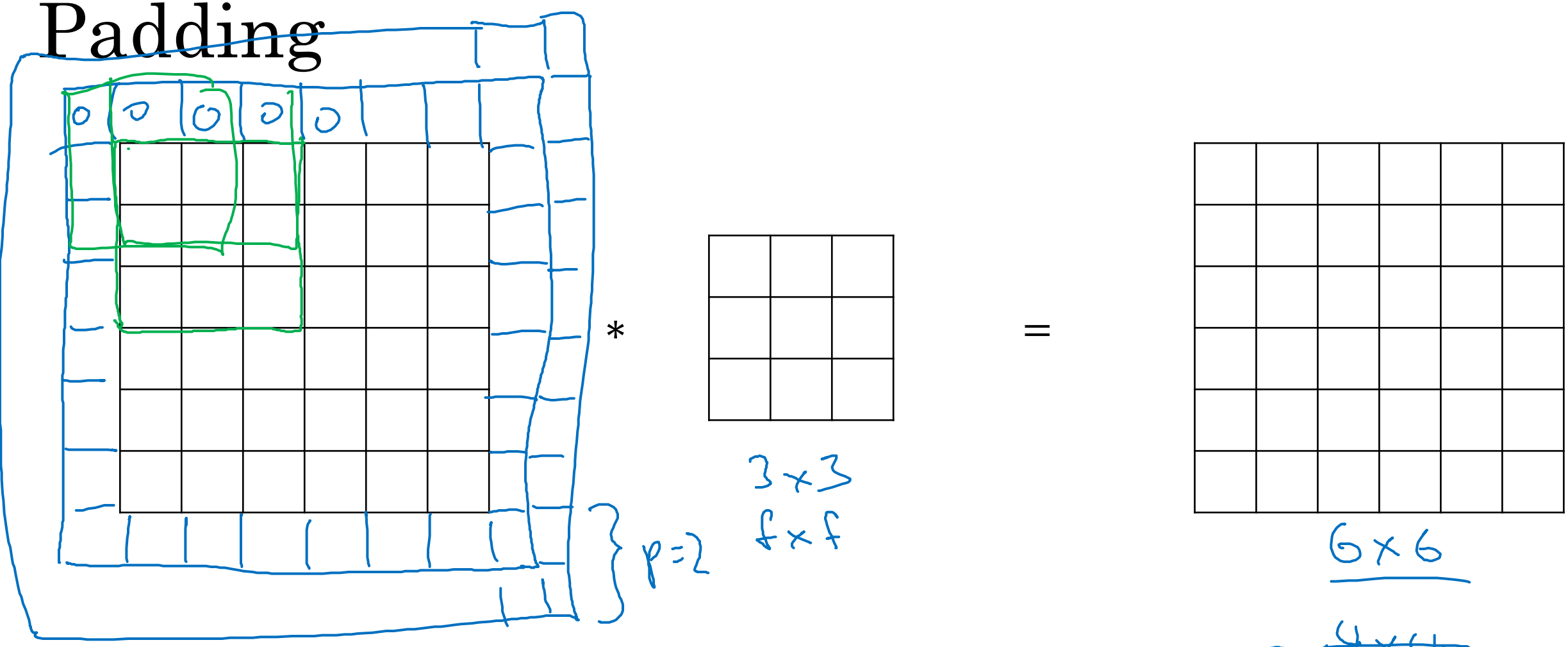
Convolutional Neural Networks

Padding

Padding



$*$

3×3
f×f

$=$

6×6

p=2

6×6 → 8×8
n×n

n-f+1 × n-f+1
6-3+1=4

~~4×4~~

p = padding = 1

n+2p-f+1 × n+2p-f+1
6+2-3+1 × ___ = 6×6

Valid and Same convolutions

↳ no padding

"Valid": $n \times n \quad * \quad f \times f \quad \rightarrow \quad n-f+1 \times n-f+1$

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

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

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

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

$3 \times 3 \quad p = \frac{3-1}{2} = 1$

$5 \times 5 \quad p = 2$

f is usually odd

1x1

3x3

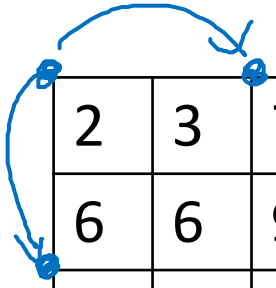
5x5

7x7

Convolutional Neural Networks

Strided convolutions

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

stride $= 2$

$=$

91	100	83
69	91	127
44	72	74

3×3

$\lfloor z \rfloor = \text{floor}(z)$

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

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 $\quad$ $f \times f$ filter

padding p $\quad$ stride s

Output size:

