Convolutional Neural Networks

Computer vision

deeplearning.ai

Computer Vision Problems

Image Classification

64x64

Cat? (0/1)
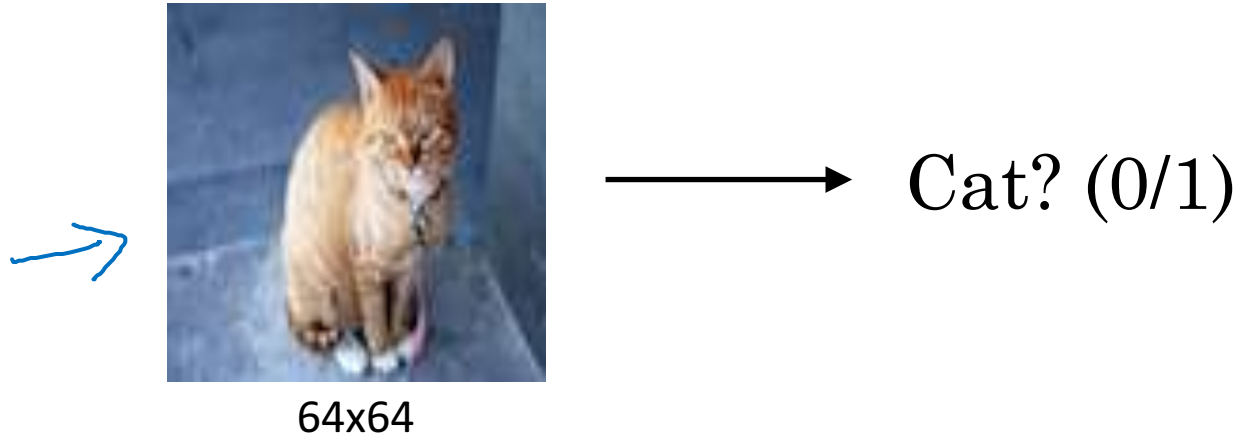
Object detection

Neural Style Transfer

Andrew Ng

Deep Learning on large images

Cat? (0/1)

64x64 $\times 3$

12288

$1000 \times 1000 \times 3$
$= 3$ million

$x \in \mathbb{R}^{3M}$

$W^{[1]}$ $(1000, 3m)$

3 billion

x_1
x_2
$\vdots$
x_n

$\hat{y}$

$\curvearrowleft 3M$ $\curvearrowleft 1000$

Andrew Ng
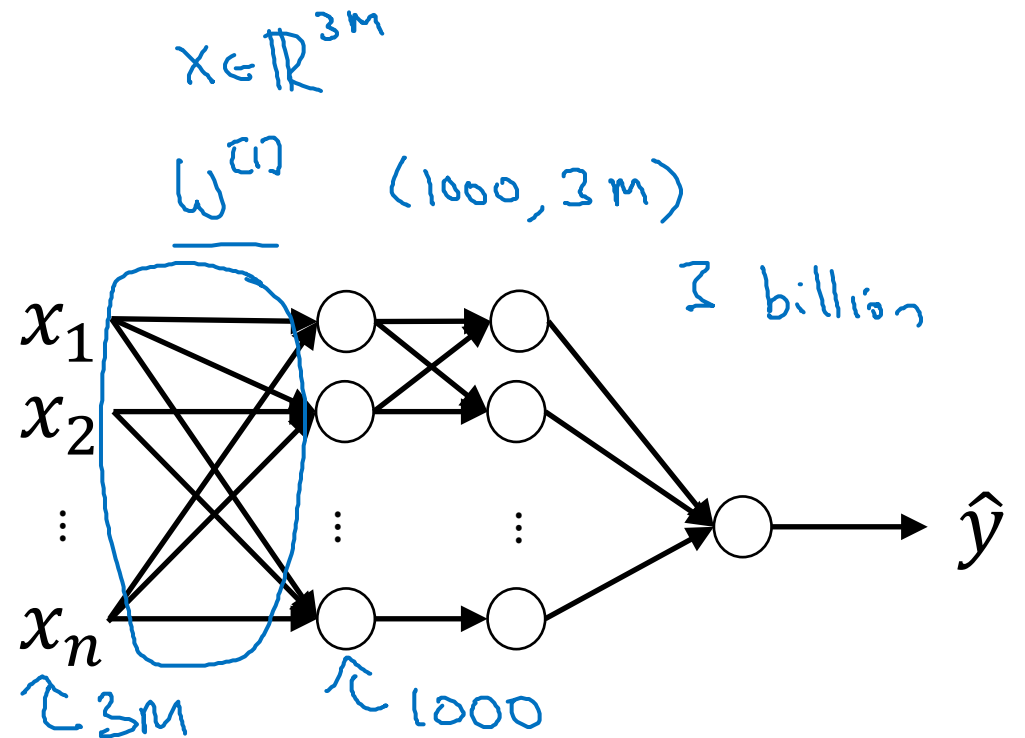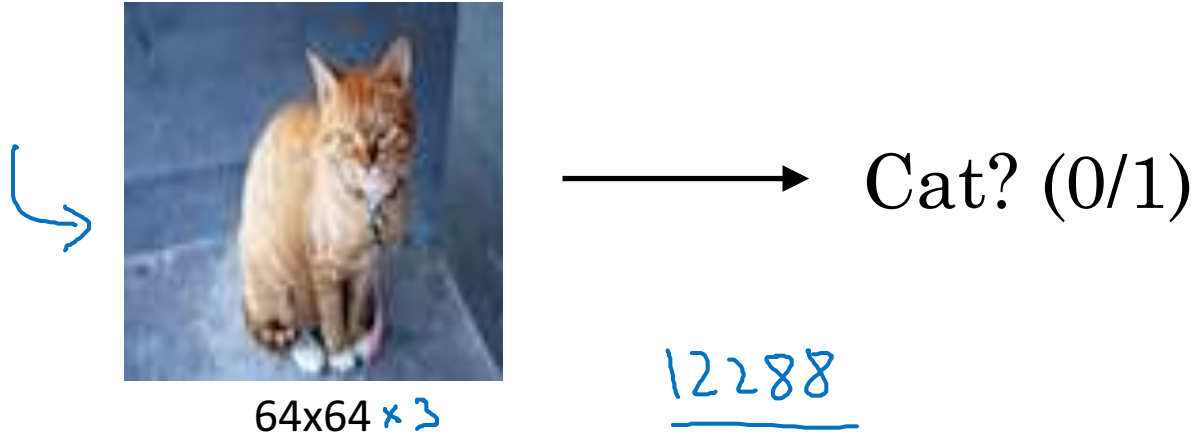
deeplearning.ai

