UNIVERSITY OF LIVERPOOL

COMP 532
Machine Learning and BioInspired Optimization

Lecture 12: Deep Learning

Dr. Shan Luo

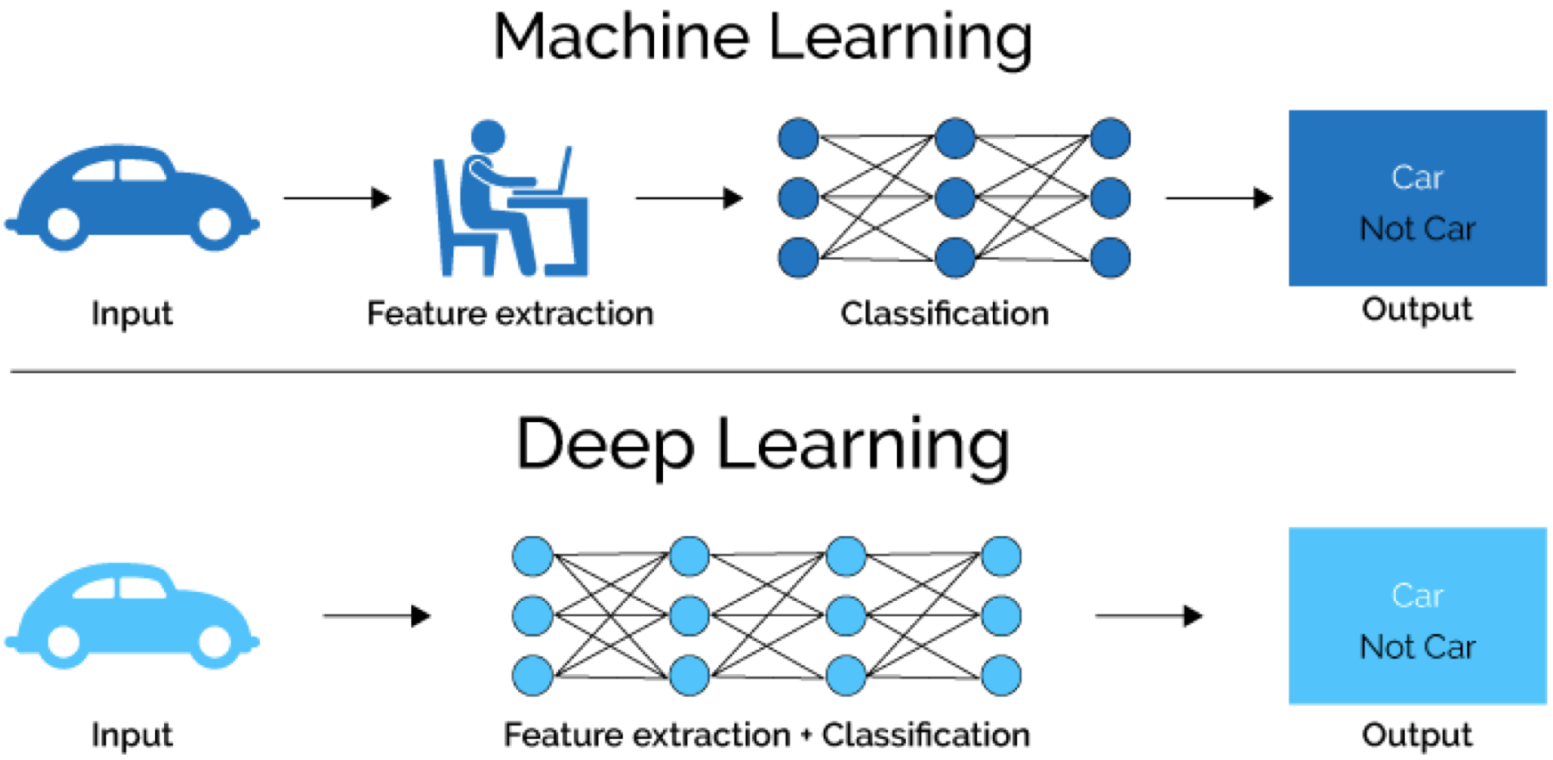
Department of Computer Science

shan.luo@liverpool.ac.uk

Recap

- What is Deep Learning?

Deep Learning is a family of methods that use deep architectures to learn high-level feature representations.



Recap

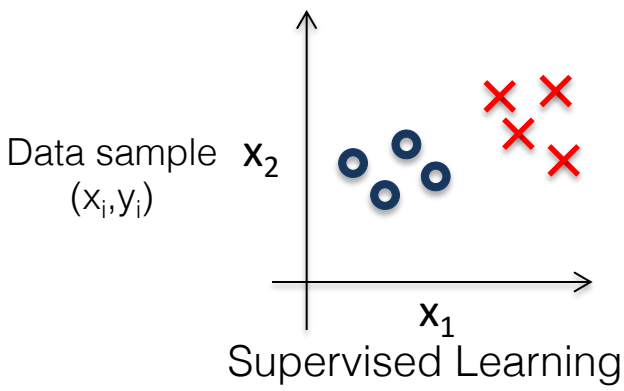
- What is Deep Learning?

➢ Composition of non-linear transformation of the data.

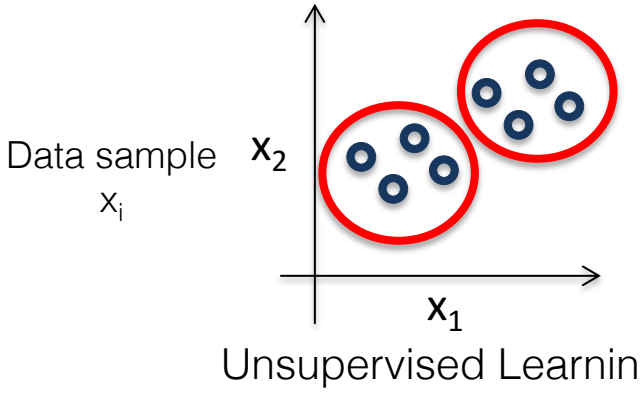
➢ Goal: Learn useful representations, a.k.a., features, directly from data.

➢ Many varieties, can be supervised (e.g., CNNs, and unsupervised (e.g., autoencoders, sparse coding).

➢ Today is about ConvNets, which is supervised learning.



Supervised Learning



Unsupervised Learning

Overview (2 lectures)

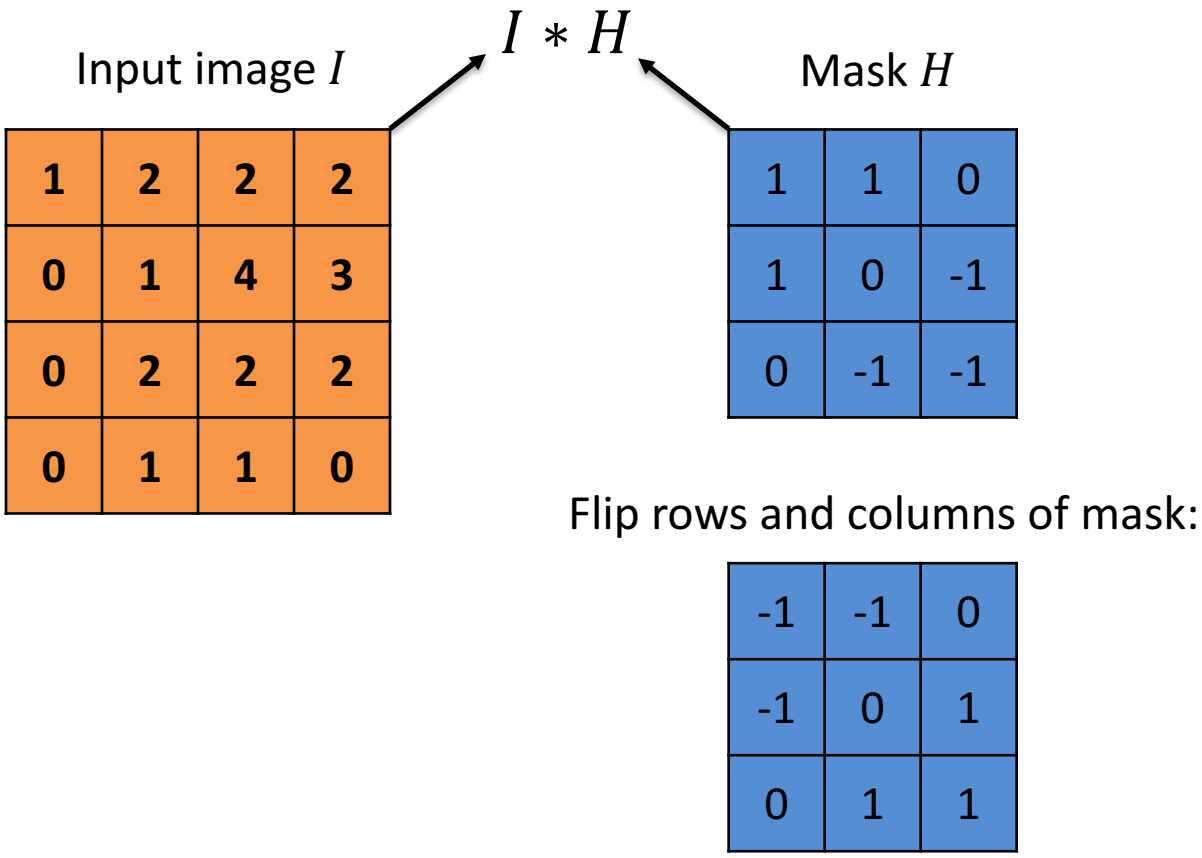
- Why Deep Learning?

- Convolutional Neural Networks
  - What are they?
  - Why are they useful?

