Machine Learning

Lecture 13: Convolutional Neural Networks

Fall 2023

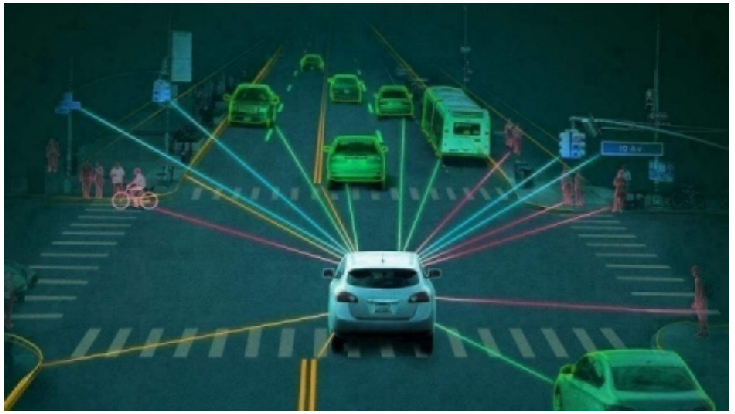
Instructor: Xiaodong Gu

Vision in the Wild

The Rise and Impact of Computer Vision

Robotics

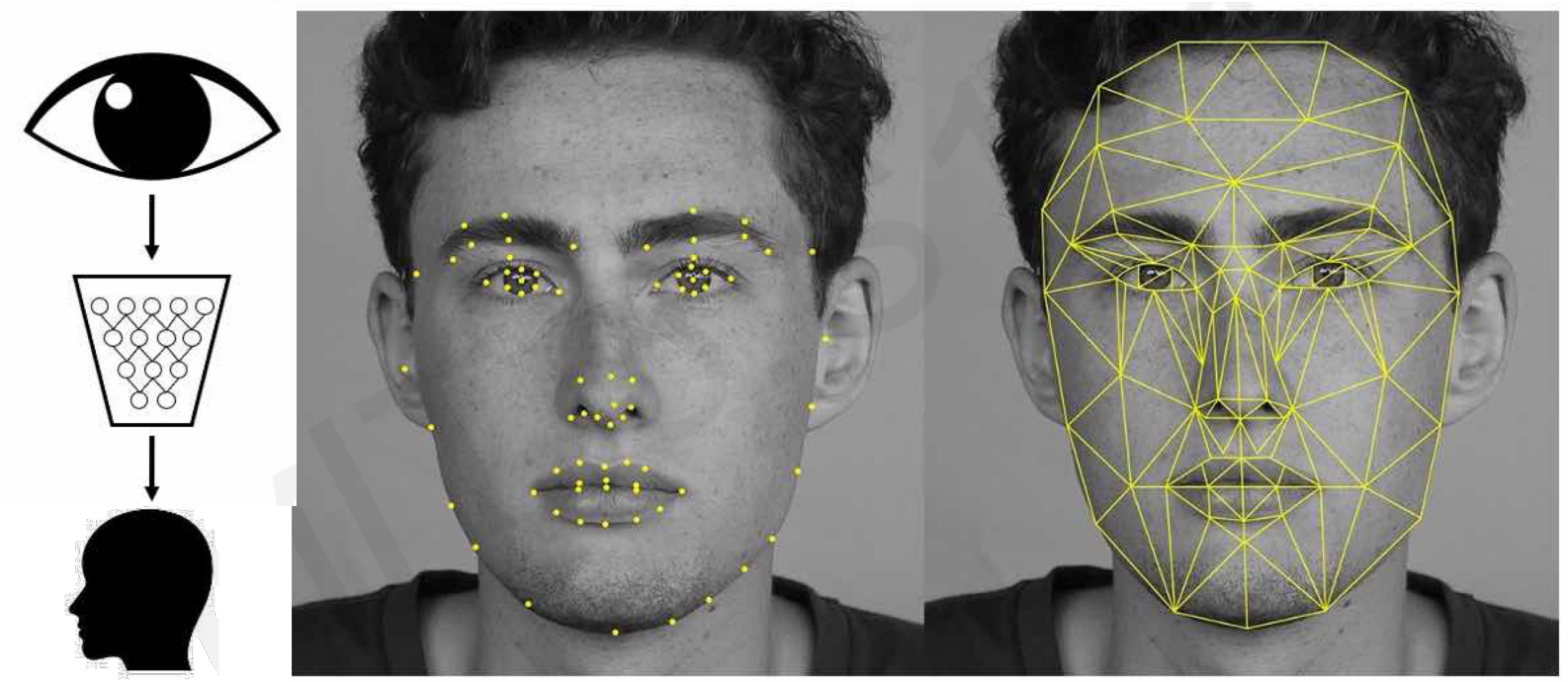
Autonomous driving



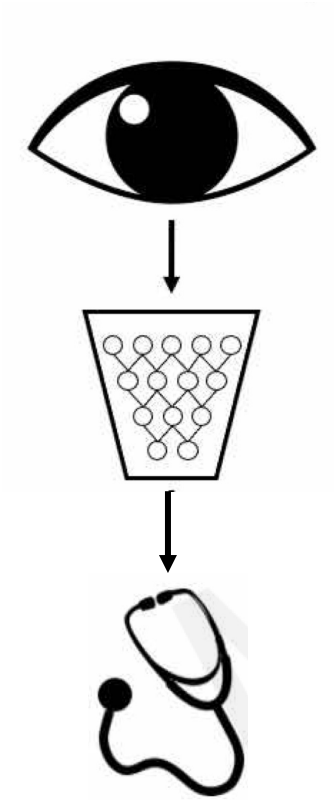
Military

Biology & Medicine

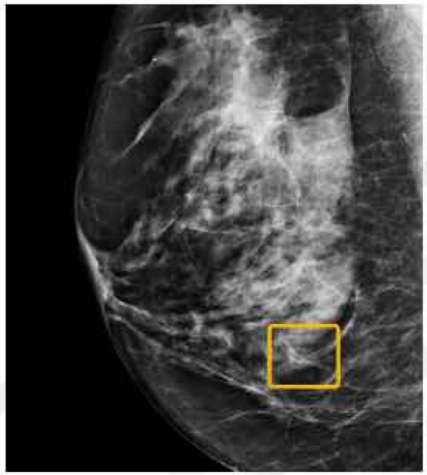
Impact: Facial Detection & Recognition



Impact: Medicine, Biology, Healthcare



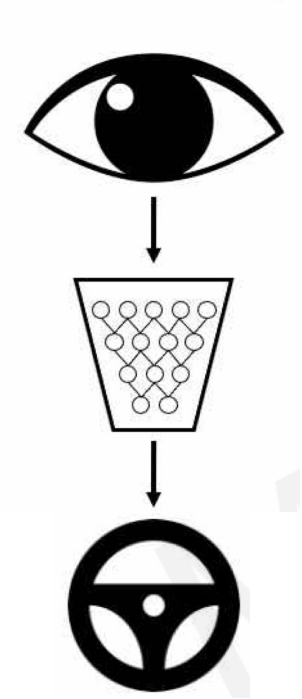
Breast cancer



COVID-19

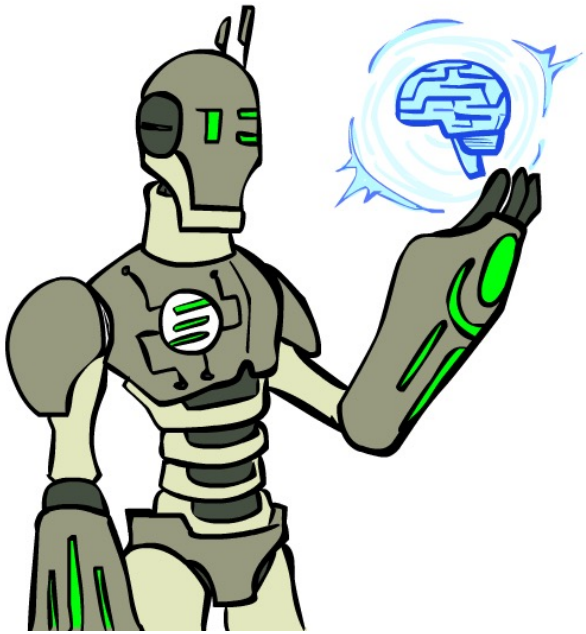
Skin cancer

Impact: Self-Driving Cars



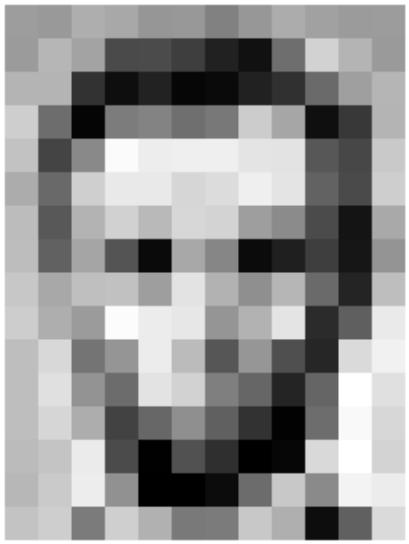
Today

- Feedforward neural networks
- **Convolutional neural networks**
- Recurrent neural networks
- …



What Computers See?

- **Images are numbers**



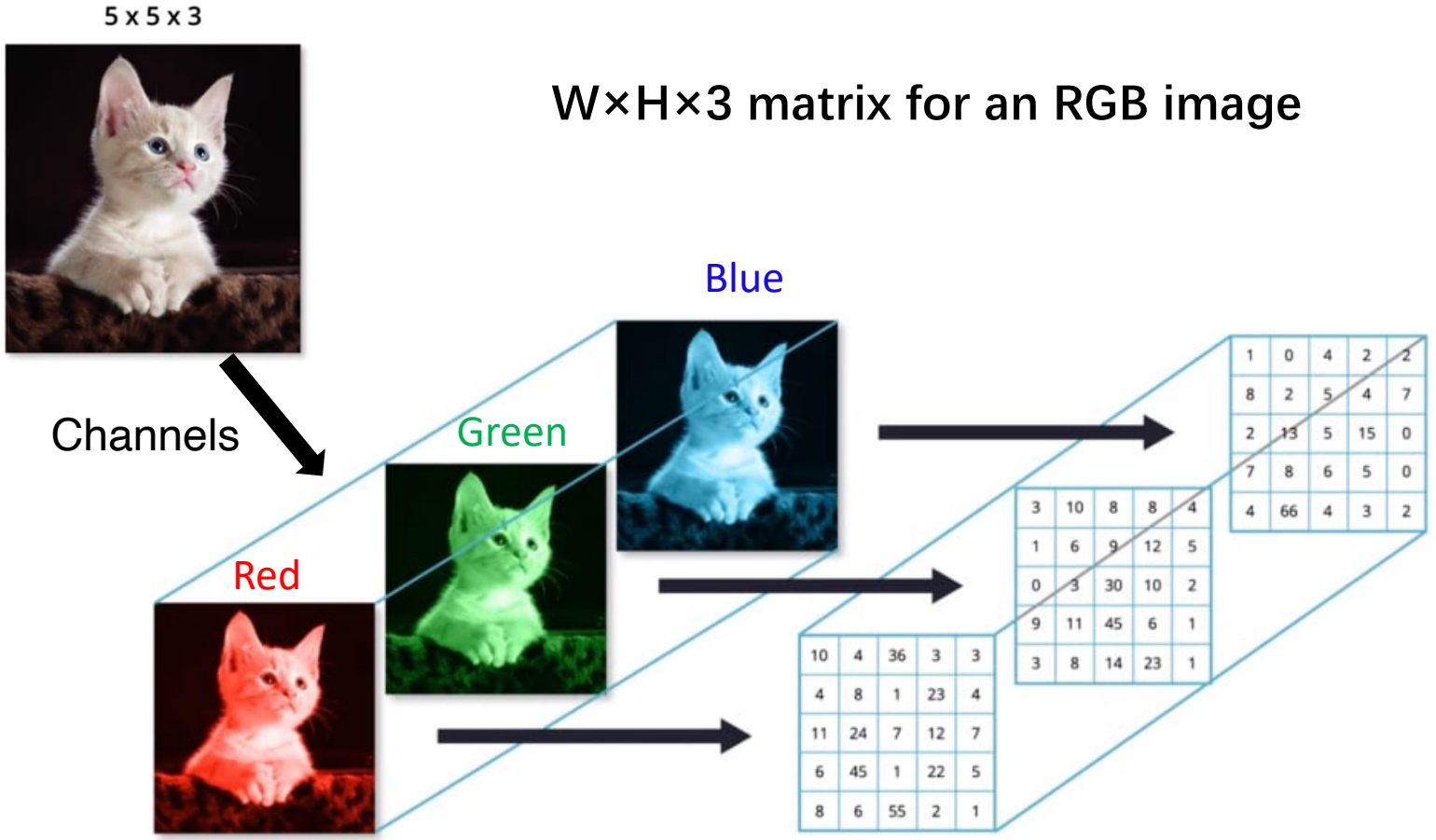
157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

An image is just a matrix of numbers [0,255]

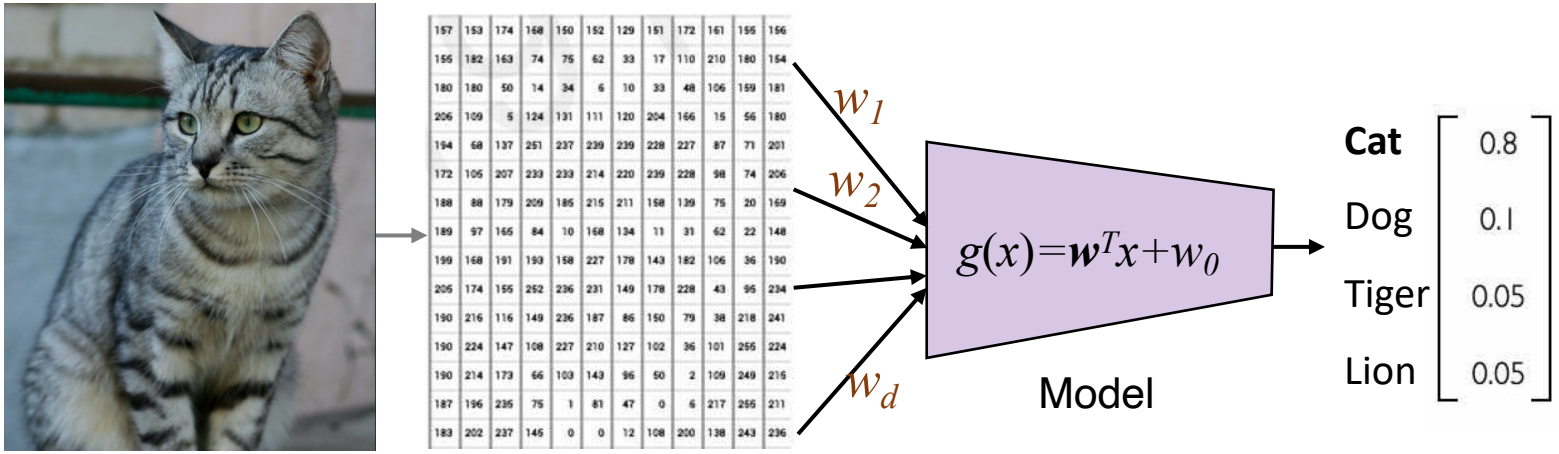
What Computers See?

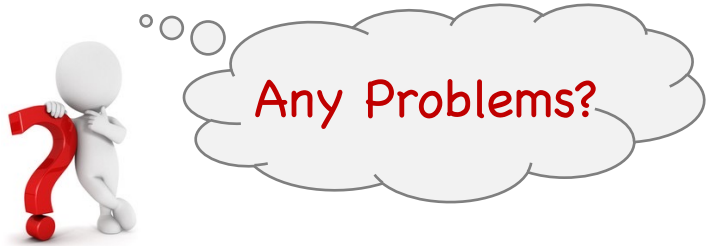
- Images are numbers



How can computer ~~see~~understand images?

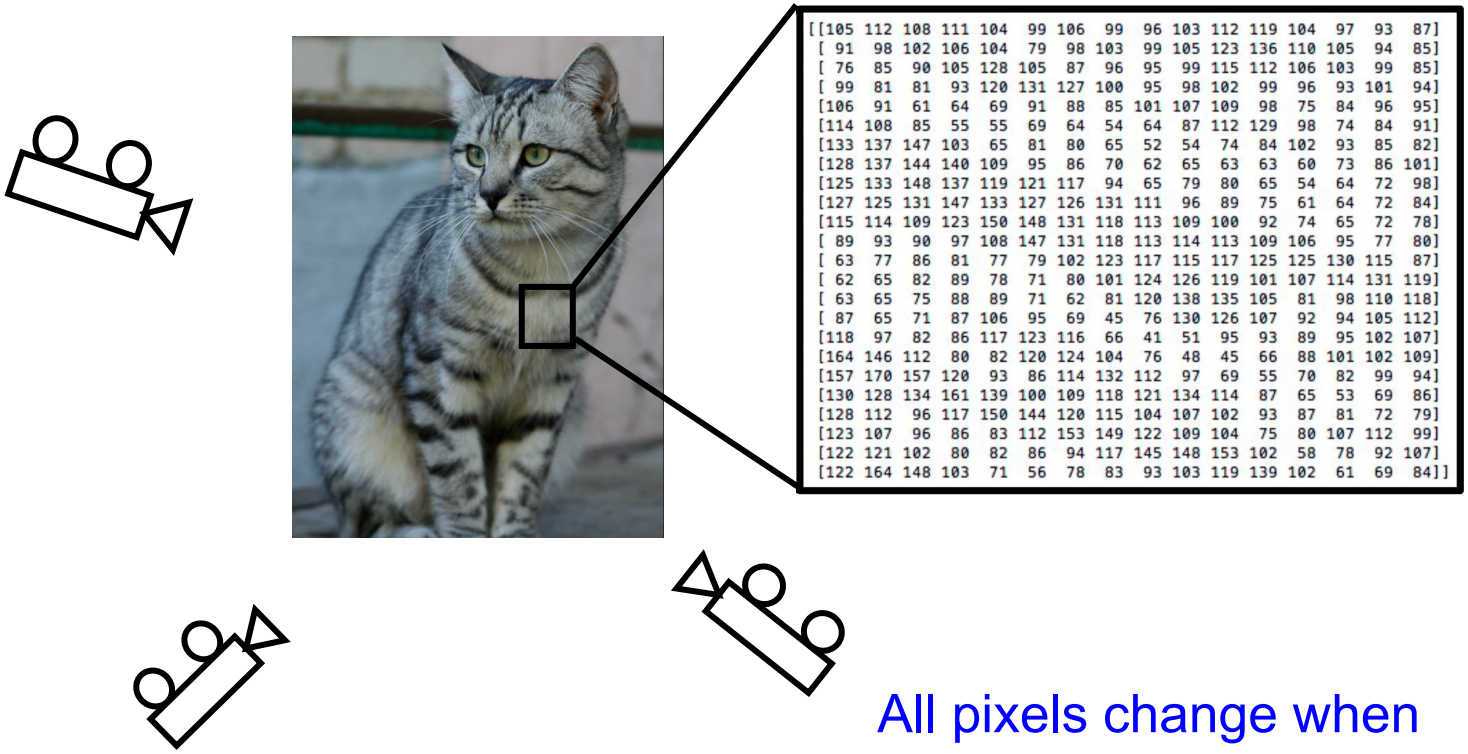
- **Idea #1: Pixels as features?**





Challenges of Image Modeling

- **Viewpoint Variation**



All pixels change when the camera moves!

