CS361

(Software Engineering Program)

Artificial Intelligence II - Applied Machine Learning

Lecture 9

A Basic Introduction to Deep Learning & Convolutional Neural Networks [CNNs]

Amr S. Ghoneim

(Assistant Professor, Computer Science Dept.)

Helwan University

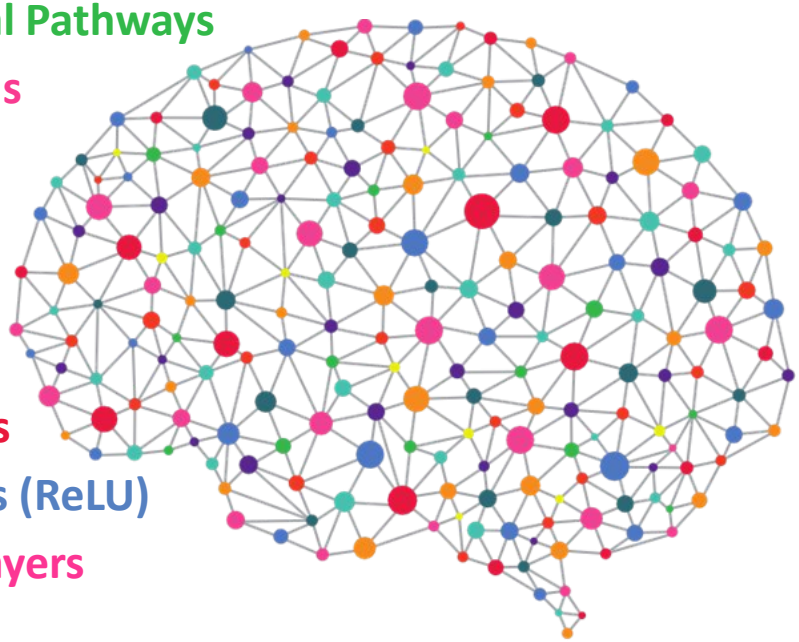
Fall 2019

Lecture is based on its counterparts in the following courses (& the following resources):

- *CS231n: Convolutional Neural Networks for Visual Recognition,* **Stanford University (California USA)**, Stanford School of Engineering.
- *CS 898: Deep Learning and Its Applications,* **University of Waterloo (Waterloo - Ontario, Canada),** David R. Cheriton School of Computer Science
- *Introduction to Deep Learning,* **UIUC University of Illinois at Urbana - Champaign (Illinois USA)**, Computer Science Department.
- *CS231A: Computer Vision, From 3D Reconstruction to Recognition*, **Stanford University (California USA)**, Computer Science Dept. - The Computational Vision & Geometry Lab (CVGL)

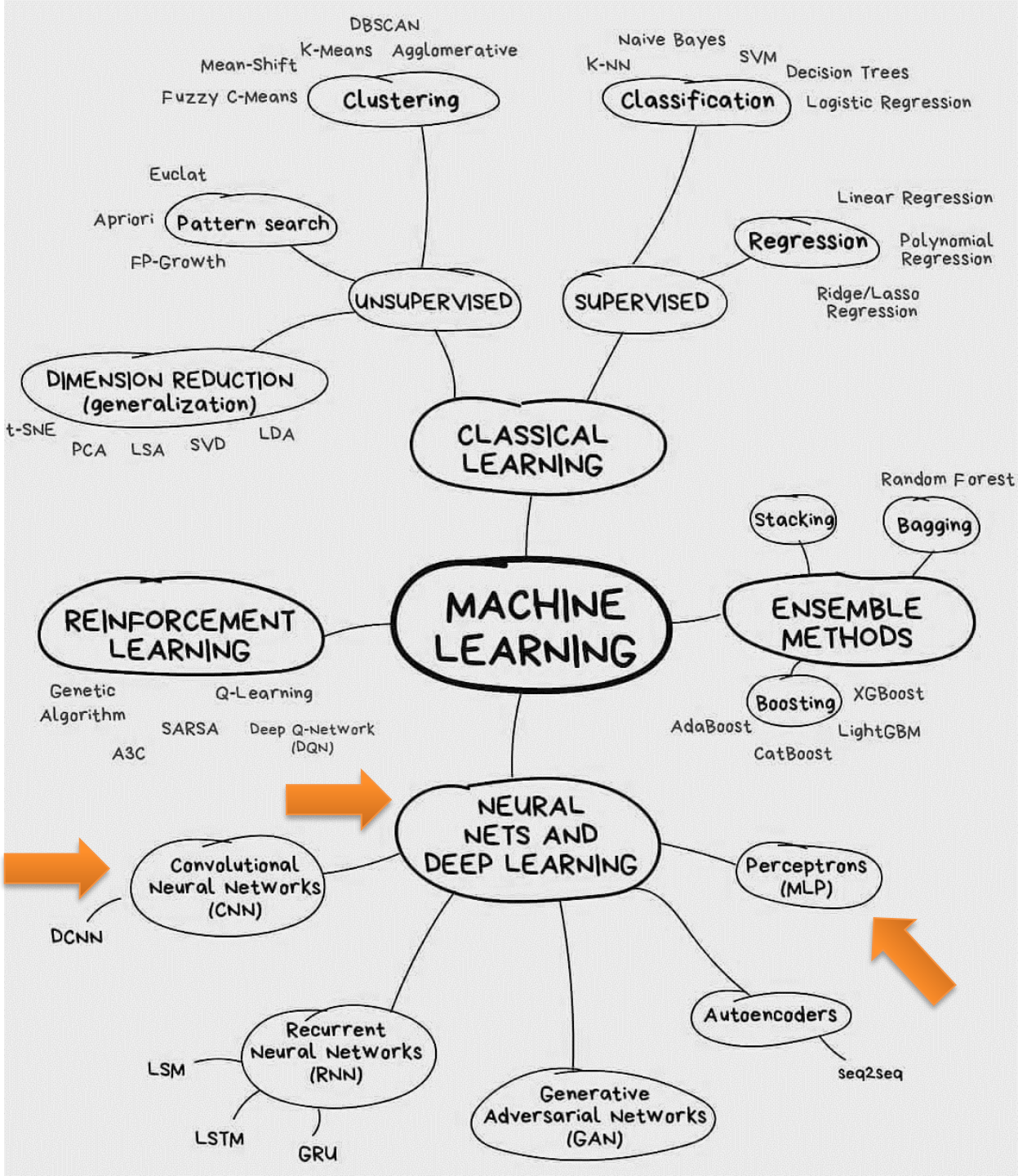
Today's Key Concepts

- Machine Learning
  - Deep Learning
- CNNs: A Bit of History
  - First Strong Results
  - Hierarchical Organization of early Visual Pathways
  - ImageNet Classification with Deep CNNs
  - AlexNet
  - LeNet-5
- Basic Concepts of CNNs
  - Overall CNN Architecture
  - Convolution Layer & Convolution Filters
  - Activation Maps & Activation Functions (ReLU)
  - ConvNet: a Sequence of Convolution Layers
  - Spatial Dimensions
  - Pooling Layer & Padding the Borders
  - Fully Connected Layer (FC layer) & Flattening



Machine Learning?

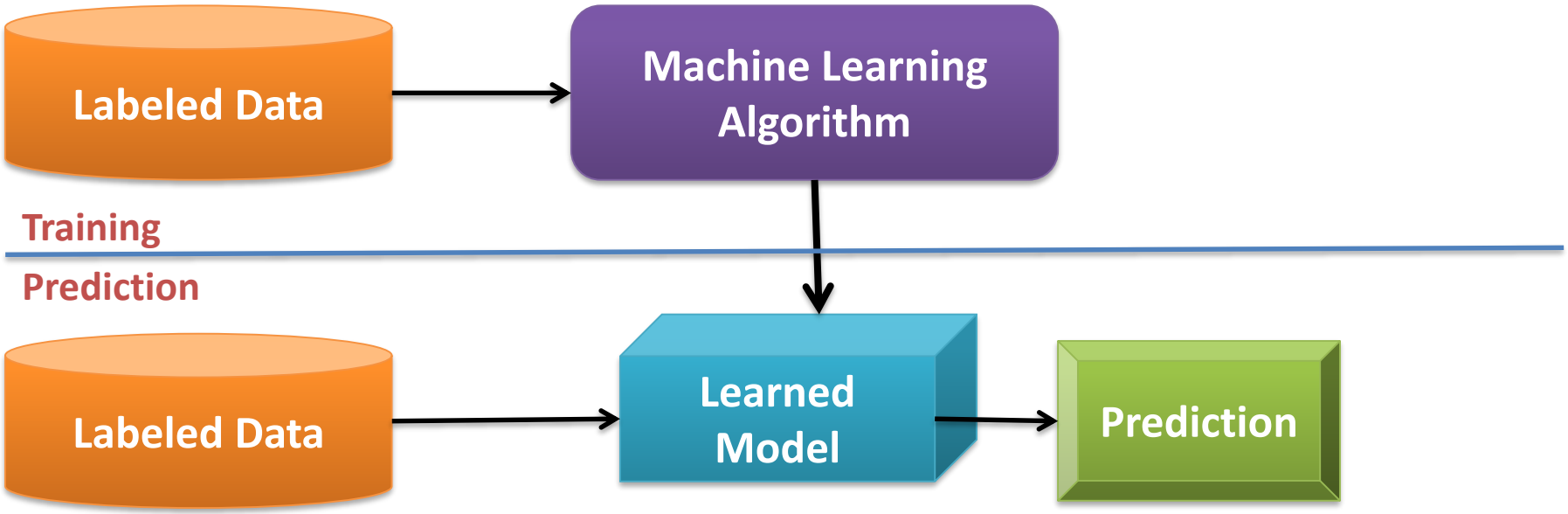
{*Artificial Intelligence*}
Machine Learning Map



Deep Learning

Recap: Machine Learning Basics

Machine learning is a field of computer science that gives computers the ability to **learn without being explicitly programmed**



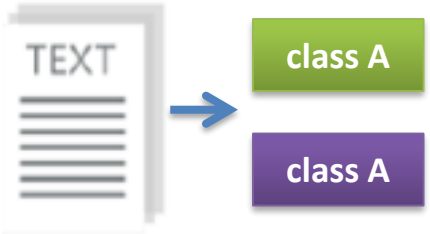
Methods that can learn from and make predictions on data.

Recap: Types of Learning

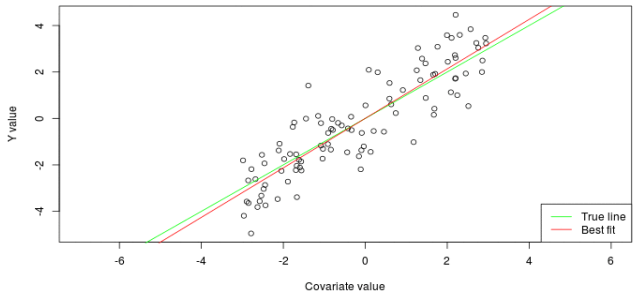
Supervised: Learning with a **labeled training** set.
Example: email *classification* with already labeled emails.

Unsupervised: Discover **patterns** in **unlabeled** data.
Example: *cluster* similar documents based on text.

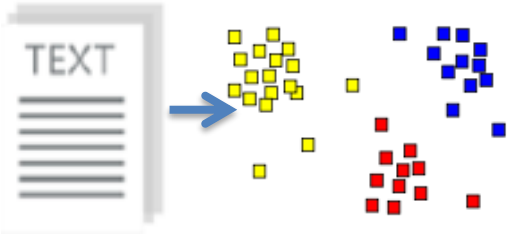
Reinforcement learning: learn to **act** based on **feedback/reward**
Example: learn to play Go, reward: *win or lose*.



Classification



Regression

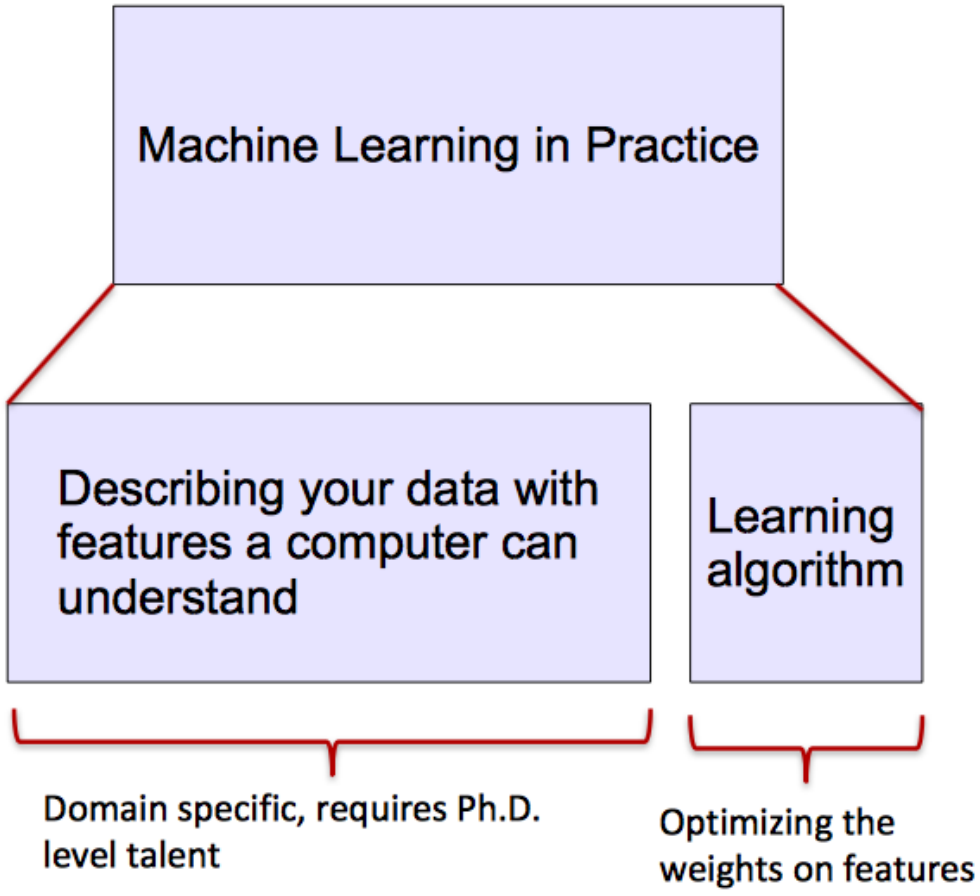


Clustering

Anomaly Detection
Sequence labeling
…

Recap: ML vs. Deep Learning

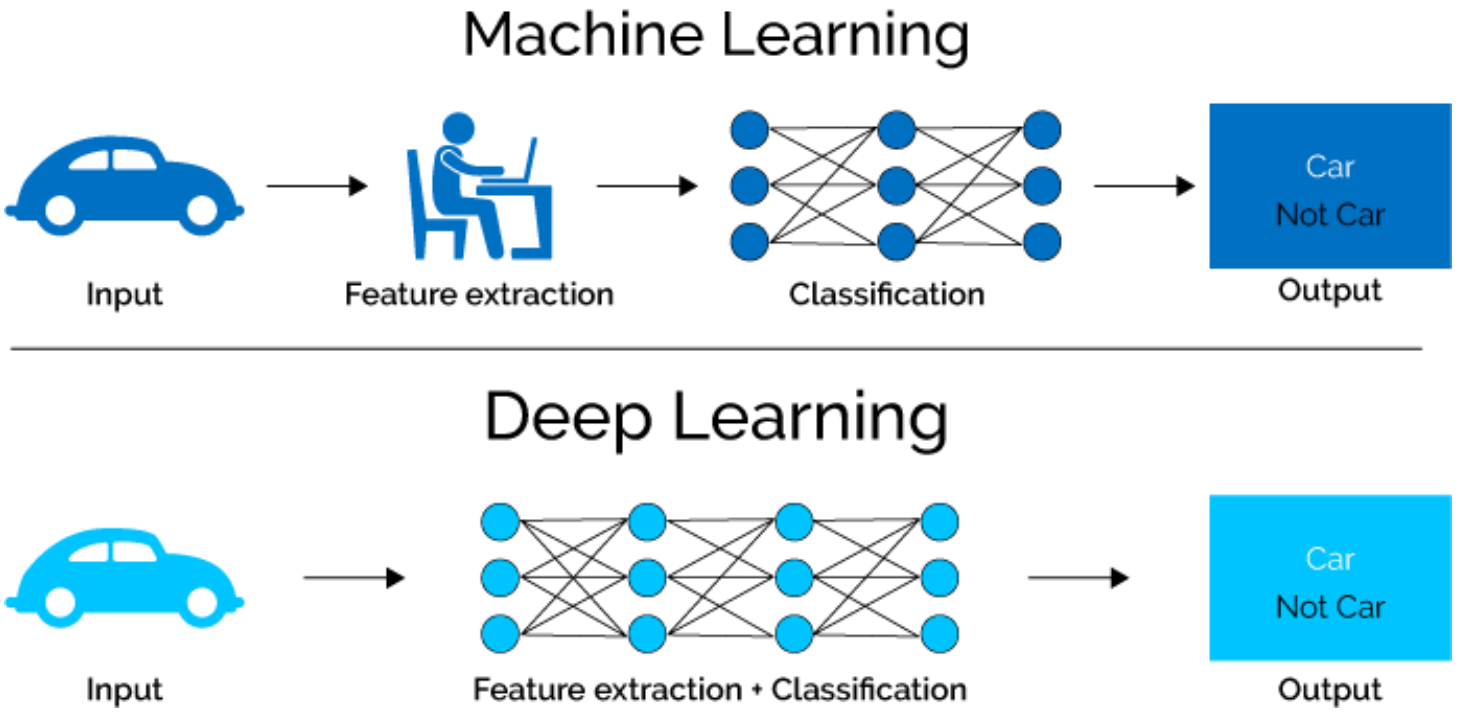
- Most machine learning methods work well because of **human-designed representations** and **input features**.
- ML becomes just **optimizing weights** to best make a final prediction.



Feature	NER
Current Word	✓
Previous Word	✓
Next Word	✓
Current Word Character n-gram	all
Current POS Tag	✓
Surrounding POS Tag Sequence	✓
Current Word Shape	✓
Surrounding Word Shape Sequence	✓
Presence of Word in Left Window	size 4
Presence of Word in Right Window	size 4

Recap: What is Deep Learning (DL) ?

- A machine learning subfield of learning **representations** of data.
- Exceptionally effective at **learning patterns**.
- Deep learning algorithms attempt to learn *(multiple levels of)* representation by using a **hierarchy of multiple layers**.
- If you provide the system **tons of information**, it begins to understand it and respond in useful ways.



https://www.xenonstack.com/blog/static/public/uploads/media/machine-learning-vs-deep-learning.png

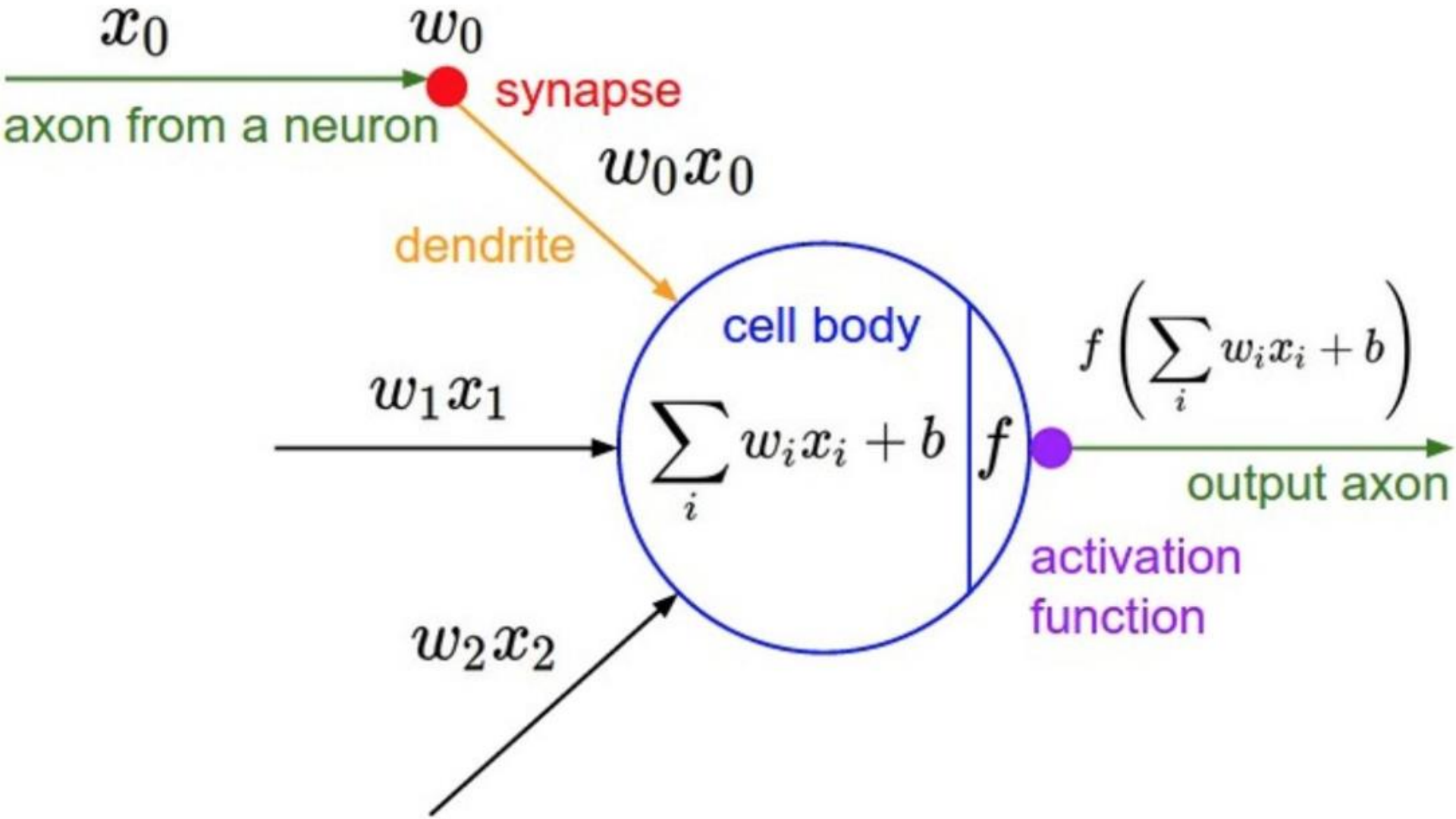
Convolutional Neural Networks (CNN, ConvNet) is a class of deep, feed-forward *(not recurrent)* artificial neural networks that are applied "mostly" to analyzing visual imagery.

Convolutional Neural Networks CNNs

A BRIEF HISTORY & APPLICATIONS

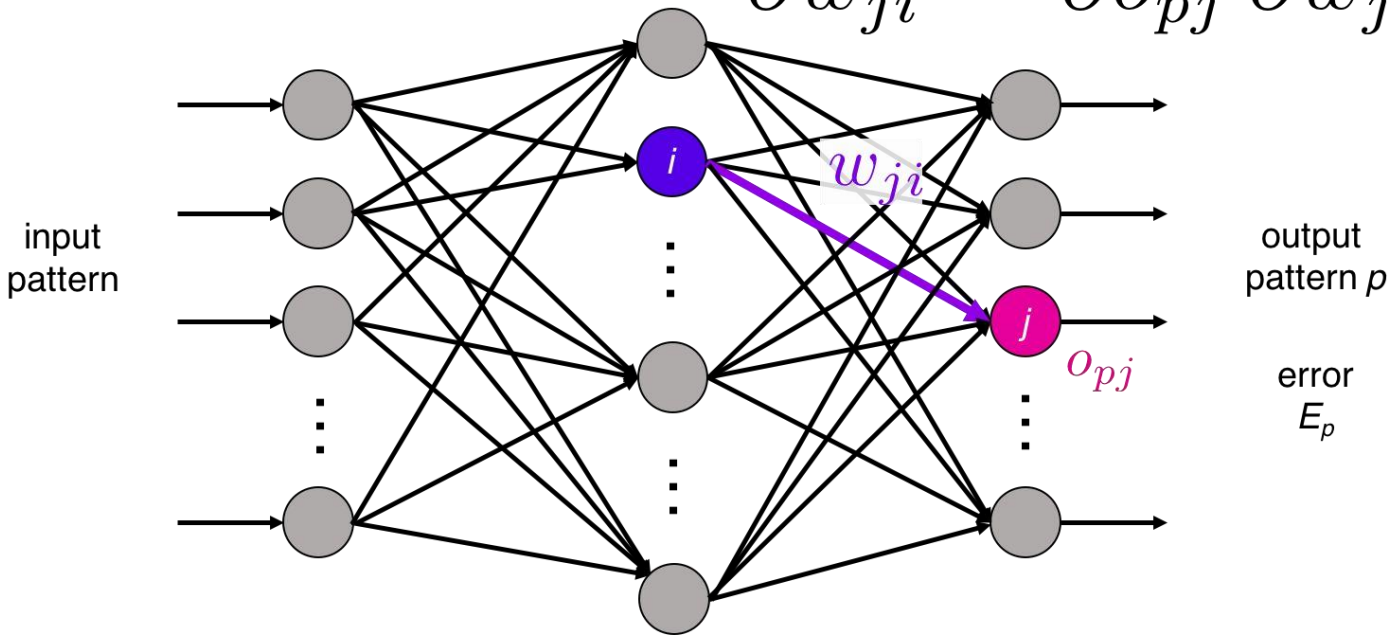
Recap: An Artificial Neuron

A Bit of History..



A Bit of History..

$$\frac{\partial E_p}{\partial w_{ji}} = \frac{\partial E_p}{\partial o_{pj}} \frac{\partial o_{pj}}{\partial w_{ji}}$$



Rumelhart et al., 1986: First time back-propagation became popular.

Illustration of Rumelhart et al., 1986 by Lane McIntosh, copyright CS231n 2017

First Strong Results ..
A Bit of History..