- Technicalities of ConvNets
  - Convolution and Pooling
  - Typical architecture

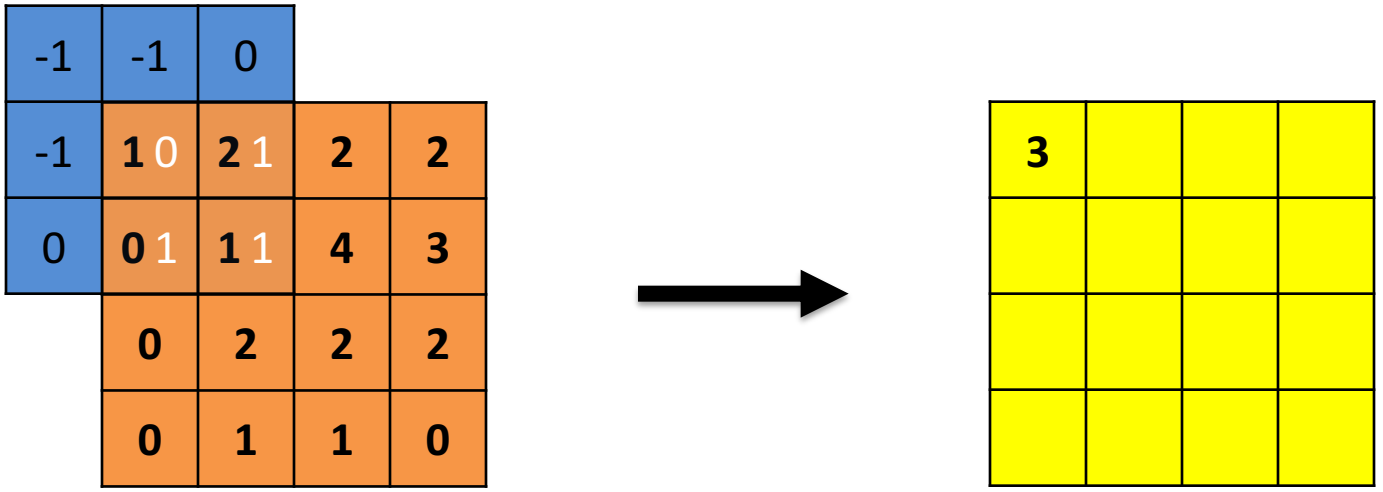
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



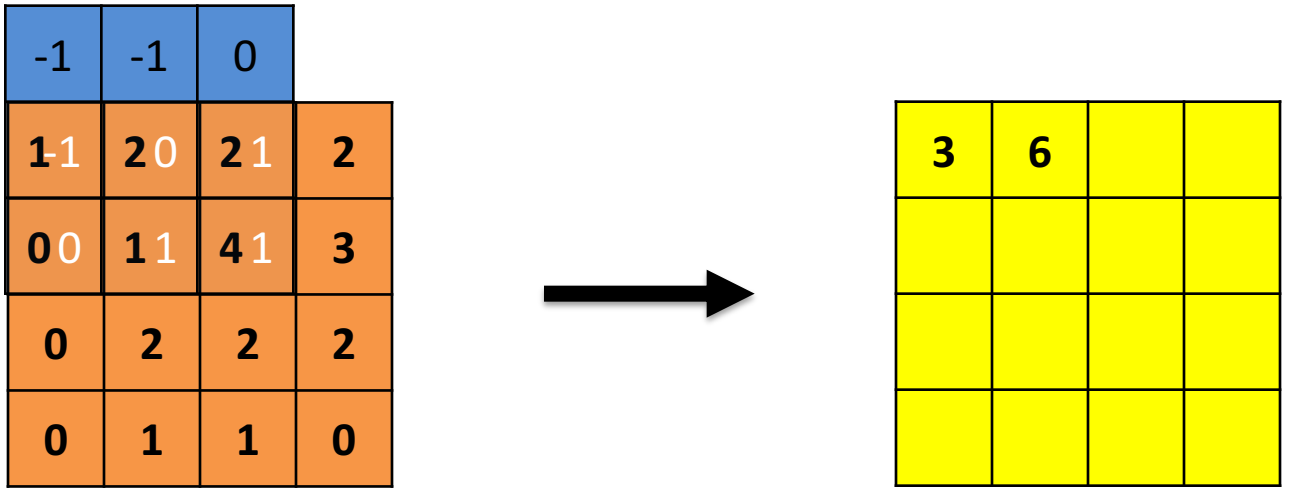
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



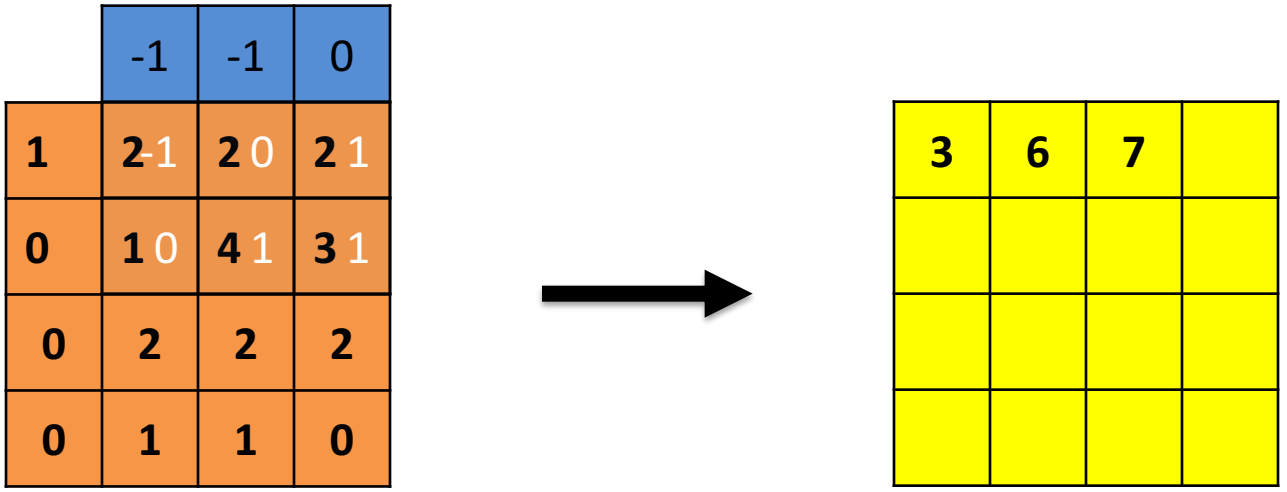
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



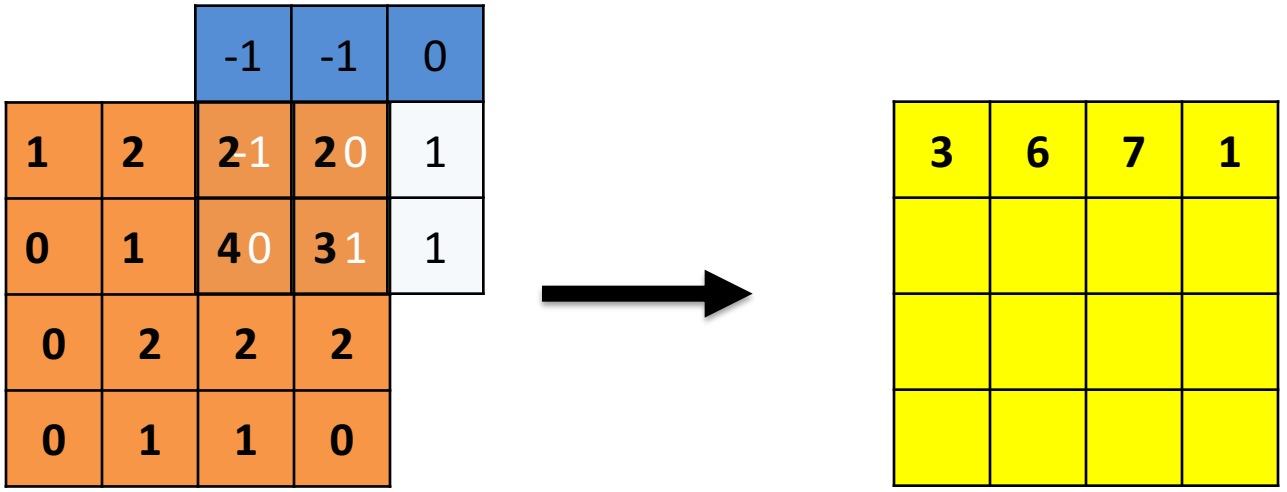
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



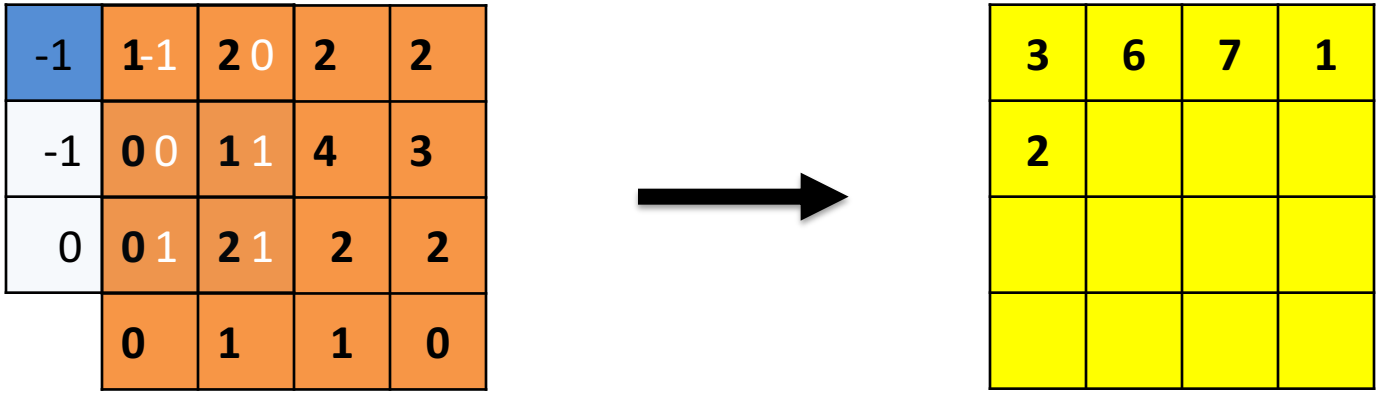
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



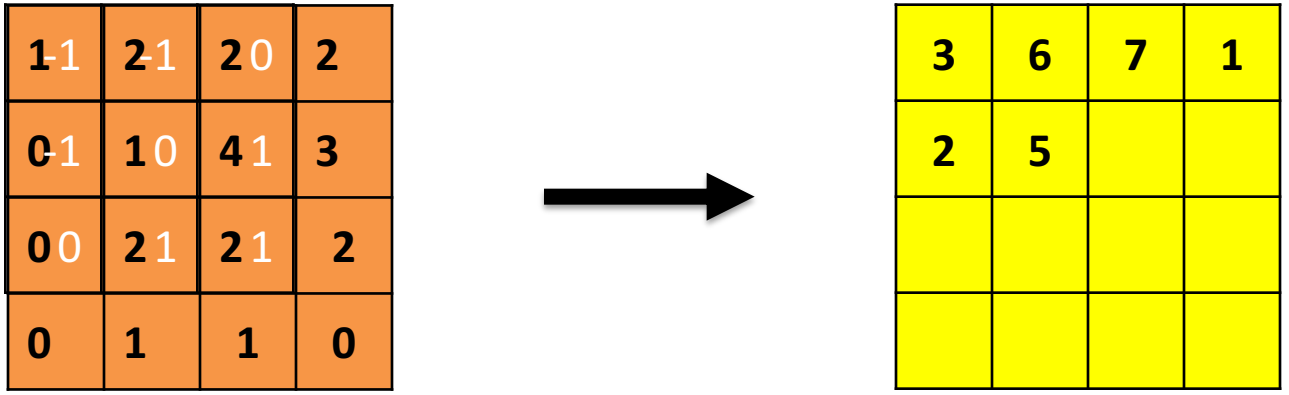
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