Convolutional
Neural Networks

Edge detection
example

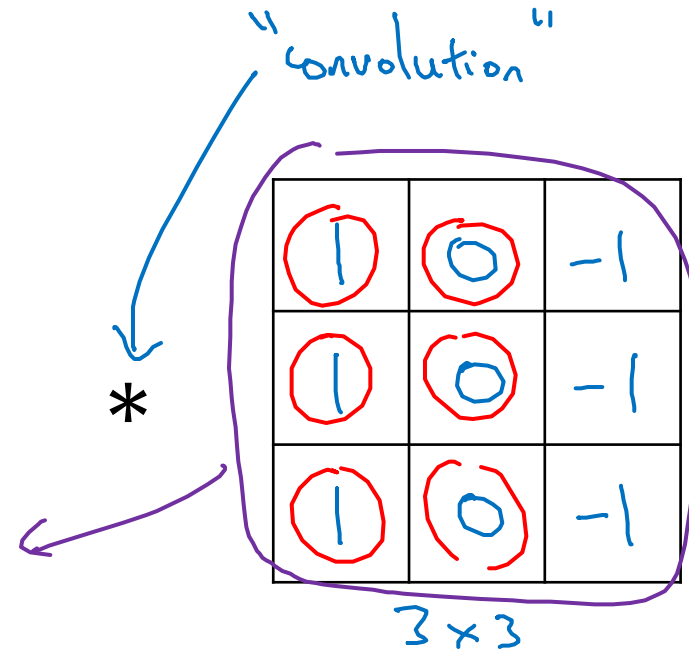# Computer Vision Problem

vertical edges

horizontal edges

Andrew Ng
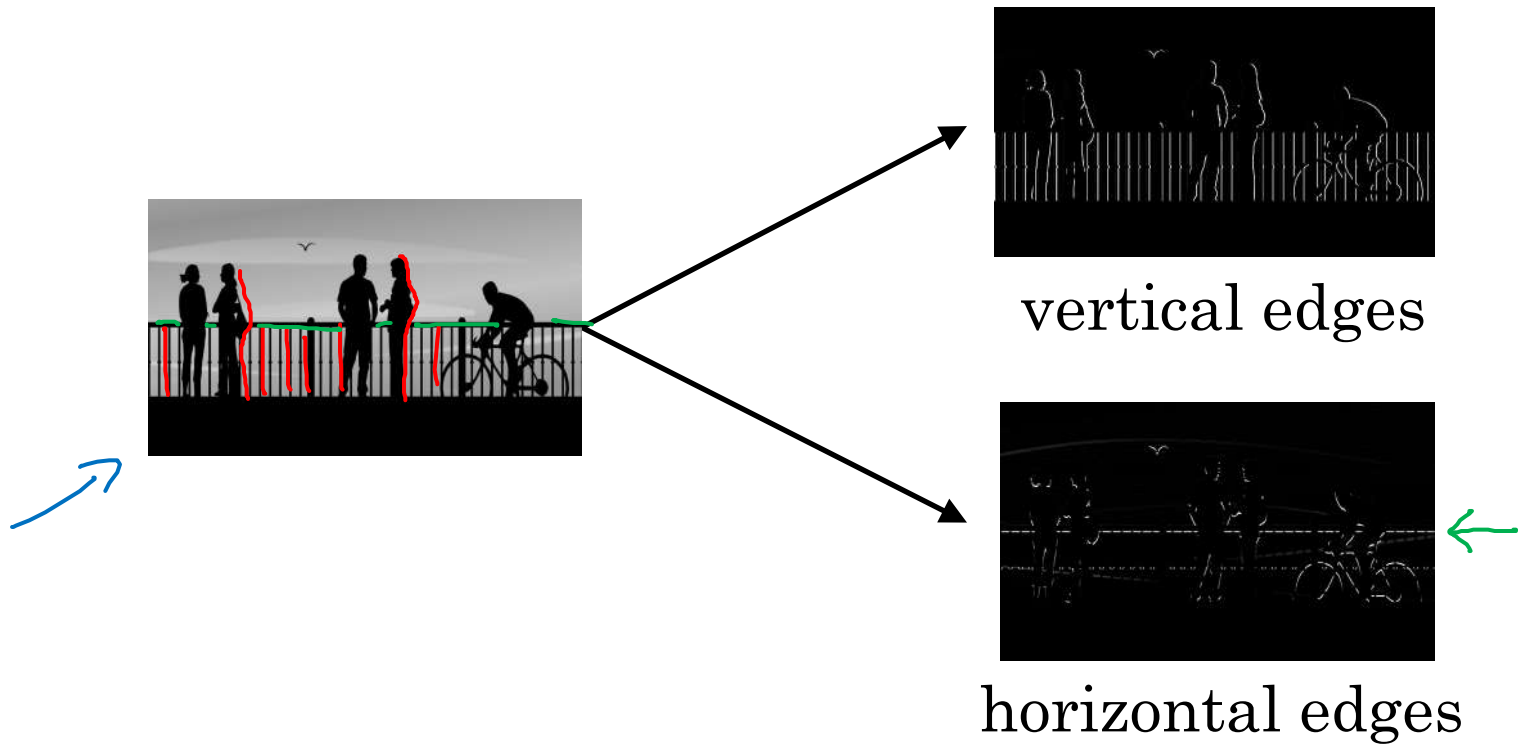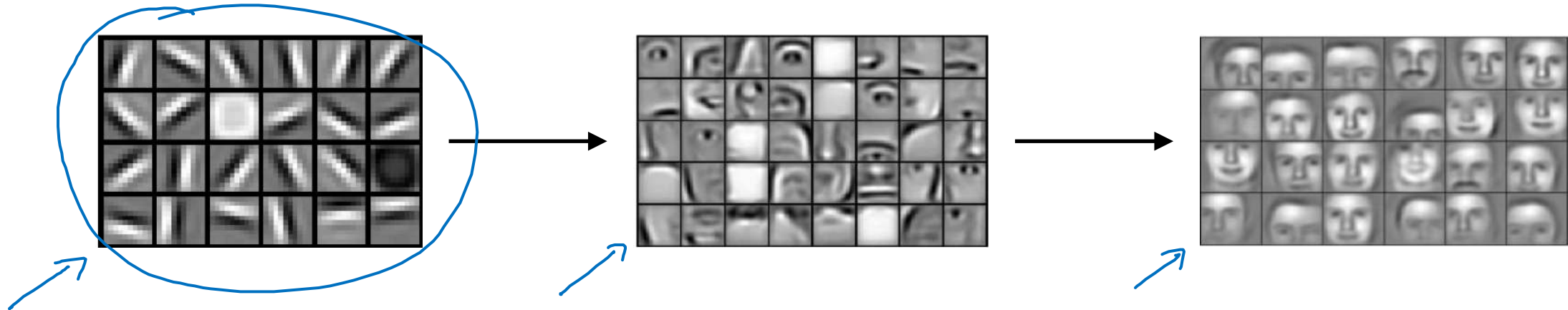
Vertical edge detection

$3 \times 1 + 1 \times 1 + 2 \times 1 + 0 \times 0 + 5 \times 0 + 7 \times 0 + 1 \times 7 + 8 \times -1 + 2 \times -1 = -5$

6×6 "convolution" 3×3 $\rightarrow$ filter kernel 4×4

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

*

1	0	-1
1	0	-1
1	0	-1

=

-5	-4	0	8
-10	-2	2	3
0	-2	-4	-7
-3	-2	-3	-16

Andrew Ng

Vertical edge detection

Andrew Ng

deeplearning.ai

Convolutional Neural Networks

More edge detection

Vertical edge detection examples

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

$*$

1	0	-1
1	0	-1
1	0	-1

$=$

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

$*$

1	0	-1
1	0	-1
1	0	-1

$=$

0	-30	-30	0
0	-30	-30	0
0	-30	-30	0
0	-30	-30	0

Andrew Ng

Vertical and Horizontal Edge Detection

1	0	-1
1	0	-1
1	0	-1

Vertical

1	1	1
0	0	0
-1	-1	-1

Horizontal

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
0	0	0	10	10	10
0	0	0	10	10	10
0	0	0	10	10	10

