6. Convolutional Neural Networks

CS 519 Deep Learning, Winter 2017
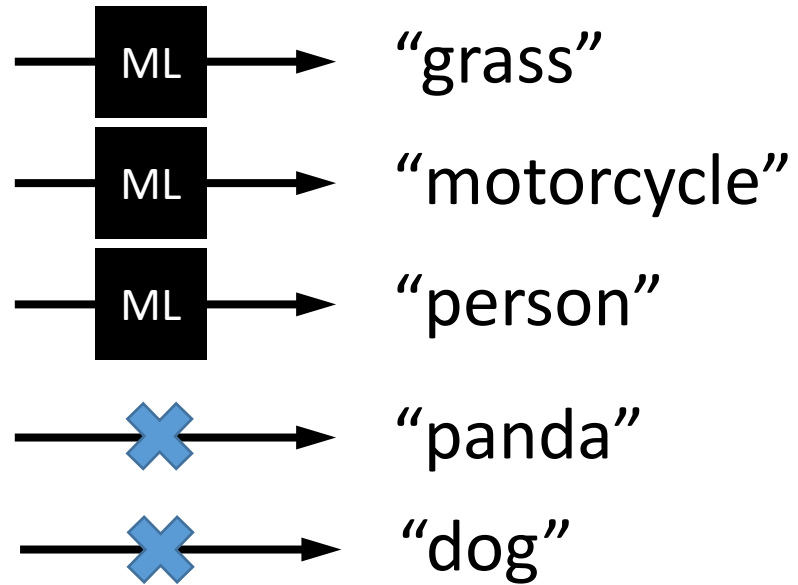
Fuxin Li

With materials from Zsolt Kira

Quiz coming up…

- Next Thursday (2/2)
- 20 minutes
- Topics:
  - Optimization
  - Basic neural networks
- No Convolutional nets in this quiz

The Image Classification Problem

(Multi-label in principle)

ML → "grass"

ML → "motorcycle"

ML → "person"

✖ → "panda"

✖ → "dog"

Neural Networks

- Extremely high dimensionality!

- 256x256 image has already 65,536 * 3 dimensions

- One hidden layer with 500 hidden units require 65,536 * 3 * 500 connections (98 Million parameters)
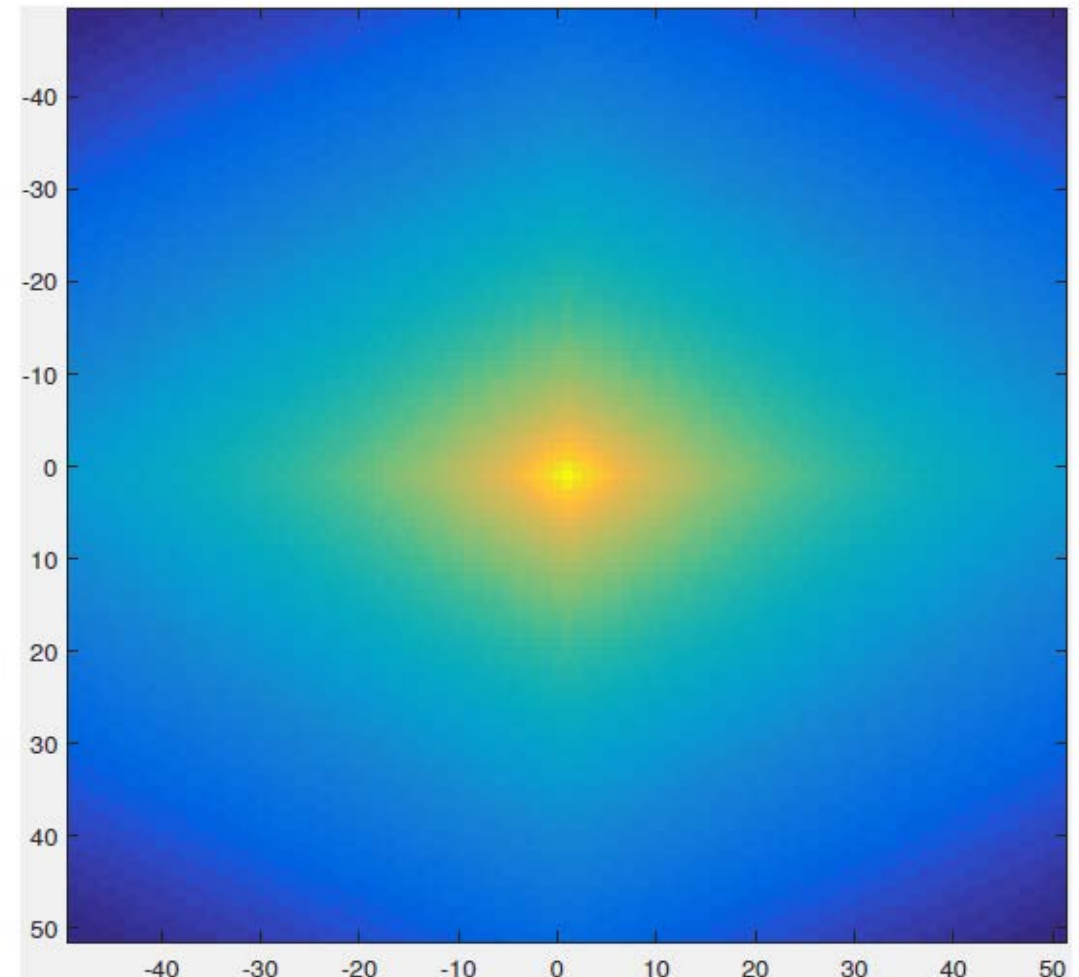
Challenges in Image Classification

Structure between neighboring pixels in natural images

The correlation prior for horizontal and vertical shifts (averaged over 1000 images) looks like this:

Takeaways:
1) Long-range correlation
2) Local correlation stronger than non-local

The convolution operator

Sobel filter

-1	0	+1
-2	0	+2
-1	0	+1

Gx

+1	+2	+1
0	0	0
-1	-2	-1

Gy

Convolution

H11	H12	H13
H21	H22	H23
H31	H32	H33

$*$

F11	F12	F13	F14	F15	F16
F21	F22	F23	F24	F25	F26
F31	F32	F33	F34	F35	F36
F41	F42	F43	F44	F45	F46
F51	F52	F53	F54	F55	F56
F61	F62	F63	F64	F65	F66