Challenges of Image Modeling

Illumination

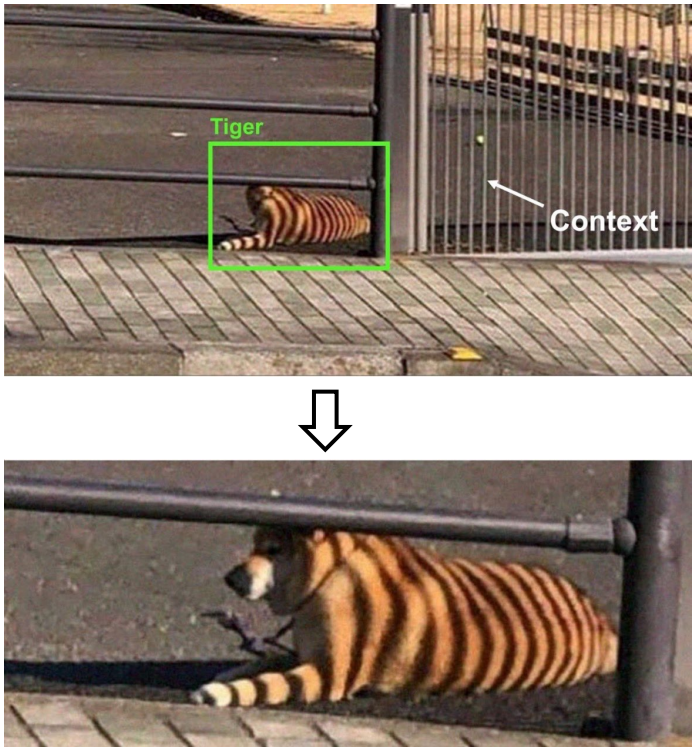
Occlusion

Deformation

Background Clutter

Challenges of Image Modeling

Context



Intra-class variation

Manual Features?

- **Idea #2: Manual features?**

Nose,
Eyes,
Mouth

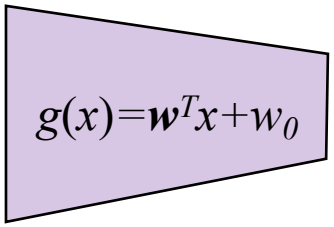
Wheels,
License Plate,
Headlights

Door,
Windows,
Step

...

Nose,
Eyes,
Mouth,
Hair,
...

Manual Features



Model

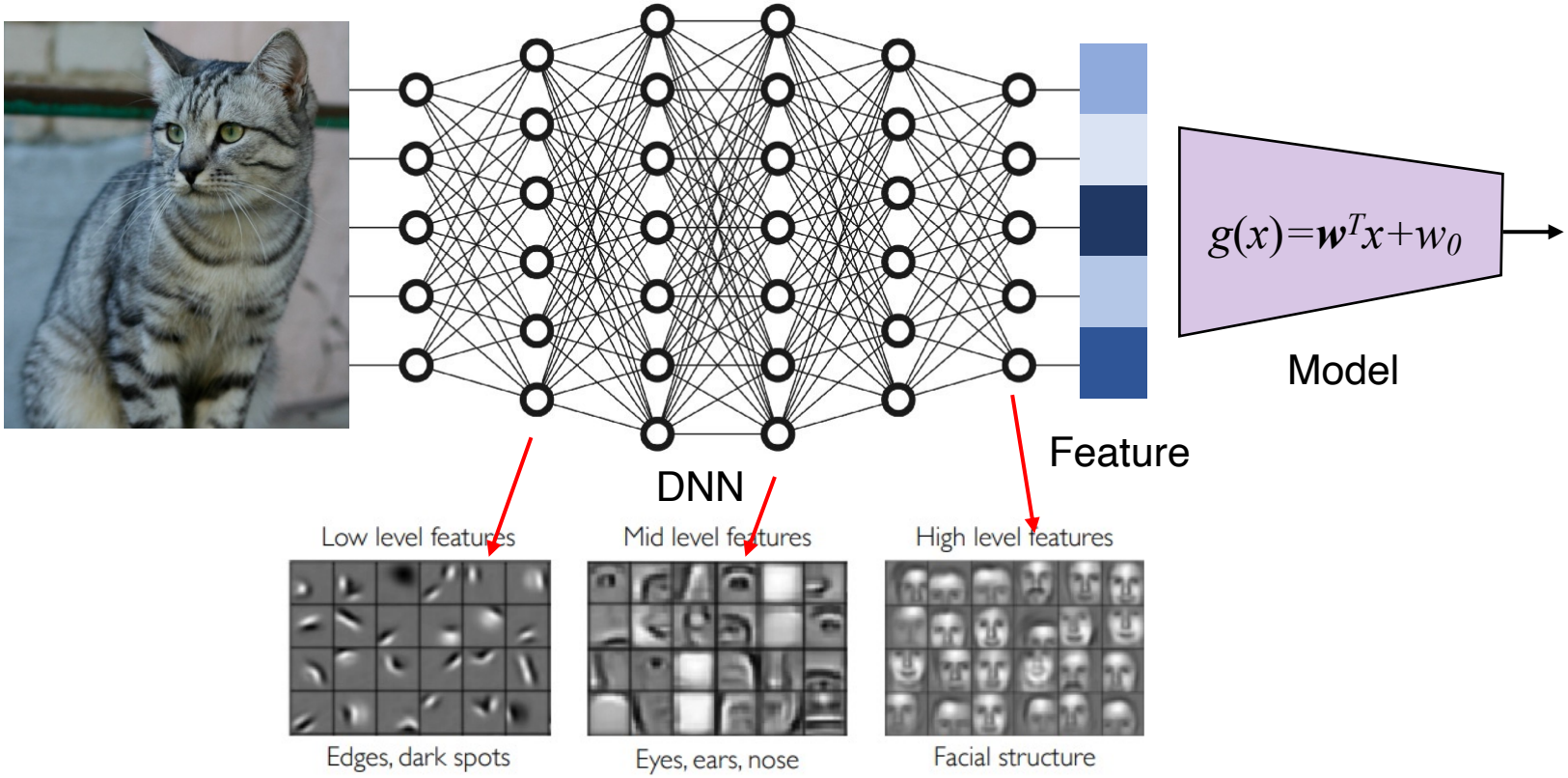
→ "cat"

Any Problems?

Learning Feature Representations

Idea #3: Deep Learning?

Can we automatically learn features from the data?



Learning Features Using Neural Networks

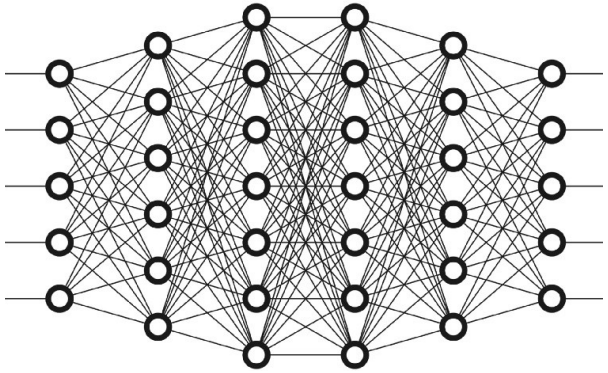
Sequence-of-Pixels

Input Image

matrix of pixel (32x32x3)

17
255
2
28
45
⋮
122

vector 3072x1



NN

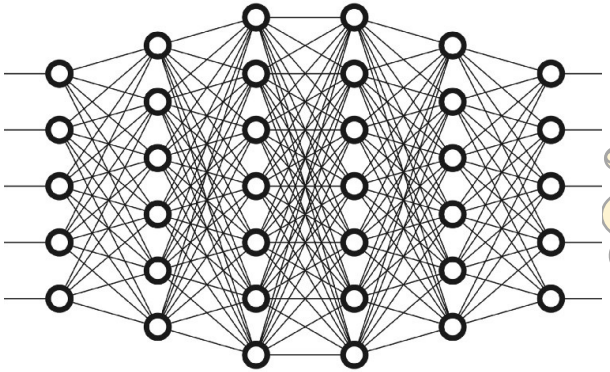
A Big Problem

All input pixels should be stretched into a 1-d vector

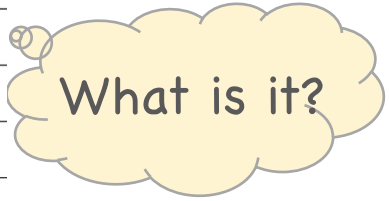
Learning Features Using Neural Networks

Other problems ?

Input Image



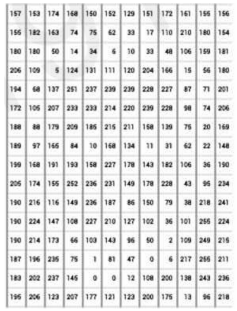
NN



- Always concern the entire image
- Sensitive to scale, position,…

Problems with Standard NNs

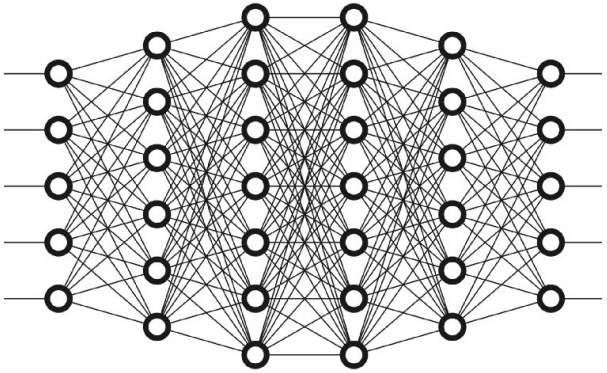
Input Image



matrix of pixel
(32x32x3)

17
255
2
28
45
⋮
122

vector
3072x1

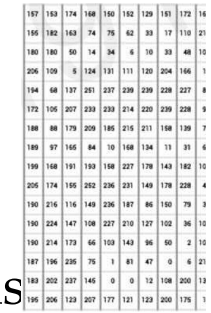


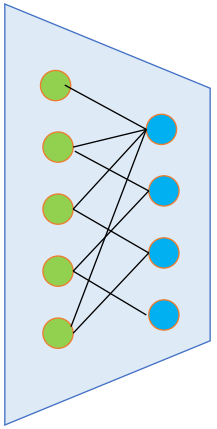
NN

- **Seq-of-Pixels:** The spatial structure of images is destroyed
- **Locality insensitive**: concern the entire picture instead of local features.
- **Translation variant:** sensitive to the global position of a feature
- **Too many parameters**: fully connected layers yield a huge volume of parameters.

Modeling Images: Design Criteria

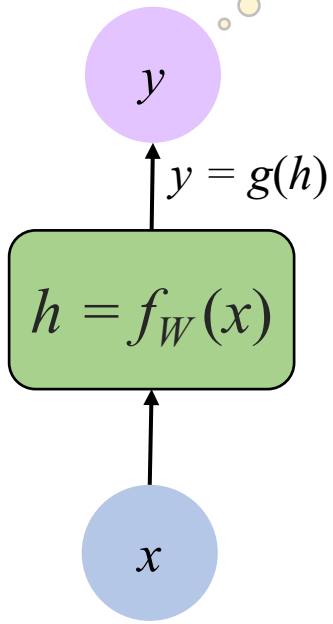
- To model images, we need to:
  1. Keep the spatial structure
  2. Sensitive to locality
  3. Handle variants in input images
  4. Share parameters across the spatial regions





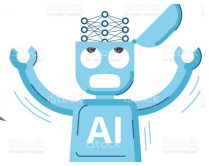
From MLP to CNN

I process all pixels at once without being aware of their spatial, scale…



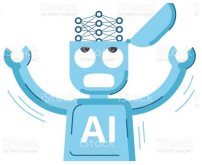
How can I "distinguish" the spatial of different objects?

Why not just focus on small, local regions and scan for interested objects?



But, how can I "focus"?

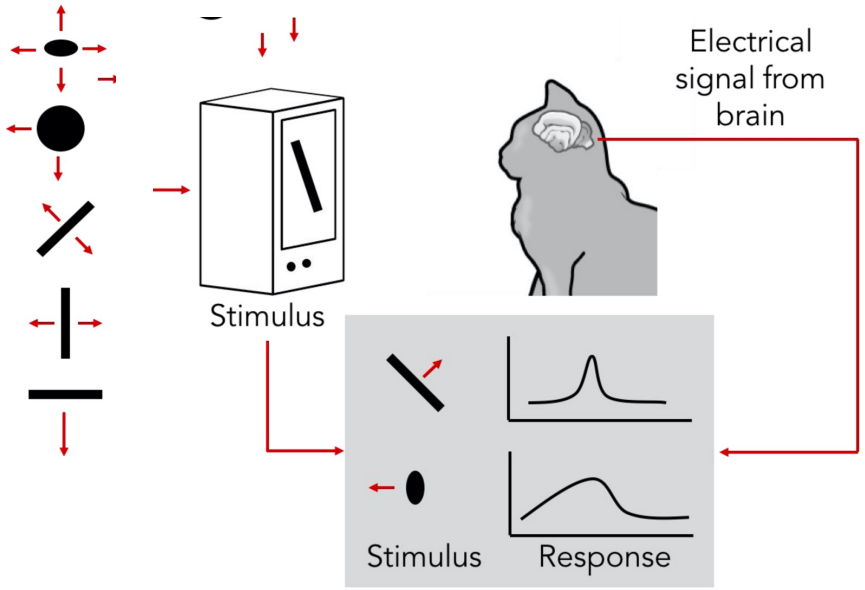
Let's get some inspirations from human brains ..



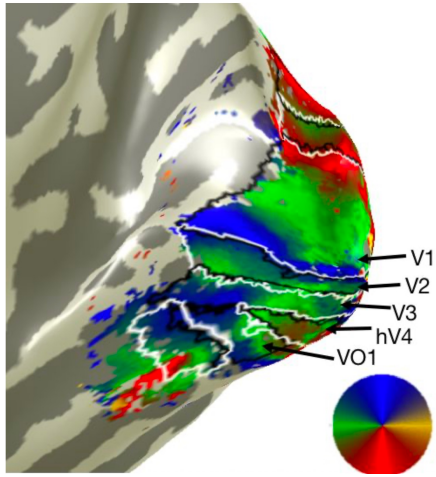
Inspired by Biology

- Each neuron in the cat's striate cortex (大脑皮层) has a **receptive field** of a specific shape.

[Hubel & Wiesel, 1959]

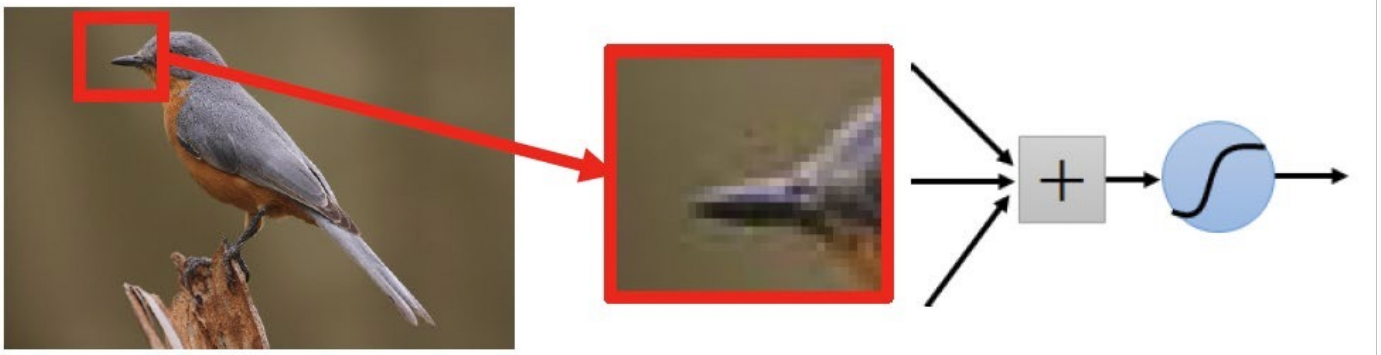


And nearby cells in cortex represent nearby regions in the visual field

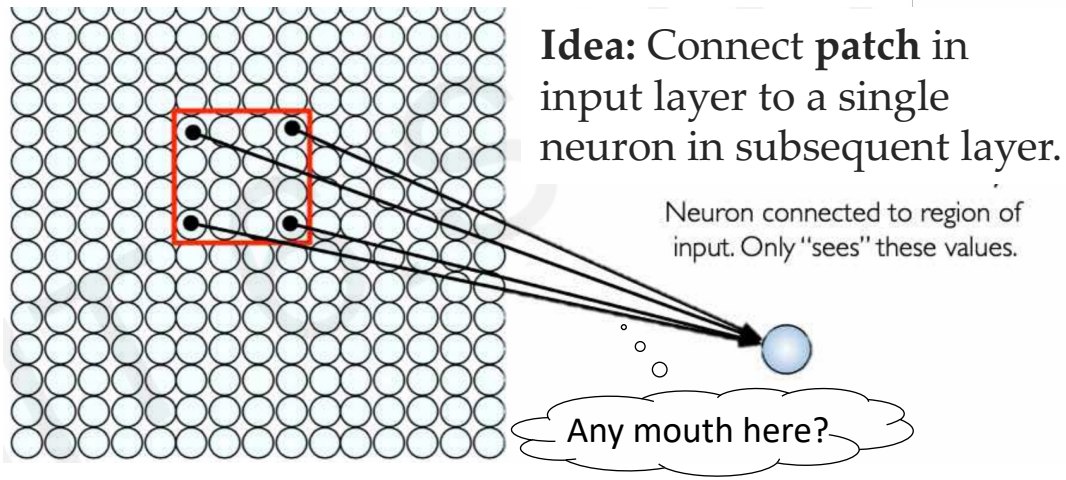


Convolution: The Idea

- This inspires us to **scan** over the image for a specific feature through weighing and activating.