Acoustic Modeling using Deep Belief Networks
Abdel-rahman Mohamed, George Dahl, Geoffrey Hinton, 2010
Context-Dependent Pre-trained Deep Neural Networks for Large Vocabulary Speech Recognition
George Dahl, Dong Yu, Li Deng, Alex Acero, 2012

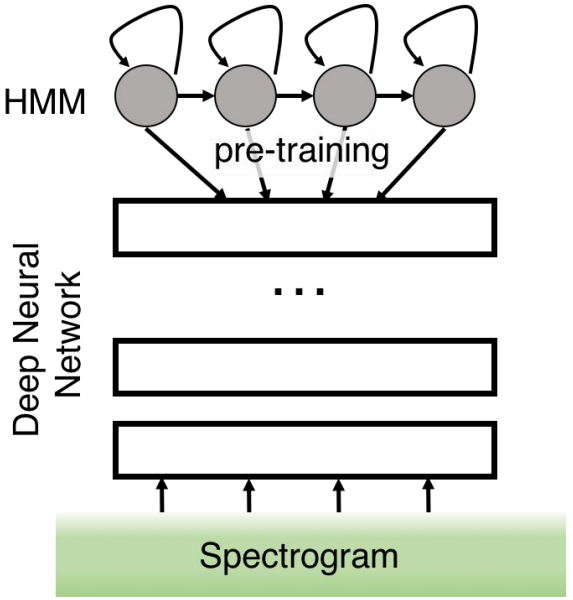
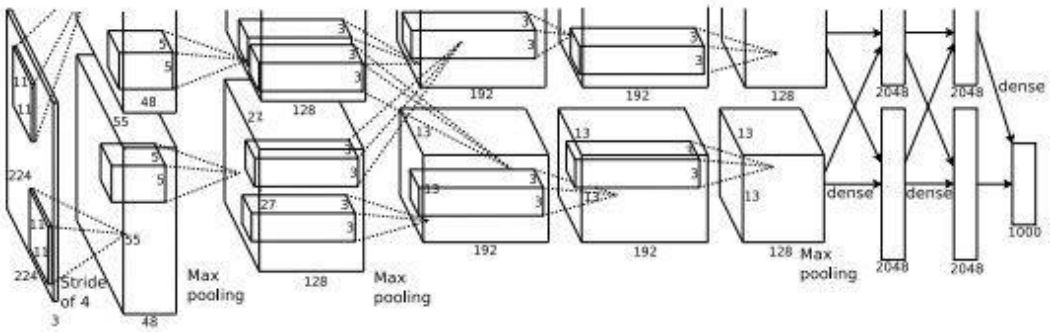
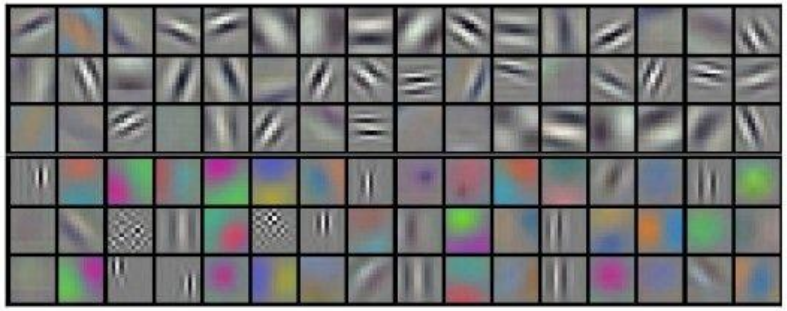


Illustration of Dahl et al. 2012 by Lane McIntosh, copyright CS231n 2017

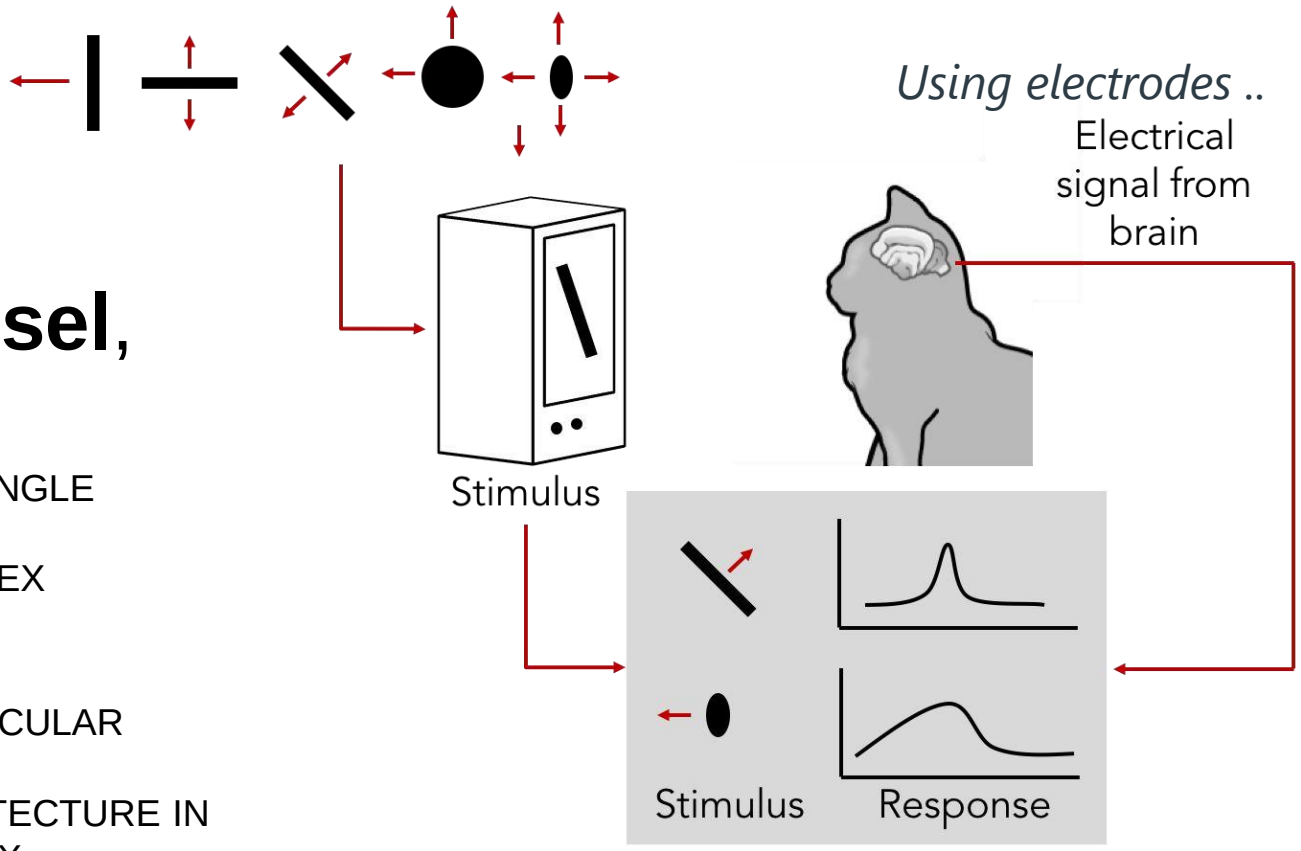
Imagenet classification with deep convolutional neural networks
Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, 2012





Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

A Bit of History ..



Hubel & Wiesel,

1959

RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

1962

RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968..

A Bit of History ..

Hubel & Wiesel,

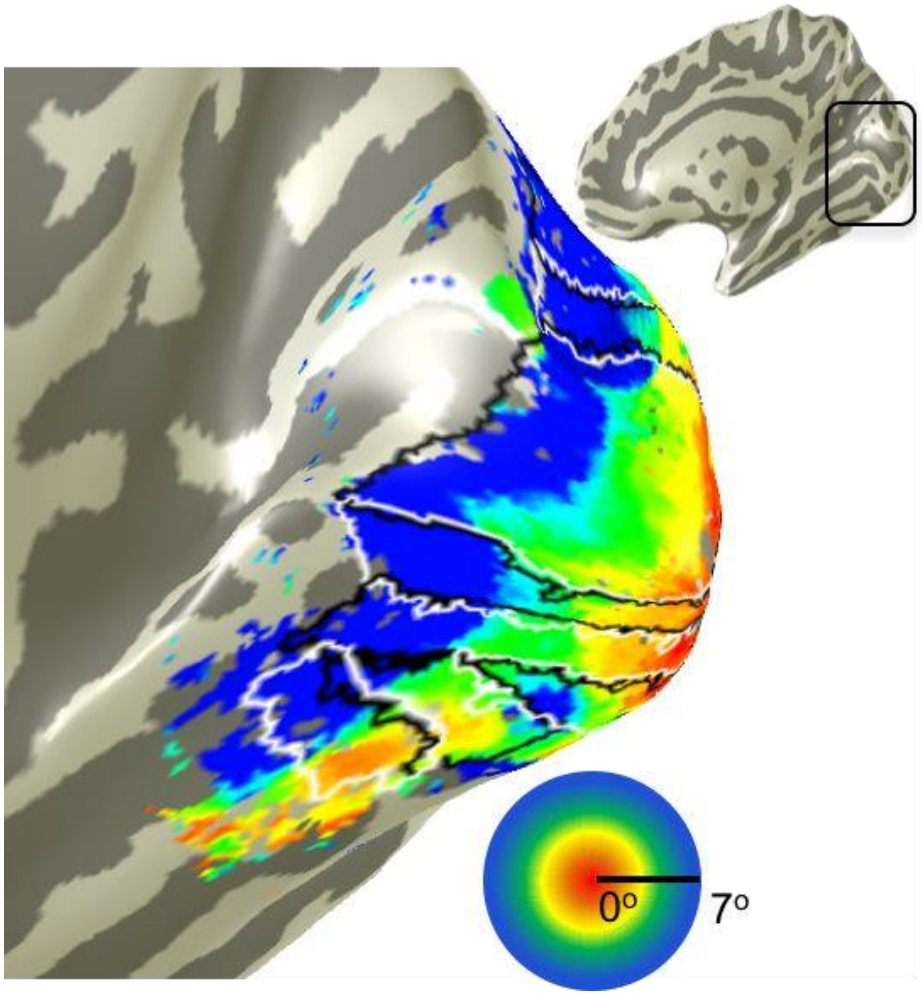
1959

RECEPTIVE FIELDS OF SINGLE NEURONES IN
THE CAT'S STRIATE CORTEX

1962

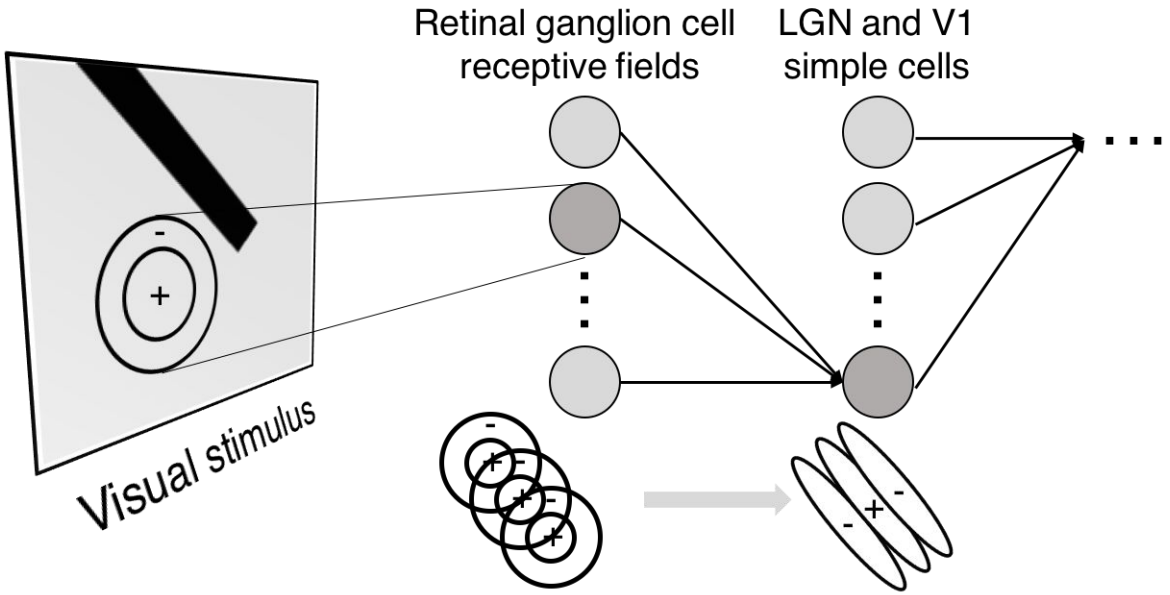
RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX

1968..



Topographical mapping in the cortex: nearby cells in cortex represent nearby regions in the visual field ..

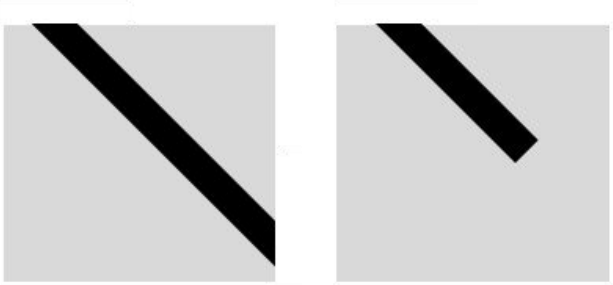
A Bit of History .. Hierarchical Organization?



Simple cells:
Response to light orientation

Complex cells:
Response to light orientation and movement

Hypercomplex cells:
response to movement with an end point

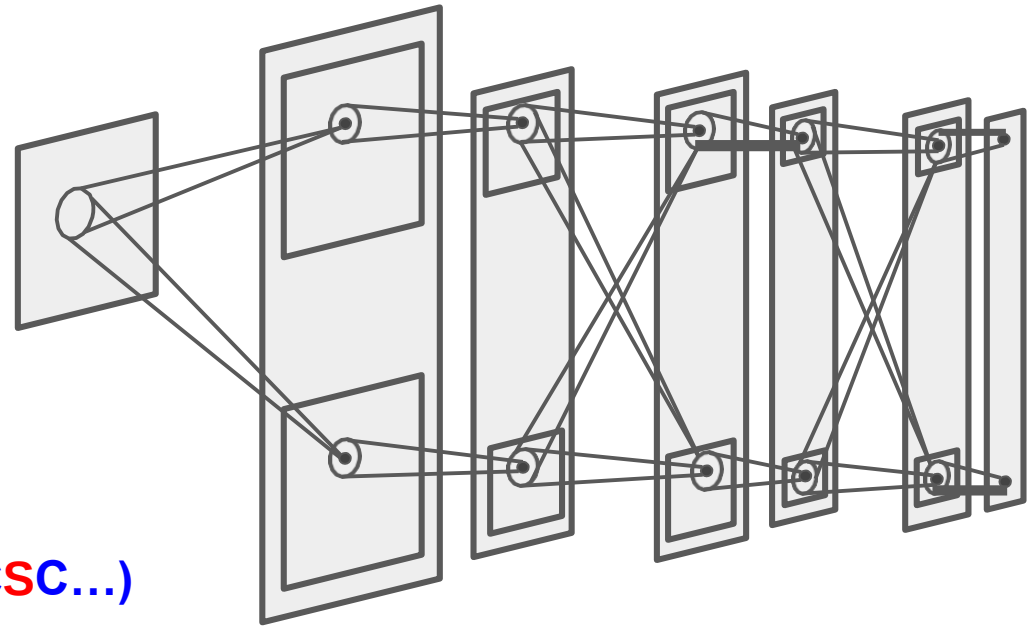


No response

Response (end point)

A Bit of History ..

Neocognitron
[Fukushima 1980]



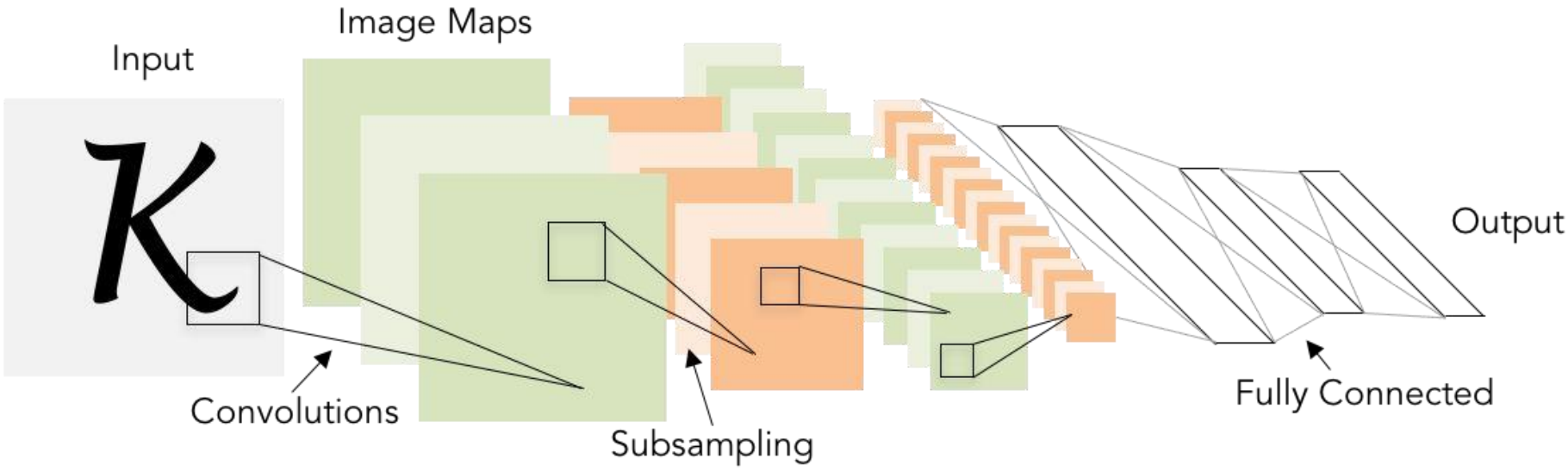
“sandwich” architecture (SCSCSC…)

simple cells: modifiable parameters

complex cells: perform pooling

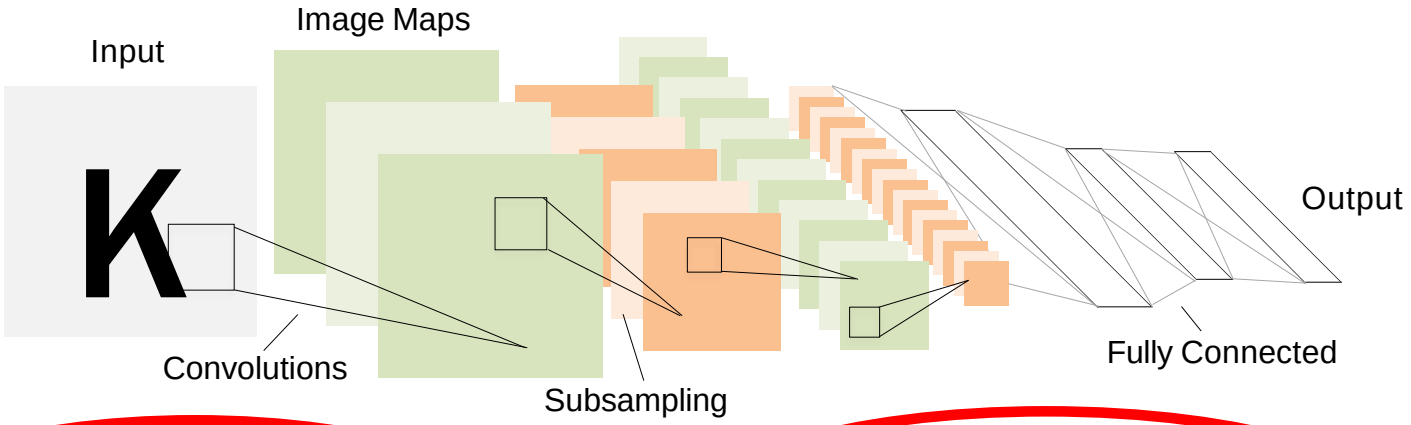
A Bit of History ..

Gradient-based learning applied to document recognition
[LeCun, Bottou, Bengio, Haffner 1998]



LeNet-5

1998
LeCun et al.



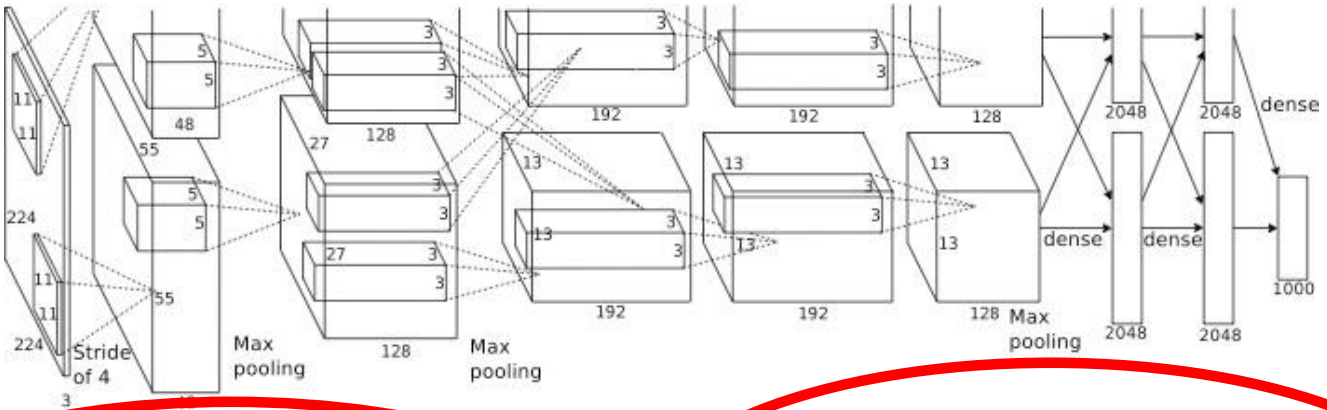
# of transistors

10^6

# of pixels used in training

10^7 NIST

2012
Krizhevsky et al.



# of transistors

10^9

GPUs

# of pixels used in training

10^{14} IMAGENET

IMAGENET

www.image-net.org

22K categories and 14M images

- Animals
  - Bird
  - Fish
  - Mammal
  - Invertebrate
- Plants
  - Tree
  - Flower
- Food
- Materials
- Structures
- Artifact
  - Tools
  - Appliances
  - Structures
- Person
- Scenes
  - Indoor
  - Geological Formations
- Sport Activities

Deng, Dong, Socher, Li, Li, & Fei-Fei, 2009

IMAGENET Large Scale Visual Recognition Challenge

The Image Classification Challenge:

1,000 object classes

1,431,167 images

Output:

Scale

T-shirt

<u>Steel drum</u>

Drumstick

Mud turtle

✔

Output:

Scale

T-shirt

Giant panda

Drumstick

Mud turtle

✗

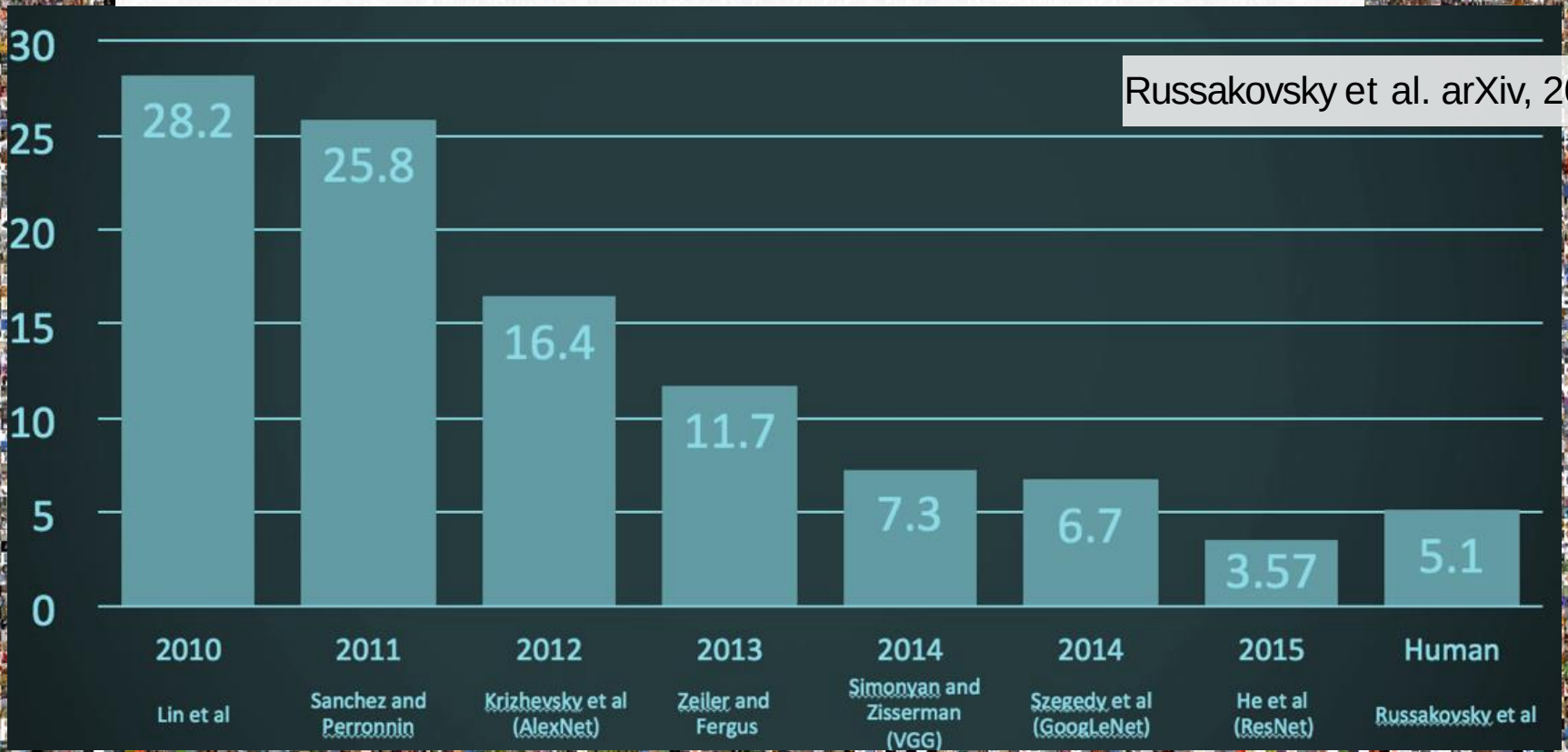
Russakovsky et al. arXiv, 2014

IMAGENET Large Scale Visual Recognition Challenge

The Image Classification Challenge:

1,000 object classes

1,431,167 images



Russakovsky et al. arXiv, 2014

IMAGENET Large Scale Visual Recognition Challenge

Year 2010

NEC-UIUC

Dense descriptor grid: HOG, LBP

Coding: local coordinate, super-vector

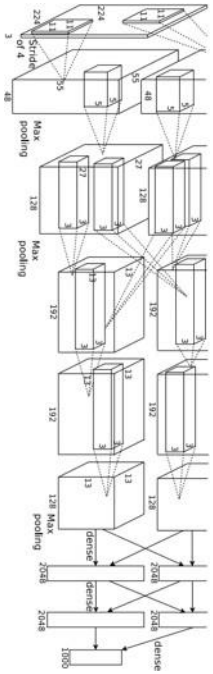
Pooling, SPM

Linear SVM

[Lin CVPR 2011]

Year 2012

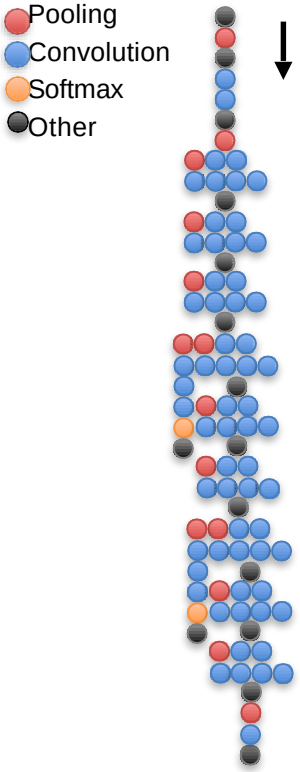
SuperVision



[Krizhevsky NIPS 2012]

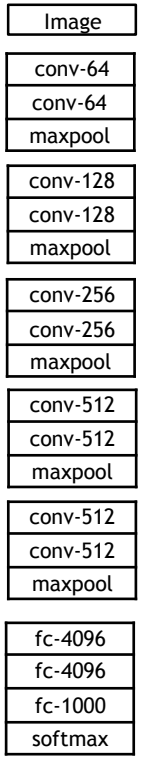
Year 2014

GoogLeNet



[Szegedy arxiv 2014]

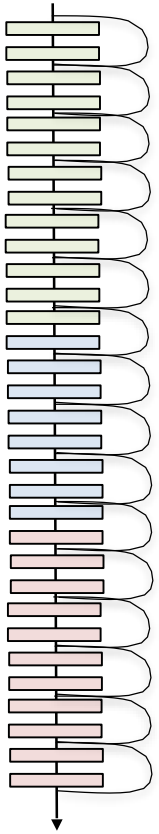
VGG



[Simonyan arxiv 2014]

Year 2015

MSRA



[He ICCV 2015]

A Bit of History ..

