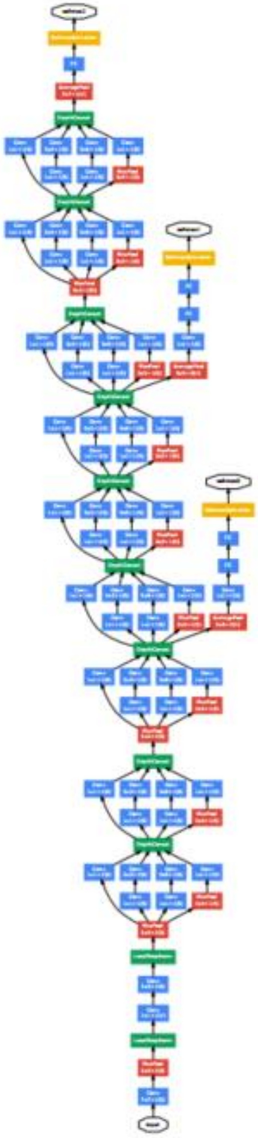
Modern ConvNet Architectures



C. Szegedy et al, "Going Deeper with Convolutions" (CVPR 2015)

CSC321: Intro to Machine Learning and Neural Networks, Winter 2016

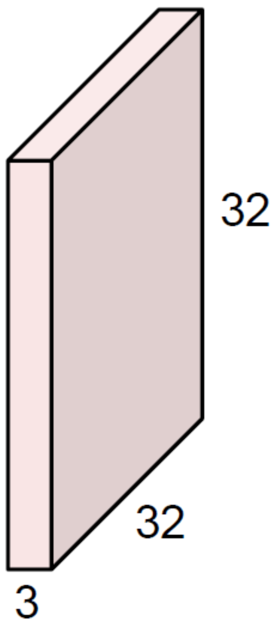
Michael Guerzhoy

Slides from Andrej Karpathy, Geoffrey Hinton, Christian Szegedy et al, Yann LeCun et al.

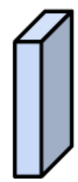
Convolution Layer

Filters always extend the full depth of the input volume

32x32x3 image

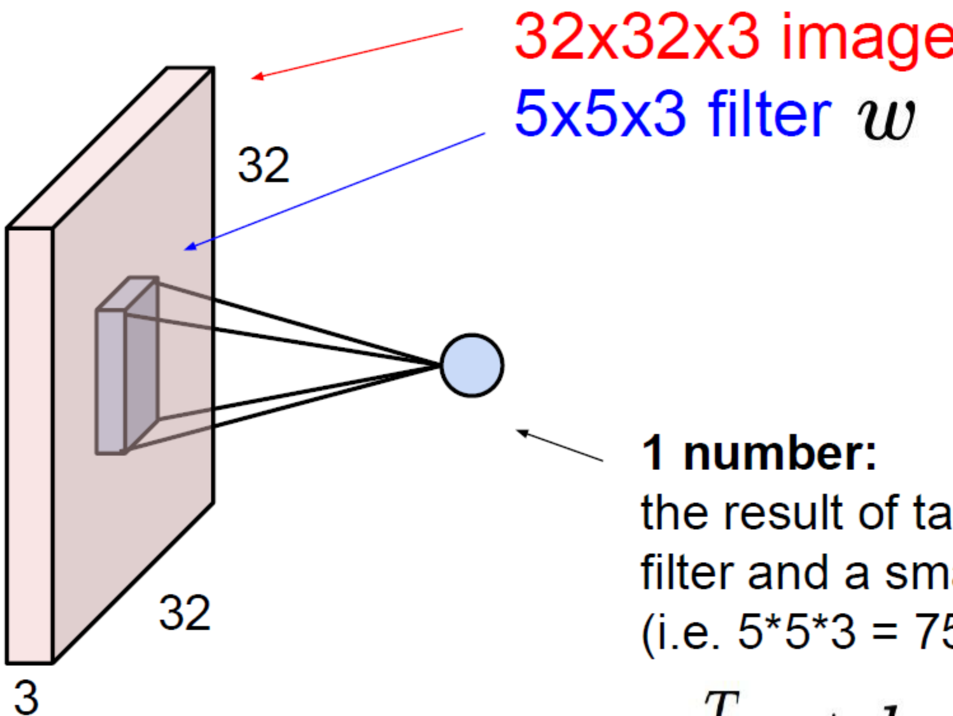


5x5x3 filter



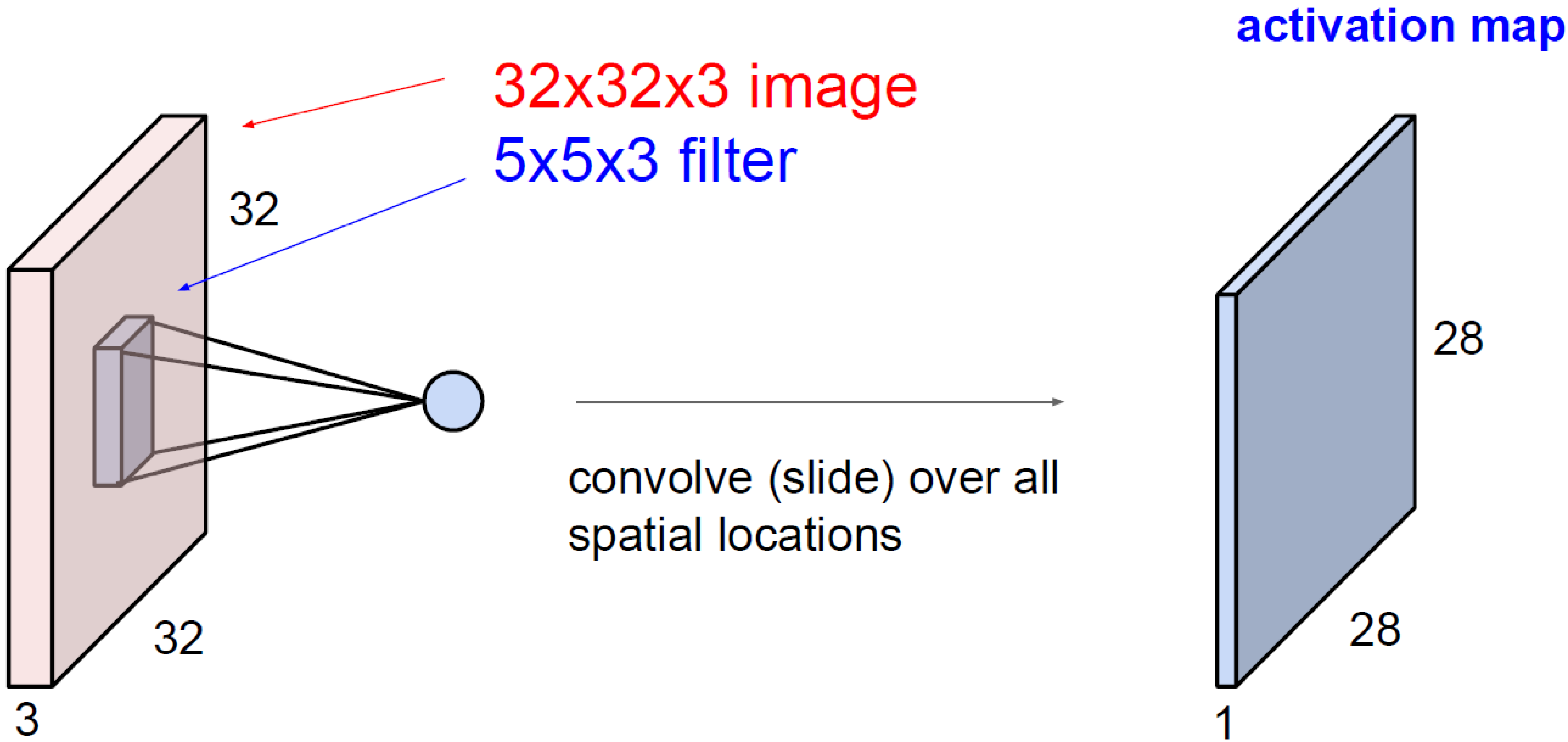
Convolve the filter with the image i.e. “slide over the image spatially, computing dot products”