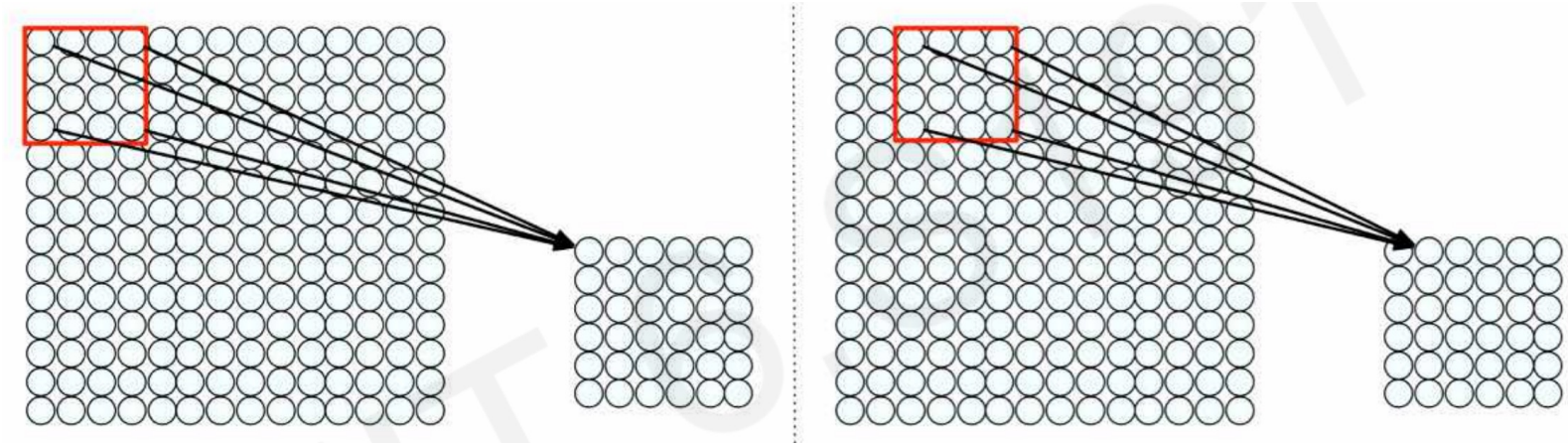


Input: 2D image.
Array of pixel values



Idea: Connect **patch** in input layer to a single neuron in subsequent layer.

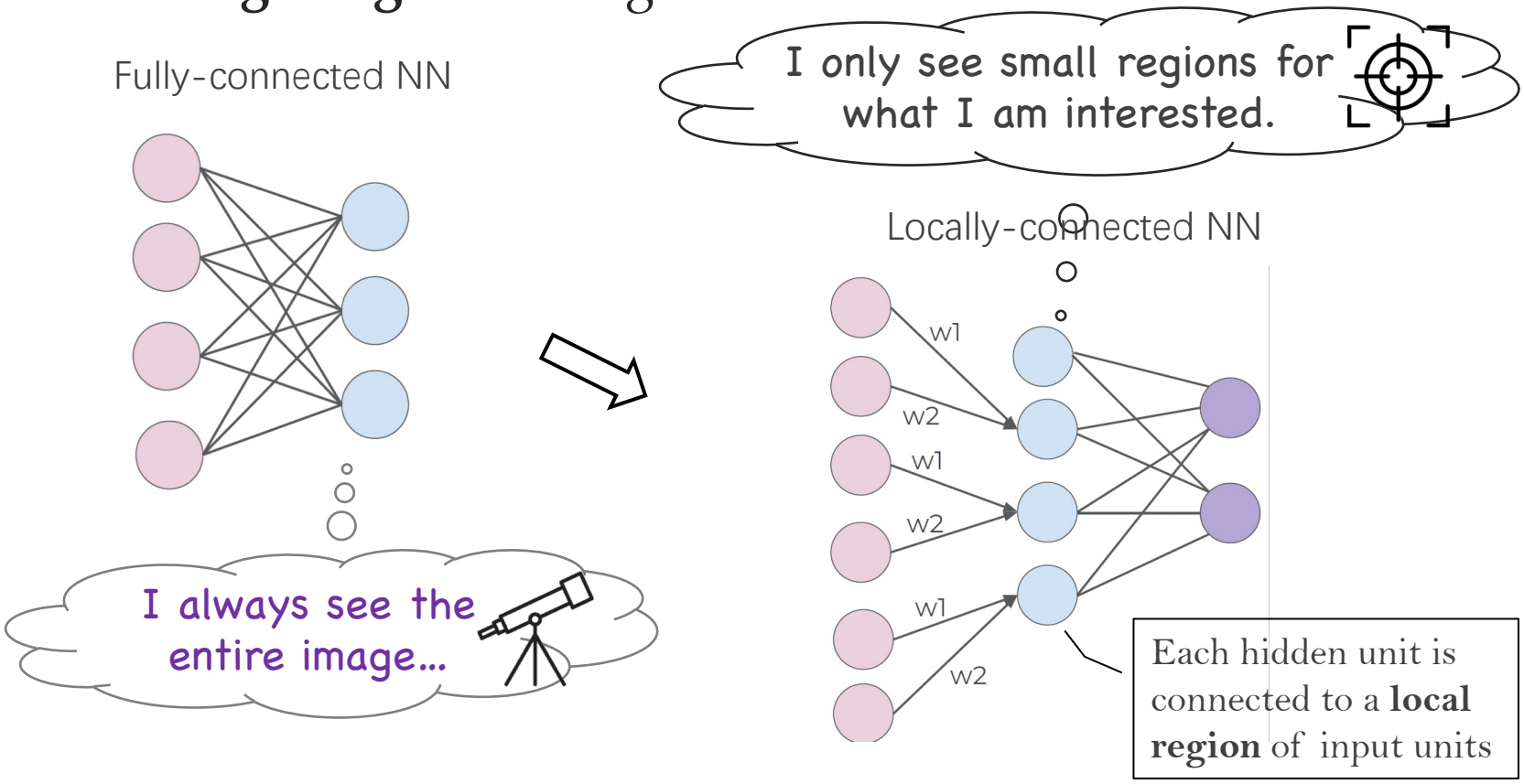
Convolution: The Idea



- **Slide** over the image spatially, computing feature transformation (how?)
- During sliding, the weights for all neuron must be **shared** (why?)

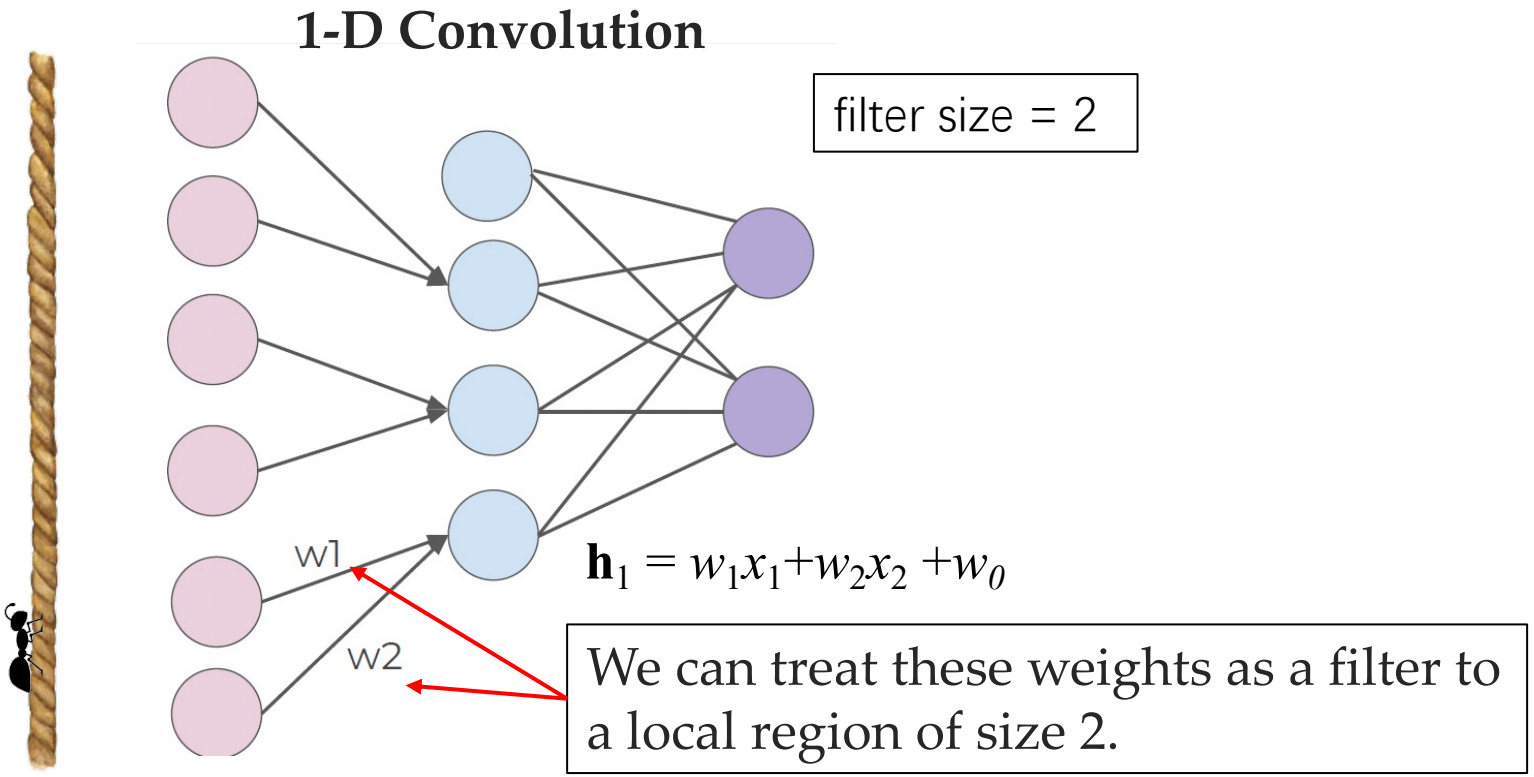
MLP vs CNN

- Restrict the scope of hidden units to focus on **local** features.
- **Sharing weights** among hidden units



Convolutions and Filters (1-D)

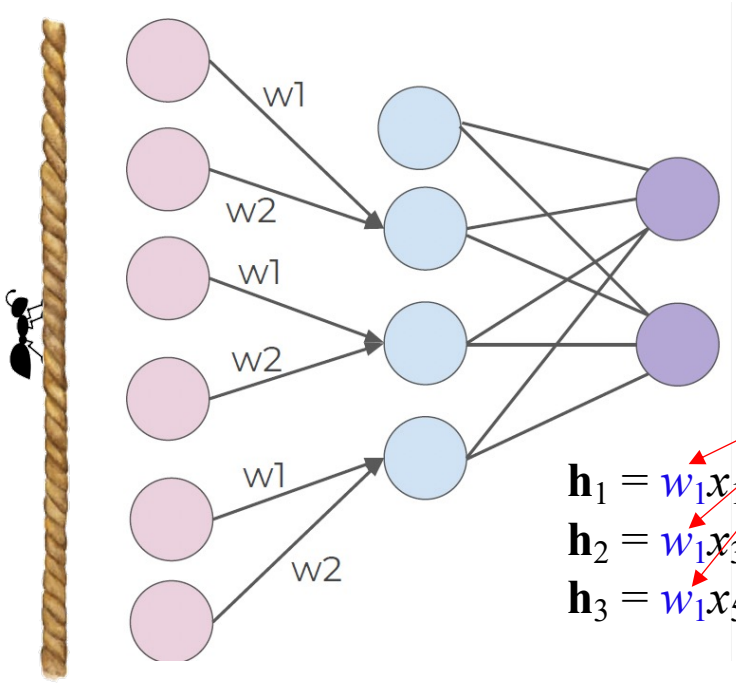
- **Filter** (a.k.a., **convolutional kernel**): weights assigned to the input region by a hidden unit.



Convolutions and Filters (1-D)

- We now apply the **same** set of weights over the **whole picture**.

1-D Convolution

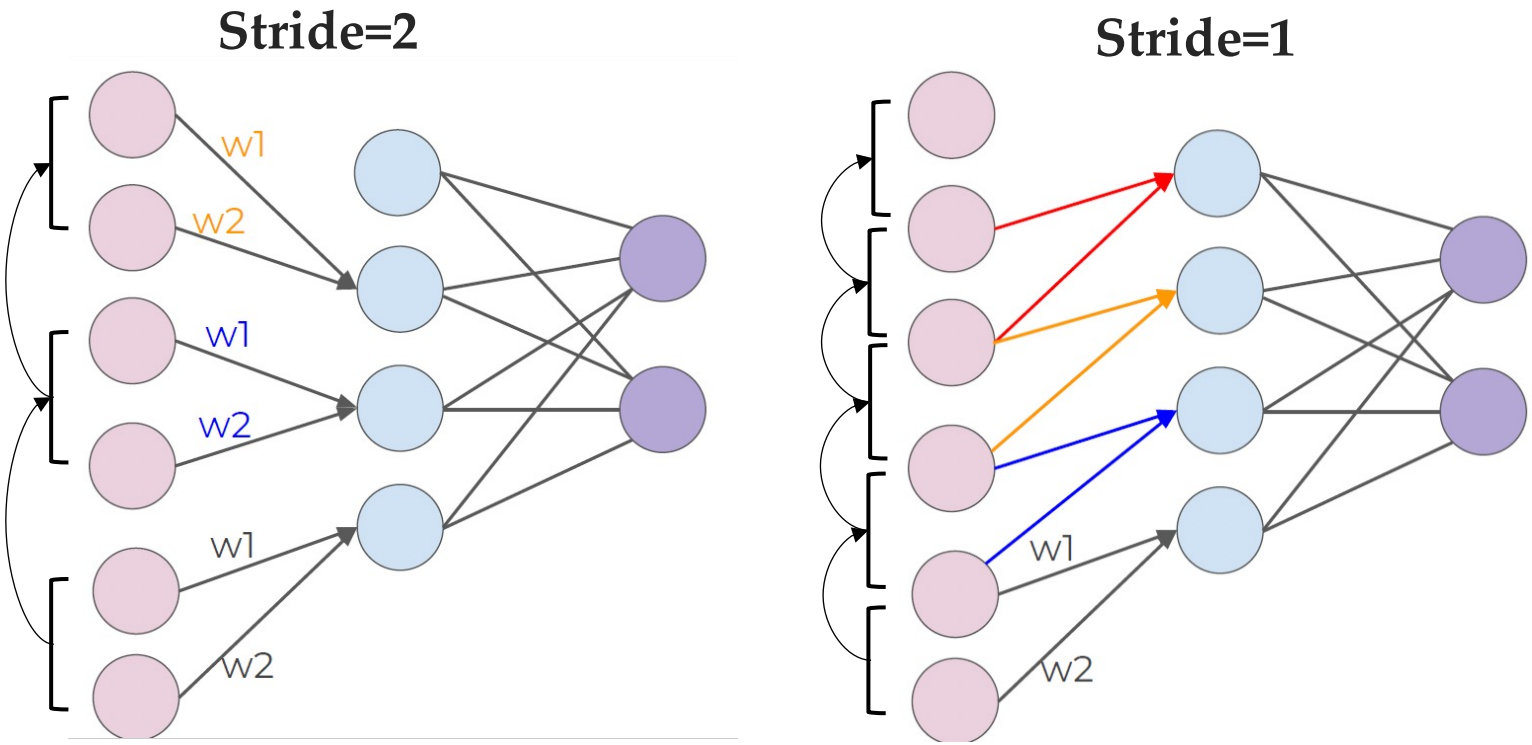


Parameters are **shared** over the whole picture for one filter.

$$\mathbf{h}_1 = w_1 x_1 + w_2 x_2 + w_0$$
$$\mathbf{h}_2 = w_1 x_3 + w_2 x_4 + w_0$$
$$\mathbf{h}_3 = w_1 x_5 + w_2 x_6 + w_0$$

Convolutions and Filters

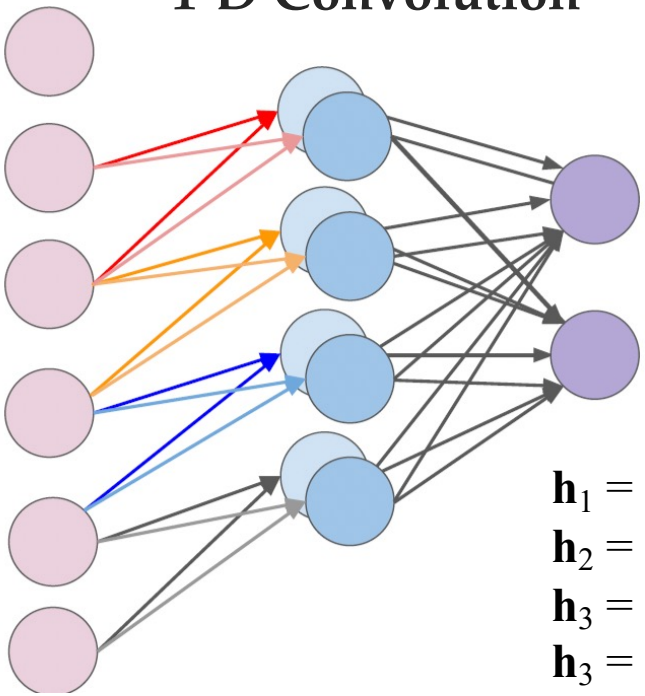
- Note that the filter scans every 2 pixels in this example.
- We can control the filter to scan every N pixels (N=**stride**).



