CNN Architectures

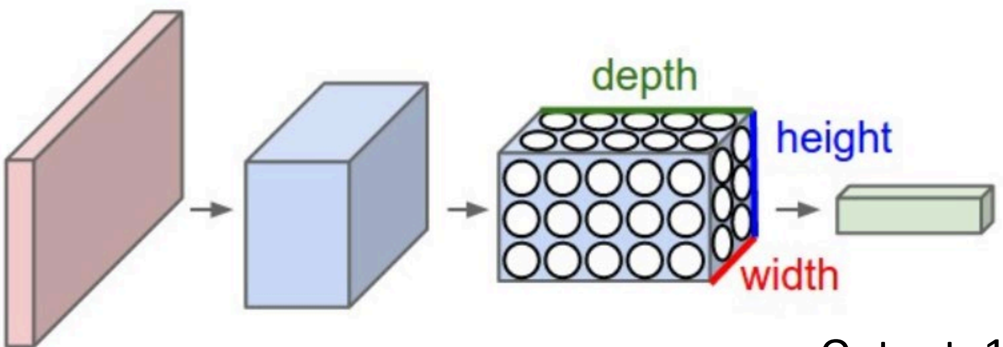
ISTD 50.035

Computer Vision

Acknowledgement: Some images are from various sources: UCF, Stanford cs231n, etc.

CNN

- 3D volumes of neurons



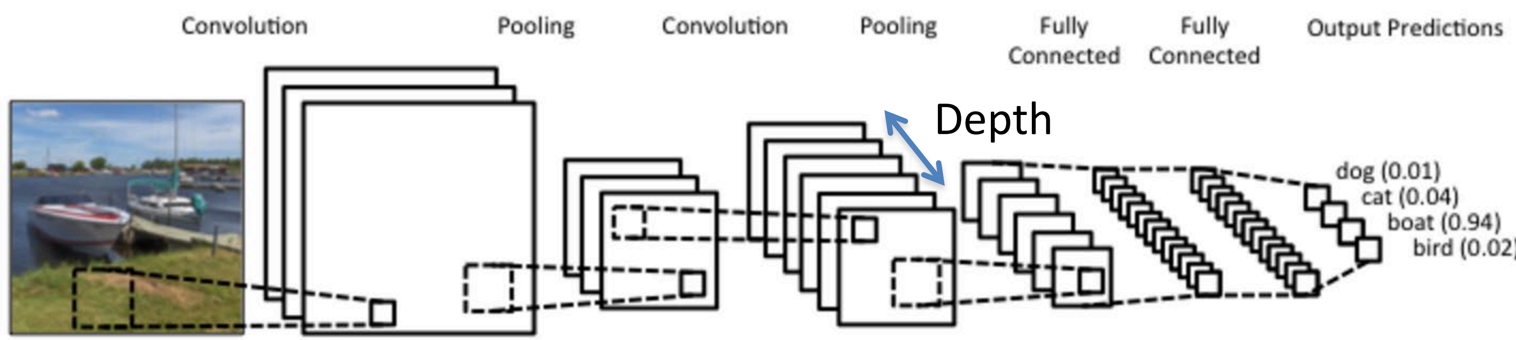
Input: width x height x numChannel

Output: 1 x 1 x numClass

- Stack of
  - Convolutional layer
  - Fully connected layer
  - Pooling layer

CNN

- One example



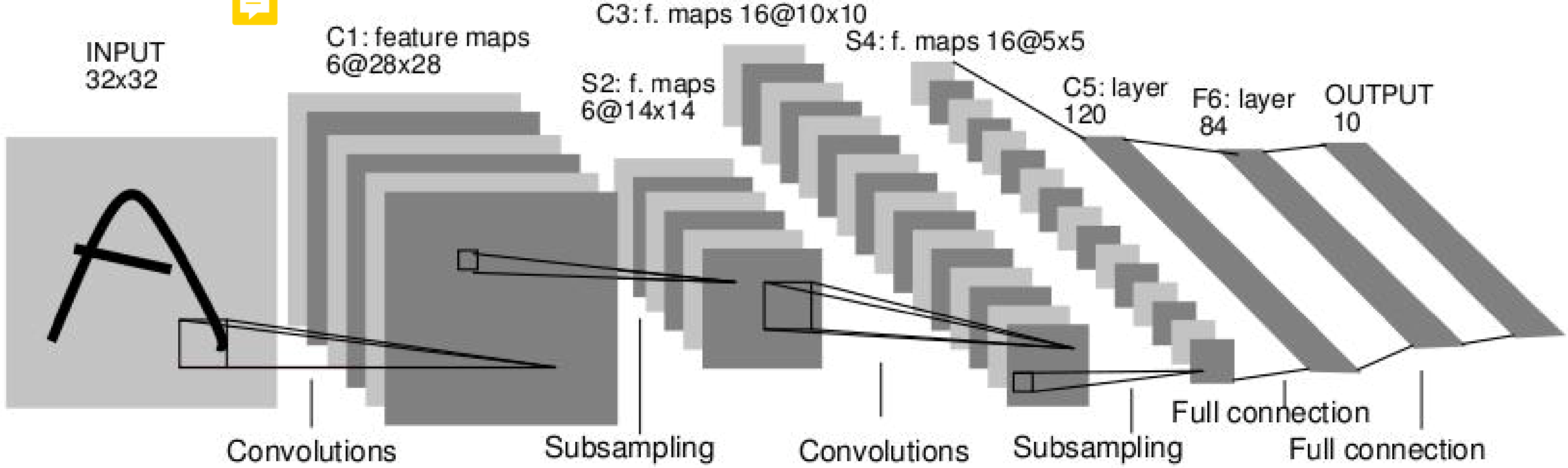
Input:
width x height x numChannel

Output:
1 x 1 x numClass

- Stack of
  - Convolutional layer
  - Fully connected layer
  - Pooling layer

CNN Architecture

- LeNet for character recognition

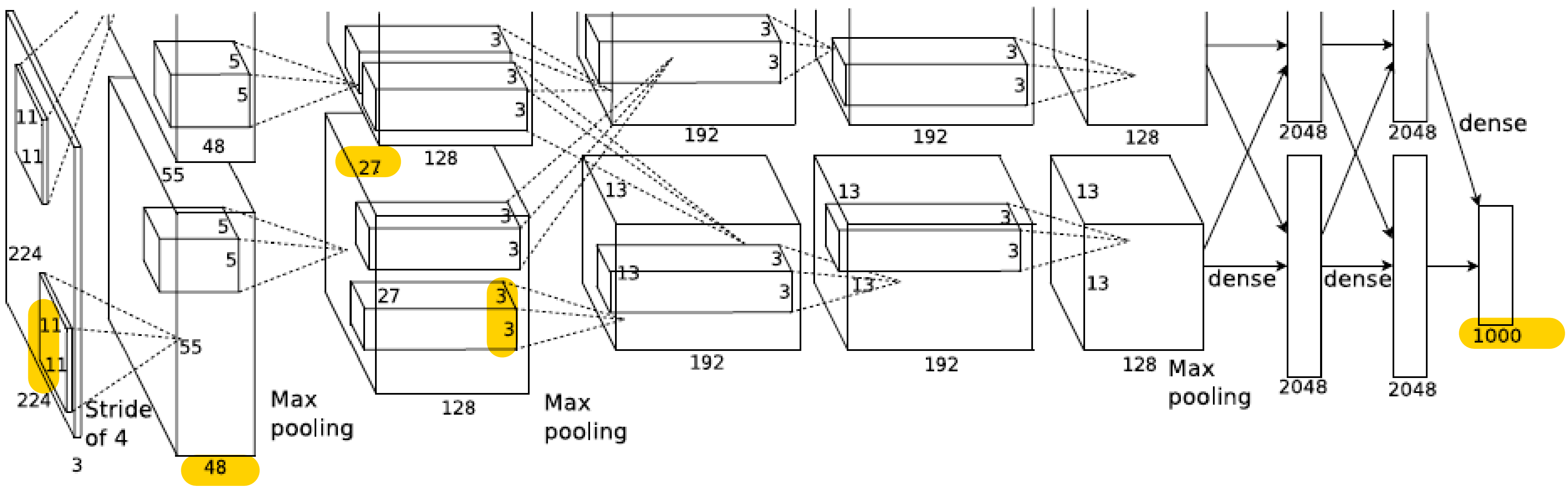


-Average pooling
-Sigmoid or tanh nonlinearity
-Trained on MNIST digit dataset (60K training examples)

