle of that region).



Convolution and receptive field

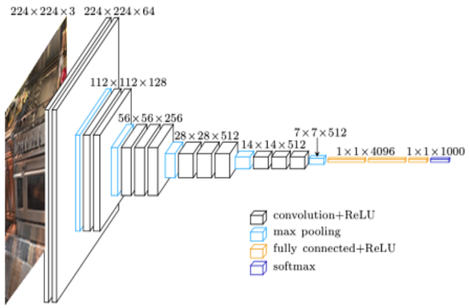
Receptive field in CNNs

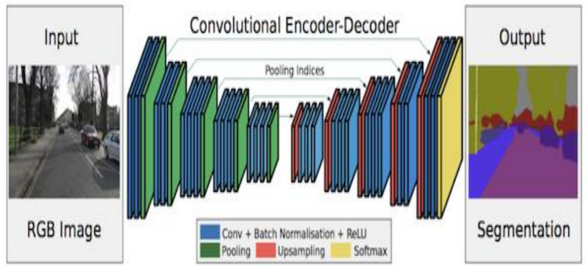


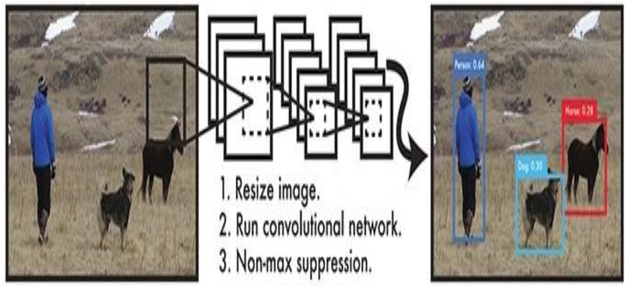
By applying a convolution C with kernel size ***k* = 3x3**, padding size ***p* = 1x1**, stride ***s* = 2x2** on an input map **5x5**, we will get an output feature map **3x3** (green map). Applying the same convolution on top of the 3x3 feature map, we will get a **2x2** feature map (orange map).

Deep CNN models and applications

- Classification
- Segmentation
- Detection







Models for classification:

- AlexNet
- VGG Net
- GoogleNet
- ResNet
- Inception-ResNet
- DenseNet / DCRN
- FractalNet
- CapsuleNet and
- IRRCNN
- …..

Models for Segmentation:

- FCN
- SegNet
- Dialted Convolution
- RefineNet
- Pyramid Scene Parsing (PSP):PSPNet
- DeepLab
- U-Net
- R2U-Net
- NABLA-N Net
- ……

Models for Detection:

- Region based CNN (RCNN)
- Fast RCNN
- Faster RCNN
- Mask RCNN
- You Only Look Once (YOLO)
- Single Short Multibox Detector (SSD)
- UD-Net
- ……

Seg.: http://blog.qure.ai/notes/semantic-segmentation-deep-learning-review

Detection: https://towardsdatascience.com/

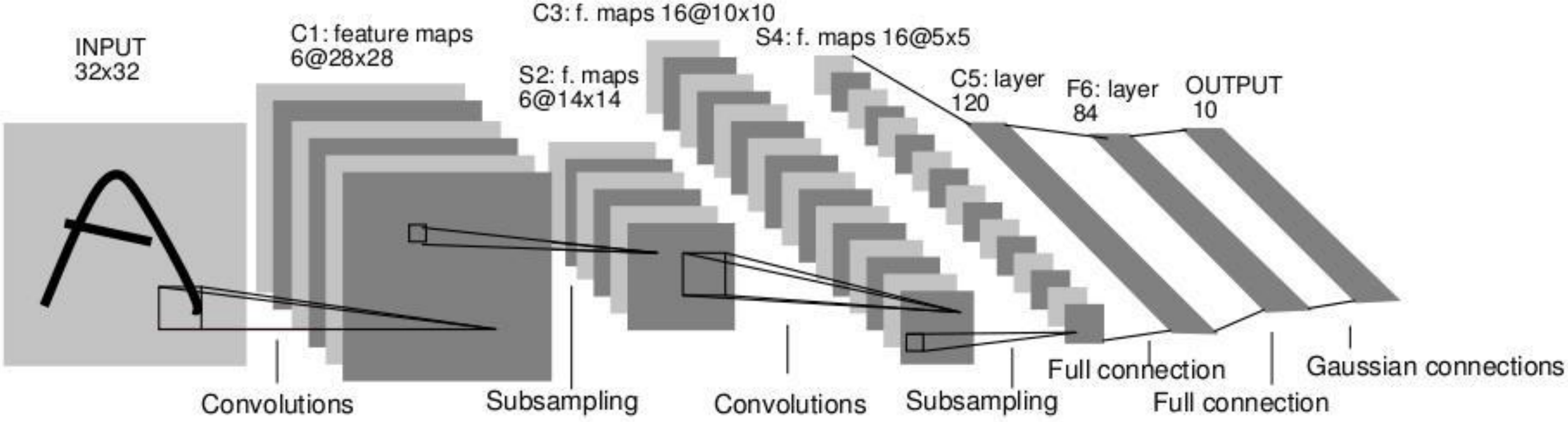
CNN models

- **LeNet :** Yann LeCun in 1998
- **AlexNet :** ILSVR winner in 2012
- **ZFNet :** Matthew Zeiler and Rob Fergue won the ILSVRC 2013, Refinement of AlexNet
- **VGGNET:** Visual Geometry Group (VGG) from Oxford University runner up of ILSVRC in 2014.
- **Network in Network (NiN)**: from NUS in 2014

CNN models

- **GoogLeNet(2014)**: Szegedy from the Google who was the winner of ILSVRC in 2014.
- **ResNet (2015) :** from Microsoft won the ILSVRC in 2015.
- **Inception-Residual Network by** C. Szegedy in 2016
- **DenseNet (Dec. 2016)** : from Cornel University by Gao Huang and others (CVPR-2017 best paper award) from Cornell University
- **FractalNet (2016):** Ultra-Deep Neural Networks without Residuals from University of Chicago.
- **PolyNet**
- **Res2Net in 2019**

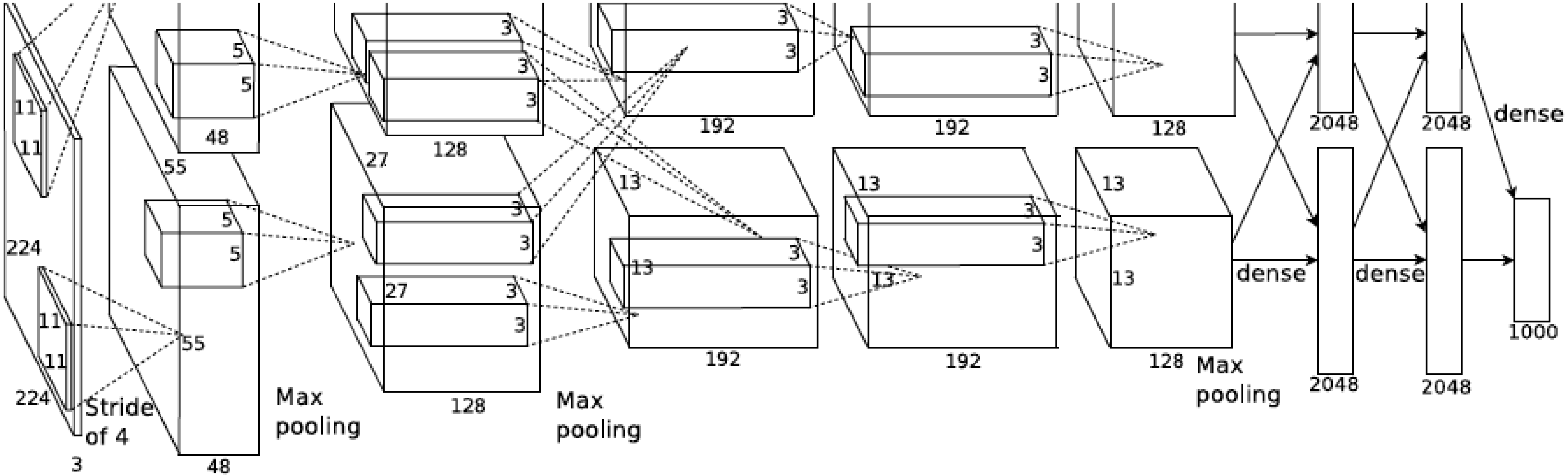
LeNet-5



- Average pooling
- Sigmoid or tanh nonlinearity
- Fully connected layers at the end
- Trained on MNIST digit dataset with 60K training examples

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, [Gradient-based learning applied to document recognition](), Proc. IEEE 86(11): 2278–2324, 1998.

AlexNet: ILSVRC 2012 winner

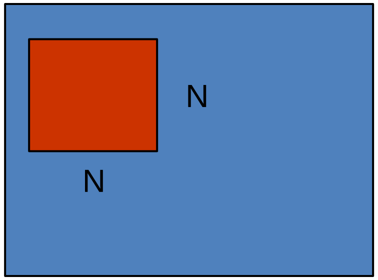


- Similar Framework to LeNet but:
  - Max pooling, ReLU nonlinearity
  - More data and biger model (7 hidden layers, 650K unit, 61M params)
  - GPU implémentation (50x speed up over CPU)
    - Trained on two GPUs for a week
  - Dropout régularisation
  - Local Response Normalization (LRN)

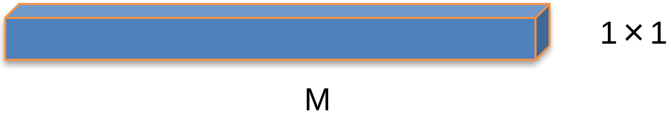
A. Krizhevsky, I. Sutskever, and G. Hinton, [ImageNet Classification with Deep Convolutional Neural Networks](), NIPS 2012

Local Response Normalization (LRN)

- **LRN** layer implements the lateral inhibition and objective to amplify the excited neuron while dampening the surrounding neurons.
- Two approaches for LRN:
  - Consider same channel or feature map and 2D neighborhood of dimension N x N, where N is the size of the normalization window. Normalize the window using the values in this neighborhood.
  - Normalizing across channels or feature maps, you will consider a neighborhood along the third dimension but at a single location.