Convolution and Filters

- We can then expand this idea to **multiple filters**.
- Each filter is detecting a different feature

1-D Convolution

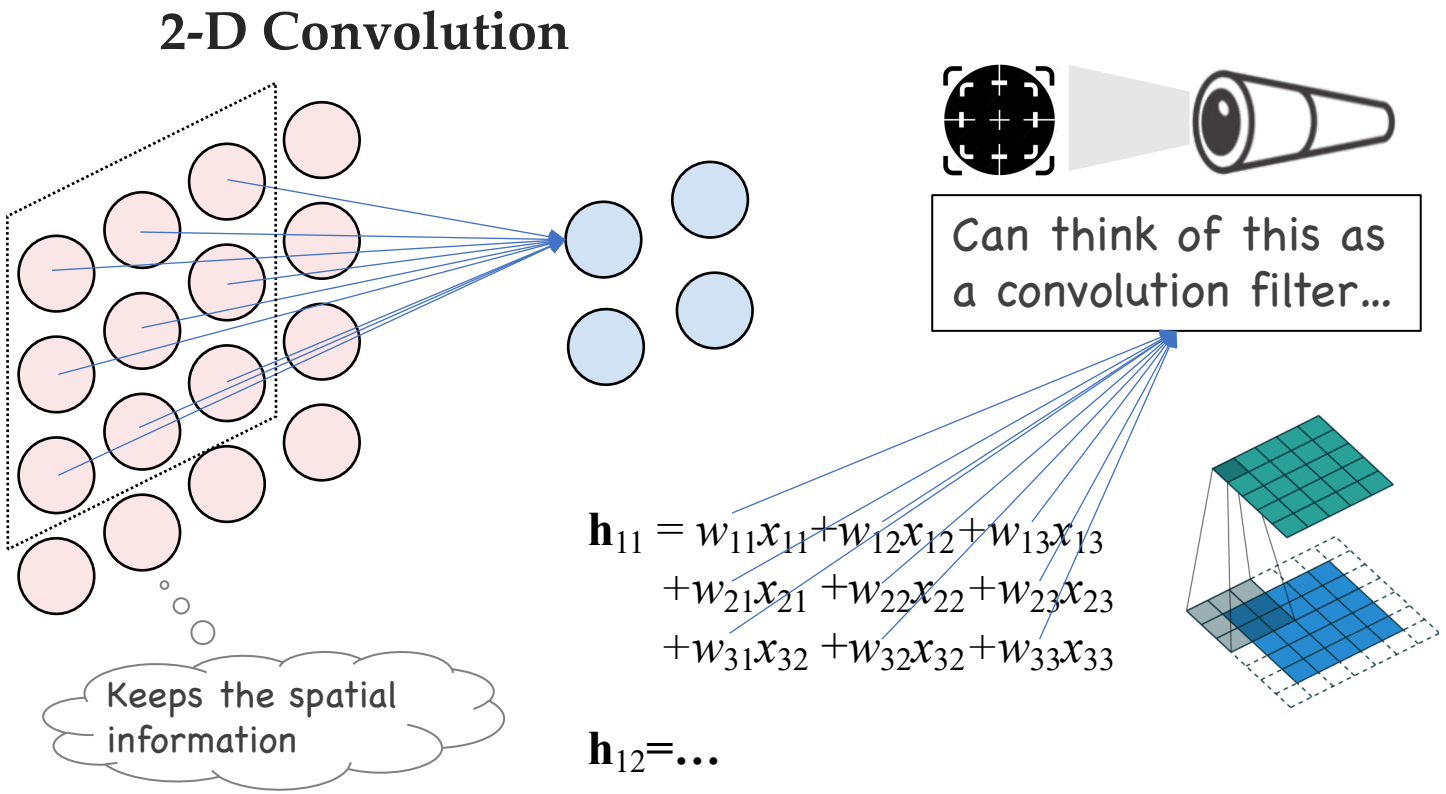


 # filters = 2
filter size =2
stride = 1

$$\mathbf{h}_1 = w_1x_1+w_2x_2+w_0 \qquad \mathbf{h}_1 = w'_1x_1+w'_2x_2+w'_0$$
$$\mathbf{h}_2 = w_1x_2+w_2x_3+w_0 \qquad \mathbf{h}_2 = w'_1x_2+w'_2x_3+w'_0$$
$$\mathbf{h}_3 = w_1x_3+w_2x_4+w_0 \qquad \mathbf{h}_3 = w'_1x_3+w'_2x_4+w'_0$$
$$\mathbf{h}_3 = w_1x_4+w_2x_5+w_0 \qquad \mathbf{h}_3 = w'_1x_4+w'_2x_5+w'_0$$

2-D Convolutions

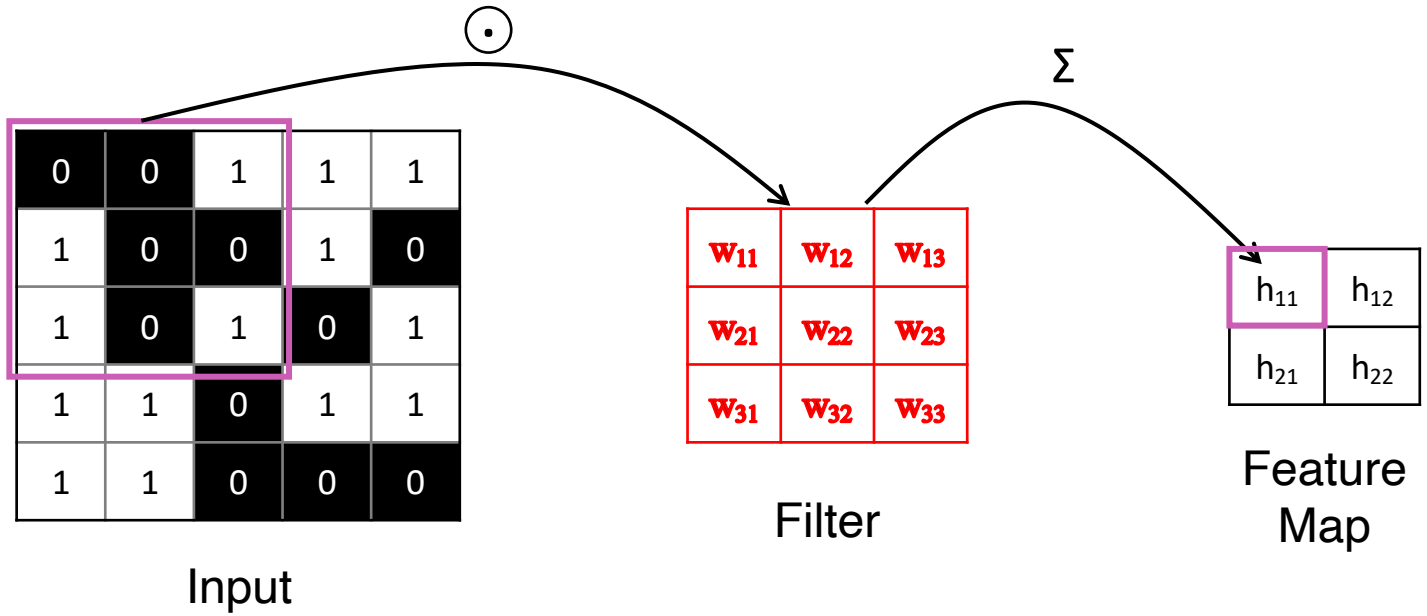
- Now let's expand these concepts to **2-D Convolution**, since we'll mainly be dealing with images.



2-D Convolutions

- **Convolutional Operation**

elementwise multiplication and sum of a filter and the image

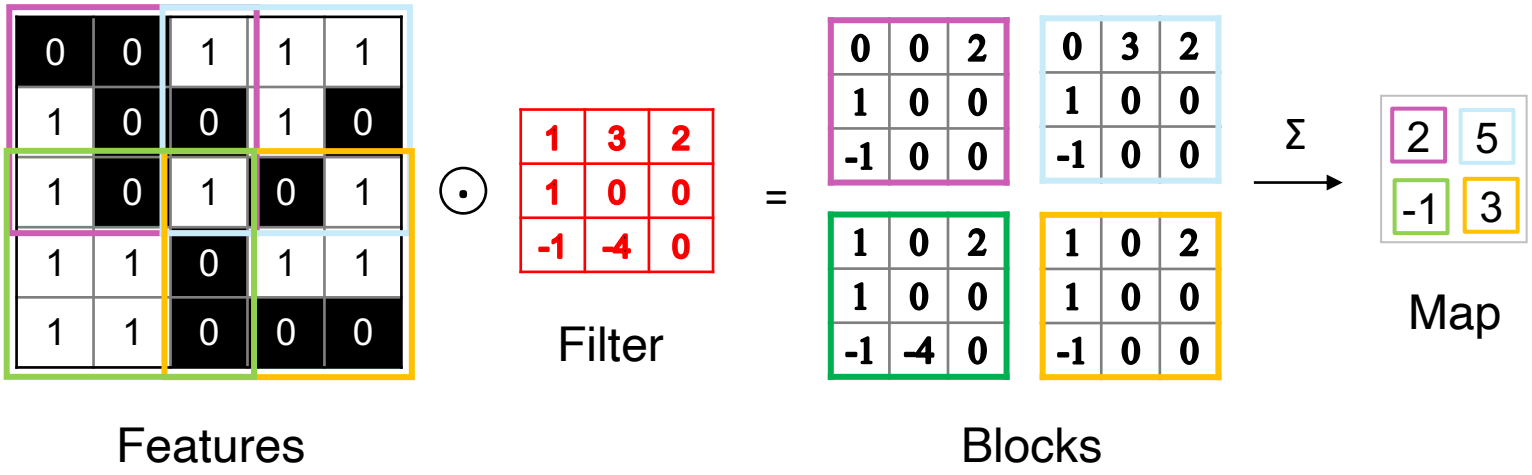


* We call it convolutional because it is related to convolution of two signals:

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

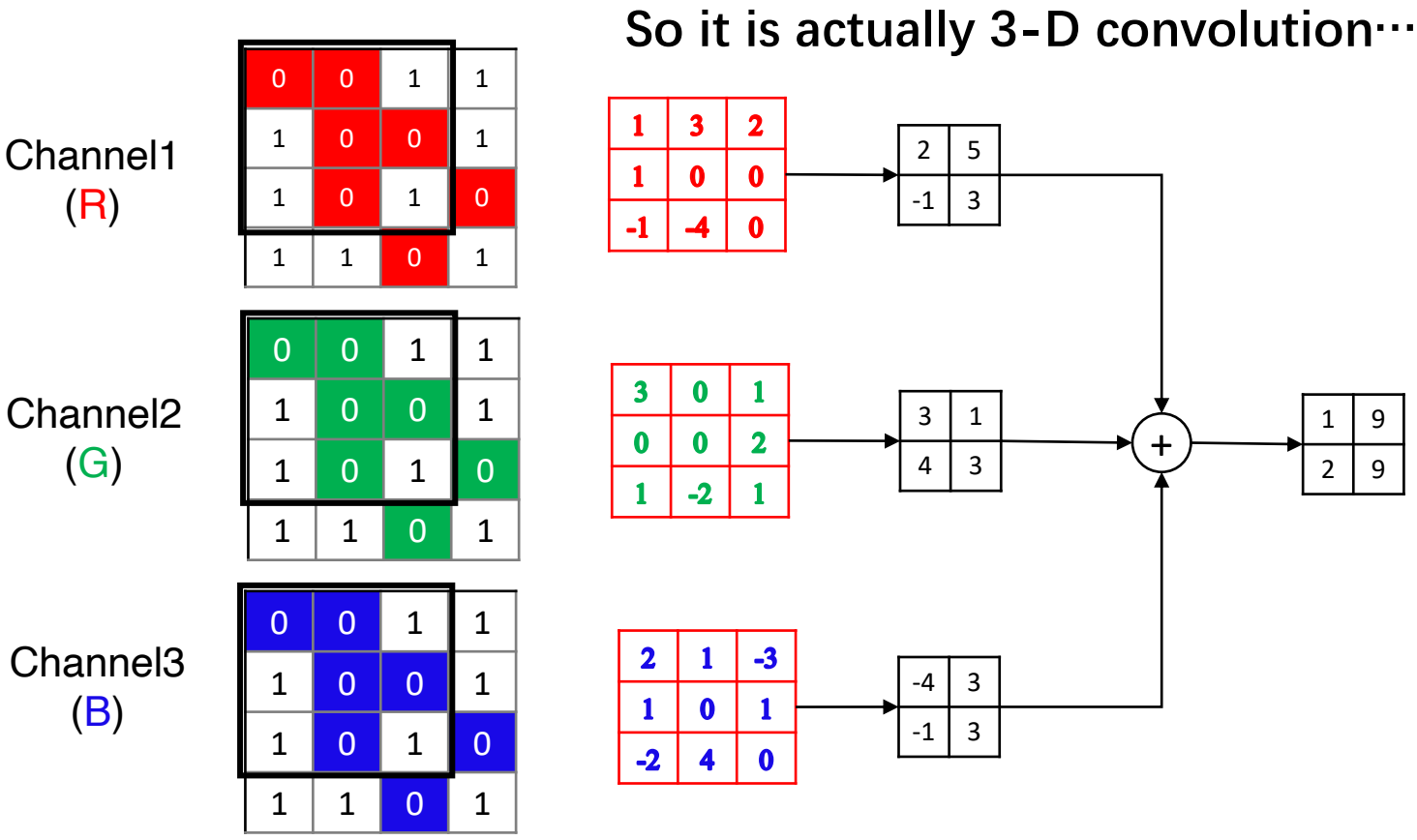
2-D Convolutions

- **Example:**



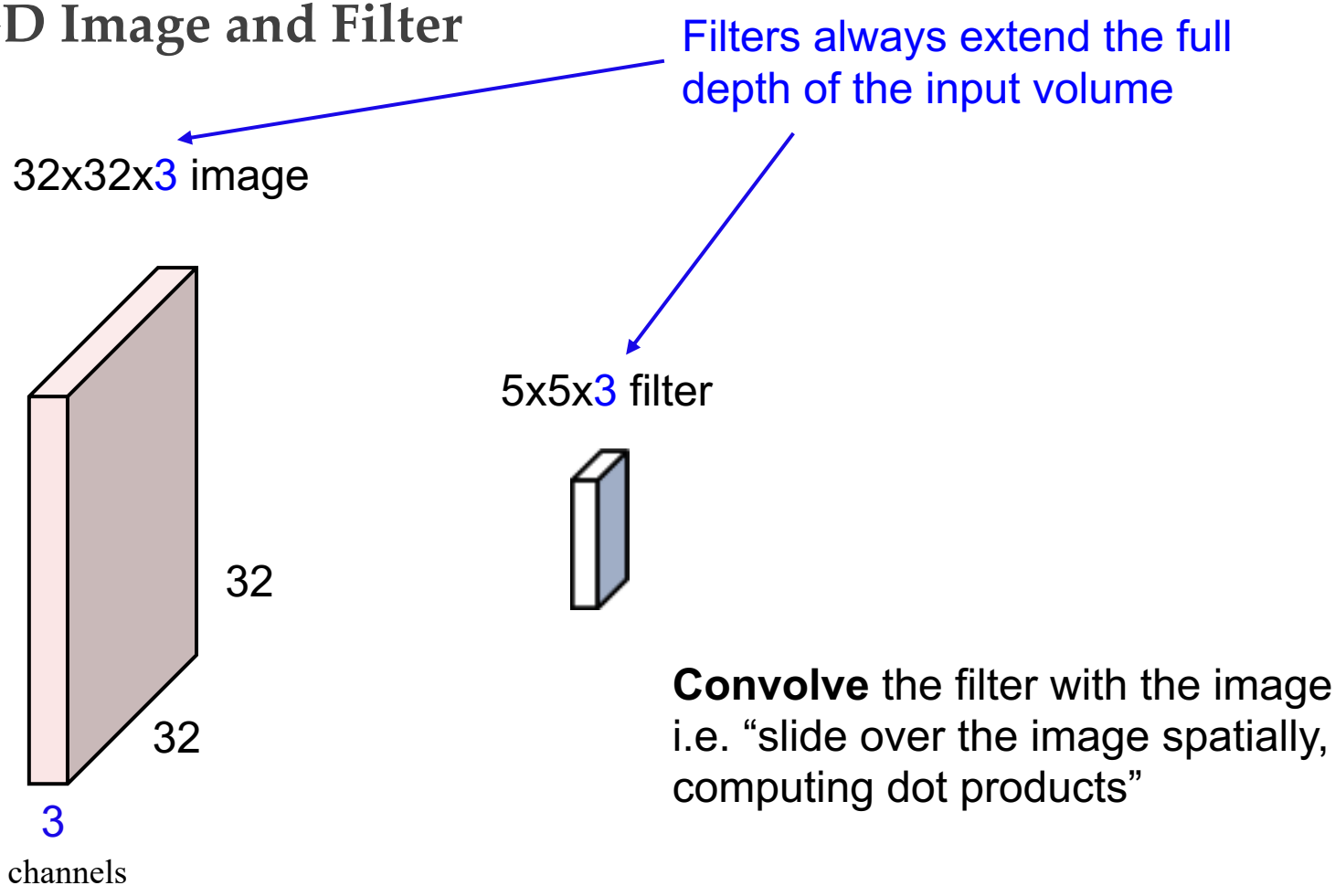
Multiple Input Channels

- What if we have color?