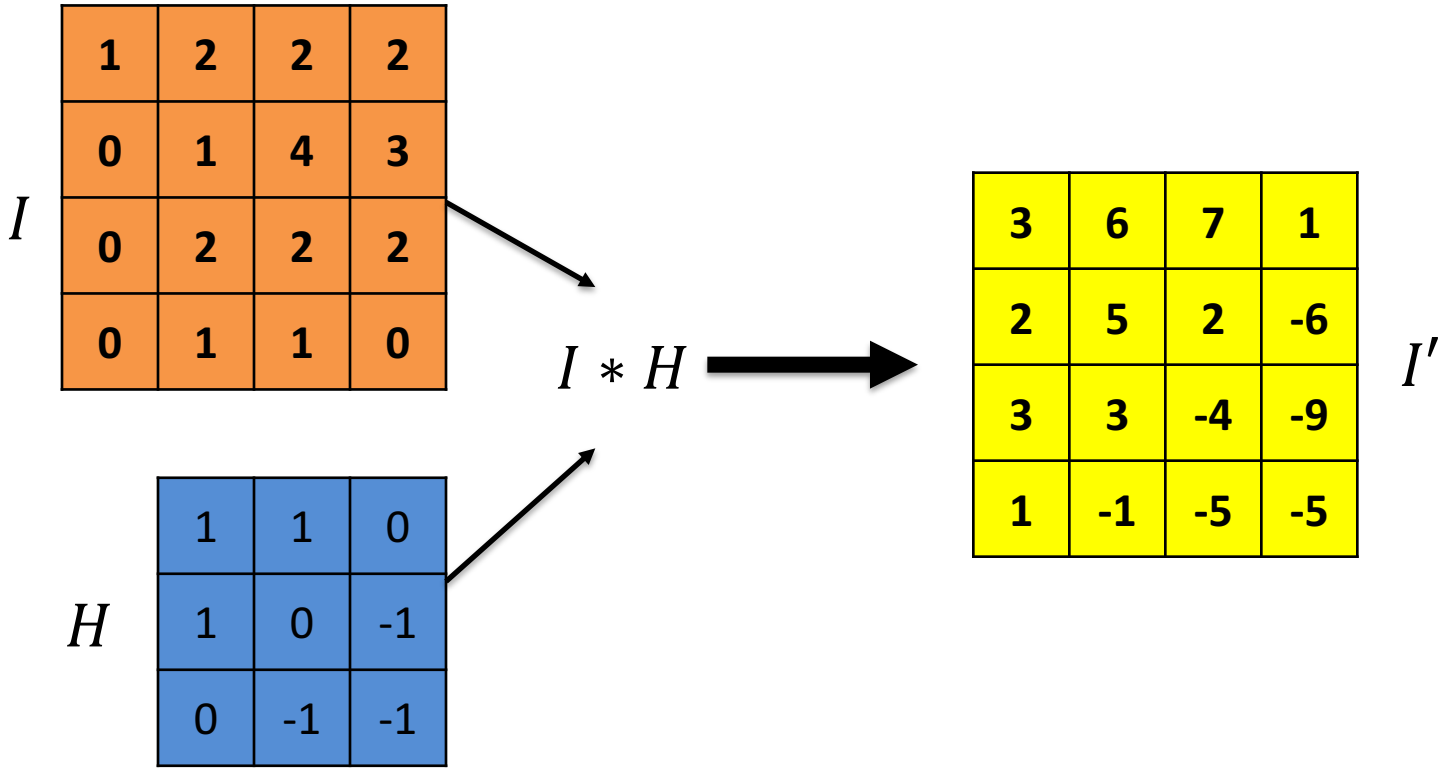
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



