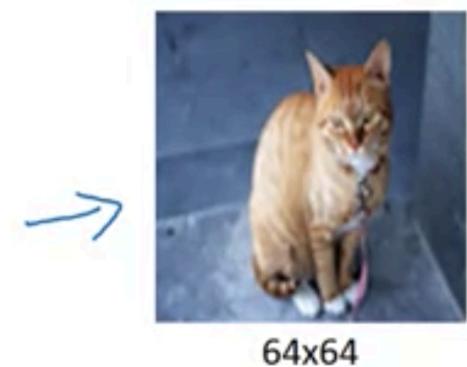


Computer Vision Problems

Image Classification



→ Cat? (0/1)

Neural Style Transfer



Object detection



Andrew Ng

Deep Learning on large images



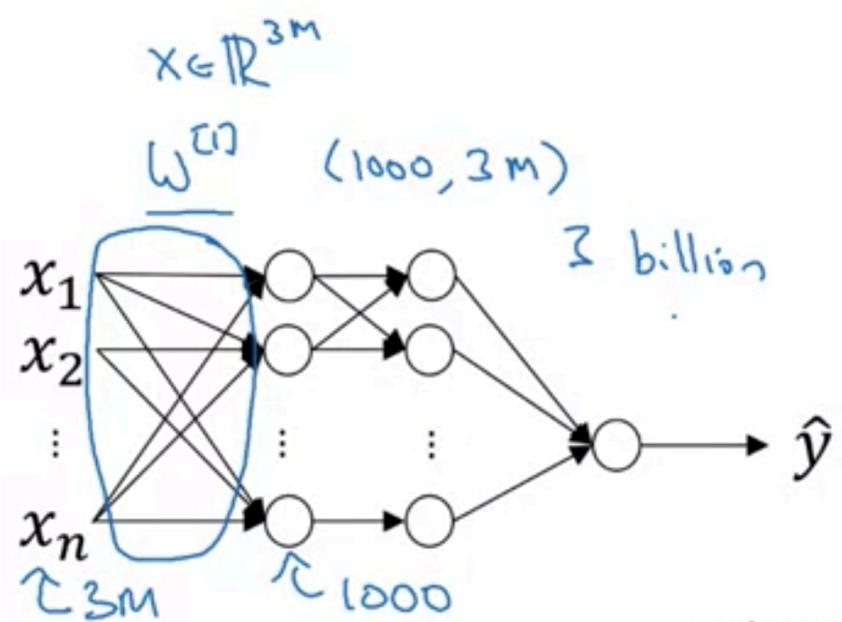
→ Cat? (0/1)

$64 \times 64 \times 3$

12288

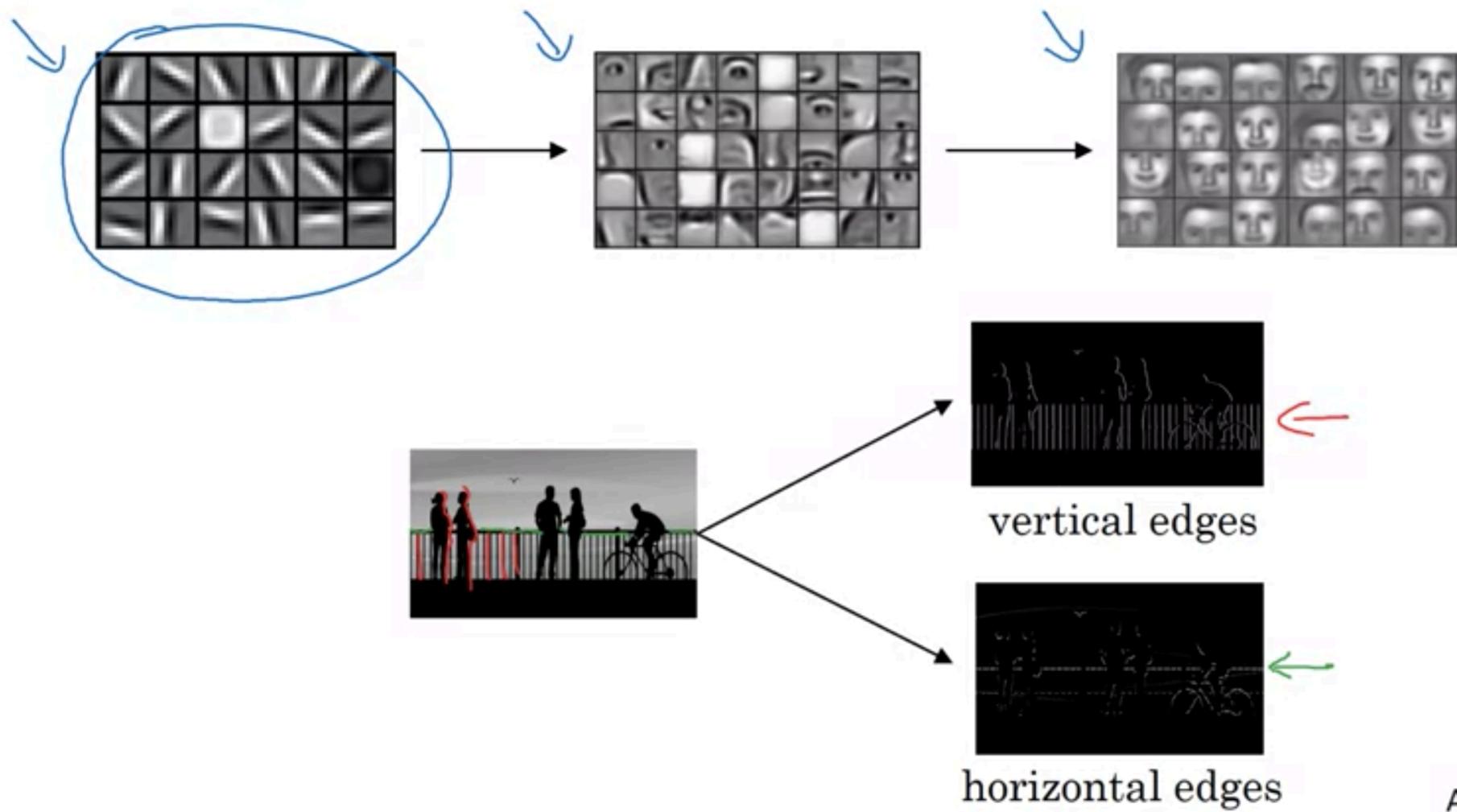


$1000 \times 1000 \times 3$
= 3 million



Andrew Ng

Computer Vision Problem



Andrew Ng

Vertical edge detection

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

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

6x6

"convolution"

*

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

3x3
filter

=

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

4x4

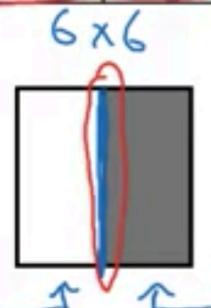
python: conv-forward
tensorflow: tf.nn.conv2d
keras: Conv2D

Andrew Ng

Vertical edge detection

A 6x6 input matrix with values 10 and 0. Green arrows point down along the first three columns, and blue arrows point down along the last three columns. A red box highlights the bottom row.

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



*

A 3x3 kernel with values 1, 0, -1 repeated across the rows. A blue arrow points down through the center column. A pink bracket indicates the kernel's width is 3. A blue bracket indicates its height is 3.