"3-D" Convolution

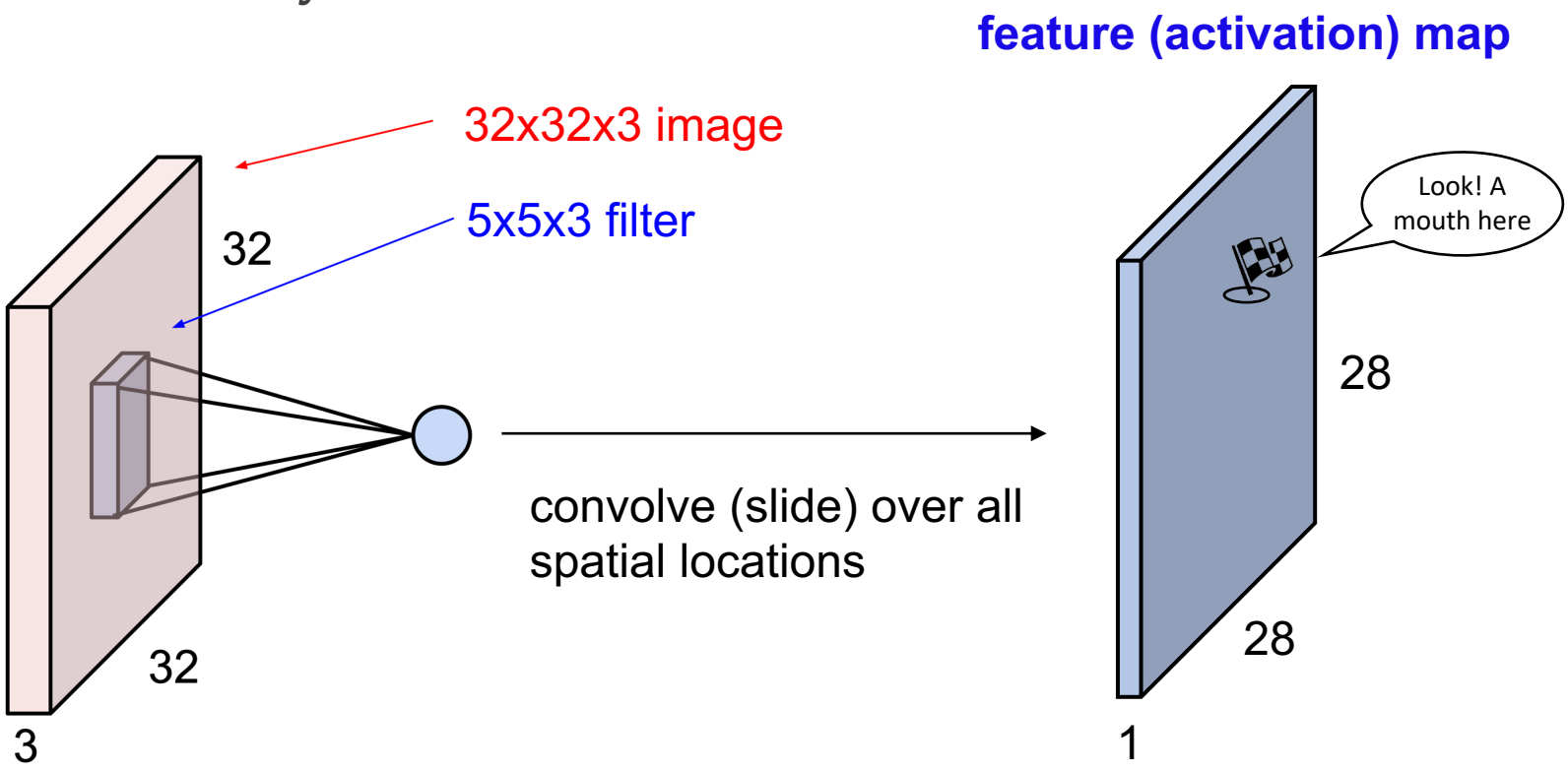
- 3-D Image and Filter



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

Feature Map

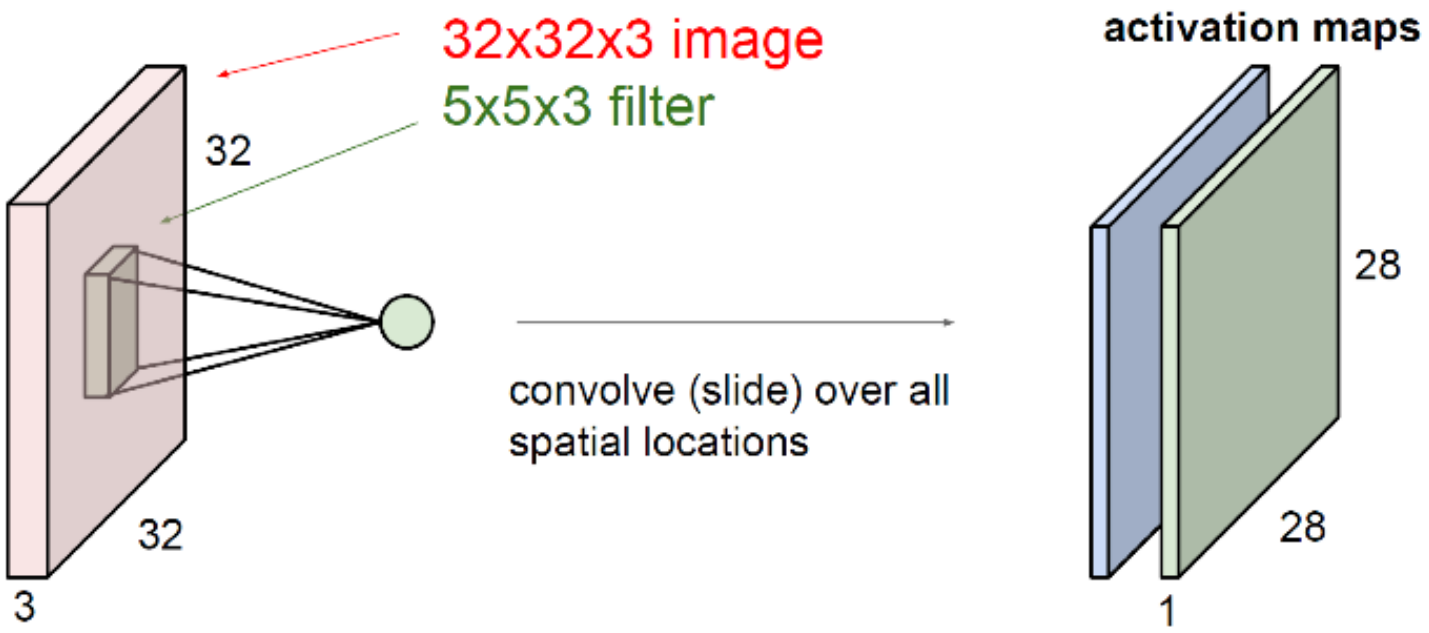
- A map that stores the locations of a specific feature activated by a filter.



Multiple Filters

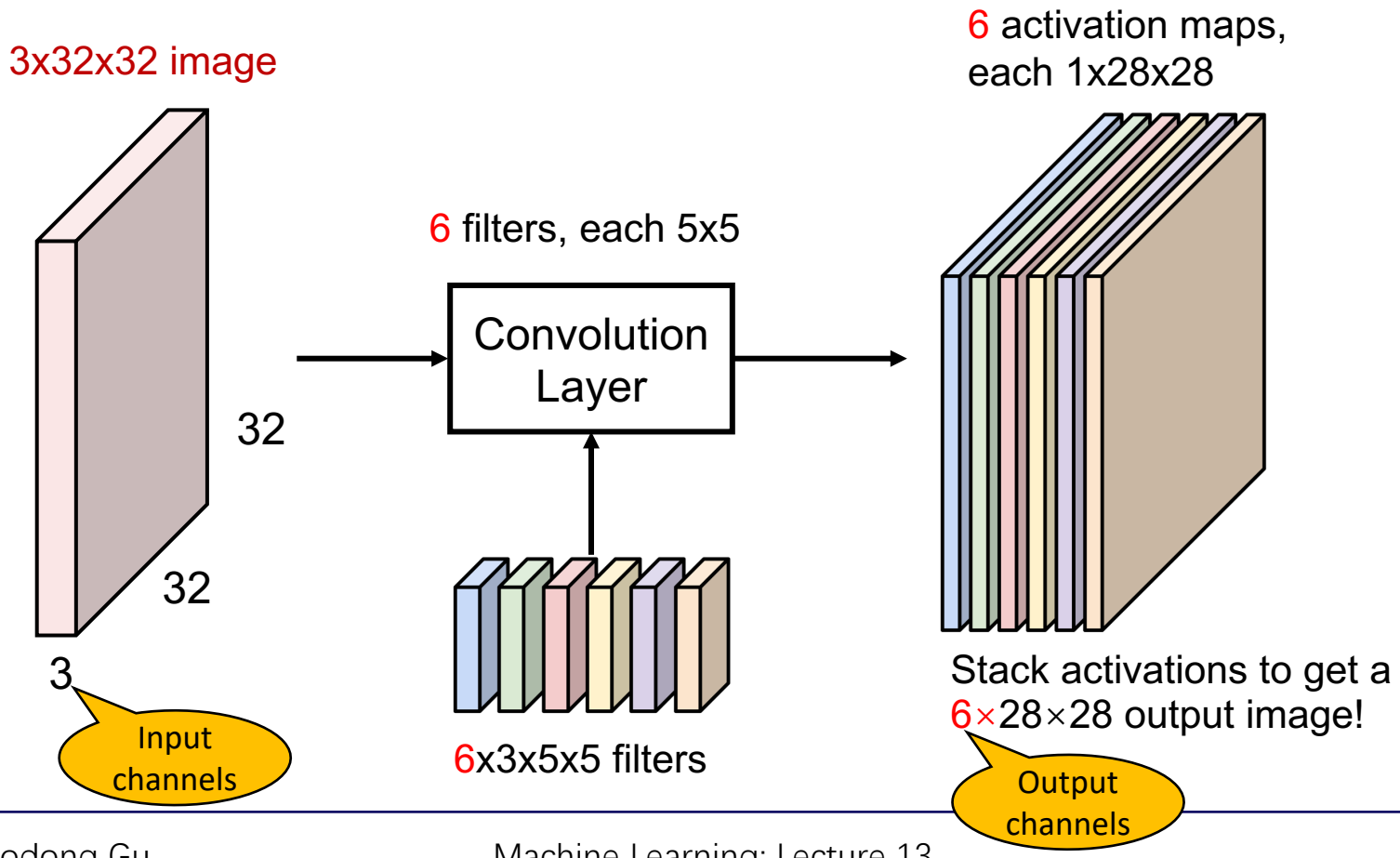
- **Using multiple filters to scan multiple features.**

consider a second filter



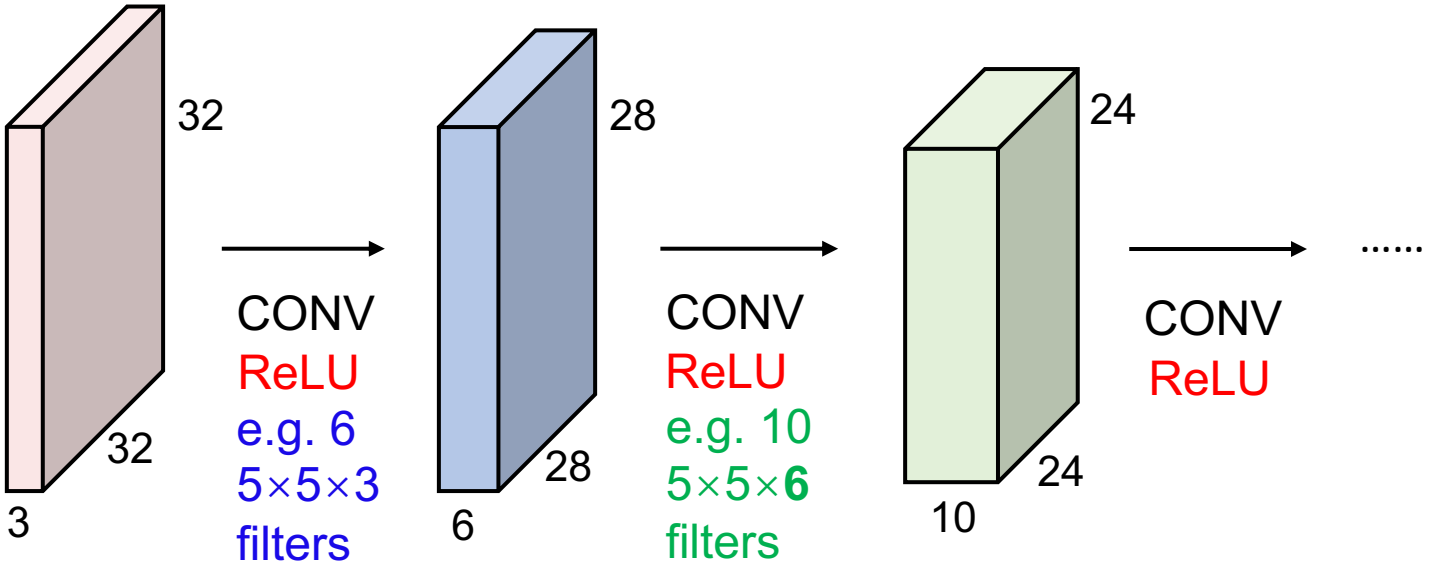
Multiple Feature Maps

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



Multiple Layers

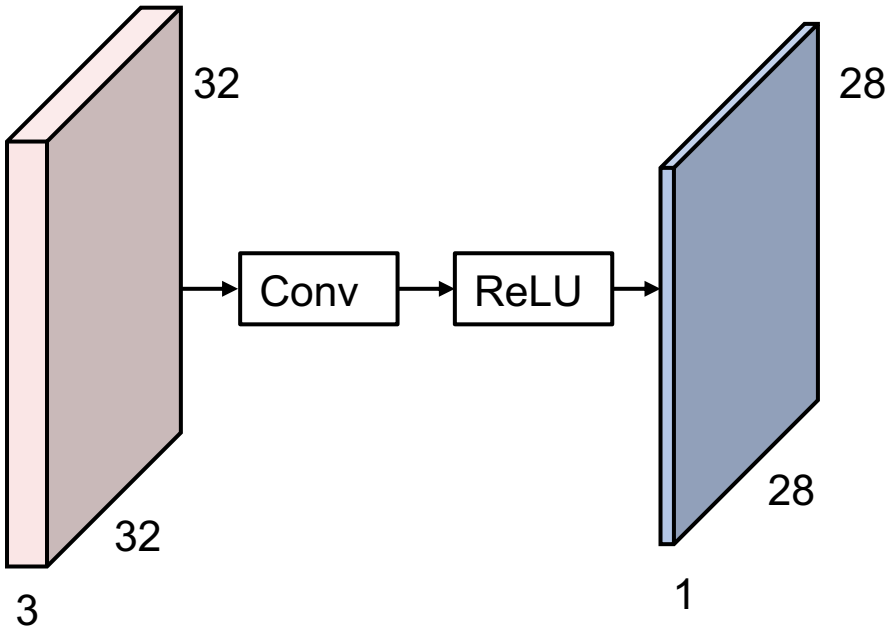
- A sequence of convolutional layers, interspersed with activation functions (e.g., ReLU).

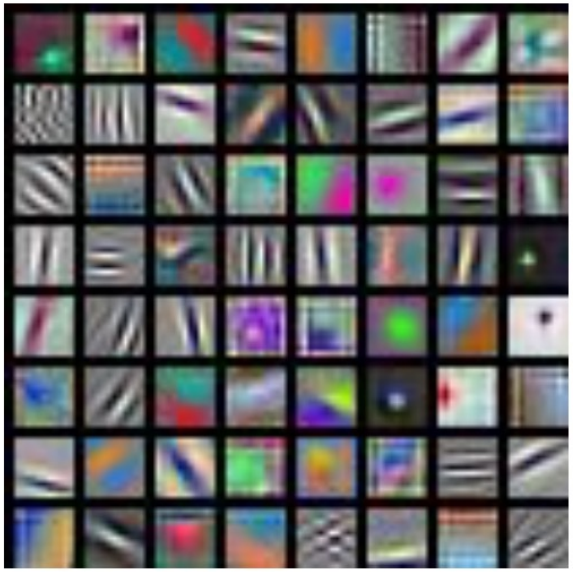


Multiple Layers

- What do convolutional filters learn?

First-layer conv filters: local image templates
(Often learns oriented edges, opposing colors)



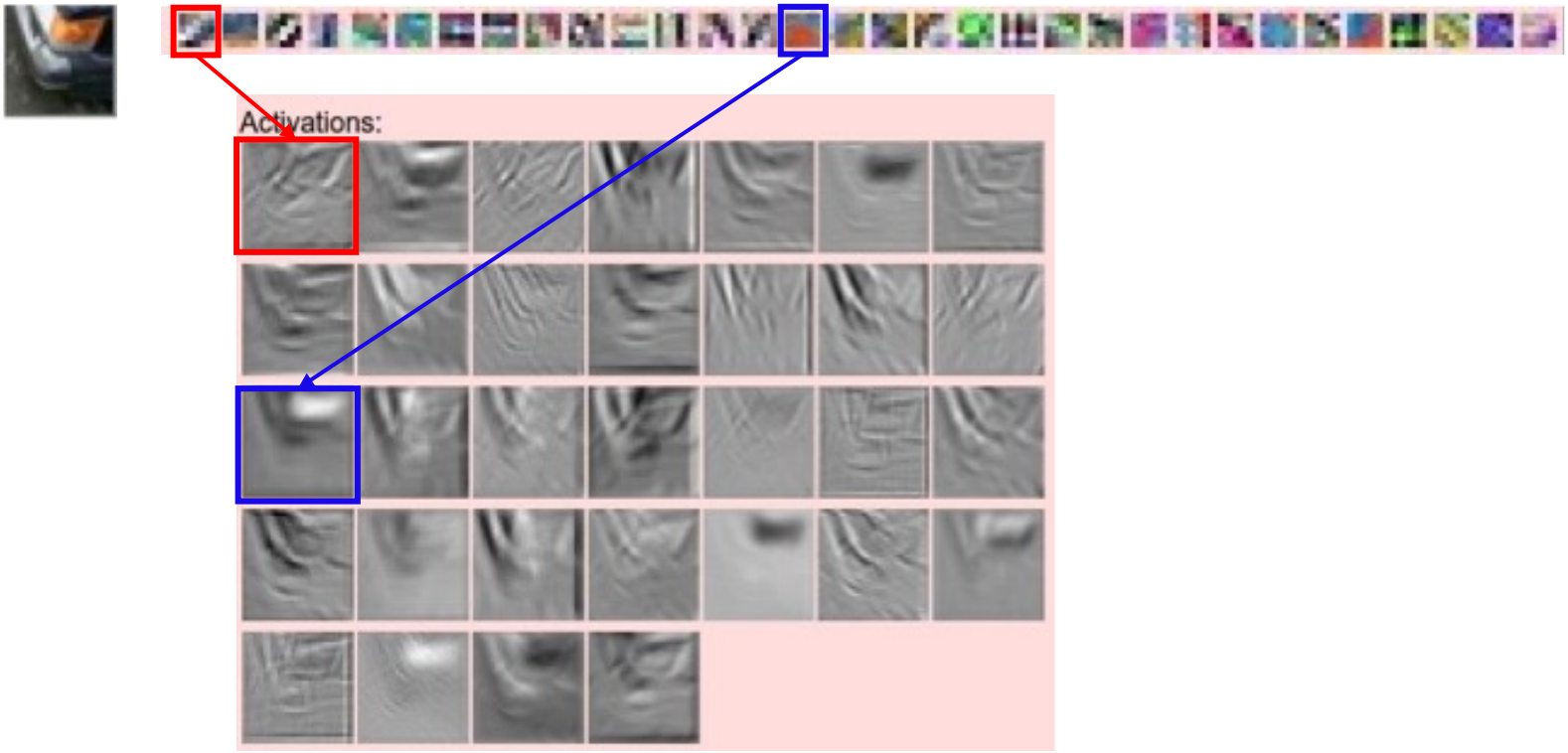


AlexNet: 64 filters, each 3x11x11

Multiple Layers

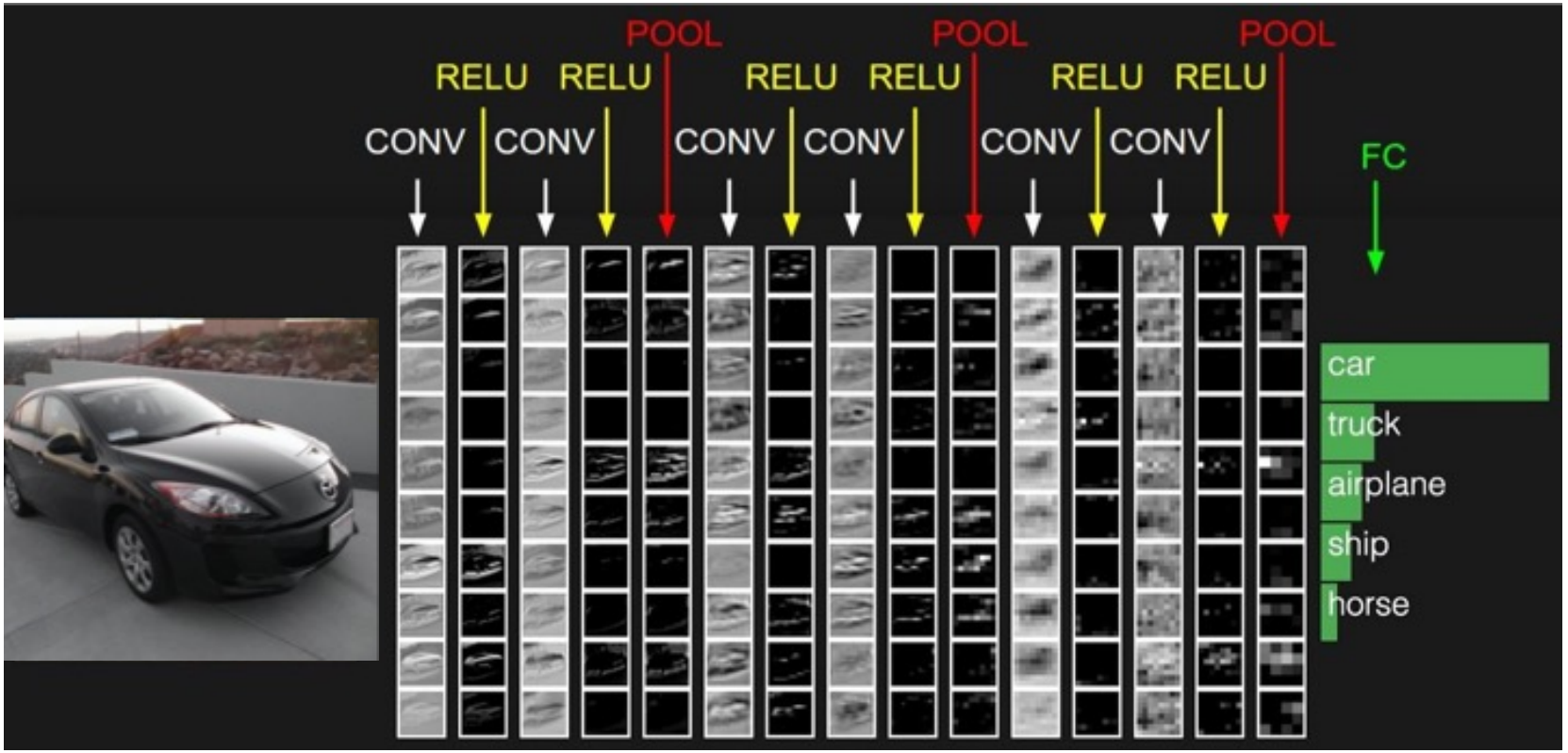
- What does each convolutional filter learn?