ImageNet Classification with Deep Convolutional Neural Networks *[Krizhevsky, Sutskever, Hinton, 2012]*

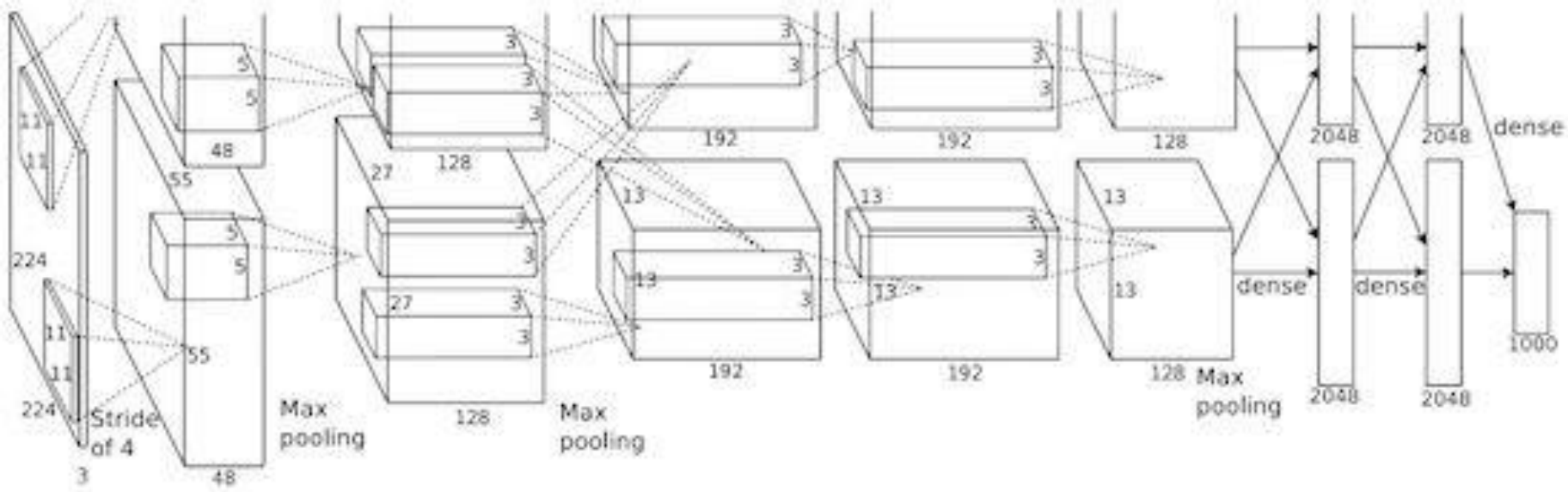
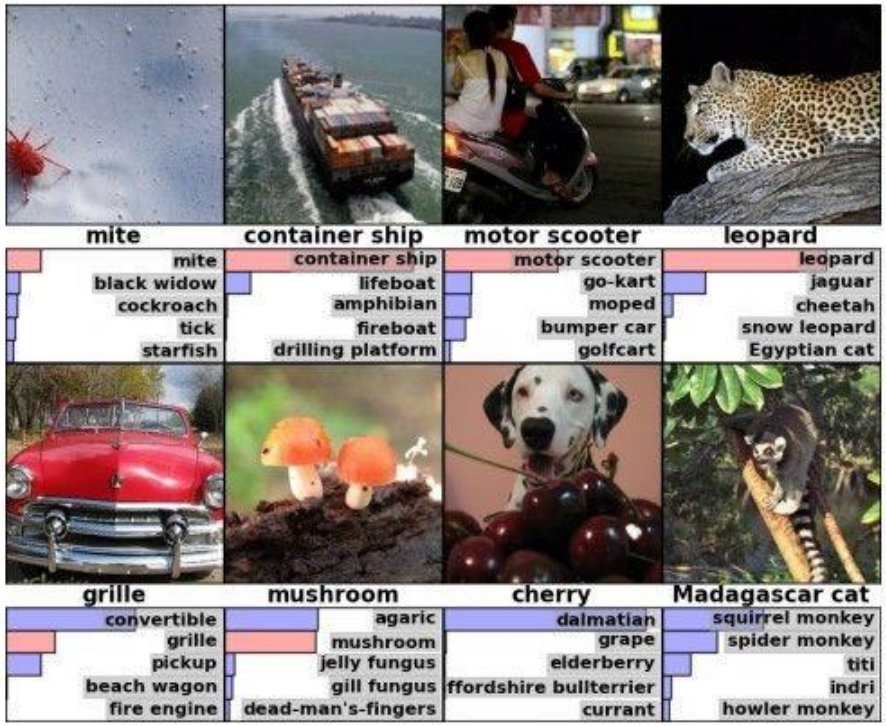


Figure copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

AlexNet

Fast-forward to today ..
ConvNets are everywhere ..

Classification



Retrieval

Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012. Reproduced with permission.

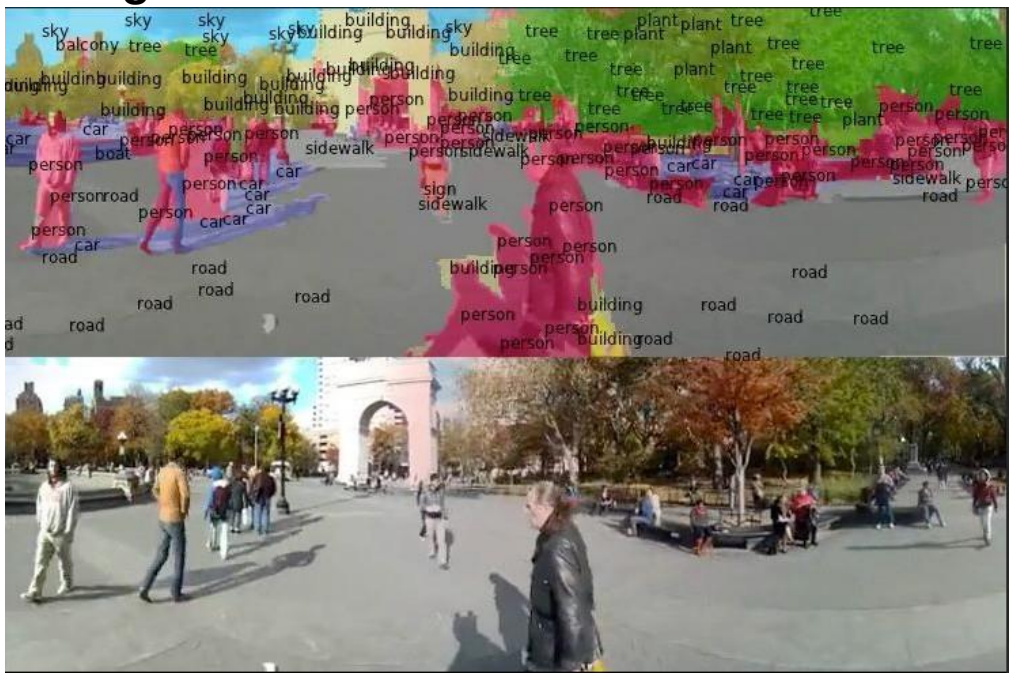
Fast-forward to today ..
ConvNets are everywhere ..

Detection

Figures copyright Shaoqing Ren, Kaiming He, Ross Girschick, Jian Sun, 2015. Reproduced with permission.

[Faster R-CNN: Ren, He, Girshick, Sun 2015]

Segmentation



Figures copyright Clement Farabet, 2012. Reproduced with permission.

[Farabet et al., 2012]

Fast-forward to today ..

ConvNets are everywhere ..

Mnih and Hinton, 2010

Photo and figure by Lane McIntosh; not actual example from Mnih and Hinton, 2010 paper.

This image by Christin Khan is in the public domain and originally came from the U.S. NOAA.

[Toshev, Szegedy 2014] Images are examples of pose estimation, not actually from Toshev & Szegedy 2014. Copyright Lane McIntosh.

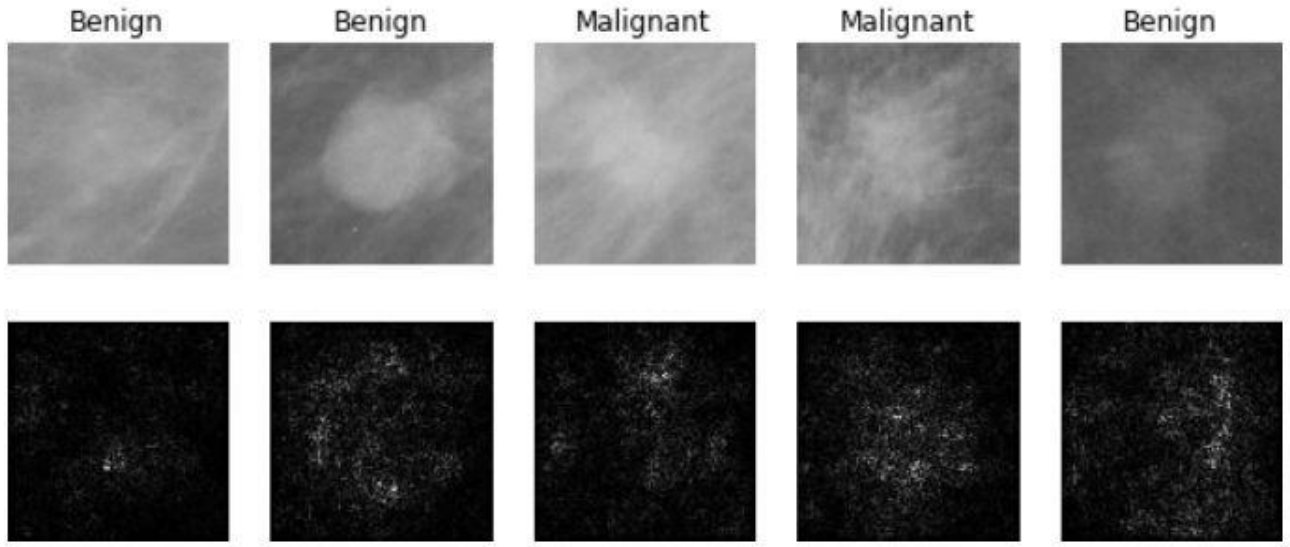
Fast-forward to today ..
ConvNets are everywhere ..

[Sermanet et al. 2011] [Ciresan et al.]

Photos by Lane McIntosh. Copyright CS231n 2017.

[Dieleman et al. 2014]

From left to right: public domain by NASA, usage permitted by ESA/Hubble, public domain by NASA, and public domain.



[Levy et al. 2016]

Figure copyright Levy et al. 2016. Reproduced with permission.

Fast-forward to today ..

ConvNets are everywhere ..

All images are CC0 Public domain: https://pixabay.com/en/luggage-antique-cat-1643010/ https://pixabay.com/en/teddy-plush-bears-cute-teddy-bear-1623436/ https://pixabay.com/en/surf-wave-summer-sport-litoral-1668716/ https://pixabay.com/en/woman-female-model-portrait-adult-983967/ https://pixabay.com/en/handstand-lake-meditation-496008/ https://pixabay.com/en/baseball-player-shortstop-infield-1045263/

Captions generated by Justin Johnson using Neuraltalk2

Image Captioning

[Vinyals et al., 2015]
[Karpathy and Fei-Fei, 2015]

No errors

A white teddy bear sitting in the grass

A man riding a wave on top of a surfboard

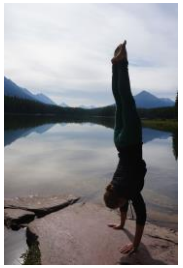
Minor errors

A man in a baseball uniform throwing a ball

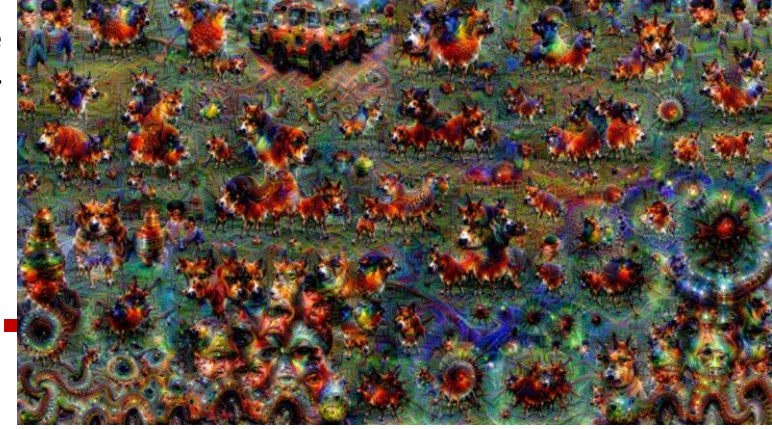
A cat sitting on a suitcase on the floor

Somewhat related

A woman is holding a cat in her hand



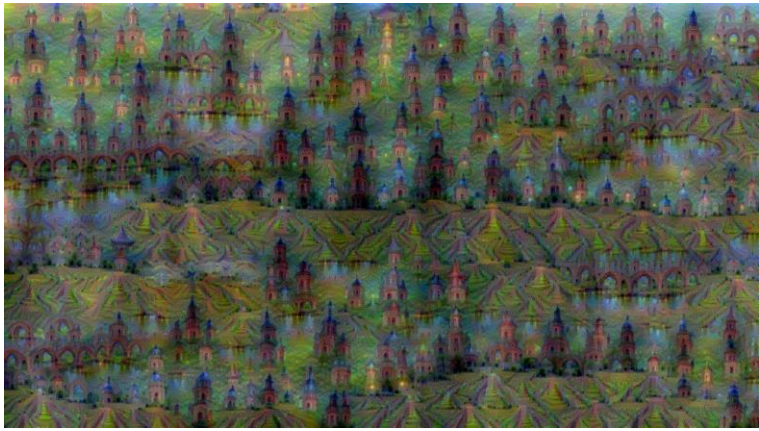
A woman standing on a beach holding a surfboard

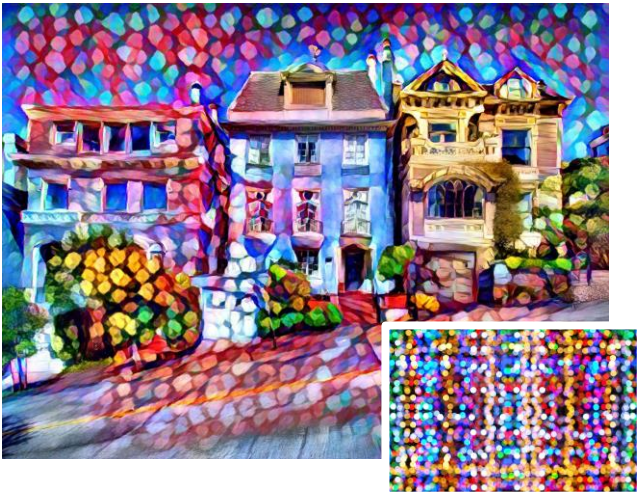


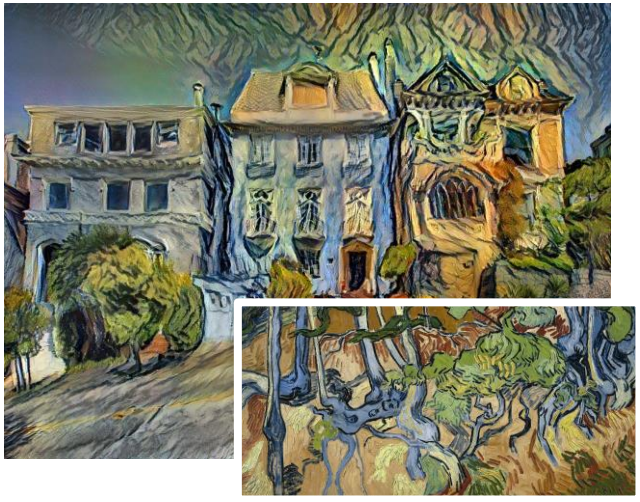
Fast-forward to today ..

ConvNets are everywhere ..

Gatys et al, "Image Style Transfer using Convolutional Neural Networks", CVPR 2016
Gatys et al, "Controlling Perceptual Factors in Neural Style Transfer", CVPR 2017







Overall CNN Architecture

Convolution Layer

– *Convolution Filters*

– *ConvNet: a Sequence of Convolution Layers*

– *Spatial Dimensions*

Pooling Layer

– *Padding the Borders*

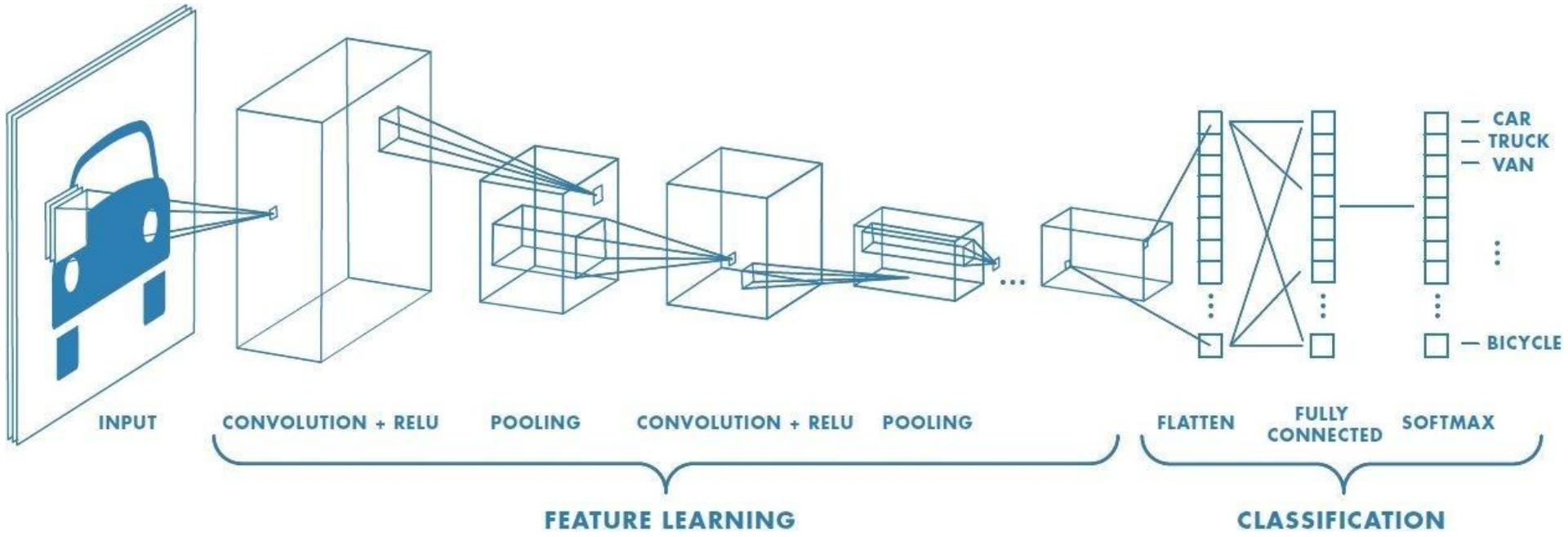
Fully Connected Layer (FC layer)

– *Flattening*

Basic Concepts of CNNs

Convolutional Neural Networks

Overall “Sample” CNN Architecture ..



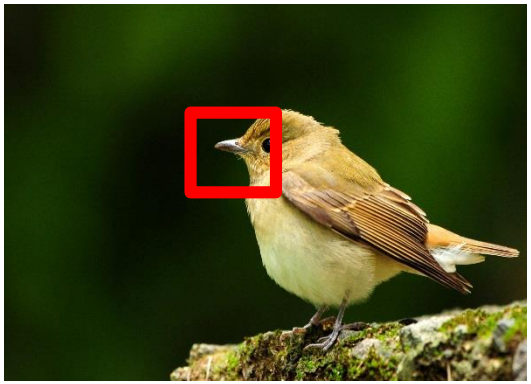
Consider Learning an Image ..

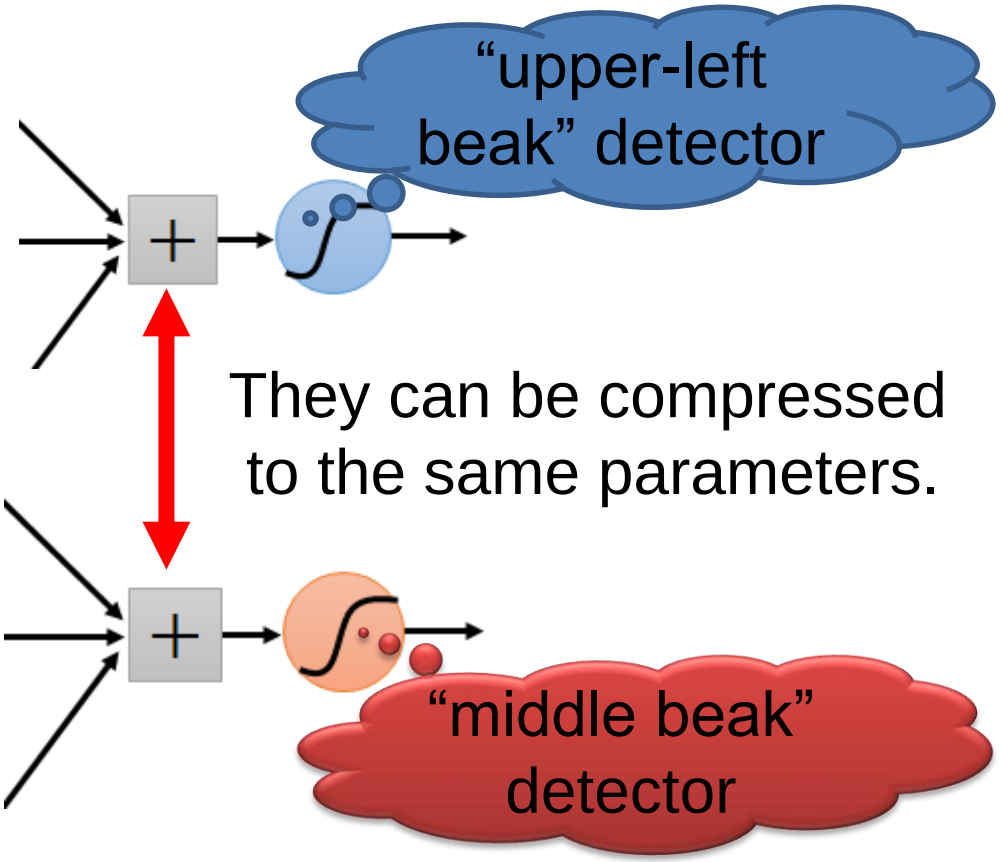
Some patterns are much smaller than the whole image ..

Can represent a small region with fewer parameters?

Same pattern appears in different places: They can be compressed!

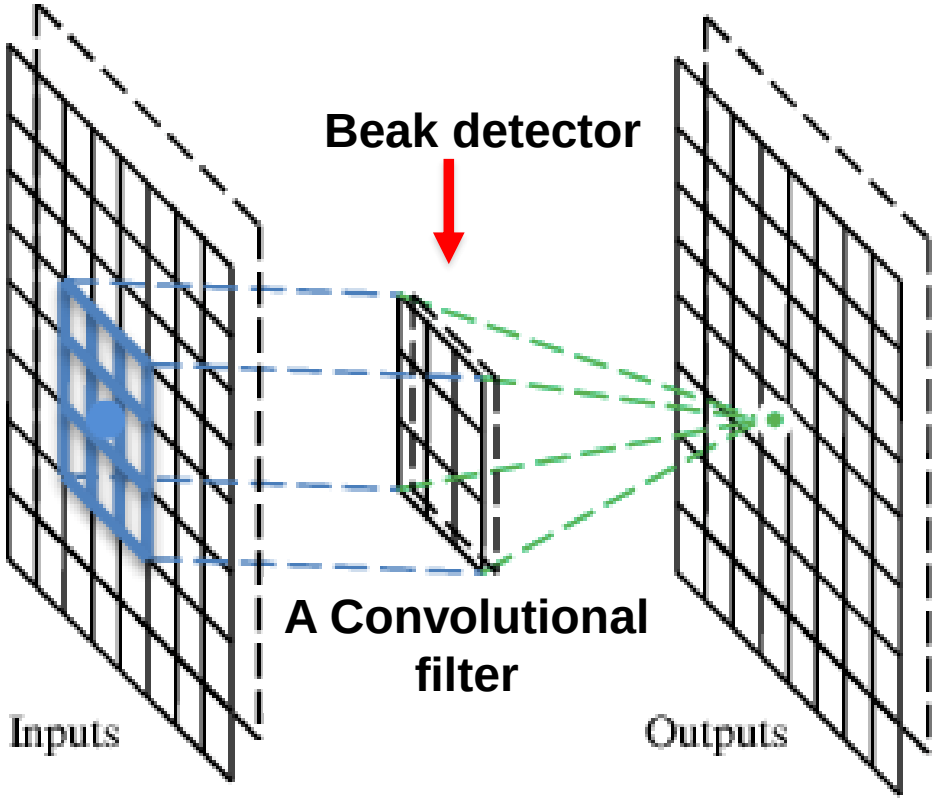
What about training a lot of such "small" detectors and each detector must "move around"?





Convolution Layer .. ?

- A CNN is a neural network with some convolutional layers (and some other layers).
- A convolutional layer has a number of filters that does convolutional operation.



Convolution Filters .. ?

An Example:

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

These are the network parameters to be learned.

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

⋮ ⋮

Each filter detects a small pattern (3 x 3).

Convolution Filters .. ?

An Example:

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

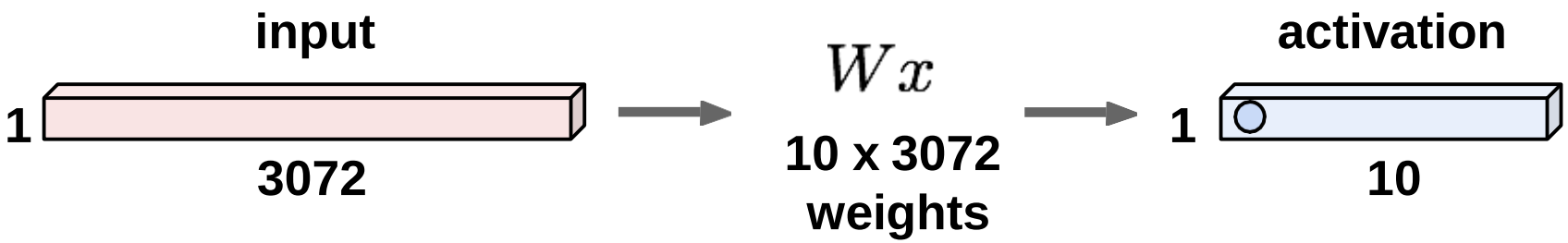
6 x 6 image

Dot product

3 -1

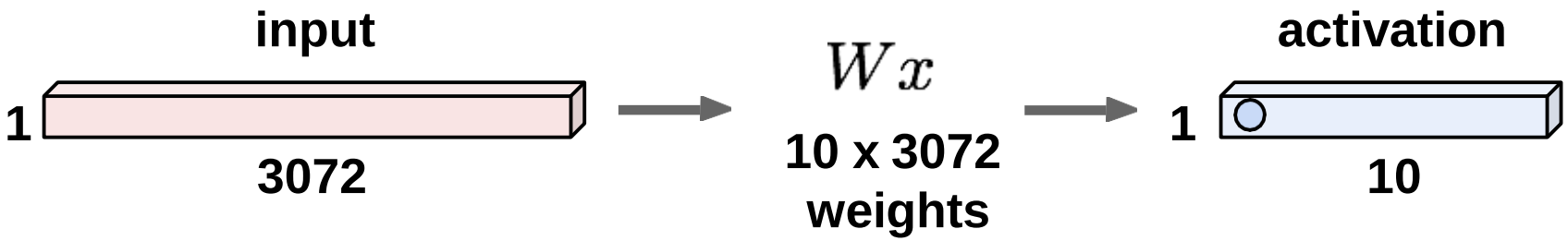
Fully Connected Layer .. ?