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

CNN Architecture

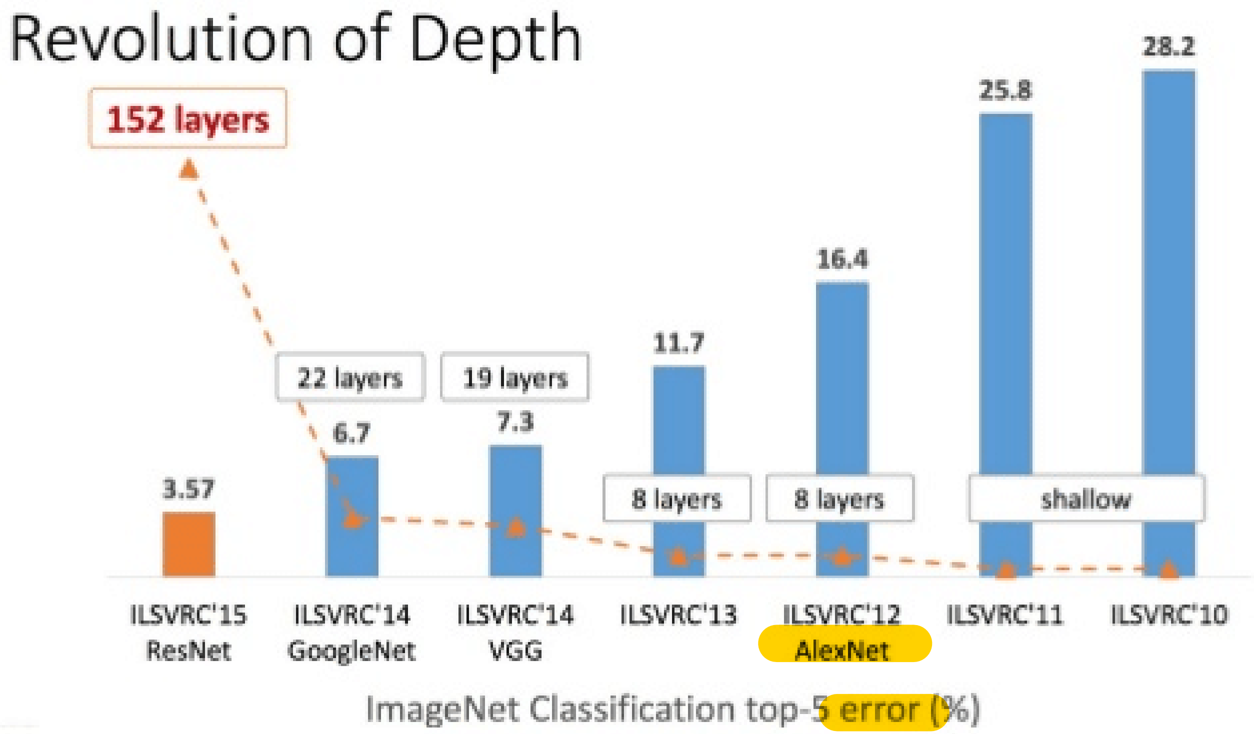
- AlexNet for image recognition



-8 layers, 650K neurons, 60M parameters
-Max pooling, ReLU nonlinearity
-1.2M training images of 1000 classes

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

CNN Architecture



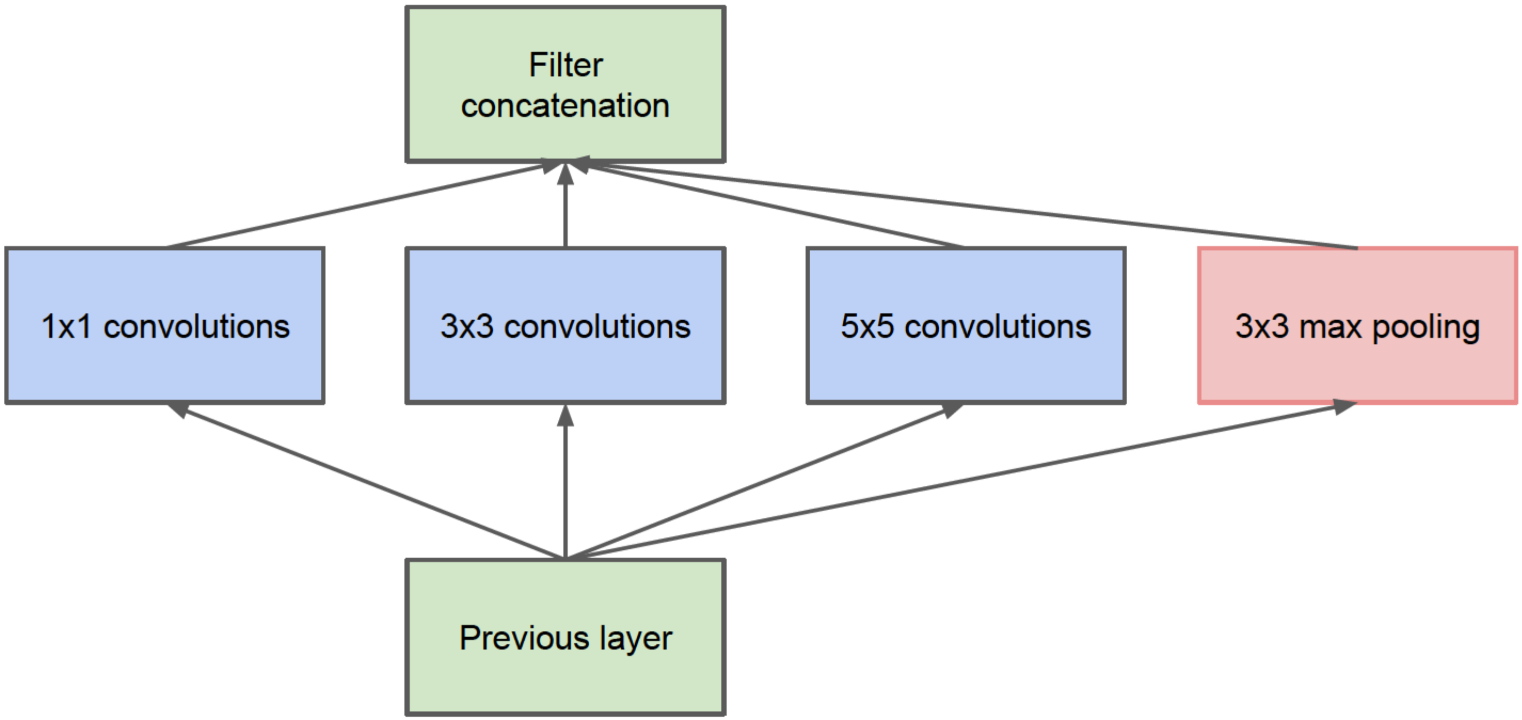
Inception module and GoogLeNet

- Filter size: hyperparameter
- What is the right filter size?
- Inception module
  - Use filters of different size in the same layer
  - Concatenate all the filter results as output
  - Use 1x1 convolution to reduce complexity
  - Increase width and depth of the network
  - All convolutions use ReLU, including 1x1 convolution for reduction / projection