Example: 32 5x5 filters one filter ⇒ one feature map



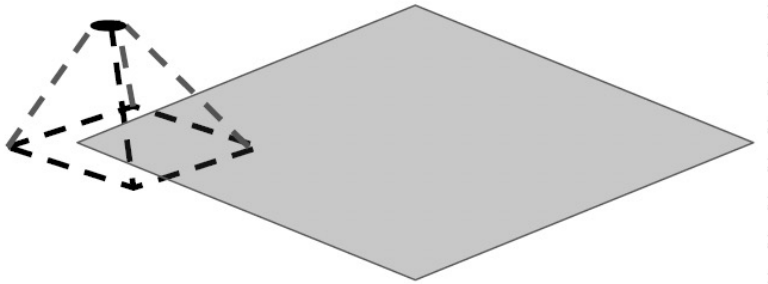
Multiple Layers

- What does each layer learn?

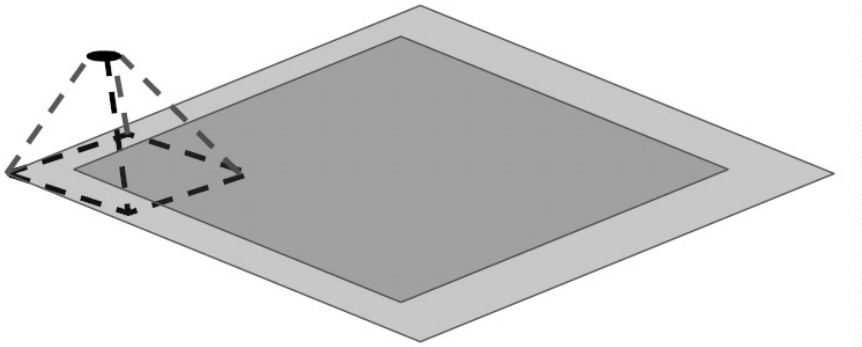


Padding

- We run into a possible issue for edge neurons! There may not be an input there for them.

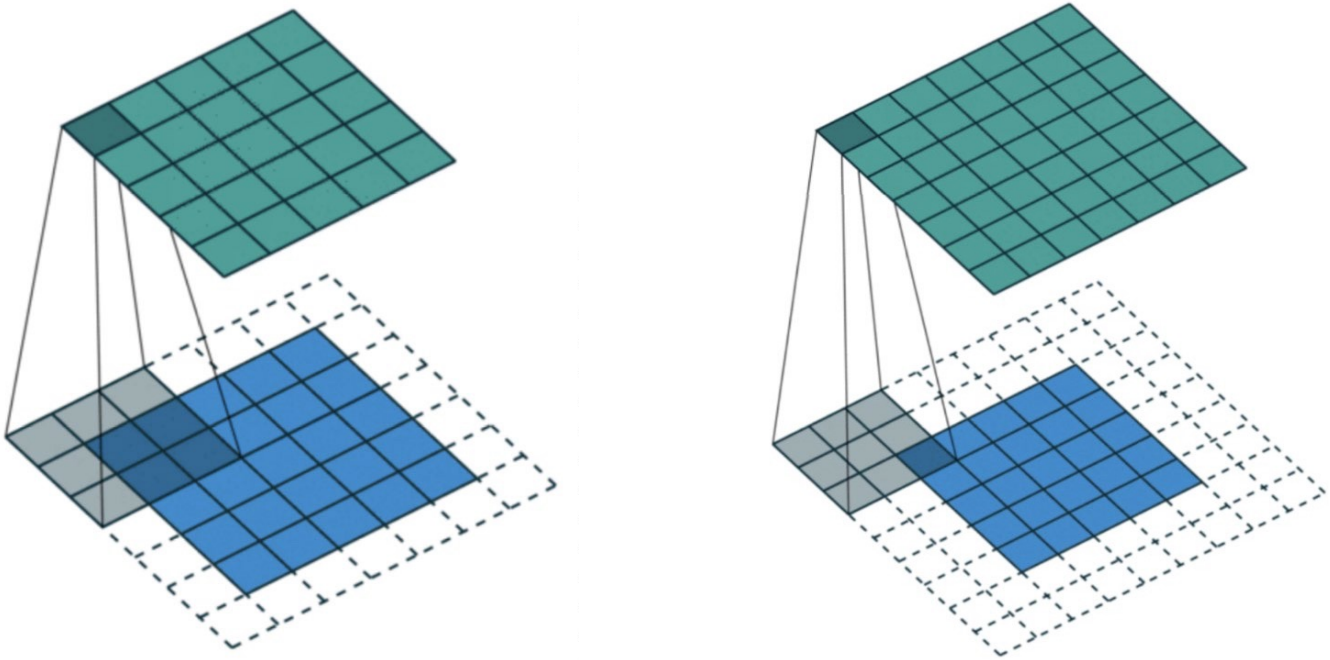


- We can fix this by adding a "**padding**" of zeros around the image.



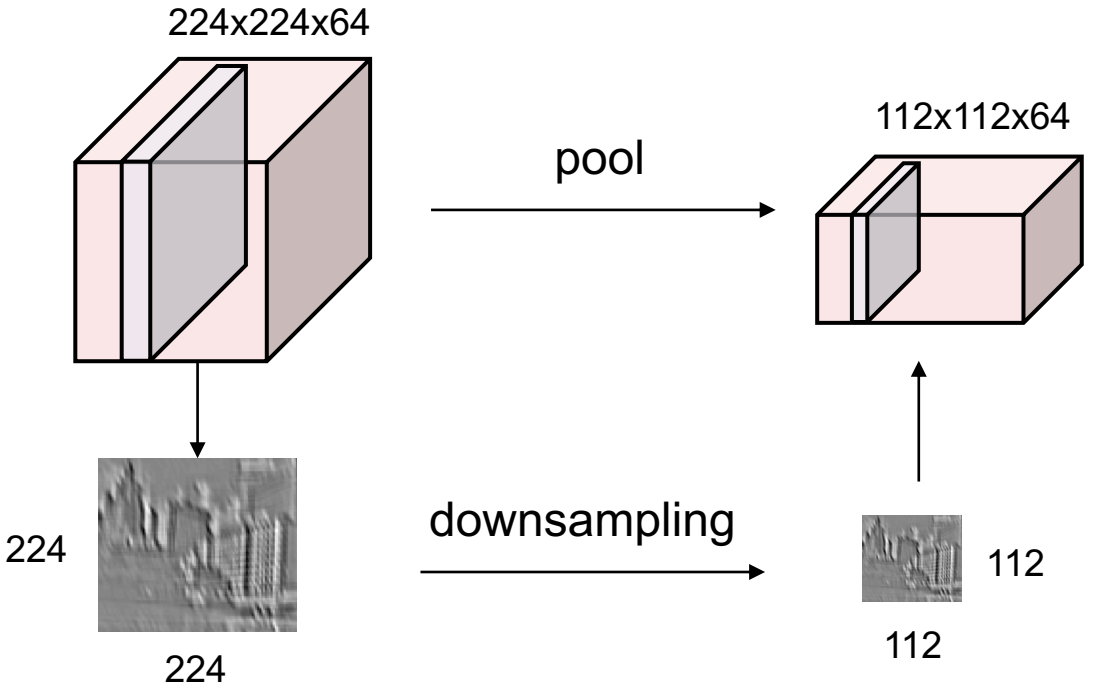
Padding

- **(Zero-)Padding**: symmetrically adding zeroes to the input matrix.
- allows the size of the input to be adjusted to our requirement.



Pooling

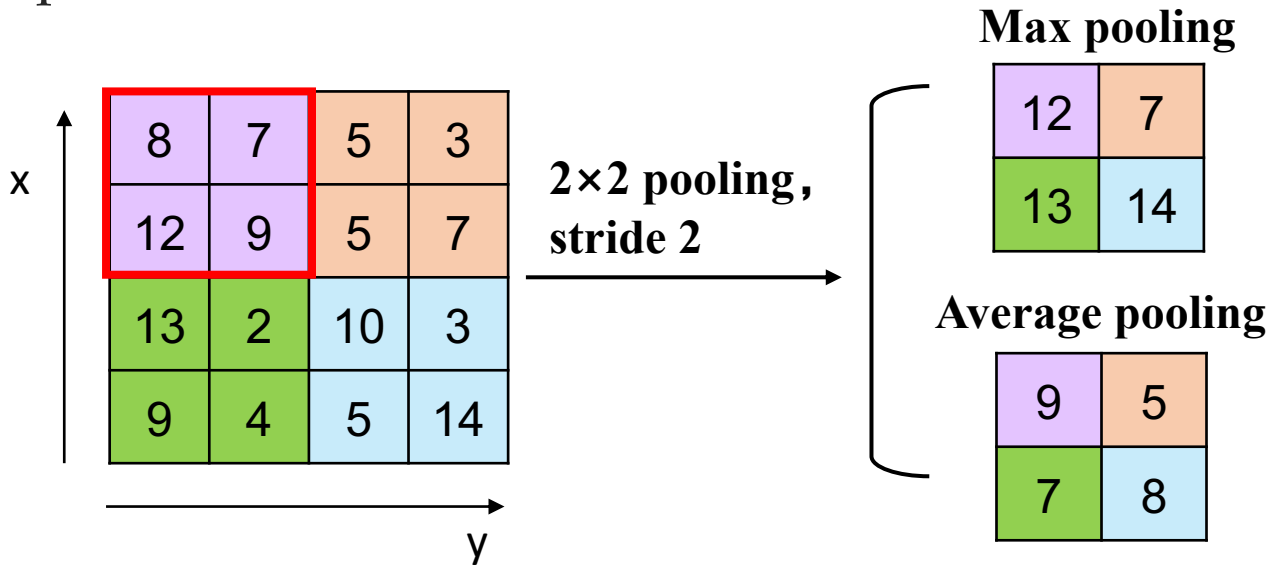
- **Downsample** the feature map to reduces the memory use.
- Makes the representations smaller and more manageable



Pooling does not change the object

Pooling

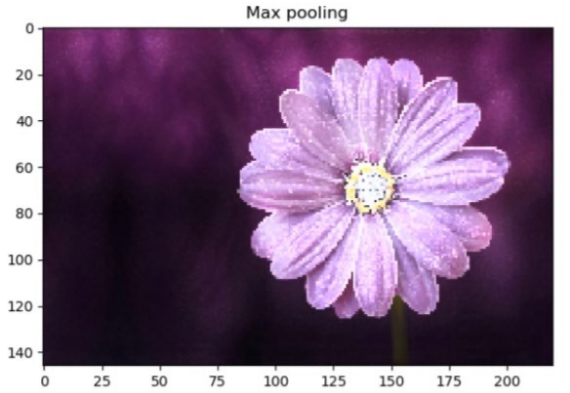
- **Max pooling**: The maximum pixel value of the patch is selected.
- **Average pooling**: The average value of all the pixels in the patch is selected.



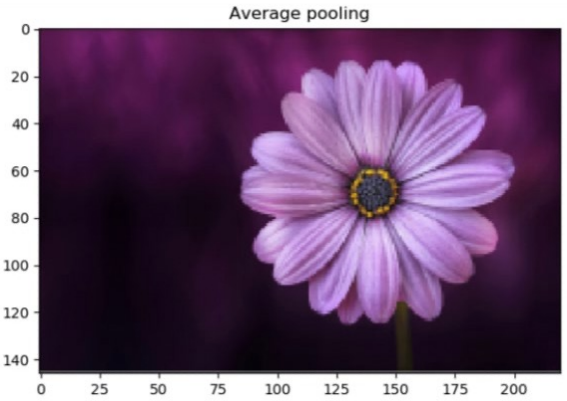
No learnable parameters
Introduces spatial invariance

Pooling

- In the following example, a filter of 9x9 is chosen.

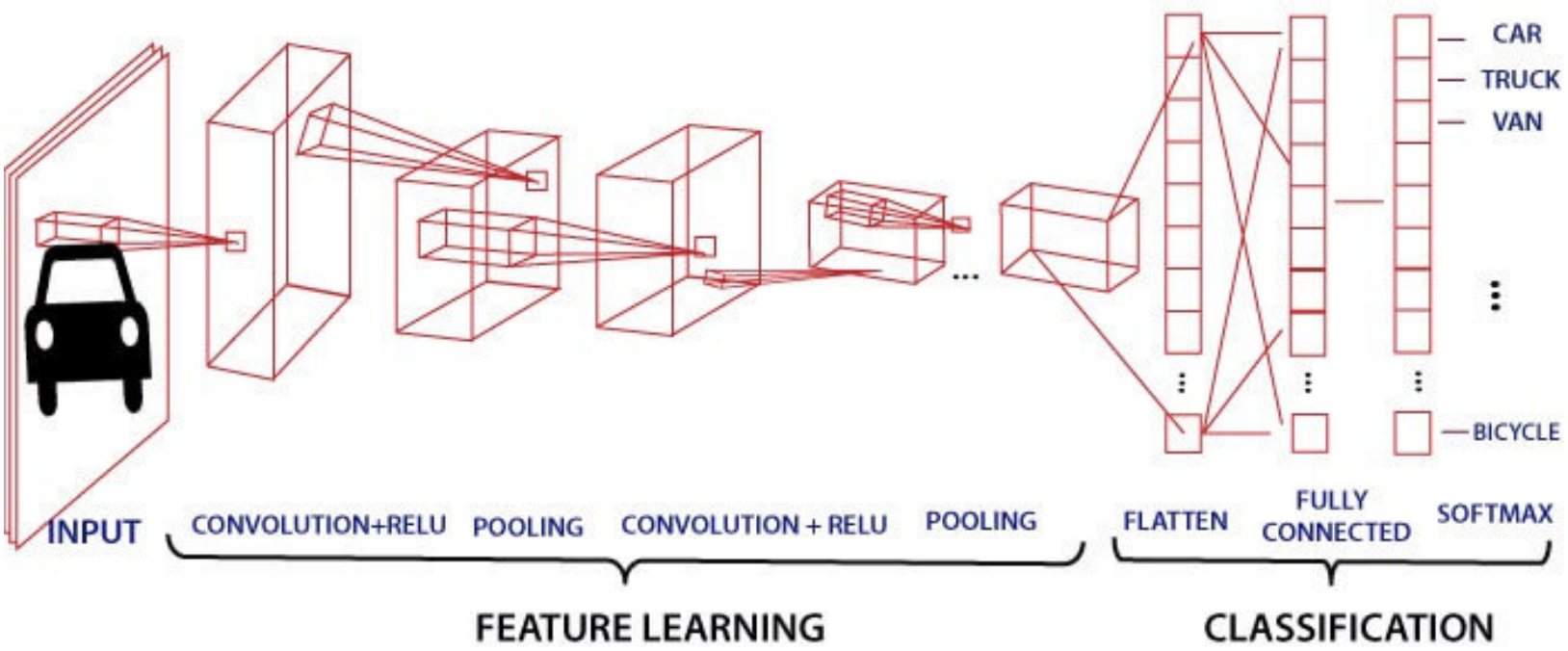


Max pooling selects the **brighter** pixels from the image.

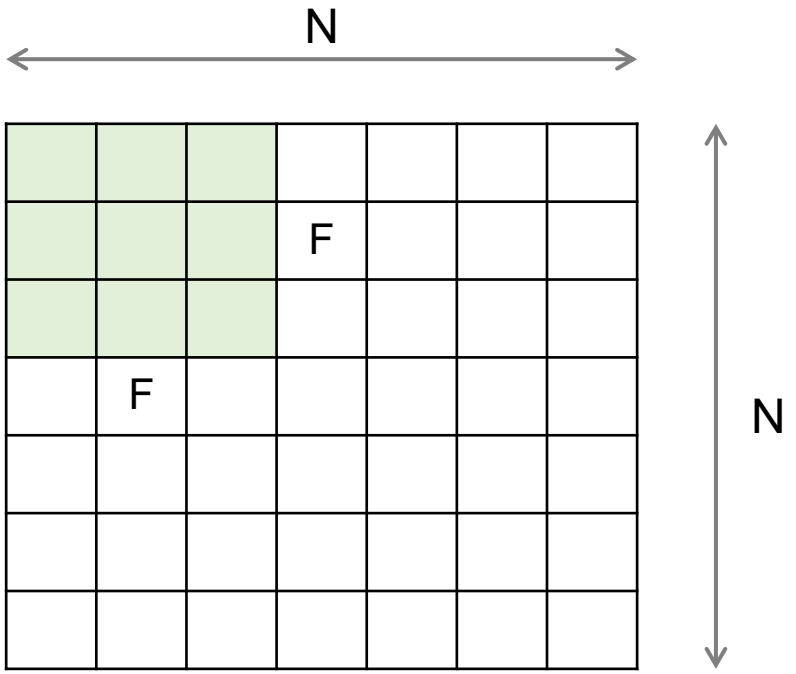


Average pooling method **smooths** out the image and hence the sharp features may not be identified.