First approach

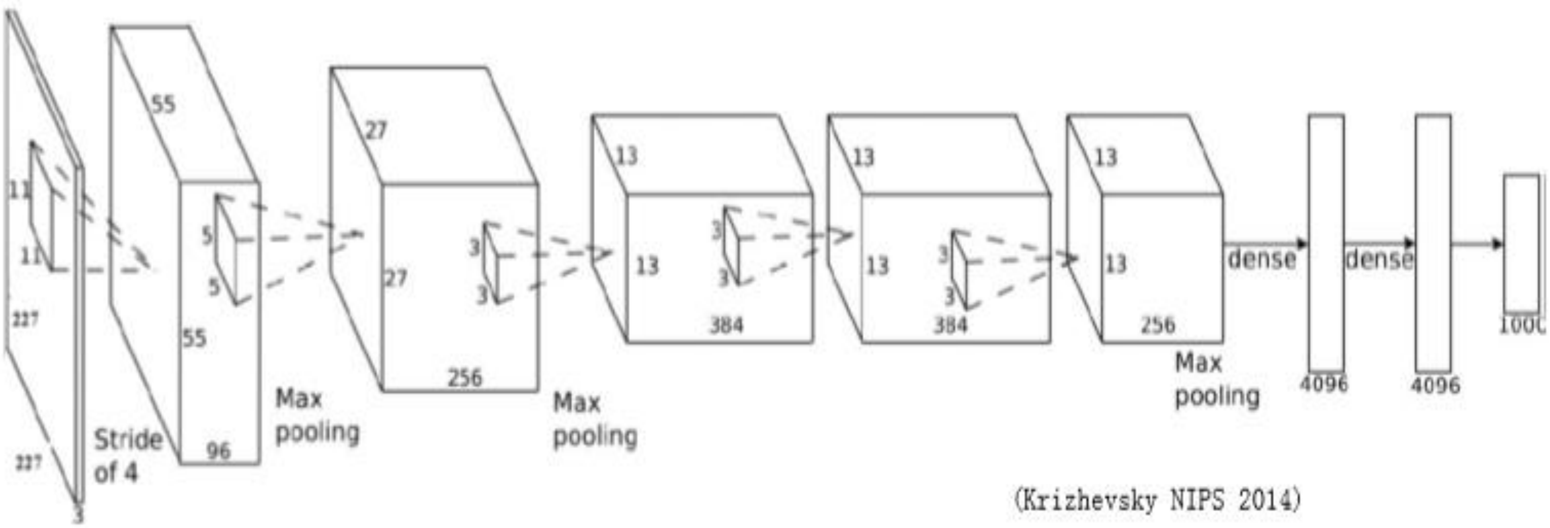


Second approach

A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

AlexNet: ILSVRC 2012 winner

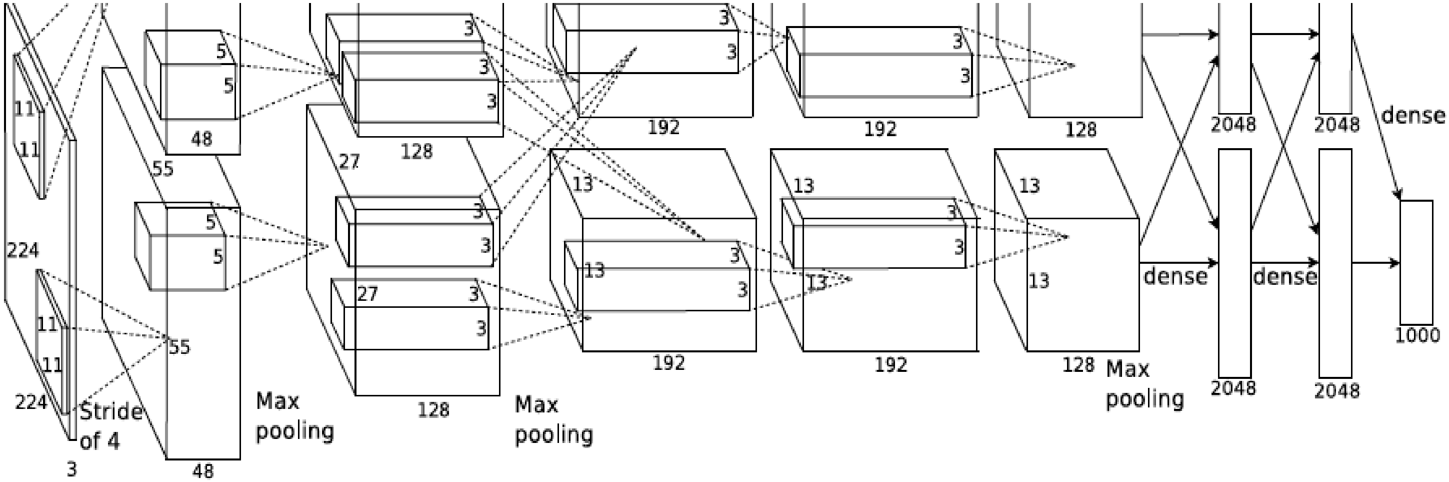
- The size of input sample is 224x224x3,
- Filter/ Kernel/ receptive field size 11,
- Stride 4 and the
- Output of the first convolution layer is 55x55x96.



A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

Summary on AlexNet :2012

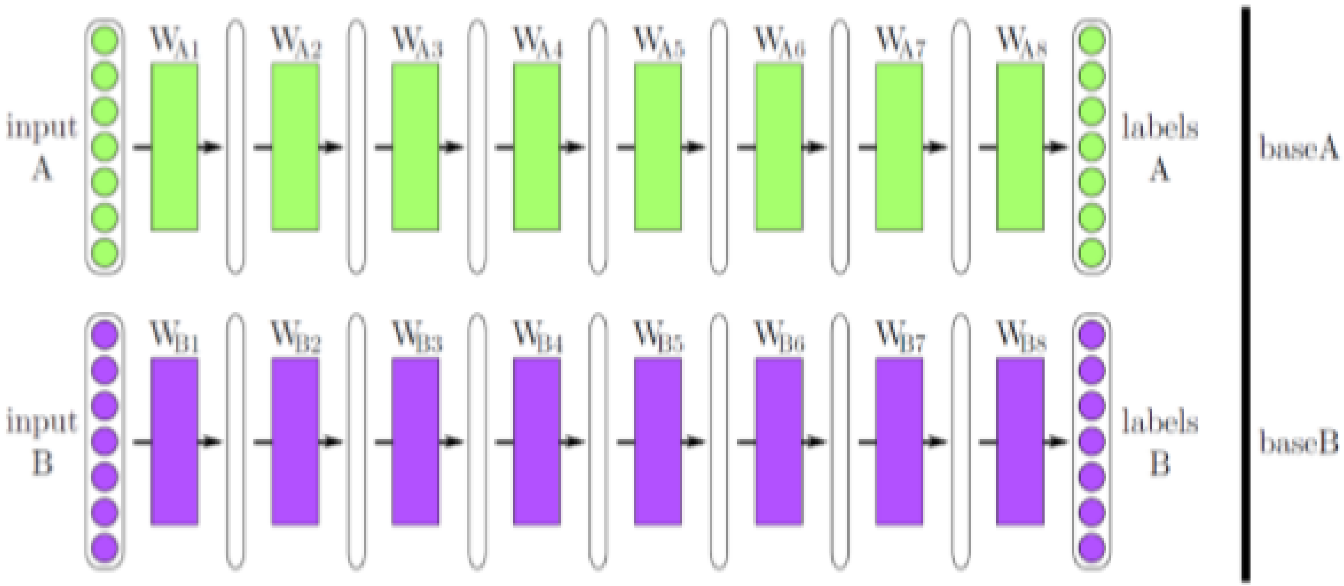
- The first deep learning model shown to be effective on large scale computer vision task.
- The first time a very large scale deep model is adopted.
- GPU is shown to be every effective on this large scale deep learning model.



A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

How it’s implemented for ImageNet

- ImageNet are divided into two groups of 500 classes, A and B
- Two 8-layer AlexNets, base A and base B, are trained on the to groups respectively



A. Krizhevsky, I. Sutskever, and G. Hinton, [ImageNet Classification with Deep Convolutional Neural Networks](), NIPS 2012

Clarifai/ ZFNet : ILSVRC 2013 winner

- Refinement of AlexNet

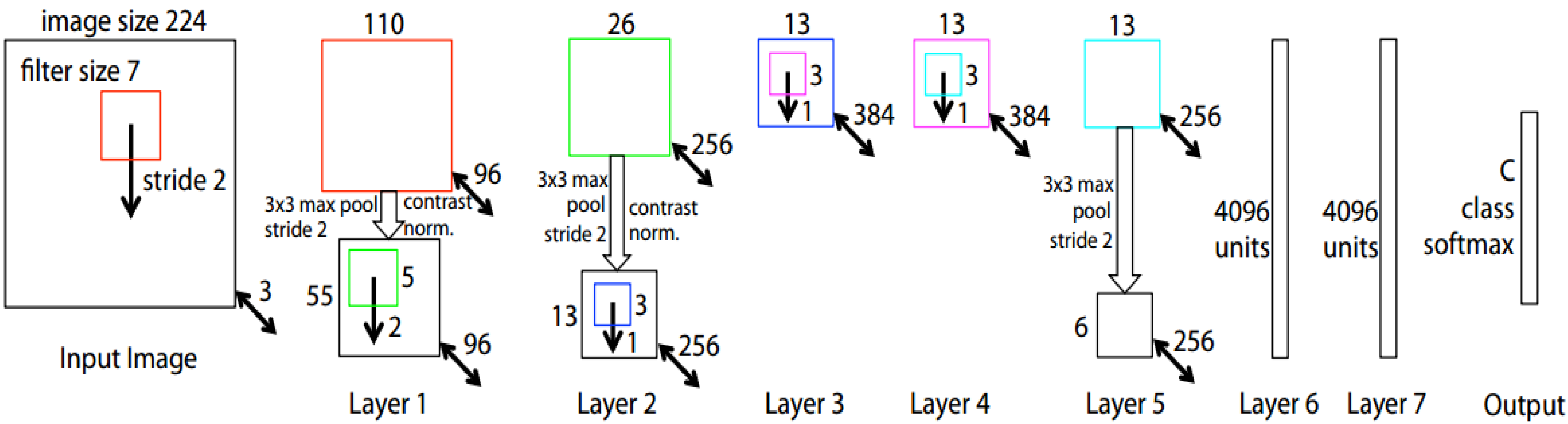


Figure 3. Architecture of our 8 layer convnet model. A 224 by 224 crop of an image (with 3 color planes) is presented as the input. This is convolved with 96 different 1st layer filters (red), each of size 7 by 7, using a stride of 2 in both x and y. The resulting feature maps are then: (i) passed through a rectified linear function (not shown), (ii) pooled (max within 3x3 regions, using stride 2) and (iii) contrast normalized across feature maps to give 96 different 55 by 55 element feature maps. Similar operations are repeated in layers 2,3,4,5. The last two layers are fully connected, taking features from the top convolutional layer as input in vector form ($6 \cdot 6 \cdot 256 = 9216$ dimensions). The final layer is a C-way softmax function, C being the number of classes. All filters and feature maps are square in shape.

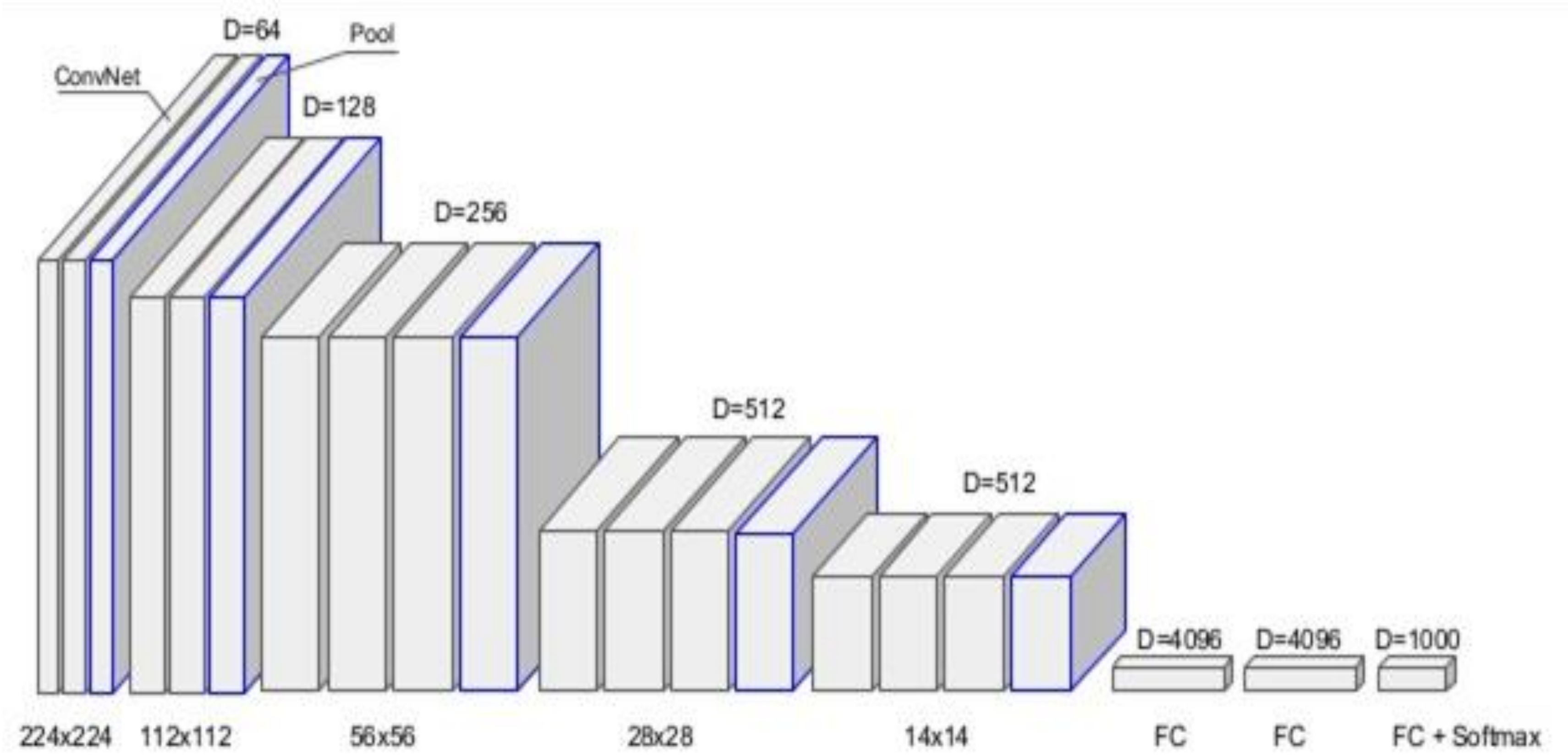
M. Zeiler and R. Fergus, Visualizing and Understanding Convolutional Networks,
ECCV 2014 (Best Paper Award winner)

Clarifai/ZFNet: ILSVRC 2013 winner

- Max-pooling layers follow first, second, and fifth convolu tional layers
- 11*11 to 7*7, stride 4 to 2 in 1st layer (increasing resolu tion of feature maps)
- Other settings are the same as AlexNet
- Reduce the error by 2%.