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

Technical note on cross-correlation vs. convolution

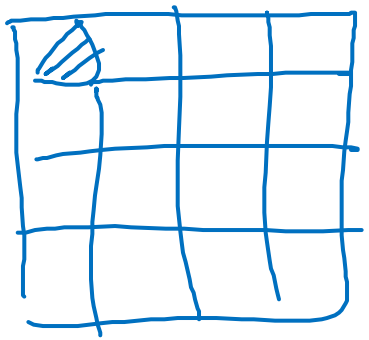
Convolution in math textbook:

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

*

3	4	5
1	0	2
-1	9	7

=



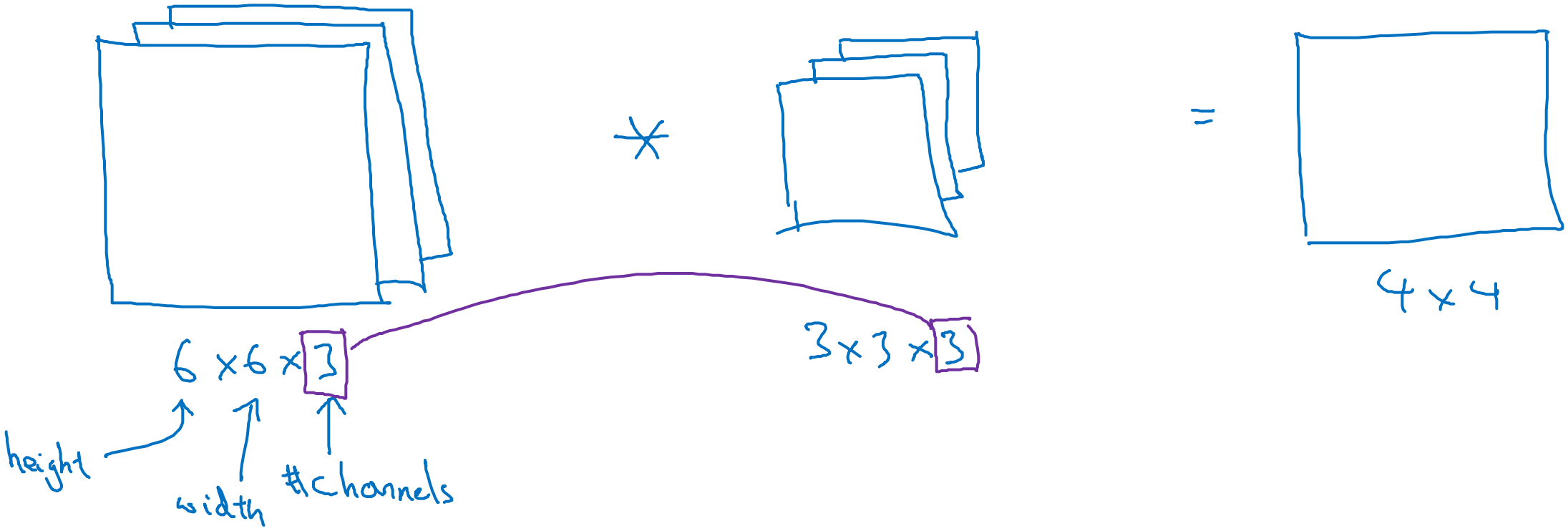
7	2	5
9	0	4
-1	1	3