$=$

G11	G12	G13	G14	G15	G16
G21	G22	G23	G24	G25	G26
G31	G32	G33	G34	G35	G36
G41	G42	G43	G44	G45	G46
G51	G52	G53	G54	G55	G56
G61	G62	G63	G64	G65	G66

I

Convolution

$I * Gx$

2D Convolution with Padding

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

*

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

=

2D Convolution with Padding

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

*

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

=

2		

$$3 \times 1 + (-1) \times 1 = 2$$

2D Convolution with Padding

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

*

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

=

2	-1	

$$1 \times (-2) + 1 \times 1 + 1 \times (-1) + 1 \times 1 = -1$$

2D Convolution with Padding

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

*

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

=

2	-1	-6

What if:

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

*

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

=

2	-1	-18

2D Convolution with Padding

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

*

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

=

2	-1	-6
4		

2D Convolution with Padding

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

*

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

=

2	-1	-6
4	-3	

2D Convolution with Padding

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

*

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

=

2	-1	-6
4	-3	-5

2D Convolution with Padding

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

*

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

=

2	-1	-6
4	-3	-5
1		

2D Convolution with Padding

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

*

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

=

2	-1	-6
4	-3	-5
1	-2	

2D Convolution with Padding

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

*

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

=

2	-1	-6
4	-3	-5
1	-2	-2

Filter size and Input/Output size

m

m

Filter

N

N

Input

$N\text{-}m\text{+}1$

$N\text{-}m\text{+}1$
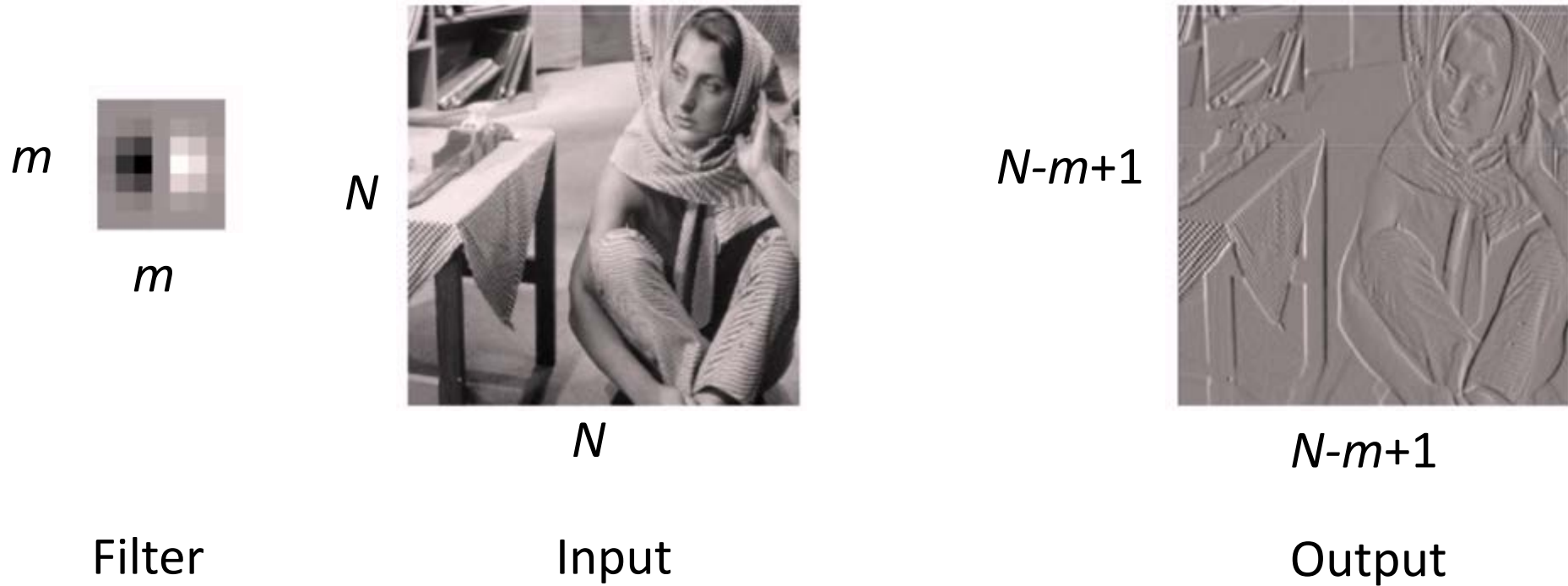
Output

- Zero padding the input so that the output is NxN

Location-invariance in images

- Image Classification
  - It does not matter where the object appears

- Object Localization
  - It does matter where the object appears
    - (Deconvolution – to be dealt with later)
  - But the rules for recognizing object are the same everywhere in the image