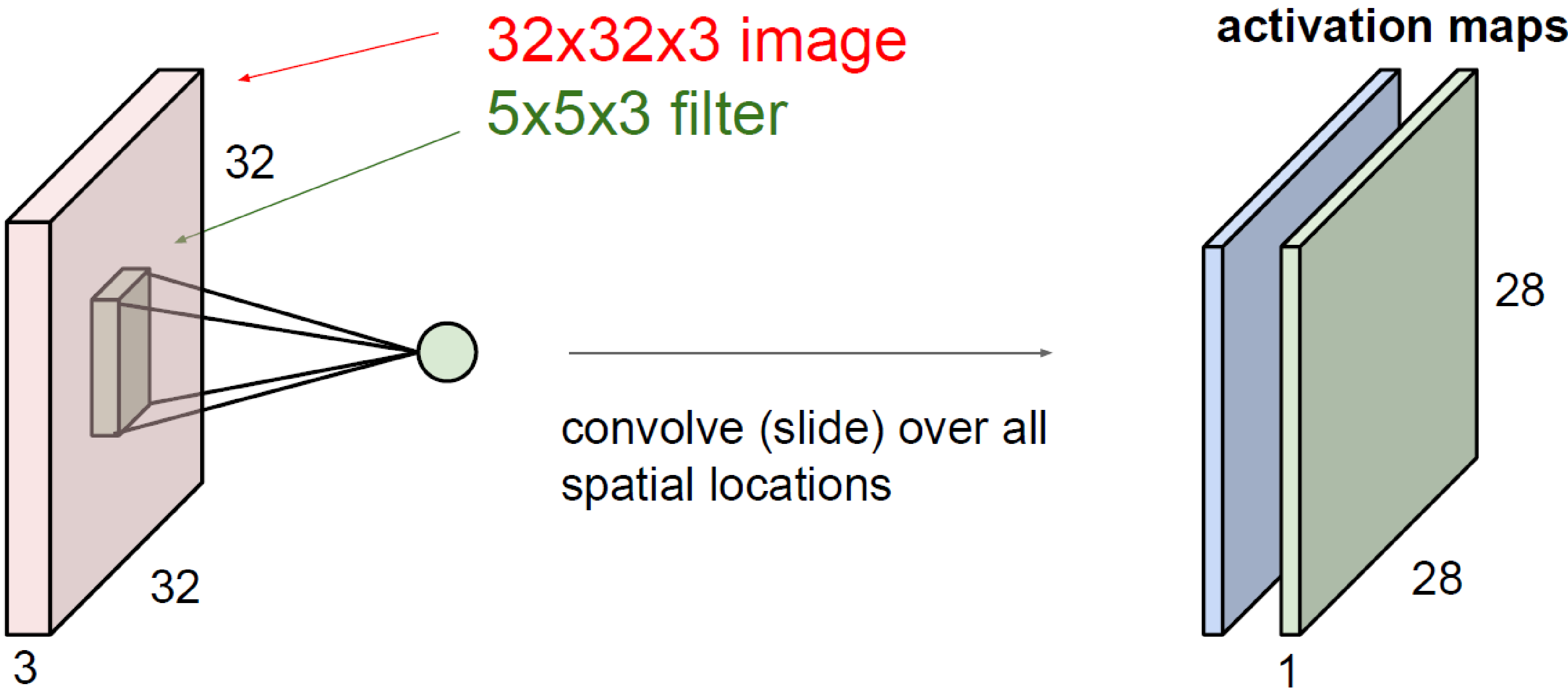
Convolution Layer

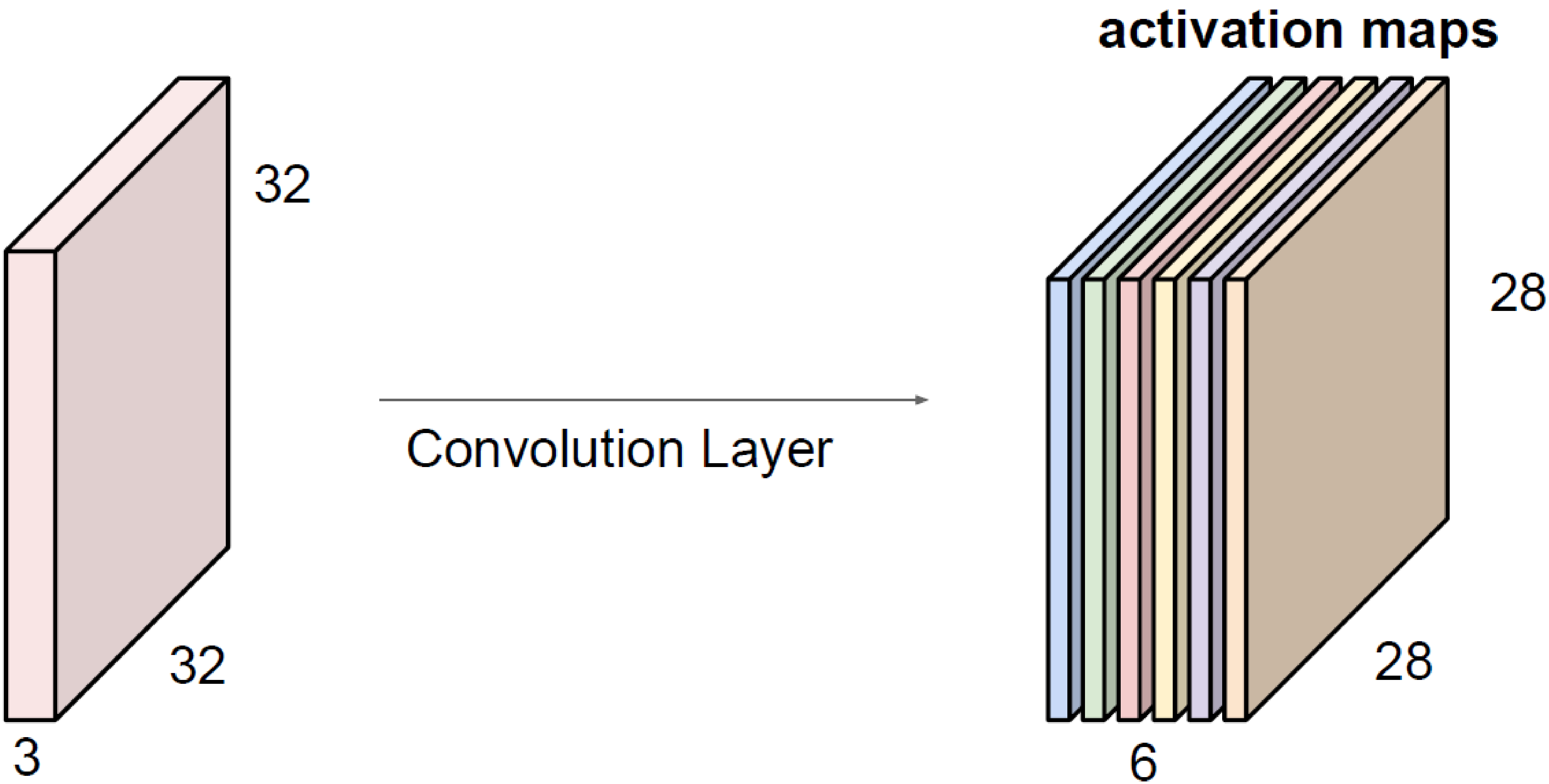


1 number:
the result of taking a dot product between the filter and a small 5x5x3 chunk of the image
(i.e. 5*5*3 = 75-dimensional dot product + bias)

$$w^T x + b$$


activation map
32x32x3 image
5x5x3 filter
32
32
3
convolve (slide) over all spatial locations
28
28
1

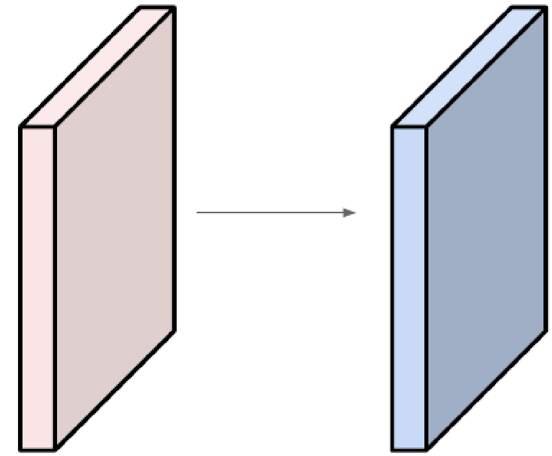

32x32x3 image
5x5x3 filter
32
32
3
convolve (slide) over all
spatial locations
activation maps
28
28
1



We stack these up to get a “new image” of size 28x28x6!

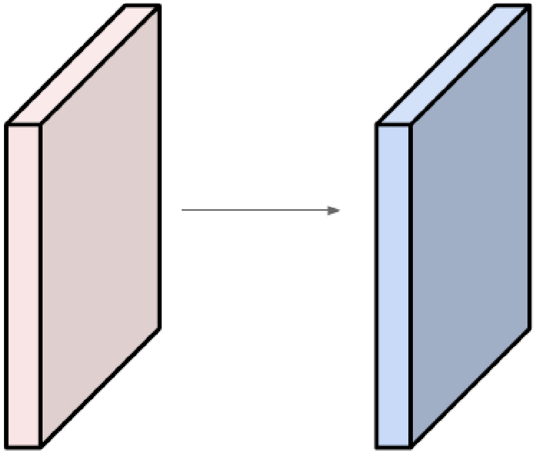
Examples time:

Input volume: **32x32x3**
10 5x5 filters with stride 1, pad 2



Number of parameters in this layer?