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

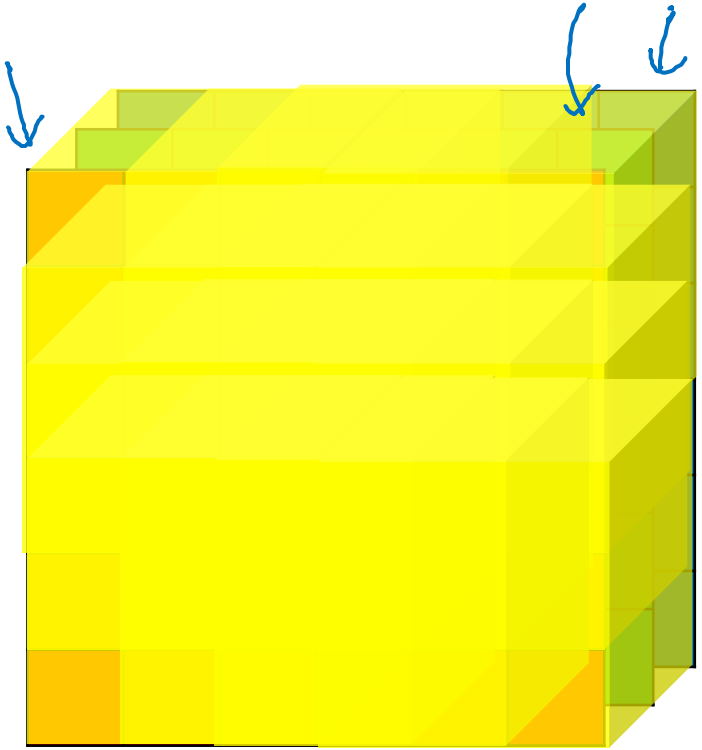
Convolutional Neural Networks

Convolutions over volumes

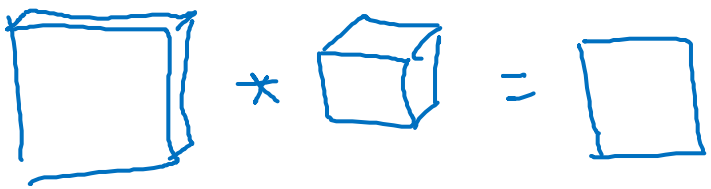
Convolutions on RGB images

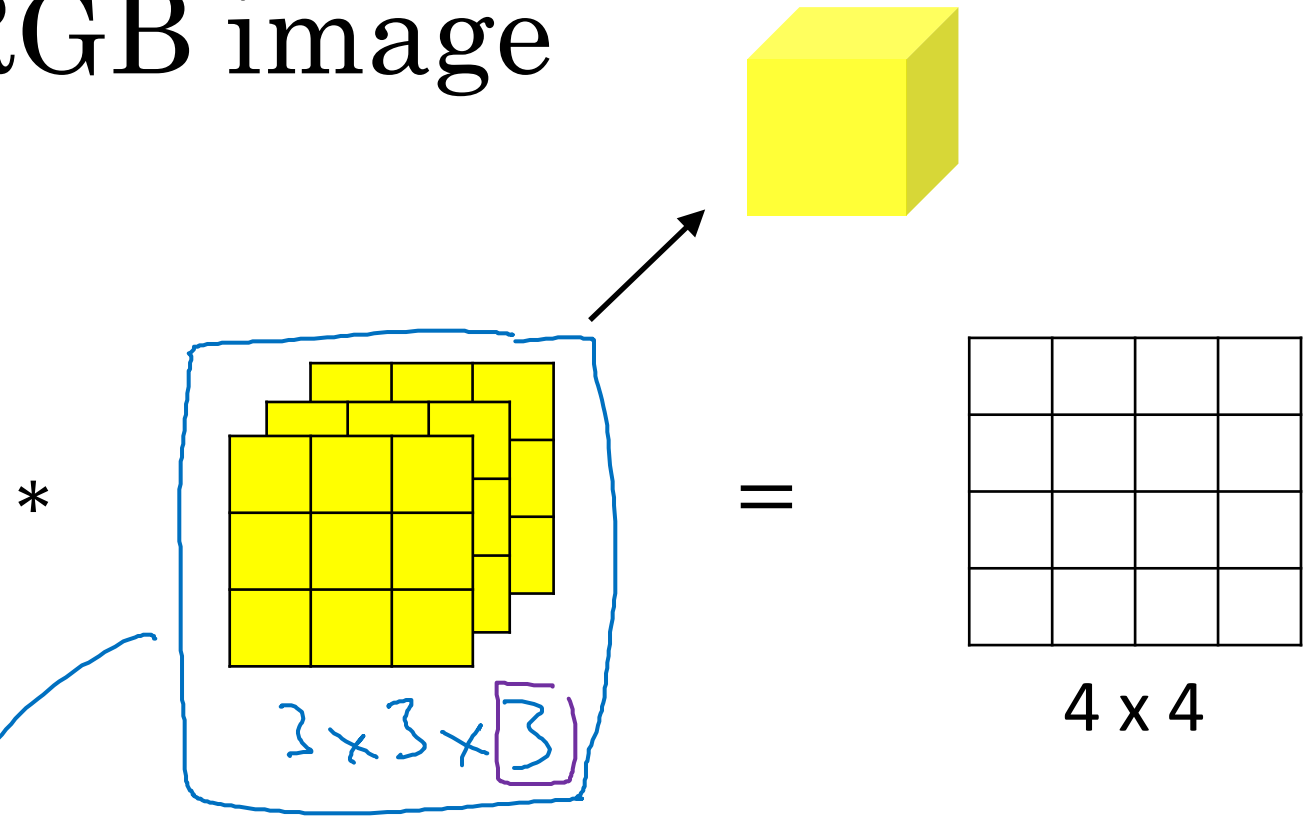


Convolutions on RGB image

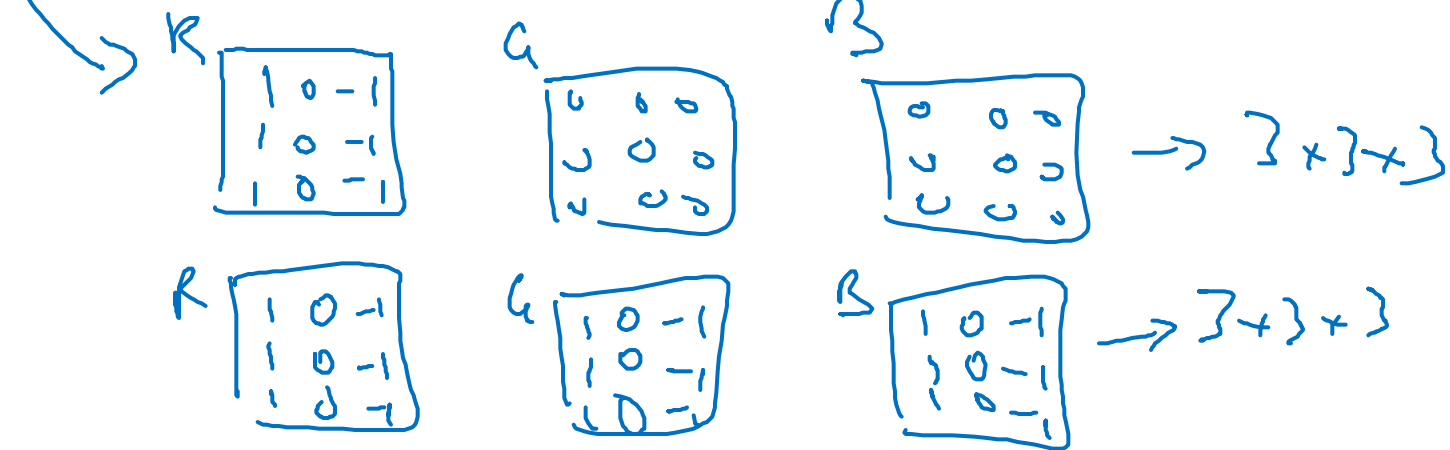