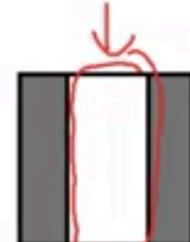
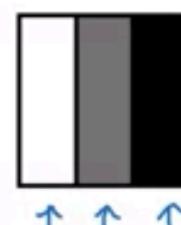
$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

=

The result of the convolution. The bottom row is highlighted with a red box. A pink bracket indicates the output's width is 4. A blue bracket indicates its height is 4.

$$\begin{bmatrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{bmatrix}$$

*



Andrew Ng

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



*

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

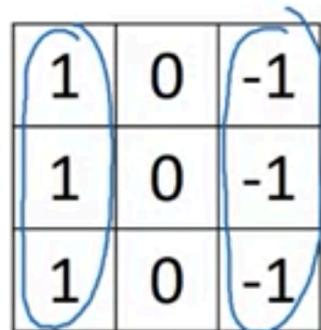
=

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



Andrew Ng

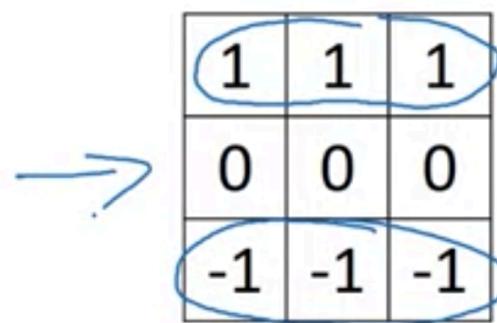
Vertical and Horizontal Edge Detection



A 3x3 matrix with values 1, 0, -1 in each row. The first and third columns are circled in blue.

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

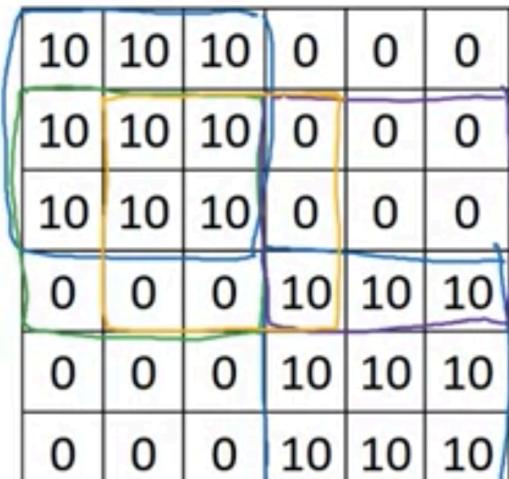
Vertical



A 3x3 matrix with values 1, 1, 1 in the top row; 0, 0, 0 in the middle row; and -1, -1, -1 in the bottom row. The first and third columns are circled in blue.

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

Horizontal

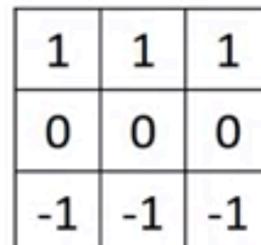


A 6x6 matrix representing an image. It has several highlighted regions: a green box in the top-left, an orange box in the middle-left, a purple box in the bottom-right, and a blue box in the bottom-right corner. A 3x3 kernel is shown below it.

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

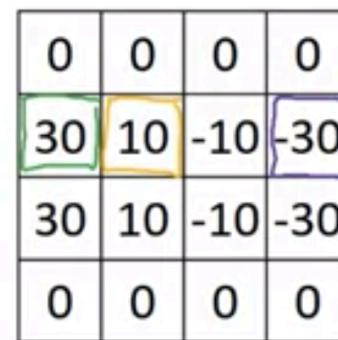
*



A 3x3 matrix with values 1, 1, 1 in the top row; 0, 0, 0 in the middle row; and -1, -1, -1 in the bottom row. The first and third columns are circled in blue.

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

=

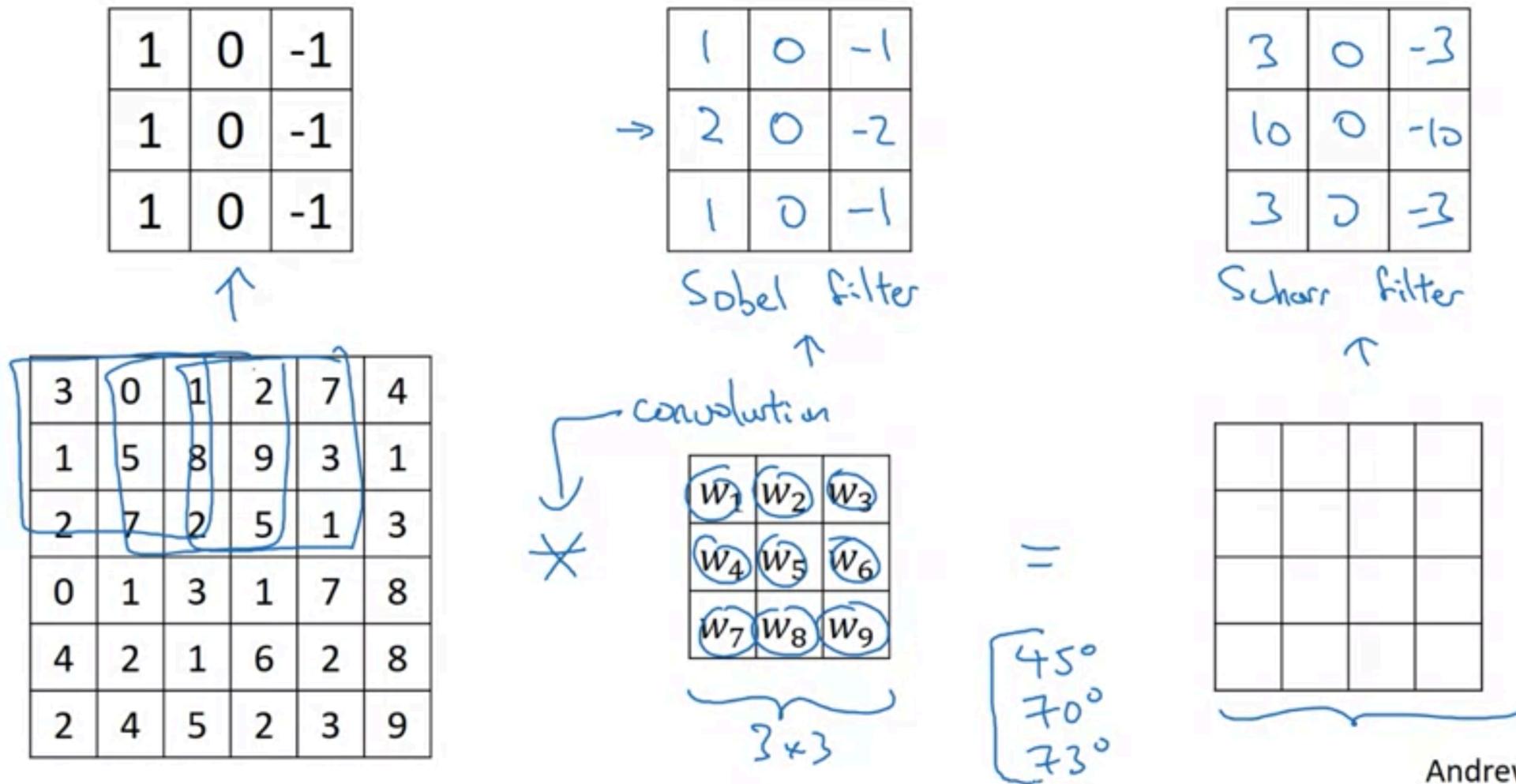


A 4x4 matrix representing the output of the convolution. It has several highlighted regions: a green box in the top-left, an orange box in the middle-left, a purple box in the bottom-right, and a blue box in the bottom-right corner. The diagonal elements are 0, 0, 0, 0.