Convolutional Networks

- Each connection is a convolution followed by ReLU

$$\text{ReLU}(I * f_6)$$

$$\text{ReLU}(I * f_1)$$
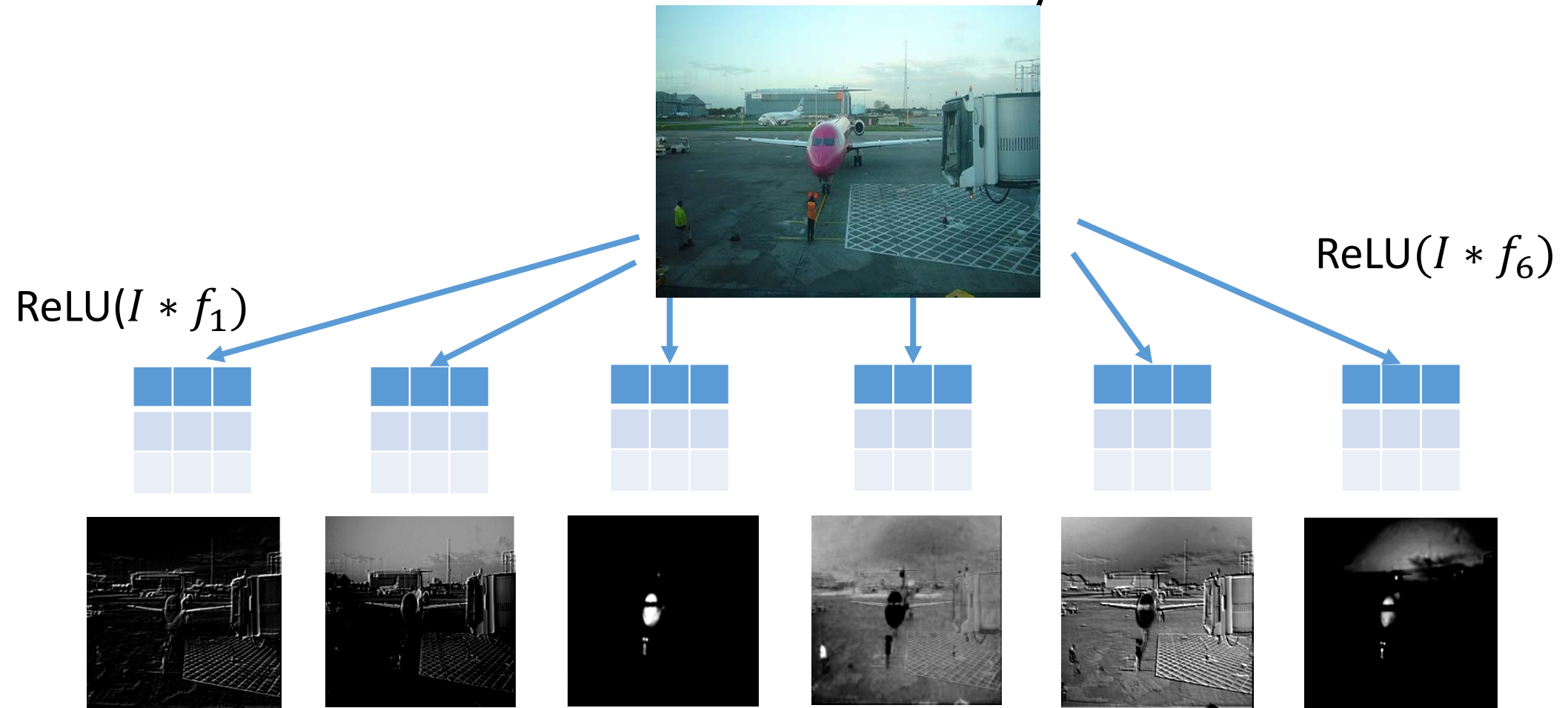
CNN: Multi-layer Architecture