6 x 6 x 3

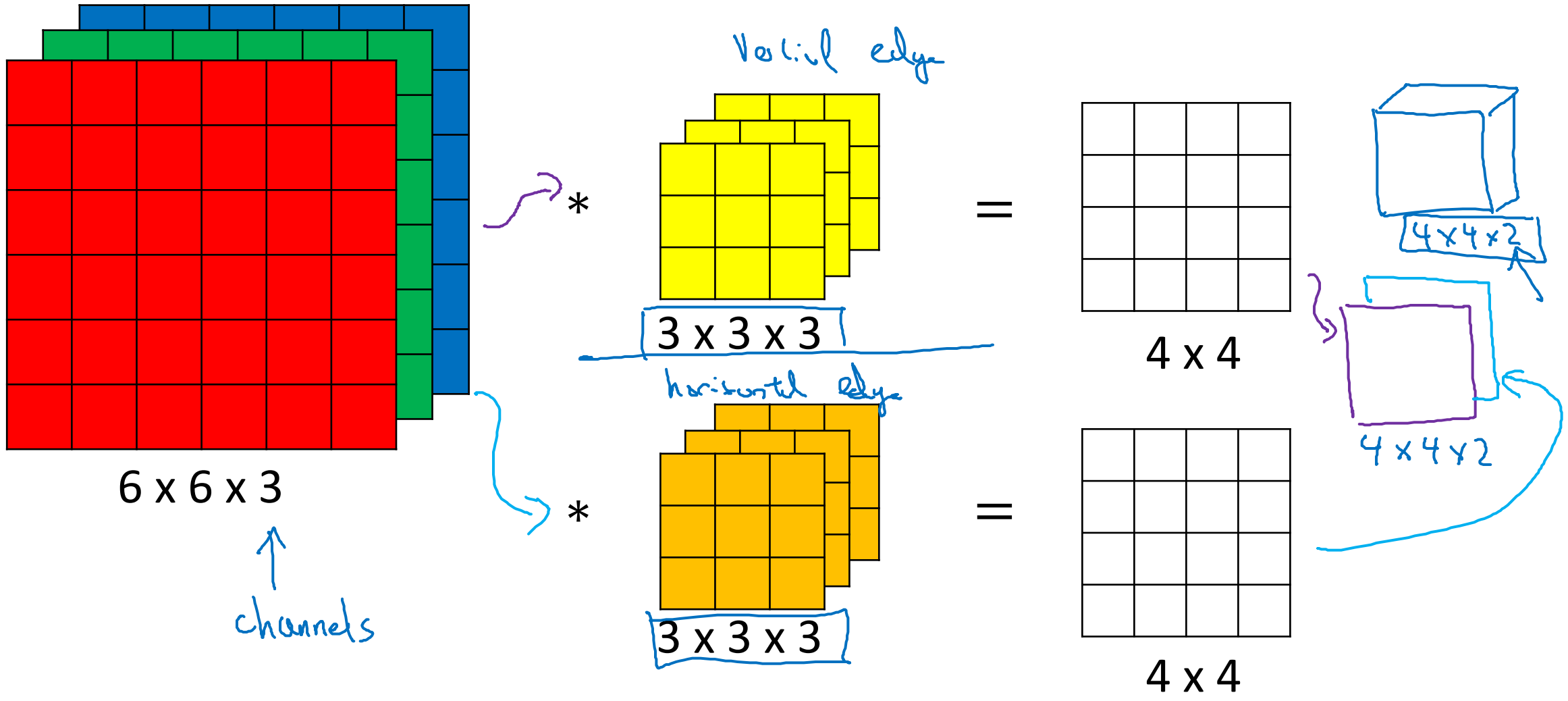




27 numbers



Multiple filters



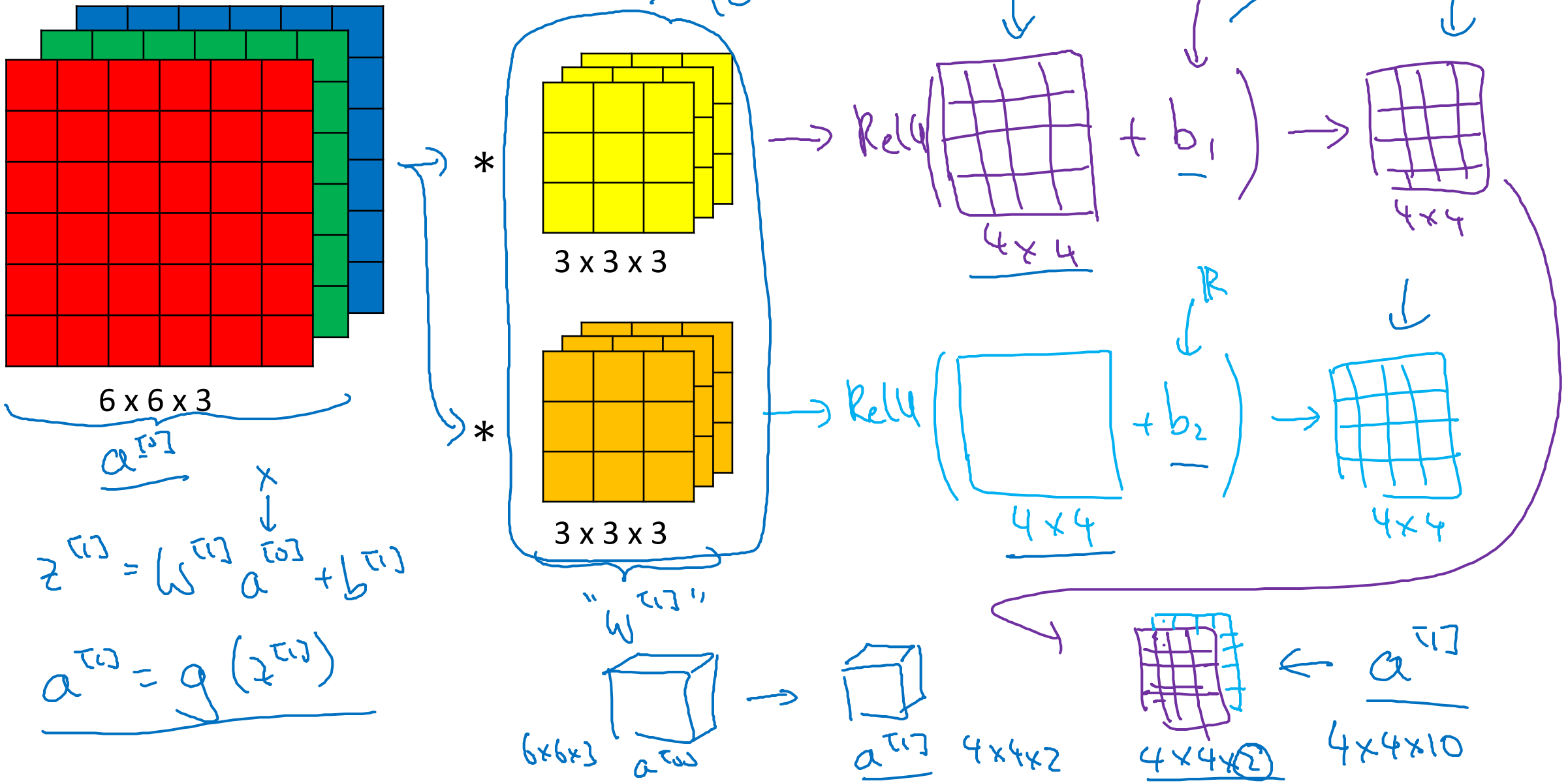
Summary: $n \times n \times n_c$ $*$ $f \times f \times n_c$ $\rightarrow$ $n-f+1 \times n-f+1 \times n_c'$

$6 \times 6 \times 3$ $\quad$ $3 \times 3 \times 3$ $\quad$ $4 \times 4 \times 2$ (n_c' = # filters)