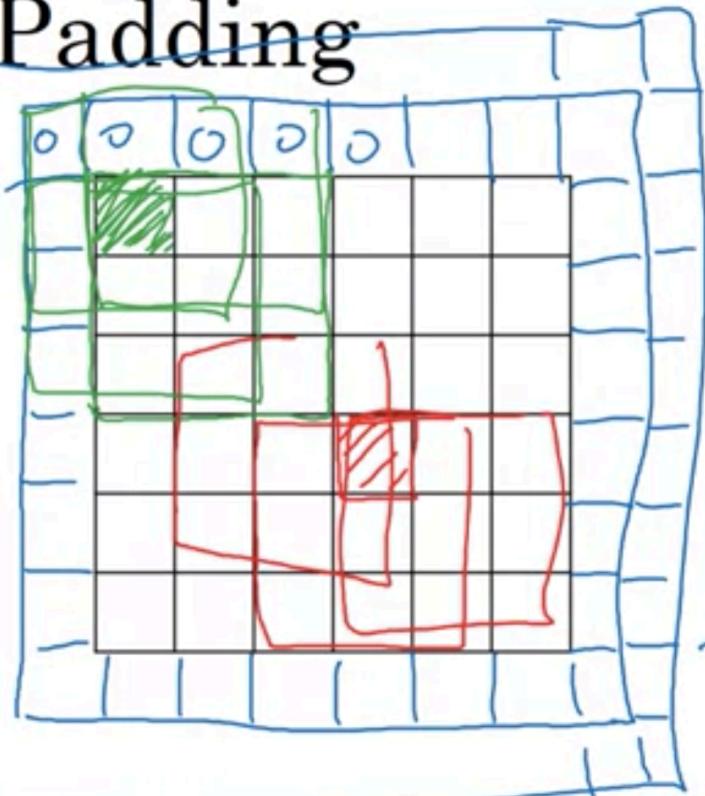
0	0	0	0
30	10	-10	-30
30	10	-10	-30
0	0	0	0

Andrew Ng

Learning to detect edges



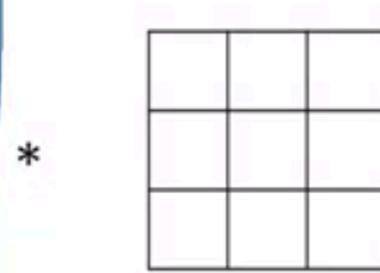
Padding



$$\frac{6 \times 6}{n \times n} \rightarrow 8 \times 8$$

$$P = \text{padding} = 1$$

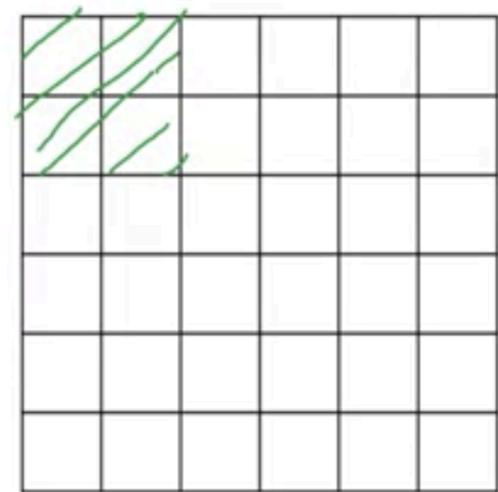
- shrinky output
- throw away info from edge



$$\begin{matrix} 3 \times 3 \\ f \times f \end{matrix}$$

$$\left. \right\} p=2$$

=



$$\underline{\underline{6 \times 6}}$$

$$\overrightarrow{\underline{\underline{4 \times 4}}}$$

$$\begin{matrix} n-f+1 & \times & n-f+1 \\ 6-3+1=4 & & \end{matrix}$$

$$\begin{matrix} n+2p-f+1 & \times & n+2p-f-1 \\ 6+2-3+1 \times & & = 6 \times 6 \end{matrix}$$

Andrew Ng

Valid and Same convolutions

↑^{n=padding}

“Valid”: $n \times n \times f \times f \rightarrow \frac{n-f+1}{f} \times \frac{n-f+1}{f}$

$6 \times 6 \times 3 \times 3 \rightarrow 4 \times 4$

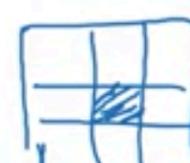
“Same”: Pad so that output size is the same as the input size.

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

$$\cancel{n + 2p - f + 1 = n} \Rightarrow p = \frac{f-1}{2}$$

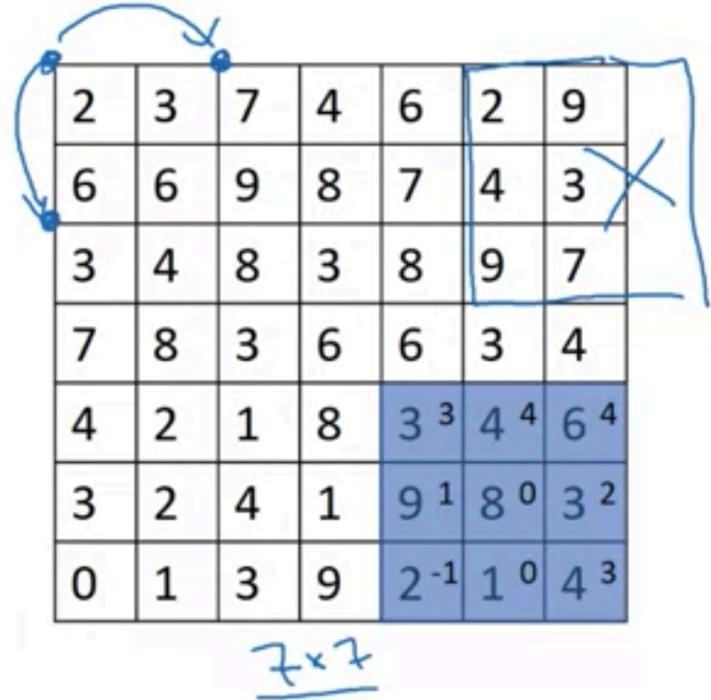
$$3 \times 3 \quad p = \frac{3-1}{2} = 1 \quad \left| \begin{array}{c} 5 \times 5 \\ f=5 \end{array} \right. \quad p=2$$

f is usually odd
 1×1
 3×3
 5×5
 7×7



Andrew Ng

Strided convolution



$n \times n$ * $f \times f$
padding p stride s
 $s=2$

$$\begin{array}{c}
 \begin{array}{|c|c|c|} \hline 3 & 4 & 4 \\ \hline 1 & 0 & 2 \\ \hline -1 & 0 & 3 \\ \hline \end{array} & * & \begin{array}{|c|c|c|} \hline 91 & 100 & 83 \\ \hline 69 & 91 & 127 \\ \hline 44 & 72 & 74 \\ \hline \end{array} \\
 \frac{3 \times 3}{\text{stride } = 2} & & \frac{3 \times 3}{\lfloor \frac{7}{2} \rfloor = \text{floor}(\frac{7}{2})}
 \end{array}$$

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

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

Andrew Ng

Summary of convolutions

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

padding p stride s

Output

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

Technical note on cross-correlation vs. convolution

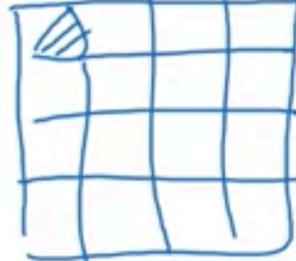
Convolution in math textbook:

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

$$A * B = C$$

where

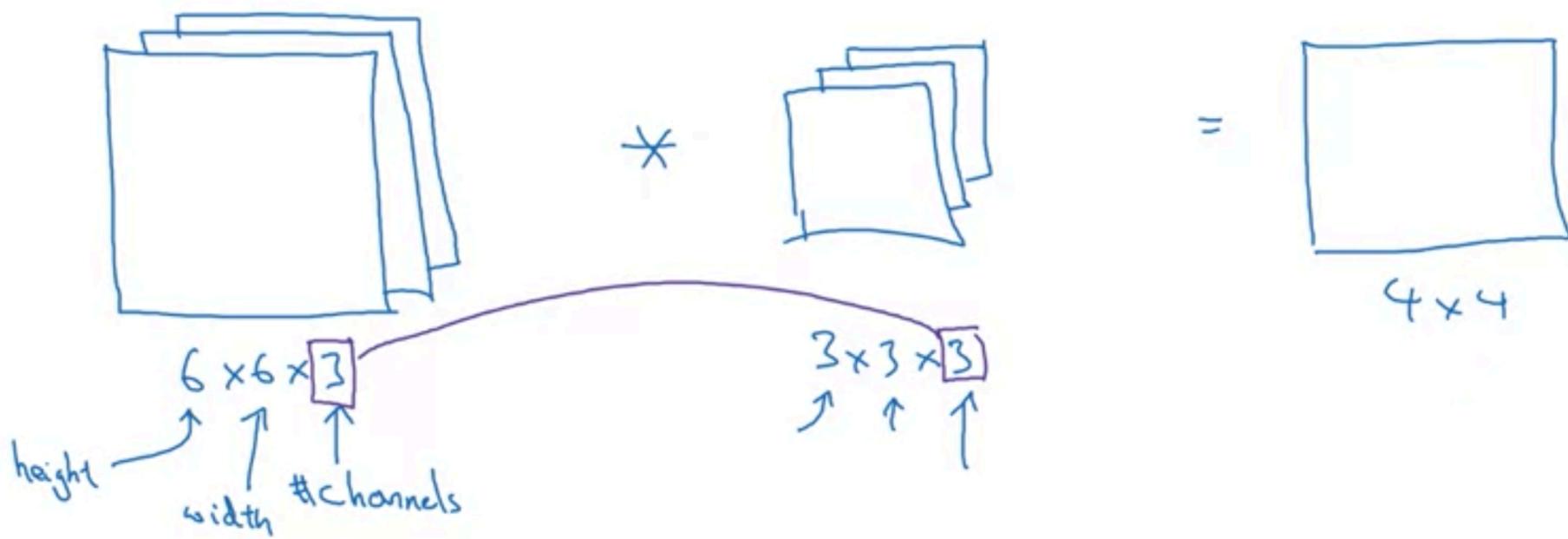
$$A = \begin{bmatrix} 2 & 3 & 7 & 4 & 6 & 2 \\ 6 & 6 & 9 & 8 & 7 & 4 \\ 3 & 4 & 8 & 3 & 8 & 9 \\ 7 & 8 & 3 & 6 & 6 & 3 \\ 4 & 2 & 1 & 8 & 3 & 4 \\ 3 & 2 & 4 & 1 & 9 & 8 \end{bmatrix}$$
$$B = \begin{bmatrix} 3 & 4 & 5 \\ 1 & 0 & 2 \\ -1 & 9 & 7 \end{bmatrix}$$
$$C = \begin{bmatrix} 7 & 2 & 5 \\ 9 & 0 & 4 \\ -1 & 1 & 3 \end{bmatrix}$$