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, "Going Deeper with Convolutions," CVPR 2015

Inception module and GoogLeNet

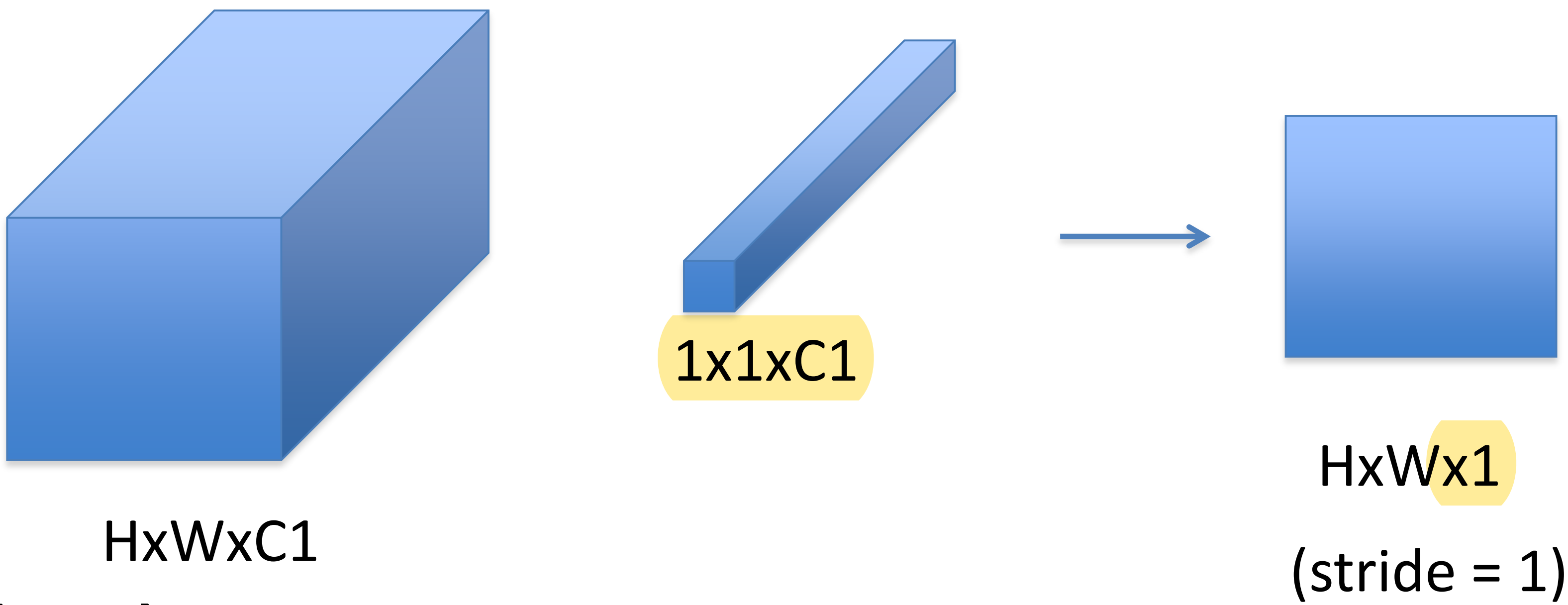
- Inception module, naïve version



Computation expensive: 28x28x192 input, need 5x5x192 filtering

Inception module and GoogLeNet

- 1x1 convolution for dimensionality reduction: reduce number of channels



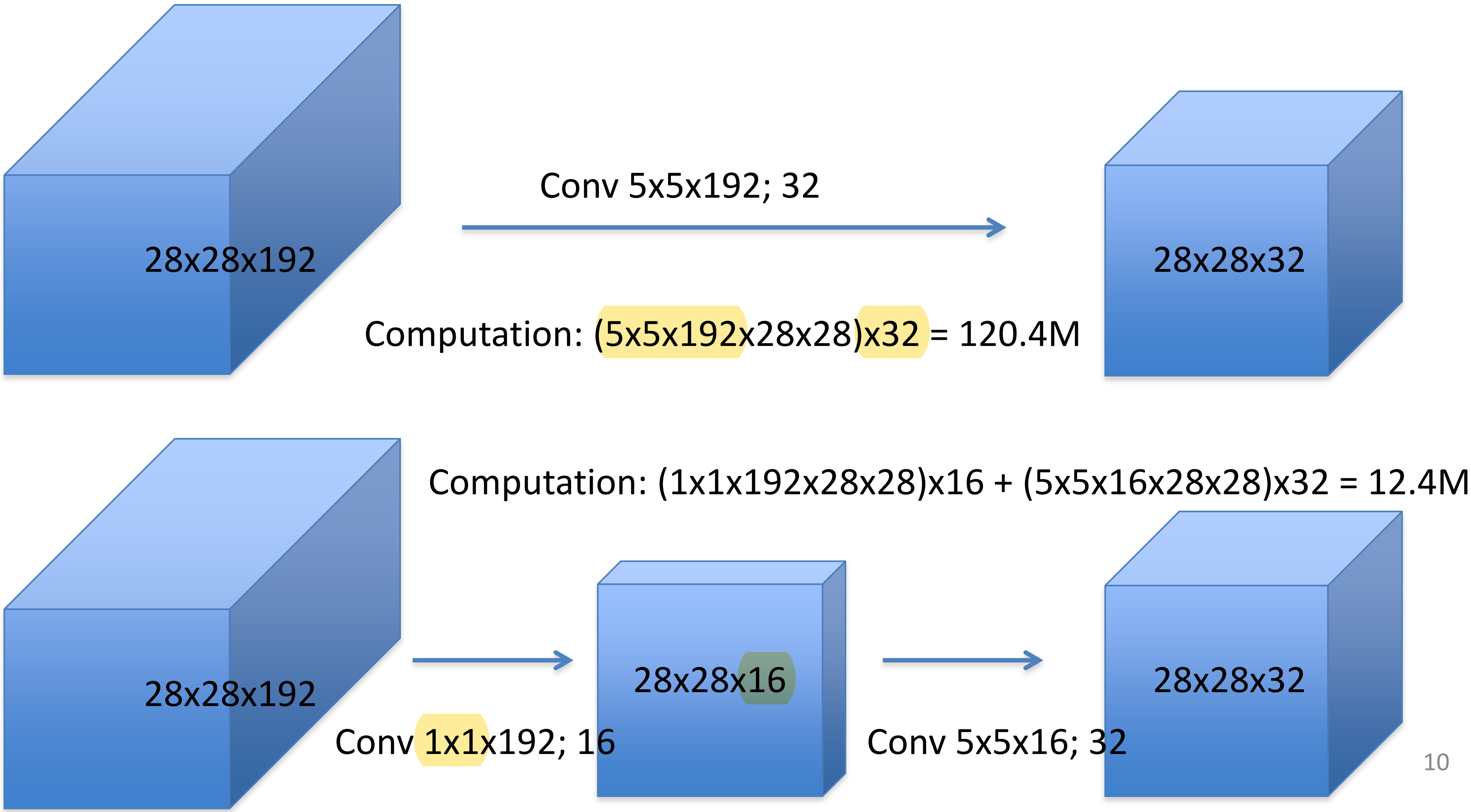
[Lin et al.; 2014]

