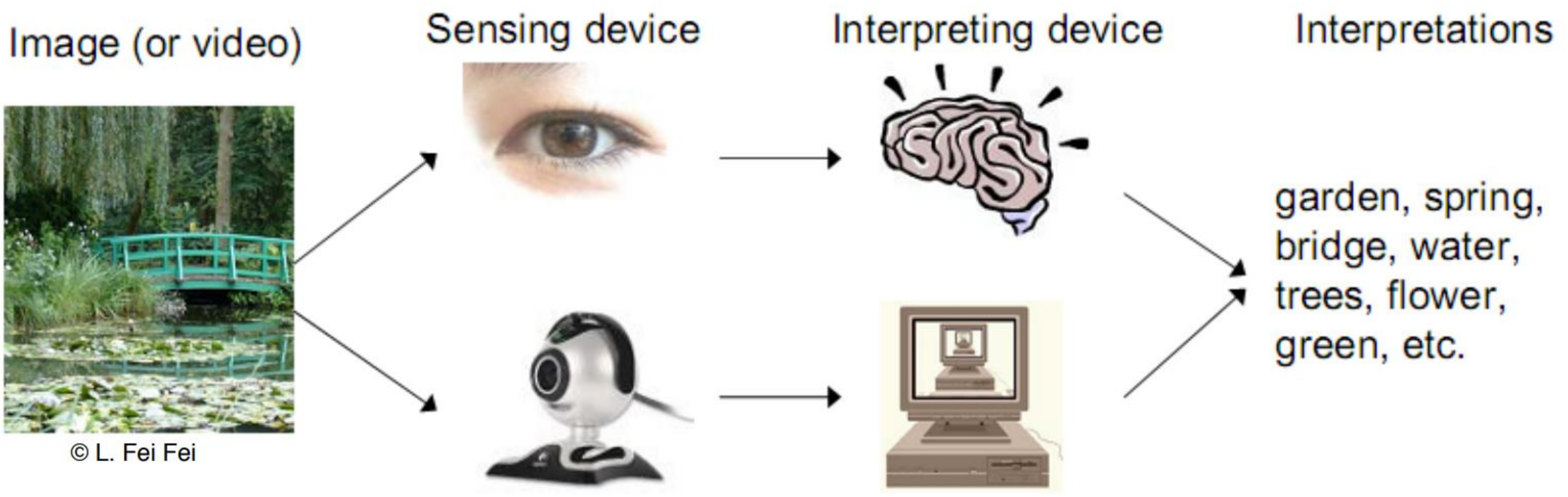
西安交通大學
XI'AN JIAOTONG UNIVERSITY

CV Introduction I

What is computer vision?

Computer Vision: The study of how computers can be programmed to extract useful information about the environment from optical images. -- S.E. Palmer, Vision science (1999)

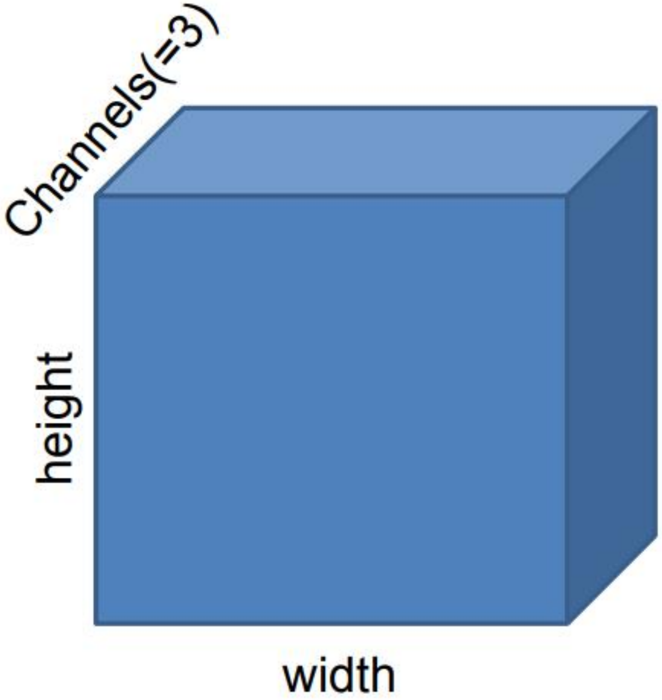


What is the input?

- A (gray-scale) image is a 2D array

0	3	2	5	4	7	6	9	8
3	0	1	2	3	4	5	6	7
2	1	0	3	2	5	4	7	6
5	2	3	0	1	2	3	4	5
4	3	2	1	0	3	2	5	4
7	4	5	2	3	0	1	2	3
6	5	4	3	2	1	0	3	2
9	6	7	4	5	2	3	0	1
8	7	6	5	4	3	2	1	0

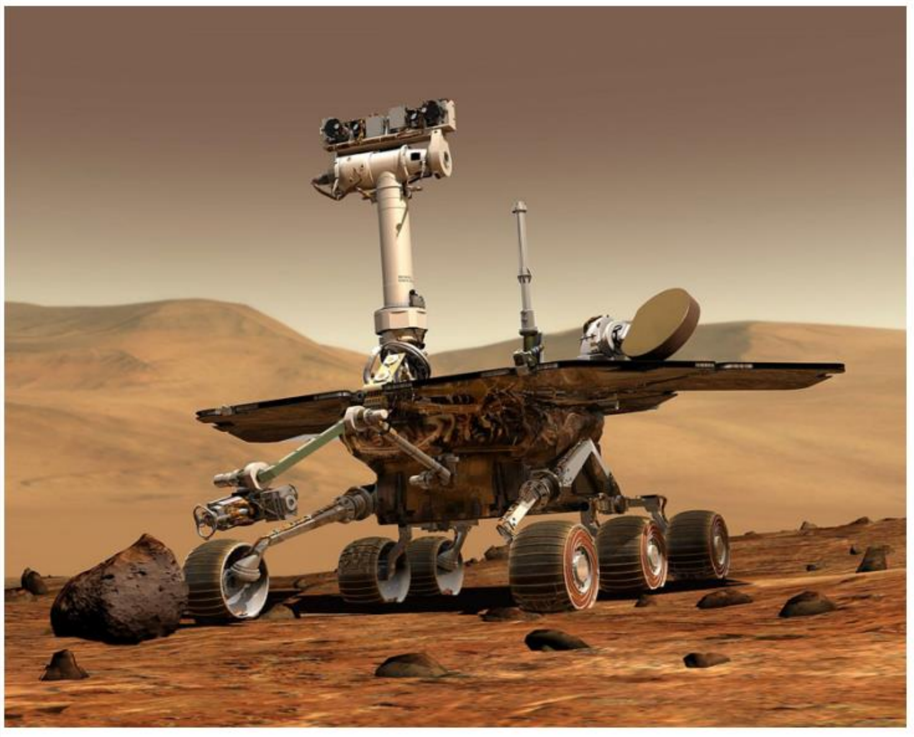
What is the input?



What is the output?

- Depends on what we want to do with the image

What do we do with images?



Examples 1: Robotics

- Understanding terrain and identifying obstacles

What do we do with images?

Examples 1: Robotics

- Understanding terrain and identifying obstacles
- Identifying people and understanding their intentions

What do we do with images?