$$=$$


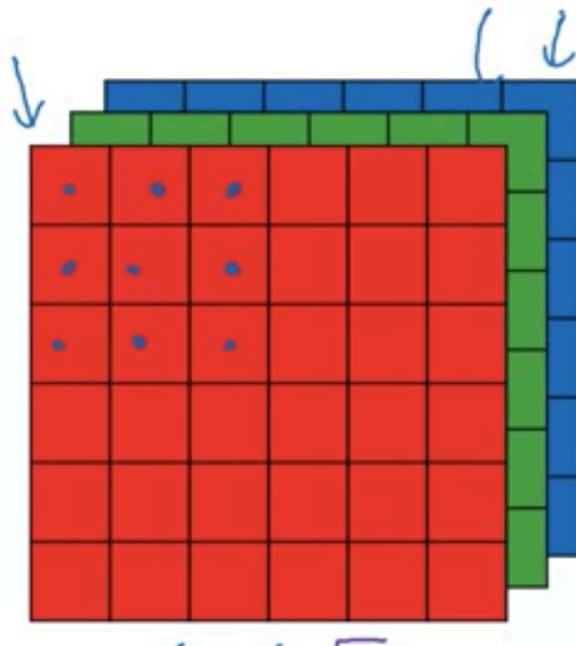
$$(A * B) * C = A * (B * C)$$

Andrew Ng

Convolutions on RGB images

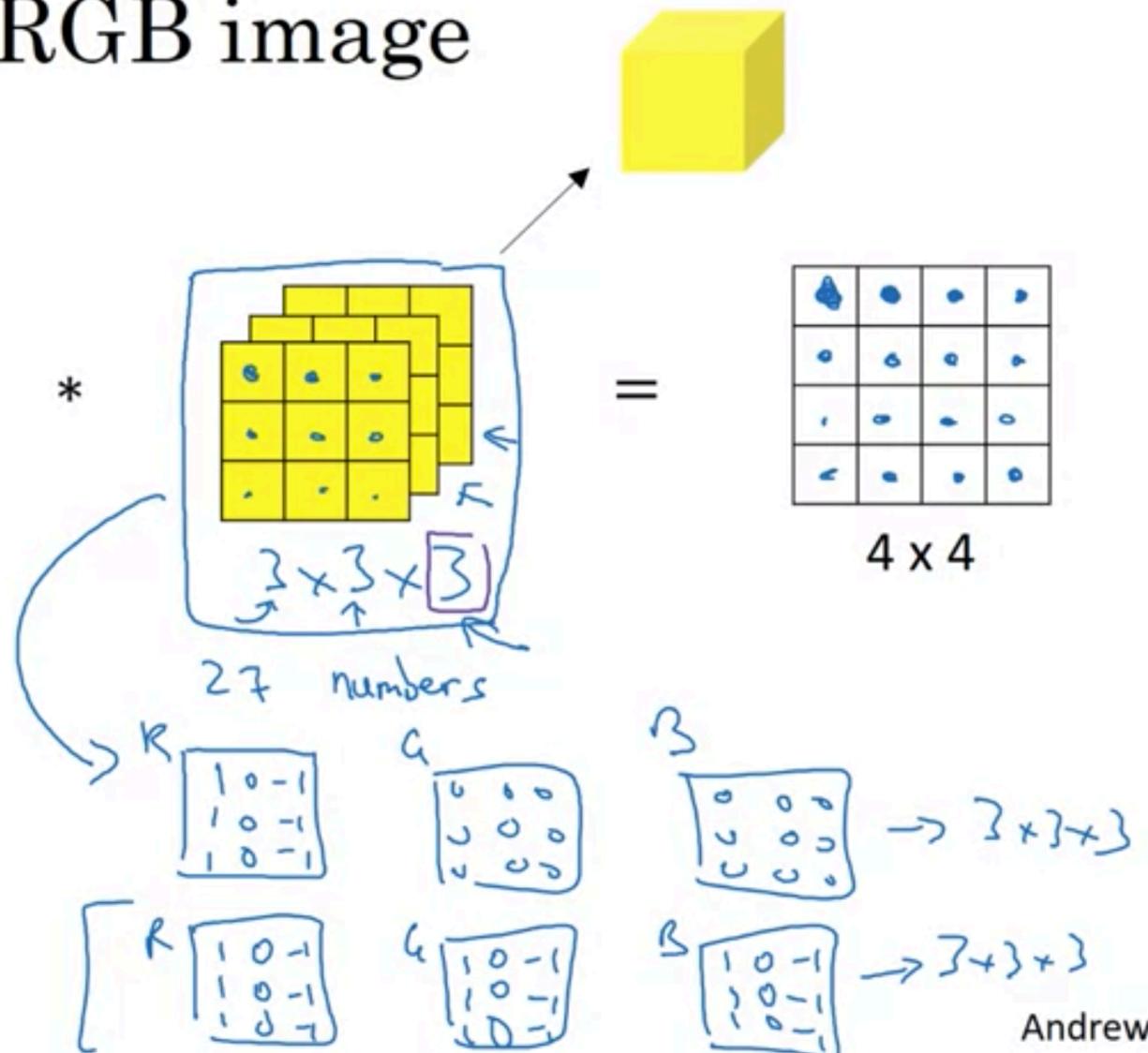


Convolutions on RGB image

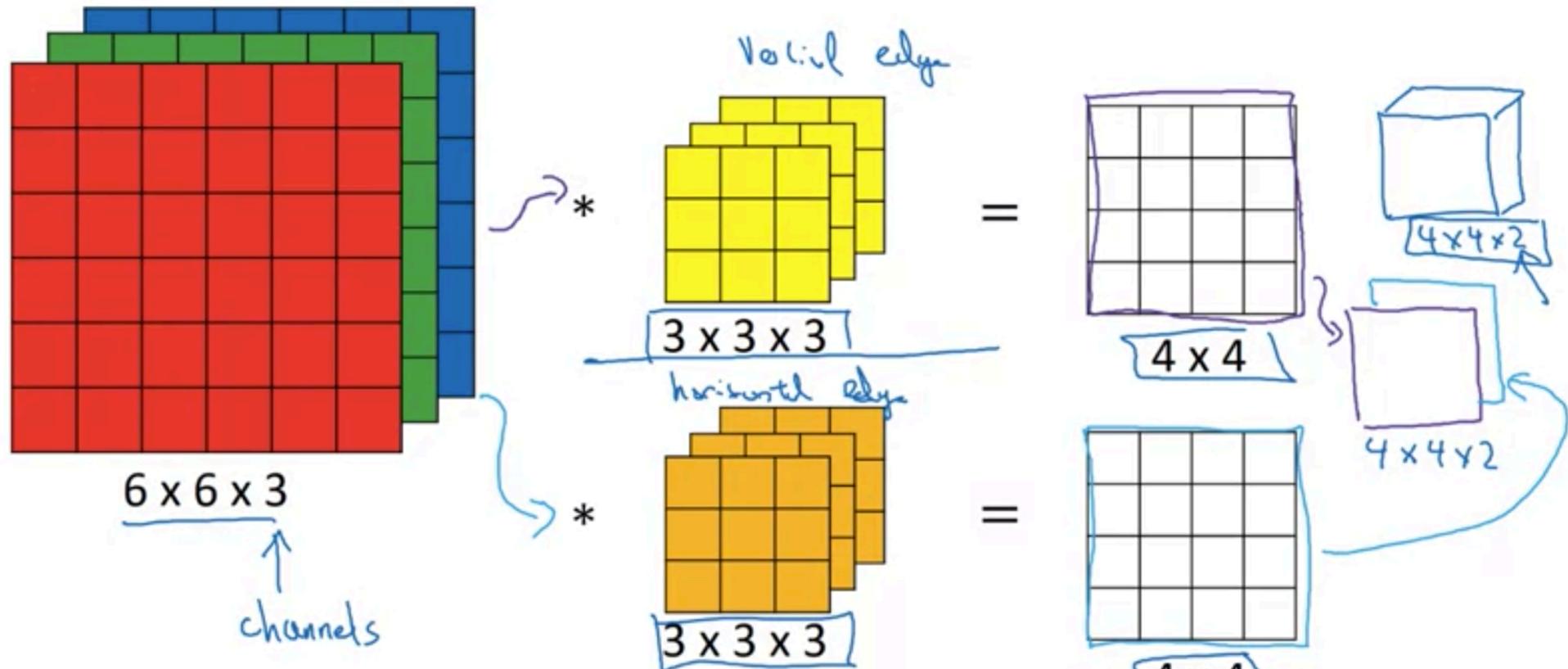


$$\boxed{\text{square}} * \boxed{\text{cube}} = \boxed{\text{square}}$$

Diagram illustrating the convolution operation: An input image (square) is convolved with a kernel (cube) to produce a result (square). The input image has dimensions $6 \times 6 \times 3$.



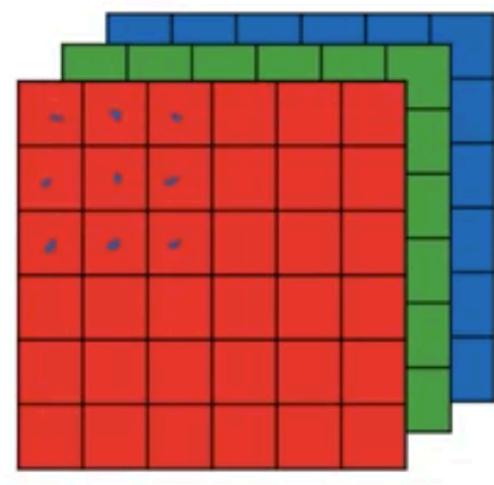
Multiple filters



Summary: $n \times n \times n_c$ \star $f \times f \times n_c$ \rightarrow $\frac{n-f+1}{4} \times \frac{n-f+1}{4} \times n'_c$
 $6 \times 6 \times 3$ $3 \times 3 \times 3$ $\frac{2}{\# \text{filters}}$