The Big Picture



A Closer Look at Spatial Dimensions



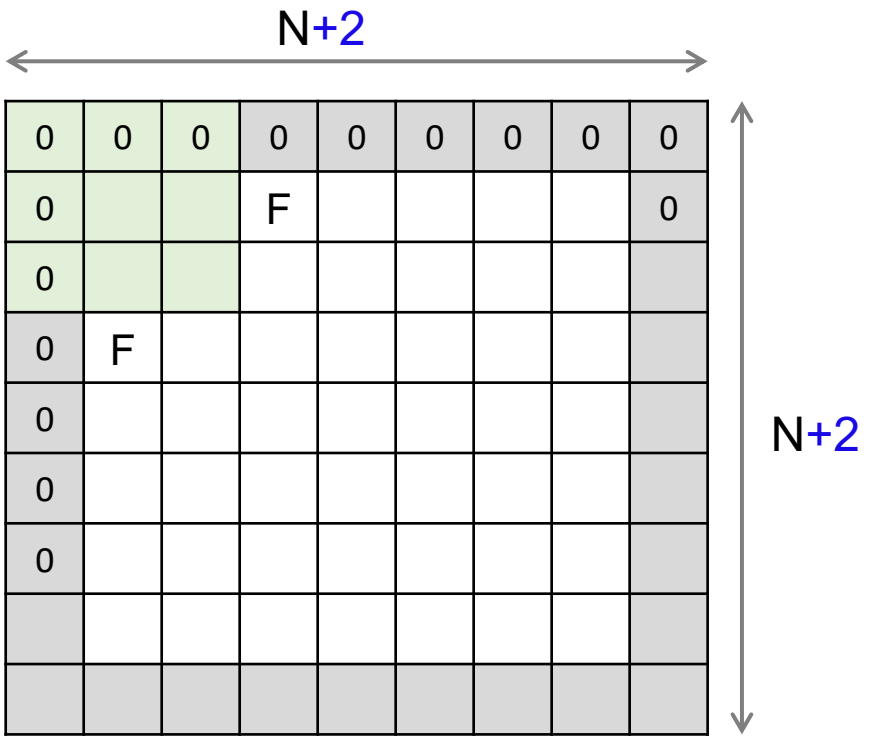
Input size: **N×N**
Filter size: **F×F**

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 :\

A Closer Look at Spatial Dimensions

In practice: Common to zero **pad** the border



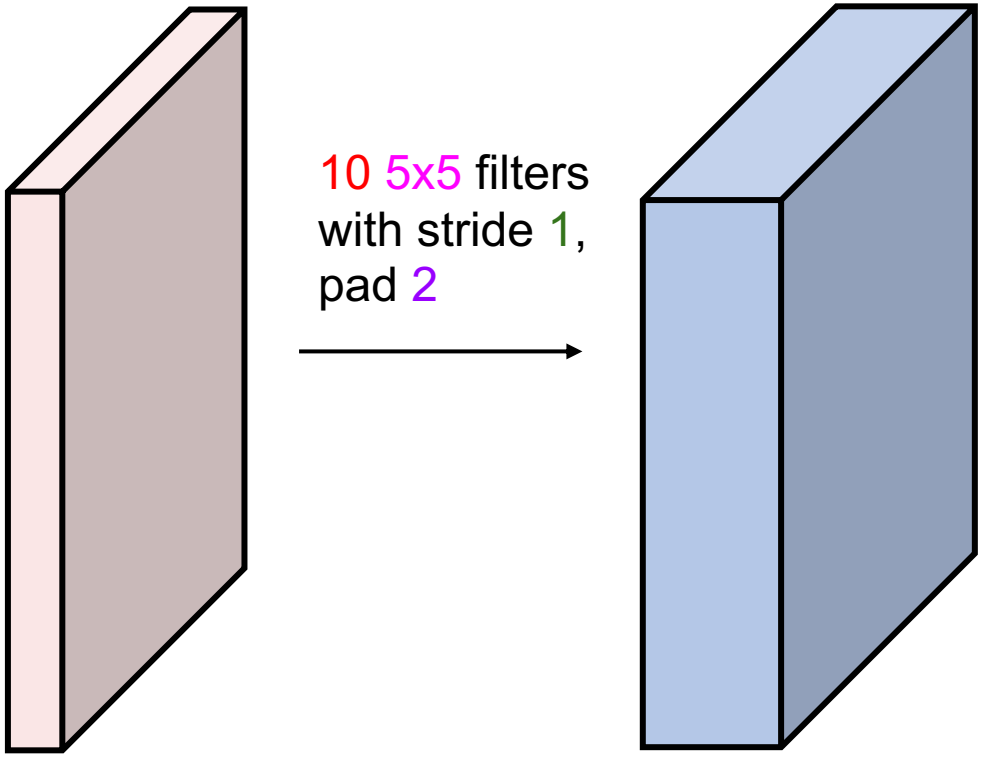
Pad with P pixel border

Output size:
(N +2P - F) / stride + 1

e.g. N = 7, F = 3:
stride 1 => (7 +2 - 3)/1 + 1 = 7
stride 2 => (7 +2 - 3)/2 + 1 = 4
stride 3 => (7 +2 - 3)/3 + 1 = 3

Quiz

- **What is the output size ?** Hint: (N+2P-F)/stride+1



Input volume:
32x32x3

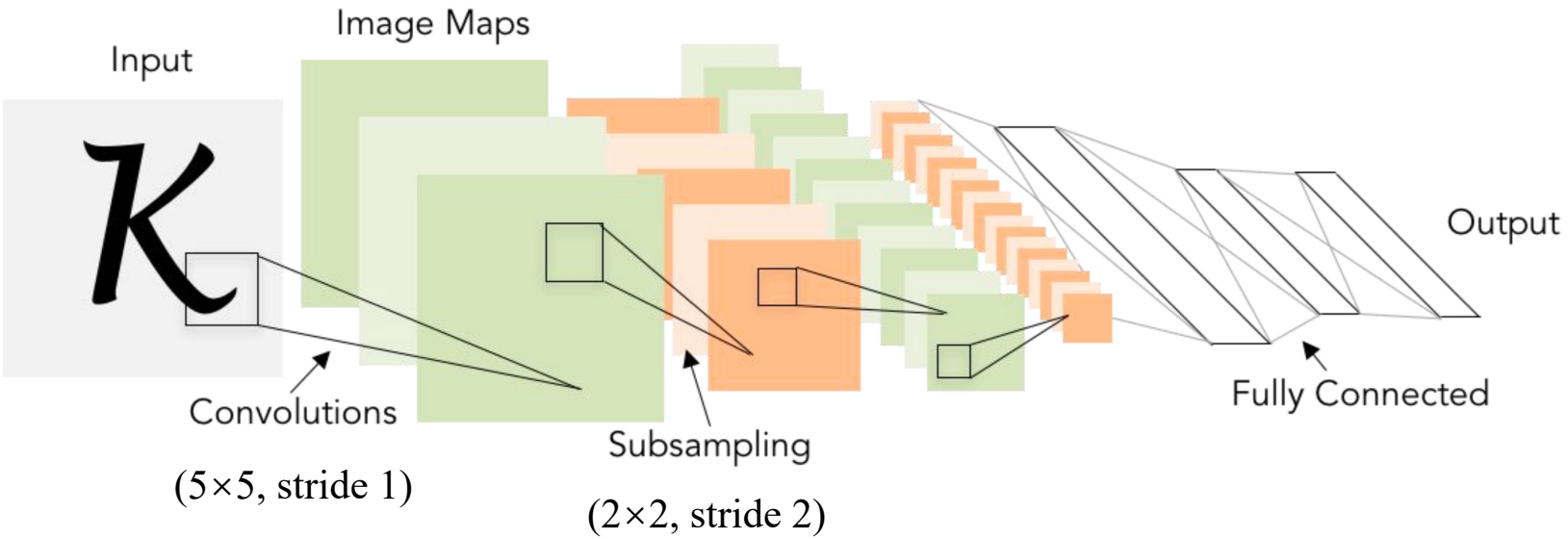
Output volume size = ?

Thinking..

- Can we let CNN act as MLP? How?
- Can we apply CNN for texts?

Some CNNs

LeNet-5 [LeCun et al., 1998]



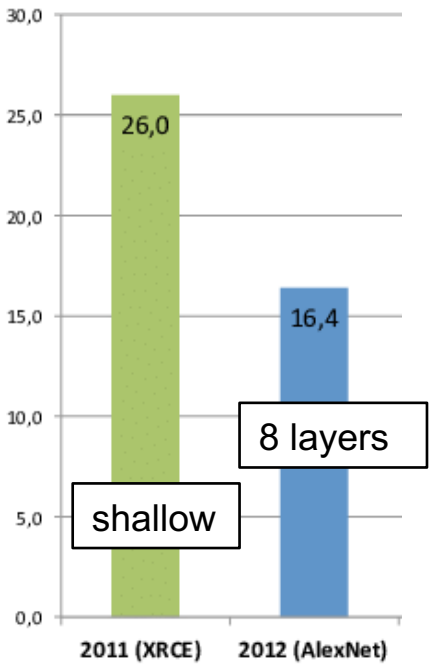
Architecture: [CONV1-POOL1-CONV2-POOL2-FC1-FC2]

Conv filters were 5×5, applied at stride 1
Subsampling (Pooling) layers were 2×2 applied at stride 2

AlexNet

- The first deep learning winner

ImageNet Classification Error (%)



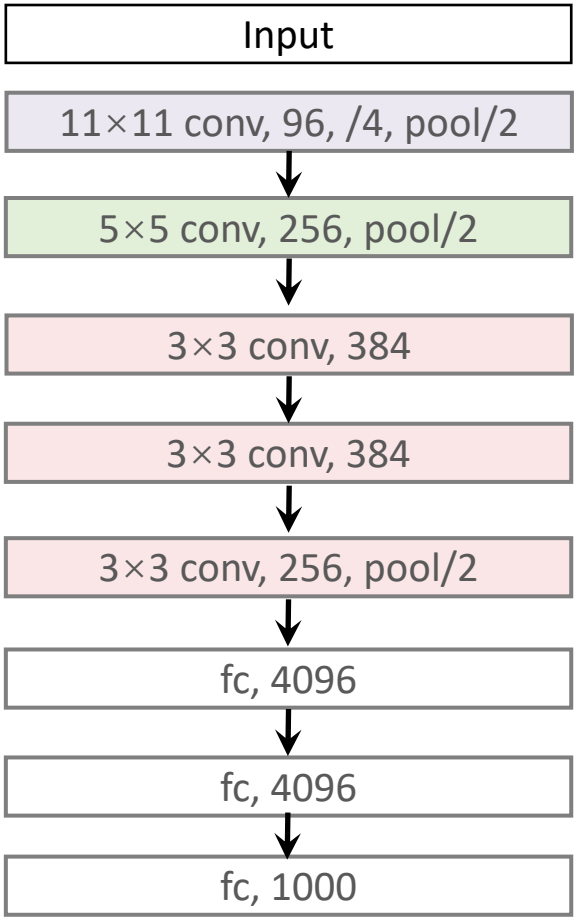
The ImageNet Large Scale Visual Recognition Challenge (ILSVRC)

ImageNet is a large scale image database organized according to the WordNet hierarchy, in which each node is depicted by thousands of images. **ImageNet** contains 14,197,122 images within more than 20k categories.

AlexNet [Krizhevsky et al. NIPS 2012]

- **Architecture:**

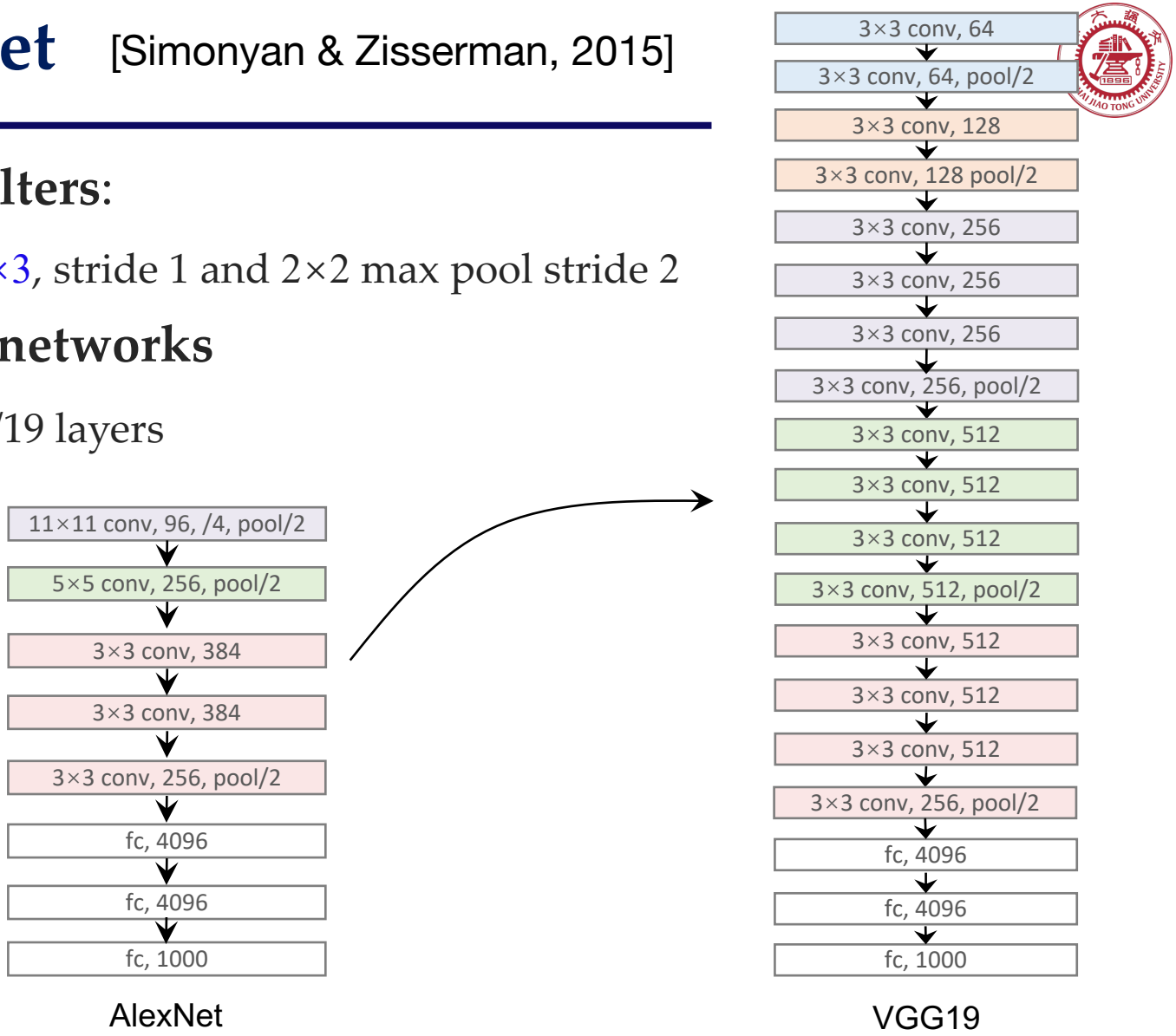
  CONV1: 96 11×11 filters at stride 4, pad 0
  MAX POOL1 : 3×3 filters at stride 2
  NORM1 : Normalization layer
  CONV2 : 256 5×5 filters at stride 1, pad 2
  MAX POOL2 : 3×3 filters at stride 2
  NORM2 : Normalization layer
  CONV3 : 384 3×3 filters at stride 1, pad 1
  CONV4 : 384 3×3 filters at stride 1, pad 1
  CONV5 : 256 3×3 filters at stride 1, pad 1
  MAX POOL3 : 3×3 filters at stride 2
  FC6 : 4096 neurons
  FC7 : 4096 neurons
  FC8 : 1000 neurons (class scores)



Krizhevsky et al: ImageNet Classification with Deep Convolutional Neural Networks, NIPS 2012

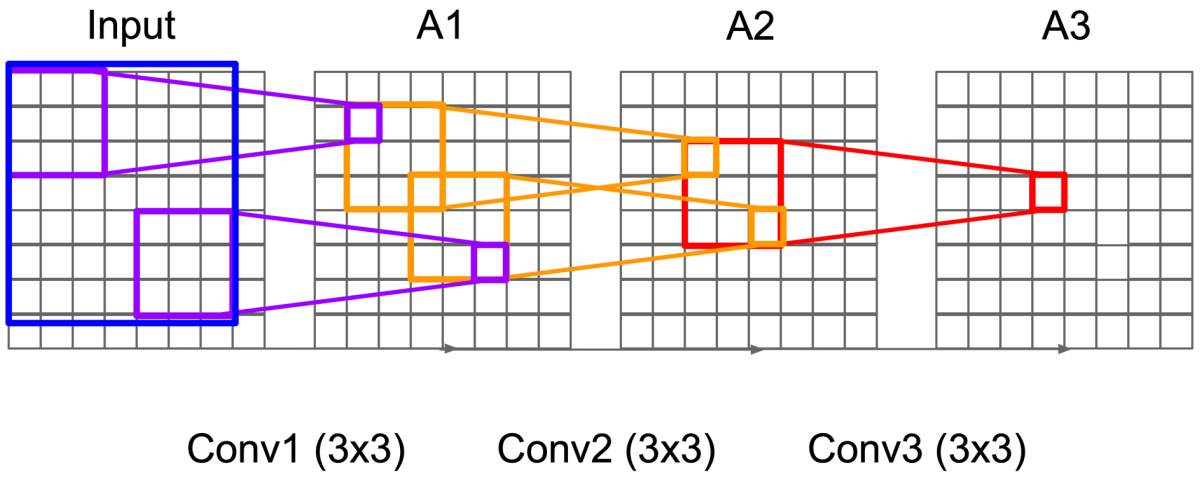
VGGNet [Simonyan & Zisserman, 2015]

- **Small filters:**
  - ⊳ only 3×3, stride 1 and 2×2 max pool stride 2
- **Deeper networks**
  - ⊳ 8 → 16/19 layers