-With C2 filters (1x1xC1 each), output HxWxC2; usually C2 < C1

-Followed by non-linear as in ordinary convolution

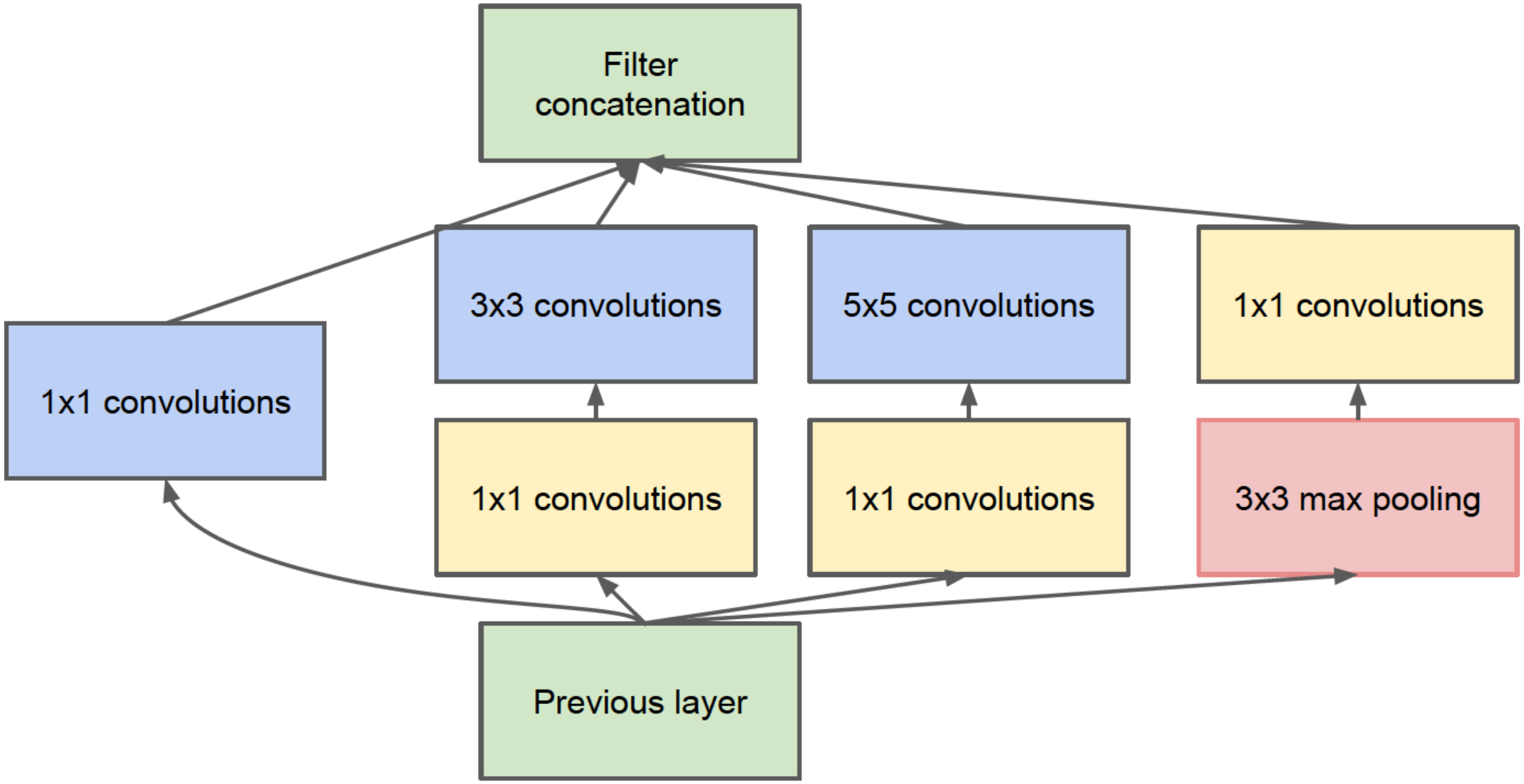
Inception module and GoogLeNet

- 1x1 convolution for dimensionality reduction



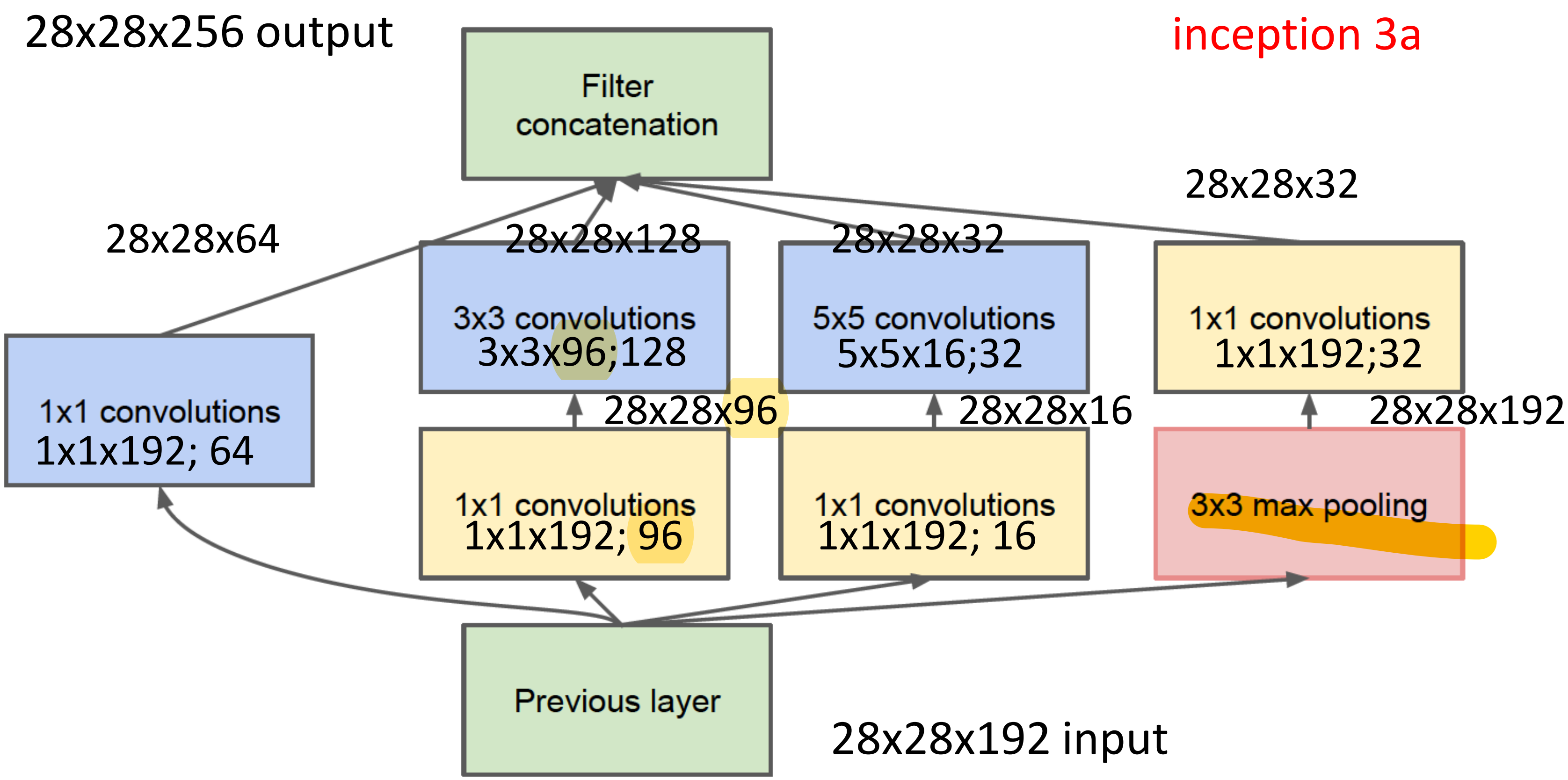
Inception module and GoogLeNet

- Inception module with dimensionality reduction