6 x 6

*

1	1	1
0	0	0
-1	-1	-1

=

0	0	0	0
30	10	-10	-30
30	10	-10	-30
0	0	0	0

Learning to detect edges

1	0	-1
1	0	-1
1	0	-1

↑

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

Sobel filter

↑

3	0	-3
10	0	-10
3	0	-3

Scharr filter

↑

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

convolution

✳

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

3×3

=

$45°$
$70°$
$73°$

Andrew Ng

deeplearning.ai

Convolutional Neural Networks

Padding

Padding

- shrinks output
- throw away into from edge

$*$ 3×3 $f \times f$

$=$

6×6

$p=2$

$\dfrac{6\times6}{n\times n} \rightarrow 8\times8$

$P = padding = \underline{1}$

$n-f+1 \times n-f+1$

$6-3+1=4$

$\rightarrow \dfrac{4\times4}{}$

$n+2p-f+1 \times n+2p-f+1$

$6+2-3+1 \times \_\_ = 6\times6$

Andrew Ng
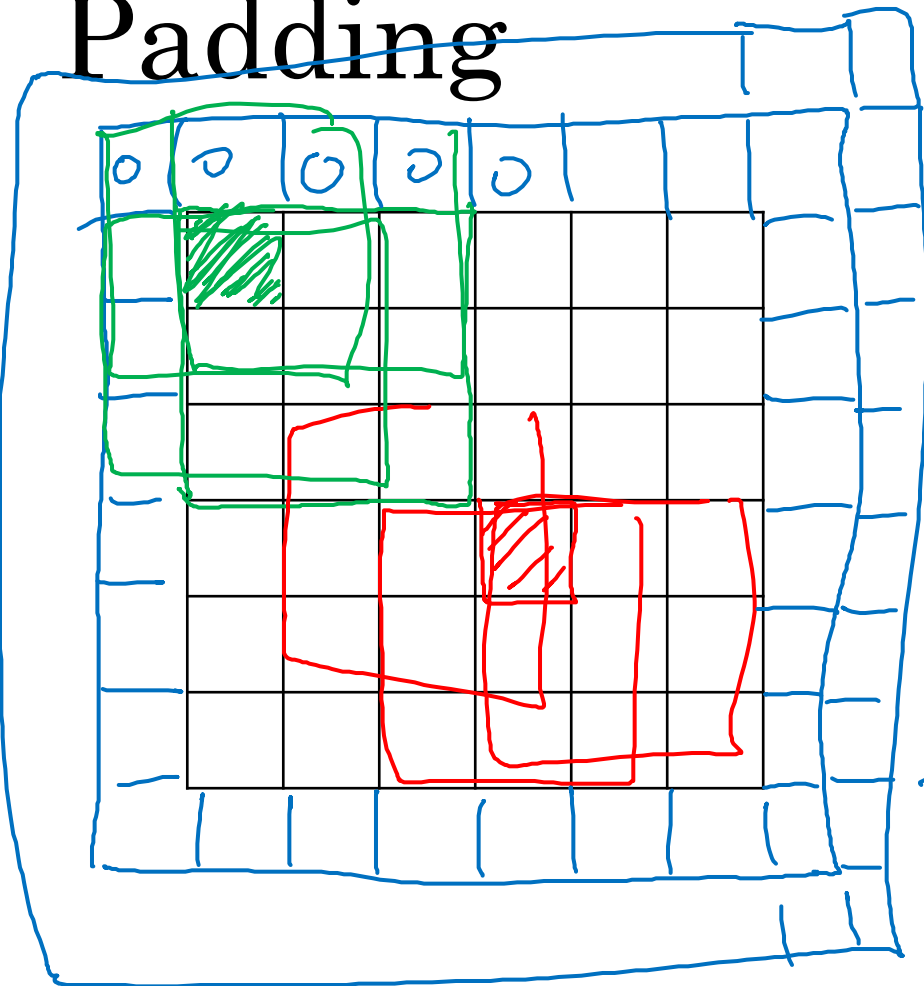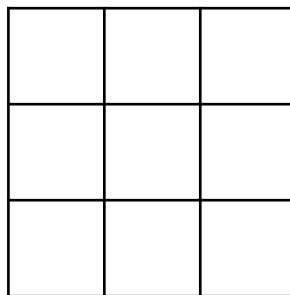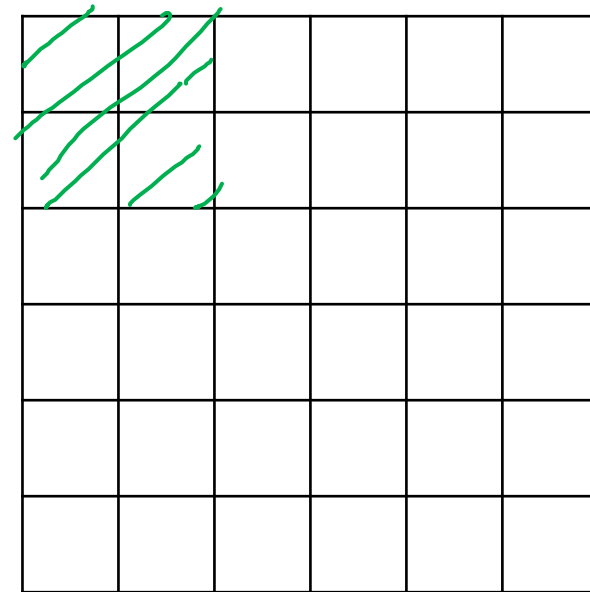
Valid and Same convolutions

$\rightarrow$ no padding
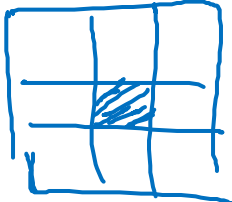
"Valid": $n \times n \quad * \quad f \times f \quad \longrightarrow \quad \underline{n - f + 1} \times n - f + 1$

$6 \times 6 \quad * \quad 3 \times 3 \quad \longrightarrow \quad 4 \times 4$

"Same": Pad so that output size is the same as the input size.

$n + 2p - f + 1 \times n + 2p - f + 1$

f is usually odd

1×1
3×3
5×5
7×7

$n + 2p - f + 1 = n \quad \Rightarrow \quad \boxed{p = \dfrac{f-1}{2}}$

$3 \times 3 \qquad p = \dfrac{3-1}{2} = 1 \quad \bigg| \quad 5 \times 5 \qquad p = 2$

$f = 5$
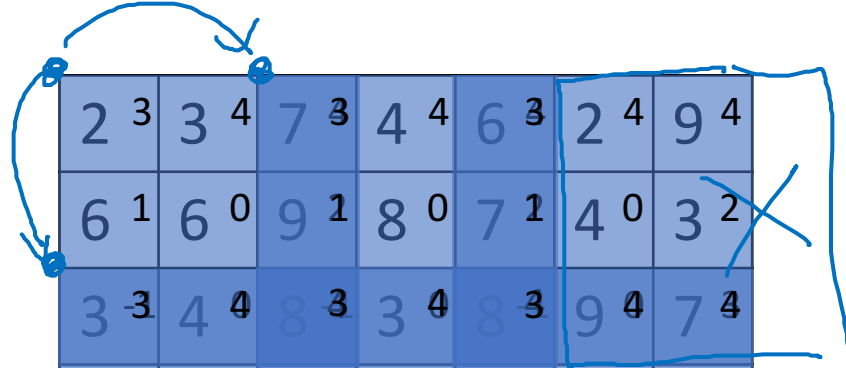
Convolutional
Neural Networks

Strided
convolutions

deeplearning.ai

Strided convolution

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

7×7

$*$

3	4	4
1	0	2
-1	0	3

3×3

$=$

91	100	83
69	91	127
44	72	74

3×3

Stride $= 2$

$\lfloor z \rfloor = floor(z)$

$n \times n \quad * \quad f \times f$

padding p stride s

$s = 2$

$$\left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n + 2p - f}{s} + 1 \right\rfloor$$

$$\frac{7 + 0 - 3}{2} + 1 = \frac{4}{2} + 1 = 3$$

Summary of convolutions

$n \times n$ image $\qquad$ $f \times f$ filter

padding p $\qquad$ stride s

Output Size:

$$\left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor \quad \times \quad \left\lfloor \frac{n+2p-f}{s} + 1 \right\rfloor$$

Technical note on cross-correlation vs. convolution

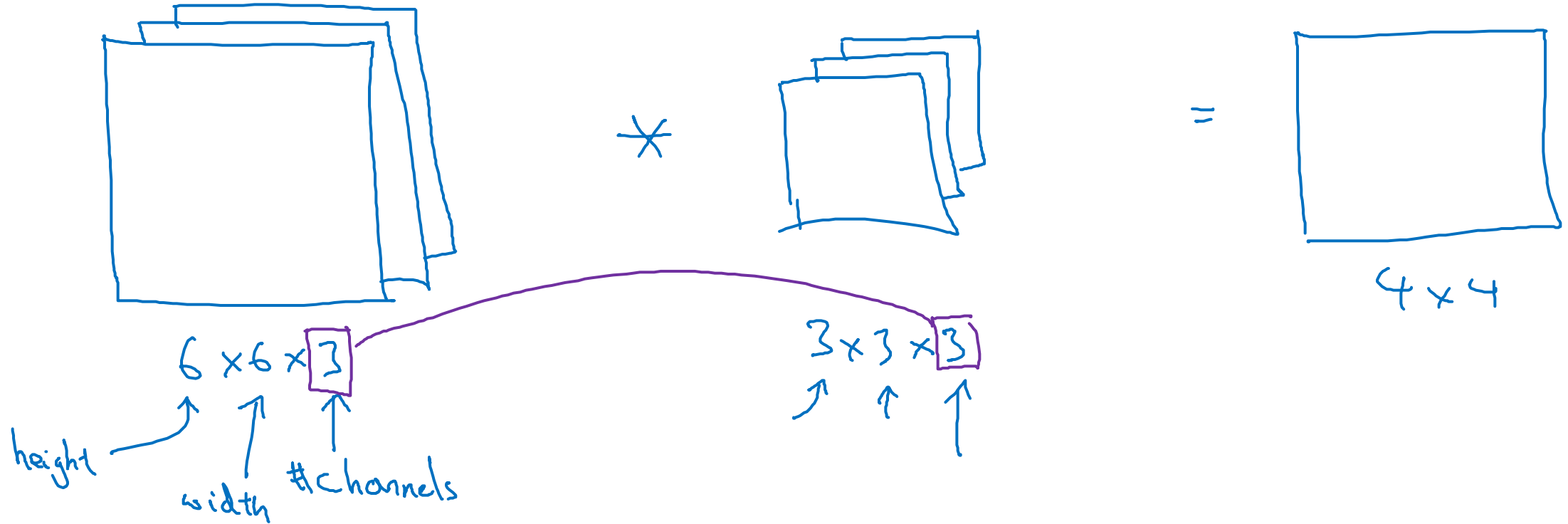Convolution in math textbook:

$$(A * B) * C = A * (B * C)$$
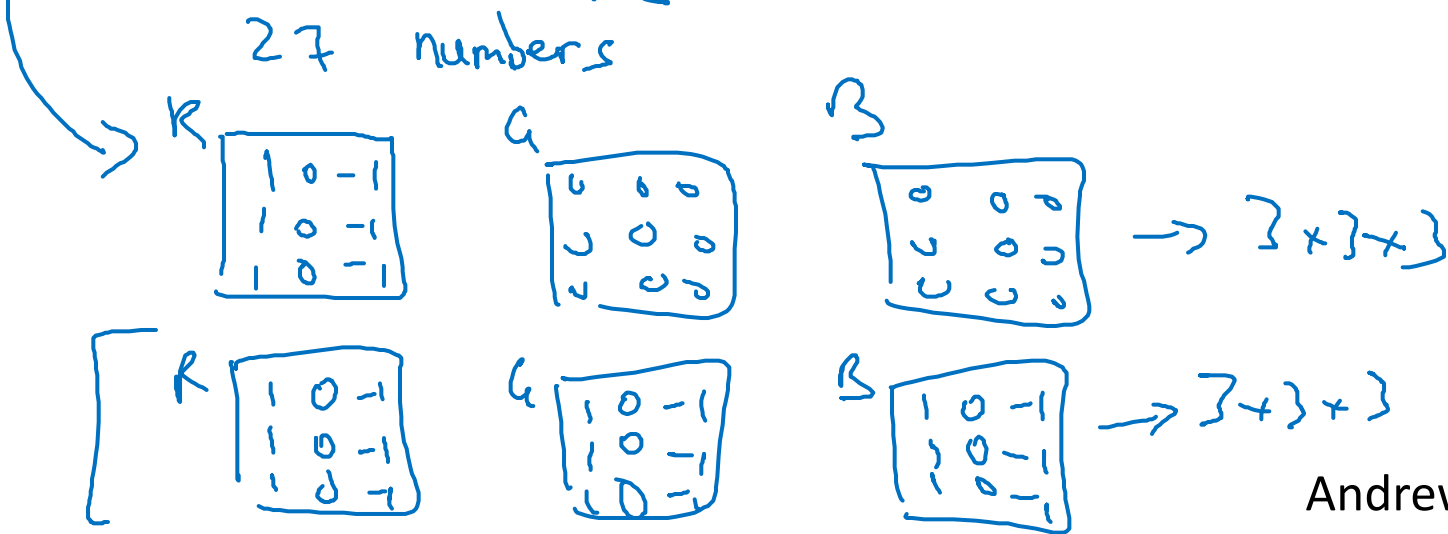
Andrew Ng

deeplearning.ai

Convolutional Neural Networks

Convolutions over volumes

Convolutions on RGB images

$6 \times 6 \times 3$

height

width

#channels

$*$

$3 \times 3 \times 3$

$=$

4×4

Convolutions on RGB image

$6 \times 6 \times \boxed{3}$

$3 \times 3 \times \boxed{3}$

27 numbers

$*$ $=$

4 x 4

R
1	0	-1
1	0	-1
1	0	-1

G

B $\rightarrow 3 \times 3 \times 3$

R
1	0	-1
1	0	-1
1	0	-1

G

B $\rightarrow 3 \times 3 \times 3$

Andrew Ng

Multiple filters

Vertical edge

horizontal edge

$6 \times 6 \times 3$

channels

$3 \times 3 \times 3$

$3 \times 3 \times 3$

4×4

4×4

$4 \times 4 \times 2$

$4 \times 4 \times 2$
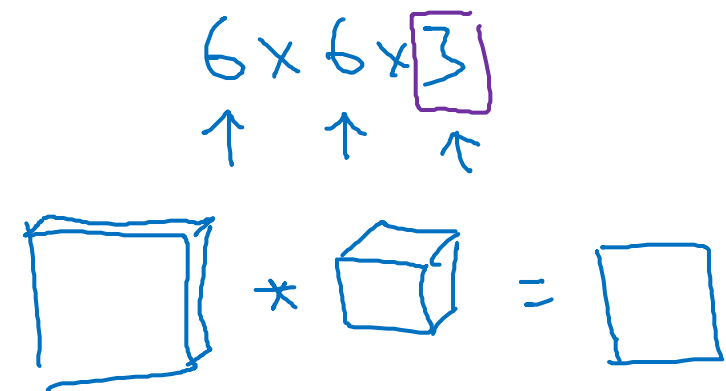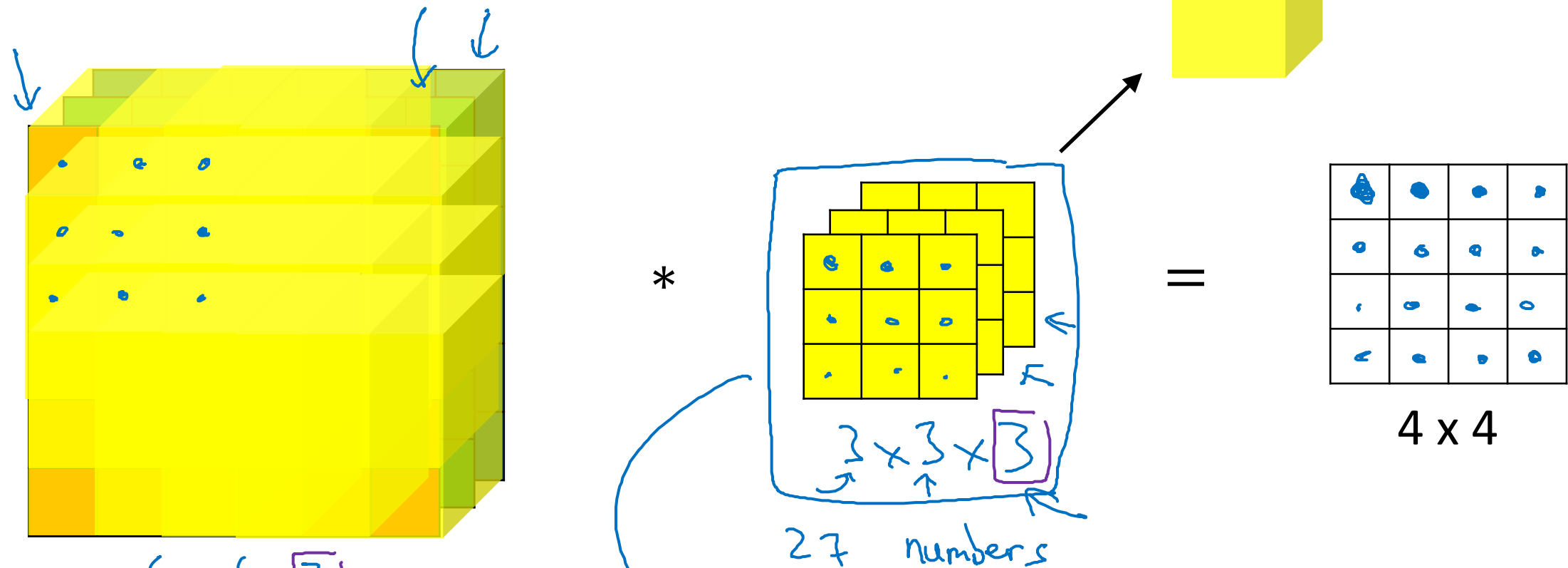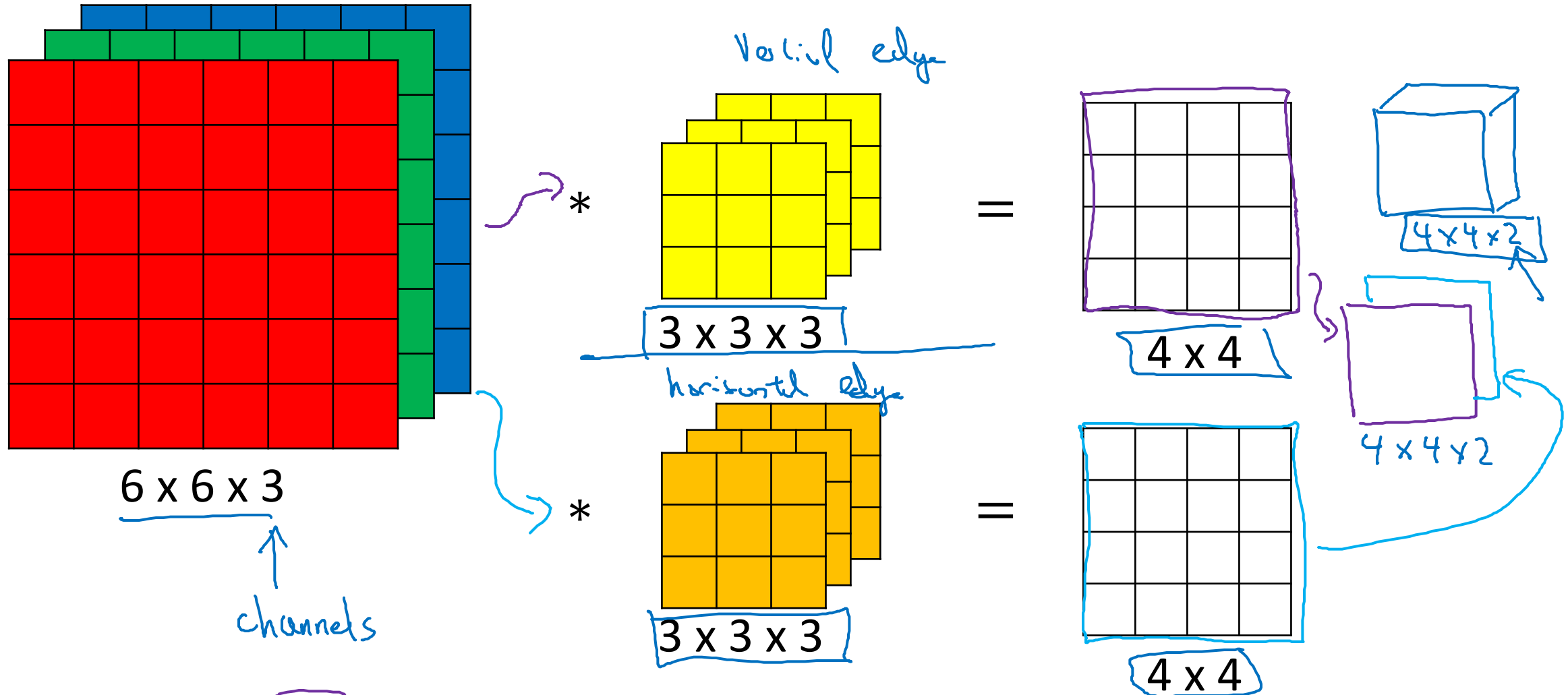
Summary: $n \times n \times \boxed{n_c}$ $* f \times f \times \boxed{n_c}$ $\longrightarrow \dfrac{n-f+1}{4} \times \dfrac{n-f+1}{4} \times \dfrac{n_c'}{\times 2}$