- Multi-layer architecture helps to generate more complicated templates

Image

2nd level

Corner1	Edge1	Corner2
Edge2	Center	Edge2
Corner3	Edge1	Corner4

Circle Detector

21
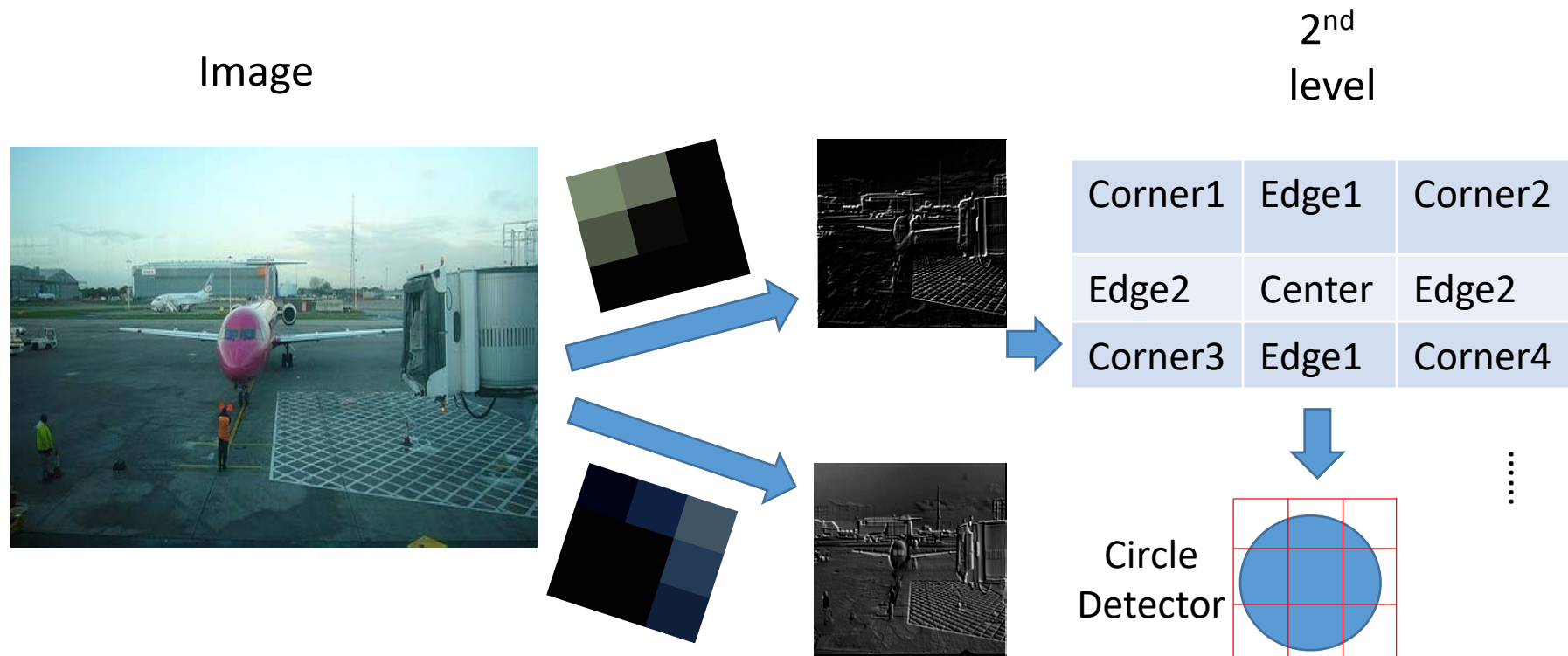
Convolutional Networks 2ⁿᵈ layer

- Each connection is a convolution

3x3x3

+ReLU

e.g. 64 filters

Convolution

3x3x64

Note different
dimensionality
for each filter
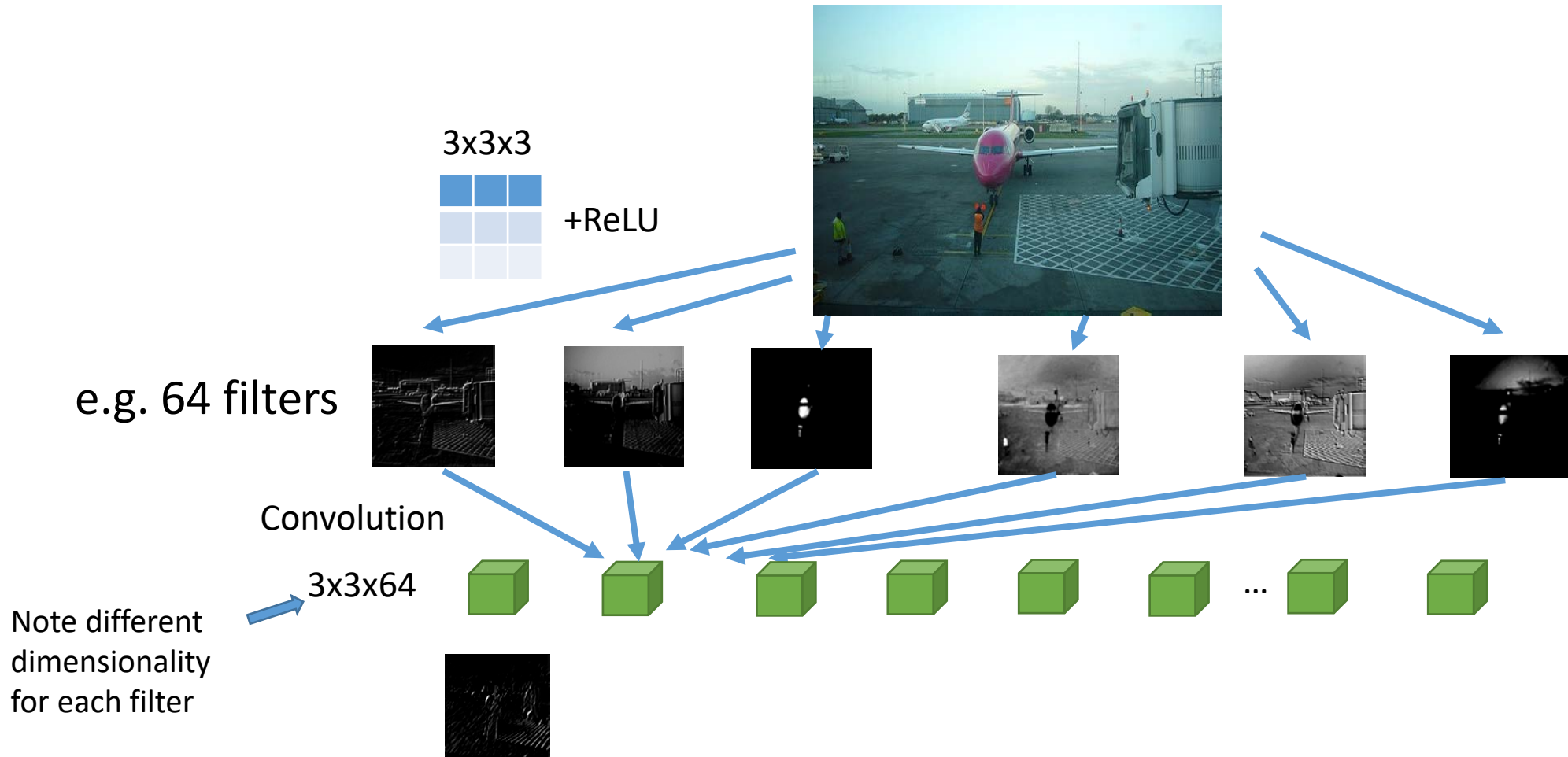
What's the shape of weights and input

- e.g. 64 filters level 1
  - 128 filters level 2

Input
224 x 224 x 3

3x3x3x64

Weights   +ReLU

Output1:
224 x 224 x 64

Convolution

3x3x64x128

...

Output1:
224 x 224 x 128
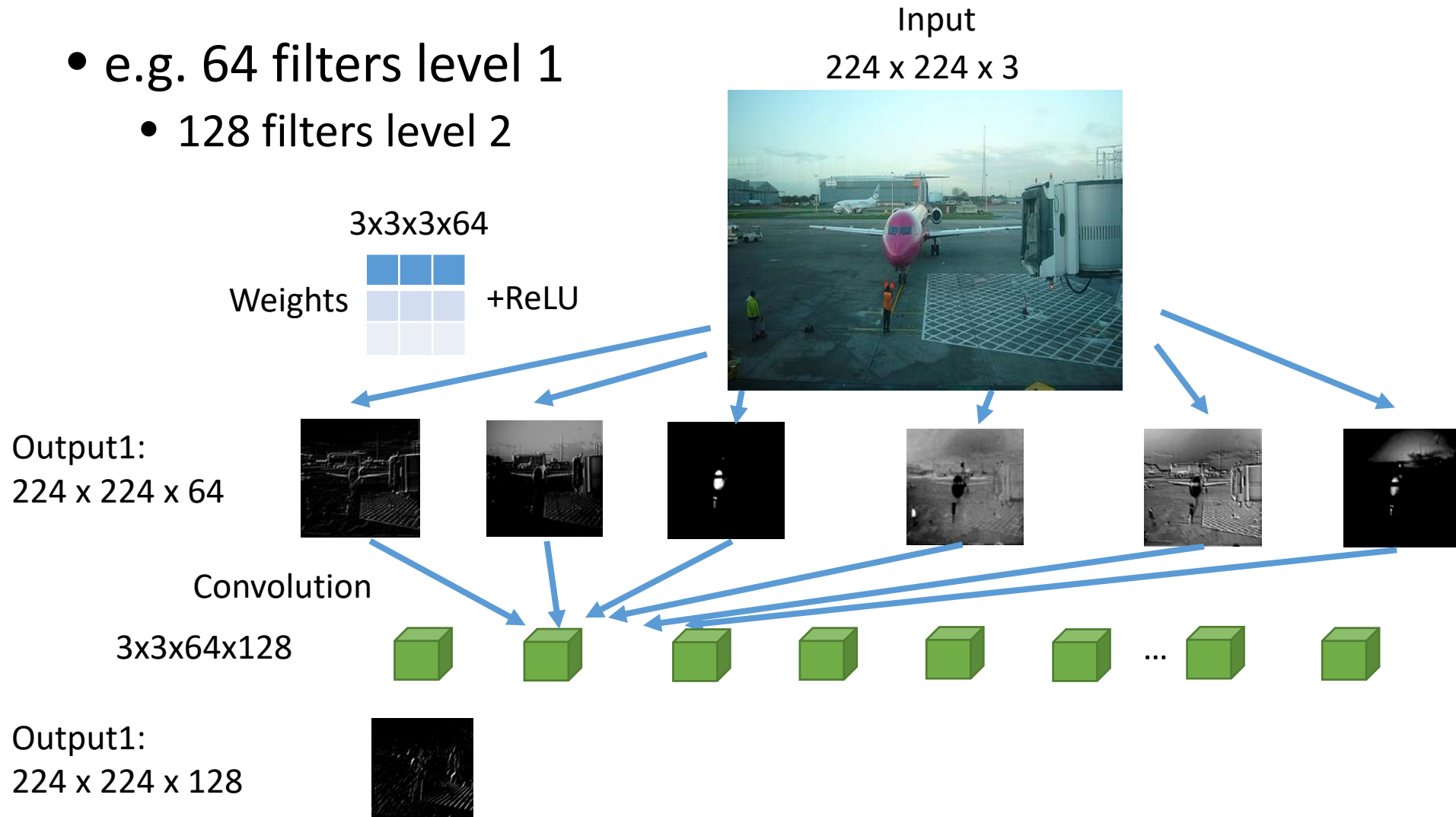
Dramatic reduction on the number of parameters