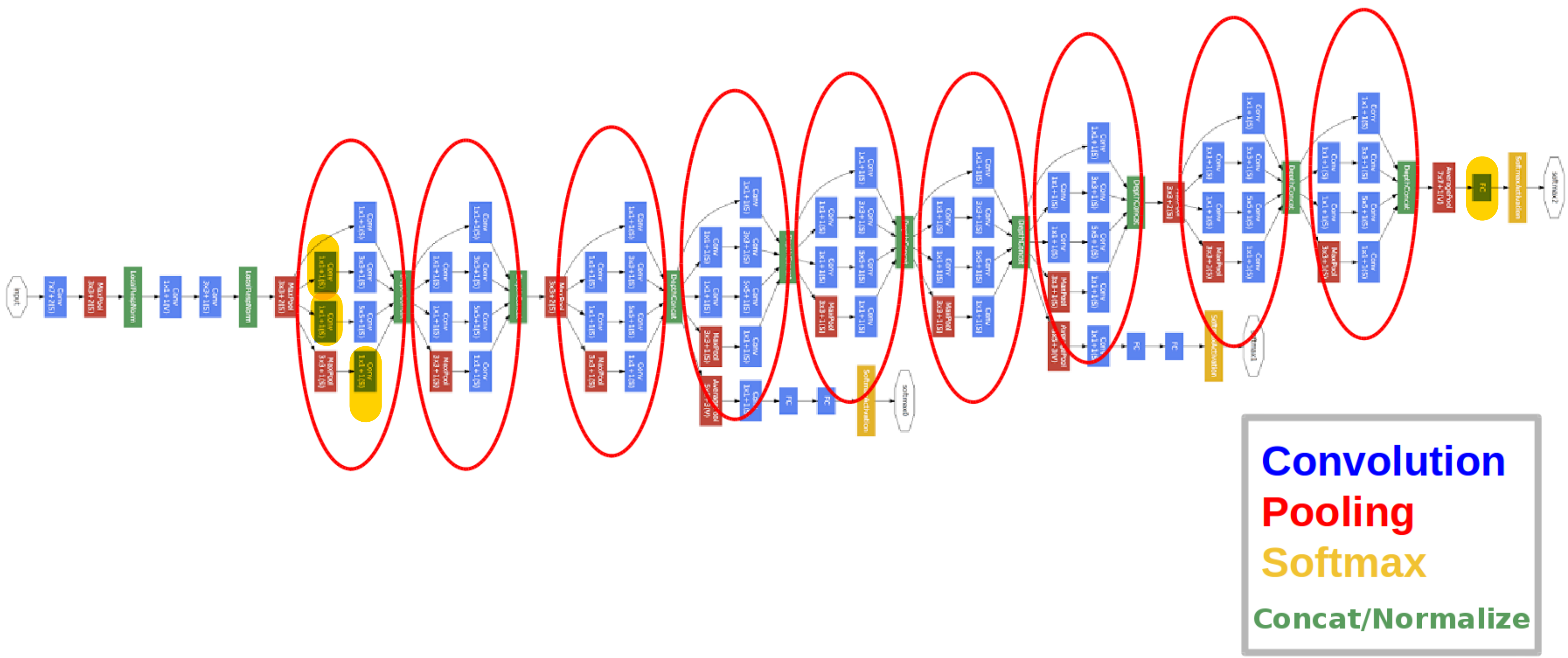
Inception module and GoogLeNet

- Inception module with dimensionality reduction



Inception module and GoogLeNet

- GoogLeNet



Inception module and GoogLeNet

Total: 27 layers (dropout, softmax not count)

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								

Table 1: GoogLeNet incarnation of the Inception architecture.