M. Zeiler and R. Fergus, [Visualizing and Understanding Convolutional Networks](), ECCV 2014 (Best Paper Award winner)

VGGNet: ILSVRC 2014 2nd place



K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015

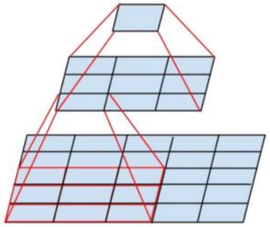
VGGNet: ILSVRC 2014 2^{nd} place

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 **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 **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 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

- Sequence of deeper networks trained progressively
- Large receptive fields replaced by successive layers of 3x3 convolutions (with ReLU in between)



- One 7x7 conv layer with C size of feature maps needs $49C^2$ weights, three 3x3 conv layers need only $27C^2$ weights
- Experimented with 1x1 convolutions

K. Simonyan and A. Zisserman, [Very Deep Convolutional Networks for Large-Scale Image Recognition](), ICLR 2015

VGGNet: ILSVRC 2014 2nd place

- 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

D	E
16 weight layers	19 weight layers
conv3-64	conv3-64
conv3-64	conv3-64
conv3-128	conv3-128
conv3-128	conv3-128
conv3-256	conv3-256
conv3-256	conv3-256
conv3-256	conv3-256
	conv3-256
conv3-512	conv3-512
conv3-512	conv3-512
conv3-512	conv3-512
	conv3-512
conv3-512	conv3-512
conv3-512	conv3-512
conv3-512	conv3-512
	conv3-512

Slide from Jie Chen

VGGNet: ILSVRC 2014 2nd place

- 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

TOTAL memory: 24M * 4 bytes ~= 93MB / image (only forward! ~*2 for bwd)

TOTAL params: 138M parameters

Slide from Jie Chen

VGGNet: ILSVRC 2014 2nd place

Most memory is in early CONV

- 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

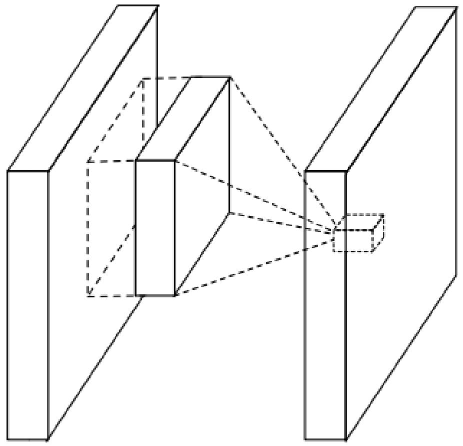
Most params are in late FC

TOTAL memory: 24M * 4 bytes ~= 93MB / image (only forward! ~*2 for bwd)

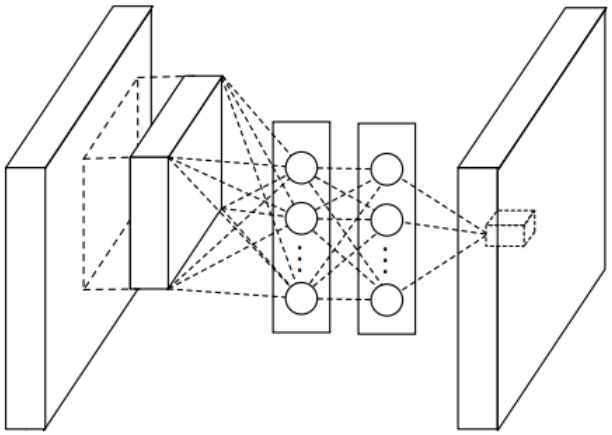
TOTAL params: 138M parameters

Slide from Jie Chen

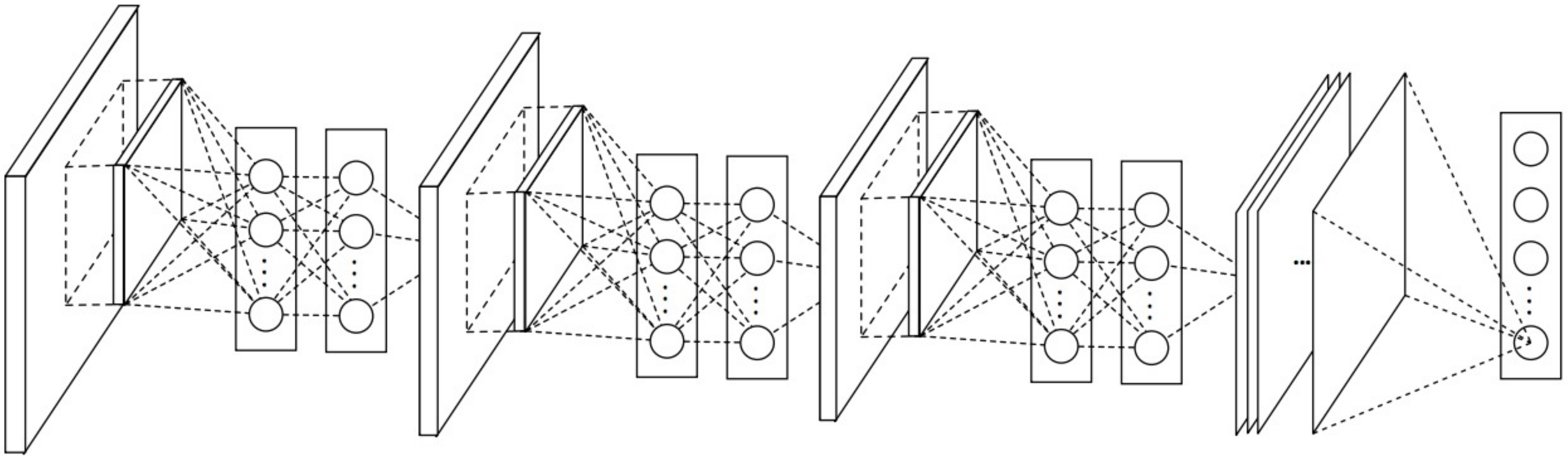
Network in network:NUS-2013



(a) Linear convolution layer

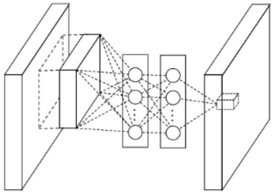


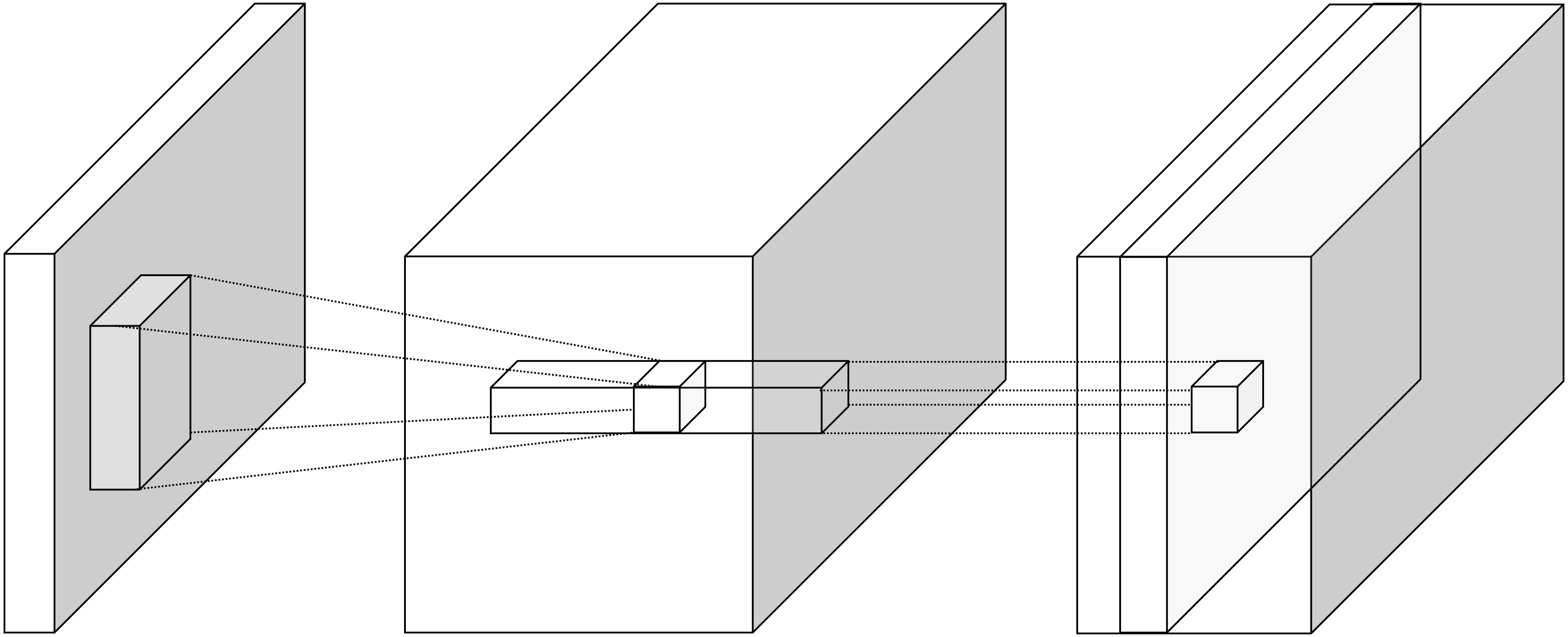
(b) Mlpconv layer



M. Lin, Q. Chen, and S. Yan, [Network in network](), ICLR 2014

1x1 convolutions

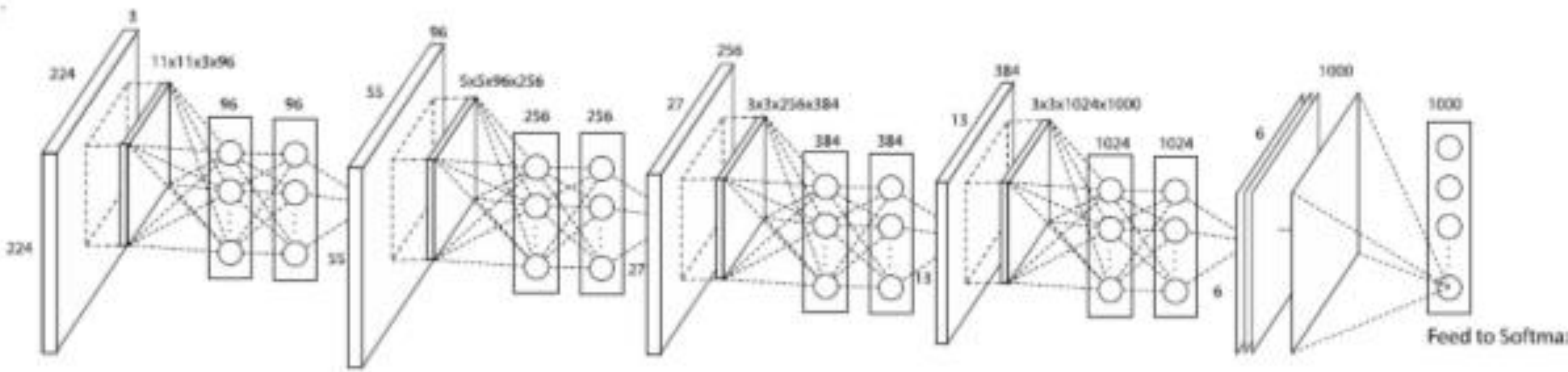




1x1 conv layer

Advantages on NiN

- Remove the two fully connected layers (fc6, fc7) of the AlexNet but add NIN into the AlexNet.