- Think about a fully-connected network on 256 x 256 image with 500 hidden units and 10 classes
  - Num. of params = 65536 * 3 * 500 + 500 * 10 = 98.3 Million
- 1-hidden layer convolutional network on 256 x 256 image with 11x11 and 500 hidden units?
  - Num. of params = 11 * 11 * 3 * 500 + 500 * 10 = 155,000
- 2-hidden layers convolutional network on 256 x 256 image with 11x11 – 3x3 and 500 hidden units in each layer?
  - Num. of params = 150,000 + 3 * 3 * 500 * 500 + 500 * 10 = 2.4 Million

Back to images

- Why images are much harder than digits?
- Much more deformation
- Much more noises
- Noisy background

Pooling

- Localized max-pooling (stride-2) helps achieving some location invariance

- As well as filtering out irrelevant background information

  e.g. $x_{out} = \max(x_{11}, x_{12}, x_{21}, x_{22})$

- What is the subgradient of this?

max pooling

20	30
112	37

12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

average pooling

13	8
79	20
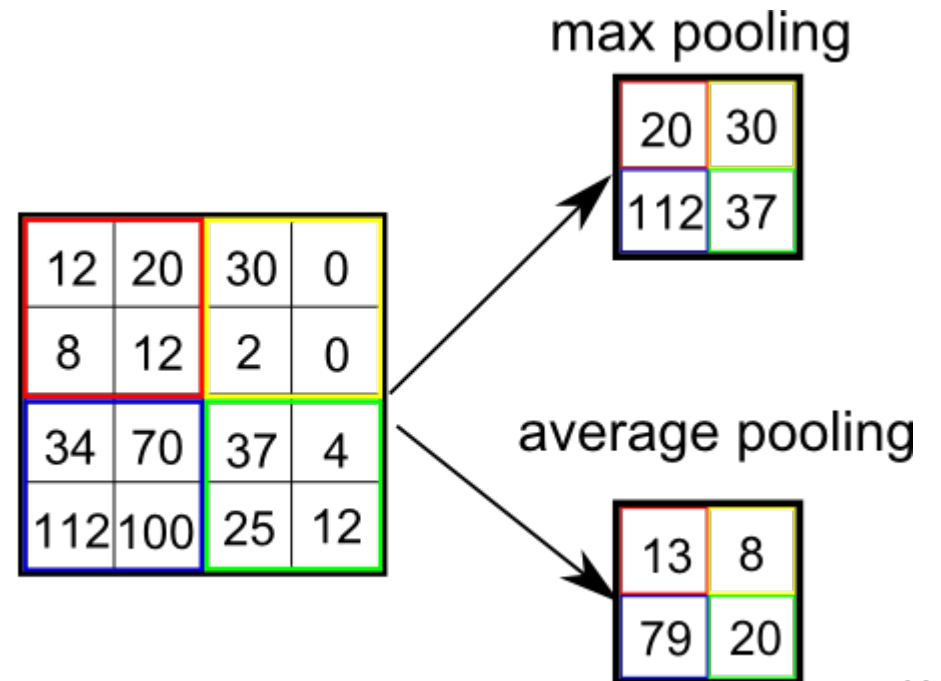
Deformation enabled by max-pooling