Number of 1x1 filters in the reduction layer before 3x3 or 5x5 convolution

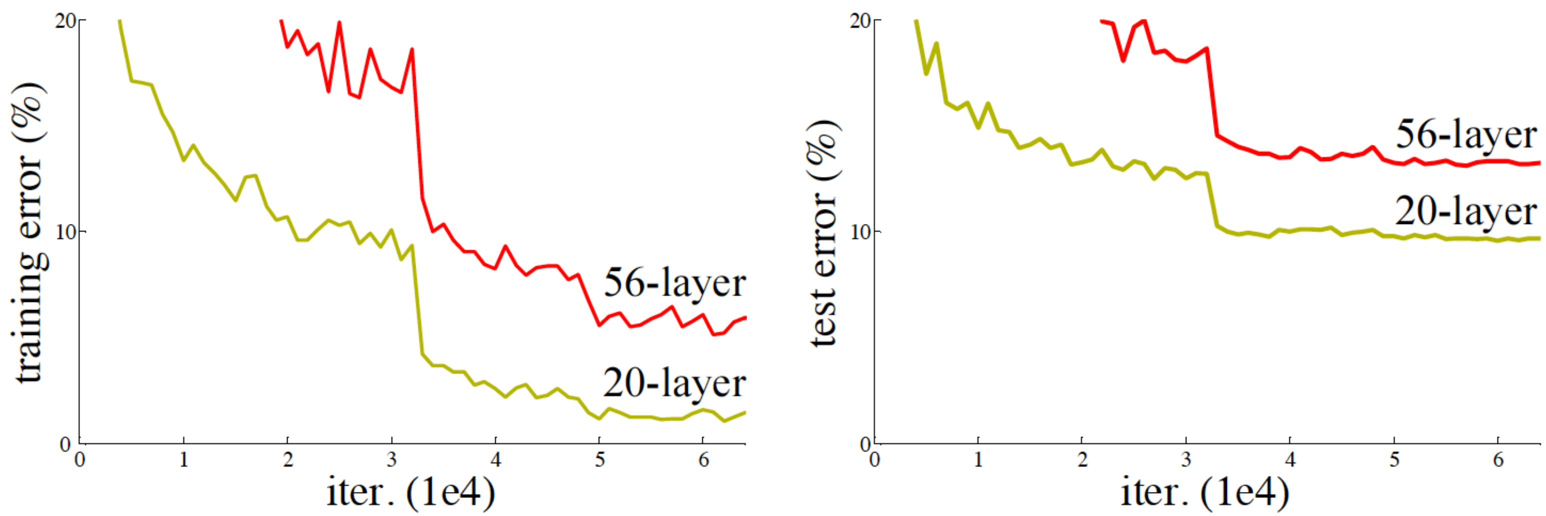
Number of 1x1 filters after max pooling

Number of 5x5 filters

Shortcut connection and Resnet

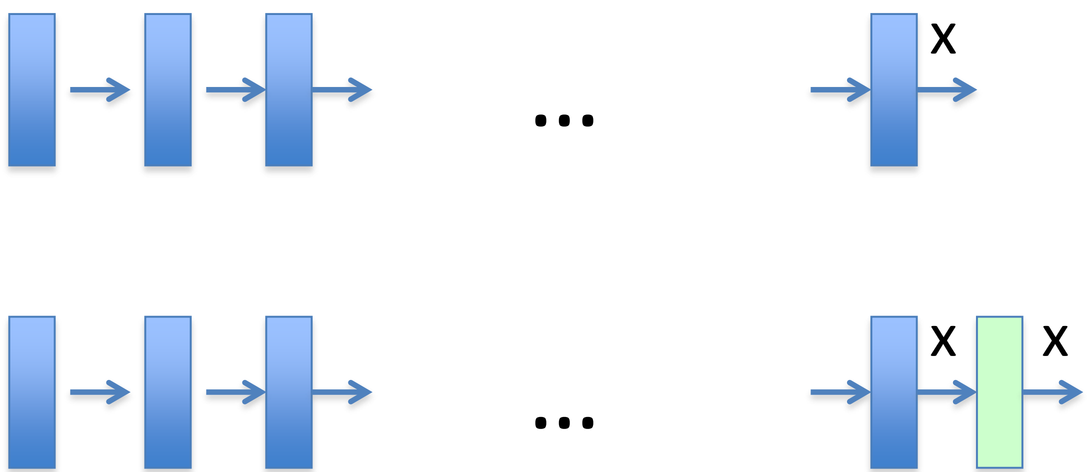
- Challenges of going deeper
  - Vanishing gradient: gradient is backpropagated to earlier layers, repeated multiplication may make the gradient very small
  - Overfitting: good in training, bad in testing
  - Difficult to learn an identity mapping in a deep architecture

Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer “plain” networks. The deeper network has higher training error, and thus test error



Shortcut connection and Resnet

- Challenges of going deeper
  - Vanishing gradient: gradient is backpropagated to earlier layers, repeated multiplication may make the gradient very small
  - Overfitting: good in training, bad in testing
  - Difficult to learn an identity mapping in a deep architecture

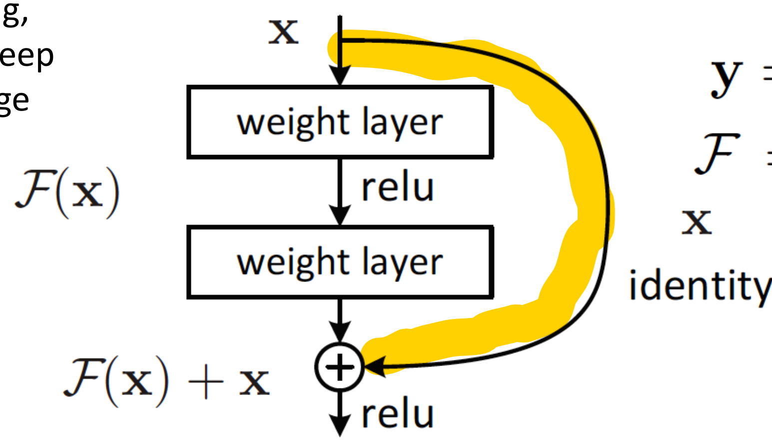


Adding an identity mapping layer would have no worse performance -> difficulty in learning an identity mapping results in poorer performance when going deeper

Shortcut connection and Resnet

- Deep residual learning
  - Learn the residual mapping instead of the entire mapping

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



$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}$$

$$\mathcal{F} = W_2\sigma(W_1\mathbf{x})$$

Element-wise addition, channel by channel (Therefore, need to ensure x and $\mathcal{F}(\mathrm{x})$ have the same spatial dim and channel)

Optimal mapping is close to an identity mapping, residual learning is a construction to ease the learning

Understanding Residual Learning

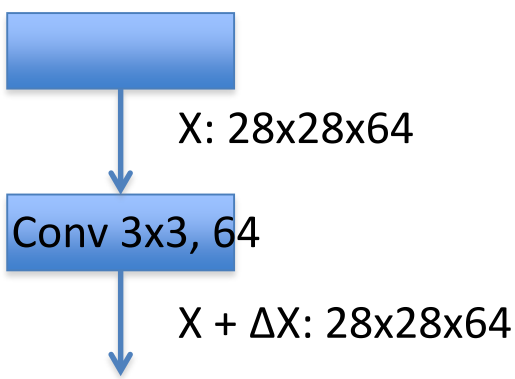
- Complex engineering system
  - To ensure that adding an additional layer does not degrade the performance (see 20 vs 56 layer example)
  - This is addressed by the skip connection, which facilitates learning of identity mapping (learning identity mapping with conventional architecture is hard)

- Ensemble learning
  - Residue networks can be seen as a collection of many paths with different length
  - These paths show ensemble-like behavior - do not strongly depend on each other
  - Most gradient of 110-layers network comes from paths that are 10-34 layers deep -> avoid vanishing gradient
  - Residual networks behave like ensembles of relatively shallow networks

- Thus, we can build deeper networks, which lead to improvement in accuracy

Understanding Residual Learning

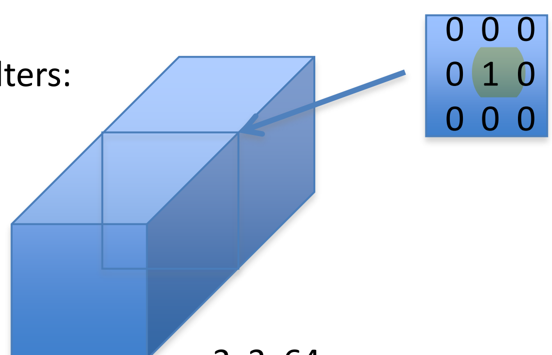
- Deep residual learning
  - Learn the residual mapping instead of the entire mapping



Want ΔX to be small, so that the output feature maps would not cause dramatic change (degrade) in performance

- Incremental approach to build complex DNN
- Need $W \approx 1$, i.e., identity mapping
- But hard to learn with data-driven approach
- An issue with optimization (non-convex, high dimensional loss function)

One of the 64 filters:

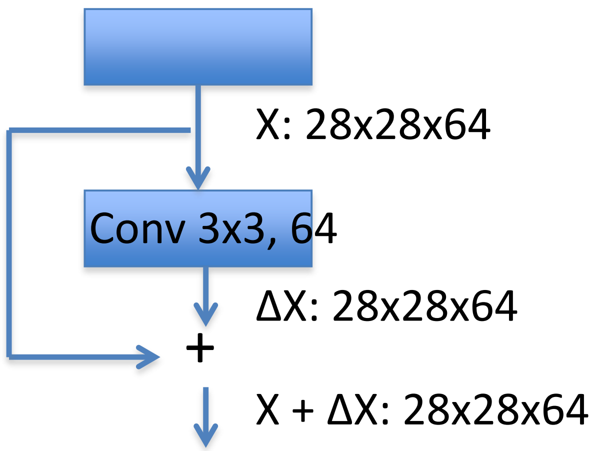


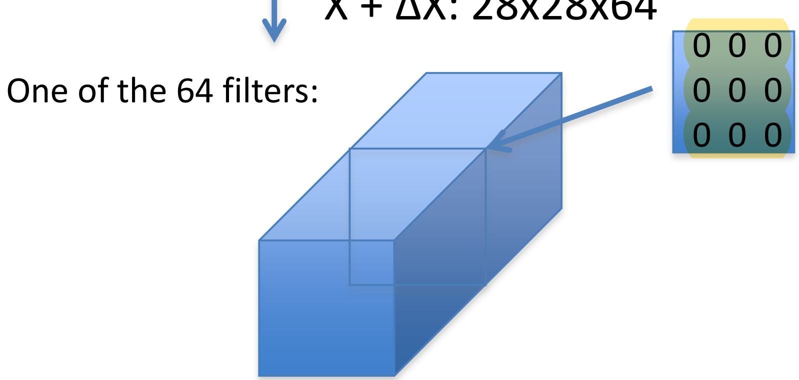
All zero in other slices

3x3x64

Understanding Residual Learning

- Deep residual learning
  – Learn the residual mapping instead of the entire mapping