	Parameter Number	Performance	Time to train (GTX Titan)
AlexNet	60 Million (230 Megabytes)	40.7% (Top 1)	8 days
NIN	7.5 Million (29 Megabytes)	39.2% (Top 1)	4 days

Slide from Jie Chen

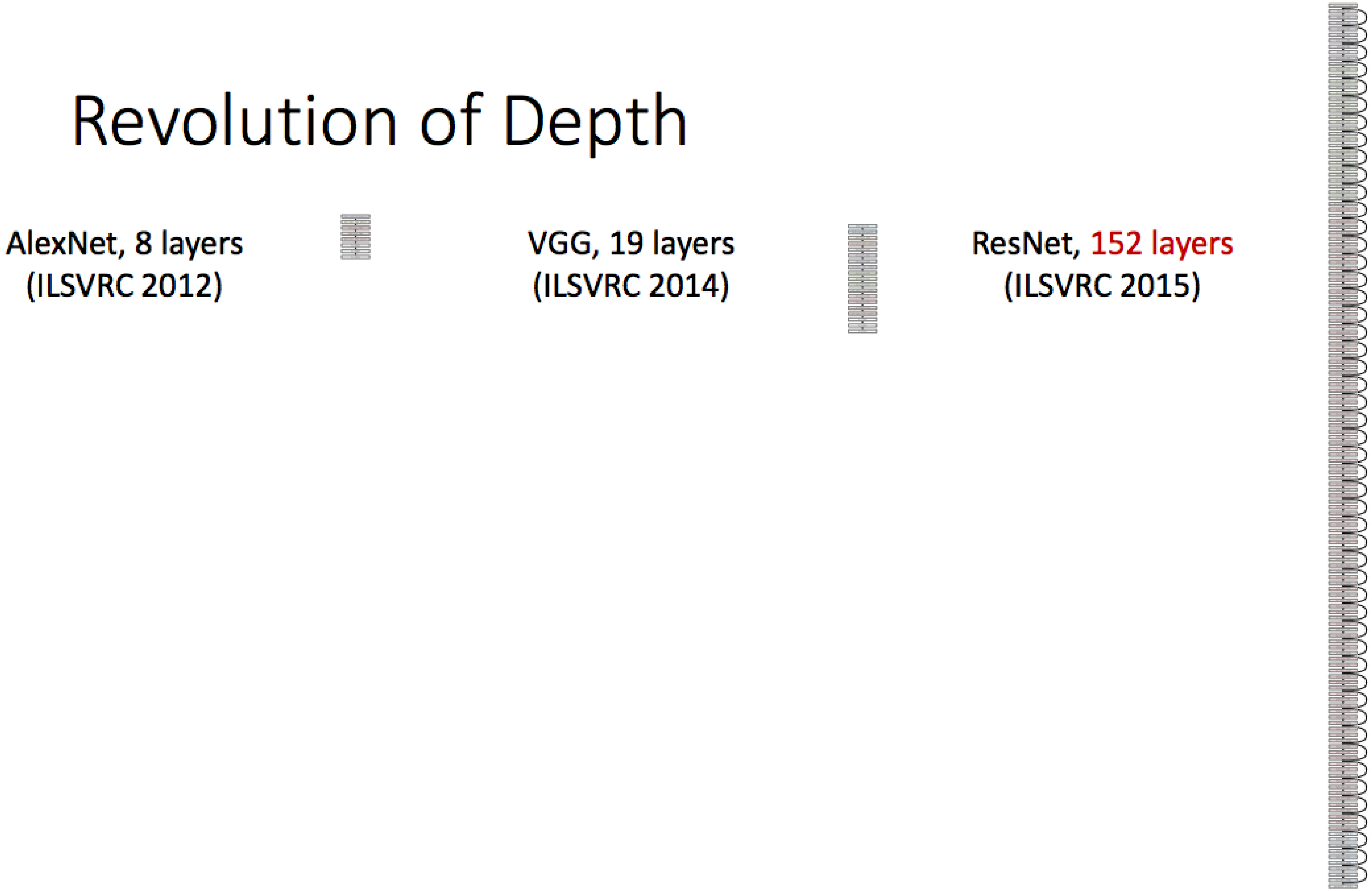
Advantages of using 1x1 Conv.

- Reduce the number of computational parameters
- Add more non-linearity on multiple levels of feature representation without dimensionality reduction
- Mapping on any number of feature maps..(higher to lower and lower to higher dimension)


WE NEED TO GO
DEEPER

“Understanding” ResNet

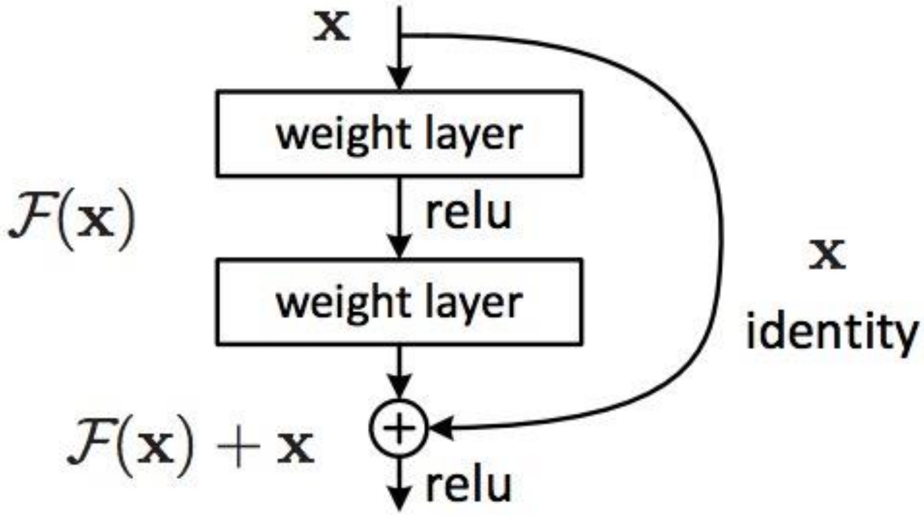
ResNet: ILSVRC 2015 winner



Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, [Deep Residual Learning for Image Recognition](), CVPR 2016

ResNet

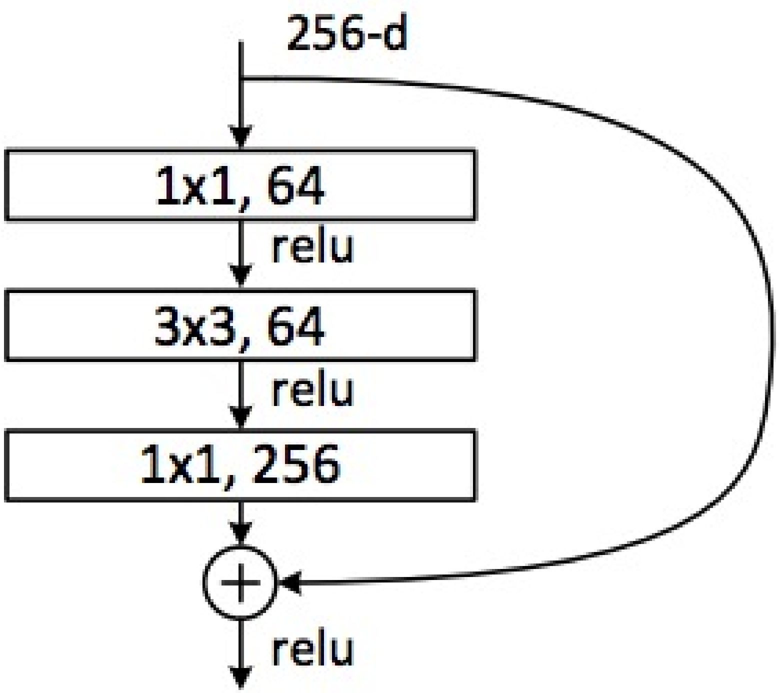
- The residual module
  - Introduce *skip* or *shortcut* connections (existing before in various forms in literature)
  - Make it easy for network layers to represent the identity mapping
  - For some reason, need to skip at least two layers



Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, CVPR 2016 (Best Paper)

ResNet

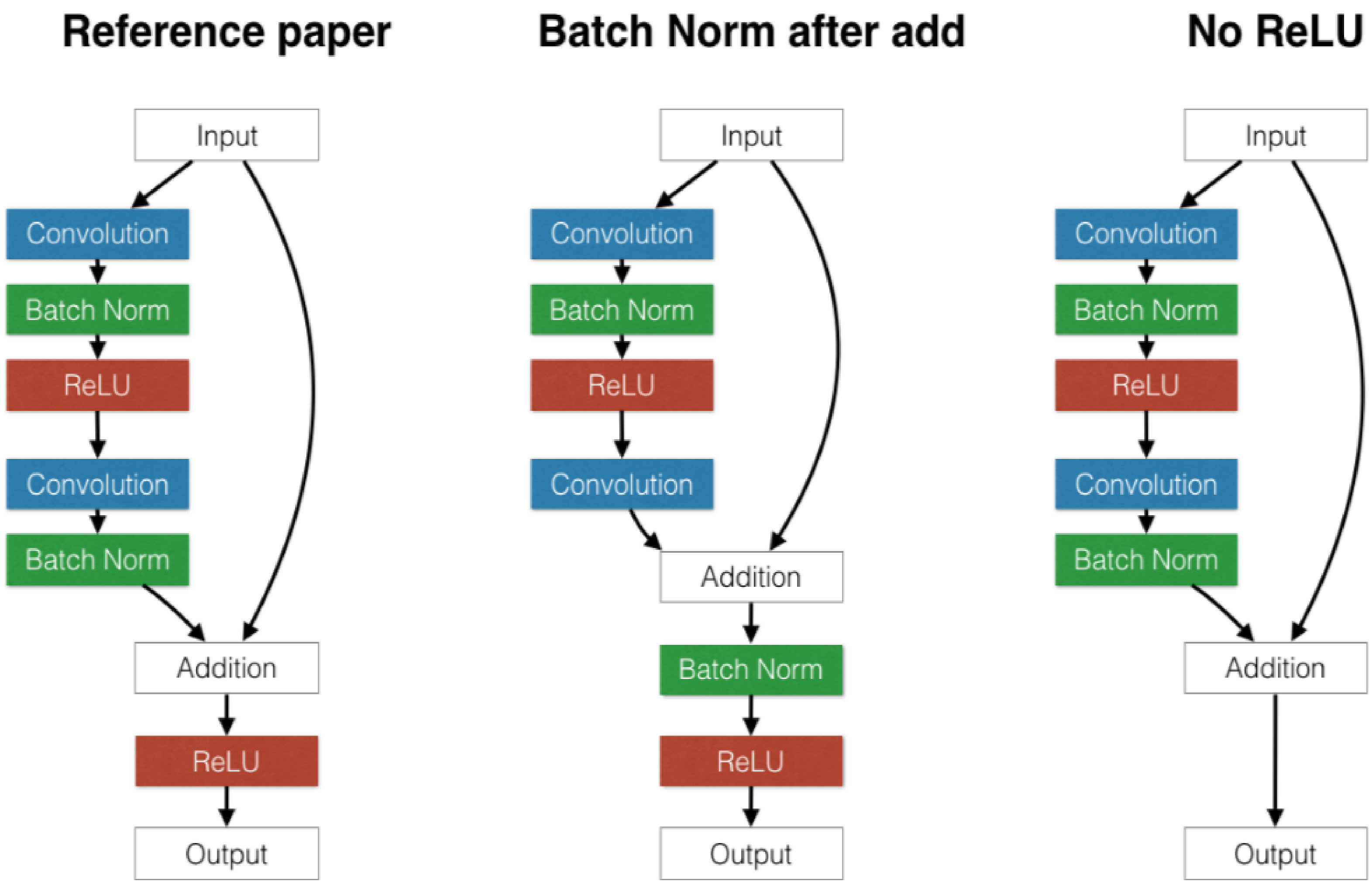
Deeper residual module (bottleneck)