max pooling

20	30
112	37

12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

New filter in
the next layer
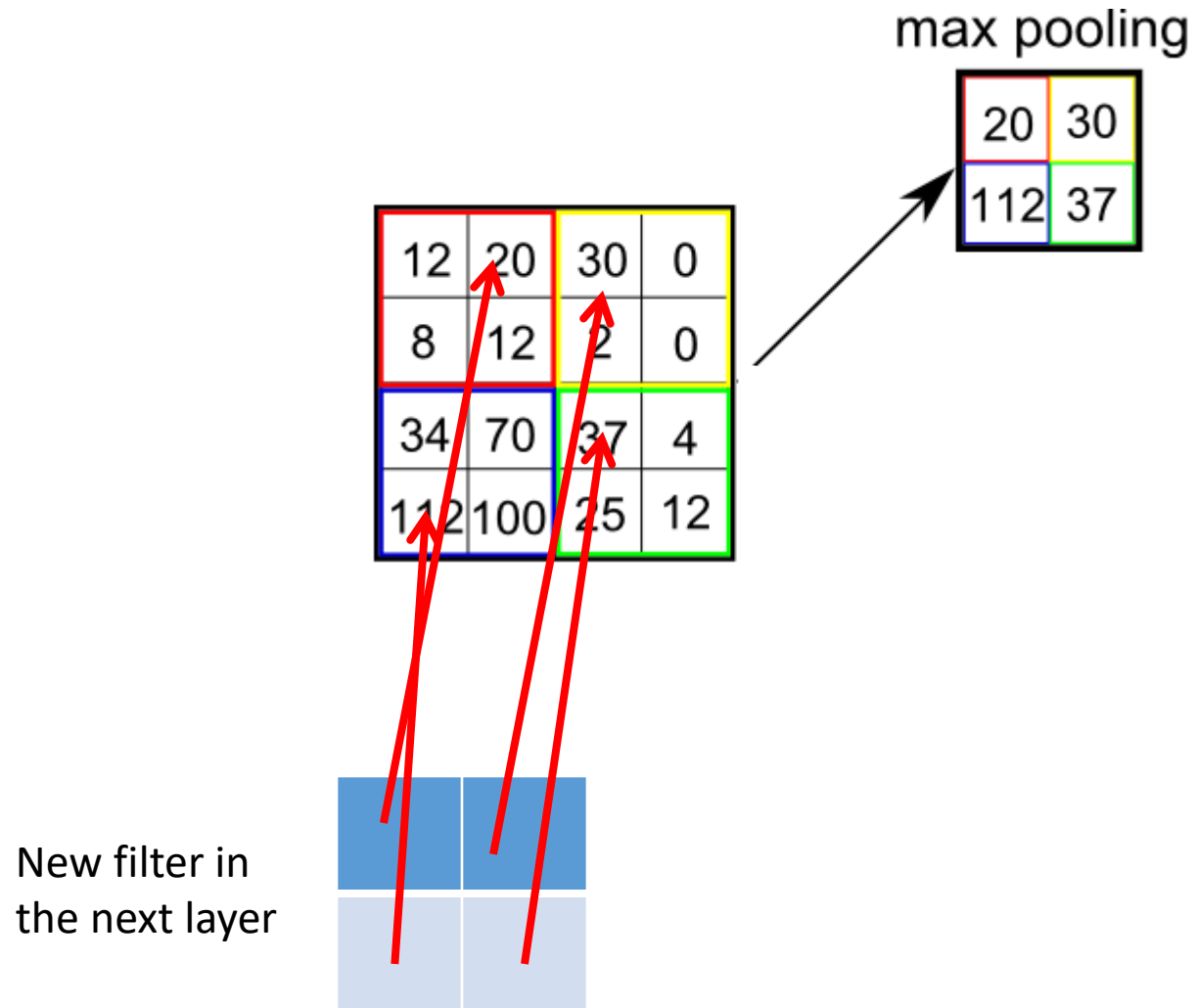
Strides

- Reduce image size by strides
    - Stride = 1, convolution on every pixel
    - Stride = 2, convolution on every 2 pixels
    - Stride = 0.5, convolution on every half pixel (interpolation, Long et al. 2015)

Stride = 2 →

The VGG Network

224 x 224

224 x 224

112 x 112

56 x 56

28 x 28

14 x 14

7 x 7

Airplane   Dog   Car   ......   SUV   Minivan   Sign   Pole

image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
FC-1000
softmax

← 64 filters

← 128 filters

Fully connected
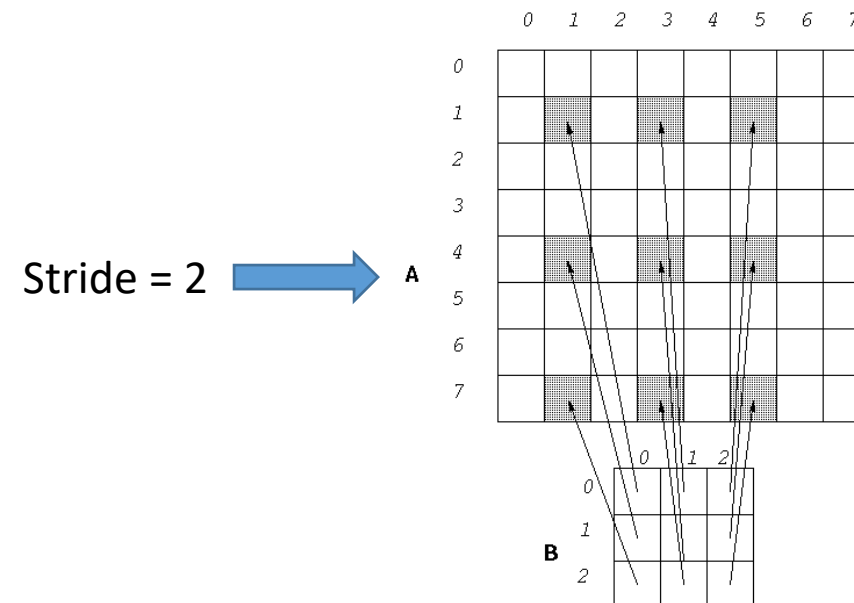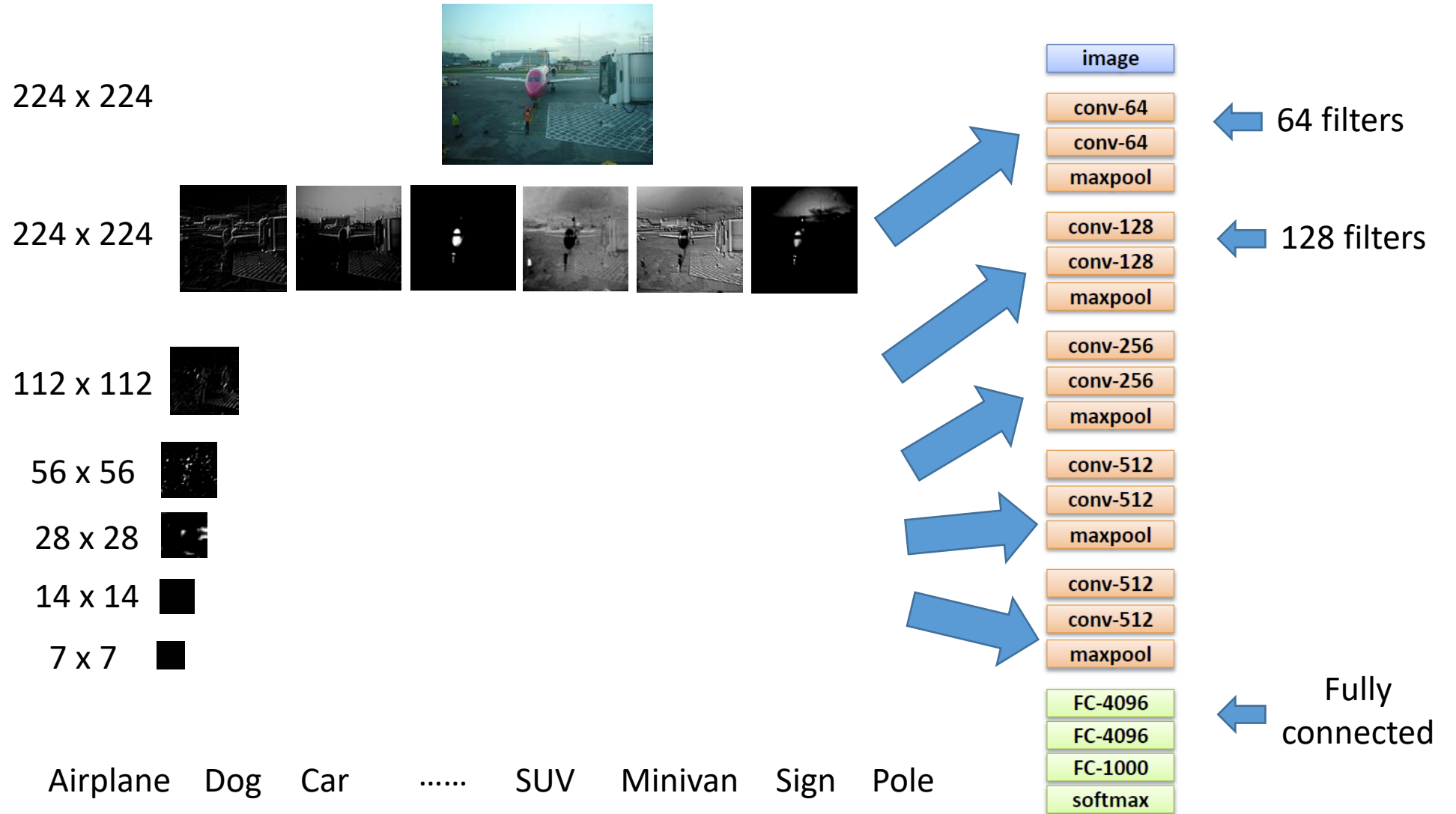
(Simonyan and Zisserman 2014)