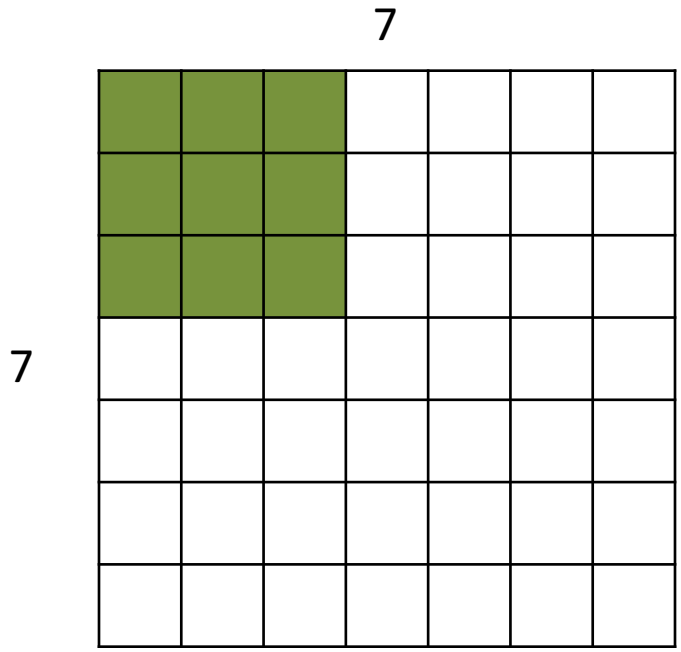
Convolution

- $I'(i,j) = I * H = \sum_{k,l} I(i-k, j-l)H(k,l)$



Convolution

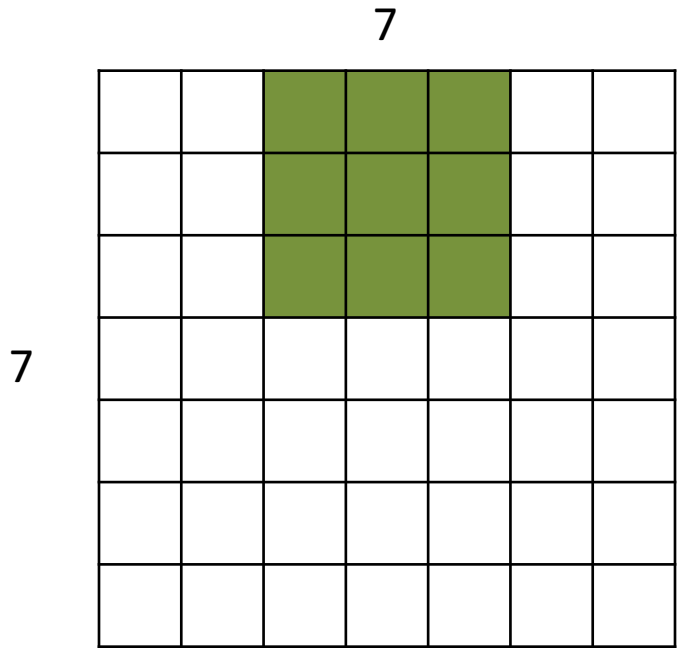
- Stride



7x7 input
assume 3x3 filter
applied with **stride 2**

Convolution

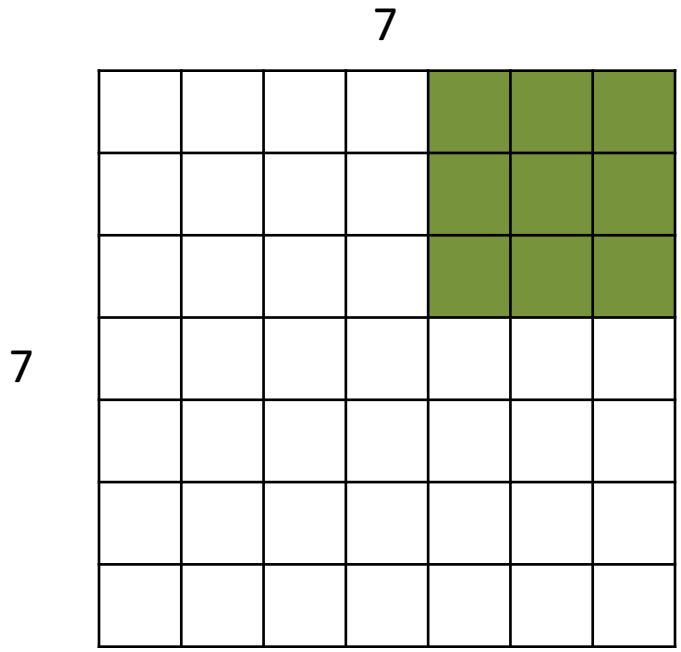
- Stride



7x7 input
assume 3x3 filter
applied with **stride 2**

Convolution

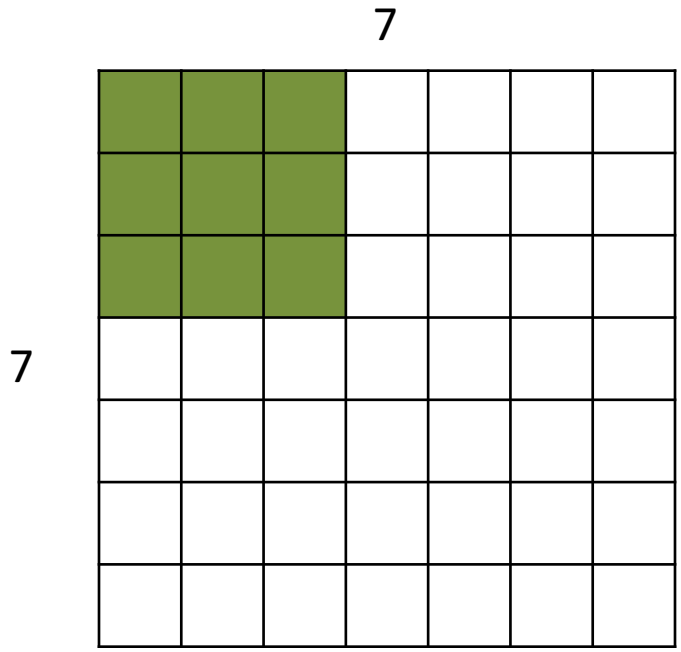
- Stride



7x7 input
assume 3x3 filter
applied with **stride 2**
=> 3x3 output

Convolution

- Stride

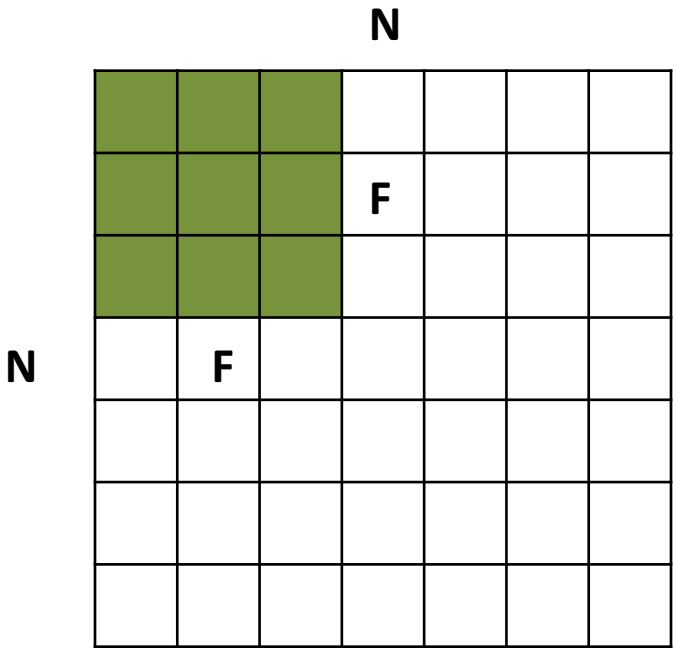


7x7 input
assume 3x3 filter
applied with **stride 3?**

doesn't fit
cannot apply 3x3 filter on 7x7 input with stride 3

Convolution

- Stride

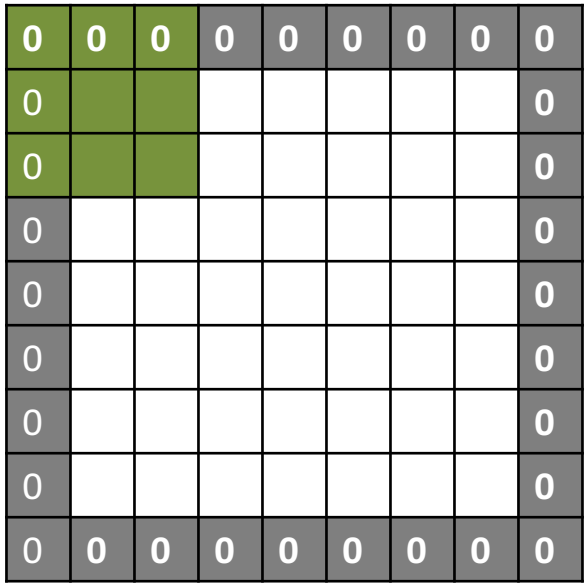


Output size:
(N - F) / stride +1

e.g., N=7, F=3:
stride 1 => (7 - 3) / 1 + 1 = 5
stride 2 => (7 - 3) / 2 + 1 = 3
stride 3 => (7 - 3) / 3 + 1 = 2.33

Convolution

- Padding



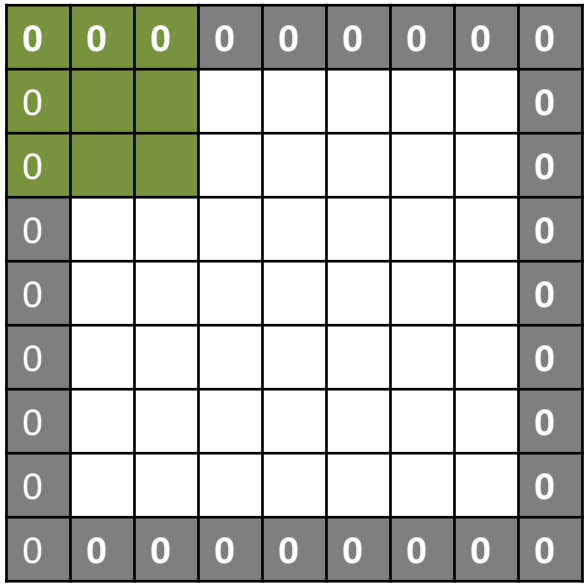
In practice, it is common to zero pad the border

e.g., input 7x7
3x3 filter, applied with stride 1
pad with 1 pixel border => what is the output?

Recall: (N - F) / stride + 1

Convolution

- Padding



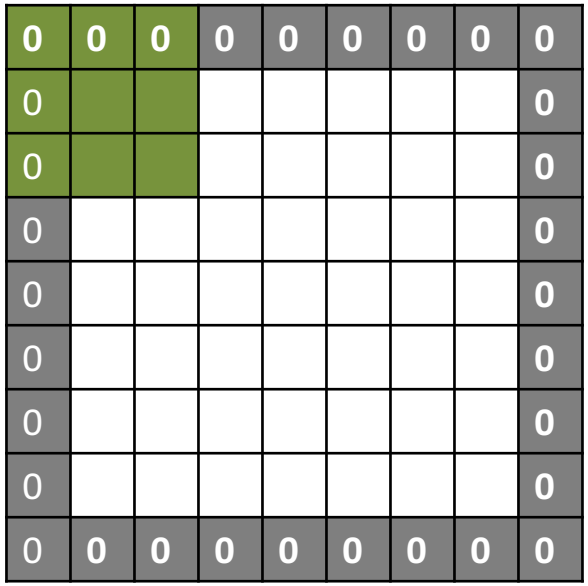
In practice, it is common to zero pad the border

e.g., input 7x7
3x3 filter, applied with stride 1
pad with 1 pixel border => what is the output?

7x7 output!

Convolution

- Padding



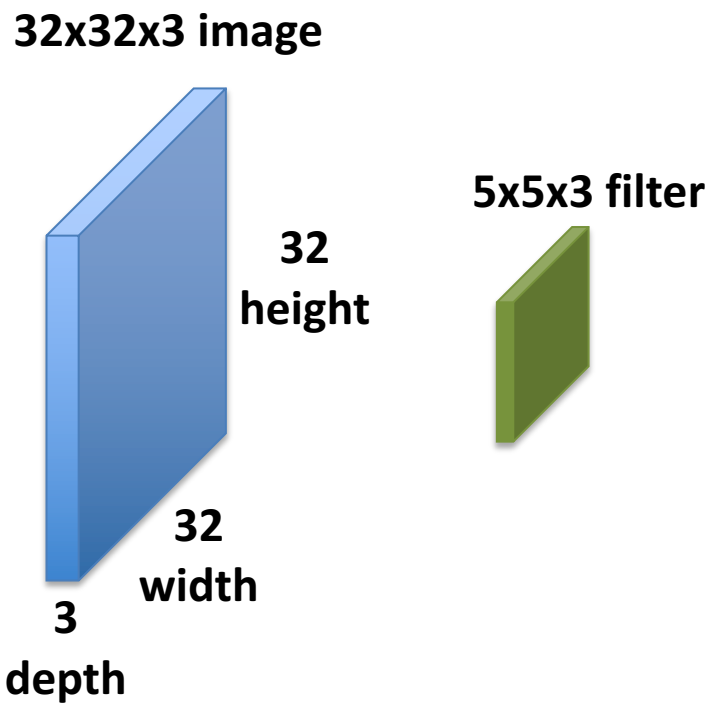
In practice, it is common to zero pad the border

e.g., input 7x7
3x3 filter, applied with stride 1
pad with 1 pixel border => what is the output?

7x7 output!

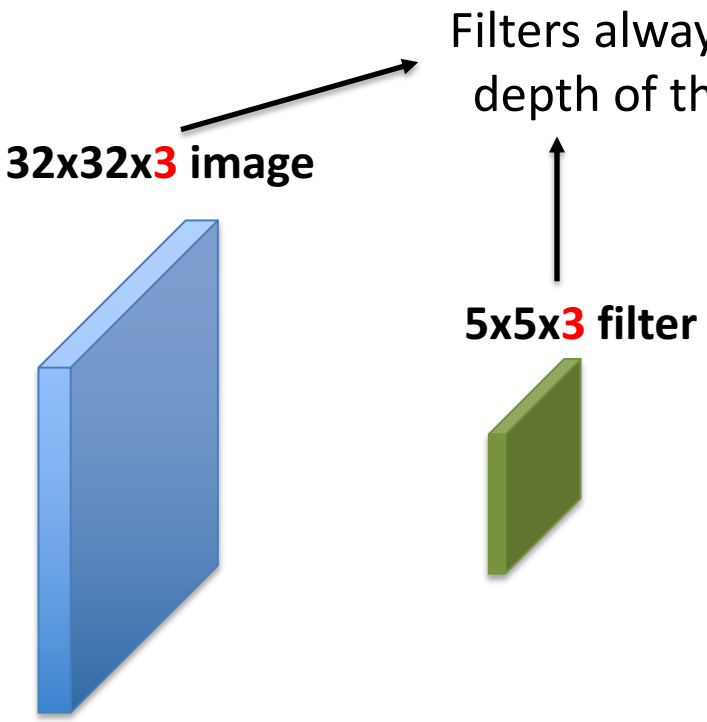
In general, common to see CONV layers with stride 1, filters of size FxF, and zero-padding with (F-1)/2. (will preserve size spatially)
e.g., F=3 => zero pad with 1
F=5 => zero pad with 2
F=7 => zero pad with 3

Convolution layer



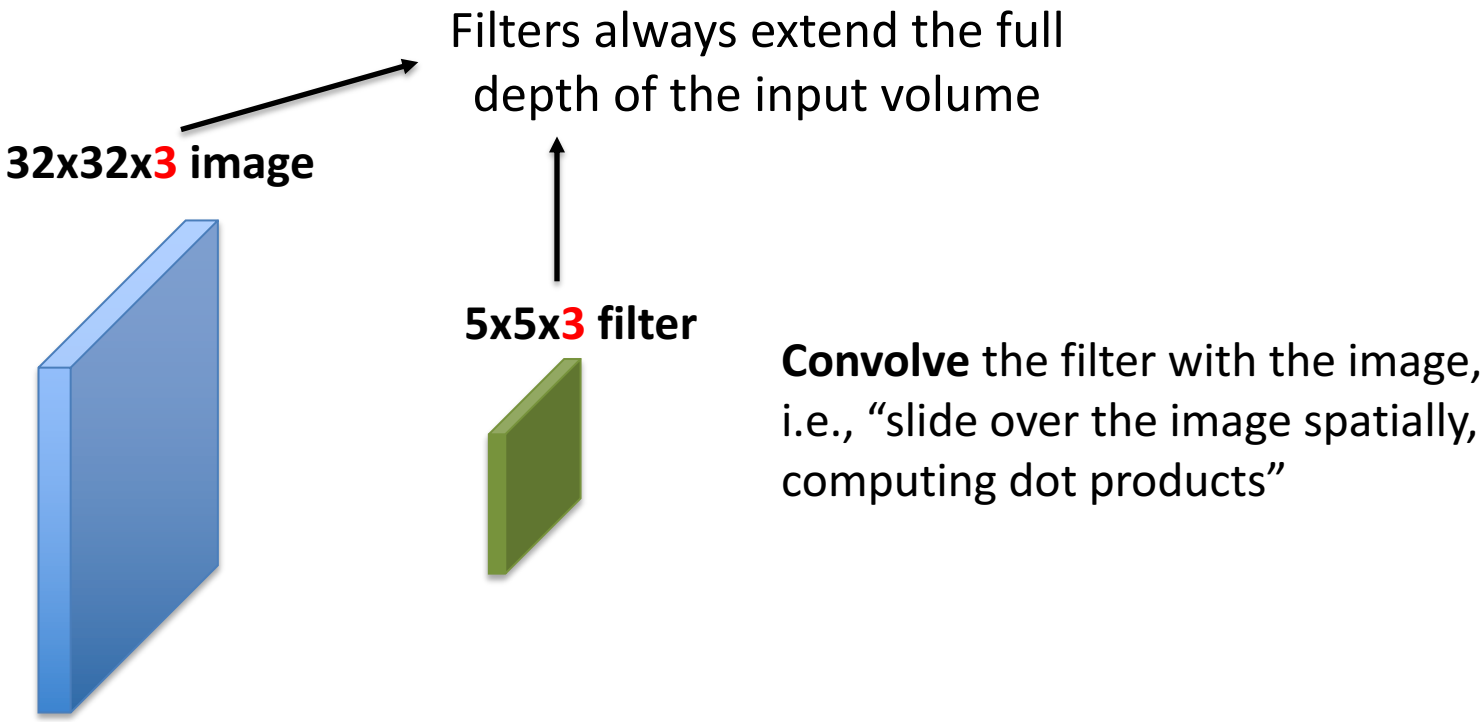
Convolve the filter with the image, i.e., “slide over the image spatially, computing dot products”