Examples time:



Input volume: **32x32x3**

10 5x5 filters with stride 1, pad 2

Number of parameters in this layer?

each filter has 5*5*3 + 1 = 76 params (+1 for bias)

=> 76*10 = **760**

Convolutional Layer

- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

Convolutional Layers Summary Again

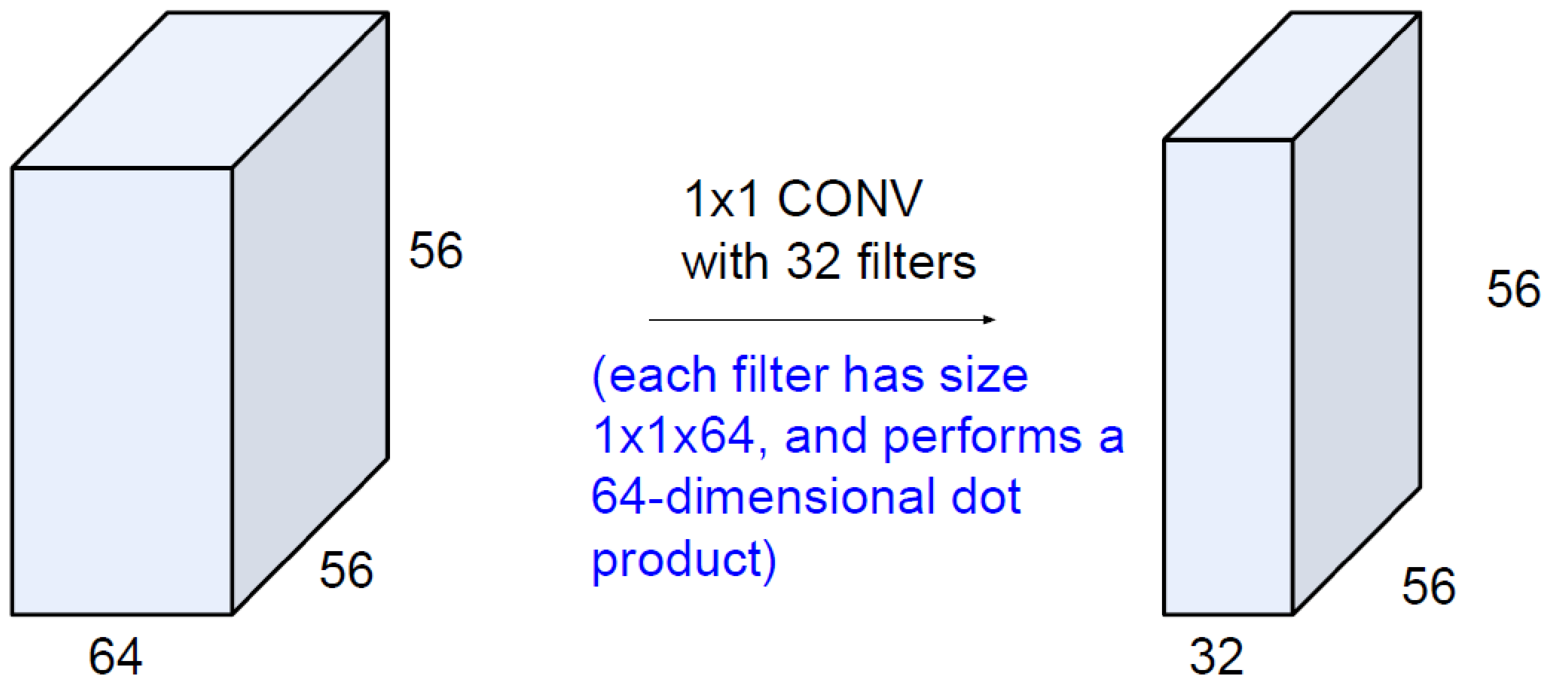
Common settings:

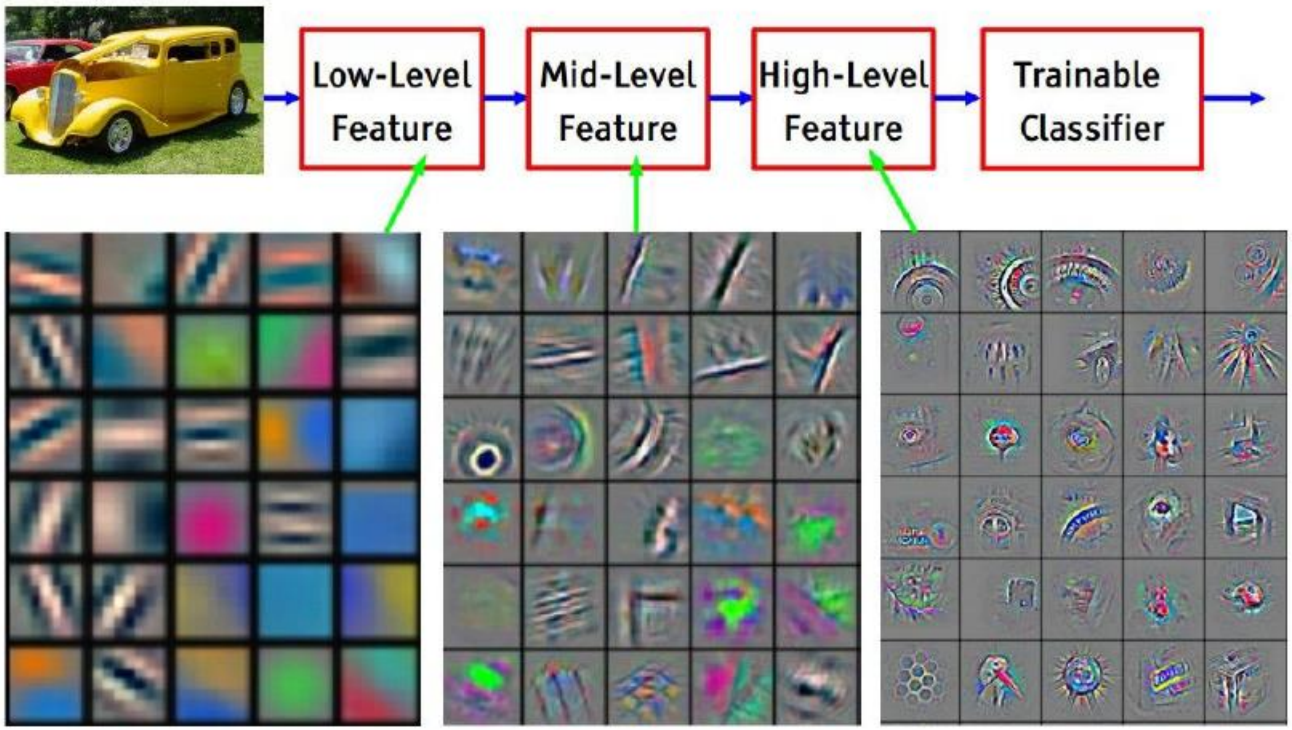
K = (powers of 2, e.g. 32, 64, 128, 512)
- F = 3, S = 1, P = 1
- F = 5, S = 1, P = 2
- F = 5, S = 2, P = ? (whatever fits)
- F = 1, S = 1, P = 0

Summary. To summarize, the Conv Layer:

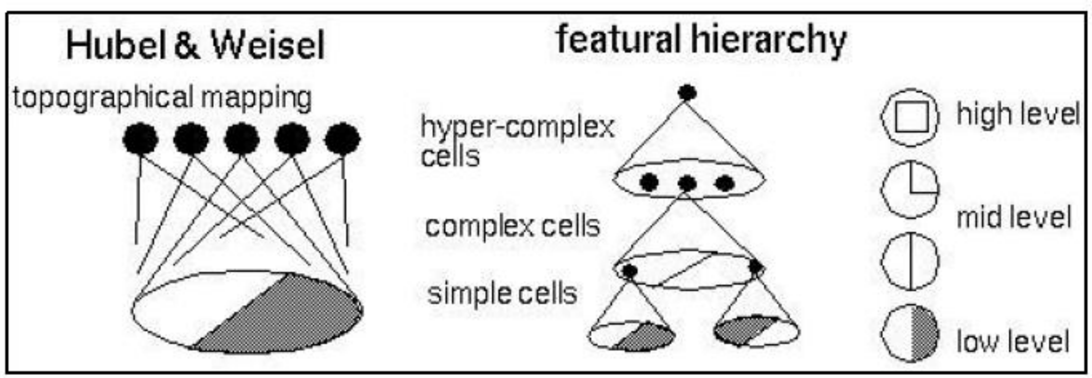
- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires four hyperparameters:
  - Number of filters K,
  - their spatial extent F,
  - the stride S,
  - the amount of zero padding P.
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F + 2P)/S + 1$
  - $H_2 = (H_1 - F + 2P)/S + 1$ (i.e. width and height are computed equally by symmetry)
  - $D_2 = K$
- With parameter sharing, it introduces $F \cdot F \cdot D_1$ weights per filter, for a total of $(F \cdot F \cdot D_1) \cdot K$ weights and K biases.
- In the output volume, the d-th depth slice (of size $W_2 \times H_2$) is the result of performing a valid convolution of the d-th filter over the input volume with a stride of S, and then offset by d-th bias.