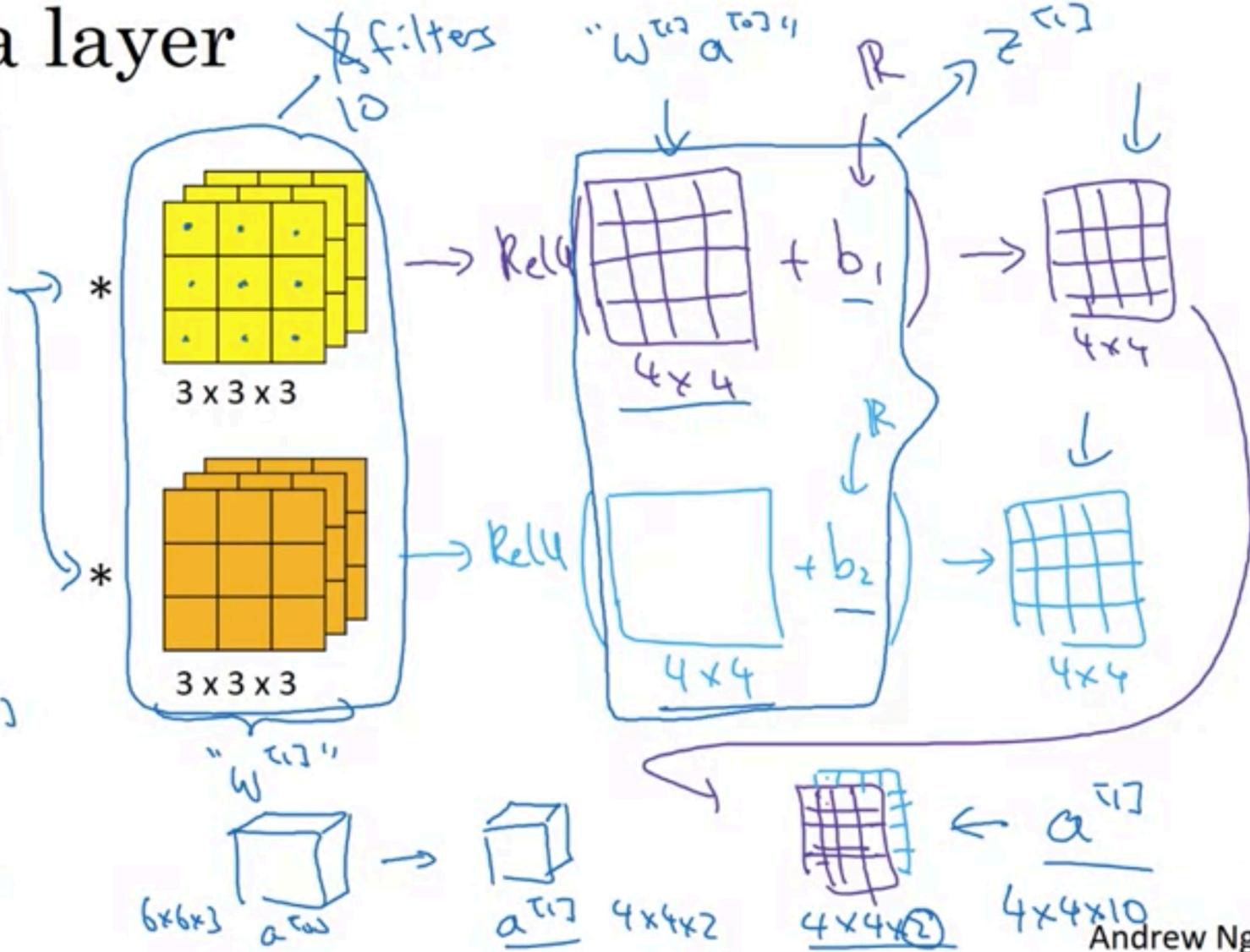
Andrew Ng

Example of a layer



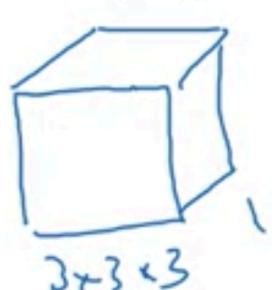
$$z^{(1)} = w^{(1)} a^{(1)} + b^{(1)}$$

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

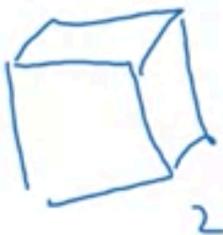


Number of parameters in one layer

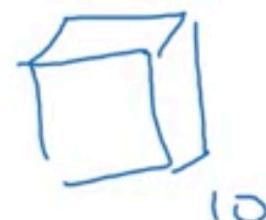
If you have 10 filters that are $3 \times 3 \times 3$ in one layer of a neural network, how many parameters does that layer have?



$3 \times 3 \times 3$



...
...



10

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: $f^{[l]H} \times f^{[l]W} \times n_c^{[l]}$

Activations: $A^{[l]} \rightarrow n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

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

bias: $n_c^{[l]} - (1, 1, 1, n_c^{[l]})$ \uparrow #filters in layer l.

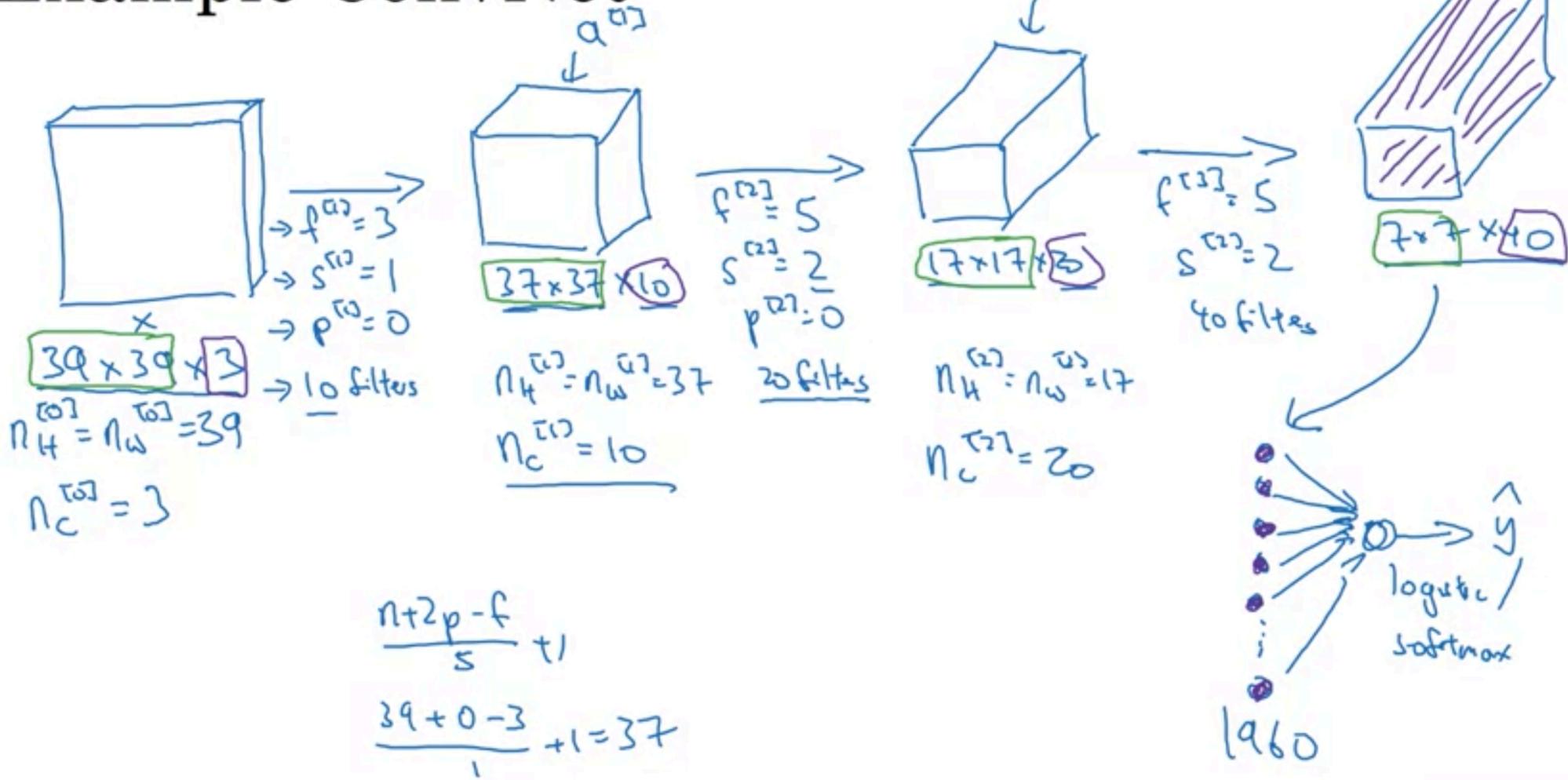
Input: $n_H^{[l-1]} \times n_W^{[l-1]} \times n_c^{[l-1]}$
Output: $n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