Convolution layer



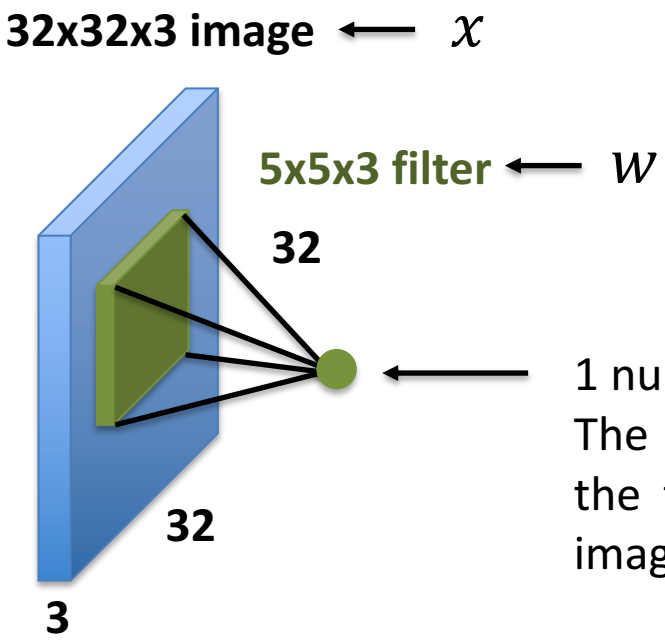
Convolve the filter with the image, i.e., “slide over the image spatially, computing dot products”

Convolution layer



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

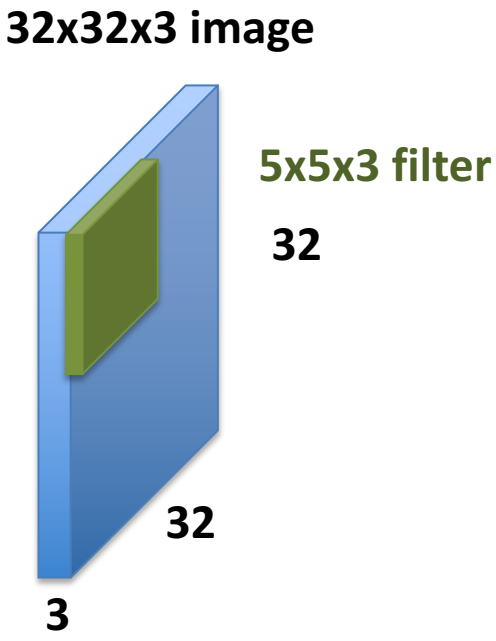
Convolution layer



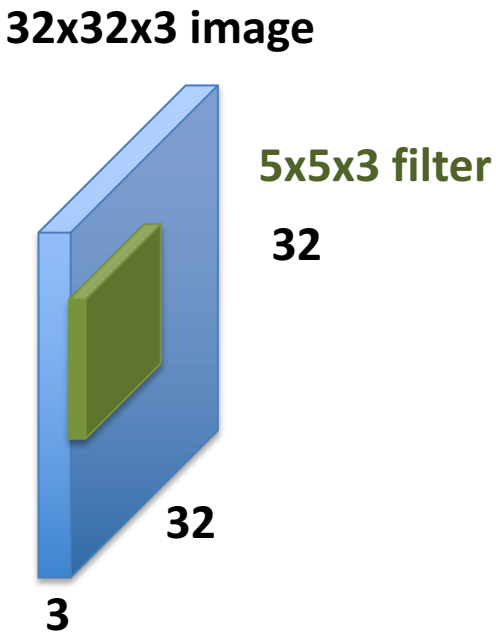
1 number:
The result of taking a dot product between the filter and a small 5x5x3 patch of the image (i.e., 5*5*3 = 75-d dot product +bias)