- Directly performing 3x3 convolutions with 256 feature maps at input and output: 256 x 256 x 3 x 3 ~ 600K operations
- Using 1x1 convolutions to reduce 256 to 64 feature maps, followed by 3x3 convolutions, followed by 1x1 convolutions to expand back to 256 maps:
  256 x 64 x 1 x 1 ~ 16K
  64 x 64 x 3 x 3 ~ 36K
  64 x 256 x 1 x 1 ~ 16K
  Total: ~70K

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, [Deep Residual Learning for Image Recognition](), CVPR 2016 (Best Paper)

ResNet



Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, CVPR 2016

ResNet

- Architectures for ImageNet problem:

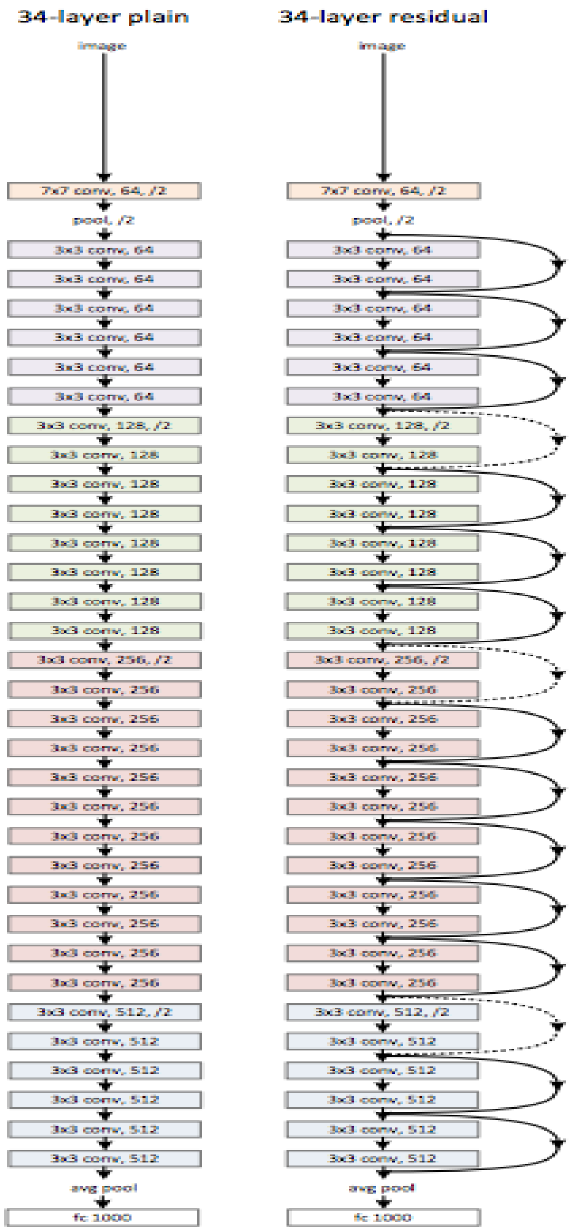
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, CVPR 2016 (Best Paper)

ResNet

- The architecture of the plain and residual networks were identical except for the skip connections

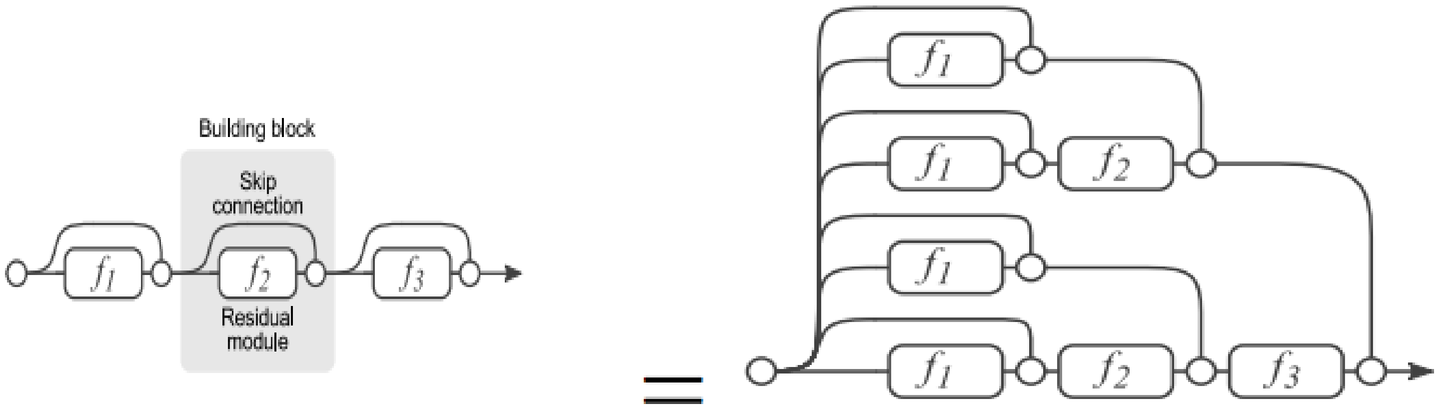
- **Result: Going deeper makes things better!**



Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, Deep Residual Learning for Image Recognition, CVPR 2016

Why do ResNets work?

- ResNets seem to work because they **facilitate the training of deeper networks**
- Are surprisingly robust to layers being dropped or reordered
- Implicitly ensembling shallower networks
- Able to learn unrolled iterative refinements

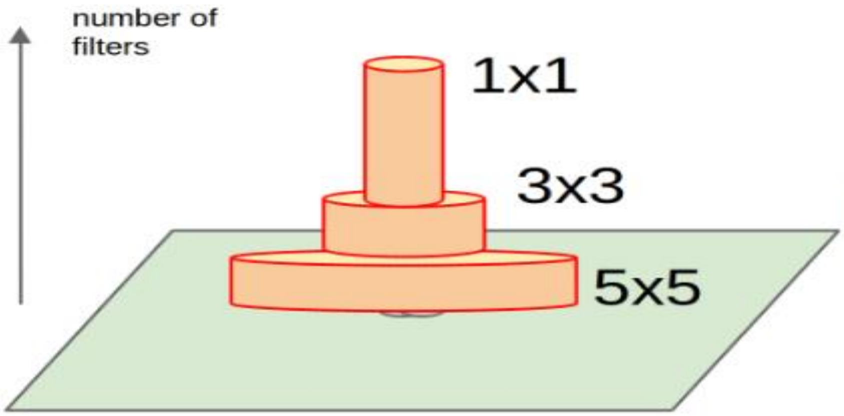


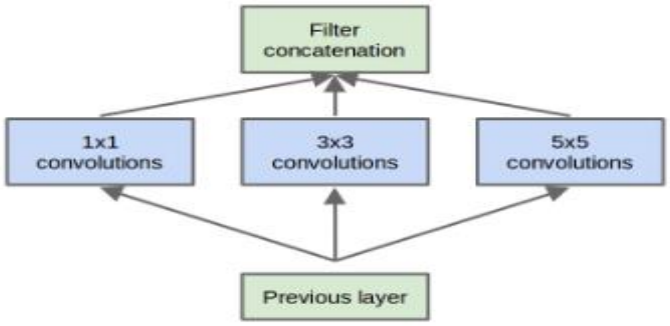
(a) Conventional 3-block residual network (b) Unraveled view of (a)

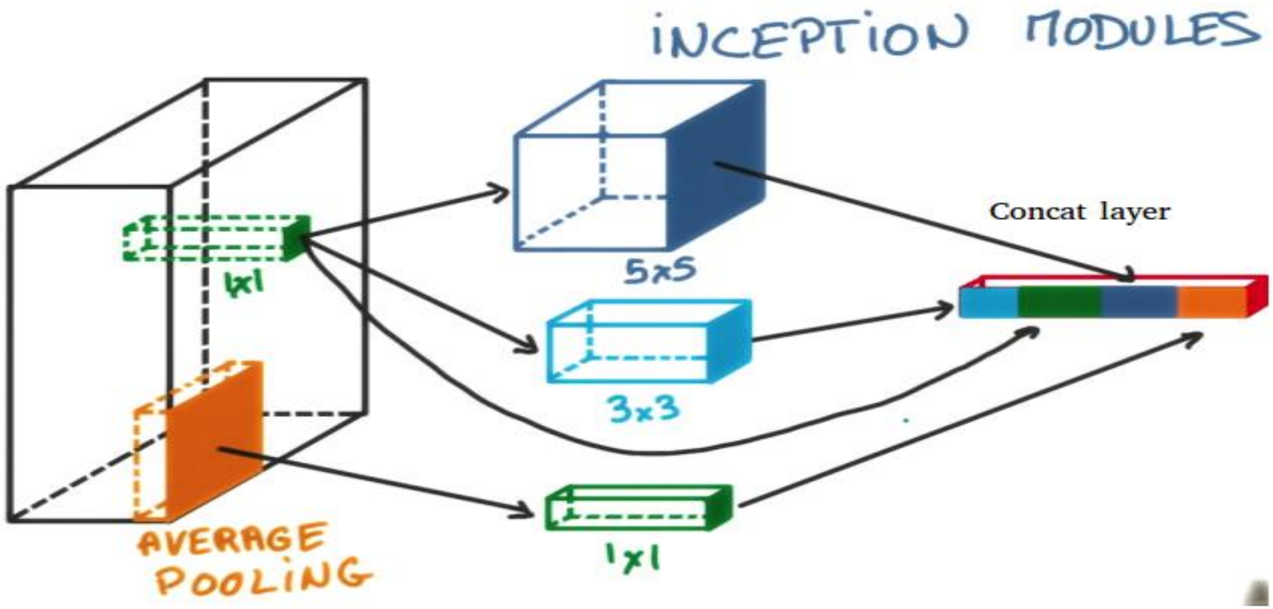
Andreas Veit, Michael Wilber, Serge Belongie, Residual Networks Behave Like Ensembles of Relatively Shallow Networks, arxiv 2016

“Understanding” Inception Module

Inception Module



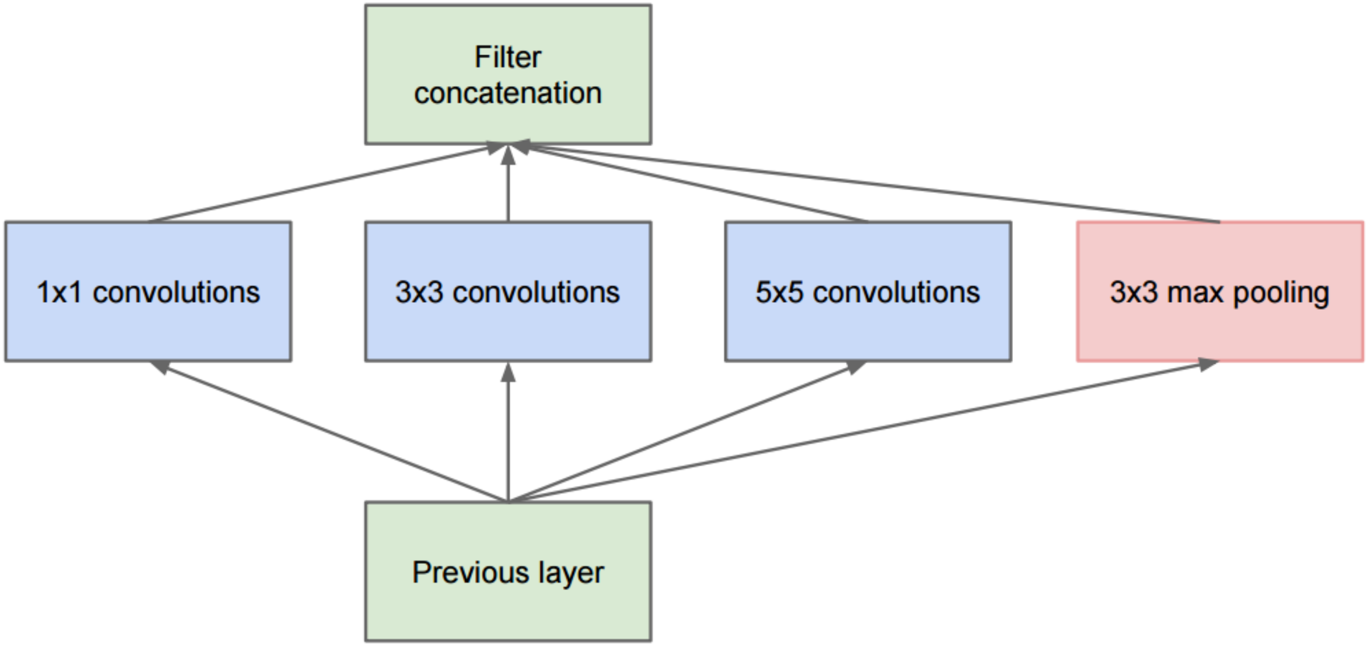




Szegedy, Christian, et al. "Rethinking the inception architecture for computer vision." *arXiv preprint arXiv:1512.00567* (2015).