32 x 32 x 3 image ► stretch to 3072 x 1



Fully Connected Layer

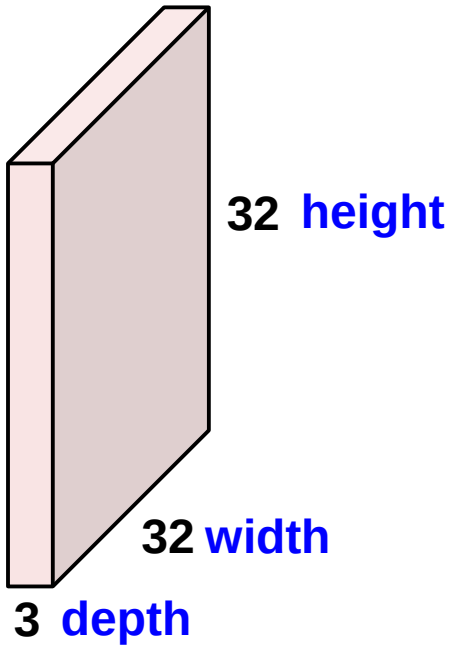
32 x 32 x 3 image ► stretch to 3072 x 1



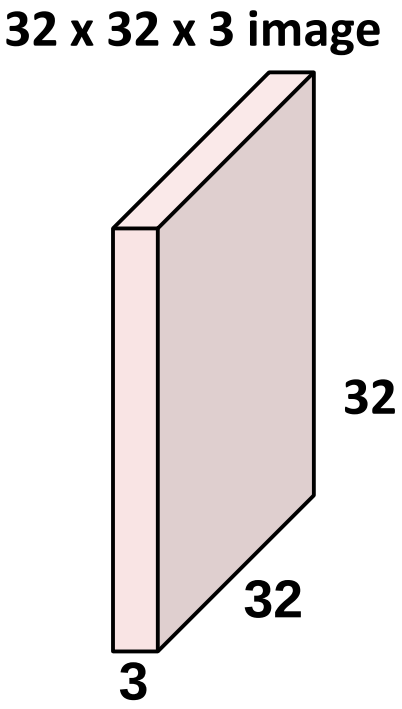
1 number:
the result of taking a dot product between a row of W and the input (a 3072-dimensional dot product)

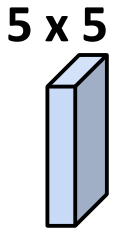
Convolution Layer

32 x 32 x 3 image ► preserve spatial structure



Convolution Layer

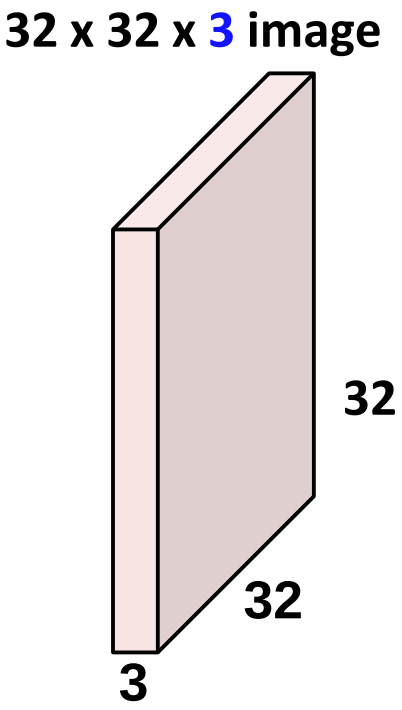


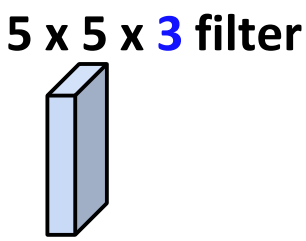


Convolve the filter with the image i.e. "slide over the image spatially, computing dot products"

Convolution Layer

Filters always extend the full depth of the input volume

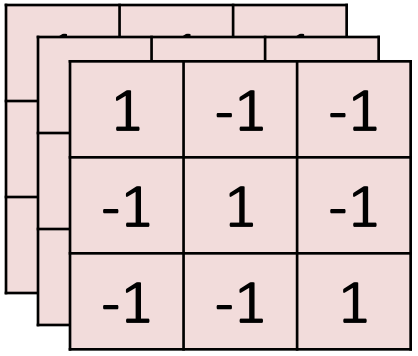




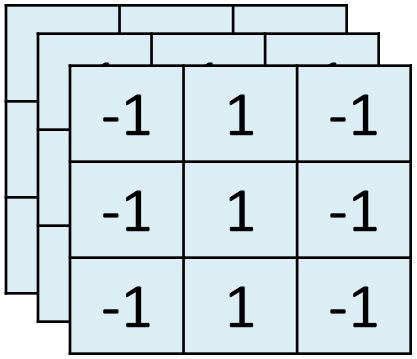
Convolve the filter with the image
i.e. "slide over the image spatially, computing dot products"

Color Image: RGB 3 channels

Convolution Layer

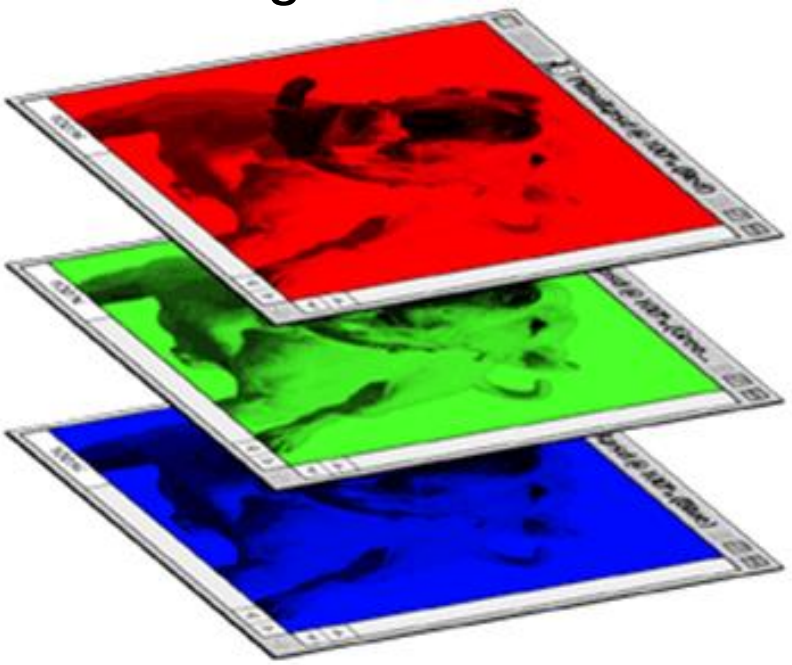


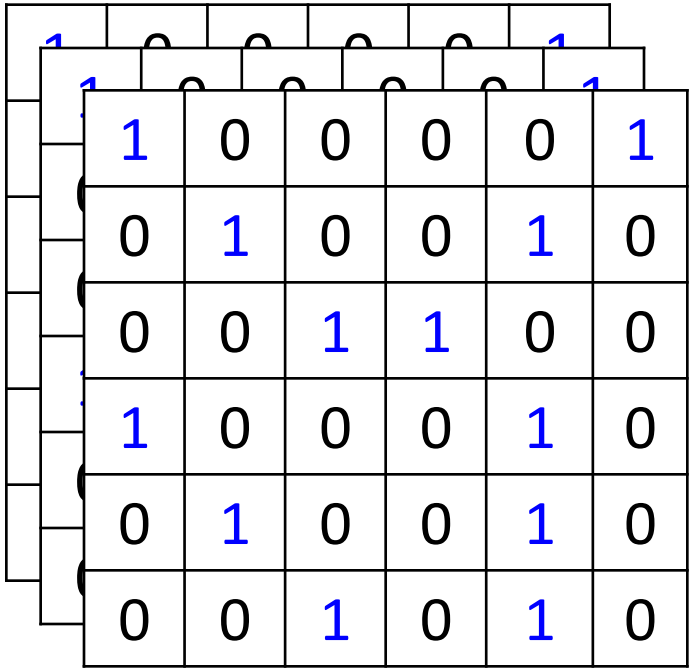
Filter 1



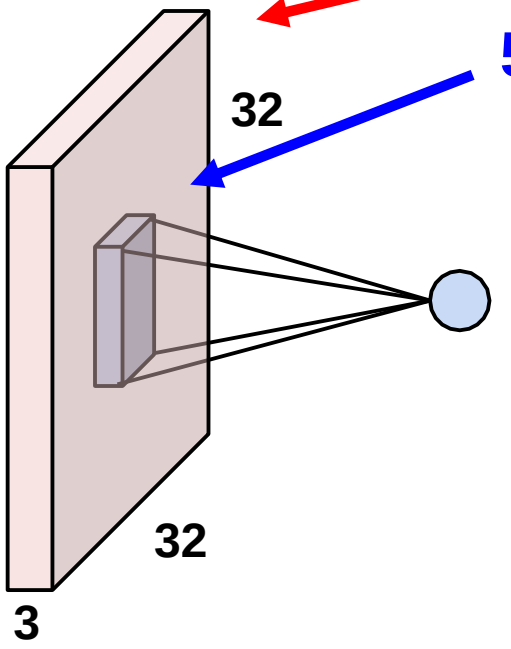
Filter 2

Color image





Convolution Layer



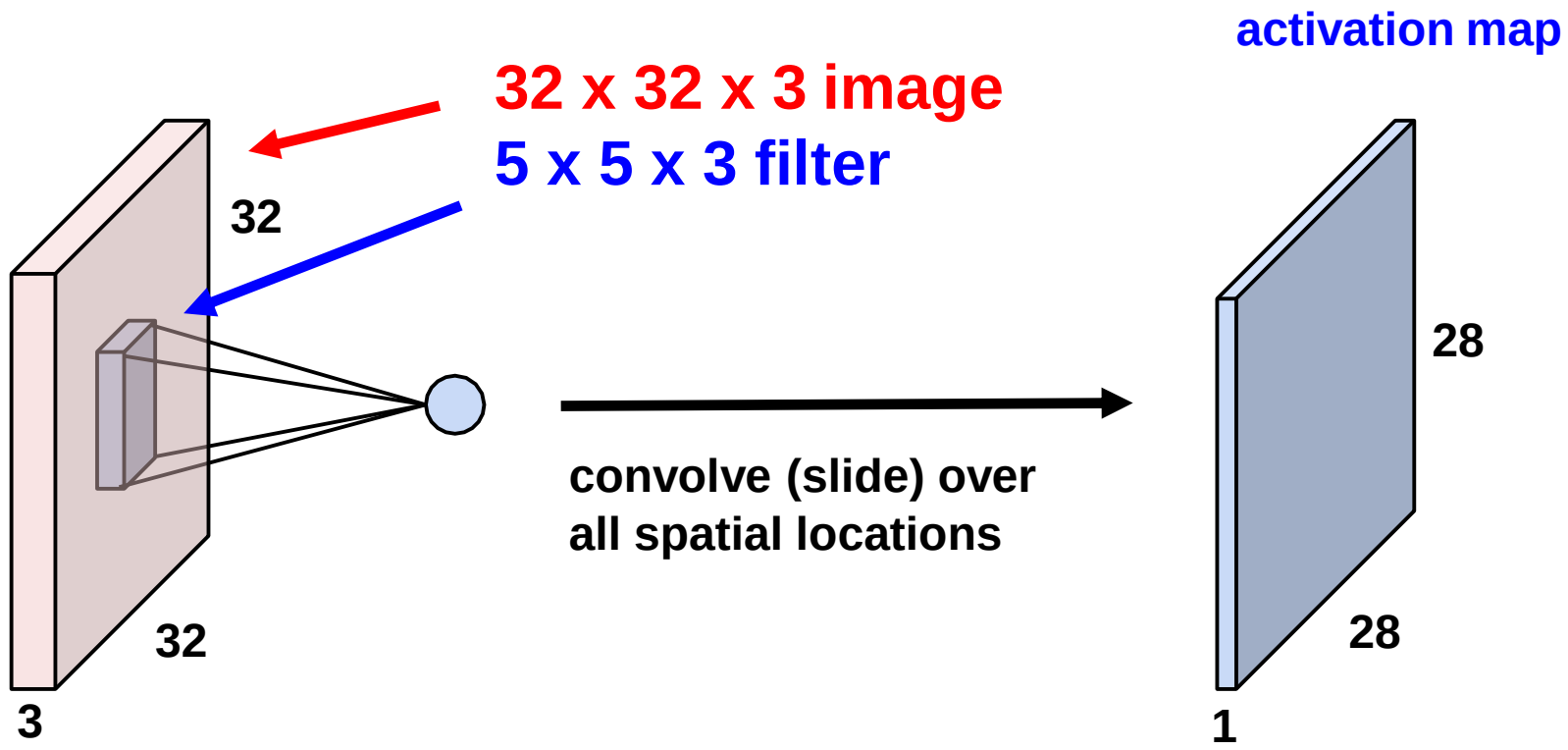
32 x 32 x 3 image

5 x 5 x 3 filter w

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)

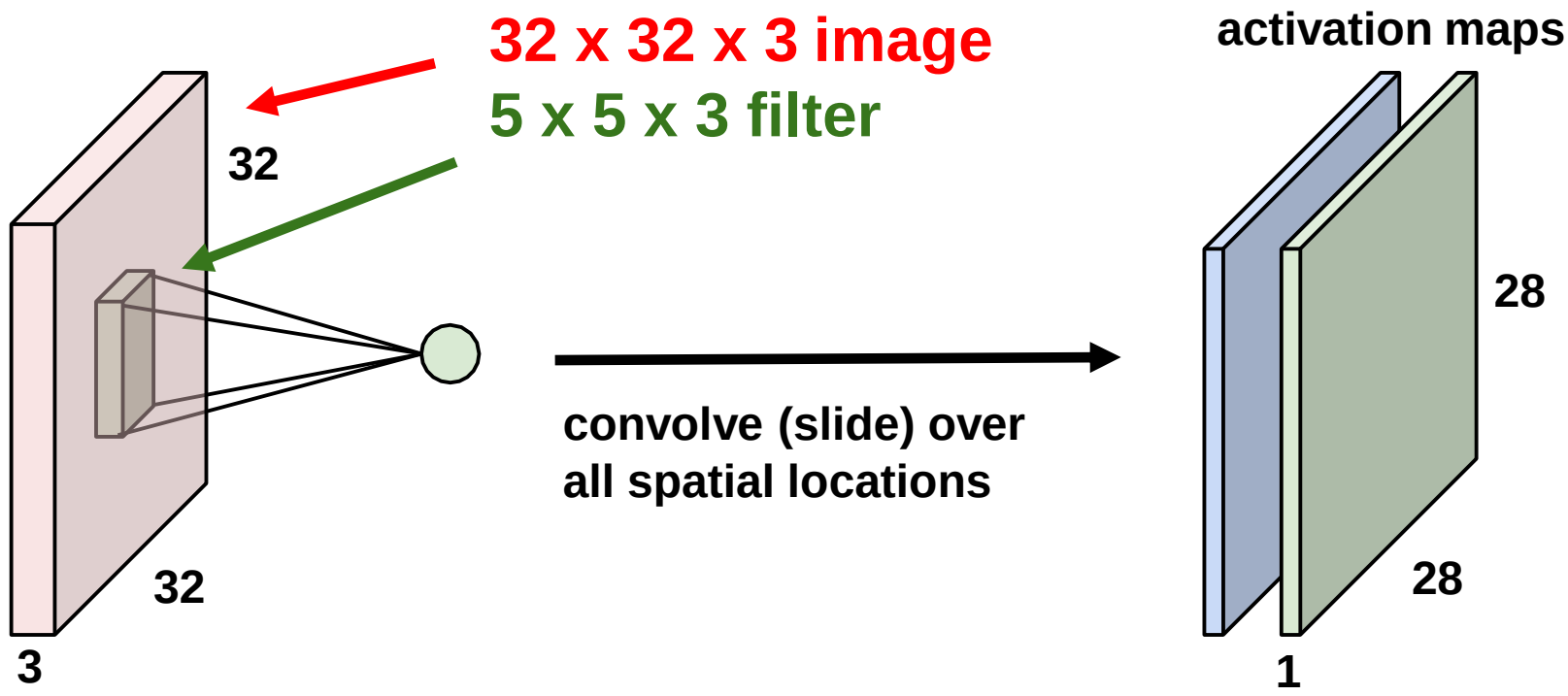
$$w^T x + b$$

Convolution Layer



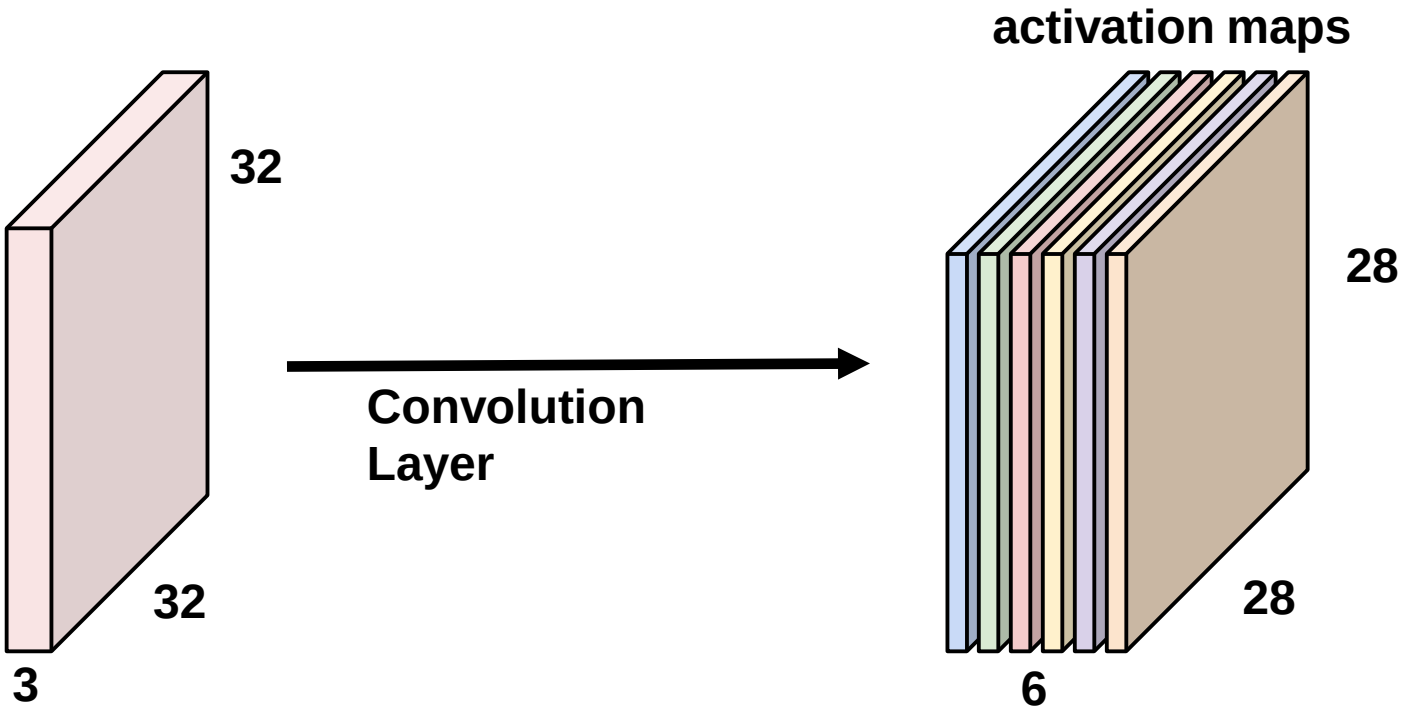
Convolution Layer

consider a second, green filter



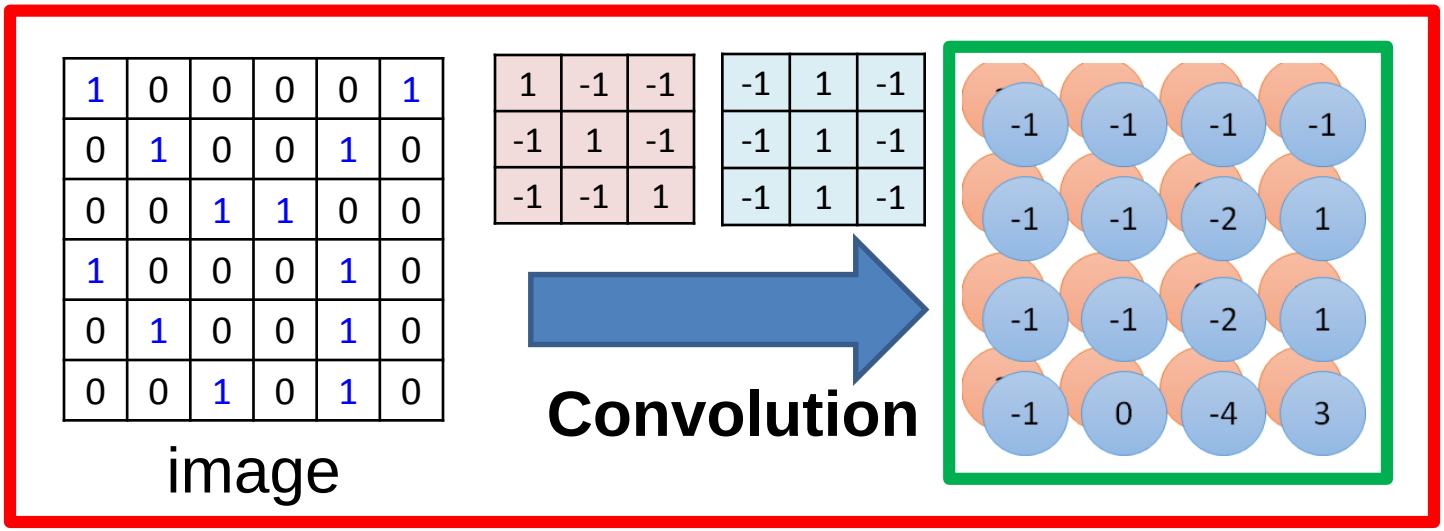
Convolution Layer

For example, if we had 6 5x5 filters, we'll get 6 separate activation maps:

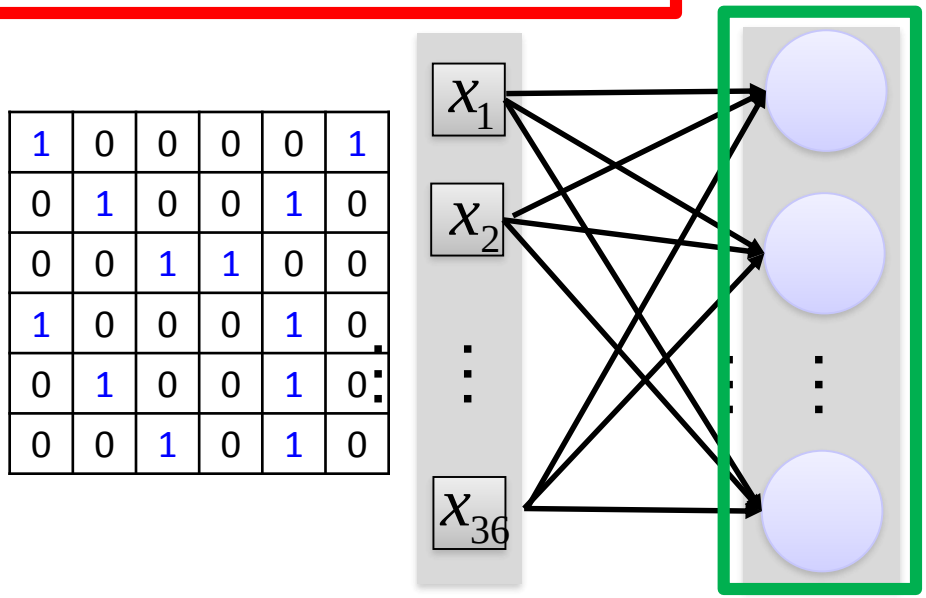


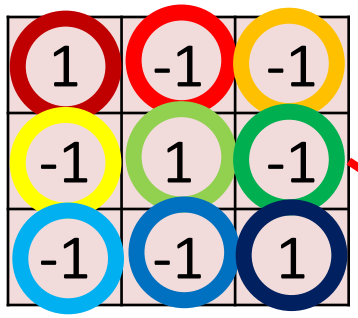
We stack these up to get a "new image" of size 28 x 28 x 6!