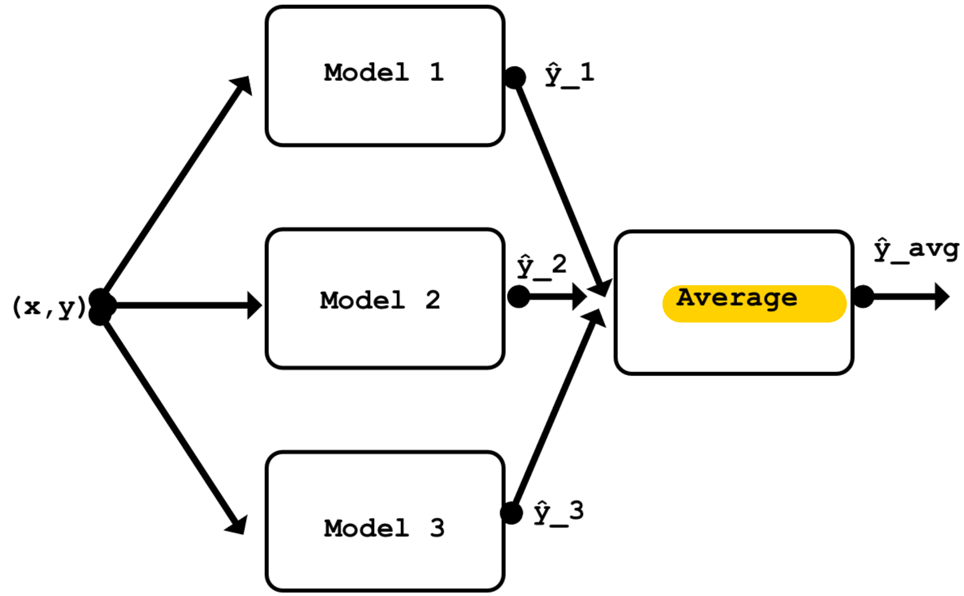


Want ΔX to be small, so that the output feature maps would not cause dramatic change (degrade) in performance
- With skip connection, need W ≈ 0
- Easy to learn with regularization on W

"We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers."

Ensemble learning

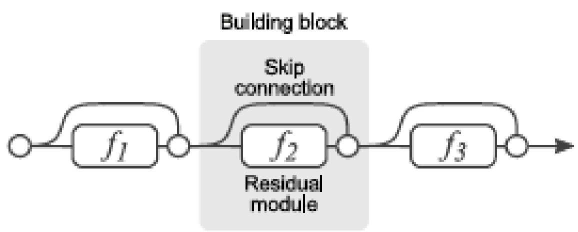
- Ensemble methods use multiple learning algorithms to improve prediction
- Each model is trained independently
- Inputs need to be evaluated multiple times
- Smooth performance w.r.t. number of models



Understanding Residual Learning

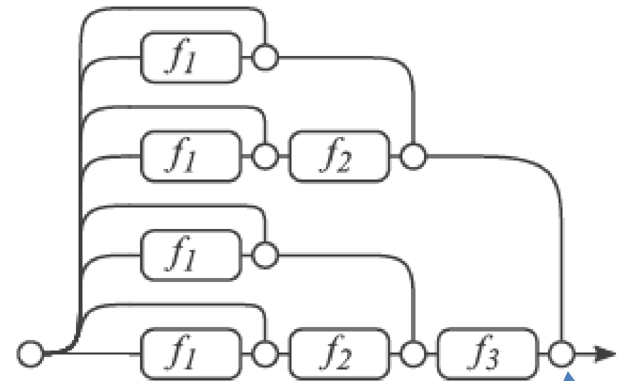
- Another reasoning: Ensembles of networks

-Residue networks can be viewed as a collection of many paths, instead of a single ultra-deep network



(a) Conventional 3-block residual network

=



(b) Unraveled view of (a)

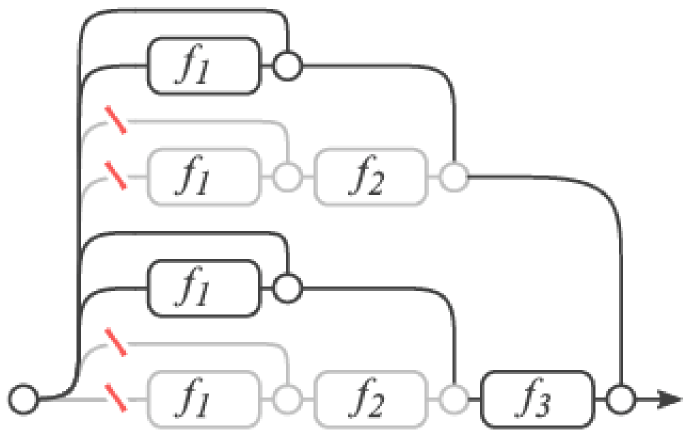
Averaging many networks

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

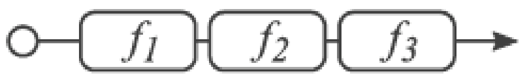
Understanding Residual Learning

- Another reasoning: Ensembles of networks

-These paths do not strongly depend on each other, even they are trained jointly
-Exhibit ensemble-like behavior: overall performance correlates smoothly with the number of valid paths



(a) Deleting f_2 from unraveled view



(b) Ordinary feedforward network

Figure 2: Deleting a layer in residual networks at test time (a) is equivalent to zeroing half of the paths. In ordinary feed-forward networks (b) such as VGG or AlexNet, deleting individual layers alters the only viable path from input to output.

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

Understanding Residual Learning

- Another reasoning: Ensembles of networks

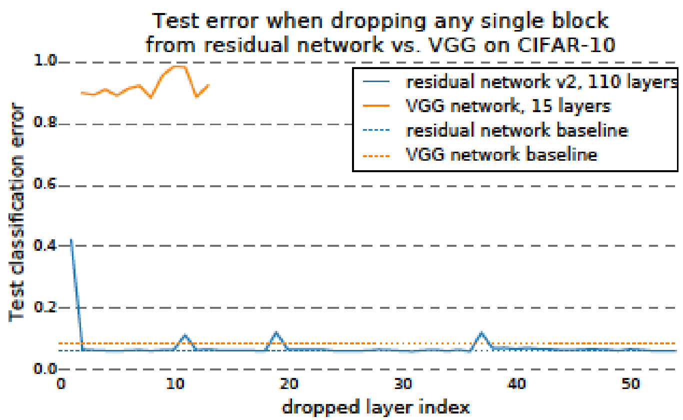


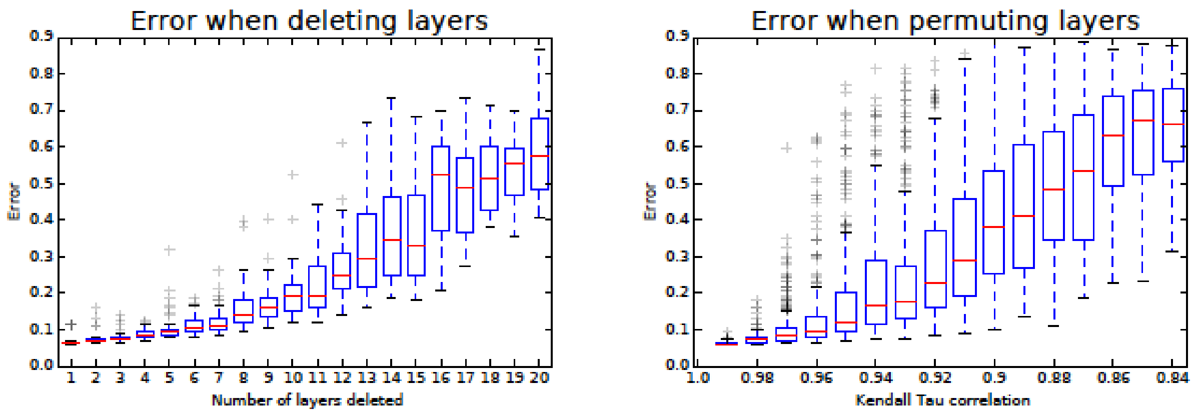
Figure 3: Deleting individual layers from VGG and a residual network on CIFAR-10. VGG performance drops to random chance when any one of its layers is deleted, but deleting individual modules from residual networks has a minimal impact on performance. Removing downsampling modules has a slightly higher impact.