$6 \times 6 \times 3$ $3 \times 3 \times 3$ #filters

Andrew Ng
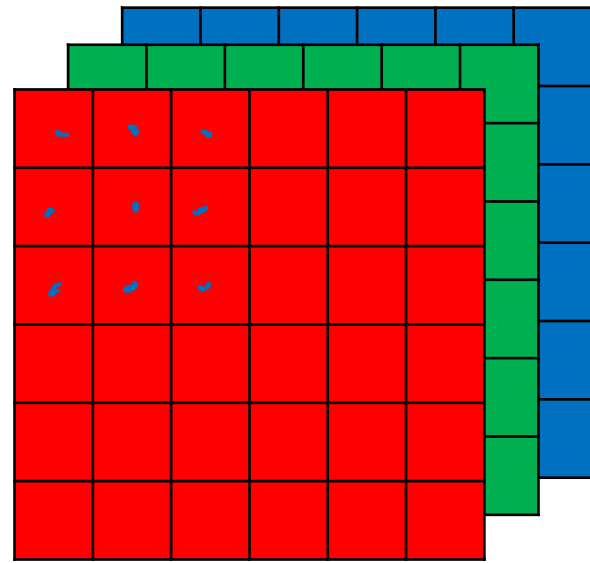
deeplearning.ai

Convolutional Neural Networks

One layer of a convolutional network
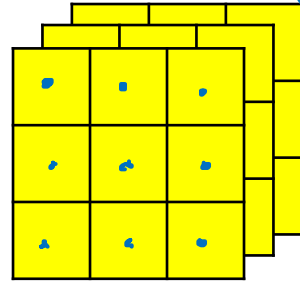
Example of a layer

6 x 6 x 3

$a^{[0]}$
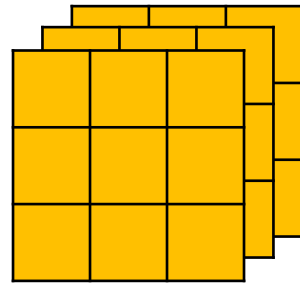
$$z^{[1]} = w^{[1]} a^{[0]} + b^{[1]}$$

$$a^{[1]} = g(z^{[1]})$$

10 filters

3 x 3 x 3

3 x 3 x 3

"$w^{[1]}$"

$6 \times 6 \times 3$ $a^{[0]}$ → $a^{[1]}$ $4 \times 4 \times 2$

"$w^{[1]} a^{[0]}$"

$\mathbb{R}$ → $z^{[1]}$

→ ReLU + b_1 4 x 4 → 4 x 4

→ ReLU + b_2 4 x 4 → 4 x 4

$4 \times 4 \times 2$ ← $a^{[1]}$

$4 \times 4 \times 10$

Andrew Ng
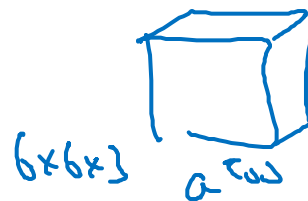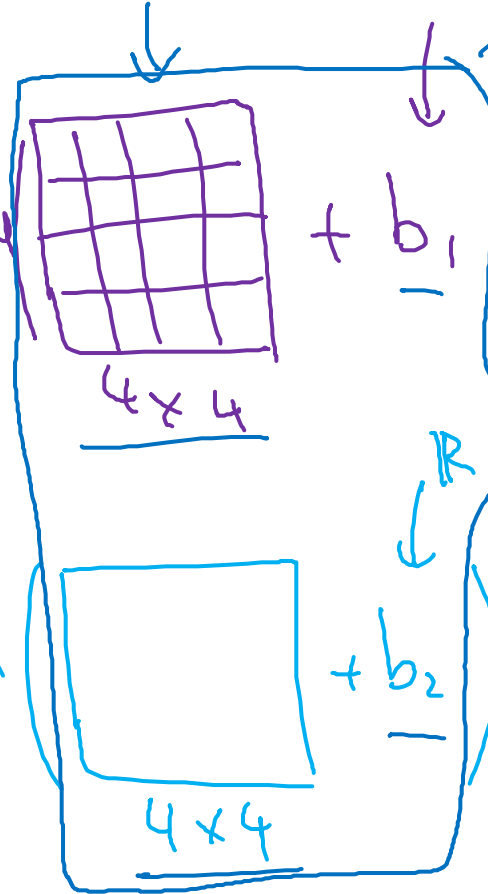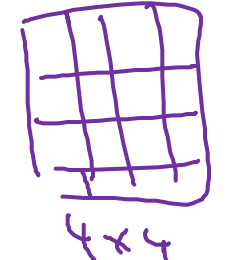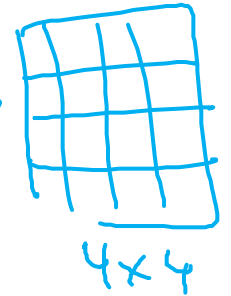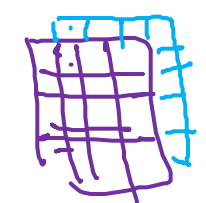
Number of parameters in one layer

If you have 10 filters that are 3 x 3 x 3 in one layer of a neural network, how many parameters does that layer have?

$3 \times 3 \times 3$

27 parameters.

+ bias

→ 28 parameters.