Convolution versus Fully-Connected



Fully-Connected ▸





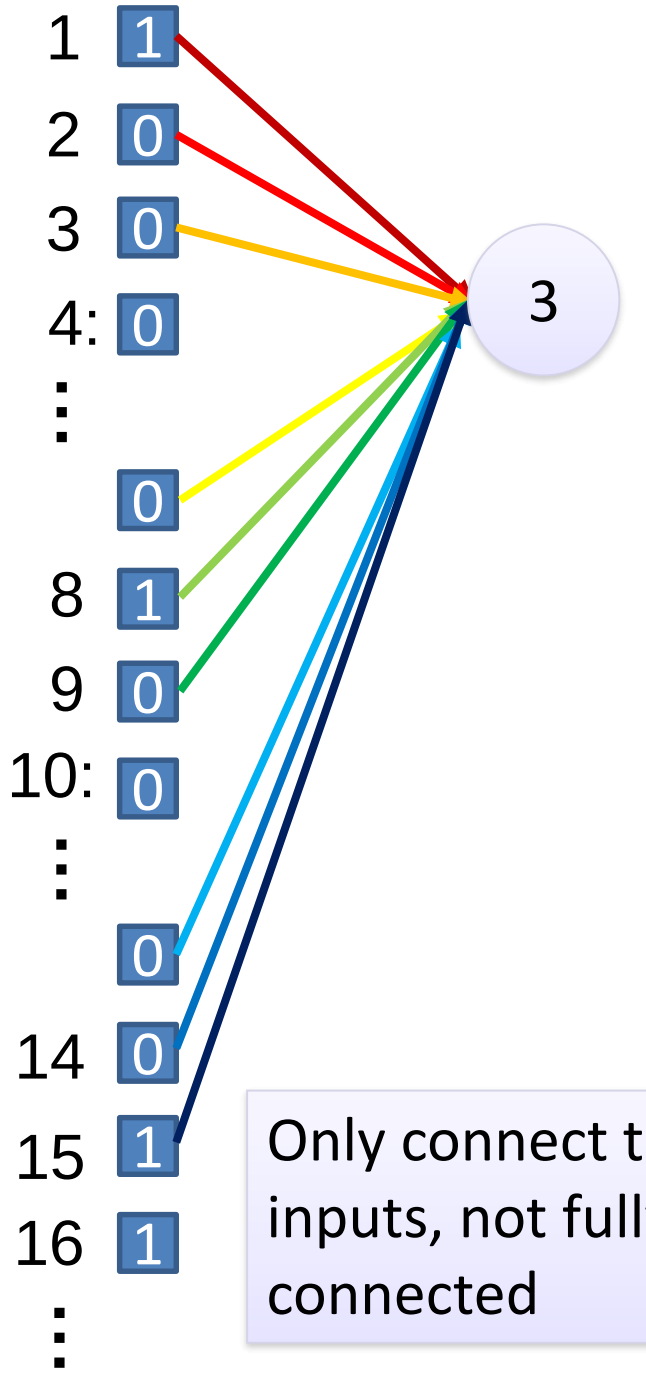
Filter 1

1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

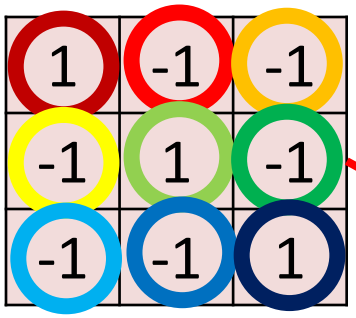
6 x 6 image

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

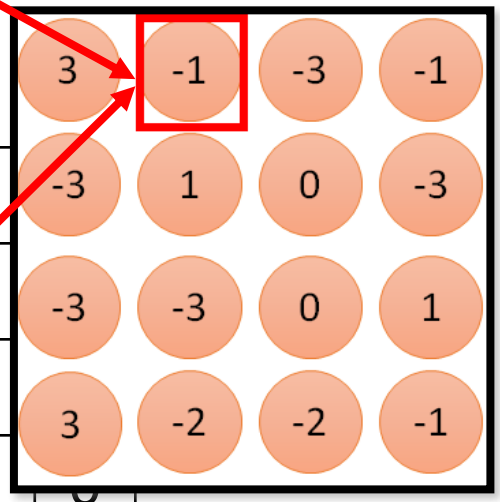
Fewer parameters



Only connect to 9 inputs, not fully connected



Filter 1

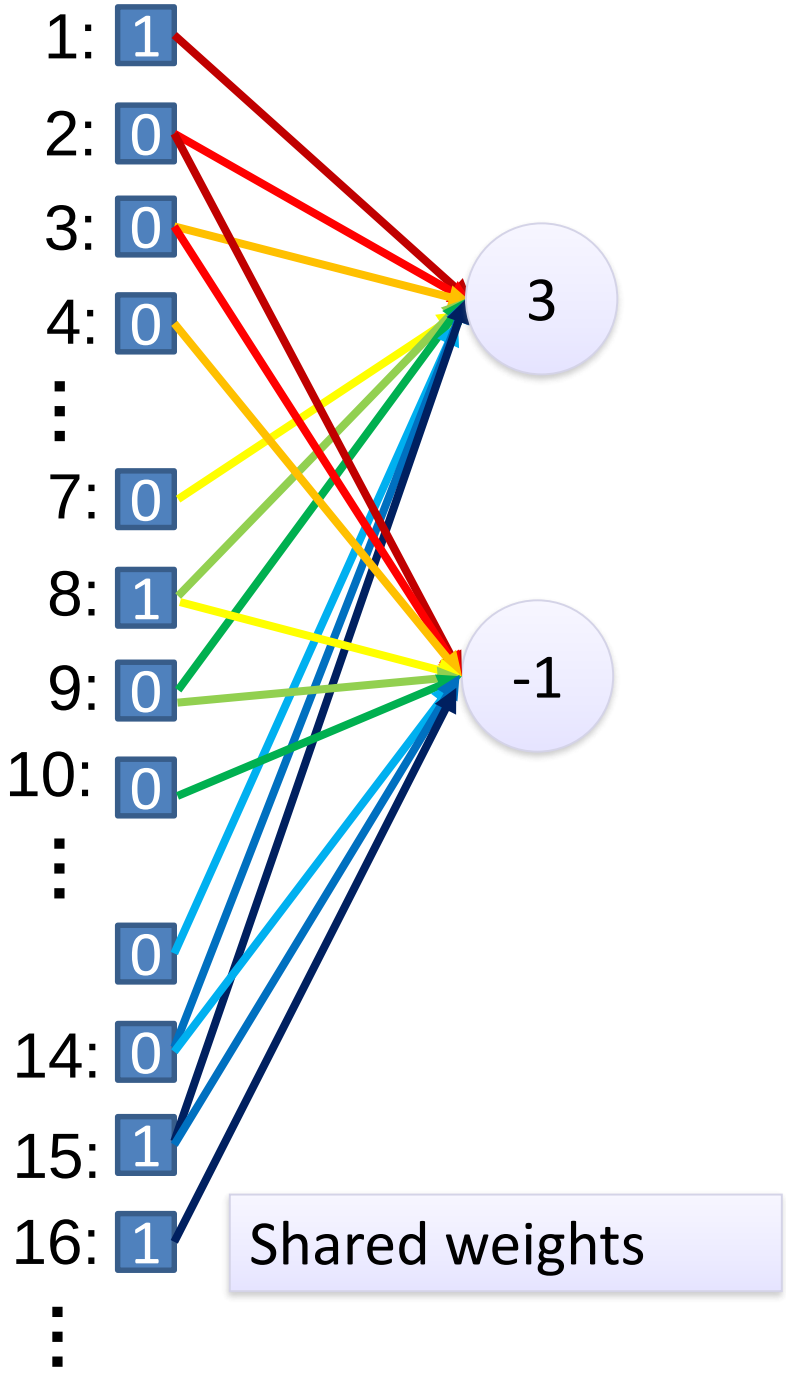


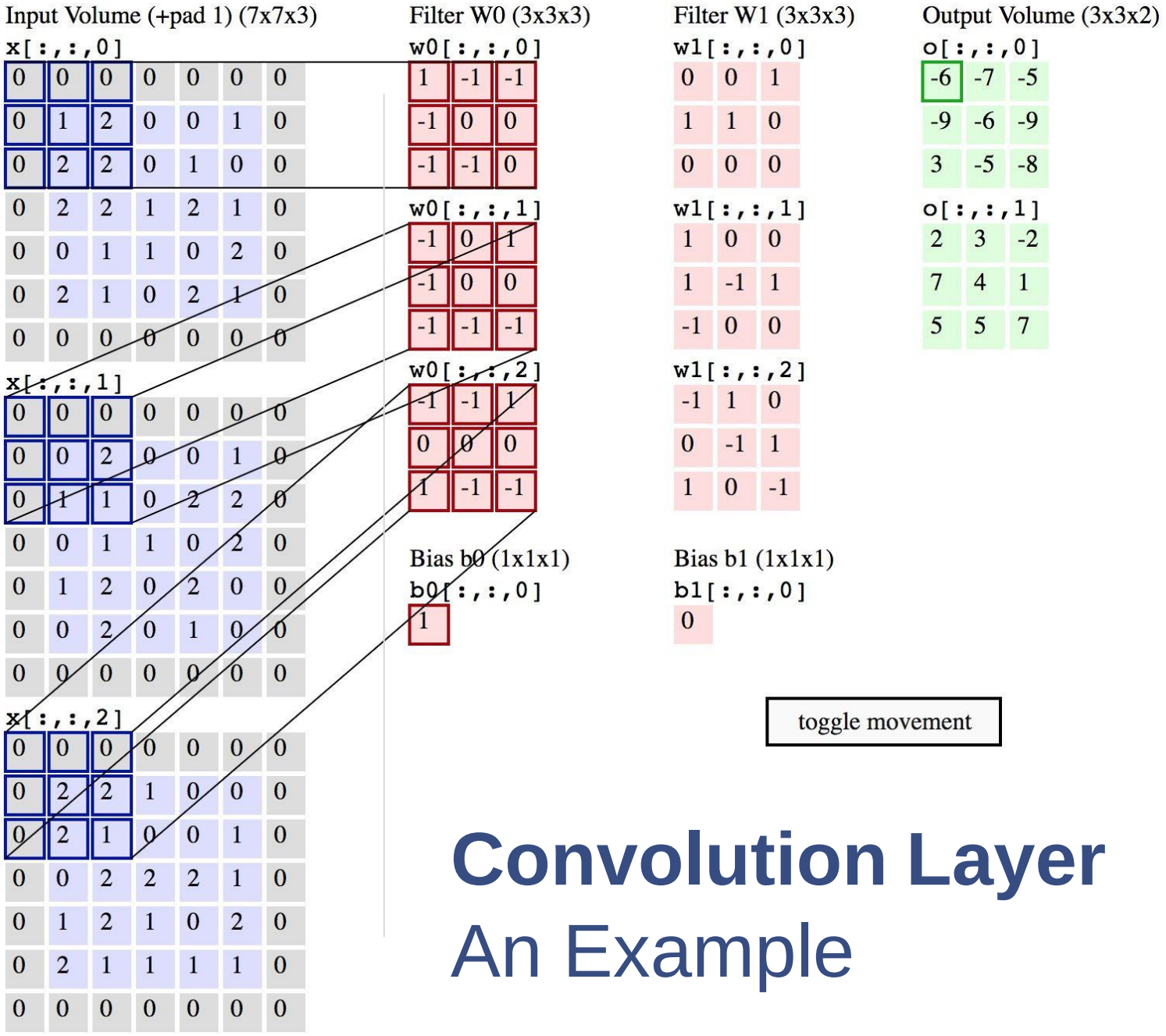
1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Fewer parameters

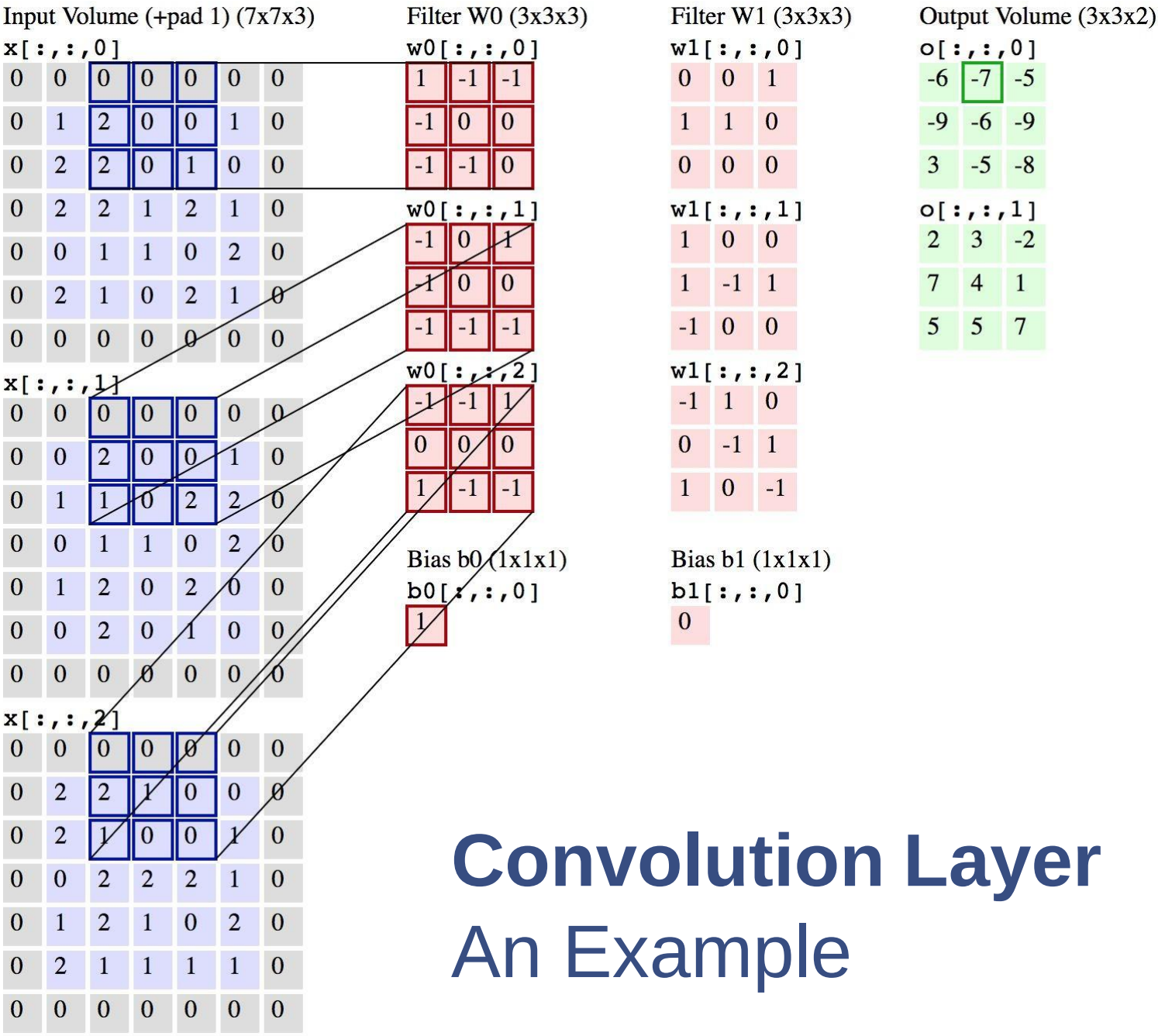
Even fewer parameters





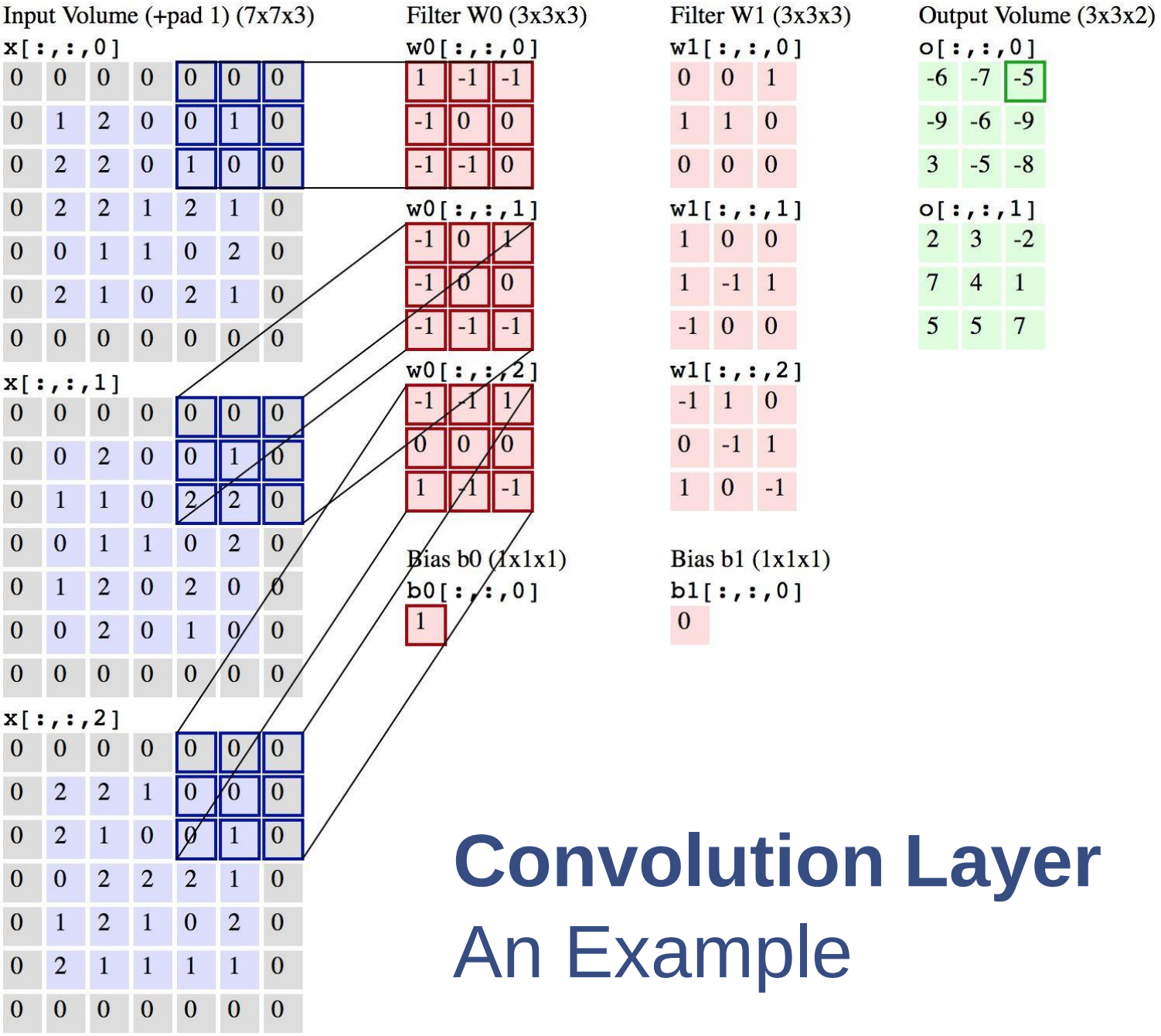
Convolution Layer

An Example



Convolution Layer

An Example

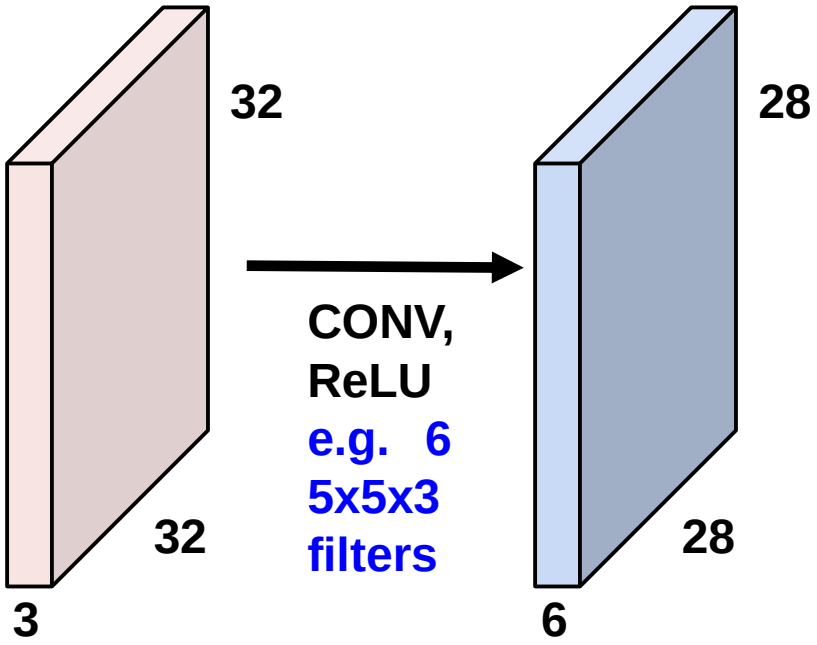


Convolution Layer

An Example

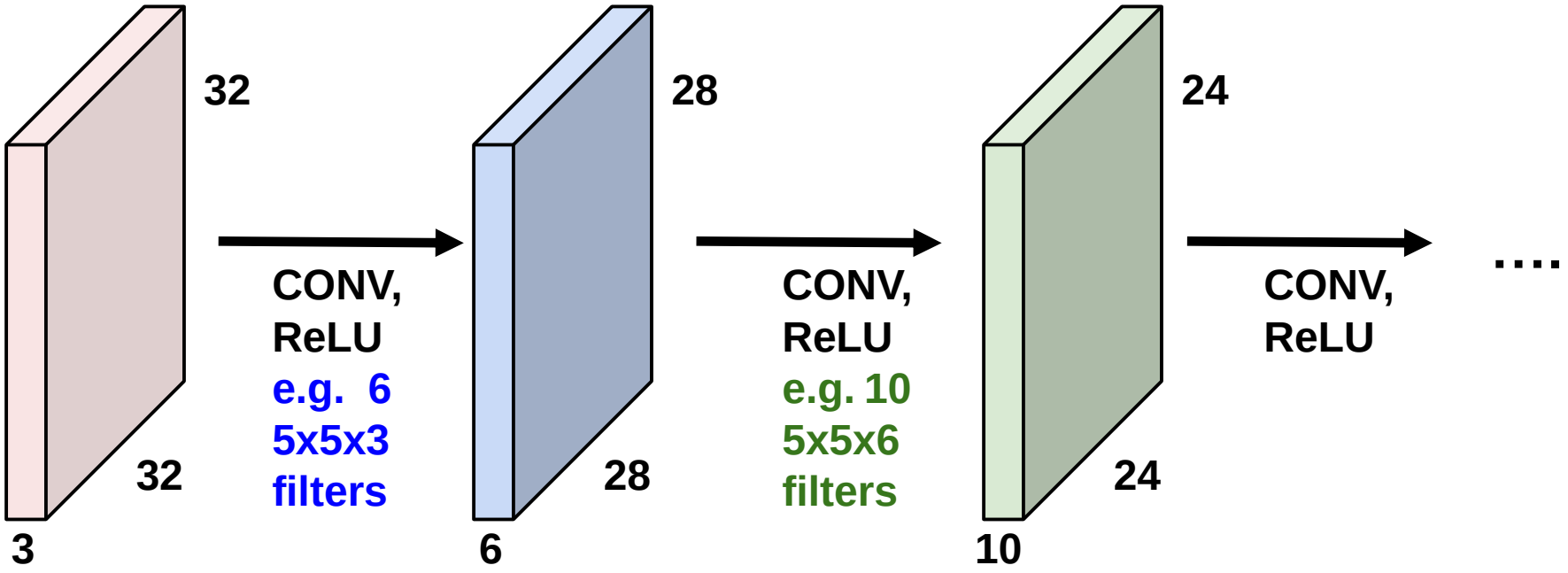
Convolution Layer

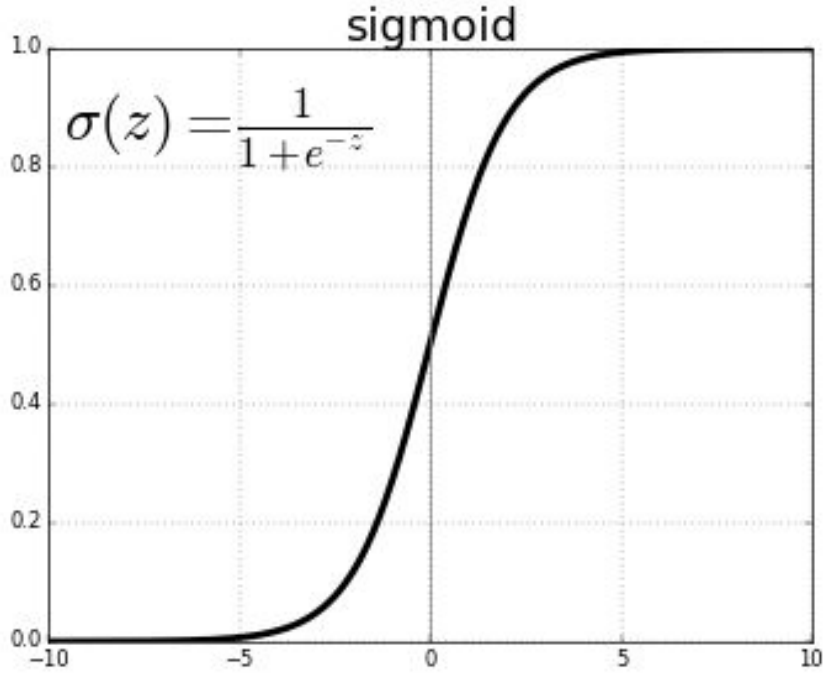
Preview: ConvNet is a sequence of Convolution Layers, interspersed with activation functions.

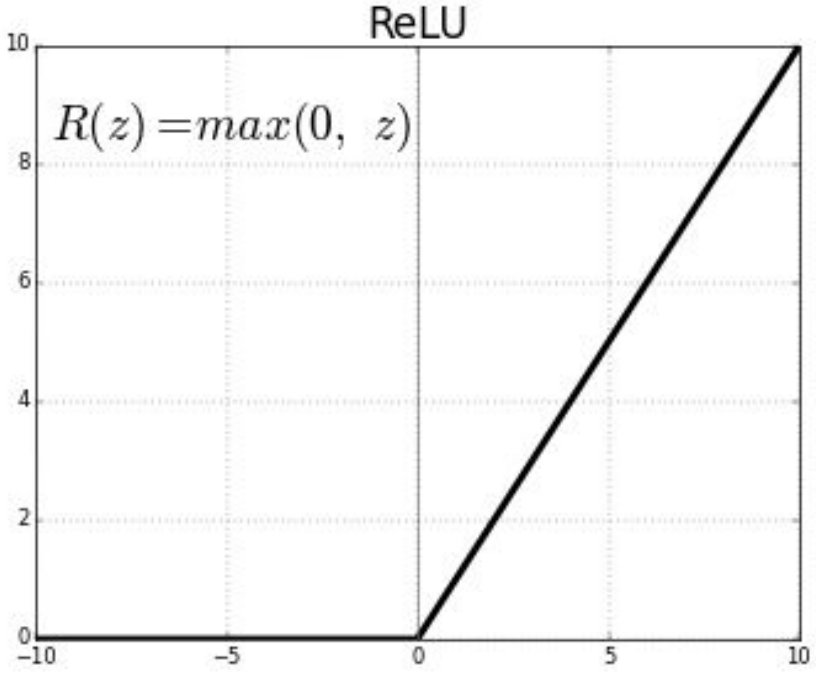


Convolution Layer

Preview: ConvNet is a sequence of Convolutional Layers, interspersed with activation functions







Activation Layer

- Used to increase non-linearity of the network without affecting receptive fields of conv layers.
- Prefer ReLU, results in faster training.

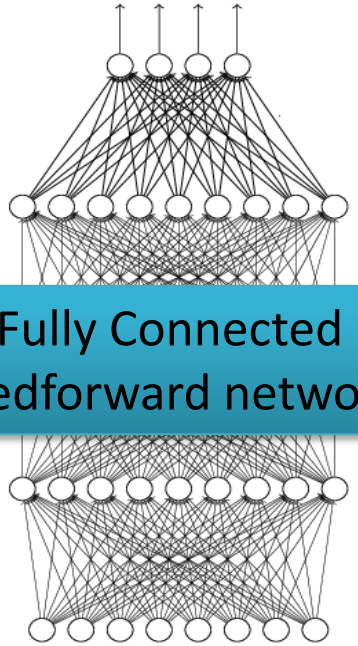
Other types:

Leaky ReLU, Randomized Leaky ReLU, Parameterized ReLU, Exponential Linear Units (ELU), Scaled Exponential Linear Units, Tanh, Hard-tanh, Soft-tanh, Soft-sign, Soft-max, Soft-plus...

The Whole CNN

Preview

Cat? Dog? ..



Fully Connected Feedforward network

Convolution, ReLU

Pooling

Convolution, ReLU

Pooling

Can repeat many times. For example ..