Example 2: Internet Vision

- Recognizing obscene/ violent content
- Creating new content (image editing)


Facebook Users Are Uploading 350 Million New Photos Each Day

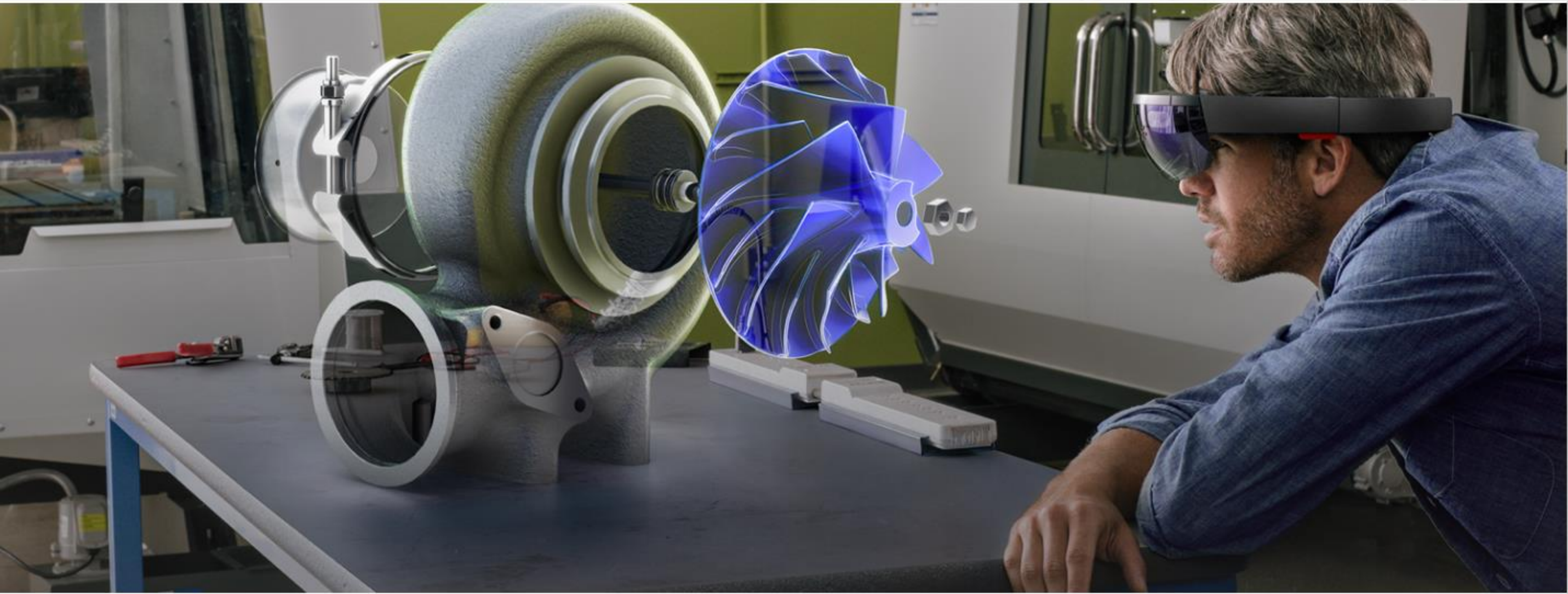
Cooper Smith

Sep. 18, 2013, 8:00 AM 23,351

What do we do with images?

Example 3: AR/VR

- Understand 3D structure of the world



The goals of computer vision

- Reconstruction

  Understanding 3D structure of the world

- Grouping / Re-organization

  Group pixels into objects

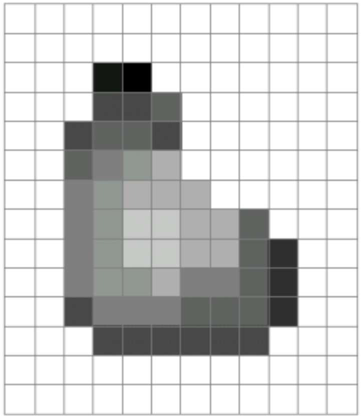
- Recognition

  Classify objects, scenes, actions…

Image filtering

What is an image?

- A grid (matrix) of intensity values



=

255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	20	0	255	255	255	255	255	255	255
255	255	255	75	75	75	255	255	255	255	255	255
255	255	75	95	95	75	255	255	255	255	255	255
255	255	96	127	145	175	255	255	255	255	255	255
255	255	127	145	175	175	175	255	255	255	255	255
255	255	127	145	200	200	175	175	95	255	255	255
255	255	127	145	200	200	175	175	95	47	255	255
255	255	127	145	145	175	127	127	95	47	255	255
255	255	74	127	127	127	95	95	95	47	255	255
255	255	255	74	74	74	74	74	74	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255
255	255	255	255	255	255	255	255	255	255	255	255

(common to use one byte per value: 0 = black, 255 = white)

Images as functions

- Can think of image as a function, f, from $\mathbb{R}^2$ to $\mathbb{R}$ or $\mathbb{R}^M$

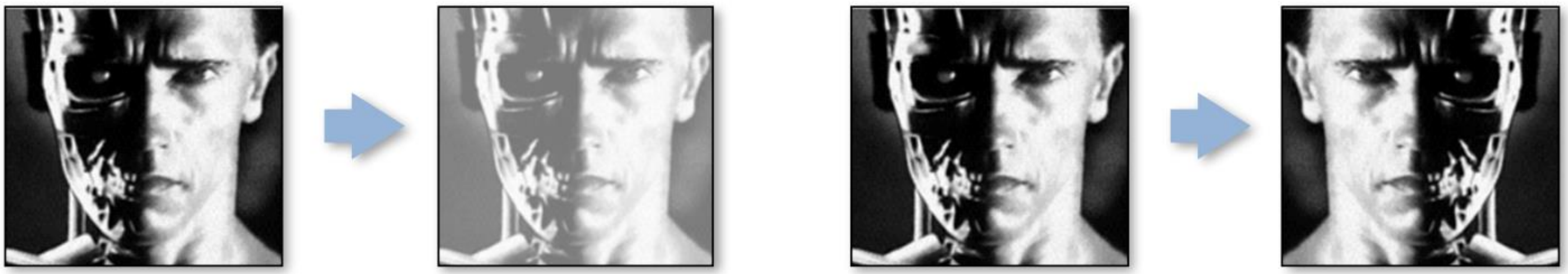
  - Grayscale: $f(x, y)$ gives intensity at position (x, y)

$$f: [a, b] \times [c, d] \to [0,255]$$

  - Color: $f(x, y) = [r(x, y), g(x, y), b(x, y)]$

Image Processing: Image transformations

Input: Image --> Output: Image



$g(x,y) = f(x,y) + 20$ $\qquad$ $g(x,y) = f(-x,y)$

Image denoising

Noise reduction

- Nearby pixels are likely to belong to same object

  - thus likely to have similar color

- Replace each pixel by average of neighbors

Mean filtering

0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	10	0	0	0	0
0	0	10	20	20	20	10	40	0	0
0	10	20	30	0	20	10	0	0	0
0	10	0	30	40	30	20	10	0	0
0	10	20	30	40	30	20	10	0	0
0	10	20	10	40	30	20	10	0	0
0	10	20	30	30	20	10	0	0	0
0	0	10	20	20	0	10	0	20	0
0	0	0	10	10	10	0	0	0	0

$$(0 + 0 + 0 + 10 + 40 + 0 + 10 + 0 + 0) / 9 = 6.66$$

Mean filtering

0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	10	0	0	0	0
0	0	10	20	20	20	10	40	0	0
0	10	20	30	0	20	10	0	0	0
0	10	0	30	40	30	20	10	0	0
0	10	20	30	40	30	20	10	0	0
0	10	20	10	40	30	20	10	0	0
0	10	20	30	30	20	10	0	0	0
0	0	10	20	20	0	10	0	20	0
0	0	0	10	10	10	0	0	0	0