(1x1 convolutions?)



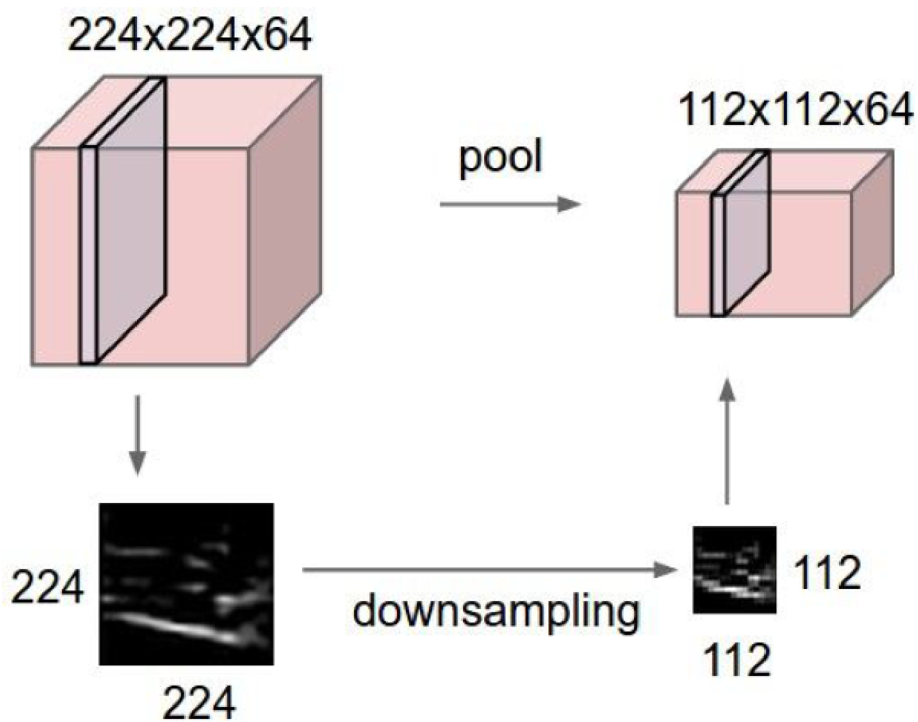


Feature visualization of convolutional net trained on ImageNet from [Zeiler & Fergus 2013]



Pooling Layer

- makes the representations smaller and more manageable
- operates over each activation map independently:



Pooling Layer

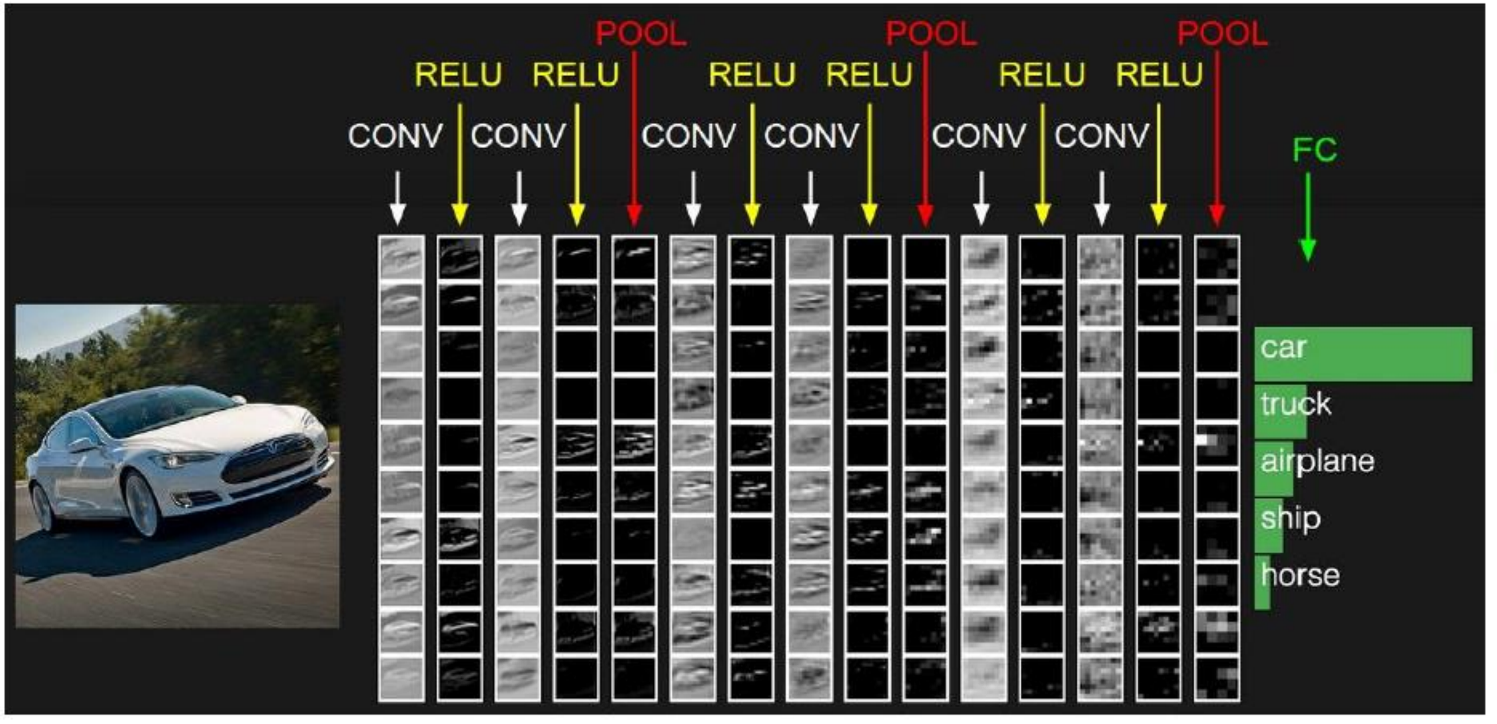
Common settings:

F = 2, S = 2
F = 3, S = 2

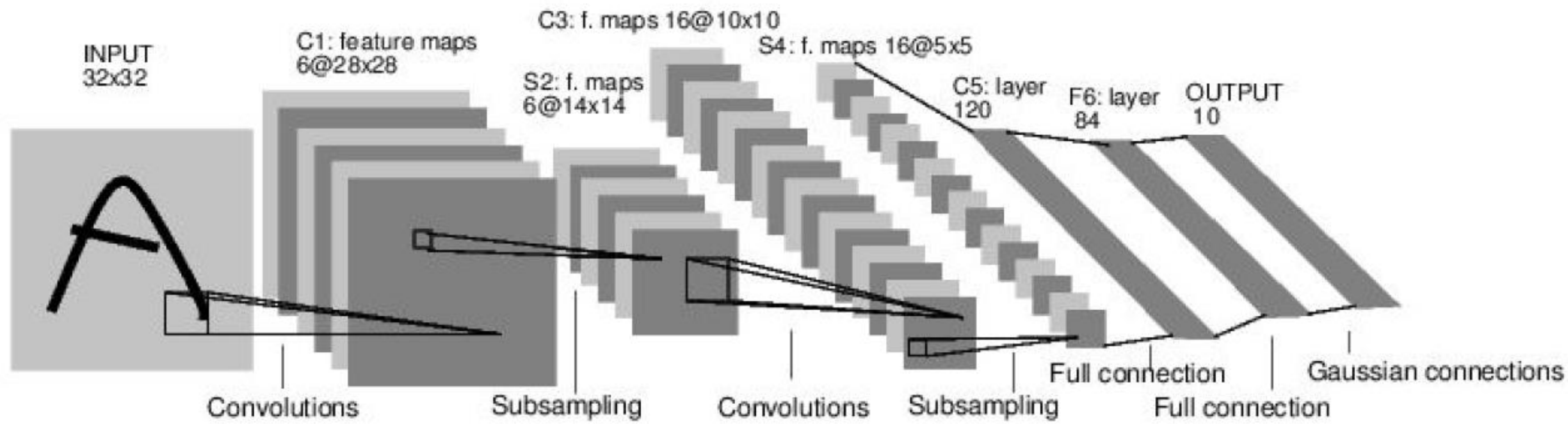
- Accepts a volume of size $W_1 \times H_1 \times D_1$
- Requires three hyperparameters:
  - their spatial extent F,
  - the stride S,
- Produces a volume of size $W_2 \times H_2 \times D_2$ where:
  - $W_2 = (W_1 - F)/S + 1$
  - $H_2 = (H_1 - F)/S + 1$
  - $D_2 = D_1$
- Introduces zero parameters since it computes a fixed function of the input
- Note that it is not common to use zero-padding for Pooling layers