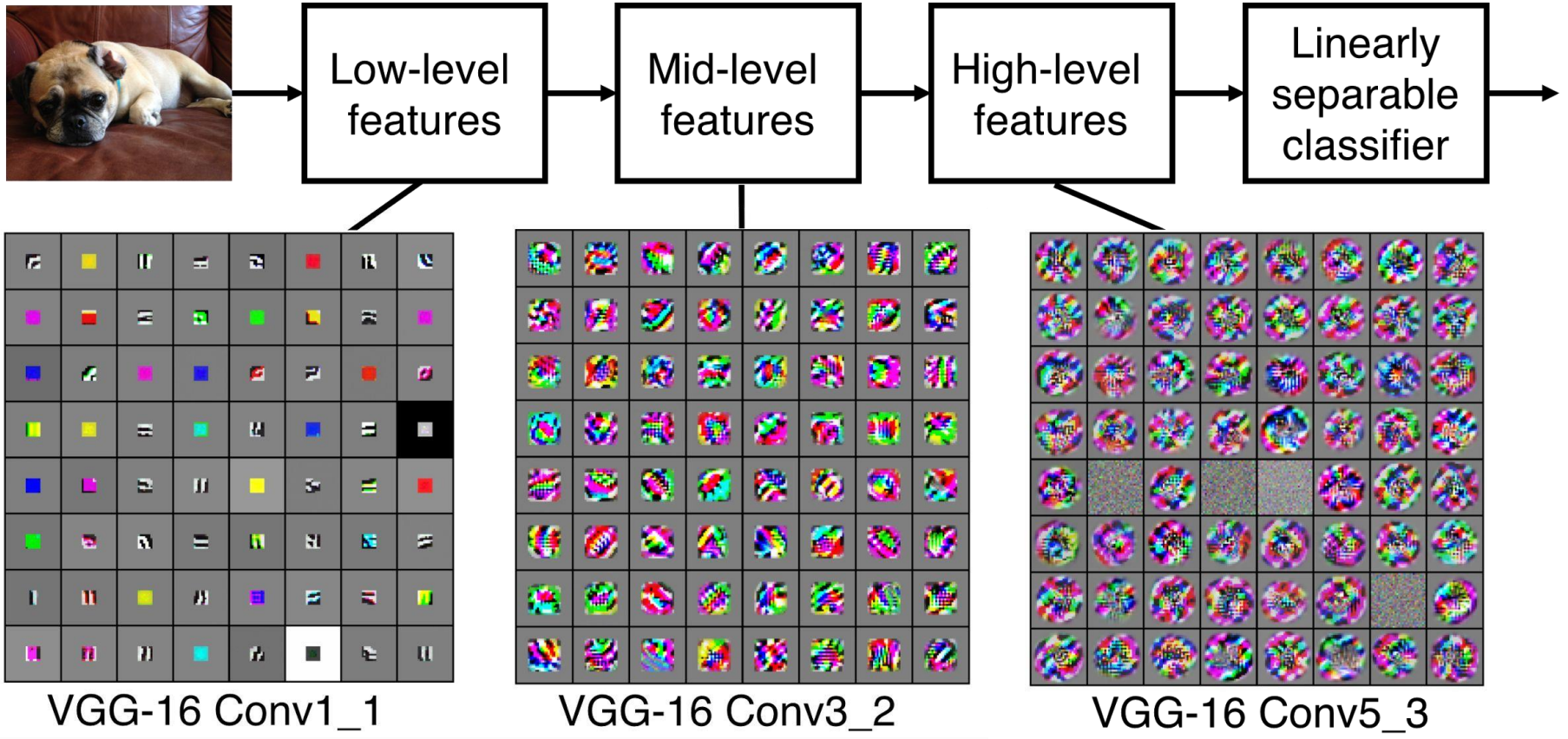
Flattened

Convolution Layer

Preview

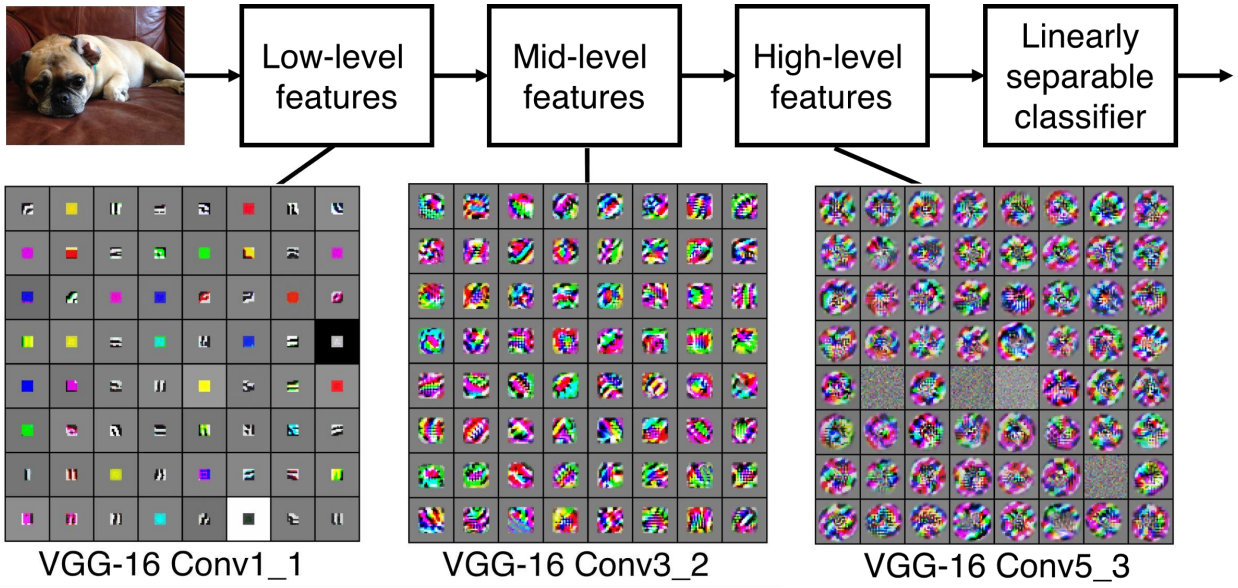
[Zeiler and Fergus 2013]

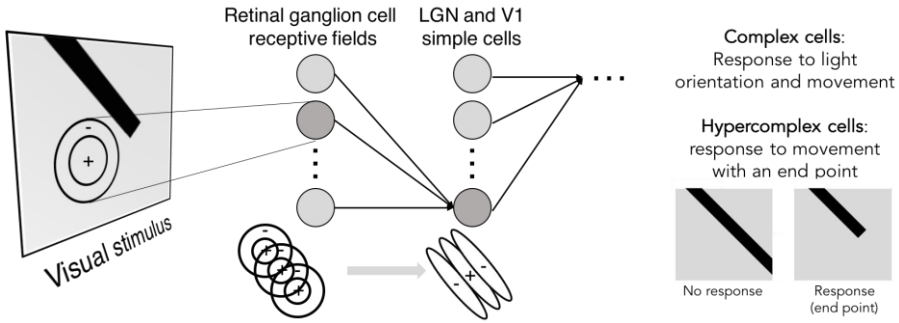
Visualization of VGG-16 by Lane McIntosh. VGG-16 architecture from [Simonyan and Zisserman 2014].



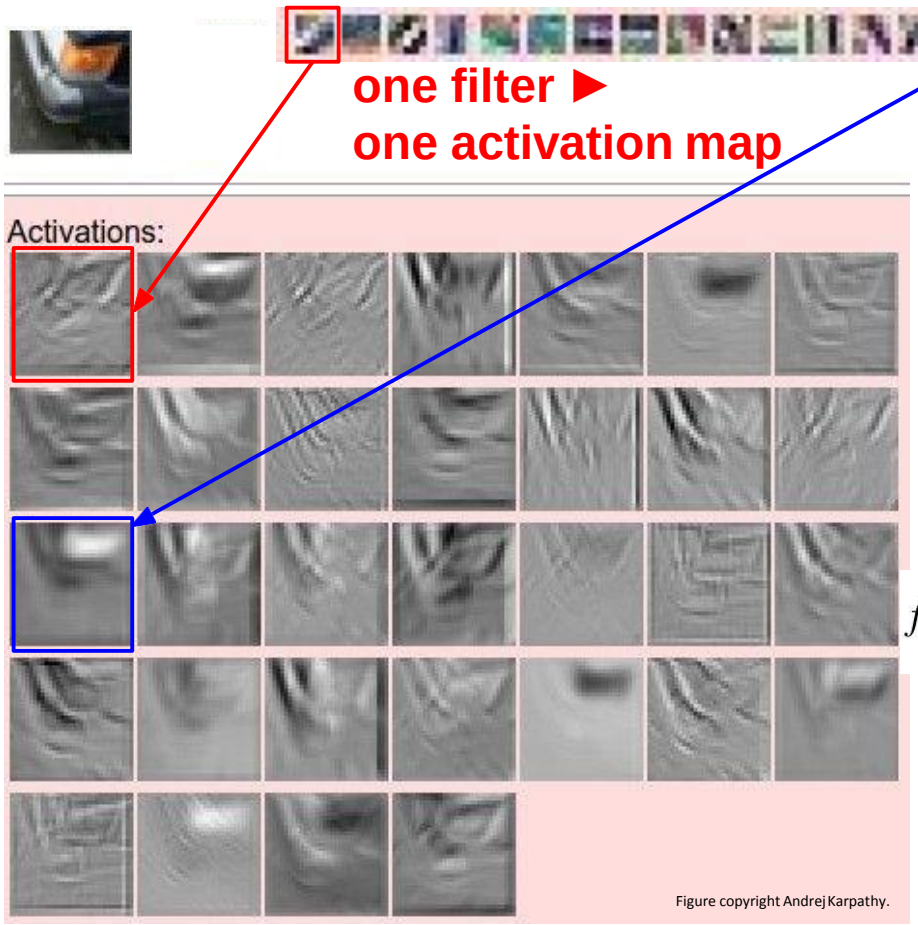
Convolution Layer

Preview





Preview



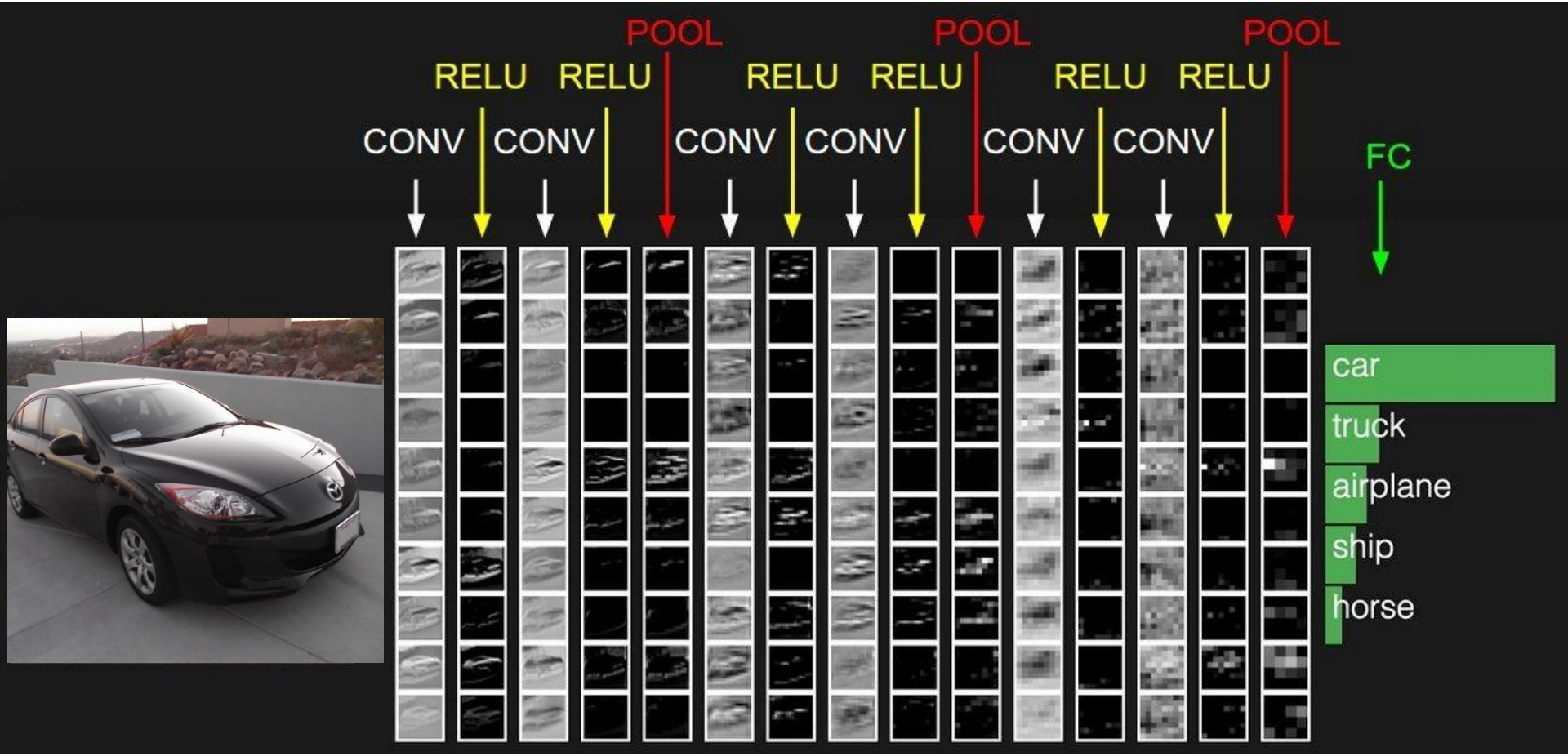
example 5x5 filters
(32 total)

We call the layer convolutional because it is related to convolution of two signals:

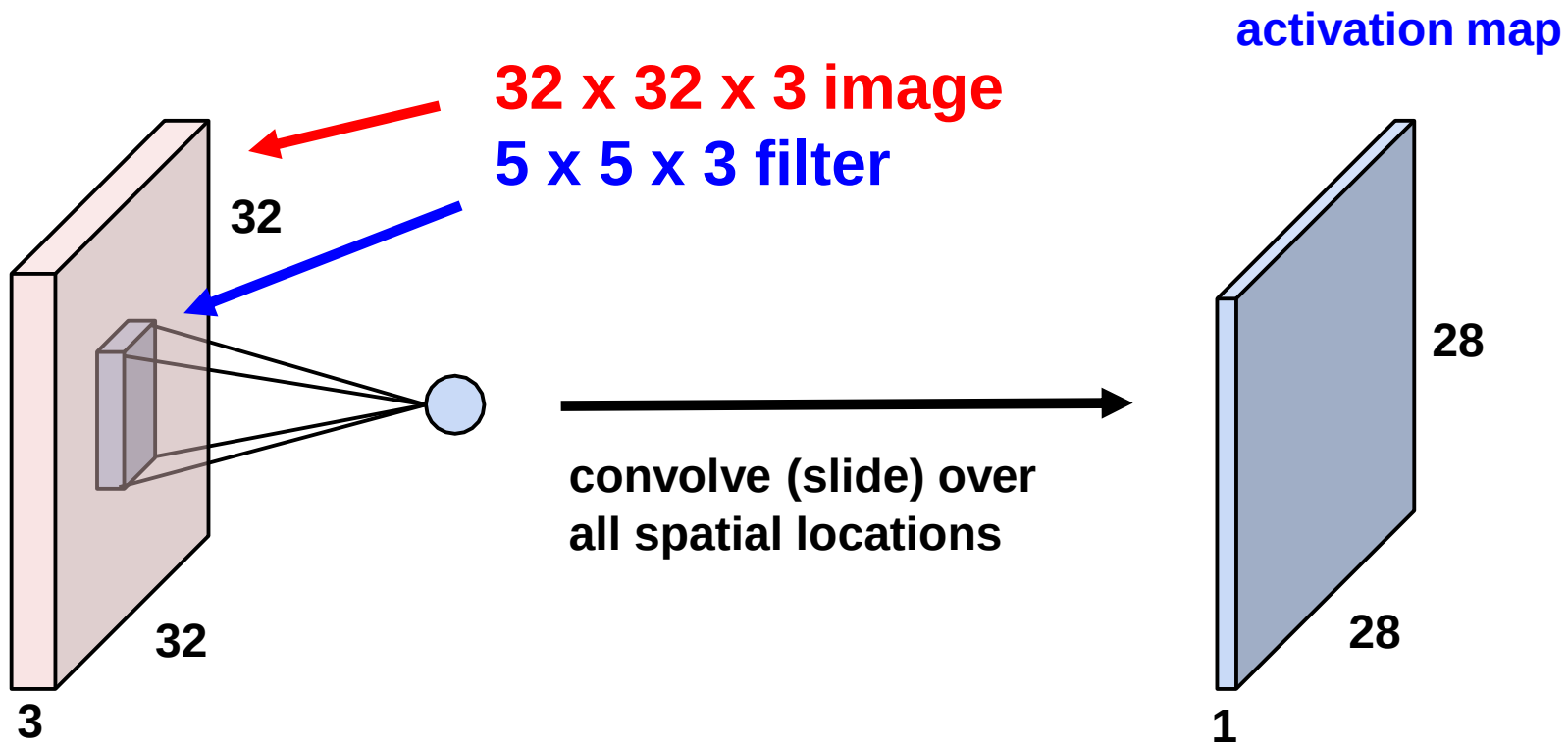
$$f[x,y] * g[x,y] = \sum_{n_1=-\infty}^{\infty} \sum_{n_2=-\infty}^{\infty} f[n_1,n_2] \cdot g[x-n_1, y-n_2]$$

elementwise multiplication and sum of a filter and the signal (image)

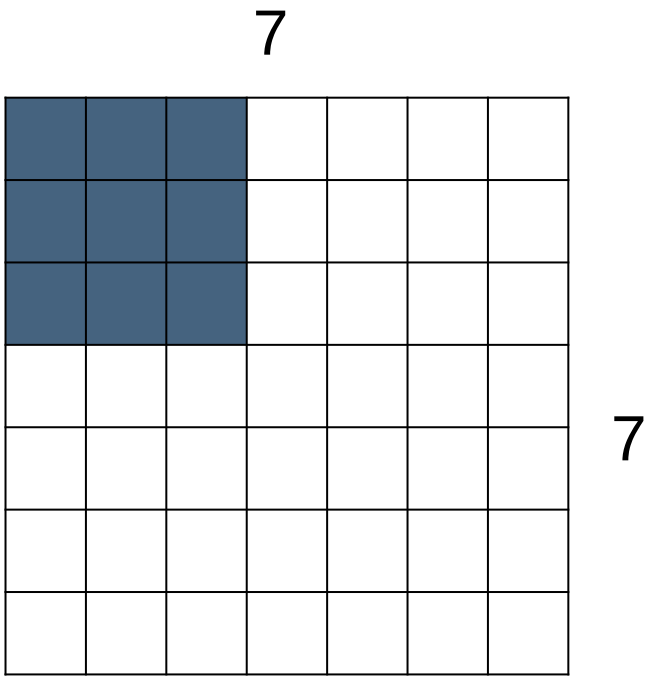
Preview


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

A closer look at spatial dimensions:

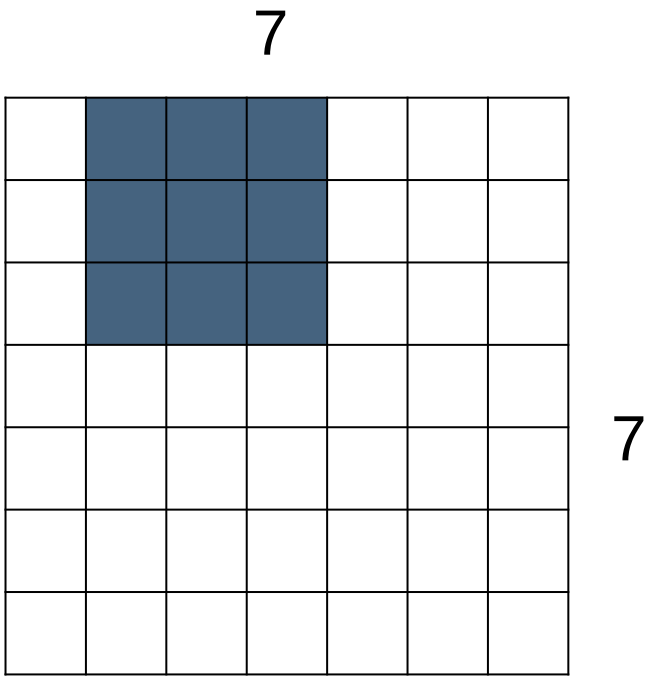


A closer look at spatial dimensions:



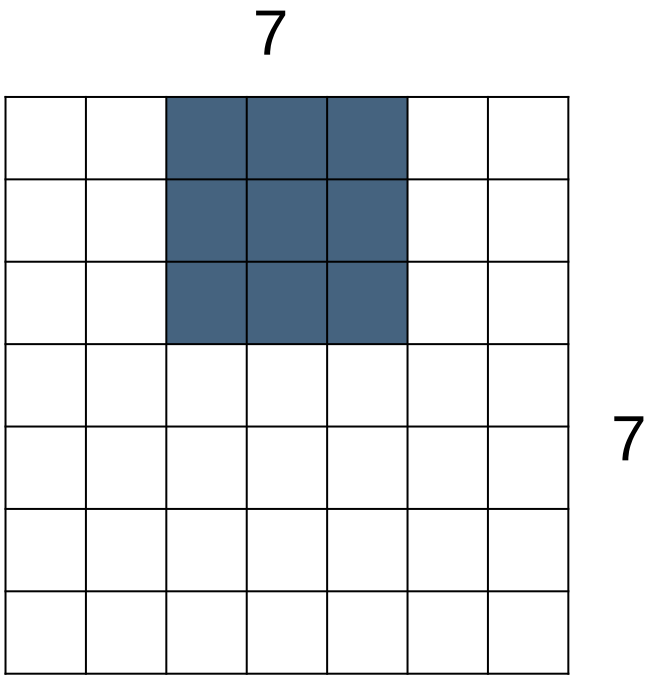
- 7 x 7 input (spatially)
- assume 3 x 3 filter

A closer look at spatial dimensions:



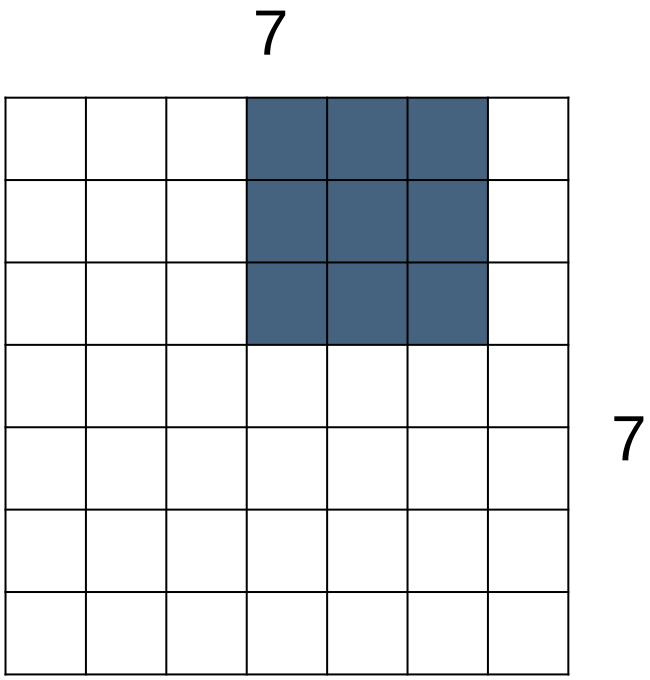
- 7 x 7 input (spatially)
- assume 3 x 3 filter

A closer look at spatial dimensions:



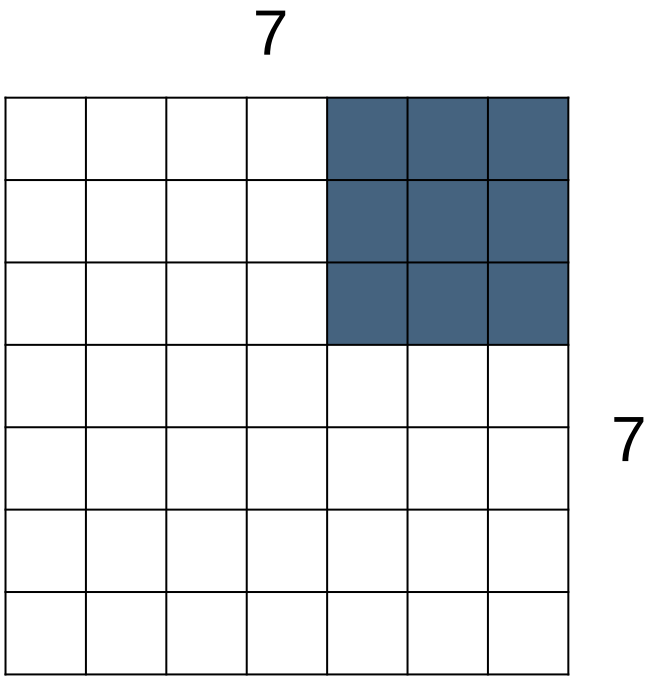
- 7 x 7 input (spatially)
- assume 3 x 3 filter

A closer look at spatial dimensions:



- 7 x 7 input (spatially)
- assume 3 x 3 filter

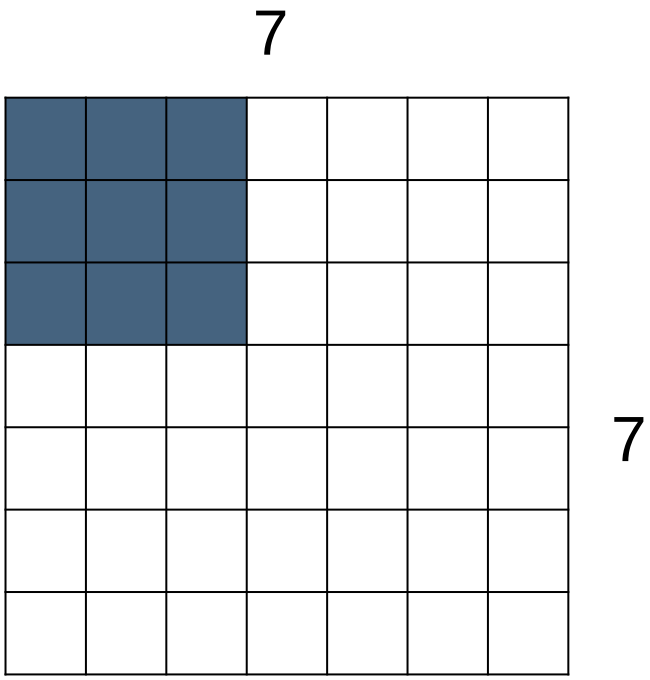
A closer look at spatial dimensions:



- 7 x 7 input (spatially)
- assume 3 x 3 filter

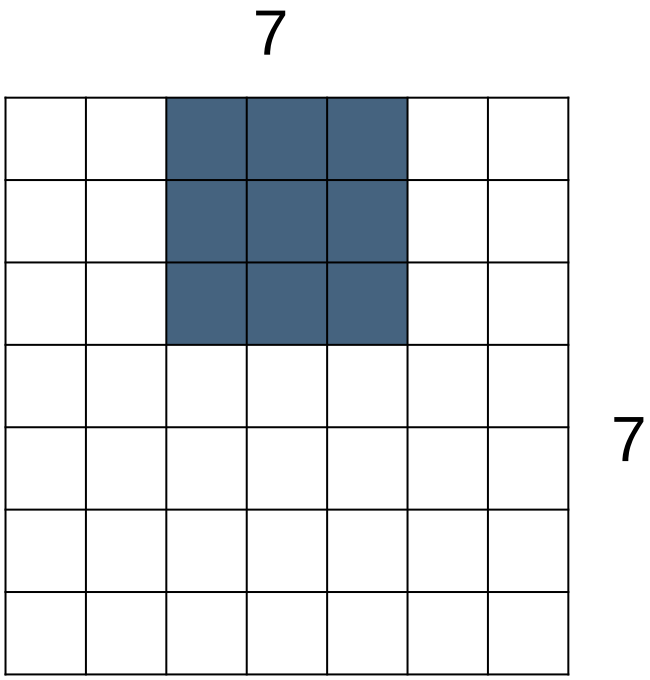
► **5 x 5 output**

A closer look at spatial dimensions:



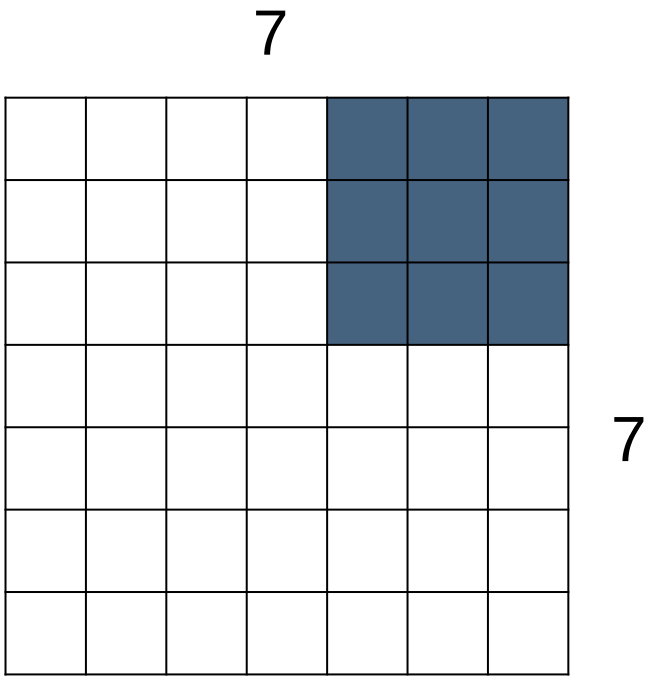
- 7 x 7 input (spatially)
- assume 3 x 3 filter
- applied **with stride 2**

A closer look at spatial dimensions:



- 7 x 7 input (spatially)
- assume 3 x 3 filter
- applied **with stride 2**

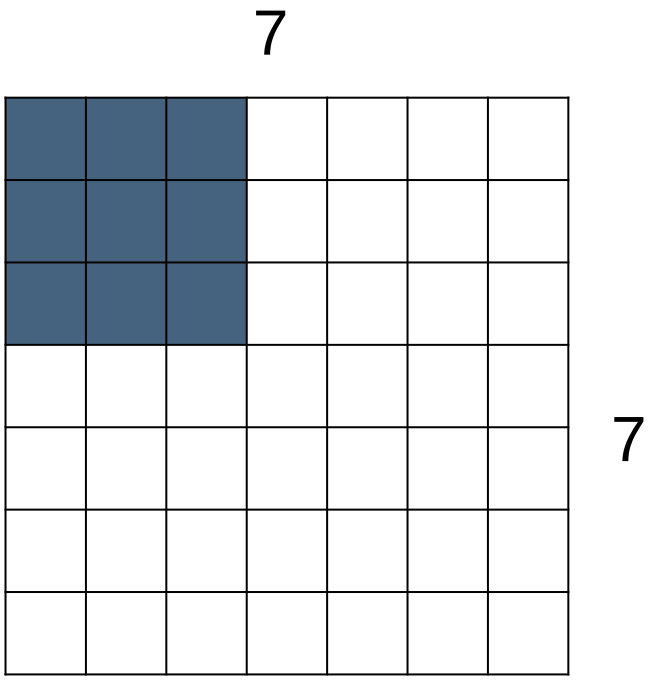
A closer look at spatial dimensions:



- 7 x 7 input (spatially)
- assume 3 x 3 filter
- applied **with stride 2**

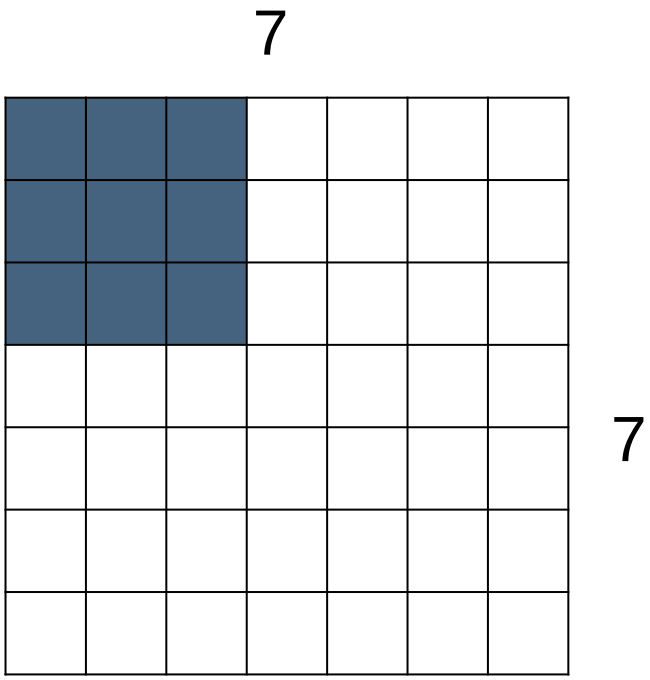
► **3x3 output!**

A closer look at spatial dimensions:



- 7 x 7 input (spatially)
- assume 3 x 3 filter
- applied **with stride 3?**

A closer look at spatial dimensions:

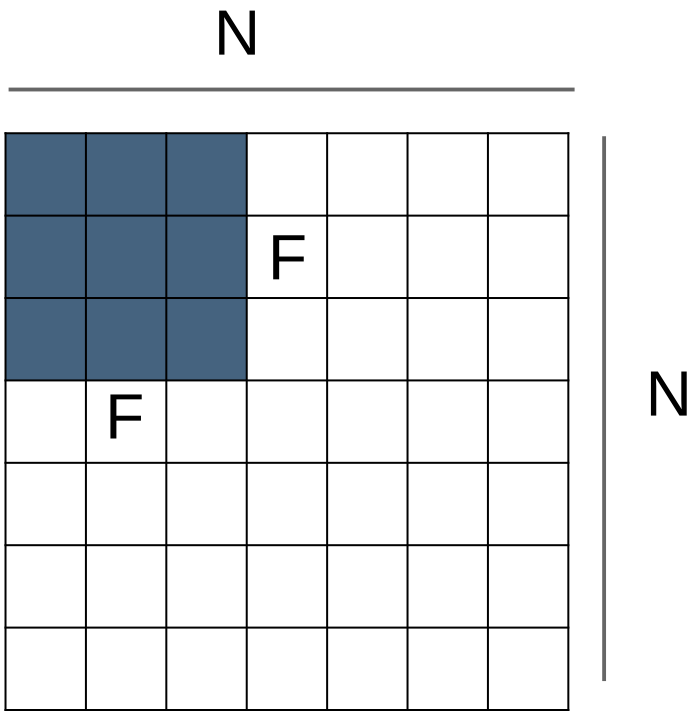


- 7 x 7 input (spatially)
- assume 3 x 3 filter
- applied **with stride 3?**

Doesn't fit!

- cannot apply 3 x 3 filter on a 7 x 7 input with stride 3.

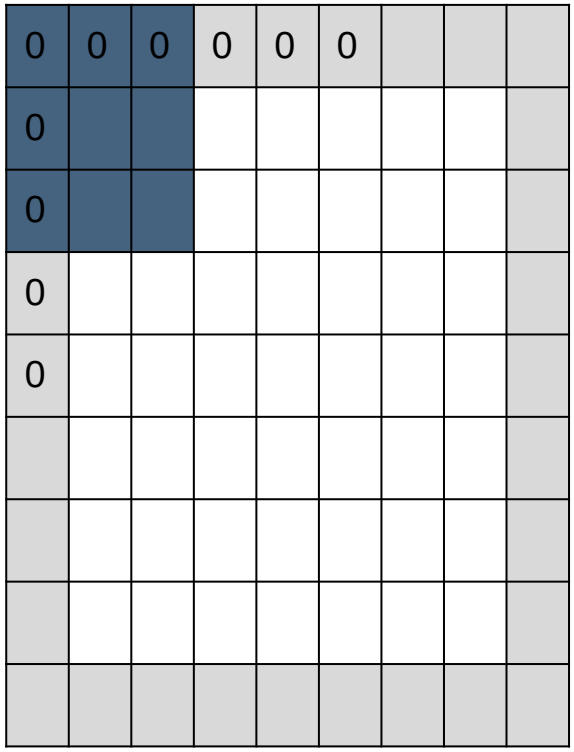
A closer look at spatial dimensions:



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 :\

In practice: Common to zero pad the border



E.g.:

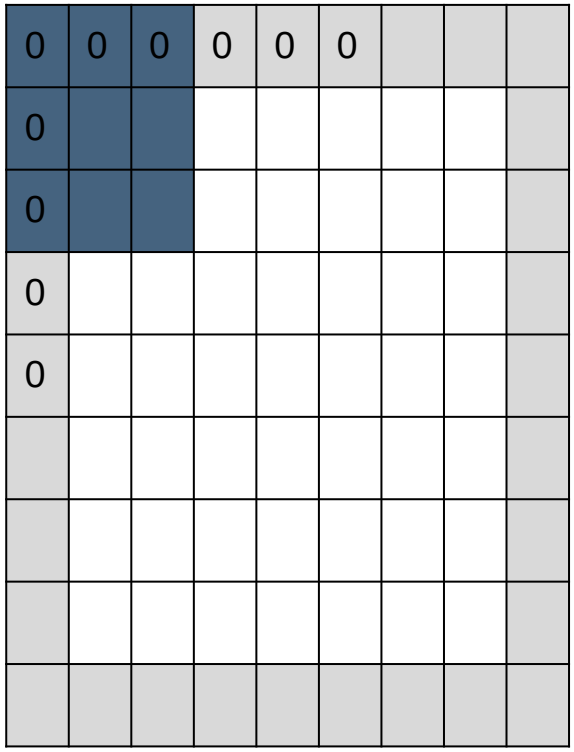
- input **7 x 7**
- **3 x 3** filter,
- applied with **stride 1**
- **Pad with 1 pixel** border

► what is the output?

(recall:)

(N - F) / stride + 1

In practice: Common to zero pad the border



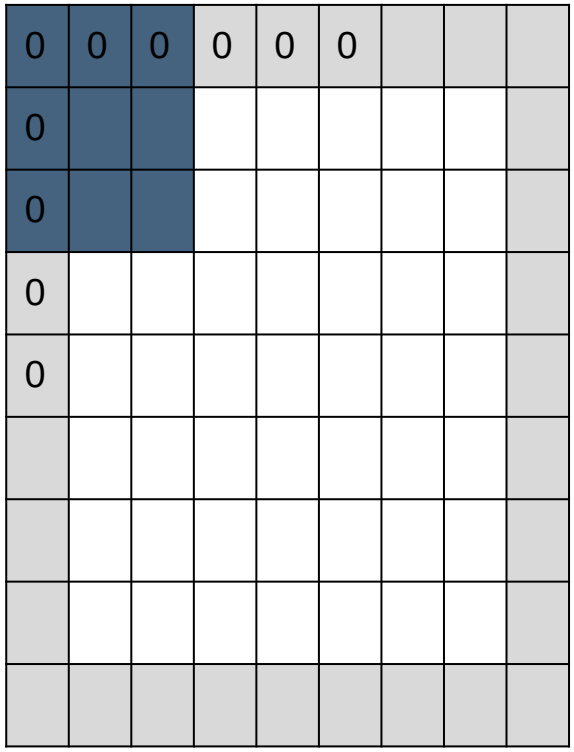
E.g.:

- input **7 x 7**
- **3 x 3** filter,
- applied with **stride 1**
- **Pad with 1 pixel** border

► what is the output?

7 x 7 output!

In practice: Common to zero pad the border



E.g.:

- input **7 x 7**
- **3 x 3** filter,
- applied with **stride 1**
- **Pad with 1 pixel** border

► what is the output?

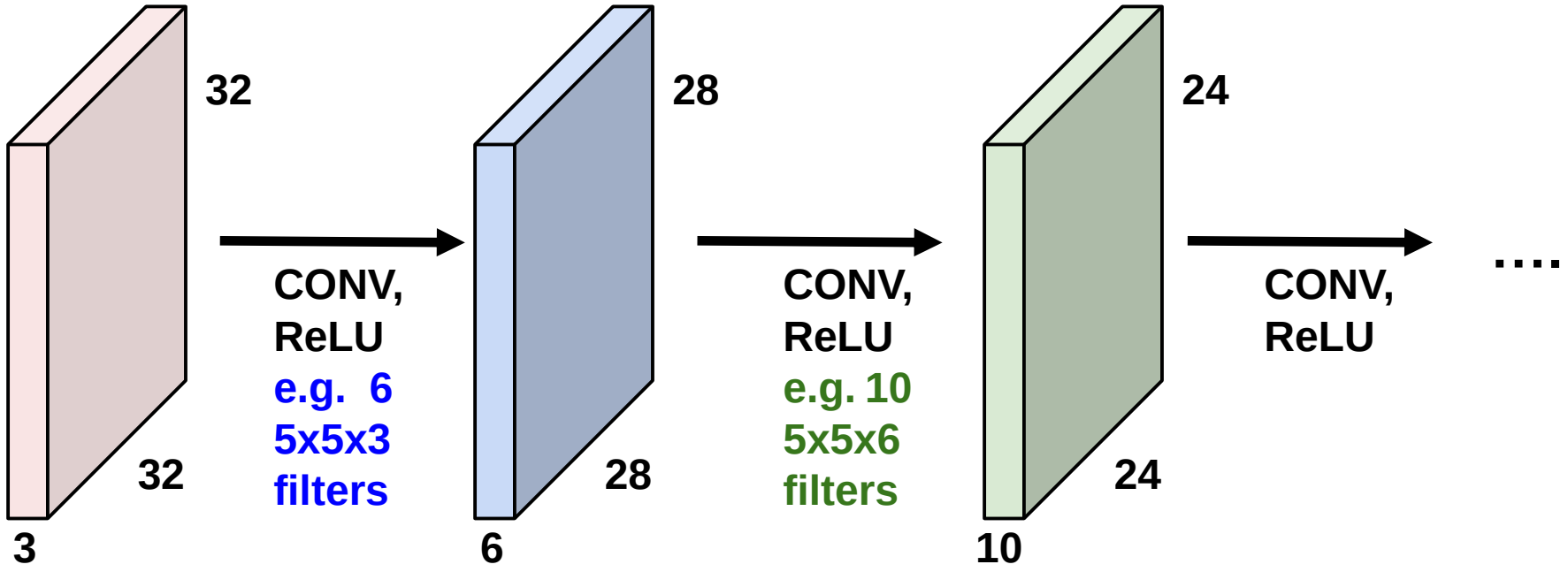
7 x 7 output!

In general, common to see CONV layers with stride 1, filters of size F x F, and zero-padding with (F-1) / 2 (will preserve size spatially).

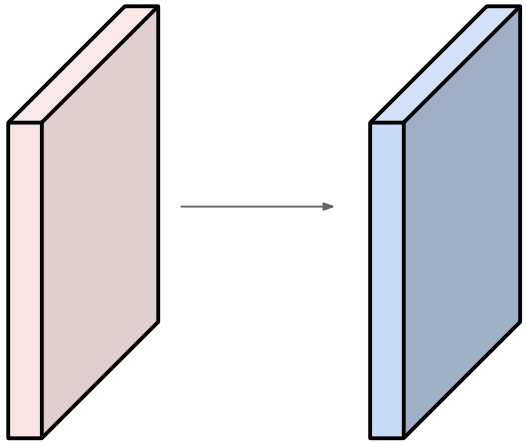
E.g. F = 3 ► zero pad with 1
F = 5 ► zero pad with 2
F = 7 ► zero pad with 3

Remember back to:

E.g. 32 x 32 input convolved repeatedly with 5x5 filters shrinks volumes spatially! (32 ► 28 ► 24 ...). Shrinking too fast is not good, doesn't work well.



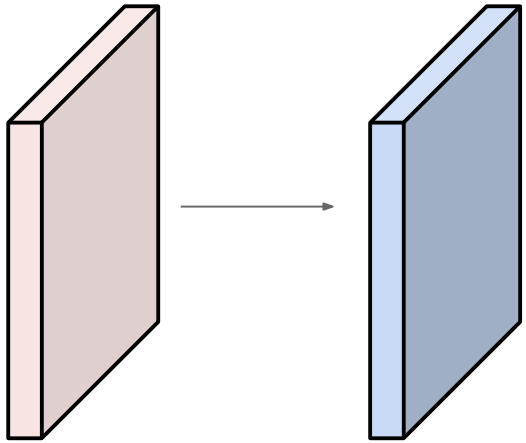
Example:



Input volume: **32 x 32 x 3**
10 5 x 5 filters with stride 1, pad 2

Output volume size?

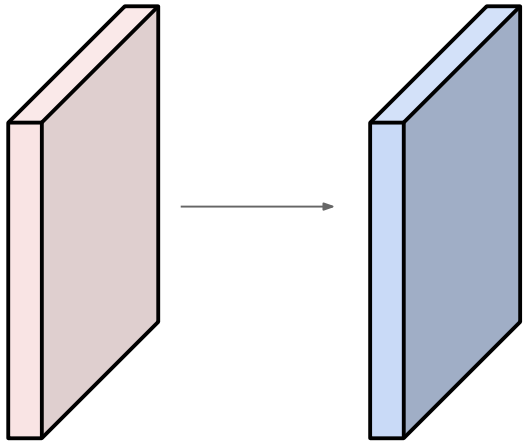
Example:



Input volume: **32 x 32 x 3**
10 5 x 5 filters with stride 1, pad 2

Output volume size:
(32 + 2 * 2 - 5) / 1 + 1 = 32
spatially, so
32 x 32 x 10

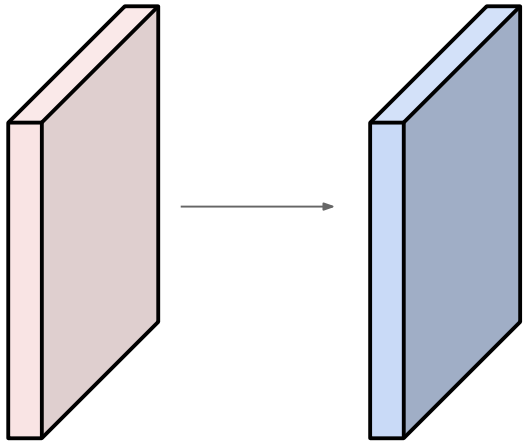
Example:



Input volume: **32 x 32 x 3**
10 5 x 5 filters with stride 1, pad 2

Number of parameters in this layer?

Example:

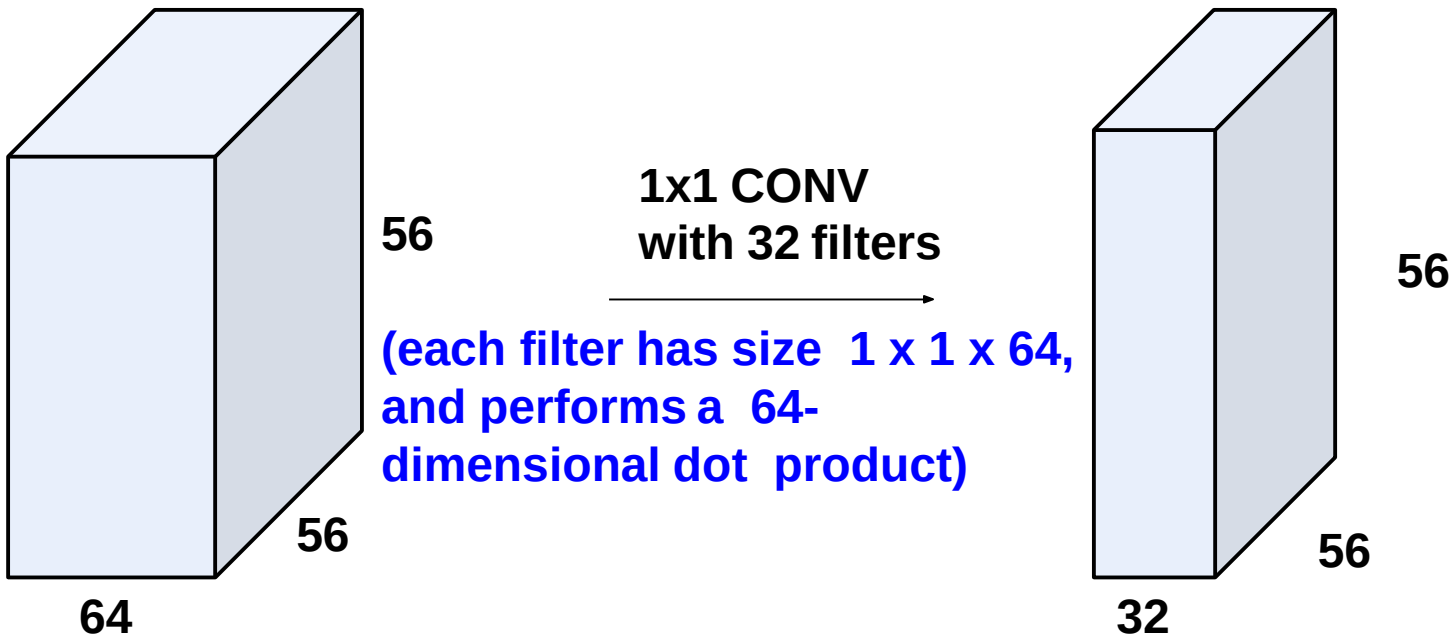


Input volume: **32 x 32 x 3**
10 5 x 5 filters with stride 1, pad 2

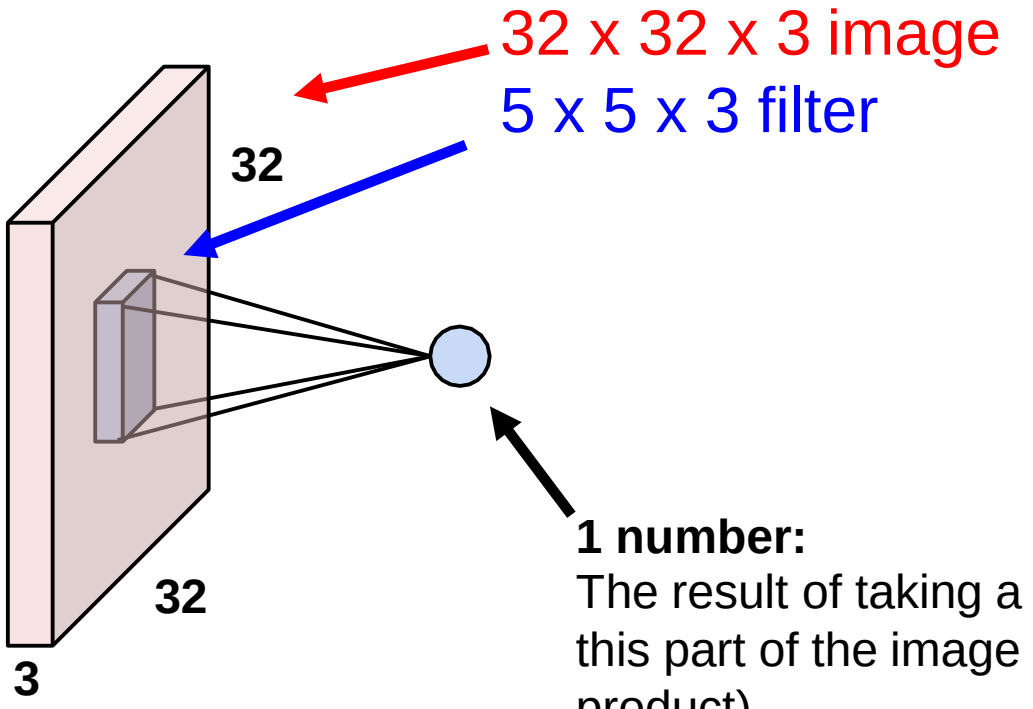
Number of parameters in this layer?
Each filter has 5 * 5 * 3 + 1 = 76 params *(+1 for bias)*

► 76 * 10 = **760**

(btw, 1x1 convolution layers make perfect sense)

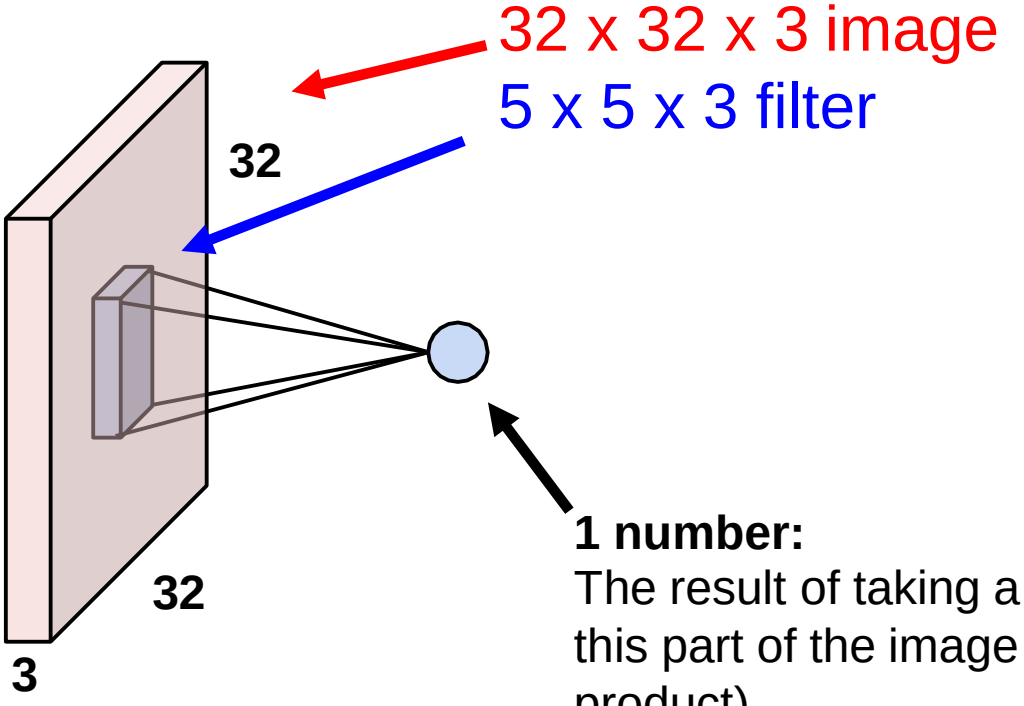


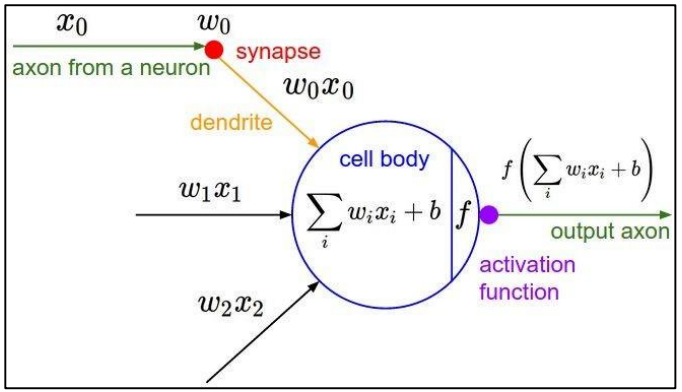
The Brain / Neuron view of the CONV Layer



1 number:
The result of taking a dot product between the filter and this part of the image (i.e. 5 * 5 * 3 = 75-dimensional dot product).