$(0 + 0 + 0 + 0 + 0 + 10 + 0 + 0 + 0 + 0 + 20 + 10 + 40 + 0 + 0 + 20 + 10 + 0 + 0 + 0 + 30 + 20 + 10 + 0 + 0) / 25 = 6.8$

Mean filtering

0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	10	0	0	0	0
0	0	10	20	20	20	10	40	0	0
0	10	20	30	0	20	10	0	0	0
0	10	0	30	40	30	20	10	0	0
0	10	20	30	40	30	20	10	0	0
0	10	20	10	40	30	20	10	0	0
0	10	20	30	30	20	10	0	0	0
0	0	10	20	20	0	10	0	20	0
0	0	0	10	10	10	0	0	0	0

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

$$(0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 10)/9 = 1.11$$

Mean filtering

0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	10	0	0	0	0
0	0	10	20	20	20	10	40	0	0
0	10	20	30	0	20	10	0	0	0
0	10	0	30	40	30	20	10	0	0
0	10	20	30	40	30	20	10	0	0
0	10	20	10	40	30	20	10	0	0
0	10	20	30	30	20	10	0	0	0
0	0	10	20	20	0	10	0	20	0
0	0	0	10	10	10	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	1	4	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

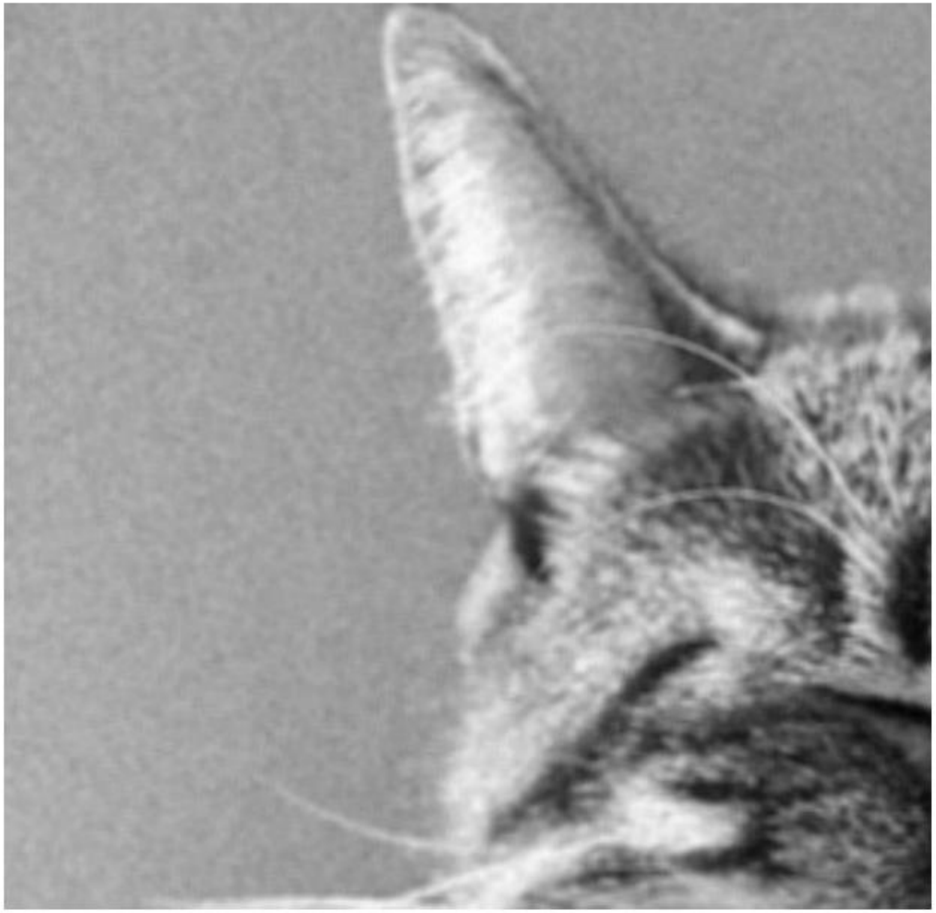
$$(0 + 0 + 0 + 0 + 0 + 10 + 0 + 10 + 20)/9 = 4.44$$

Mean filtering

0	0	0	0	0	0	0	0	0	0
0	0	0	10	10	10	0	0	0	0
0	0	10	20	20	20	10	40	0	0
0	10	20	30	0	20	10	0	0	0
0	10	0	30	40	30	20	10	0	0
0	10	20	30	40	30	20	10	0	0
0	10	20	10	40	30	20	10	0	0
0	10	20	30	30	20	10	0	0	0
0	0	10	20	20	0	10	0	20	0
0	0	0	10	10	10	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	1	4	8	10	8	9	6	4	0
0	4	11	13	16	11	12	7	4	0
0	6	14	19	23	19	18	10	6	0
0	8	18	23	28	23	17	8	2	0
0	8	16	26	31	30	20	10	3	0
0	10	18	27	29	27	17	8	2	0
0	8	14	22	22	20	11	8	3	0
0	4	11	17	17	12	6	4	2	0
0	0	0	0	0	0	0	0	0	0

Noise reduction using mean filtering



Question?

Filters

- Filtering
  - Form a new image whose pixels are a combination of the original pixels
- Why?
  - To get useful information from images
    - E.g., extract edges or contours (to understand shape)
  - To enhance the image
    - E.g., to blur to remove noise
    - E.g., to sharpen to “enhance image”

Mean filtering

- Replace pixel by mean of neighborhood

10	5	3
4	5	1
1	1	7

Local image data

f

	4.1	

Modified image data

$S[f]$

$$S[f](m,n) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} f(m+i, n+j)/9$$

A more general version

10	5	3
4	5	1
1	1	7

Local image data

	7	

Kernel / filter

$$S[f](m,n) = \sum_{i=-1}^{1} \sum_{j=-1}^{1} \color{red}{w(i,j)} f(m+i, n+j)$$

A more general version

0	10	5	7	0
5	11	6	8	3
9	22	4	5	1
2	9	14	6	7
3	10	15	12	9

Local image data

		7		

Kernel size = 2k+1

$$S[f](m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

A more general version

$$S[f](m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

- $w(i,j) = \frac{1}{(2k+1)^2}$ for mean filter
- If $w(i,j) \geq 0$ and sum to 1, weighted mean
- But $w(i,j)$ can be arbitrary real numbers!