Inception Module

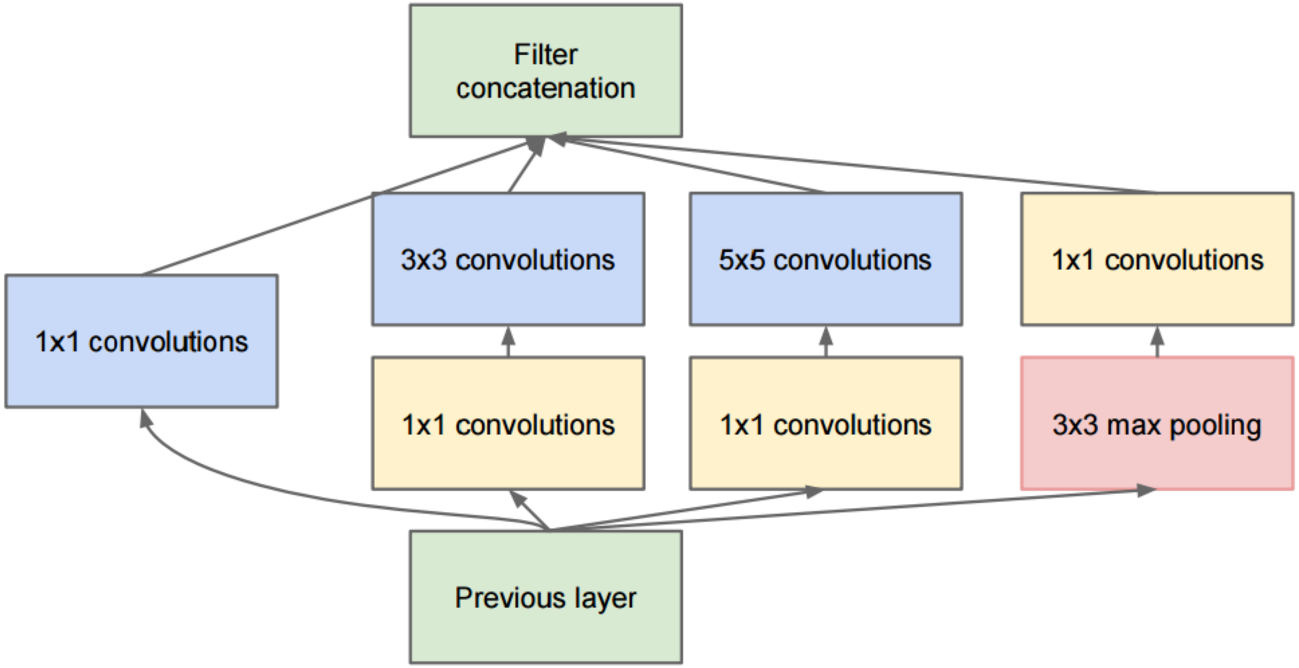
- Parallel paths with different receptive field sizes and operations are meant to capture sparse patterns of correlations in the stack of feature maps



C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015

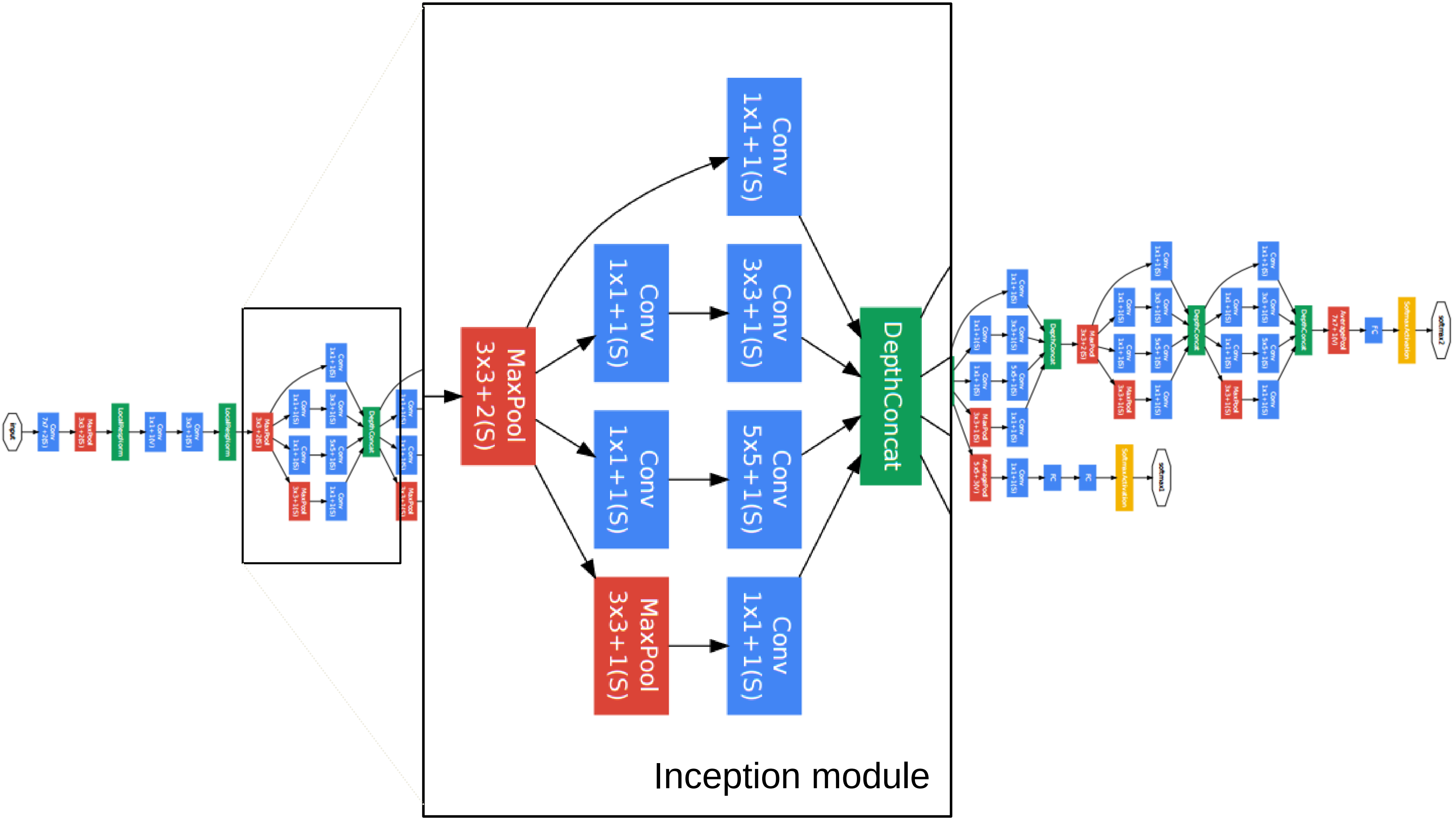
Inception Module

- Parallel paths with different receptive field sizes and operations are meant to capture sparse patterns of correlations in the stack of feature maps
- Use 1x1 convolutions for dimensionality reduction before expensive convolutions



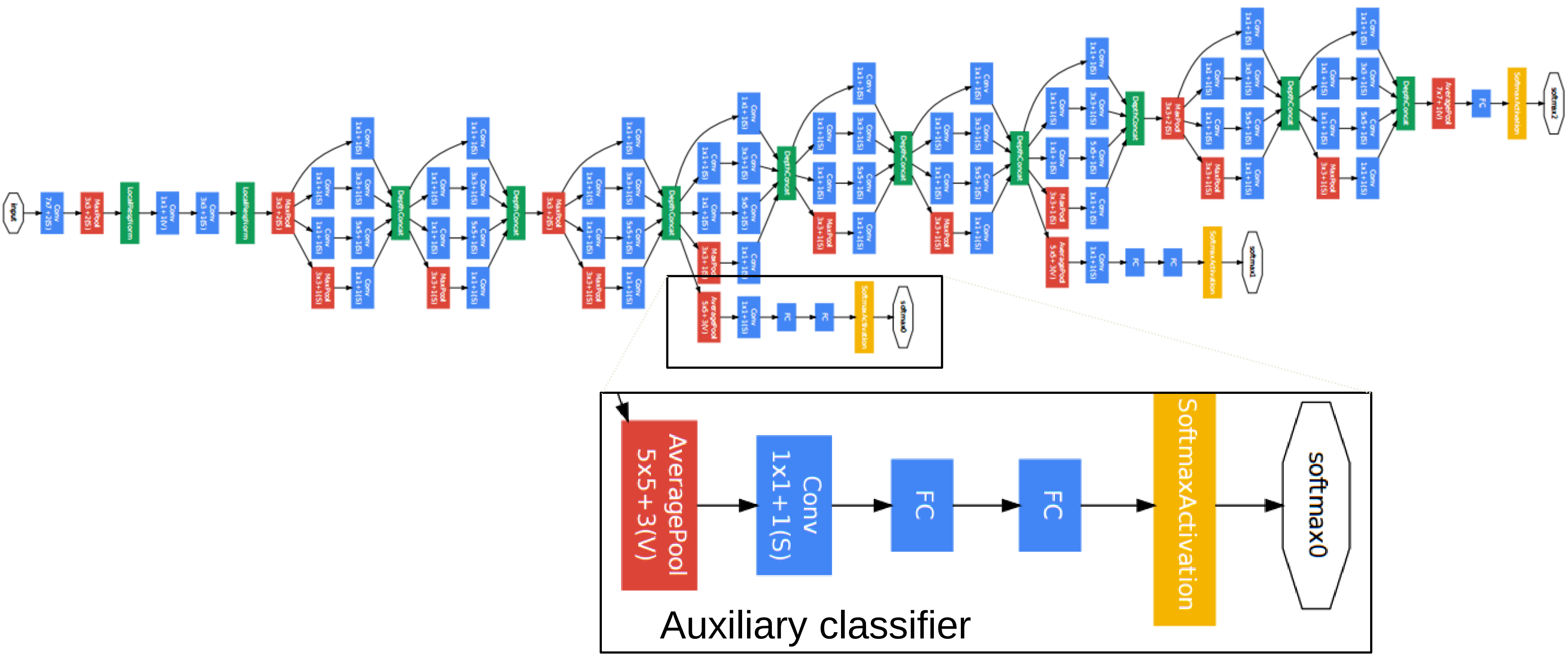
C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015

GoogLeNet



C. Szegedy et al., Going deeper with convolutions, CVPR 2015

GoogLeNet



C. Szegedy et al., [Going deeper with convolutions](), CVPR 2015

GoogLeNet

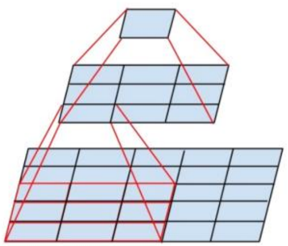
- An alternative view:

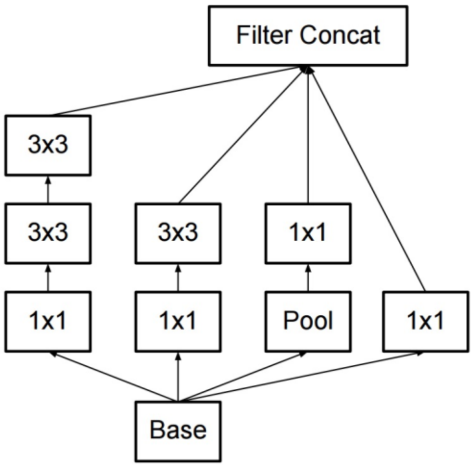
type	patch size/ stride	output size	depth	#1×1	#3×3 reduce	#3×3	#5×5 reduce	#5×5	pool proj	params	ops
convolution	7×7/2	112×112×64	1							2.7K	34M
max pool	3×3/2	56×56×64	0								
convolution	3×3/1	56×56×192	2		64	192				112K	360M
max pool	3×3/2	28×28×192	0								
inception (3a)		28×28×256	2	64	96	128	16	32	32	159K	128M
inception (3b)		28×28×480	2	128	128	192	32	96	64	380K	304M
max pool	3×3/2	14×14×480	0								
inception (4a)		14×14×512	2	192	96	208	16	48	64	364K	73M
inception (4b)		14×14×512	2	160	112	224	24	64	64	437K	88M
inception (4c)		14×14×512	2	128	128	256	24	64	64	463K	100M
inception (4d)		14×14×528	2	112	144	288	32	64	64	580K	119M
inception (4e)		14×14×832	2	256	160	320	32	128	128	840K	170M
max pool	3×3/2	7×7×832	0								
inception (5a)		7×7×832	2	256	160	320	32	128	128	1072K	54M
inception (5b)		7×7×1024	2	384	192	384	48	128	128	1388K	71M
avg pool	7×7/1	1×1×1024	0								
dropout (40%)		1×1×1024	0								
linear		1×1×1000	1							1000K	1M
softmax		1×1×1000	0								

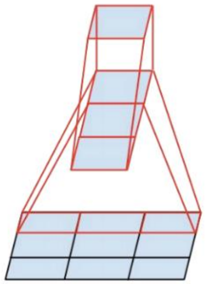
C. Szegedy et al., Going deeper with convolutions, CVPR 2015

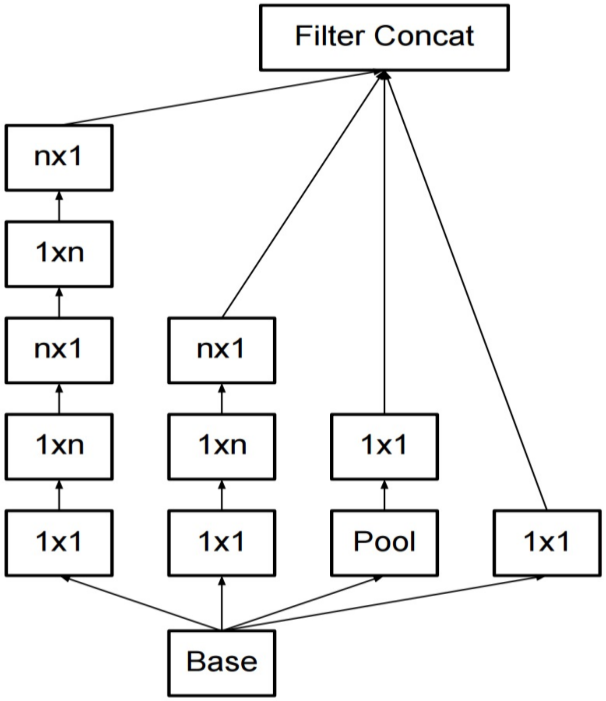
Inception v2, v3

- Regularize training with batch normalization, reducing importance of auxiliary classifiers
- More variants of inception modules with aggressive factorization of filters





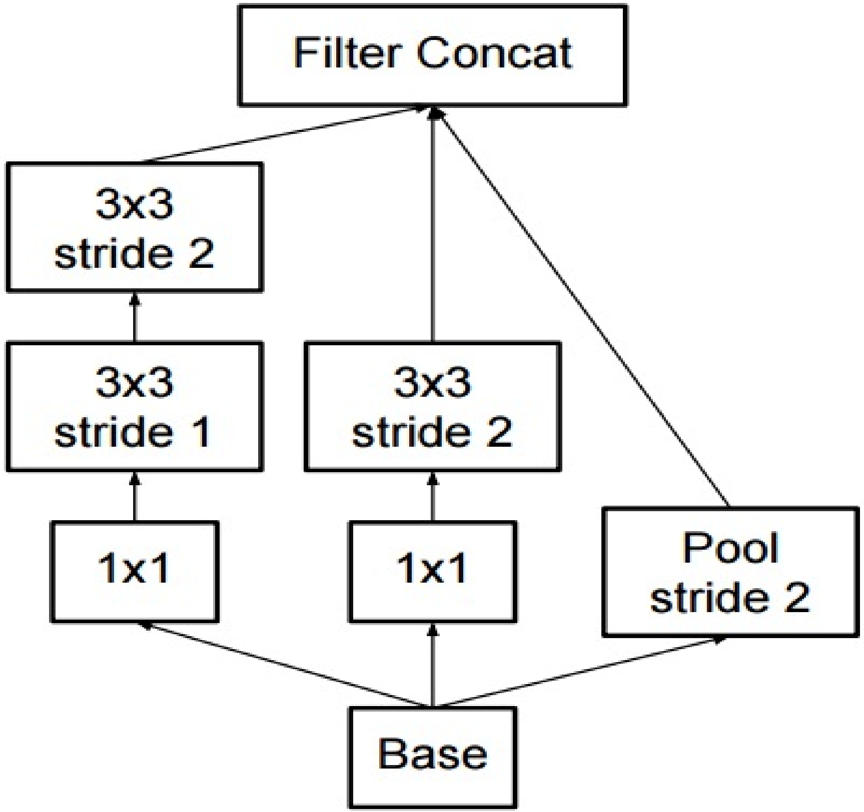




C. Szegedy et al., Rethinking the inception architecture for computer vision, CVPR 2016

Inception v2, v3

- Increase the number of feature maps while decreasing spatial resolution (pooling)



C. Szegedy et al., [Rethinking the inception architecture for computer vision](), CVPR 2016

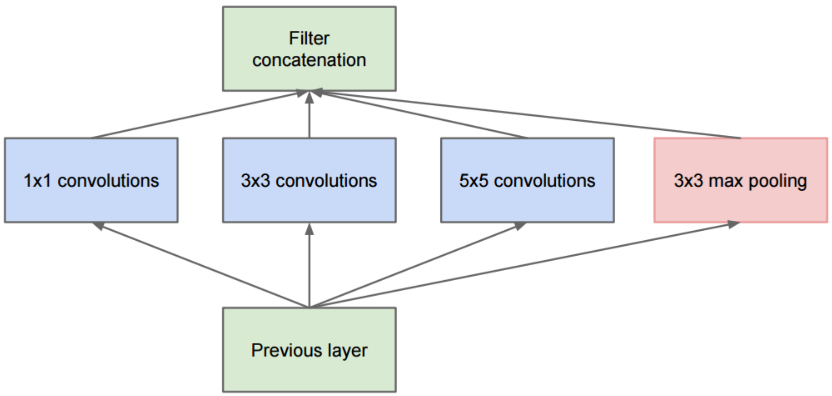
What’s new?

- **Batch Normalization (BN)** is used
- 1x1 convolution for dimensionality (z-axis) reduction
- Average pooling introduce in Inception module
- Instead of 5x5 filter uses dual 3x3 filter.

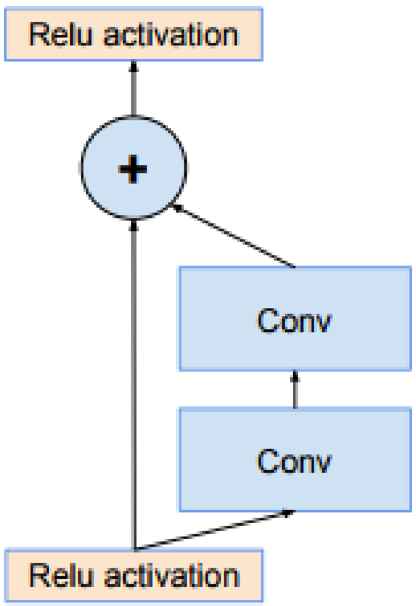
Advantages

- Reduce 90% computational parameters of AleNet
- Multiple receptive field for better stack of feature representation.
- Achieve excellent performance using limited number of parameters.

Inception-ResNet



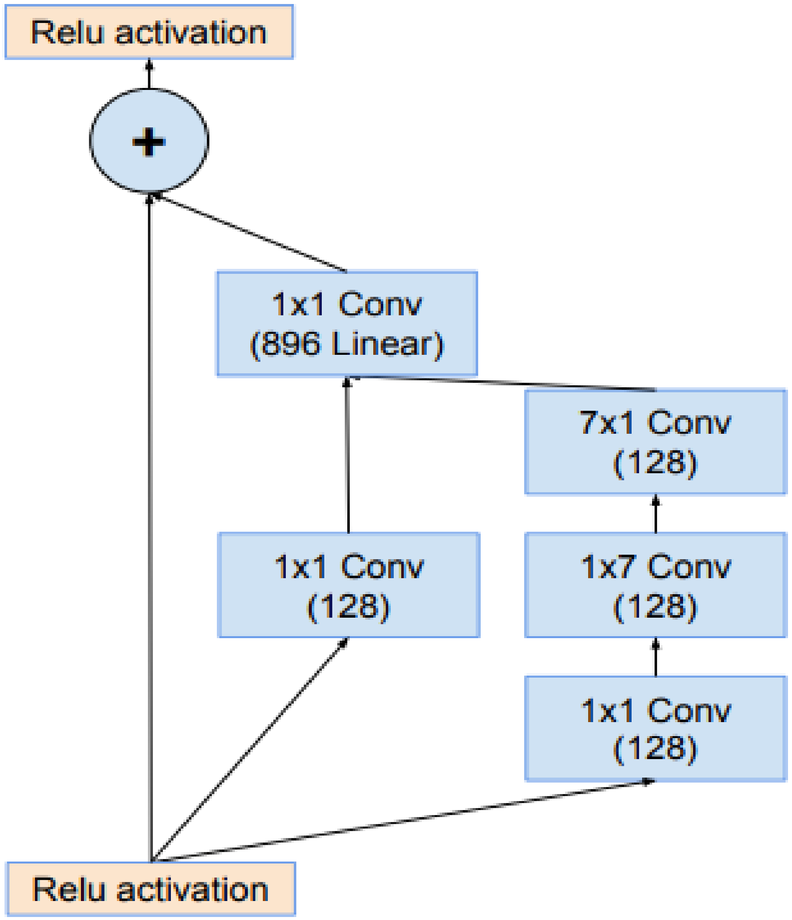
Inception connections as introduced in C. Szegedy et al