The Brain / Neuron view of the CONV Layer

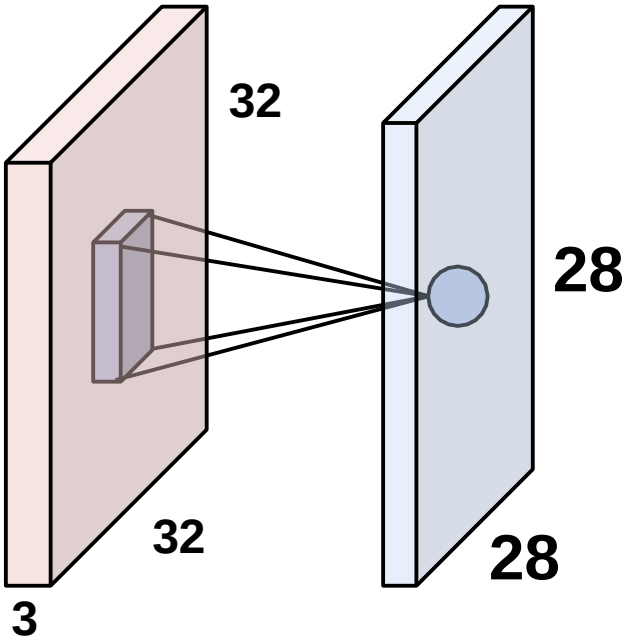


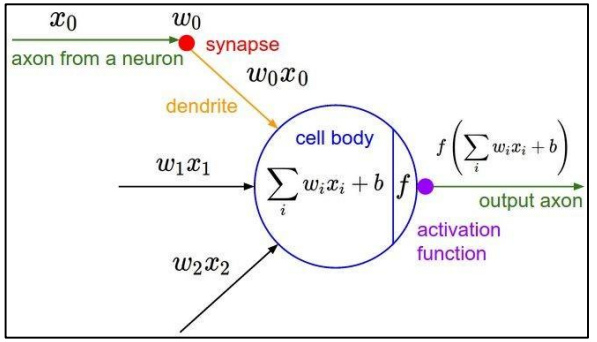


It's just a neuron with local connectivity ..

1 number:
The result of taking a dot product between the filter and this part of the image (i.e. 5 * 5 * 3 = 75-dimensional dot product).

The Brain / Neuron view of the CONV Layer



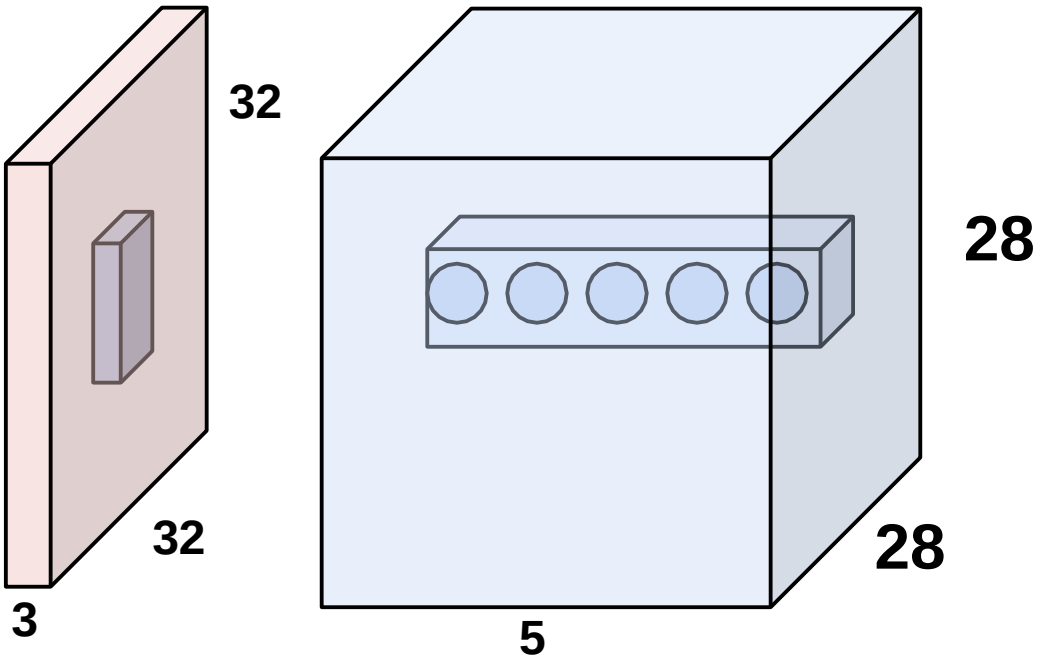


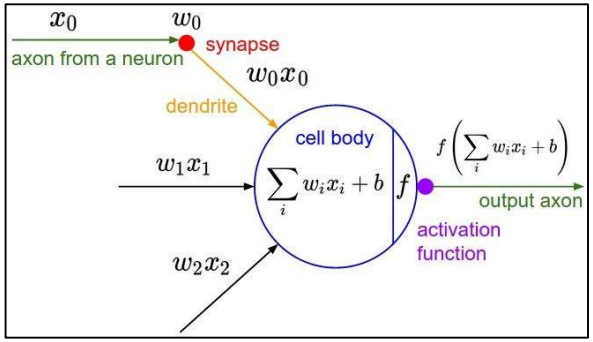
An activation map is a 28 x 28 sheet of neuron outputs:

1. **Each is connected to a small region in the input.**
2. **All of them share parameters.**

"5 x 5 filter" ► "5 x 5 receptive field for each neuron"

The Brain / Neuron view of the CONV Layer





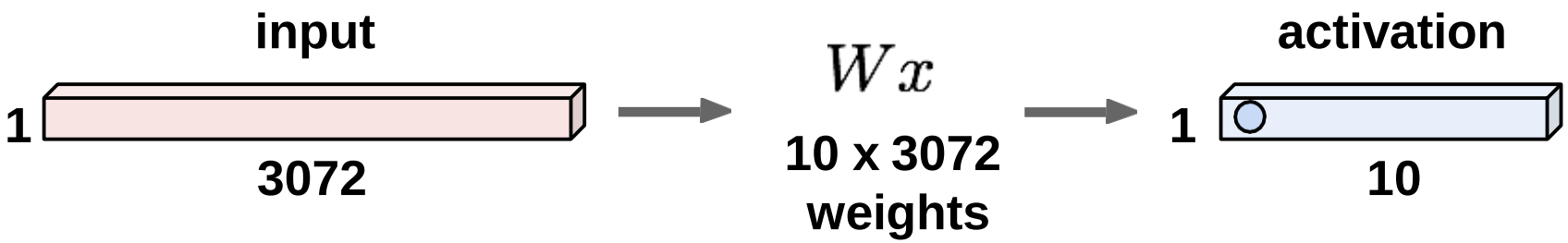
E.g. with 5 filters, CONV layer consists of neurons arranged in a 3D grid (28 x 28 x 5)

There will be 5 different neurons all looking at the same region in the input volume

Reminder:
Fully Connected Layer

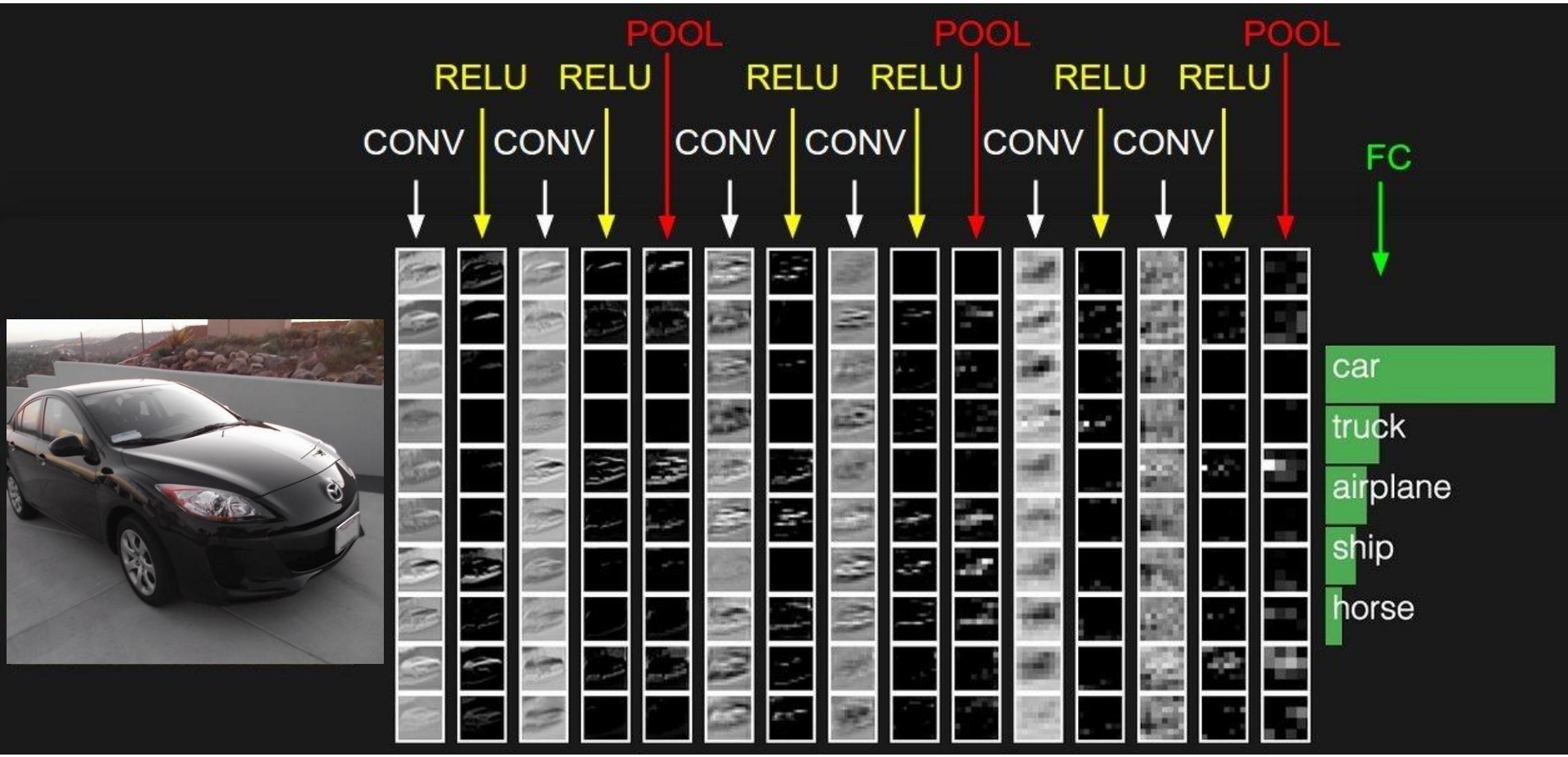
32 x 32 x 3 image ► stretch to 3072 x 1

Each neuron looks at the full input volume



1 number:
the result of taking a dot product between a row of W and the input (a 3072-dimensional dot product)

Two more layers to go ..
Pooling (POOL) / Fully Connected (FC)



Why Pooling?

Subsampling pixels will not change the object.

bird

Subsampling

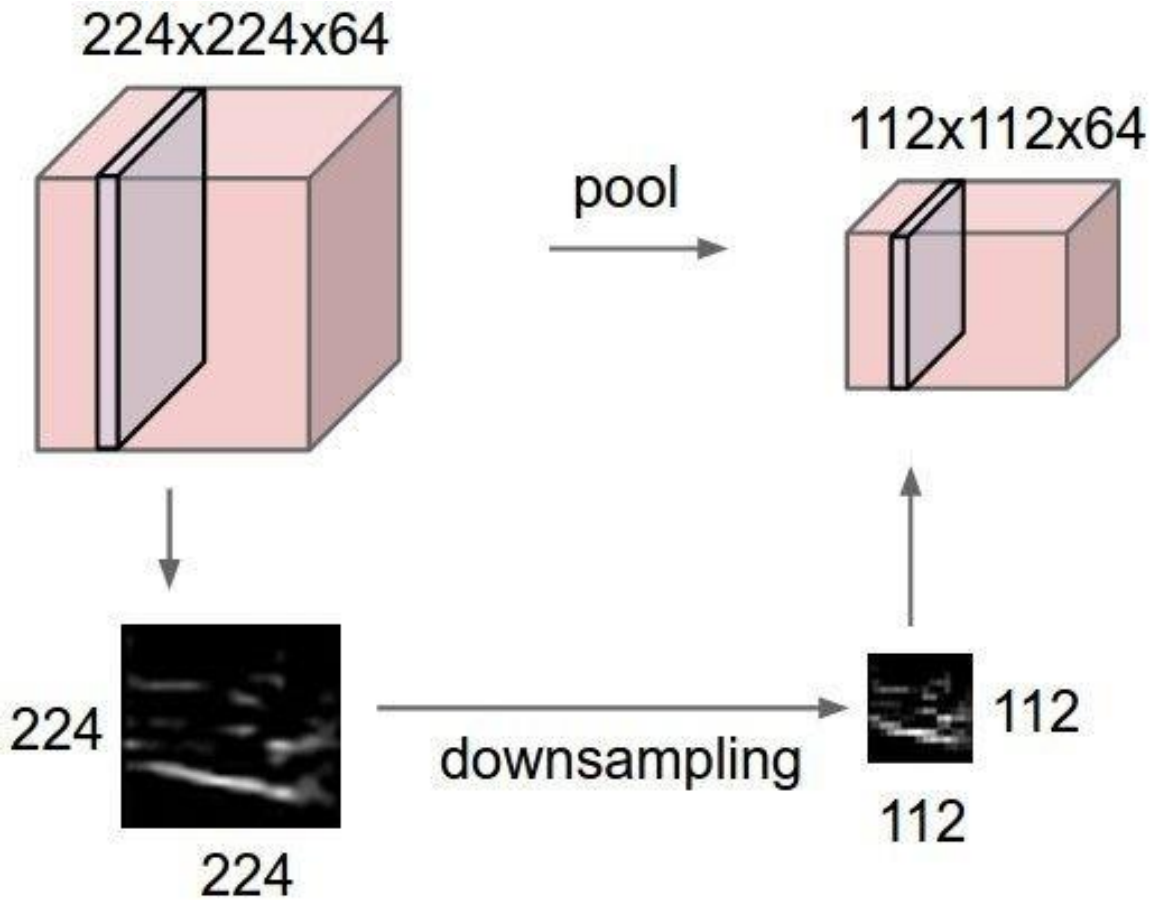
bird

We can subsample the pixels to make image smaller

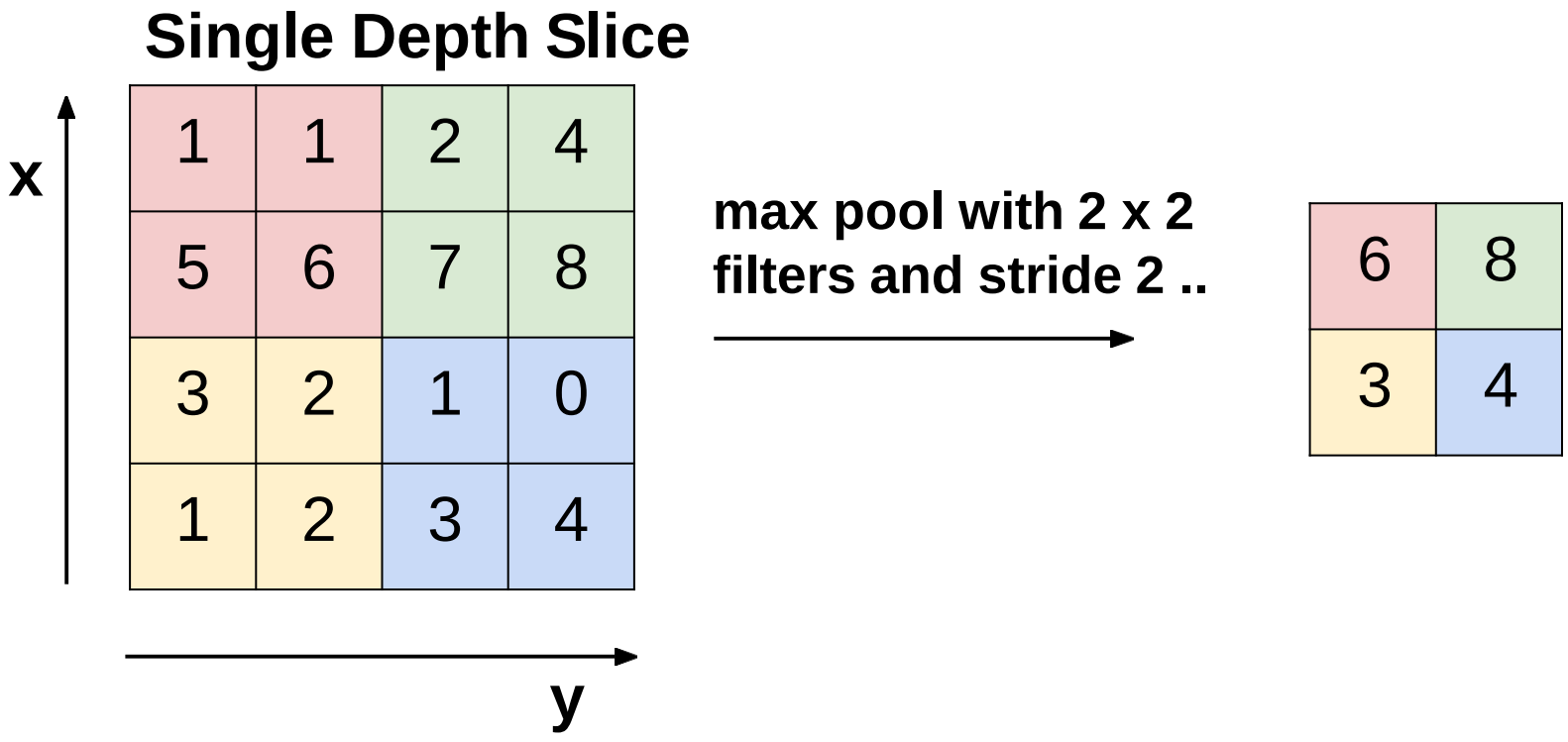
fewer parameters to characterize the image.

- Makes the representations smaller and more manageable.
- Operates over each activation map independently:

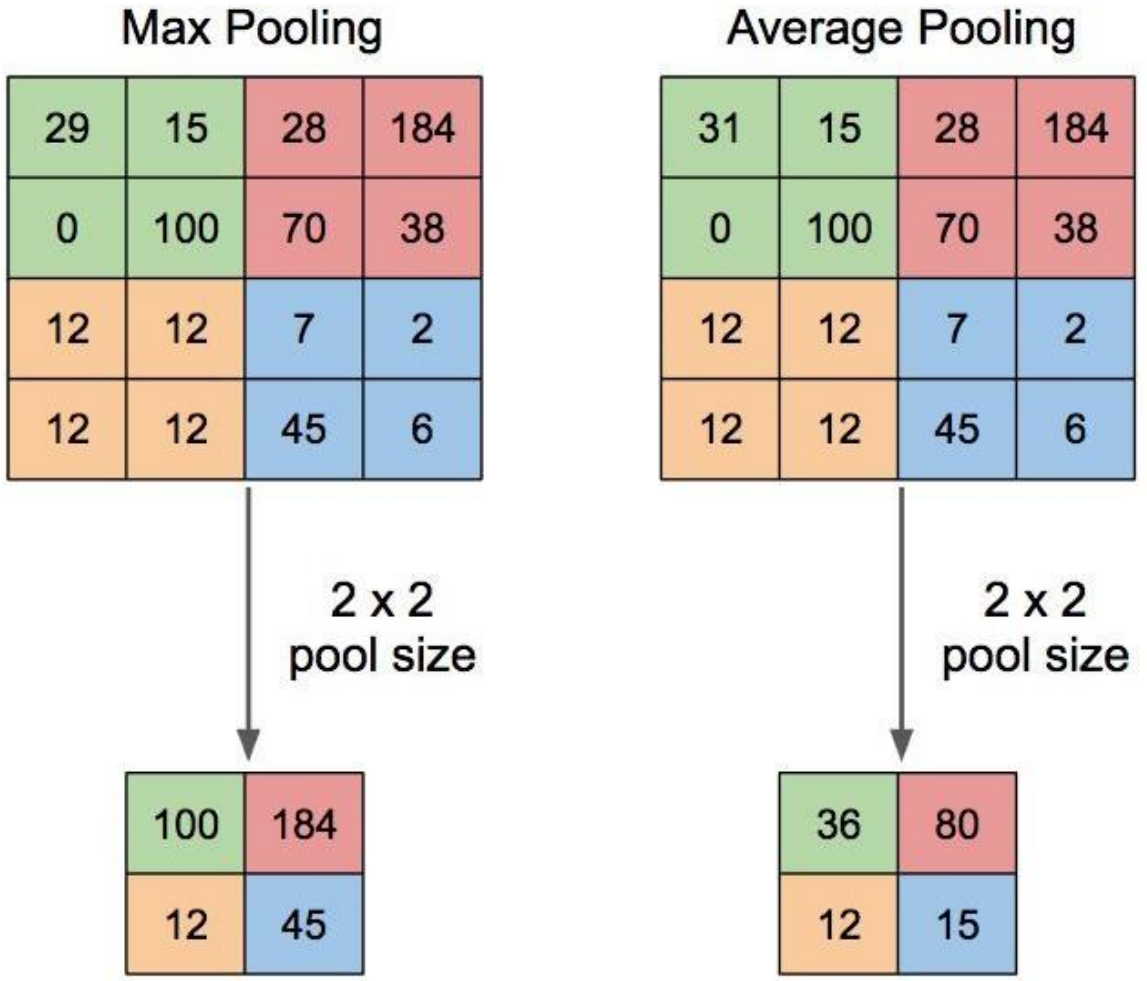
Pooling Layer



MAX Pooling


Single Depth Slice
x
y
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
max pool with 2 x 2
filters and stride 2 ..
6 8
3 4

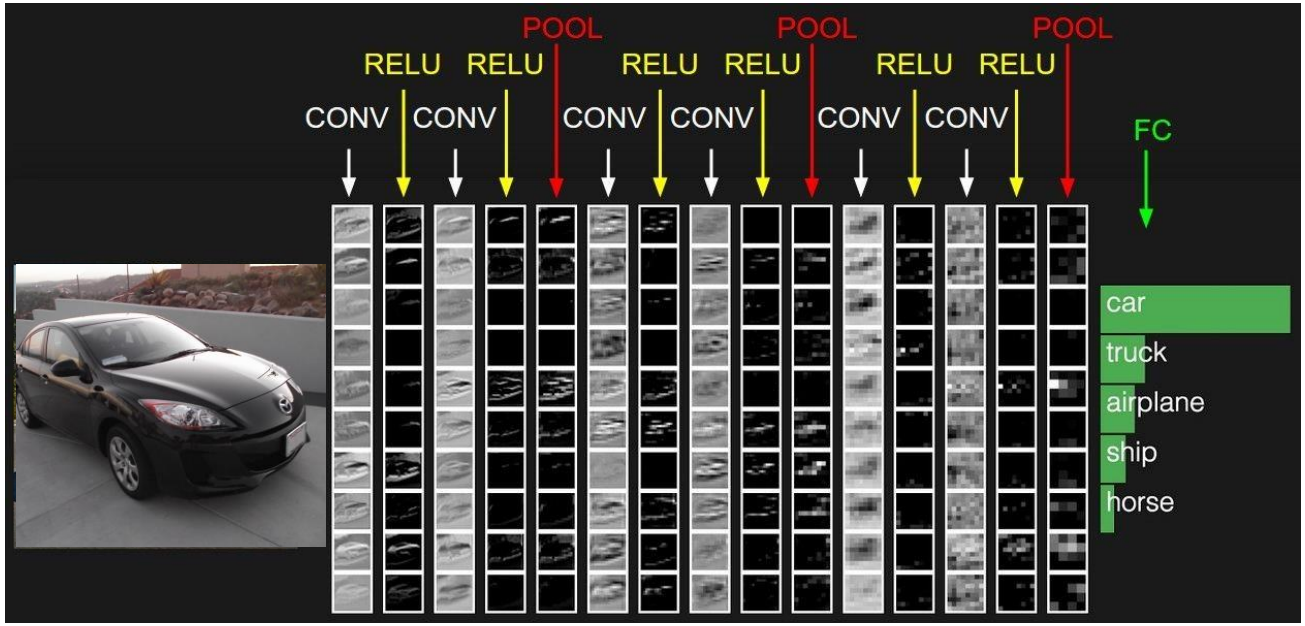
Pooling Layer



- Convolutional layers provide activation maps.
- Pooling layer applies non-linear down-sampling on activation maps.
- Pooling is aggressive (discard info); the trend is to use smaller filter size and abandon pooling.

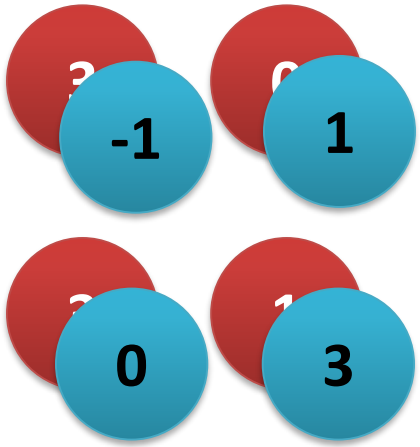
Fully Connected Layer (FC layer)

- Contains neurons that connect to the entire input volume, as in ordinary Neural Networks

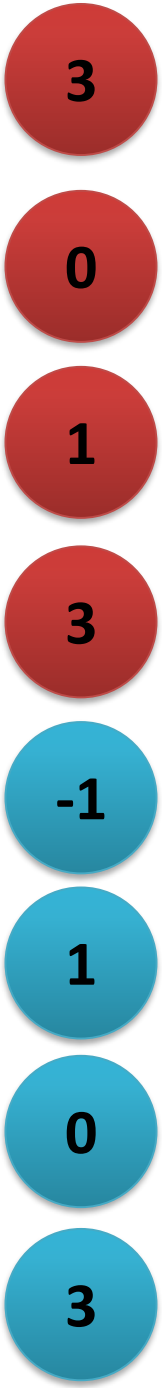


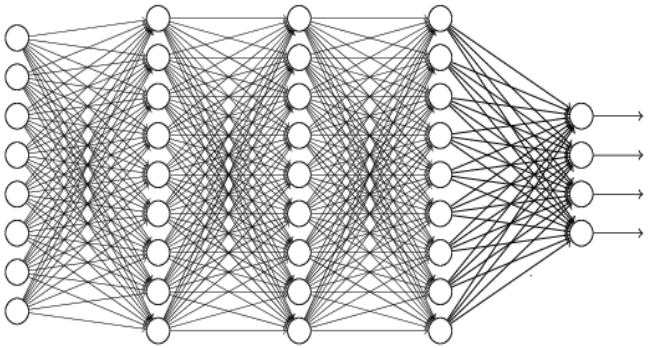
Flattening

Fully Connected Layer (FC layer)



Flattened





Fully Connected Feedforward network

A CNN compresses a fully connected network in two ways:

- **Reducing number of connections.**
- **Shared weights on the edges.**
- **“Max” pooling further reduces the complexity.**

Thanks! … *Questions?*