Convolution and cross-correlation

Convolution and cross-correlation

- Cross correlation

$$S[f] = w \otimes f$$

$$S[f](m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

- Convolution

$$S[f] = w * f$$

$$S[f](m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m-i, n-j)$$

Cross-correlation

1	2	3
4	5	6
7	8	9

w

1	2	3
4	5	6
7	8	9

f

$$1*1 + 2*2 + 3*3 + 4*4 + 5*5 + 6*6 + 7*7 + 8*8 + 9*9$$

Convolution

1	2	3
4	5	6
7	8	9

w

1	2	3
4	5	6
7	8	9

f

$$1*9 + 2*8 + 3*7 + 4*6 + 5*5 + 6*4 + 7*3 + 8*2 + 9*1$$

Properties: Linearity

$$(w \otimes f)(m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

$$f'(m,n) = af(m,n)$$

$$(w \otimes f')(m,n) = a(w \otimes f)(m,n)$$

Properties: Linearity

$$(w \otimes f)(m,n) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} w(i,j) f(m+i, n+j)$$

$$f' = af + bg$$

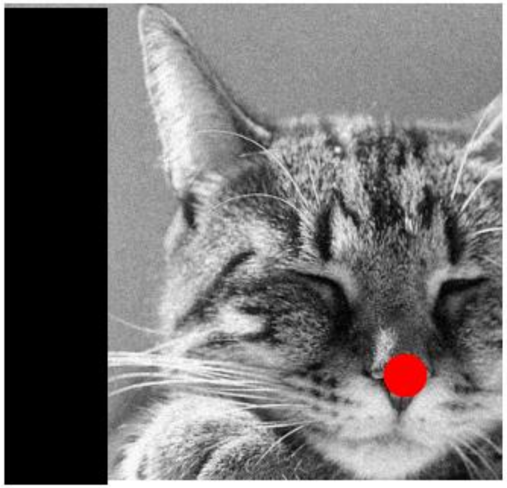
$$w \otimes f' = a(w \otimes f) + b(w \otimes g)$$

Properties: Shift invariance

$$f'(m,n) = f(m - m_0, n - n_0)$$
$$(w \otimes f')(m,n) = (w \otimes f)(m - m_0, n - n_0)$$

- Shift, then convolve = convolve, then shift
- Output of convolution does not depend on where the pixel is

f

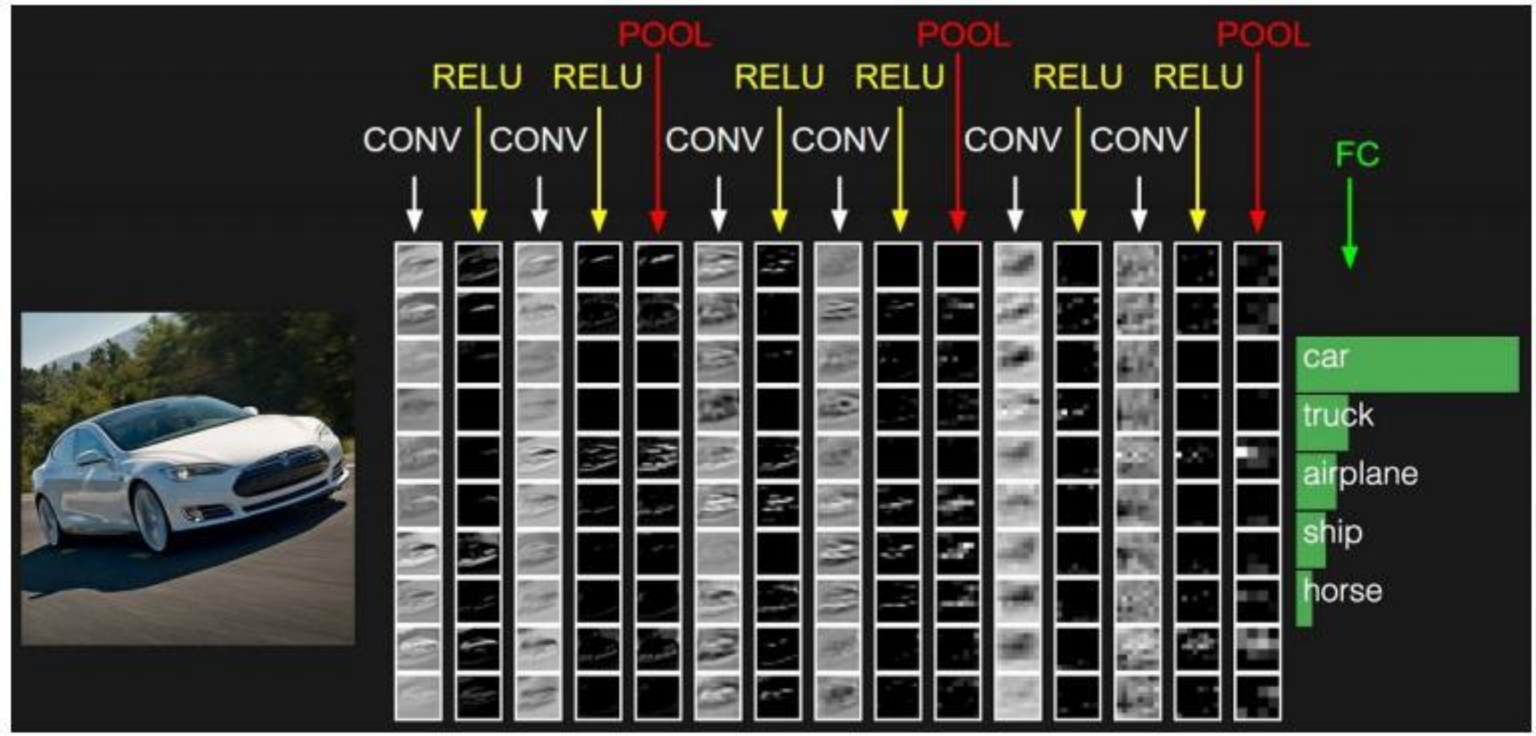


f'