$$w^T x + b$$

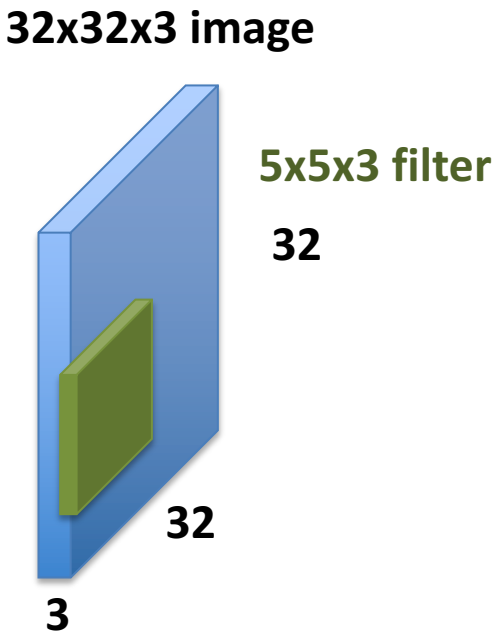
Convolution layer



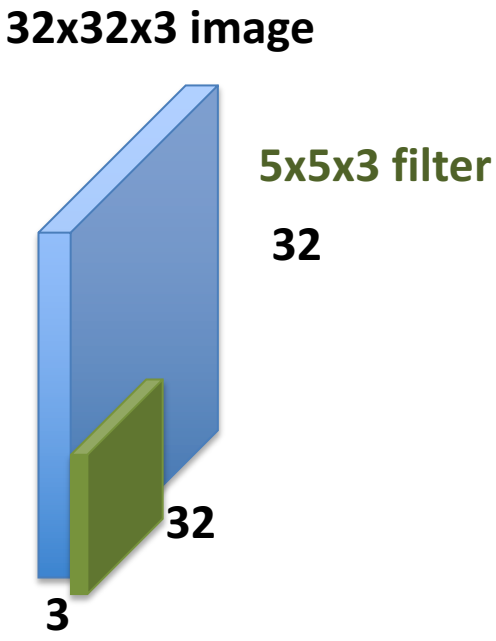
Convolution layer



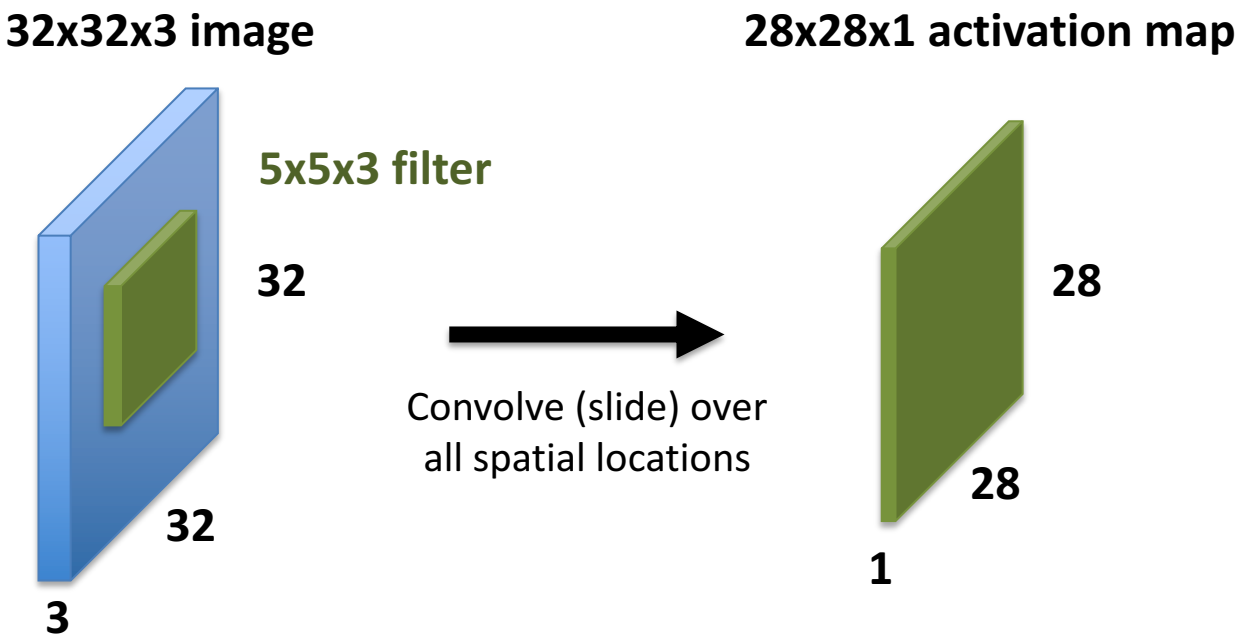
Convolution layer



Convolution layer

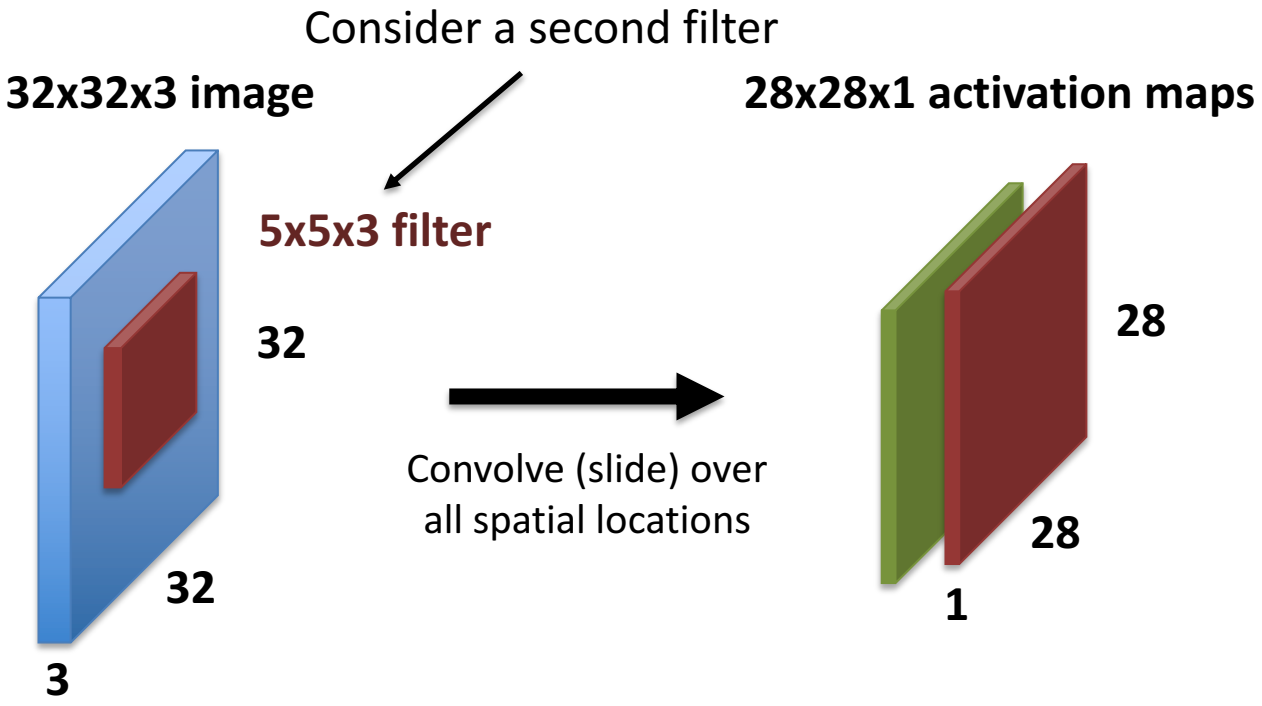


Convolution layer



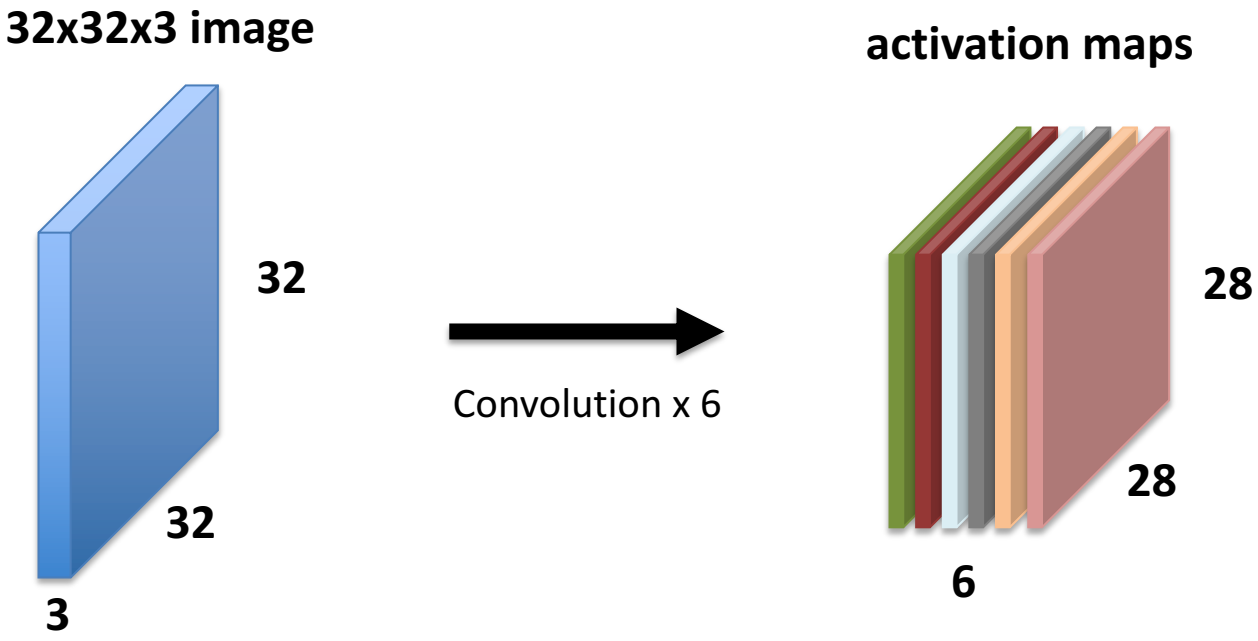
Recall: (N - F) / stride + 1

Convolution layer



Recall: (N - F) / stride + 1

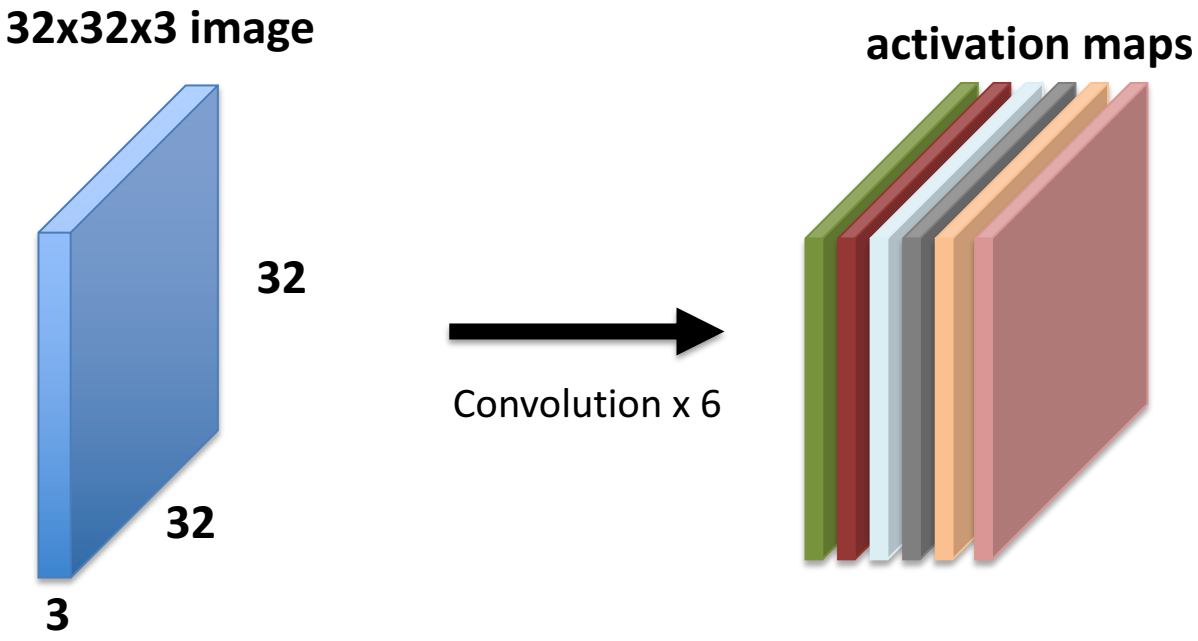
Convolution layer



If we have 6 5x5x3 filters, we get 6 separate activation maps.

And we stack these up to get a ”new image” of size 28x28x6.

Convolution layer



If we have 6 5x5x3 filters with stride 1, pad 2, output volume size?

Spatially, (32 + 2x2 - 5)/1 +1 =32, so output volume size: 32x32x6

Convolution layer

- Previous layer: a volume of size $W_1 \times H_1 \times D_1$
- Four hyperparameters are required:
  - Number of filters K
  - Filter size F
  - The stride S
  - The amount of zero padding P
- Next layer: a volume of size $W_2 \times H_2 \times D_2$
  - $W_2 = (W_1 - F + 2P) / S + 1$
  - $H_2 = (H_1 - F + 2P) / S + 1$
  - $D_2 = K$

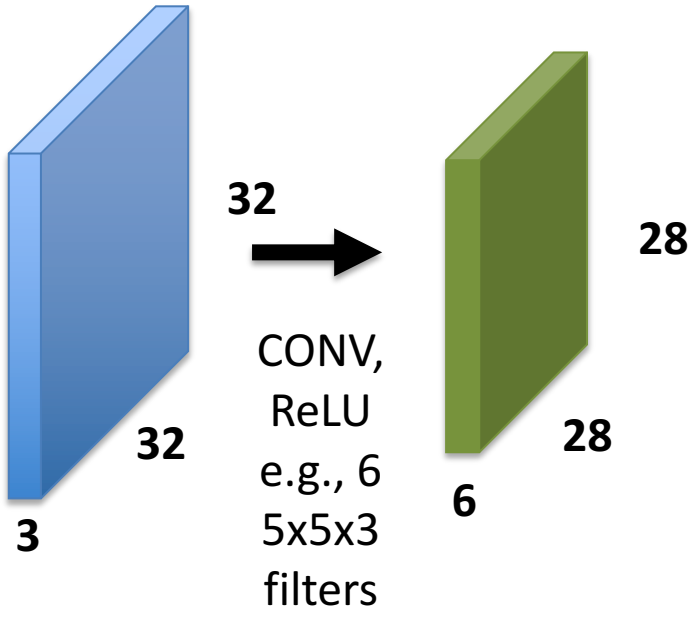
Common settings:

K = (powers of 2, e.g., 32, 64, 128, 512)

- F =3, S=1, P=1
- F =5, S=1, P=2
- F =1, S=1, P=0

ConvNet

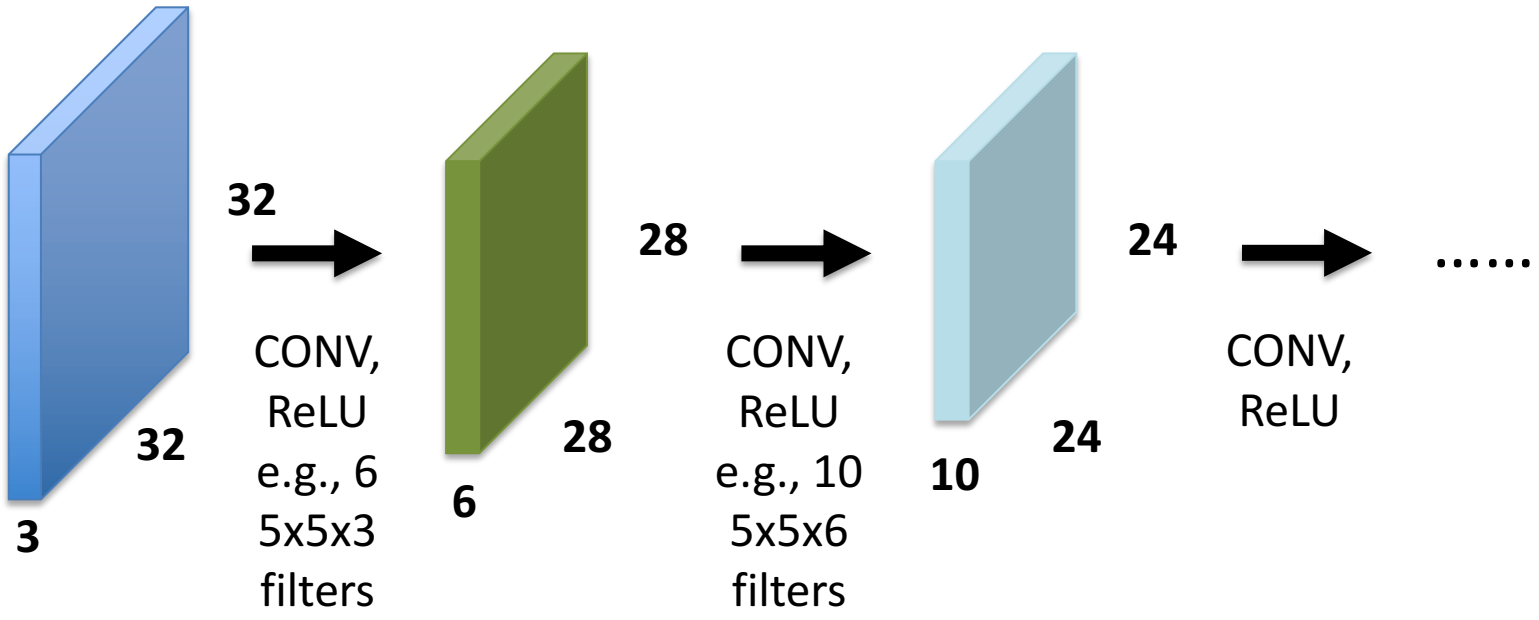
(*Convolutional Nuerual Networks*)