280 parameters.
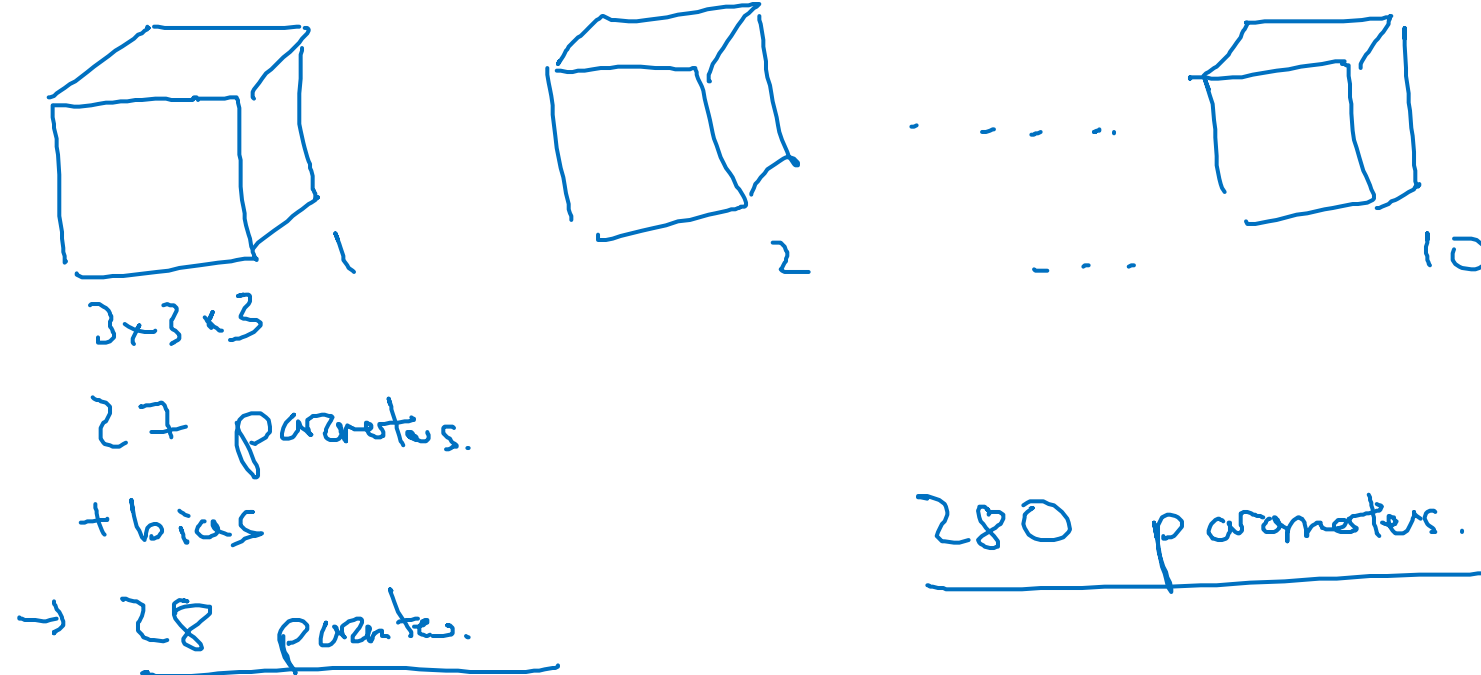
Summary of notation

If layer l is a convolution layer:

$f^{[l]}$ = filter size

$p^{[l]}$ = padding

$s^{[l]}$ = stride

$n_c^{[l]}$ = number of filters

→ Each filter is: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]}$

Activations: $a^{[l]} \rightarrow n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$

Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$

bias: $n_c^{[l]} - (1,1,1,n_c^{[l]})$

↗ #filters in layer l.

Input: $n_H^{[l-1]} \times n_w^{[l-1]} \times n_c^{[l-1]}$

Output: $n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$

$n_{HW}^{[l]} = \left\lfloor \dfrac{n_{Hw}^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$

$A^{[l]} \rightarrow m \times n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$

$n_c \times n_H \times n_w$
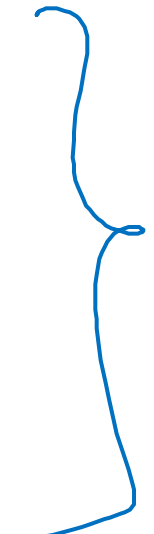
deeplearning.ai

Convolutional Neural Networks

A simple convolution network example

Example ConvNet

$a^{[1]}$

$a^{[2]}$

x

$\boxed{39 \times 39} \times \boxed{3}$

$n_H^{[0]} = n_W^{[0]} = 39$

$n_c^{[0]} = 3$

$\rightarrow f^{[1]} = 3$
$\rightarrow s^{[1]} = 1$
$\rightarrow p^{[1]} = 0$
$\rightarrow 10$ filters

$\boxed{37 \times 37} \times \boxed{10}$

$n_H^{[1]} = n_W^{[1]} = 37$

$n_c^{[1]} = 10$

$f^{[2]} = 5$
$s^{[2]} = 2$
$p^{[2]} = 0$

20 filters

$\boxed{17 \times 17} \times \boxed{20}$

$n_H^{[2]} = n_W^{[2]} = 17$

$n_c^{[2]} = 20$

$f^{[3]} = 5$
$s^{[3]} = 2$

40 filters

$\boxed{7 \times 7} \times \boxed{40}$

$\frac{n + 2p - f}{s} + 1$

$\frac{39 + 0 - 3}{1} + 1 = 37$

1960

$\hat{y}$

logistic / softmax
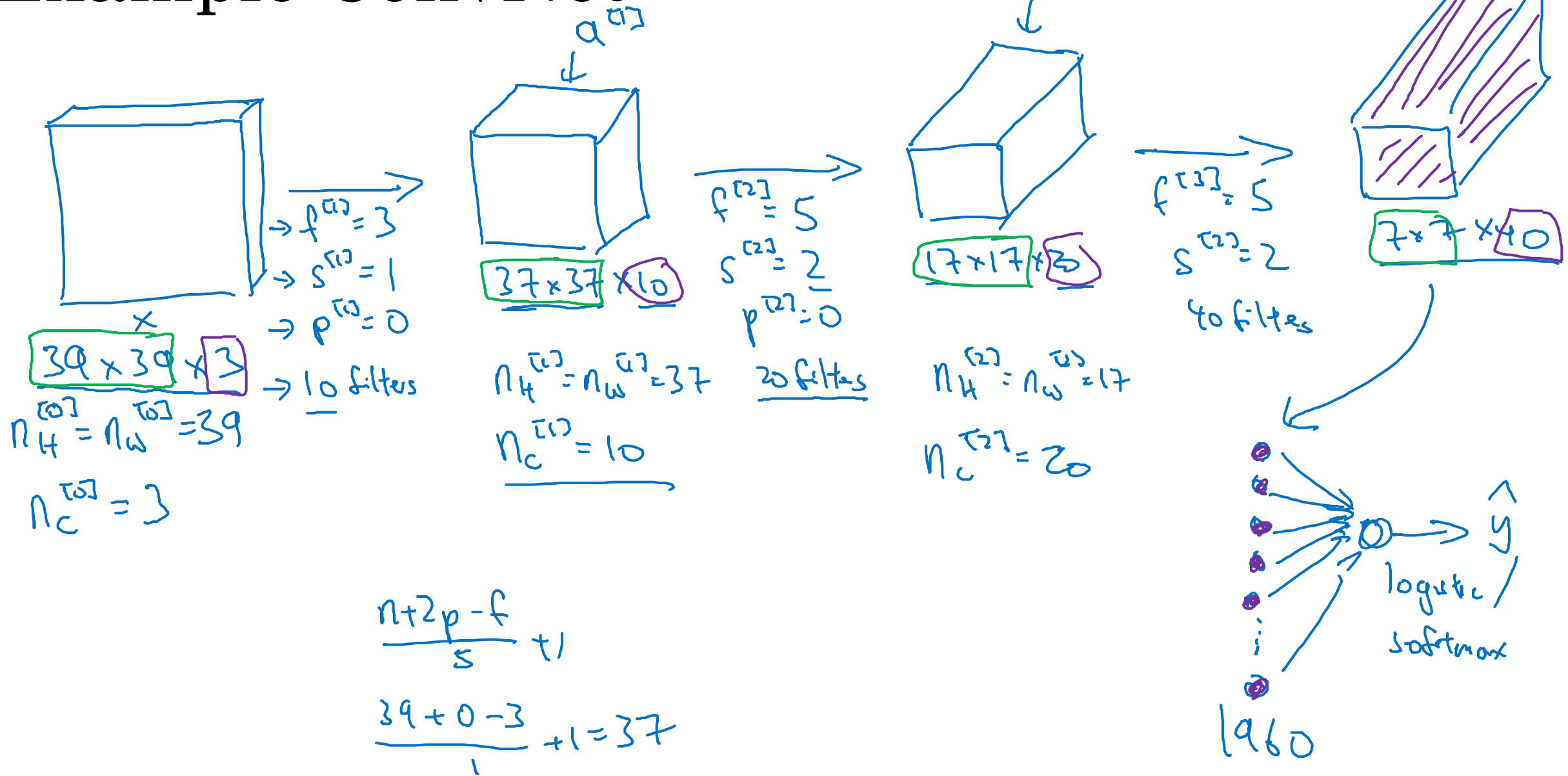
Types of layer in a convolutional network:

- Convolution $\quad$ (CONV) $\leftarrow$

- Pooling $\quad$ (POOL) $\leftarrow$

- Fully connected $\quad$ (FC) $\leftarrow$

Convolutional Neural Networks

Pooling layers

deeplearning.ai

Pooling layer: Max pooling

1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

Pooling layer: Max pooling

$5 \times 5 \times n_c$

$$\left(\frac{n+2p-f}{s} + 1 \right)$$

$3 \times 3 \times n_c$

$$\frac{f = 3}{s = 1}$$

Andrew Ng
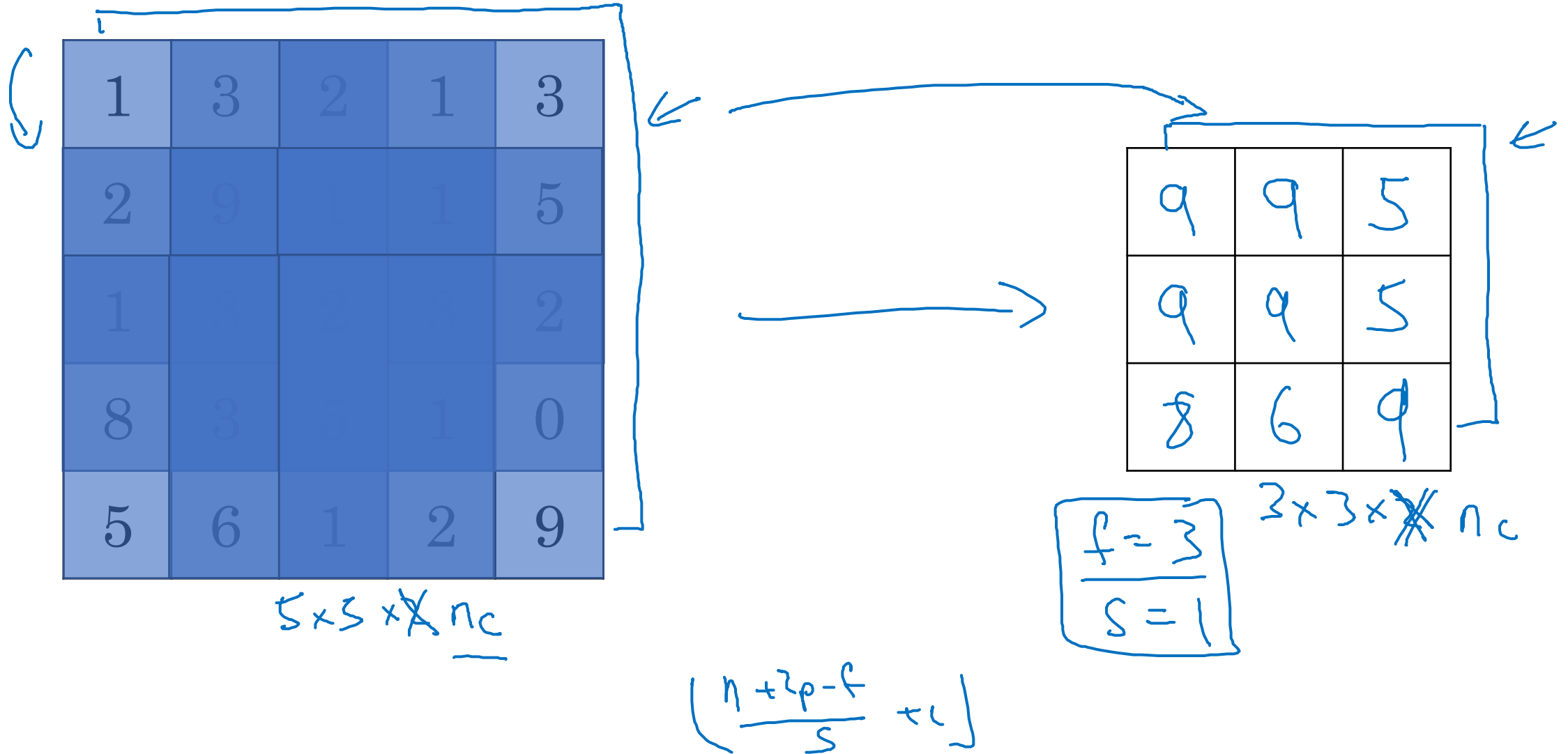
Pooling layer: Average pooling

$f = 2$

$s = 2$

$7 \times 7 \times 1000 \rightarrow 1 \times 1 \times 1000$

Andrew Ng

Summary of pooling

Hyperparameters:

$\quad$ f : filter size

$\quad$ s : stride

$\quad$ Max or average pooling

$f=2, s=2$

$f=3, s=2$

$\rightarrow$ p : padding.

No parameters to learn!

$$n_H \times n_W \times n_c$$

$$\downarrow$$

$$\left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W - f}{s} + 1 \right\rfloor$$

$$\times n_c$$

Convolutional
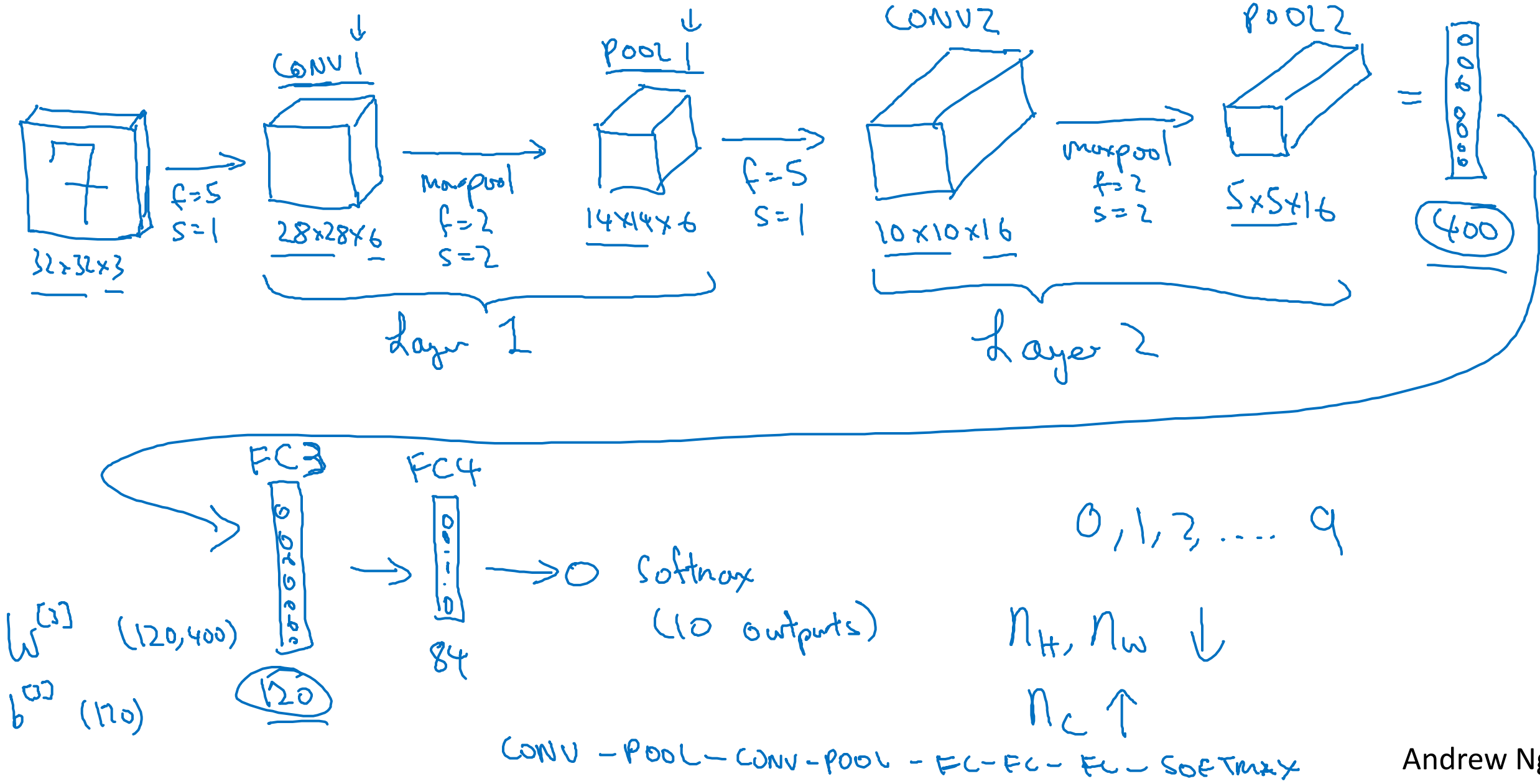Neural Networks

Convolutional neural
network example

deeplearning.ai

Neural network example (LeNet-5)

CONV1

POOL1

CONV2

POOL2

$32 \times 32 \times 3$

$f=5$
$s=1$

$28 \times 28 \times 6$

maxpool
$f=2$
$s=2$

$14 \times 14 \times 6$

$f=5$
$s=1$

$10 \times 10 \times 16$

maxpool
$f=2$
$s=2$

$5 \times 5 \times 16$

(400)