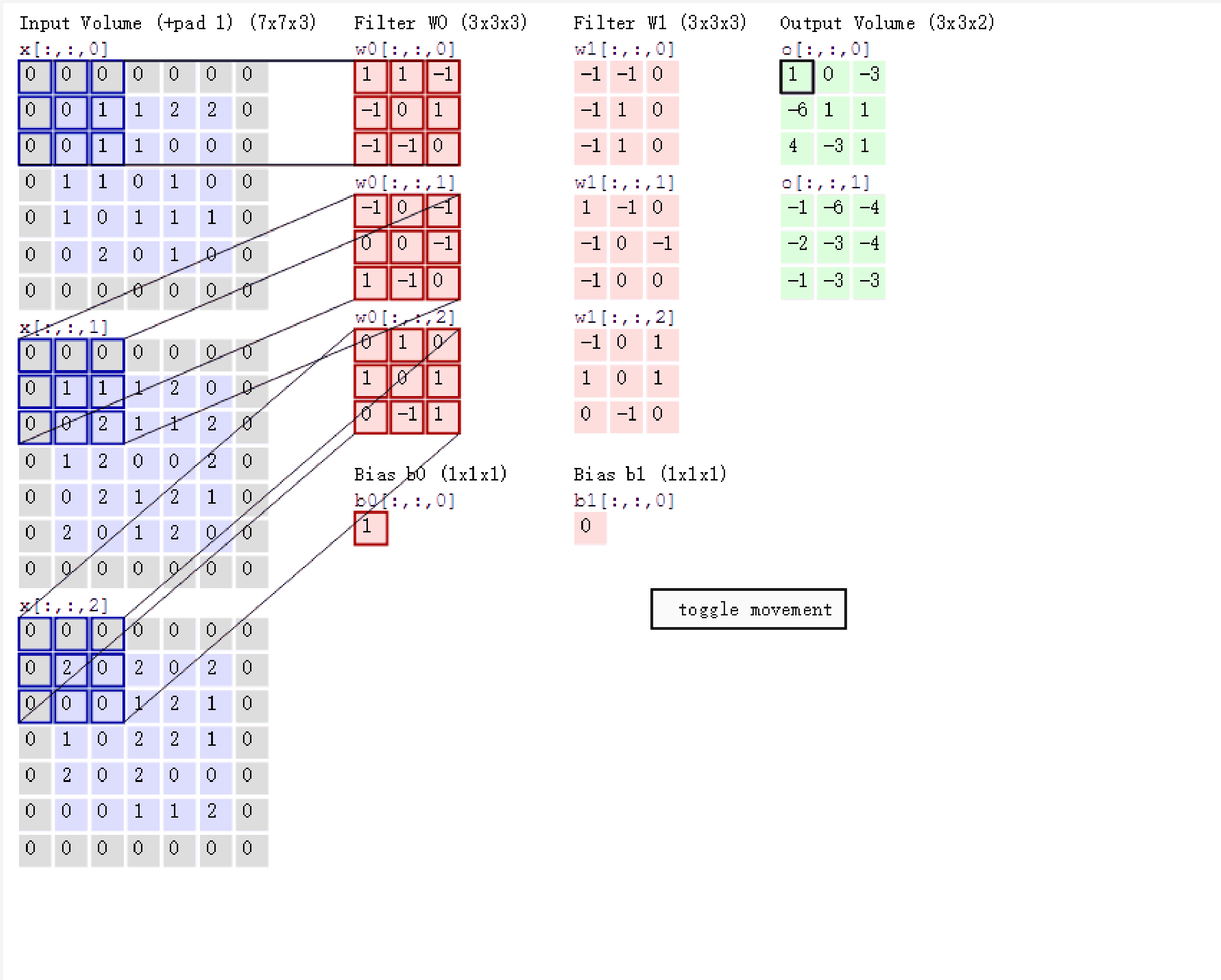
- We *like* linearity
  - Linear functions behave predictably when input changes
  - Lots of theory just easier with linear functions
- *All linear shift-invariant systems can be expressed as a convolution*

CNN

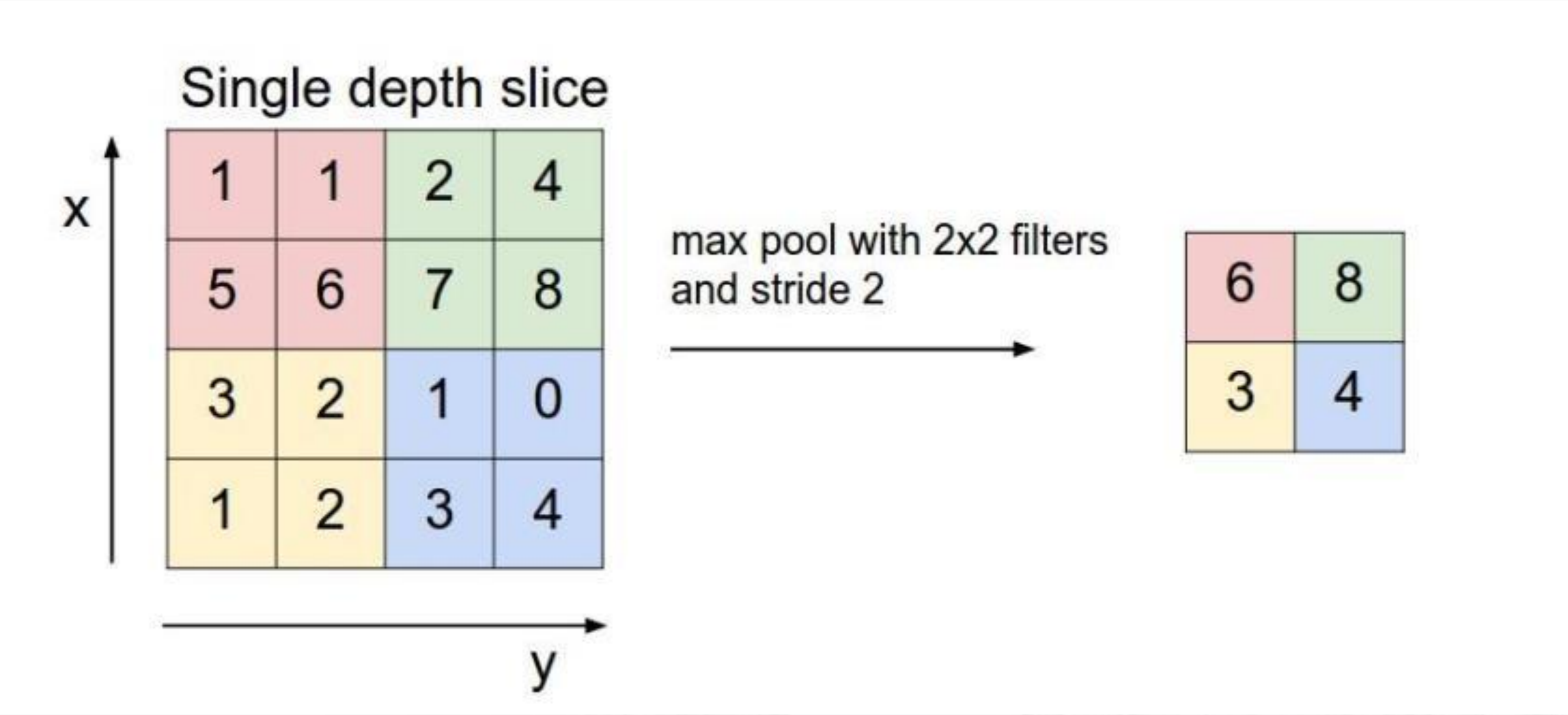
CNN for image classification



Convolution



Pooling



Single depth slice
x
1 1 2 4
5 6 7 8
3 2 1 0
1 2 3 4
y
max pool with 2x2 filters
and stride 2
6 8
3 4

Why pooling?