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

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

$$n_c \times n_H \times n_W$$

Example ConvNet

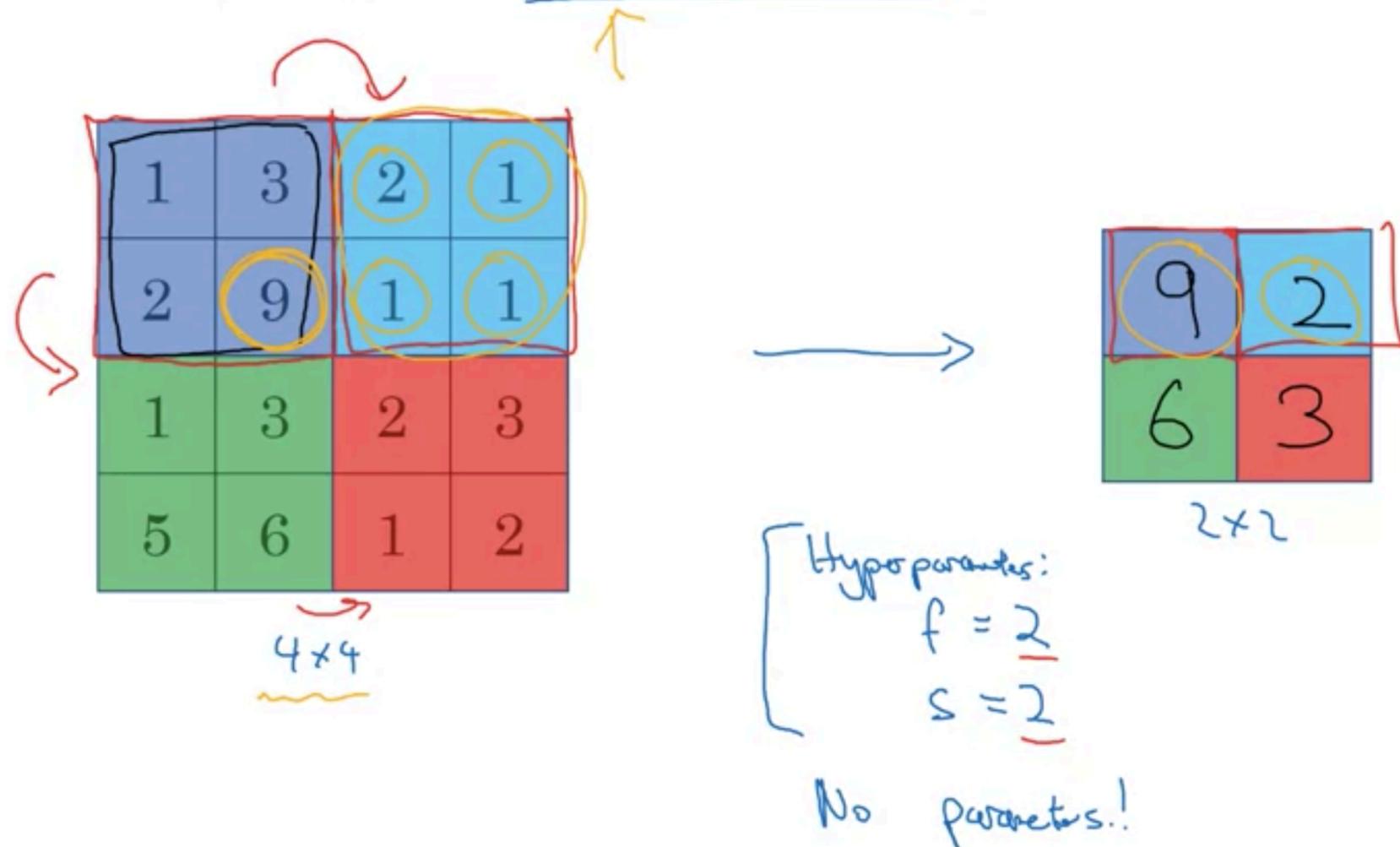


Andrew Ng

Types of layer in a convolutional network:

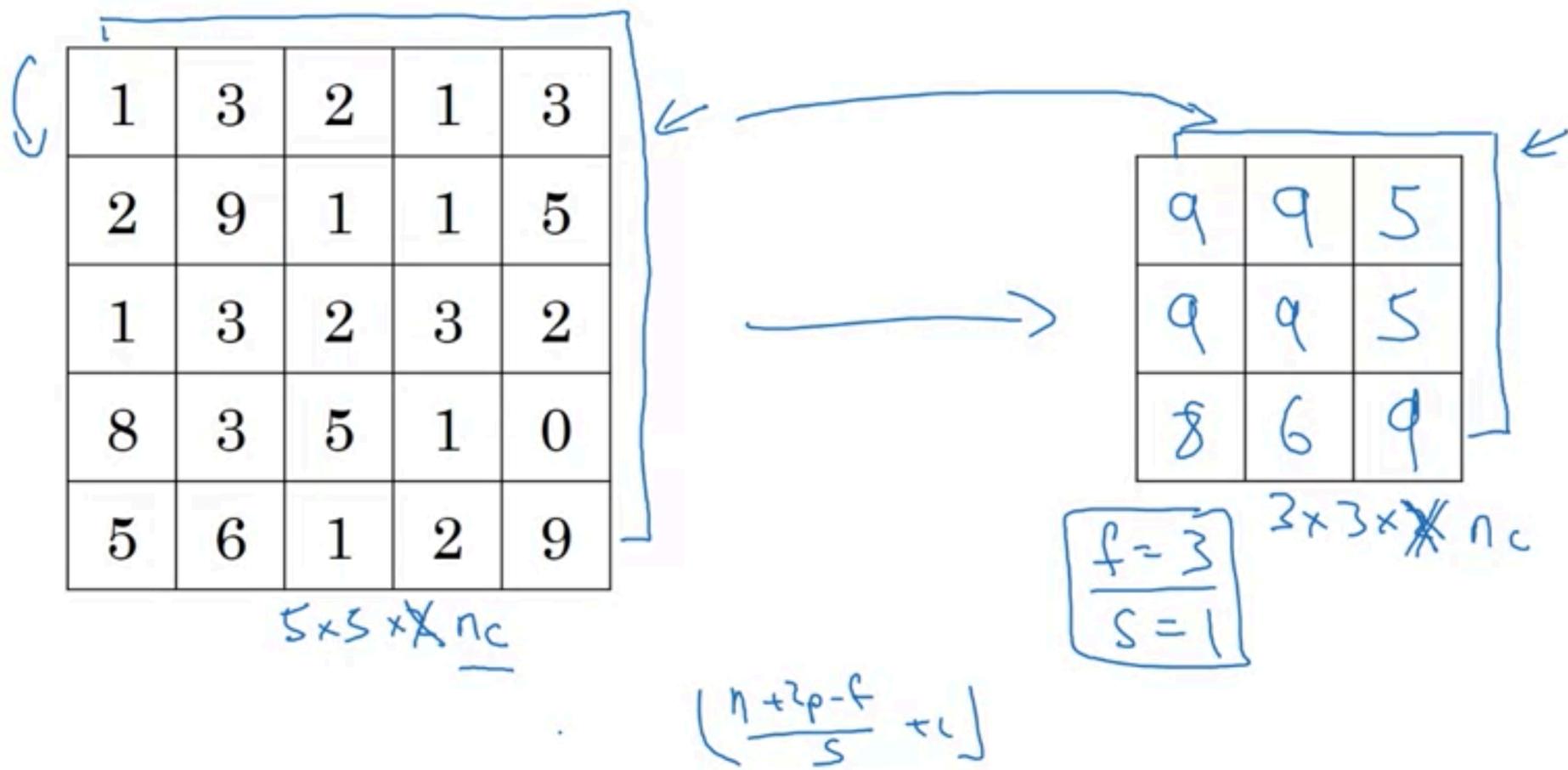
- Convolution (conv) ← }
- Pooling (pool) ← .
- Fully connected (Fc) ←