Residual connections as introduced in He et al

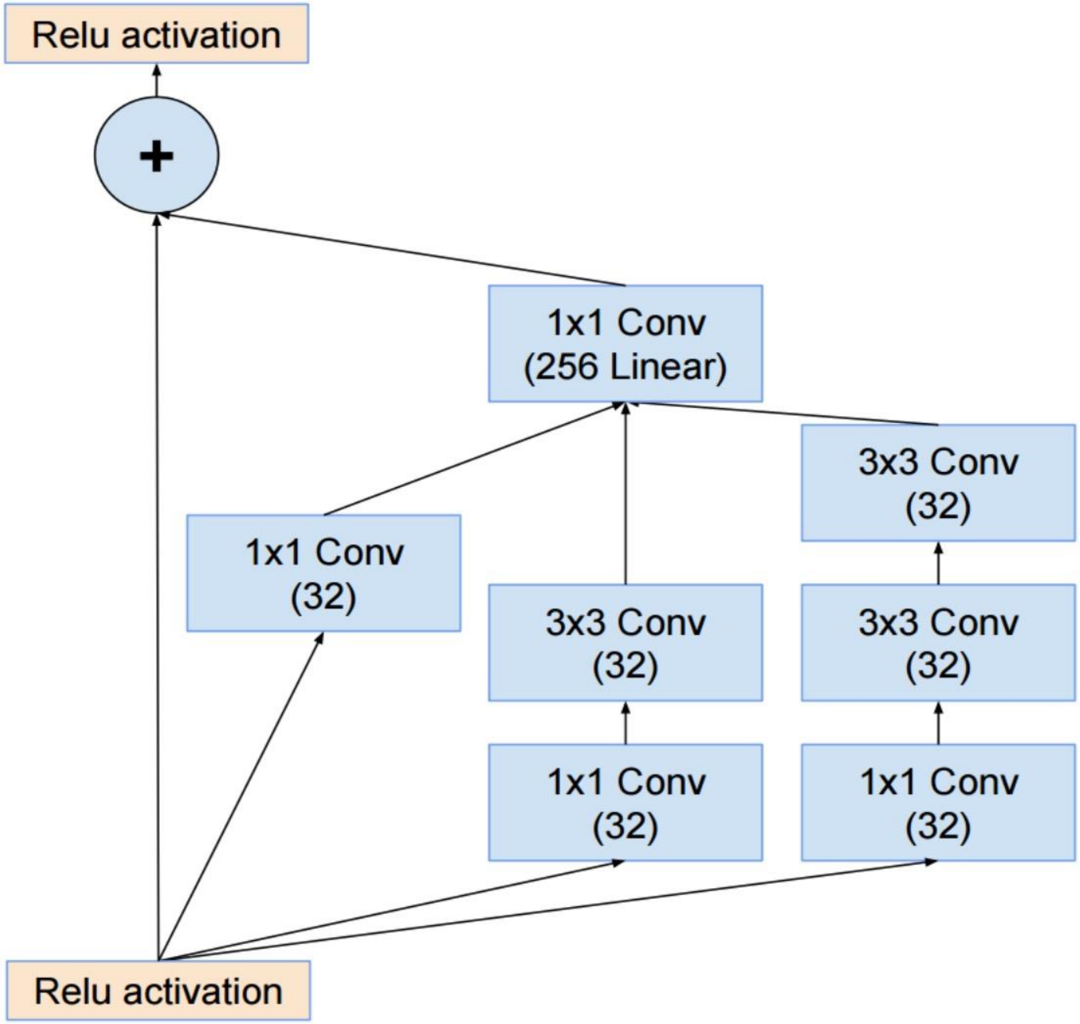
Szegedy, Christian, et al. "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning." *AAAI*. 2017.

Inception-Res module



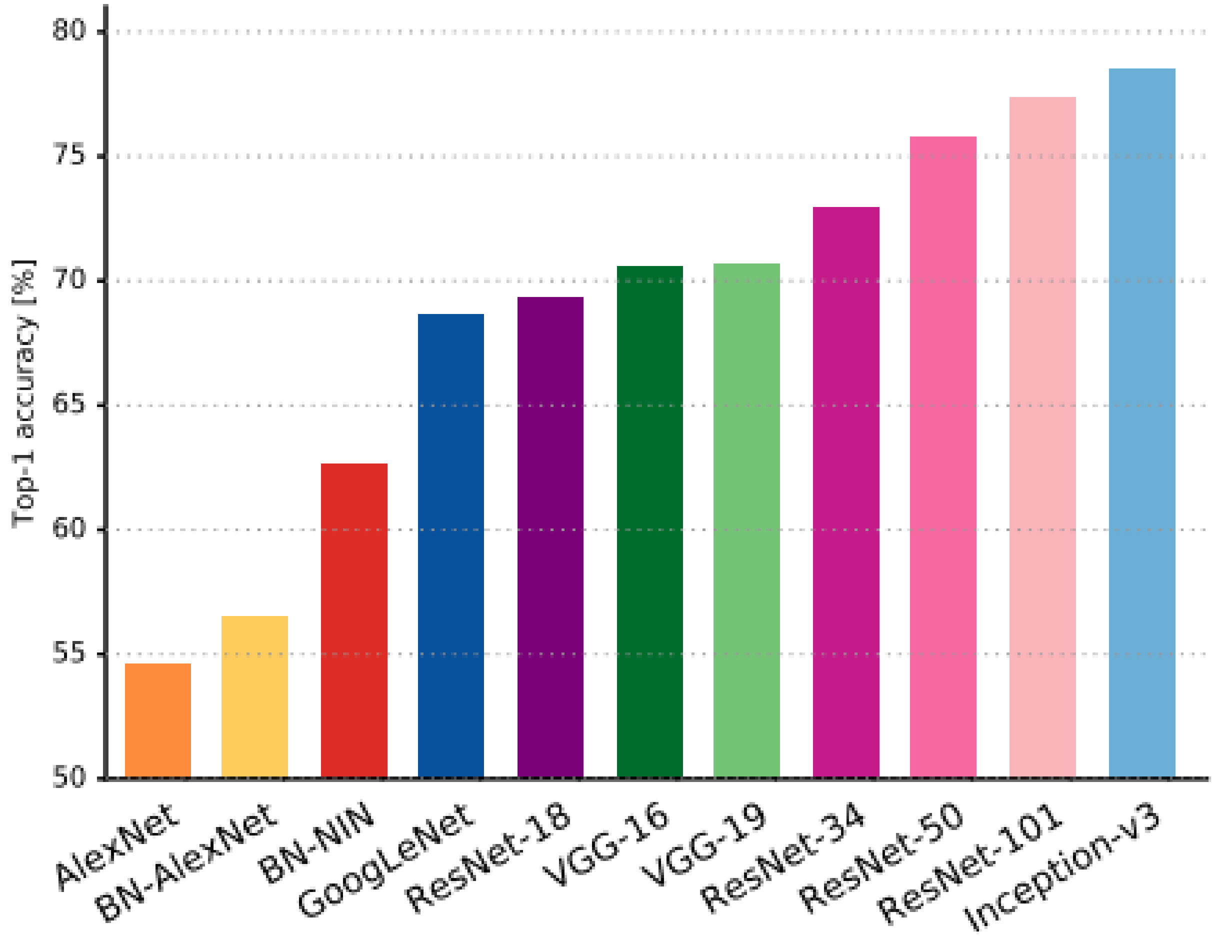
C. Szegedy et al., [Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning](), arXiv 2016

Inception v4



C. Szegedy et al., Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning, arXiv 2016

We're focusing on ImageNet



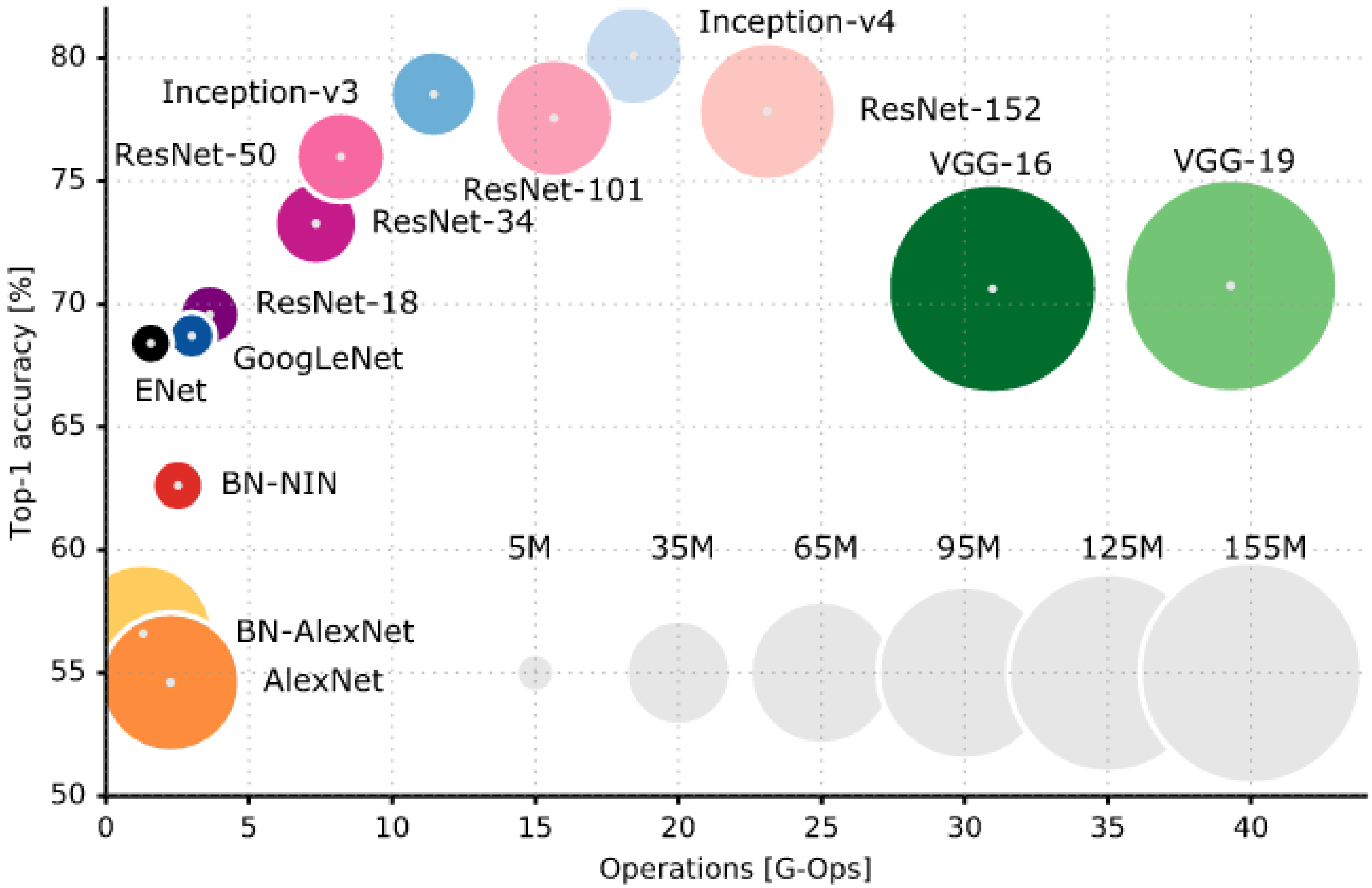
Alfredo Canziani & Eugenio Culurciello, [An Analysis of Deep Neural Network Models for Practical Applications](), arXiv 2016

Summary: ILSVRC 2012-2015

Team	Year	Place	Error (top-5)	External data
SuperVision – Toronto (AlexNet, 7 layers)	2012	-	16.4%	no
SuperVision	2012	1st	15.3%	ImageNet 22k
Clarifai – NYU (7 layers)	2013	-	11.7%	no
Clarifai	2013	1st	11.2%	ImageNet 22k
VGG – Oxford (16 layers)	2014	2nd	7.32%	no
GoogLeNet (19 layers)	2014	1st	6.67%	no
ResNet (152 layers)	2015	1st	3.57%	
Human expert*			5.1%	

http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/

Accuracy vs. efficiency



https://culurciello.github.io/tech/2016/06/04/nets.html

Summary

- Introduce different CNN architectures and advantages
- General design principles of CNN models

- What's next?
  - CNN Architectures ++
    - Hybrid Networks
    - DenseNet
    - FactralNet

References

- https://culurciello.github.io/tech/2016/06/04/nets.html
- Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86(11): 2278–2324, 1998.
- A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012
- M. Zeiler and R. Fergus, Visualizing and Understanding Convolutional Networks, ECCV 2014
- K. Simonyan and A. Zisserman, Very Deep Convolutional Networks for Large-Scale Image Recognition, ICLR 2015
- M. Lin, Q. Chen, and S. Yan, Network in network, ICLR 2014
- C. Szegedy et al., Going deeper with convolutions, CVPR 2015
- C. Szegedy et al., Rethinking the inception architecture for computer vision, CVPR 2016
- K. He, X. Zhang, S. Ren, and J. Sun, Deep Residual Learning for Image Recognition, CVPR 2016