- Subsampling pixels will not change the object

bird

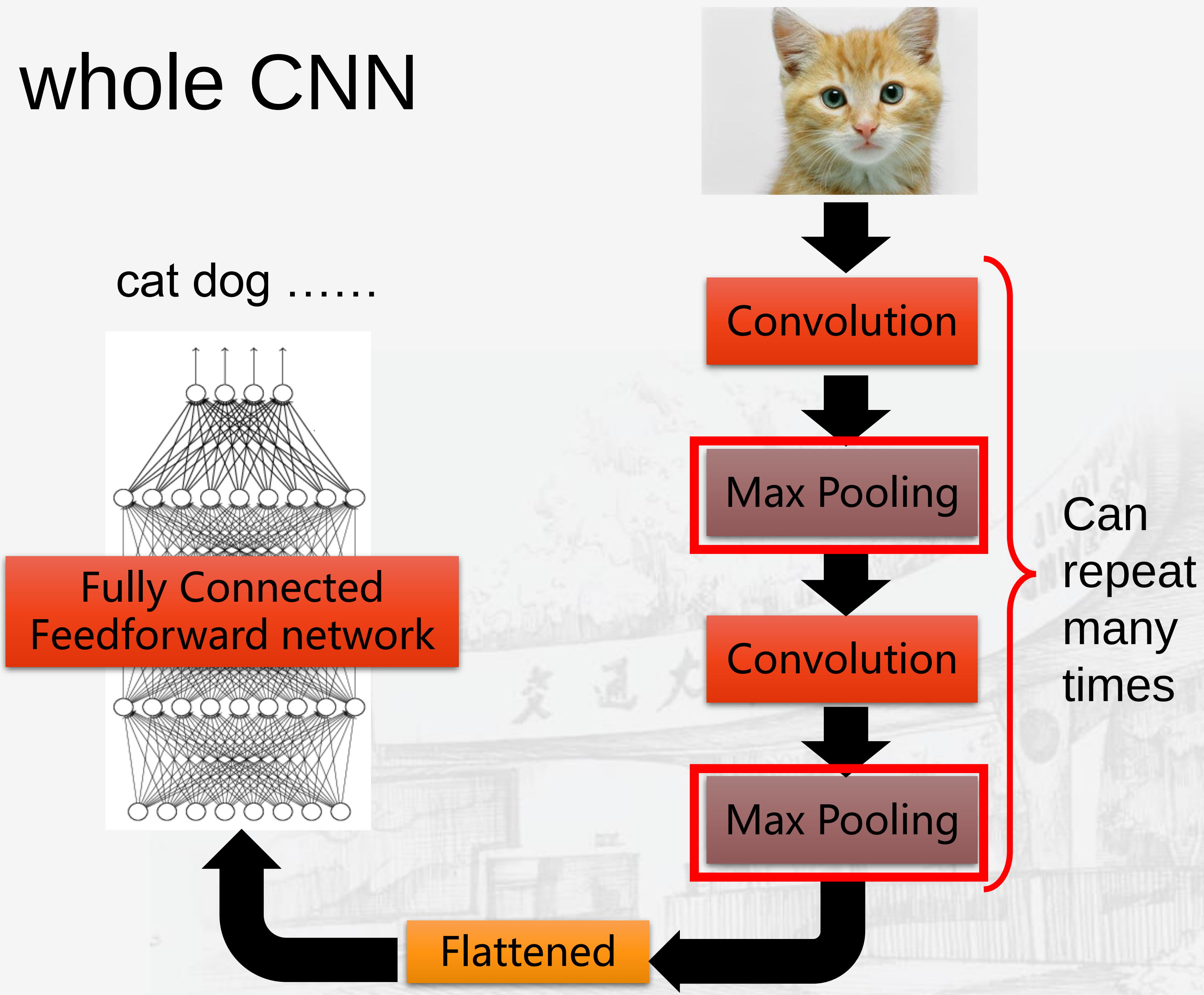
Subsampling

bird

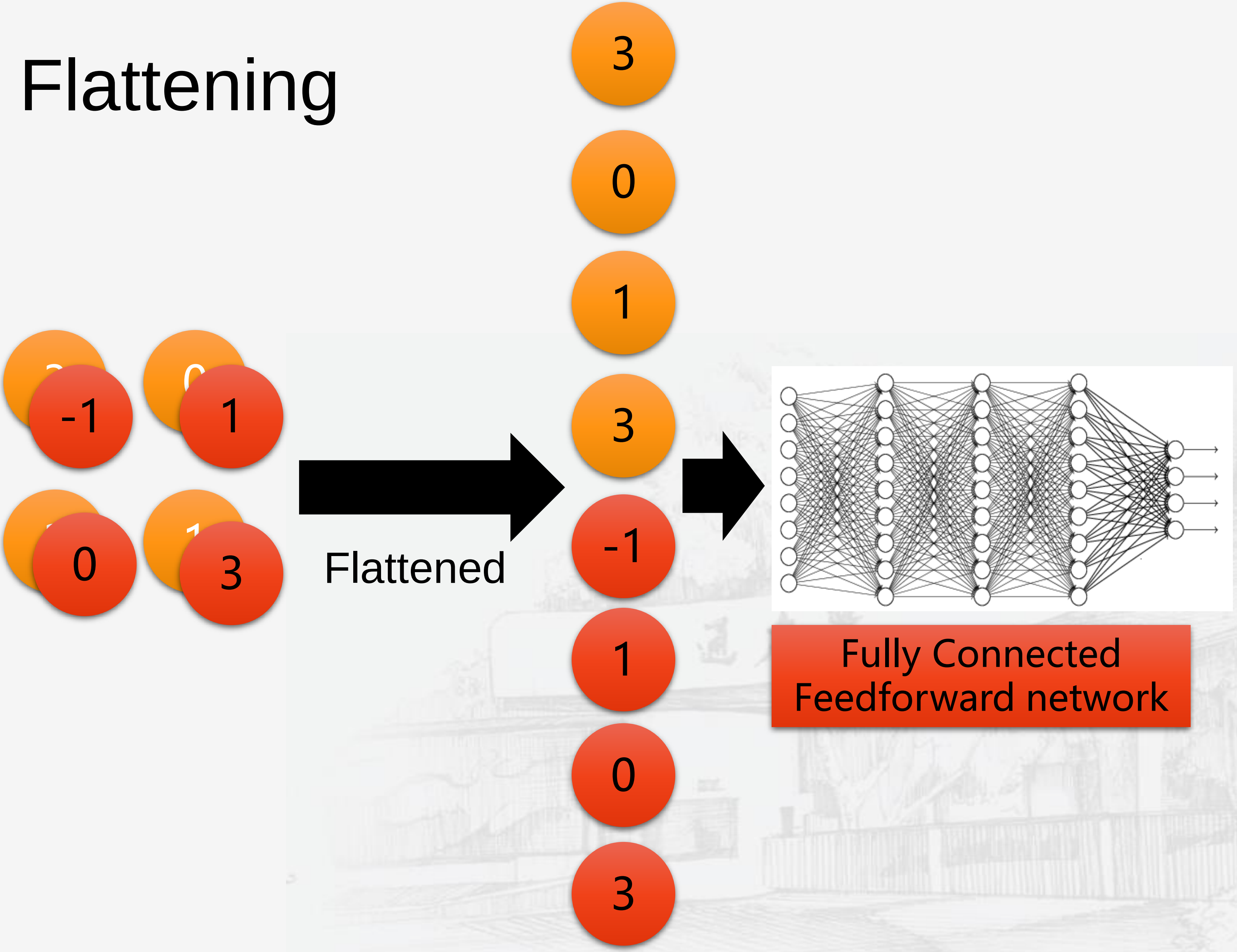
We can subsample the pixels to make image smaller

fewer parameters to characterize the image

The whole CNN



Flattening



-1
1
0
3
Flattened
3
0
1
3
-1
1
0
3
Fully Connected
Feedforward network

西安交通大學
XI'AN JIAOTONG UNIVERSITY

CV Introduction II

Convolution and Image Classifier

Wan Herun
Xi'an Jiaotong University
wanherun@stu.xjtu.edu.cn

Convolution kernel

Input images	Convolution Kernel	output images

0	0	0
0	1	0
0	0	0

0	-1	0
-1	4	-1
0	-1	0

0	-1	0
-1	5	-1
0	-1	0

Convolution kernel

-1	-1	-1
-1	9	-1
-1	-1	-1

- The convolution kernel can extract some features from original images

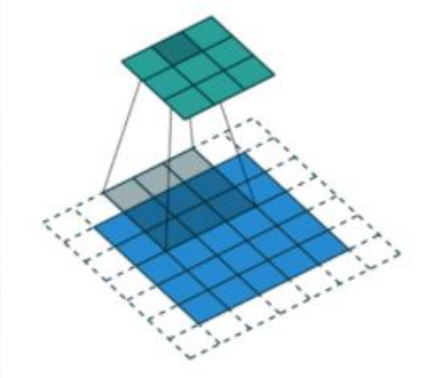
- Different convolution kernel can extract different features
  - the weight
  - the size