Fully-Connected Layer

- Contains neurons that connect to the entire lower year, as in ordinary neural networks



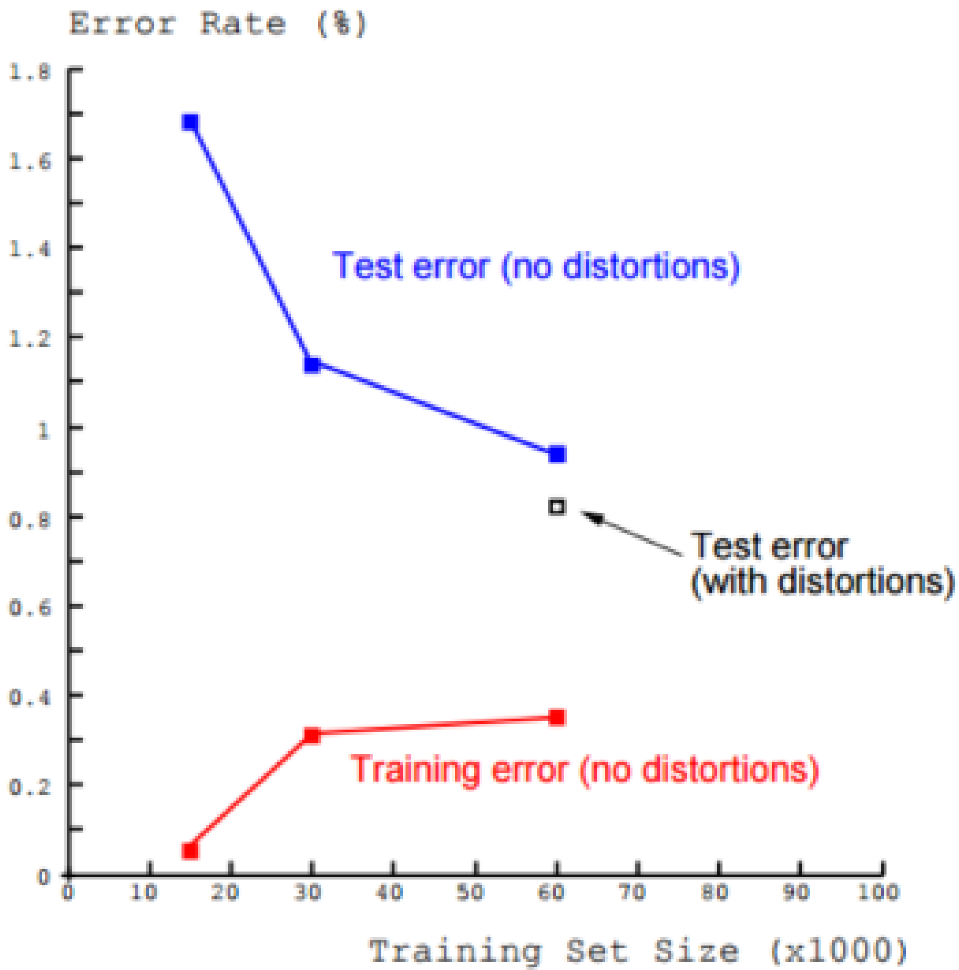
LeNet-5 (Yann LeCun et al, 1998)



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

LeNet-5 errors

LeNet Performance



IMAGENET Large Scale Visual Recognition Challenge

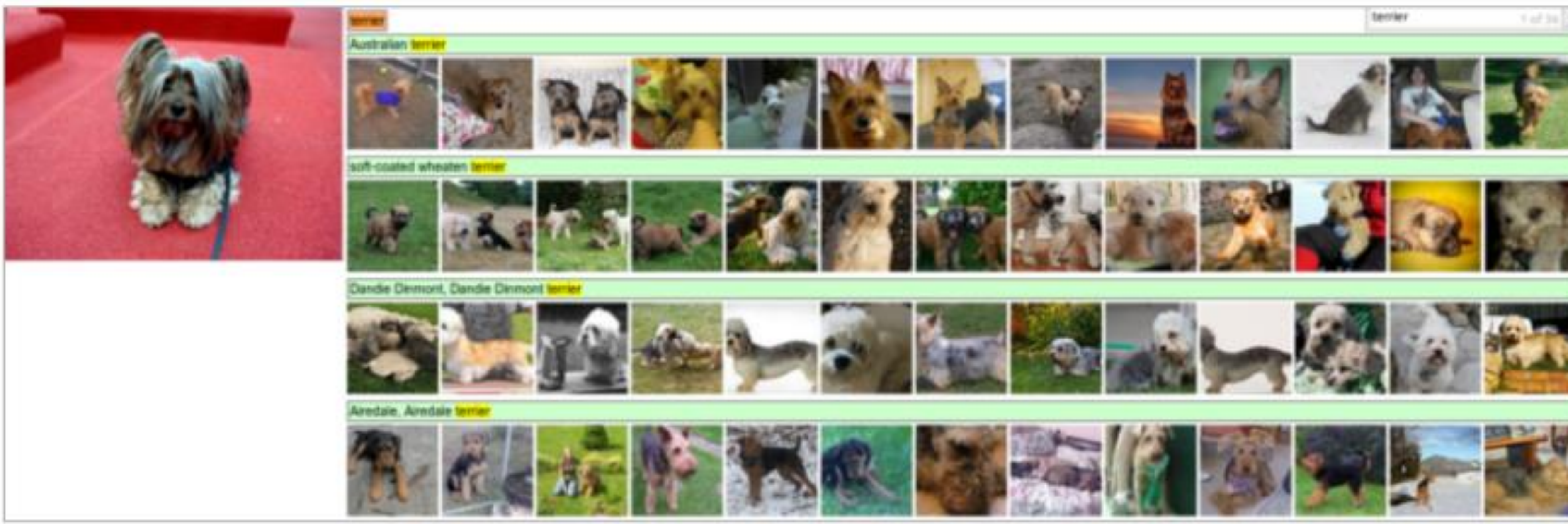
- About one million images, 1000 object categories in the training set
- Task: what is the object in the image
  - I.e., classify the image into one of 1000 categories
- Evaluation: is one of the best 5 guesses correct?


mite
container ship
motor scooter
leopard
mite
black widow
cockroach
tick
starfish
container ship
lifeboat
amphibian
fireboat
drilling platform
motor scooter
go-kart
moped
bumper car
golfcart
leopard
jaguar
cheetah
snow leopard
Egyptian cat
grille
mushroom
cherry
Madagascar cat
convertible
grille
pickup
beach wagon
fire engine
agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers
dalmatian
grape
elderberry
ffordshire bullterrier
currant
squirrel monkey
spider monkey
titi
indri
howler monkey

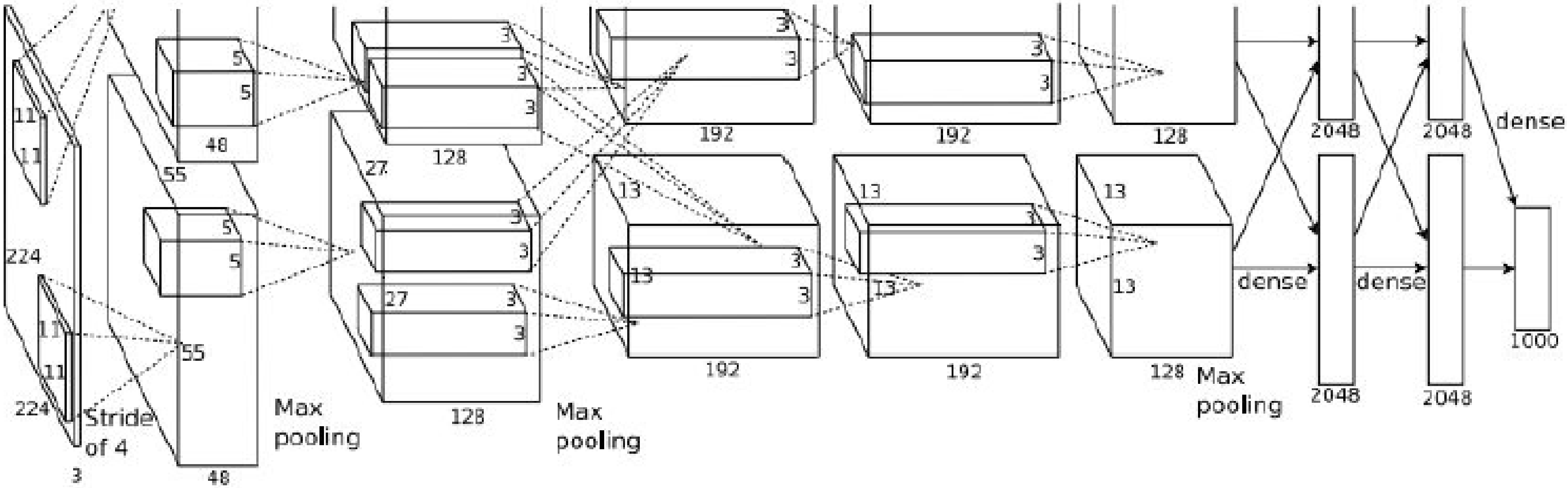
Human Performance on ImageNet

- http://karpathy.github.io/2014/09/02/what-i-learned-from-competing-against-a-convnet-on-imagenet/
- 5.1% error (i.e., none of the 5 guesses the person makes are correct)
- Try it yourself!
  - http://cs.stanford.edu/people/karpathy/ilsvrc/

Dogs Are Hard to Classify!