Pooling layer: Max pooling



Andrew Ng

Pooling layer: Max pooling



Andrew Ng

Pooling layer: Average pooling

1	3	2	1
2	9	1	1
1	4	2	3
5	6	1	2



3.75	1.25
4	2

$$f=2$$

$$s=2$$

$$\underline{7 \times 7 \times 1000} \rightarrow 1 \times 1 \times 1000$$

Andrew Ng

Summary of pooling

Hyperparameters:

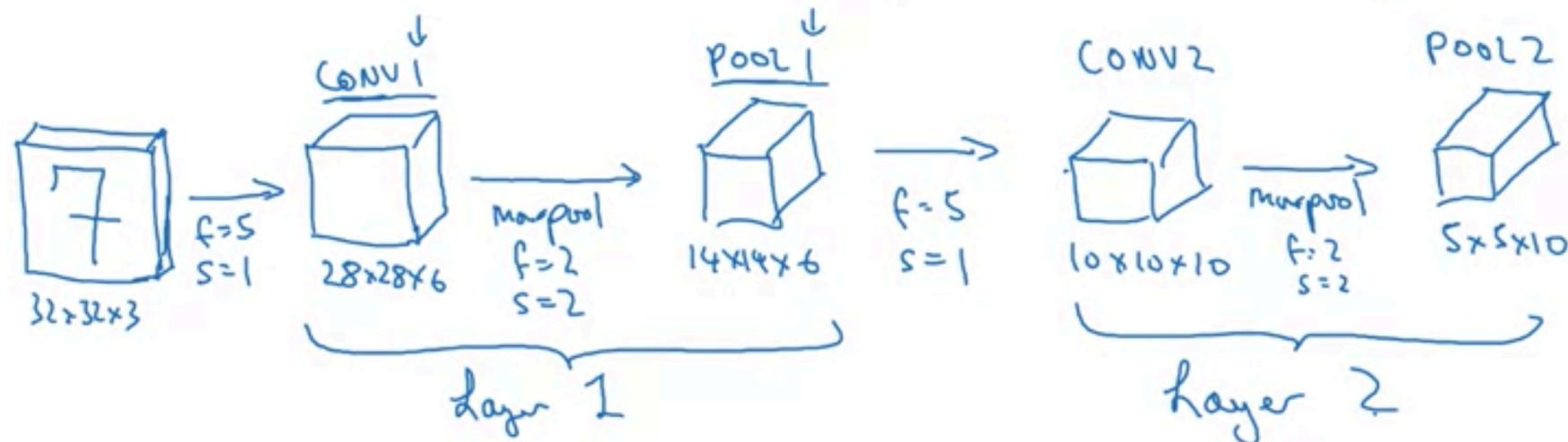
- f : filter size $f=2, s=2$
 $f=3, s=2$
 - s : stride
 - Max or average pooling
- ~~$\Rightarrow p: padding$~~

No parameters to learn!

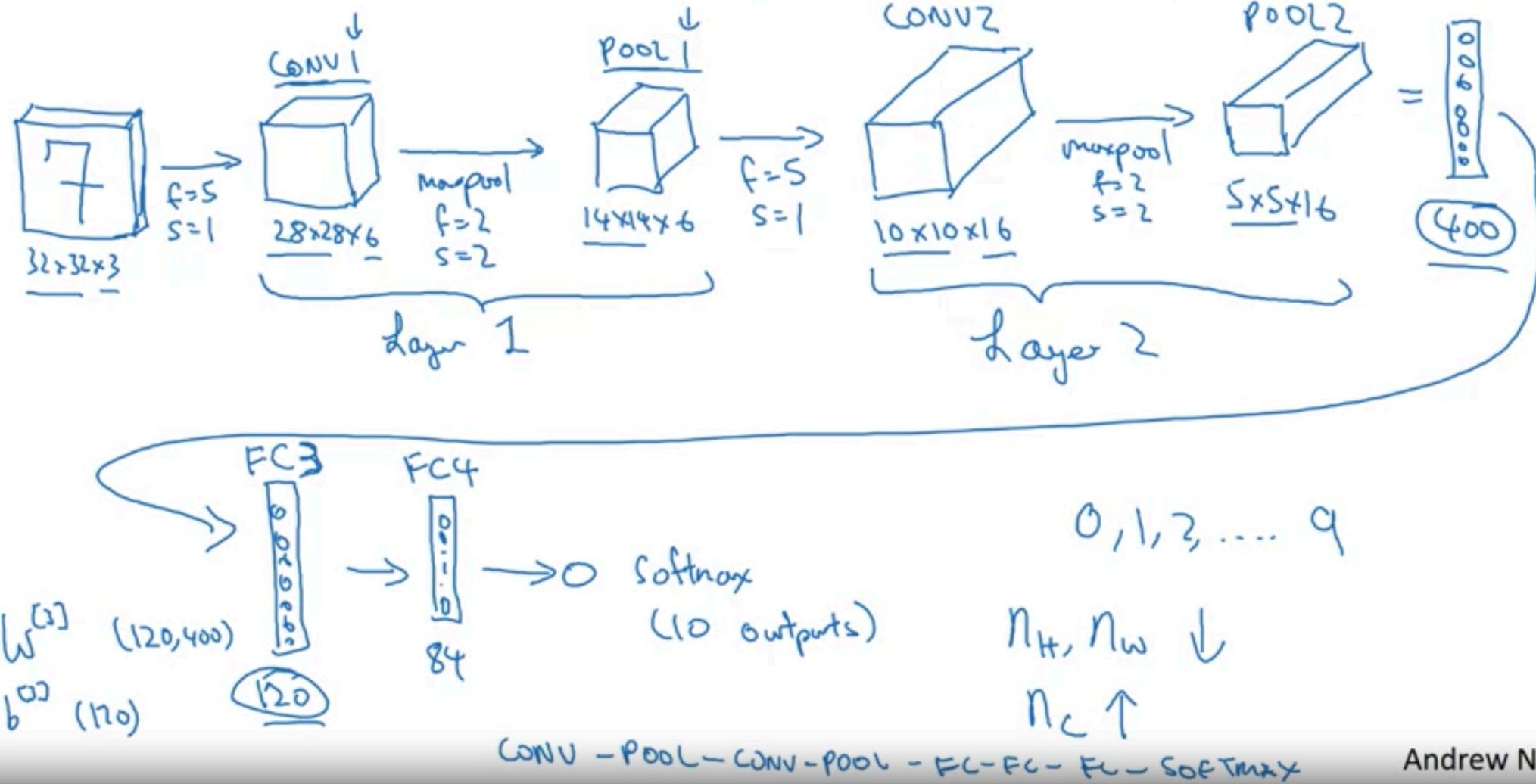
$$n_H \times n_W \times n_C$$

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

Neural network example (LeNet-5)



Neural network example (LeNet-5)



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	208 ←
POOL1	(14,14,8)	1,568	0 ←
CONV2 (f=5, s=1)	(10,10,16)	1,600	416 ←
POOL2	(5,5,16)	400	0 ←
FC3	(120,1)	120	48,001 {
FC4	(84,1)	84	10,081 }
Softmax	(10,1)	10	841

Andrew Ng