Convolution kernel

Receptive Field

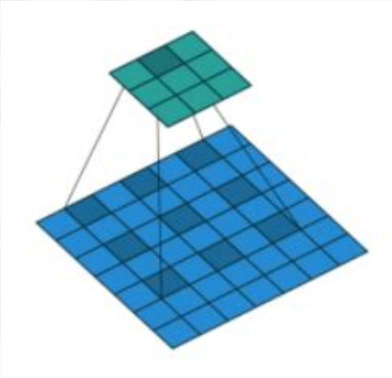
After 2 3*3 kernel?

Normal convolution



Kernel size
Stride
Padding
Input and output channels

Dilated convolution
dilation rate



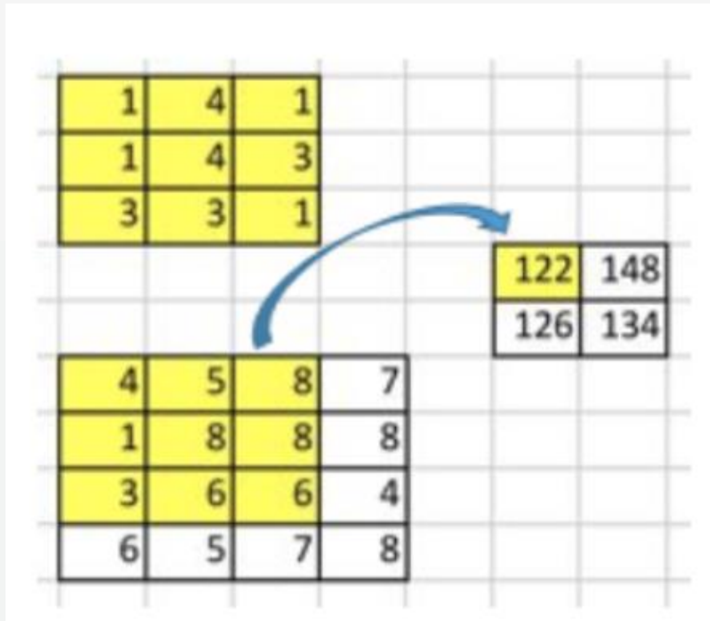
Expand the receptive field
Not lose the resolution

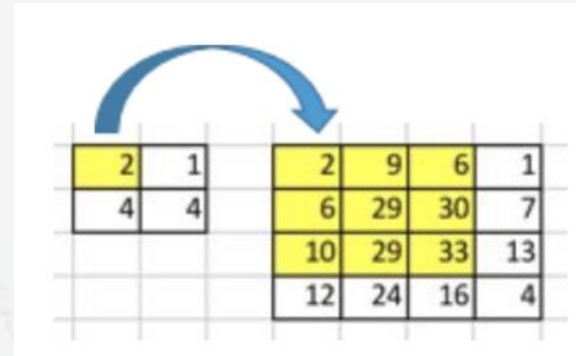
Convolution kernel

Transposed convolution

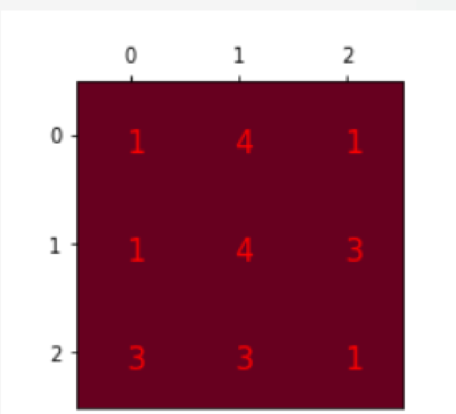
up-sampling

interpolation

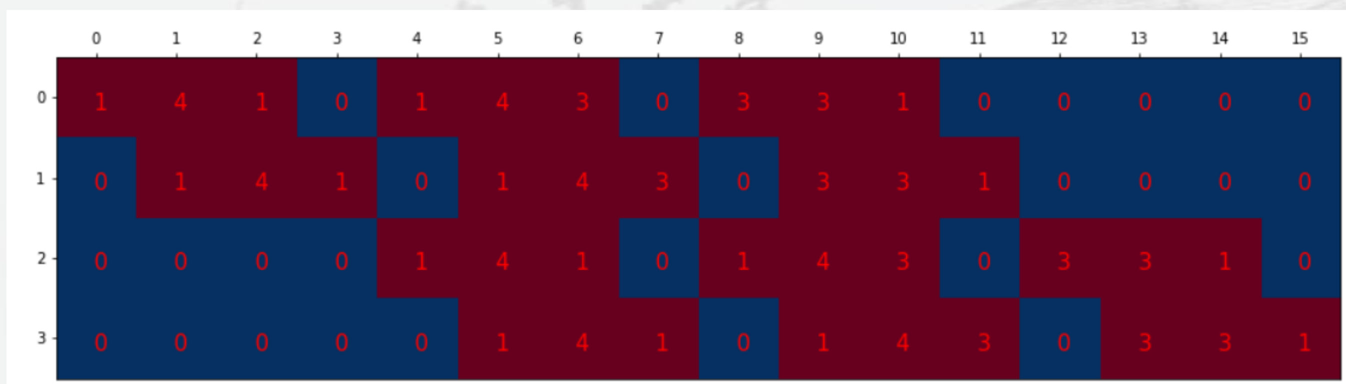




Input image: 4*4

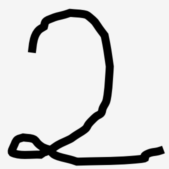


Kernel (3, 3)



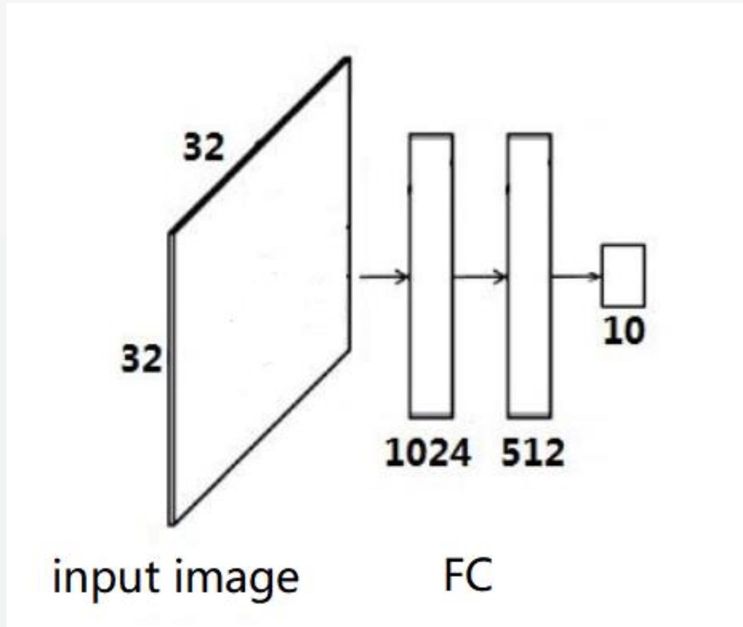
Convolution Matrix (4, 16)