Layer 1

Layer 2

FC3

FC4

softmax
(10 outputs)

$0, 1, 2, \ldots 9$

$W^{[3]}$ (120, 400)

$b^{[3]}$ (120)

(120)

84

$n_H, n_W \downarrow$

$n_C \uparrow$

CONV – POOL – CONV – POOL – FC – FC – FC – SOFTMAX

Andrew Ng

Neural network example

	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	— 3,072 $a^{[0]}$	0
CONV1 (f=5, s=1)	(28,28,8)	6,272	608 ←
POOL1	(14,14,8)	1,568	0 ←
CONV2 (f=5, s=1)	(10,10,16)	1,600	3216 ←
POOL2	(5,5,16)	400	0 ←
FC3	(120,1)	120	48120
FC4	(84,1)	84	10164
Softmax	(10,1)	10	850

Convolutional
Neural Networks

Why convolutions?

deeplearning.ai
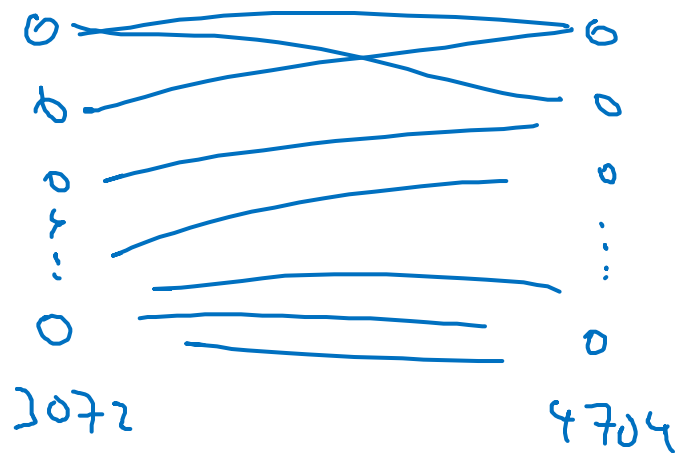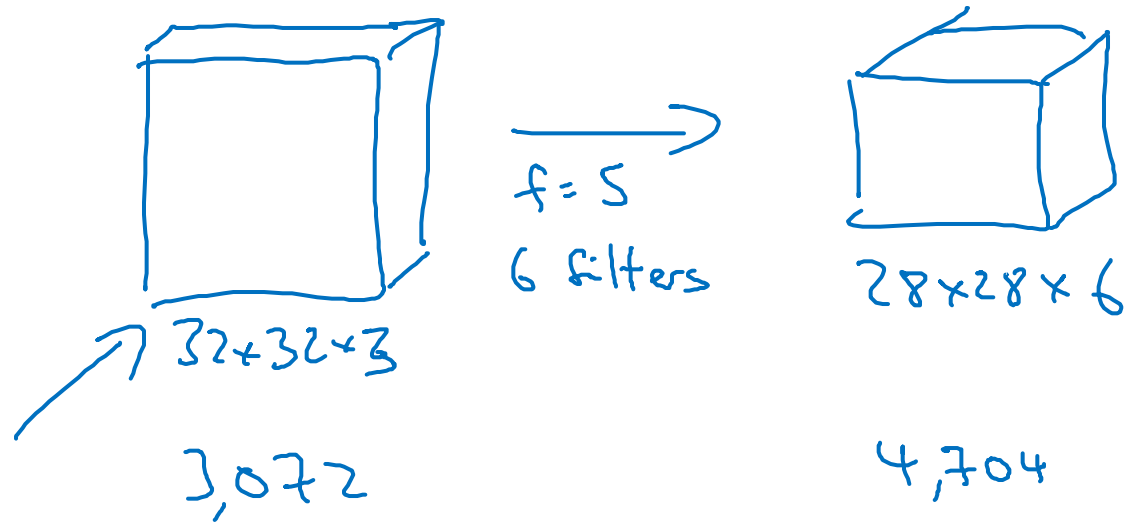
Why convolutions

$$5 \times 5 \quad - \quad 25$$

$$26$$

$$6 \times 26 = 156 \text{ parameters}$$

32×32×3

3,072

28×28×6

4,704

$$3,072 \times 4,704 \approx 14M$$

3072

4704

f=5
6 filters

Why convolutions

translation invariance

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

1	0	-1
1	0	-1
1	0	-1

3 x 3

=

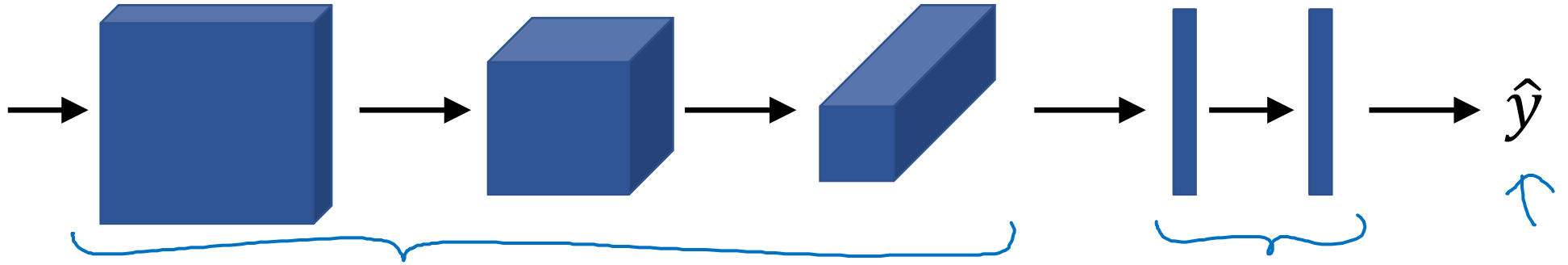| 0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0
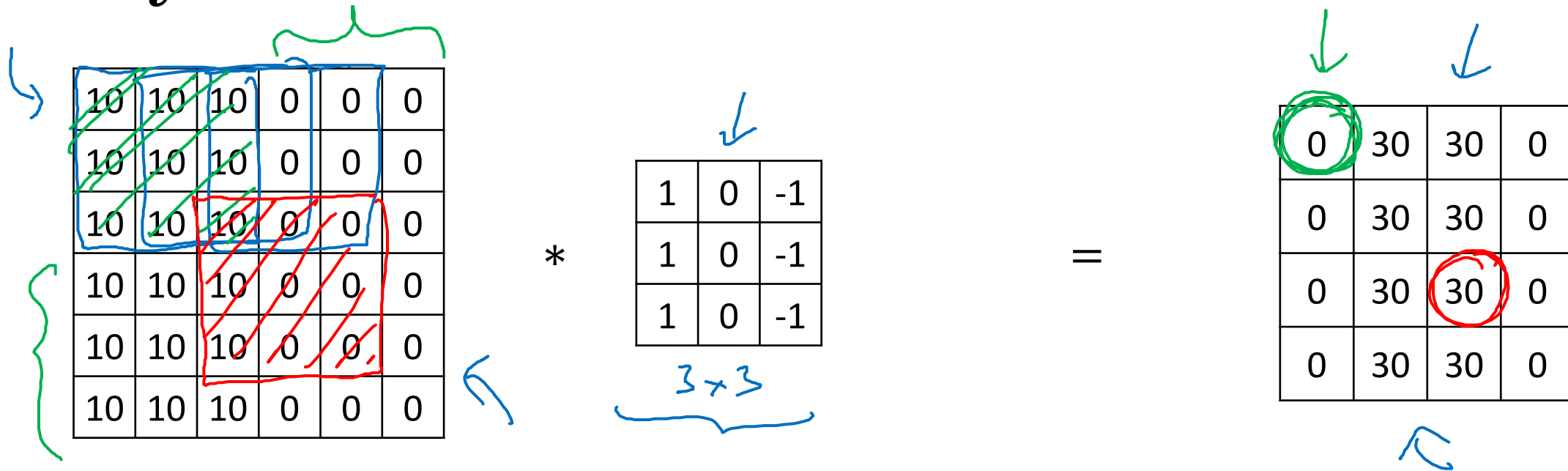
Parameter sharing: A feature detector (such as a vertical edge detector) that's useful in one part of the image is probably useful in another part of the image.

Sparsity of connections: In each layer, each output value depends only on a small number of inputs.

Andrew Ng

Putting it together

Training set $(x^{(1)}, y^{(1)}) \dots (x^{(m)}, y^{(m)})$.

w, b

$$\text{Cost } J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

Use gradient descent to optimize parameters to reduce J