-These paths do not strongly depend on each other, even they are trained jointly
-Exhibit ensemble-like behavior: overall performance correlates smoothly with the number of valid paths

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

Understanding Residual Learning

- Another reasoning: Ensembles of networks



(a) (b)

Figure 5: (a) Error increases smoothly when randomly deleting several modules from a residual network. (b) Error also increases smoothly when re-ordering a residual network by shuffling building blocks. The degree of reordering is measured by the Kendall Tau correlation coefficient. These results are similar to what one would expect from ensembles.

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

Understanding Residual Learning

- Another reasoning: Ensembles of networks

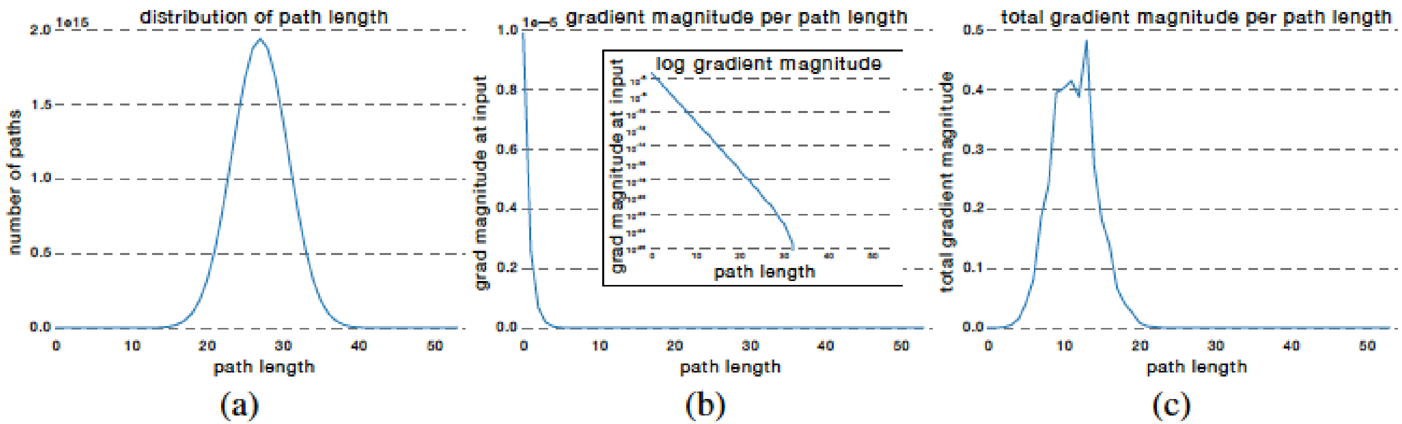
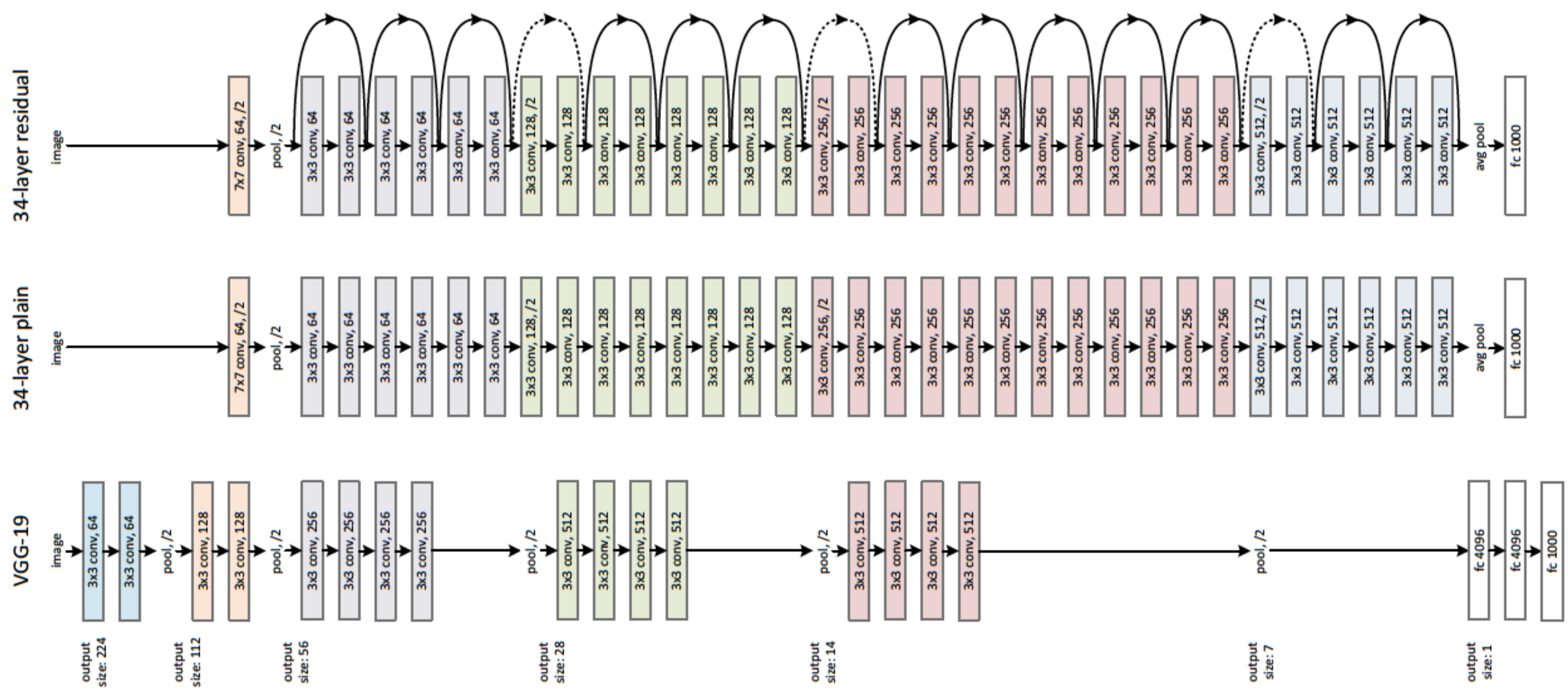


Figure 6: How much gradient do the paths of different lengths contribute in a residual network? To find out, we first show the distribution of all possible path lengths (a). This follows a Binomial distribution. Second, we record how much gradient is induced on the first layer of the network through paths of varying length (b), which appears to decay roughly exponentially with the number of modules the gradient passes through. Finally, we can multiply these two functions (c) to show how much gradient comes from all paths of a certain length. Though there are many paths of medium length, paths longer than ~20 modules are generally too long to contribute noticeable gradient during training. This suggests that the effective paths in residual networks are relatively shallow.

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

Shortcut connection and Resnet

- Resnet: stack of deep residual learning modules



Other variants: ResNet-50/101/152