Why 224x224?

- The magic number $224 = 2^5 \times 7$, so that there is always a center-surround pattern in any layer

- Another potential candidate is $2^7 \times 3 = 384$
  - Some has shown larger is better
  - However more layers + bigger = more difficult to train, need more machines to tune parameters

Backpropagation for the convolution operator

Forward pass:

Compute $f(X; W) = X * W$

Backward pass:

Compute $\dfrac{\partial Z}{\partial X} = ?$

$\dfrac{\partial Z}{\partial W} = ?$
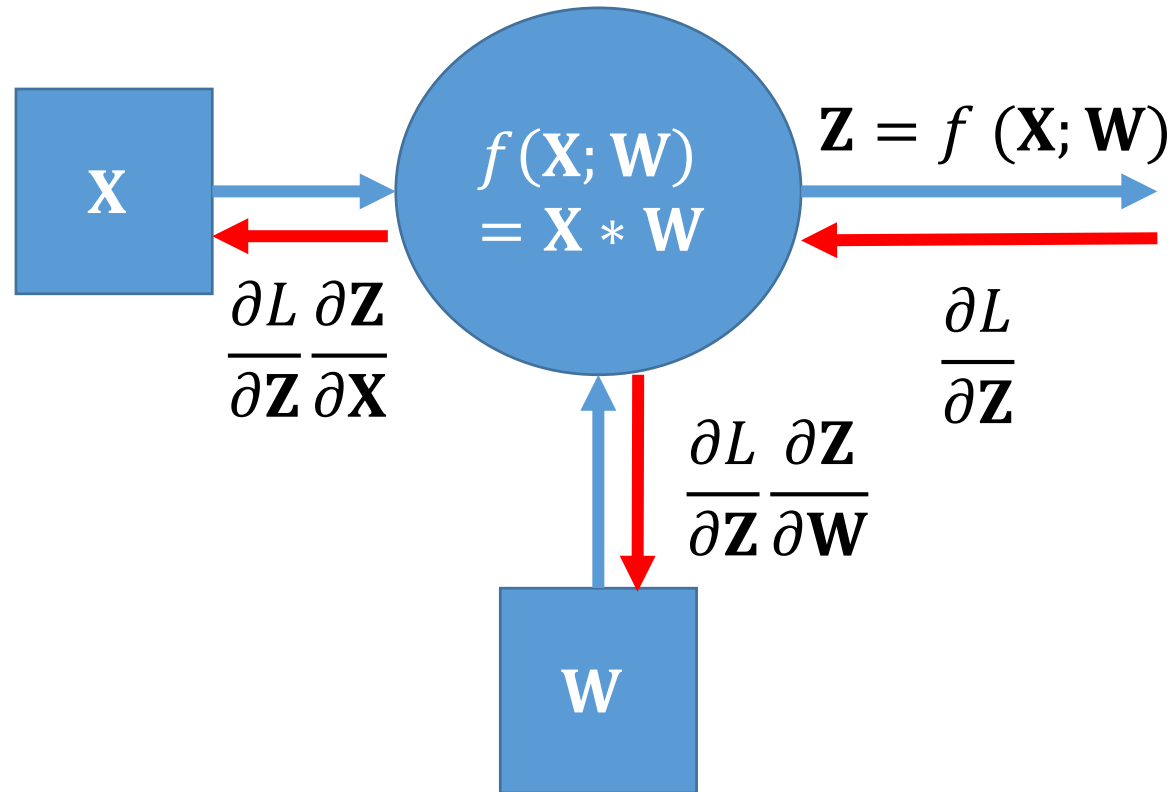
Historical Remarks: MNIST

Le Net

- Convolutional nets are invented by Yann LeCun et al. 1989
  - On handwritten digits classification
  - Many hidden layers
  - Many maps of replicated units in each layer.
  - Pooling of the outputs of nearby replicated units.
  - A wide net that can cope with several characters at once even if they overlap.
  - A clever way of training a complete system, not just a recognizer.
- This net was used for reading ~10% of the checks in North America.
- Look the impressive demos of LENET at http://yann.lecun.com