ConvNet is a sequence of convolution layers, interspersed with activation functions

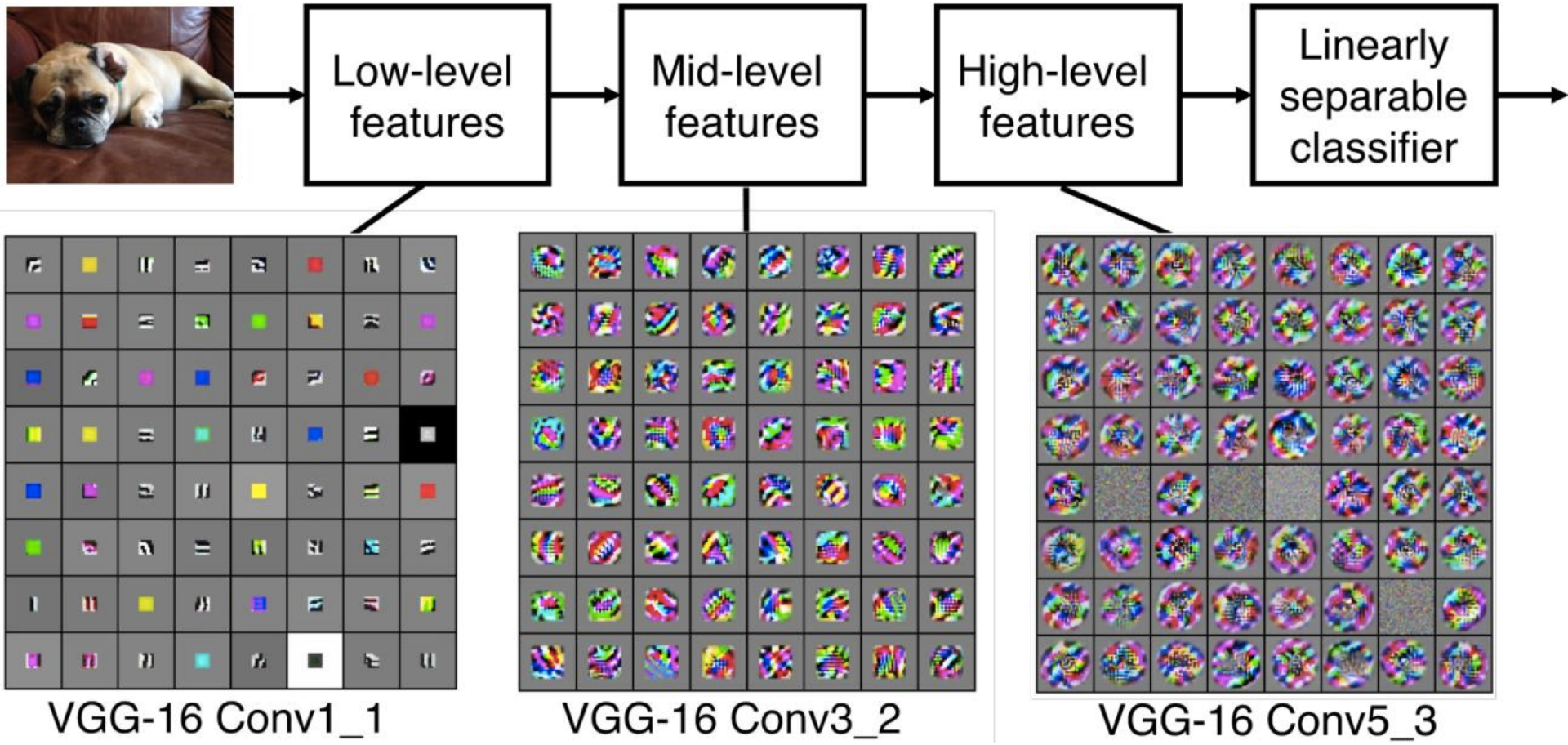
ConvNet

(*Convolutional Nuerual Networks*)



ConvNet is a sequence of convolution layers, interspersed with activation functions

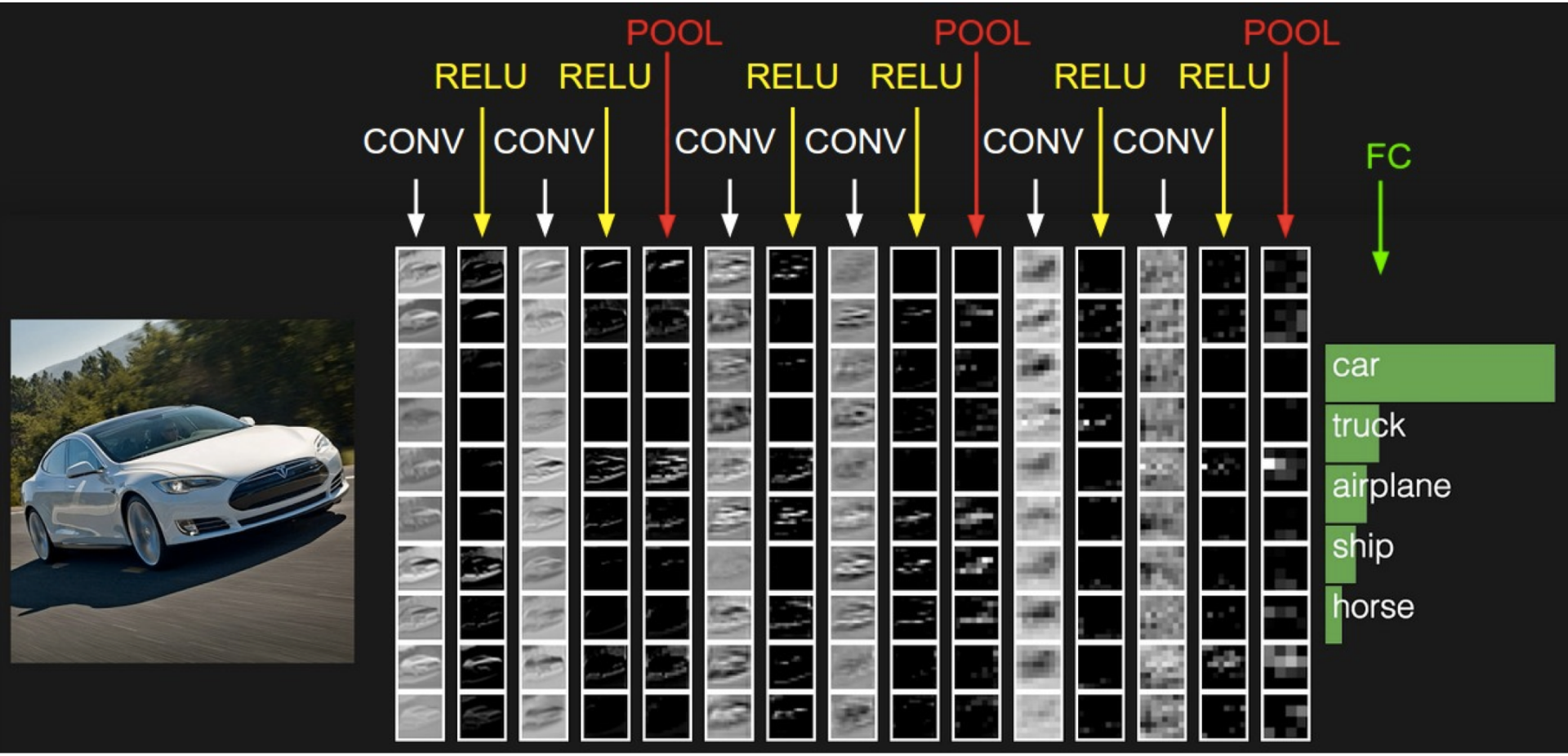
ConvNet



Zeiler and Fergus. “Visualizing and understanding convolutional networks.” *ECCV,* 2014.

ConvNet

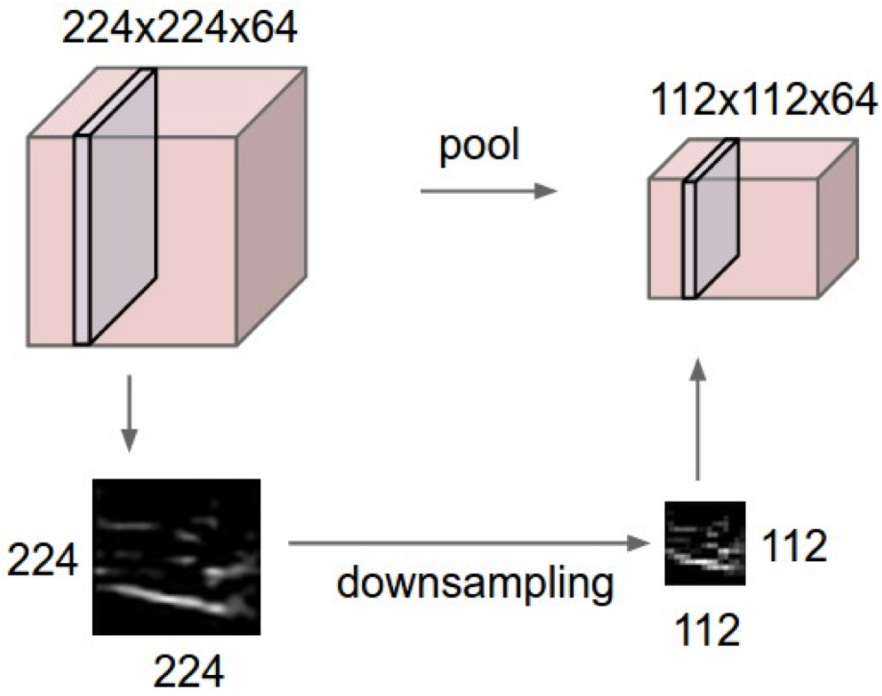
Two more types of layers: pooling and fully connected (FC)