AlexNet (Krizhevsky et al. 2012)



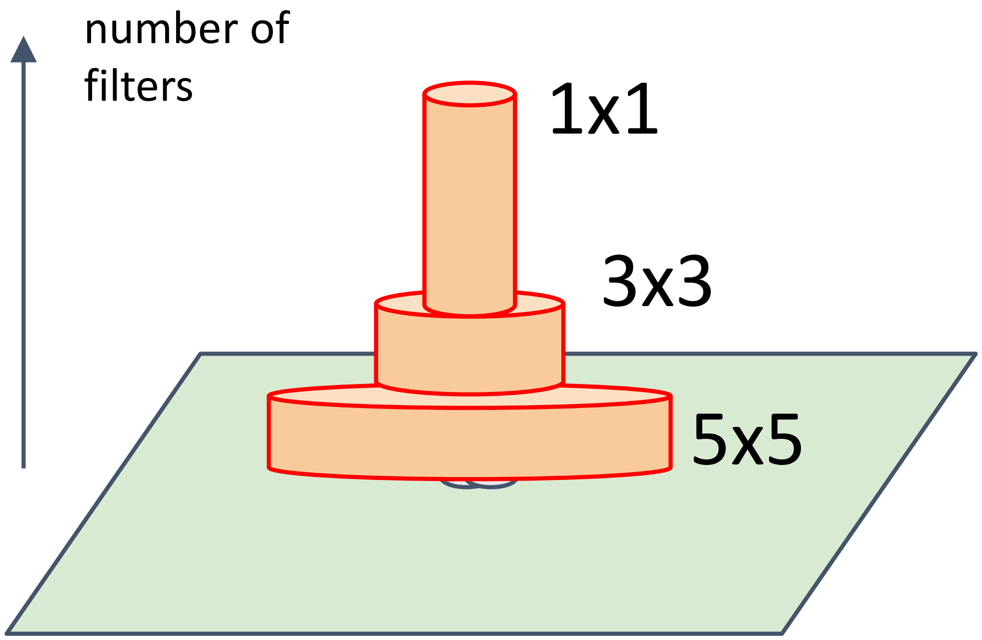
Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] 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)

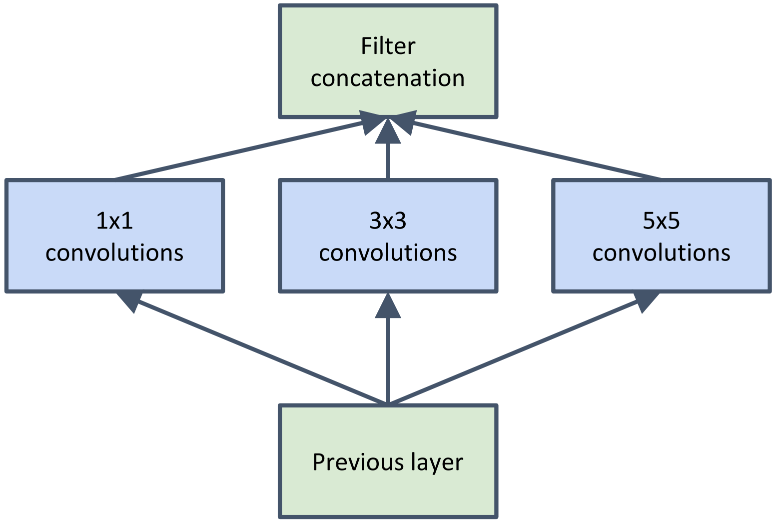
Details/Retrospectives:
- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble: 18.2% -> 15.4%

GoogLeNet (Szegedy et al, 2014)

- 6.7% error on ImageNet
  - State of the Art (at the time)
  - Close to human performance
- A very deep net
- Several Neat Tricks