The architecture of LeNet5 (LeCun 1998)
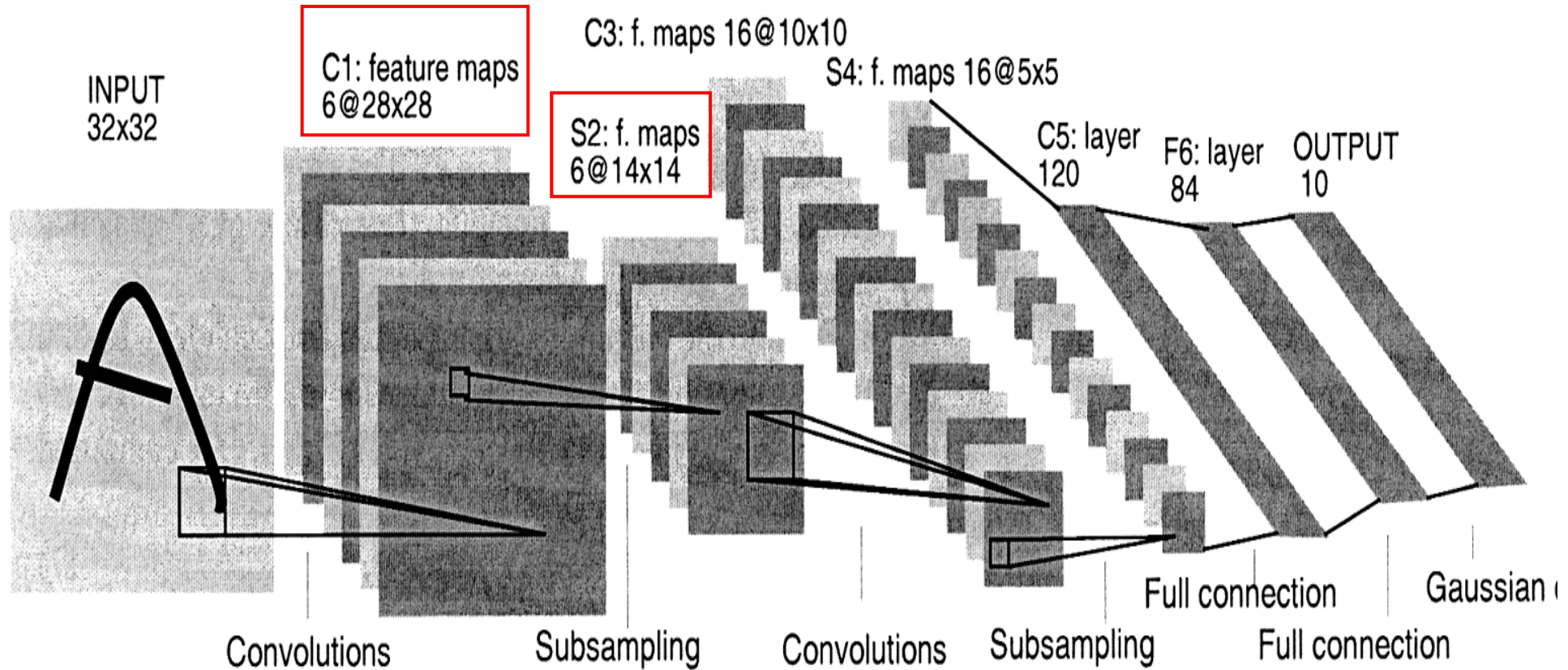
ConvNets performance on MNIST
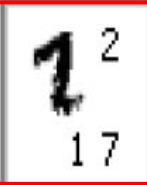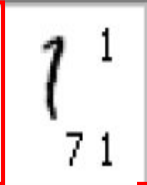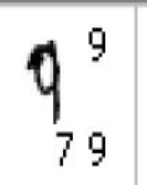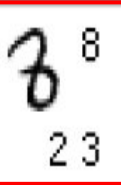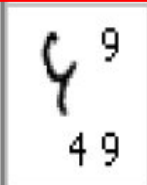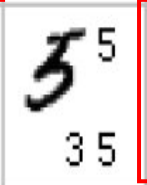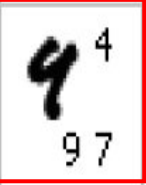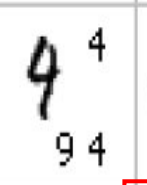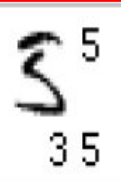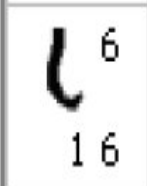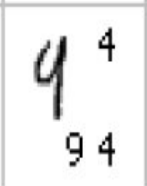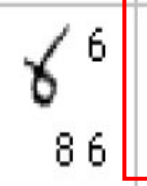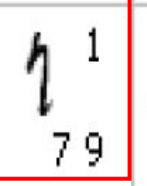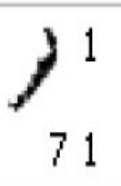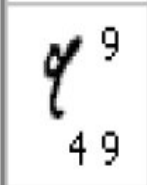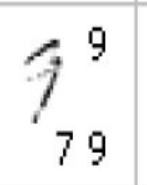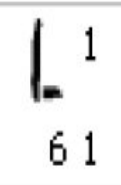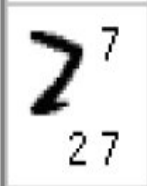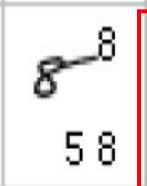
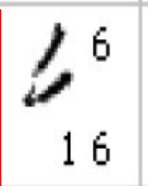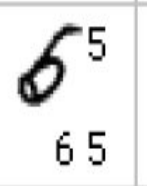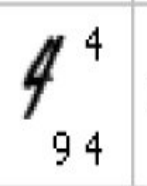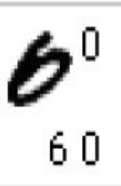|			
Convolutional net LeNet-1	subsampling to 16x16 pixels	1.7	LeCun et al. 1998
Convolutional net LeNet-4	none	1.1	LeCun et al. 1998
Convolutional net LeNet-4 with K-NN instead of last layer	none	1.1	LeCun et al. 1998
Convolutional net LeNet-4 with local learning instead of last layer	none	1.1	LeCun et al. 1998
Convolutional net LeNet-5, [no distortions]	none	0.95	LeCun et al. 1998
Convolutional net, cross-entropy [elastic distortions]	none	0.4	Simard et al., ICDAR 2003

The 82 errors made by LeNet5

The human error rate is probably about 0.2% - 0.3% (quite clean)

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

The errors made by the Ciresan *et. al.* net

The top printed digit is the right answer. The bottom two printed digits are the network's best two guesses.

The right answer is almost always in the top 2 guesses.

With model averaging they can now get about 25 errors.

What's different from back then till now

- Computers are bigger, faster
- GPUs

	LeNet (1989)	LeNet (1998)	AlexNet (2012)
classification task	digits	digits	objects
categories	10	10	1,000
image size	16×16	28×28	$256 \times 256 \times 3$
training examples	7,291	60,000	1.2 million
units	1,256	8,084	658,000
parameters	9,760	60,000	60 million
connections	65,000	344,000	652 million
total operations	11 billion	412 billion	200 quadrillion (est.)

What else is different?

- ReLU rectifier

- Max-pooling
  - Grab local features and make them global
- Dropout regularization (to-be-discussed)
  - Replaceable by some other regularization techniques

ReLU vs. Sigmoid