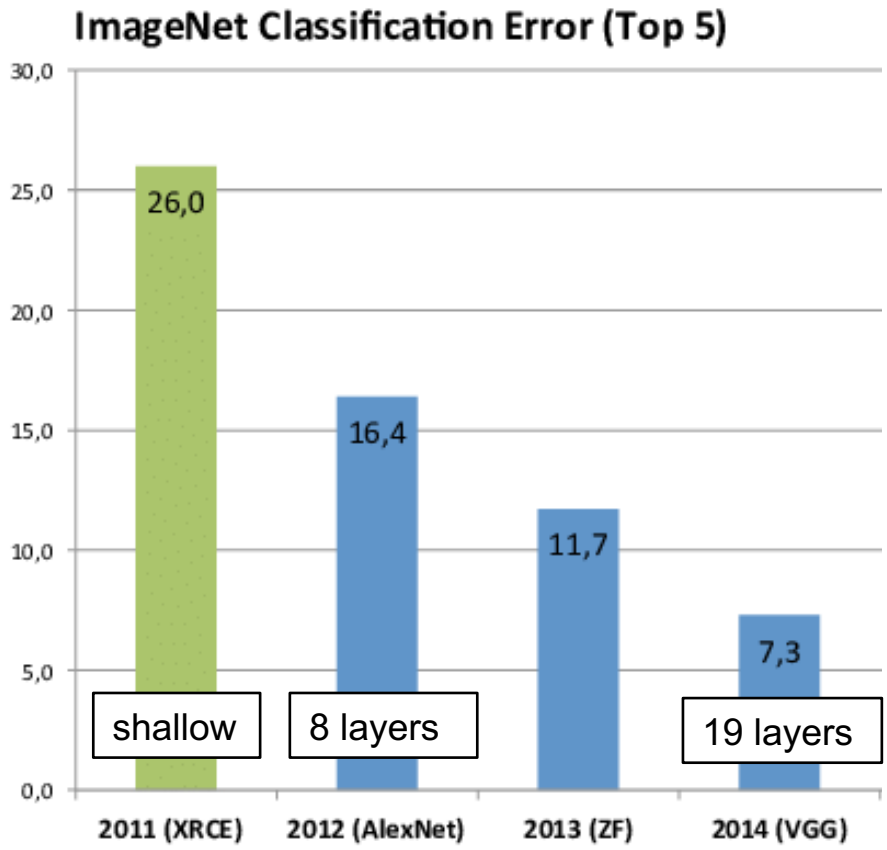


VGGNet [Simonyan & Zisserman, 2015]

- **Why use smaller filters? (3×3 conv)**
  - Stack of three 3×3 conv (stride 1) layers has same effective receptive field as one 7×7 conv layer.
  - But deeper, more non-linearities
  - And fewer parameters: $3 \times (3^2C^2)$ vs. 7^2C^2 for C channels per layer

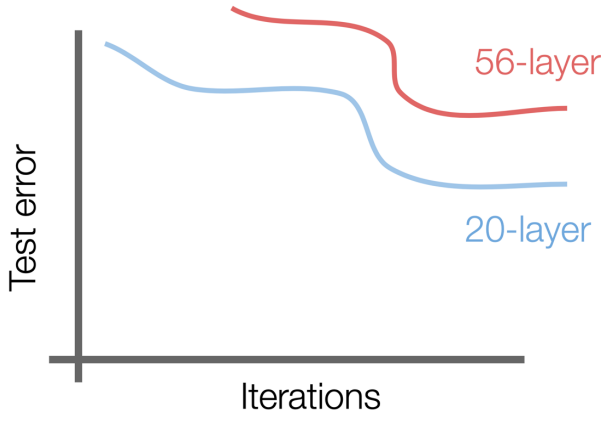


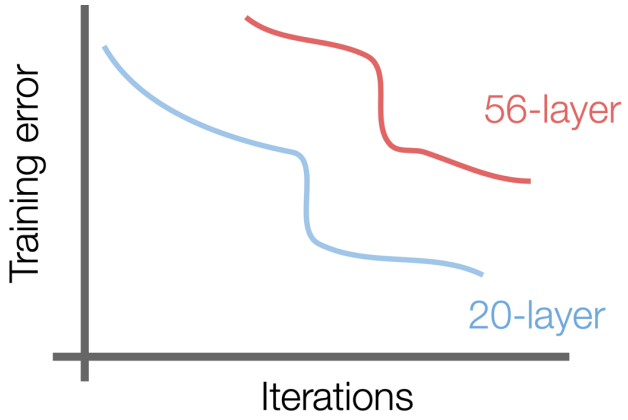
VGGNet


ImageNet Classification Error (Top 5)
30,0
25,0
20,0
15,0
10,0
5,0
0,0
26,0
16,4
11,7
7,3
shallow
8 layers
19 layers
2011 (XRCE)
2012 (AlexNet)
2013 (ZF)
2014 (VGG)

The deeper, the better?

- What happens when we continue stacking deeper layers on a "plain" convolutional neural network?





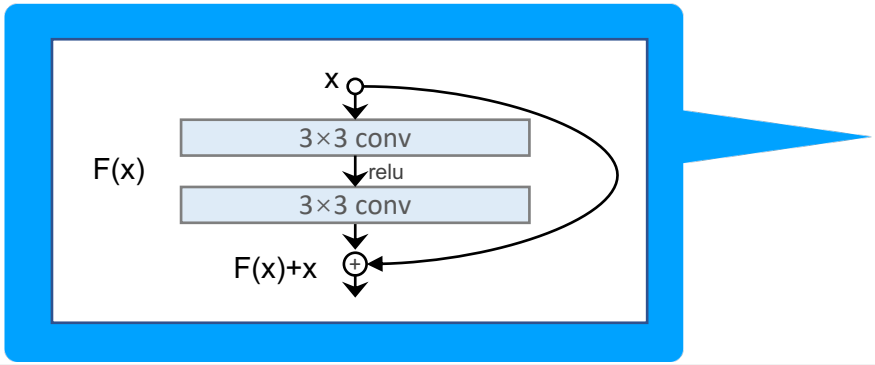
The deeper model performs worse, but it's not caused by overfitting!

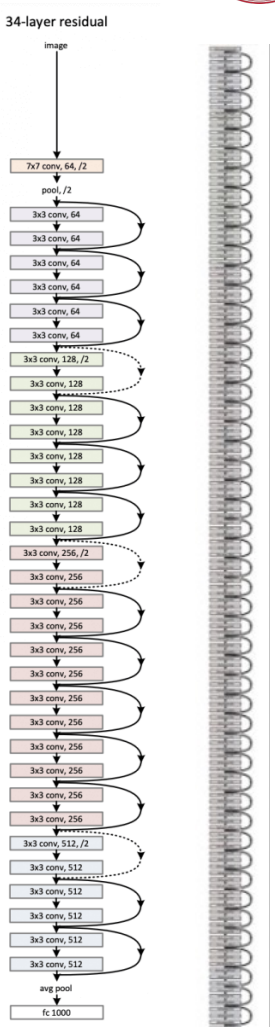
Hypothesis: the problem is an *optimization* problem, **deeper models are harder to optimize**

ResNet [He et al. 2015]

- Very deep CNNs using residual connections

  - Stack of many **residual blocks**
  - Every residual block has two 3x3 conv layers
  - …



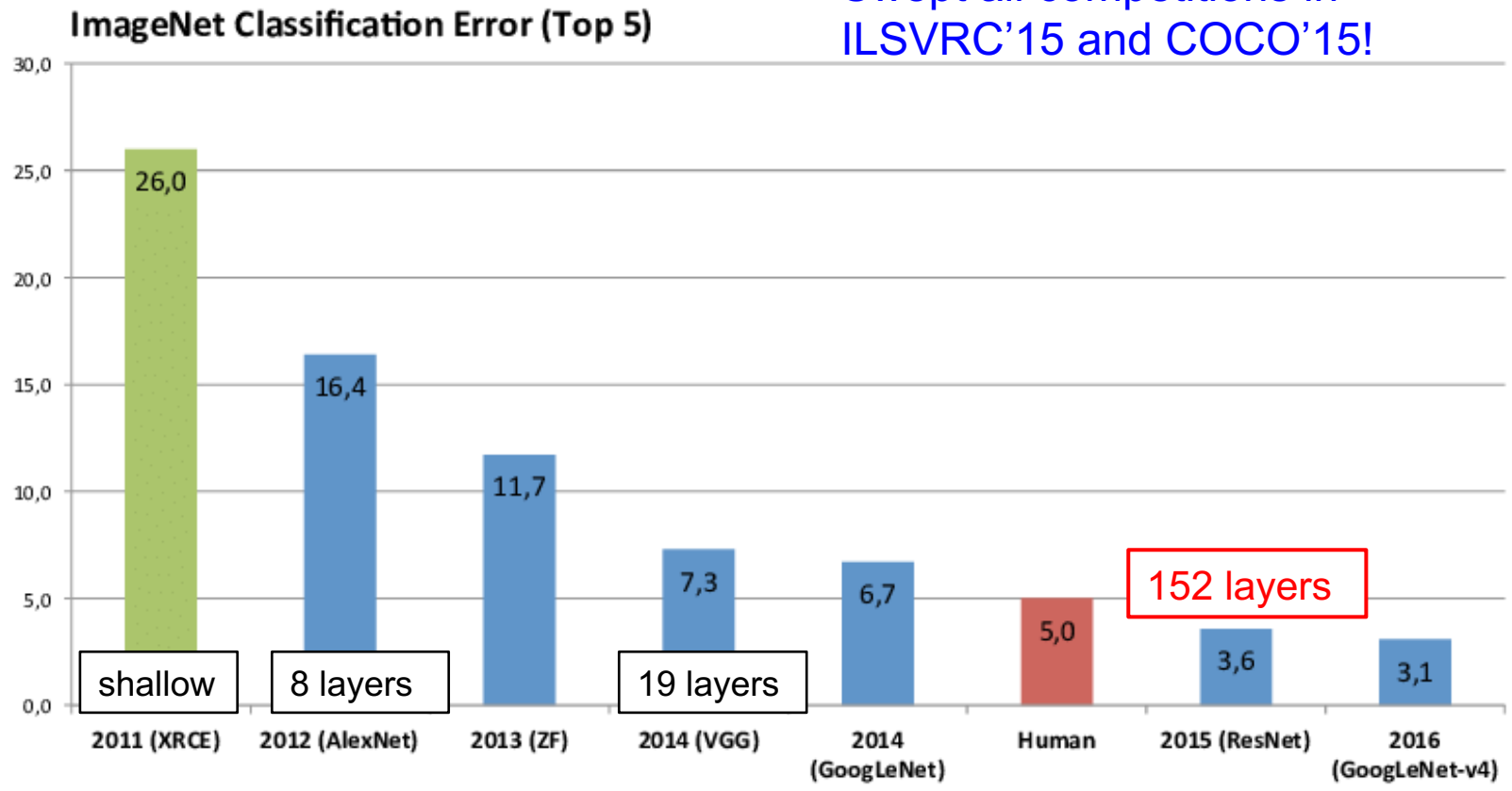


Why use residual connection?

1. gradients can propagate faster through a highway (recall LSTM)
2. within each block, only small residuals have to be learned
3. let deeper model to perform as good as shallower model by copying the learned layers from the shallower model and setting additional layers to identity mapping.

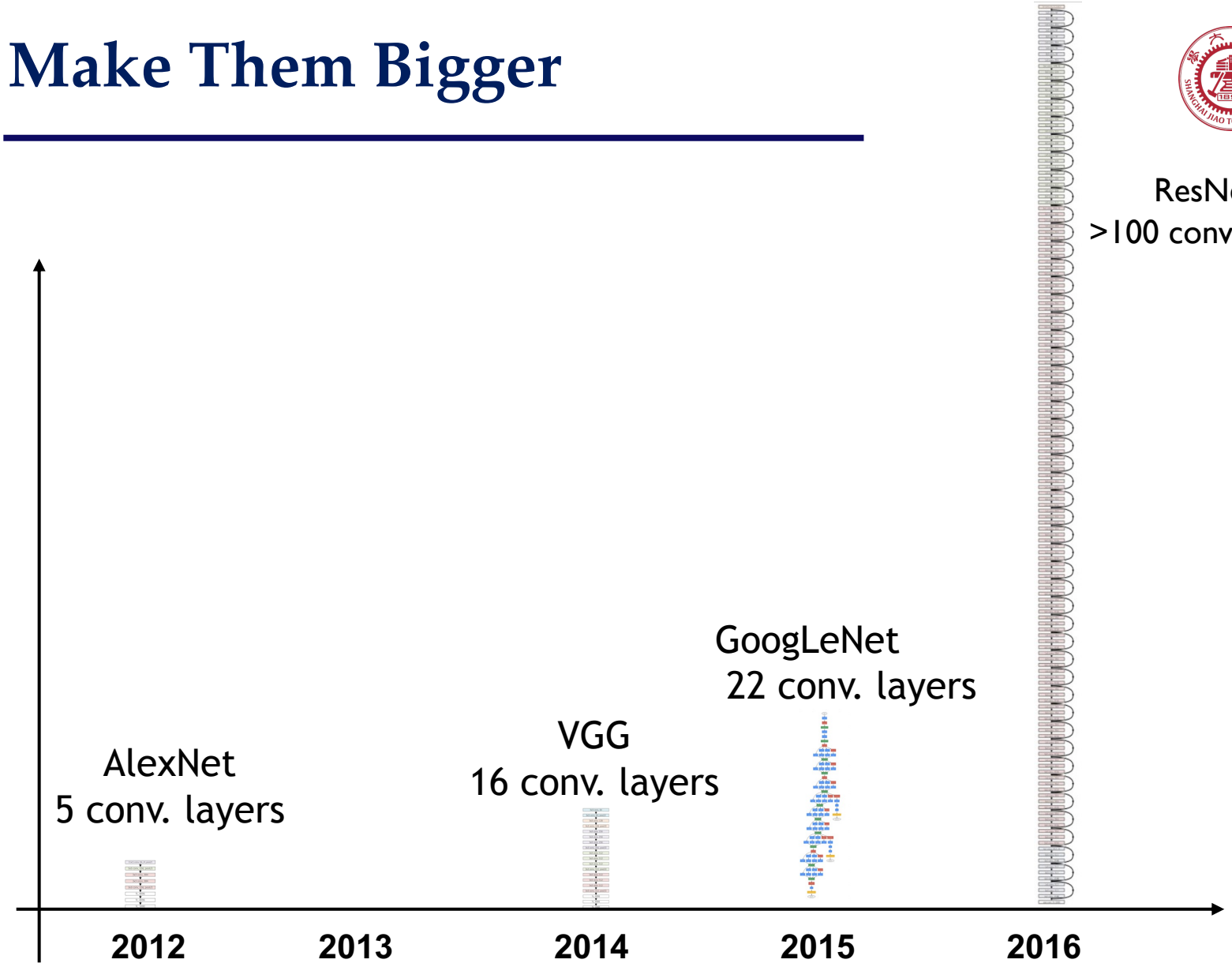
ResNet

- "The Revolution of Depth"
  - 152-layer model for ImageNet
  - Swept all competitions in ILSVRC'15 and COCO'15!

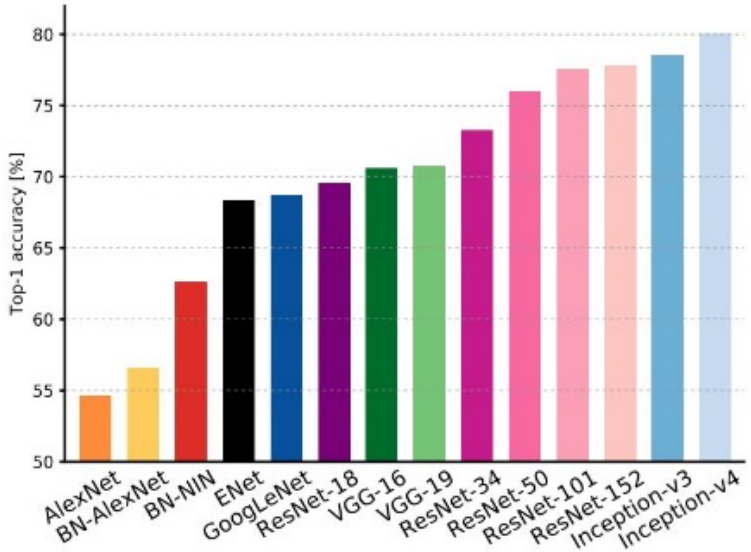


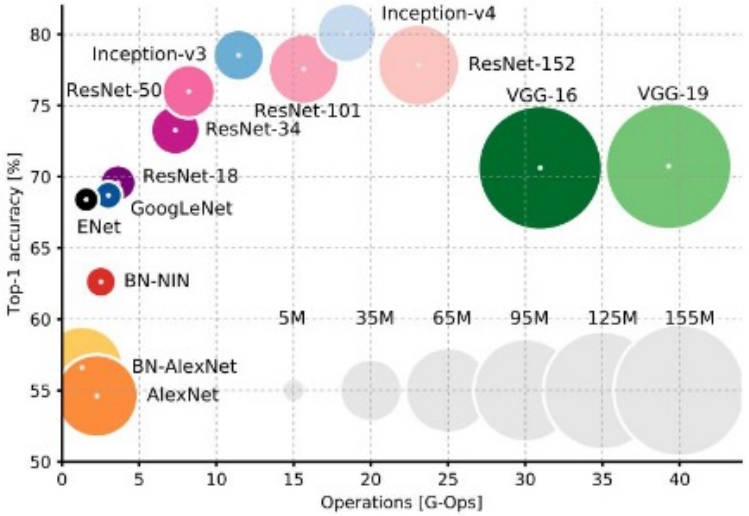
He et al. Deep Residual Learning for Image Recognition. CVPR 2016.

Make Them Bigger



Comparing Complexity..





An Analysis of Deep Neural Network Models for Practical Applications, 2017.

TIME for Coding

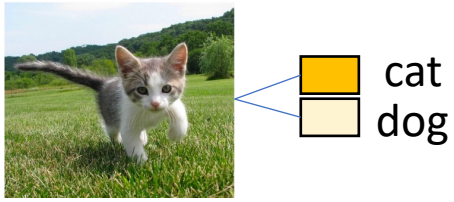
Tutorial: MNIST image recognition using PyTorch

- https://www.kaggle.com/code/turksoyomer/pytorch-tutorial-on-mnist-dataset

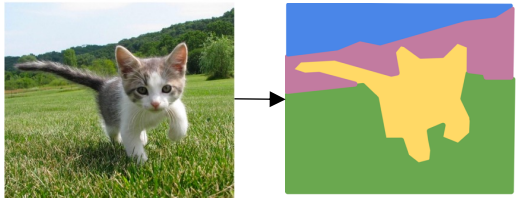
What's Next?

The World of Computer Vision

Classification



Semantic Segmentation



Object Detection

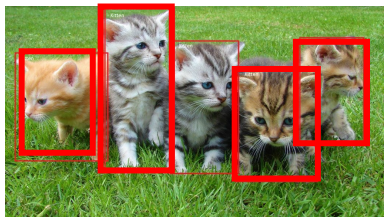


Image Captioning

Image Generation