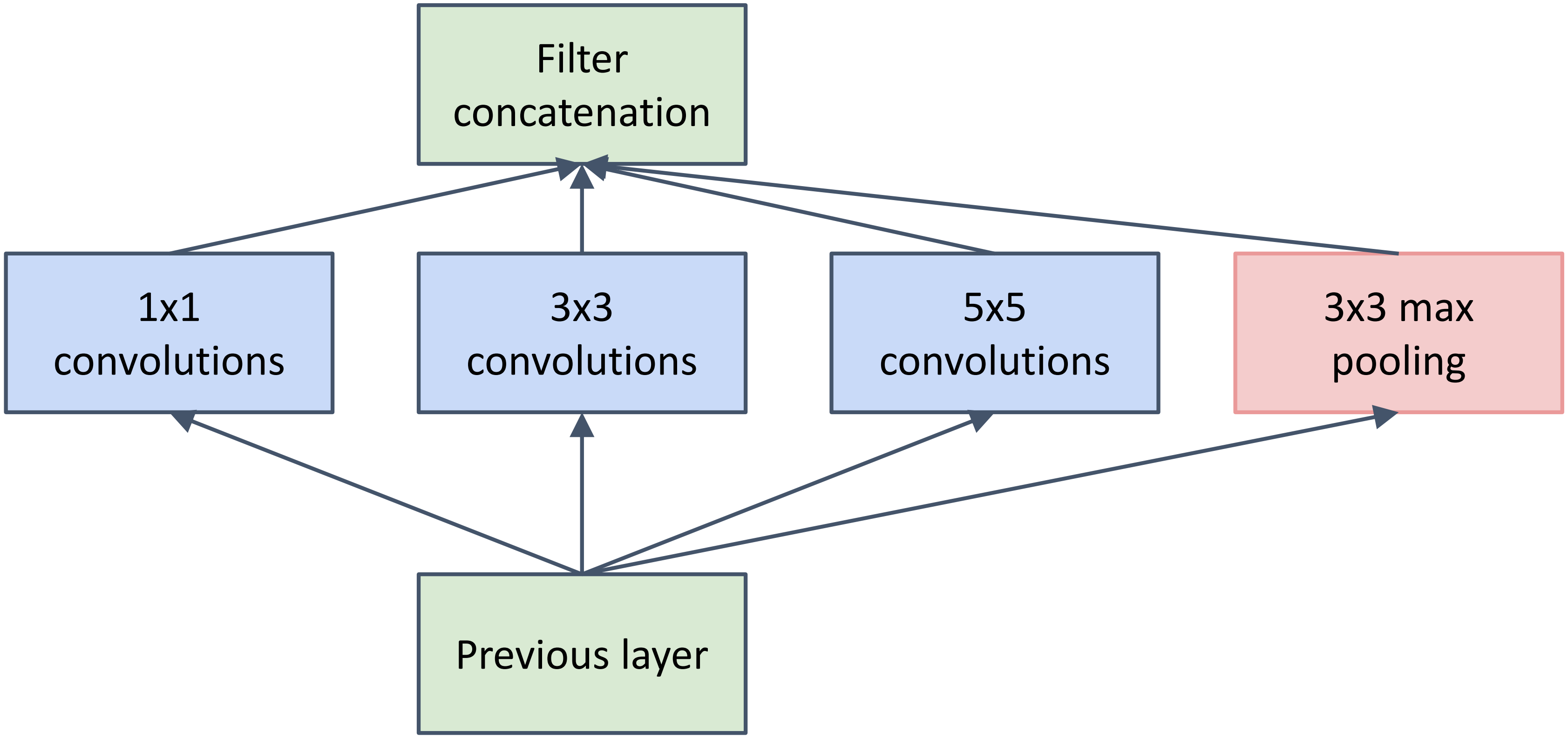
A Heterogeneous Set of Convolutions





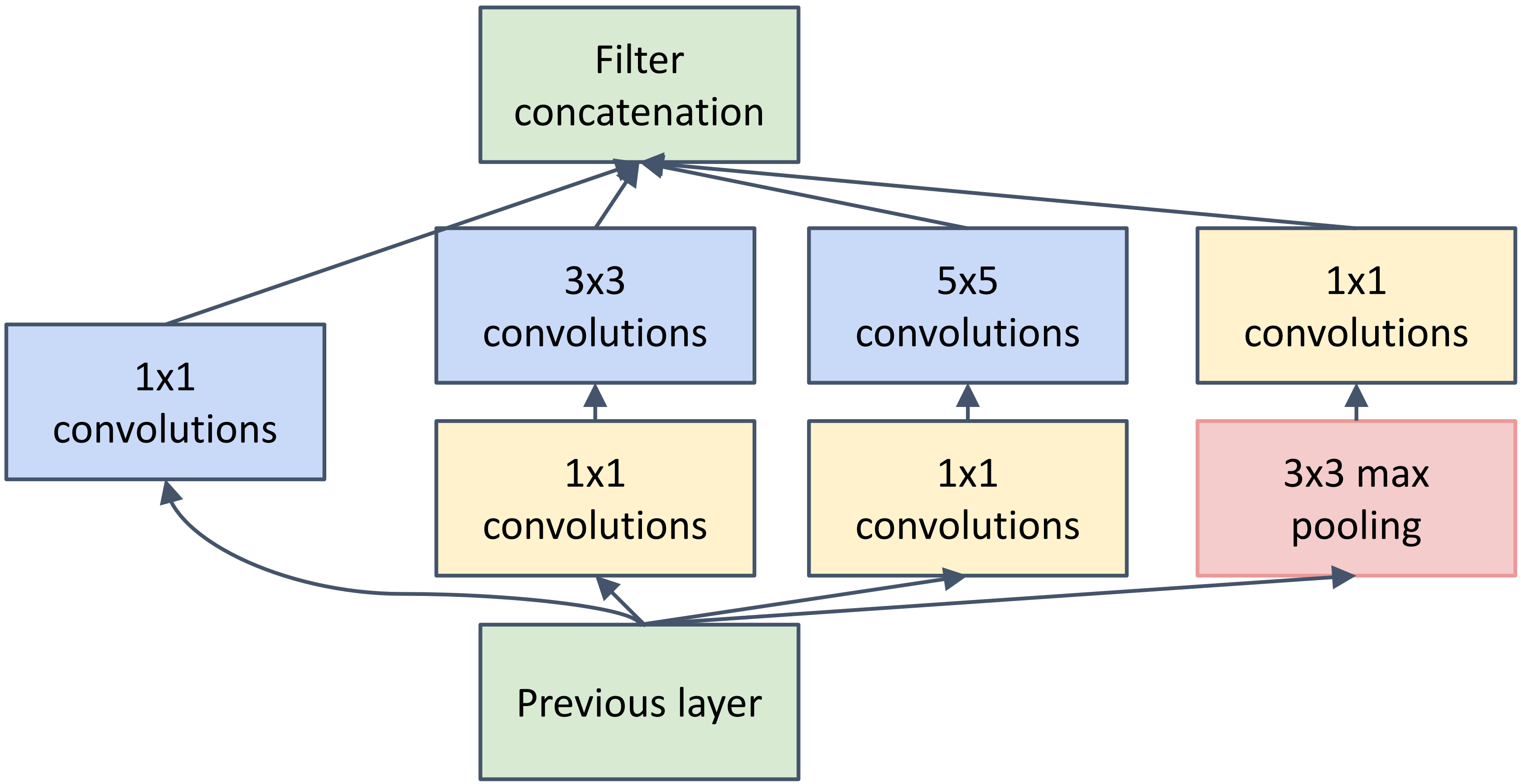
- Apply filters of several sizes so as to capture invariances at different scales
- Concatenate all the filters
- (Note: could always use 5x5 filters, but that's expensive, *and* hard to learn)

Inception Module: Basic Idea (doesn’t work, too many features)



- Do max-pooling directly too in case convolution not needed
- Super expensive if we want a decent number of filters in each layer

Inception Module



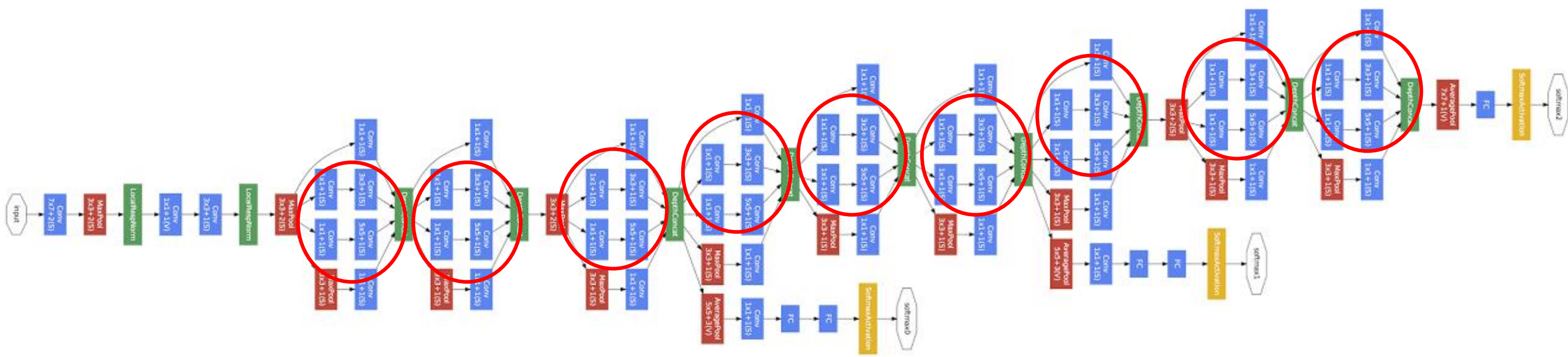
Inception Module

- The 1x1 convolutions at the bottom of the module reduce the number of inputs by a factor of

$$\frac{depth\ of\ input\ layer}{N.\,of\ 1x1\ convolutions}$$

- Decreases computation cost dramatically

Inception

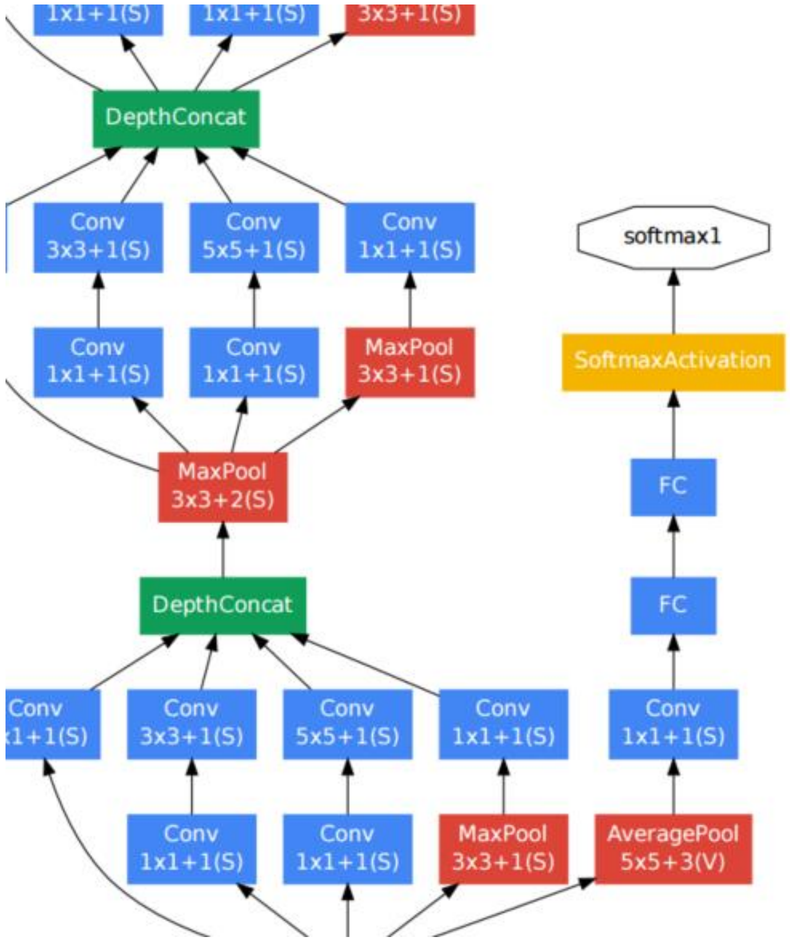


9 **Inception** modules

Network in a network in a network...