Handwriting digitals classifier



32 * 32

0/1 matrix

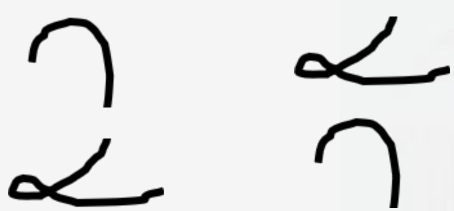


32*32*1024+
1024*512+
512*10+
(1024+512)(bias)

3M

We do not care about the whole image

Just focus on the important feature



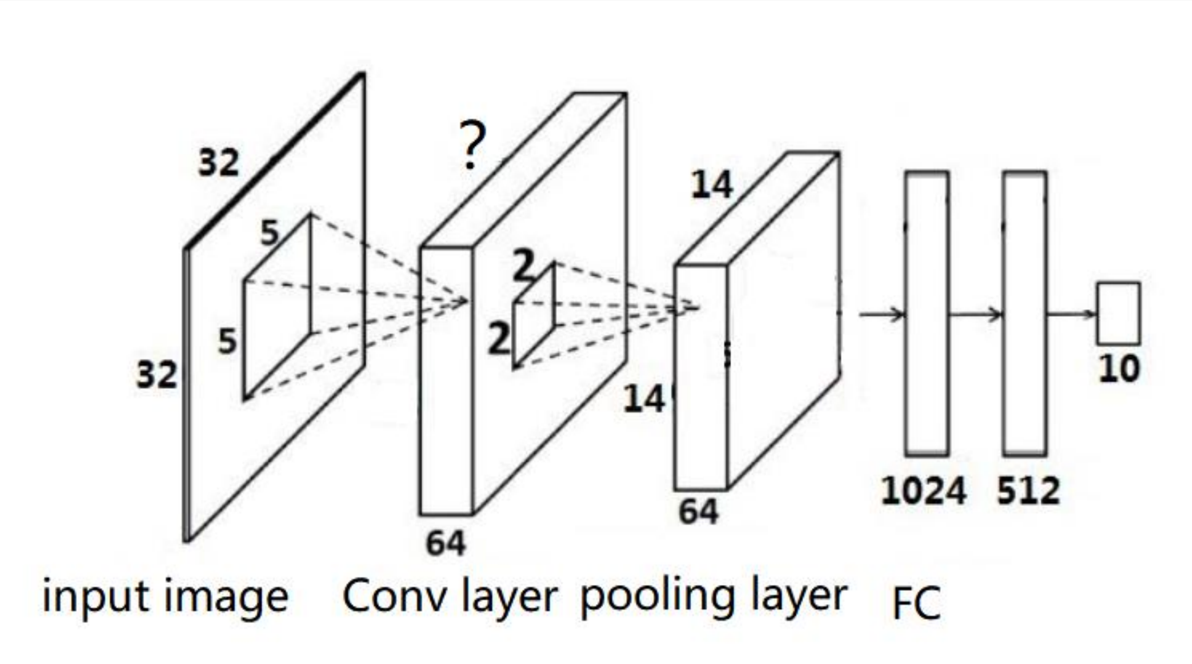
- the part of the image
- the relative position of the part

- Little correlation

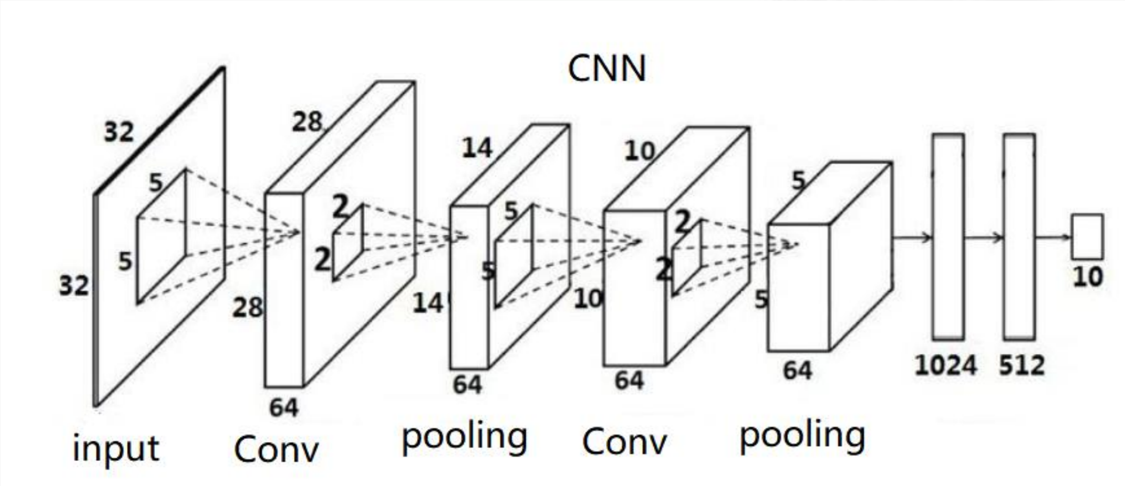
Handwriting digitals classifier

Conv layer
Extract feature

Pooling layer
Sample



cascade of classifiers



- Conv layer
- Pooling layer
- FC

Coding

VGG

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Table 2: **Number of parameters** (in millions).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

- Simple
  - 3*3 conv kernel
  - 2*2 max pooling

- More 3*3 is better

- The deeper, the better?

Residual Network

- Is learning better networks as easy as stacking more layers?

  Vanishing/ exploding gradient

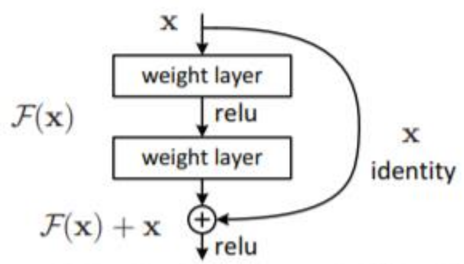


Figure 2. Residual learning: a building block.

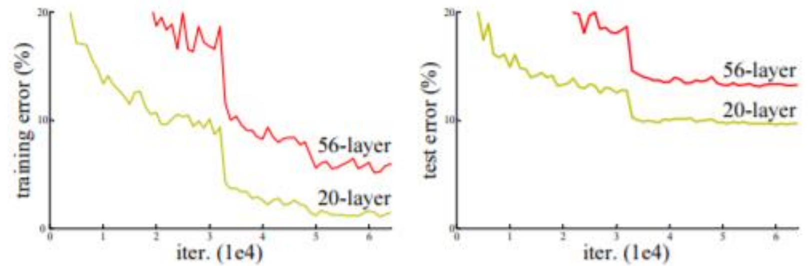


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

西安交通大學
XI'AN JIAOTONG UNIVERSITY

Thank you!

Wan Herun
Xi'an Jiaotong University
wanherun@stu.xjtu.edu.cn