ConvNet

Pooling layer

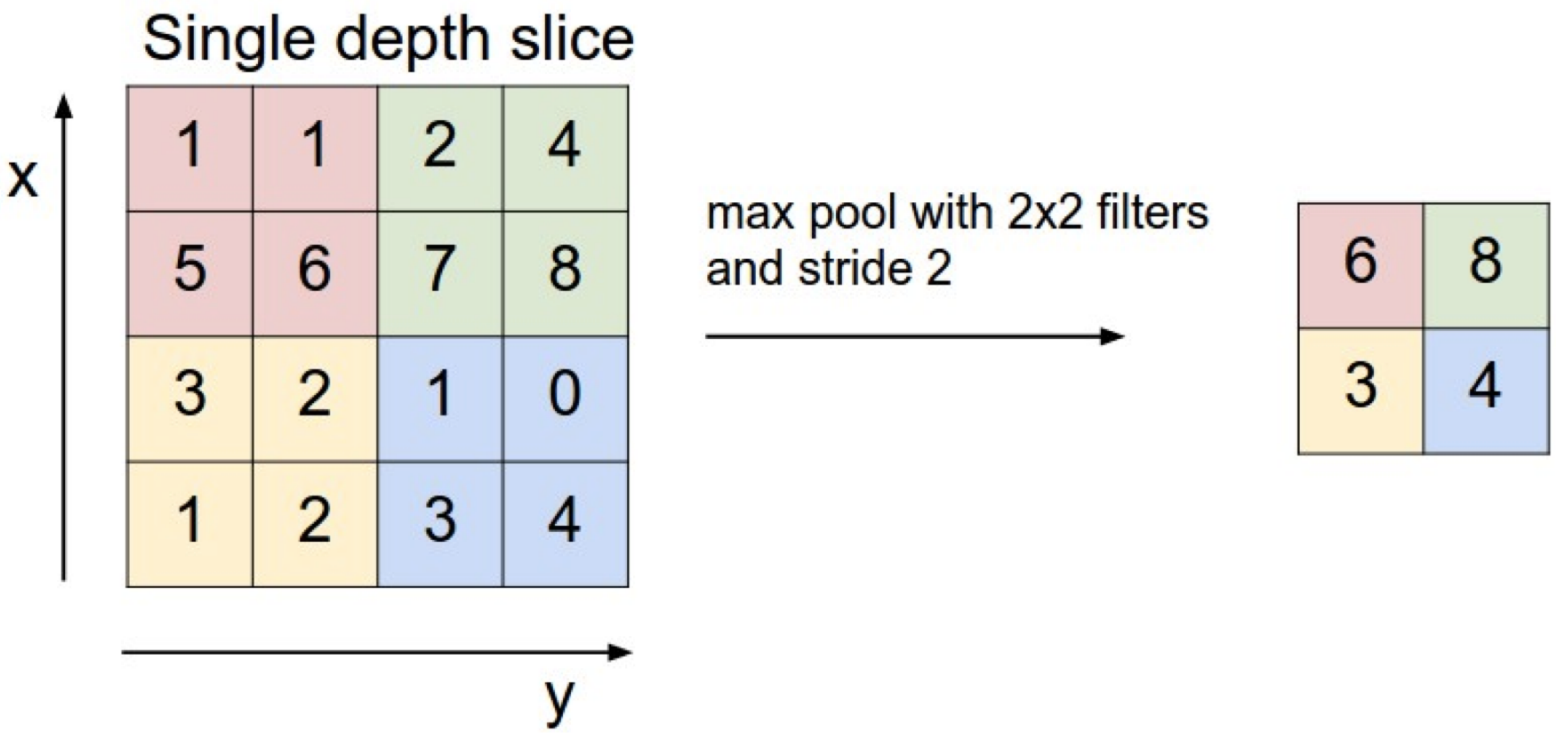
- Make the representations smaller and more manageable
- Operates over each activation map independently



ConvNet

Pooling layer

- Max pooling



ConvNet

Pooling layer

- Previous layer: a volume of size W_1 x H_1 x D_1
- Two hyperparameters are required:
  - Filter size F
  - The stride S
- Next layer: a volume of size W_2 x H_2 x D_2
  - $W_2 = (W_1 - F) / S + 1$
  - $H_2 = (H_1 - F) / S + 1$
  - $D_2 = D_1$

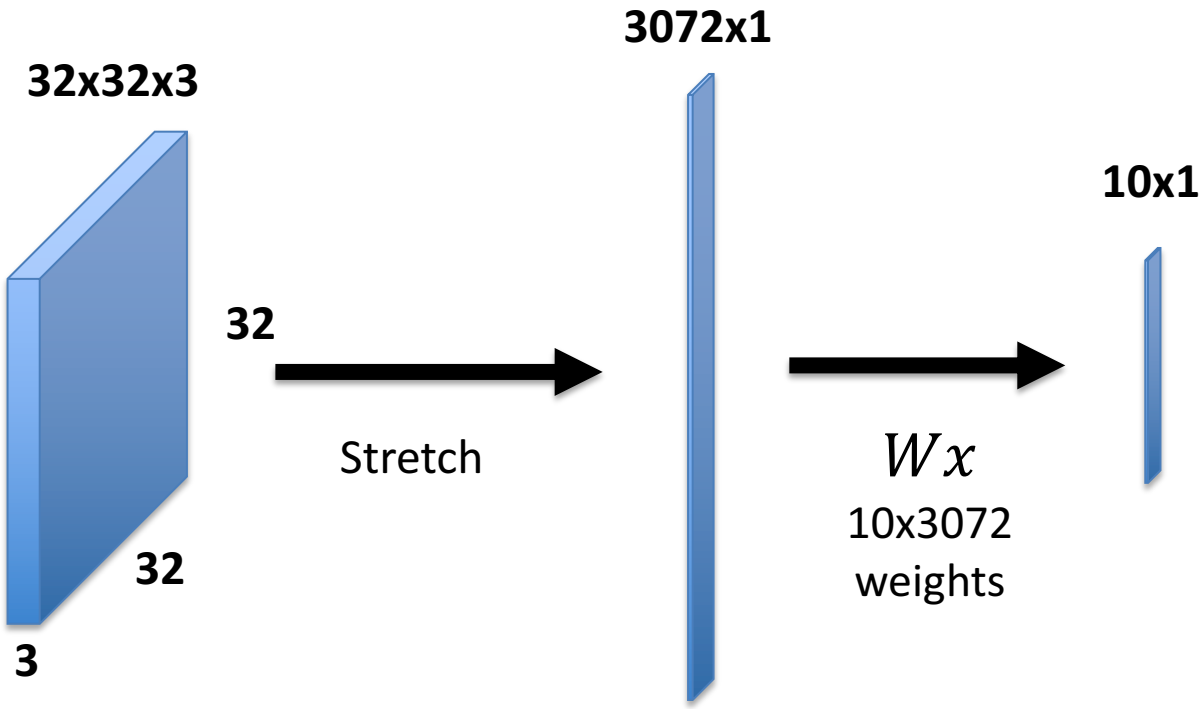
Common settings:
- F =2, S=2
- F =3, S=2

It is not common to use zero-padding for Pooling layers

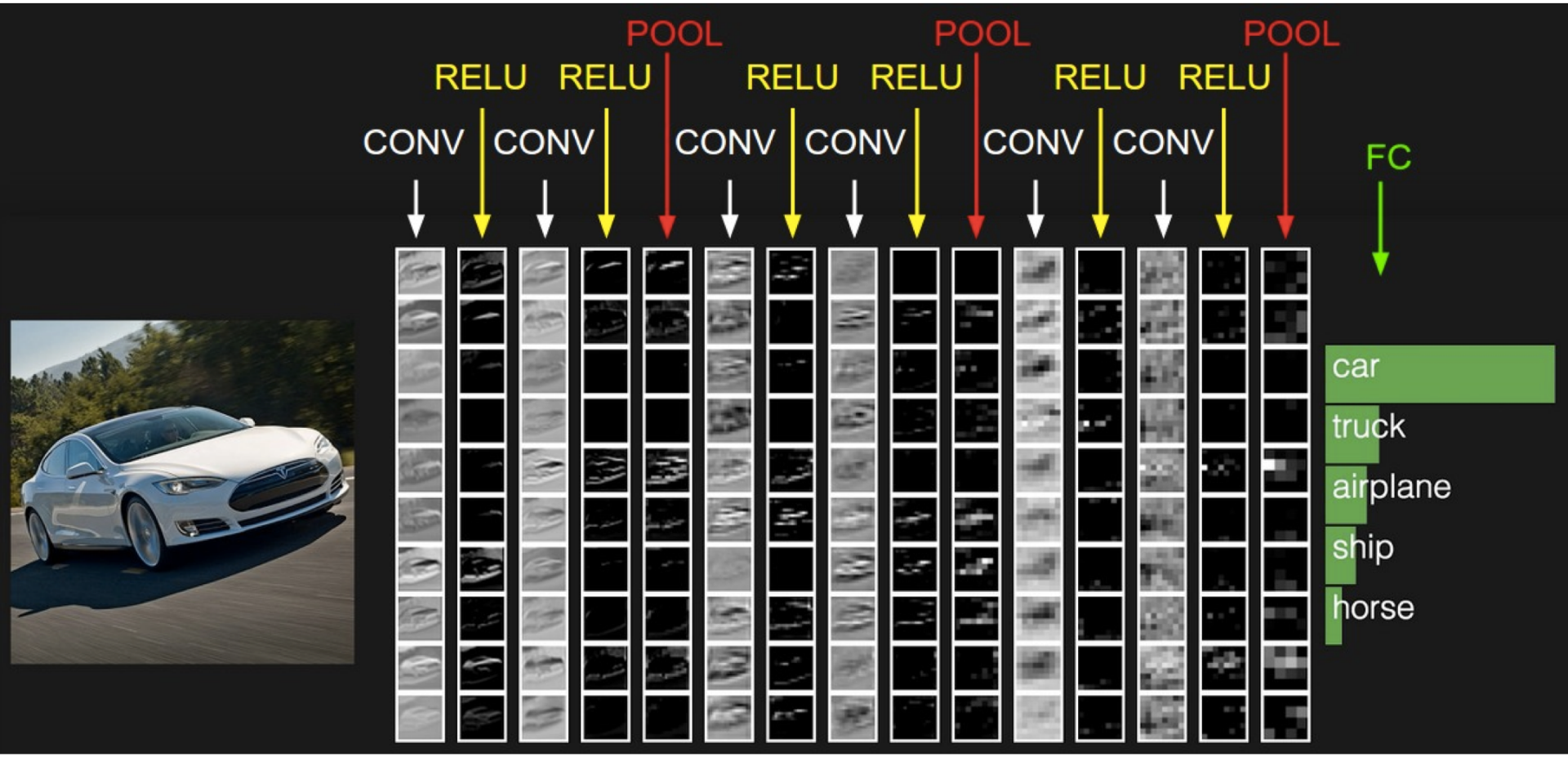
ConvNet

Fully Connected (FC) layer

- Contains neurons that connect to the entire input volume, as in ordinary neural networks



ConvNet


POOL
POOL
POOL
RELU
RELU
RELU
RELU
RELU
RELU
CONV
CONV
CONV
CONV
CONV
CONV
FC
car
truck
airplane
ship
horse