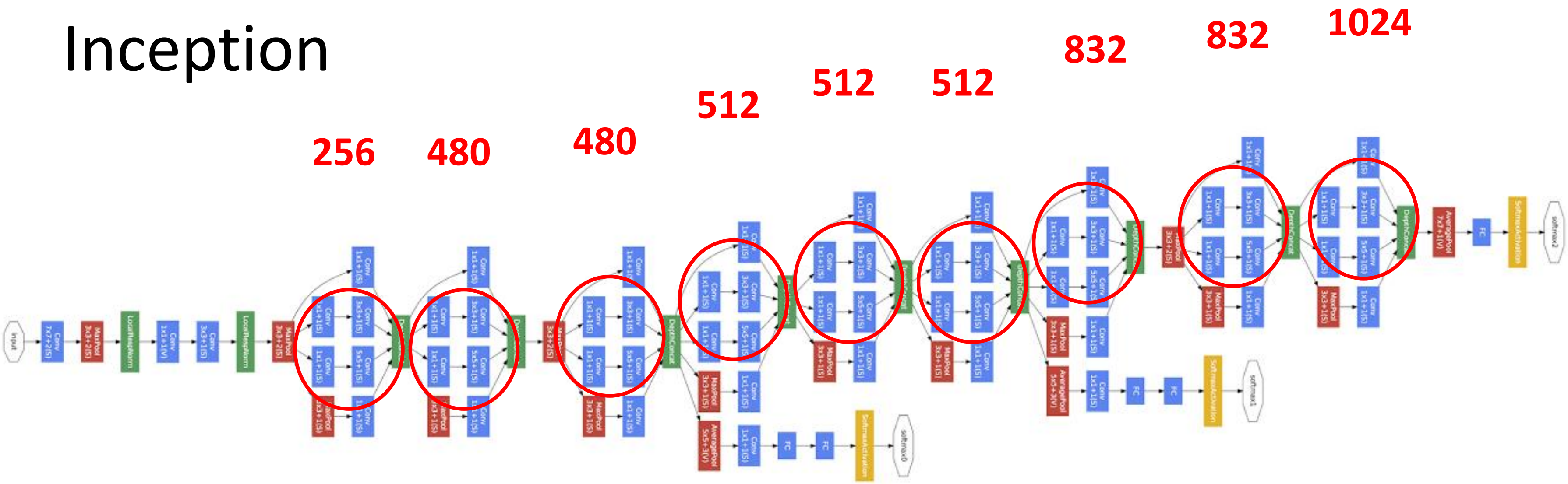
Convolution
Pooling
Softmax
Other

Look Closer



- Softmax outputs in the middle of the network, the same as at the top
  - Encourage the network to learn features that are useful for classification in the middle

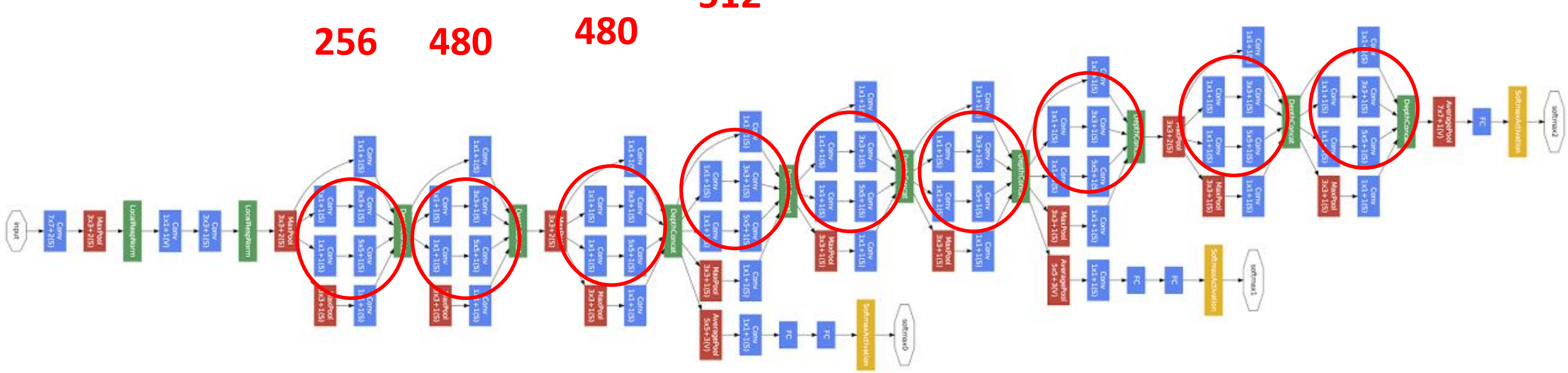
Inception



Width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules.

Can remove fully connected layers on top completely

Inception



Width of inception modules ranges from 256 filters (in early modules) to 1024 in top inception modules.

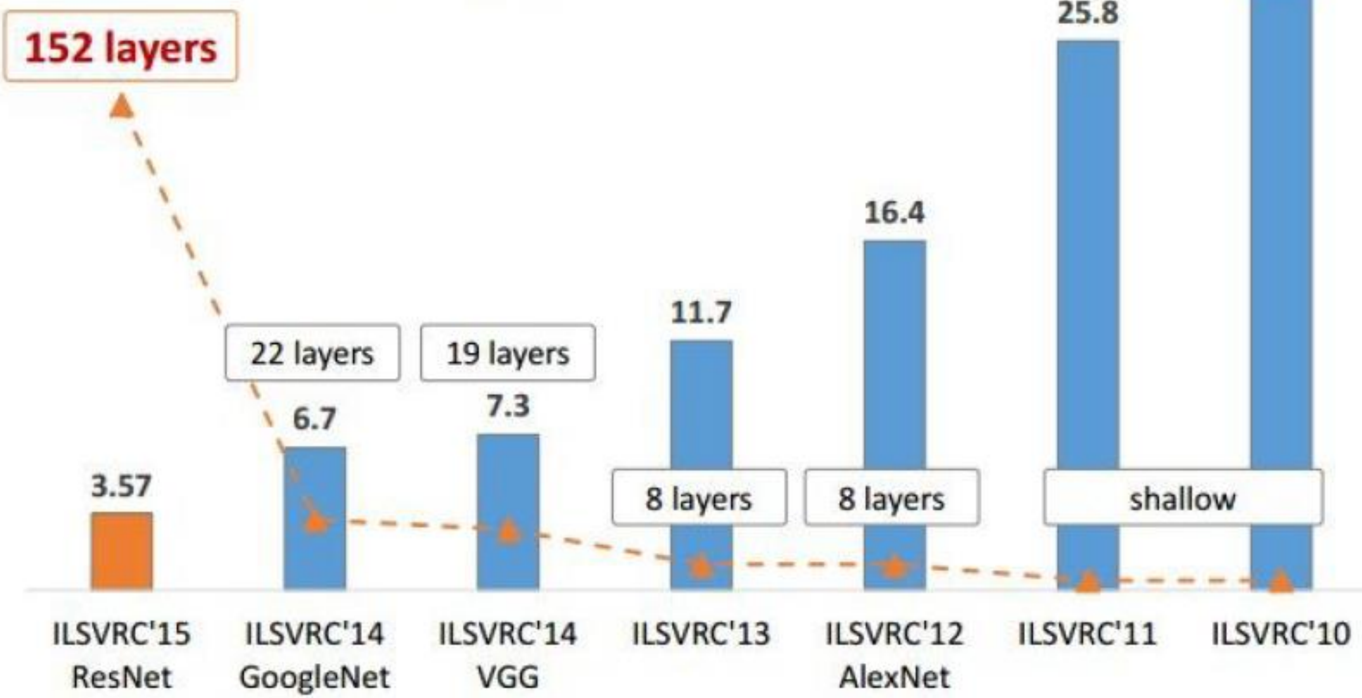
Can remove fully connected layers on top completely

Number of parameters is reduced to 5 million

Computional cost is increased by less than 2X compared to Krizhevsky's network. (<1.5Bn operations/evaluation)

Microsoft Research

Revolution of Depth



ICCV15

Kaiming He, Xiangyu Zhang, Shaoqing Ren, & Jian Sun. "Deep Residual Learning for Image Recognition". arXiv 2015.

Aside: ConvNets vs. Monkeys

- Extract the features (neuron activities) from the Inferior Temporal Cortex of Rhesus Macaques when the monkeys are looking at images
- Extract features from the top layers of ConvNets when the ConvNets are looking at images
- Use both sets of features to classify images

Deep Neural Networks Rival the Representation of Primate IT Cortex for Core Visual Object Recognition

Charles F. Cadieu[1,*], Ha Hong[1,2], Daniel L. K. Yamins[1], Nicolas Pinto[1], Diego Ardila[1], Ethan A. Solomon[1], Najib J. Majaj[1], James J. DiCarlo[1]

1 Department of Brain and Cognitive Sciences and McGovern Institute for Brain Research, Massachusetts Institute of Technology, Cambridge, MA 02139

2 Harvard–MIT Division of Health Sciences and Technology, Institute for Medical Engineering and Science, Massachusetts Institute of Technology, Cambridge, MA 02139

∗ E-mail: Corresponding cadieu@mit.edu