Convolutional Neural Networks

One layer of a convolutional network

Example of a layer



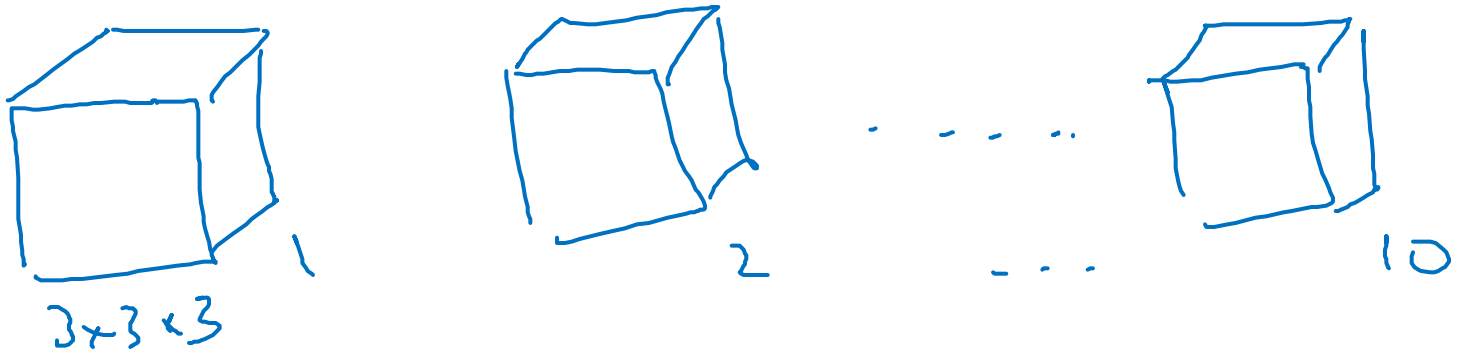
$a^{[0]}$

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

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

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?



3x3x3

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

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

Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$ ← #filters in layer l.

bias: $n_c^{[l]}$ $\quad n_c \times n_H \times n_W$

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

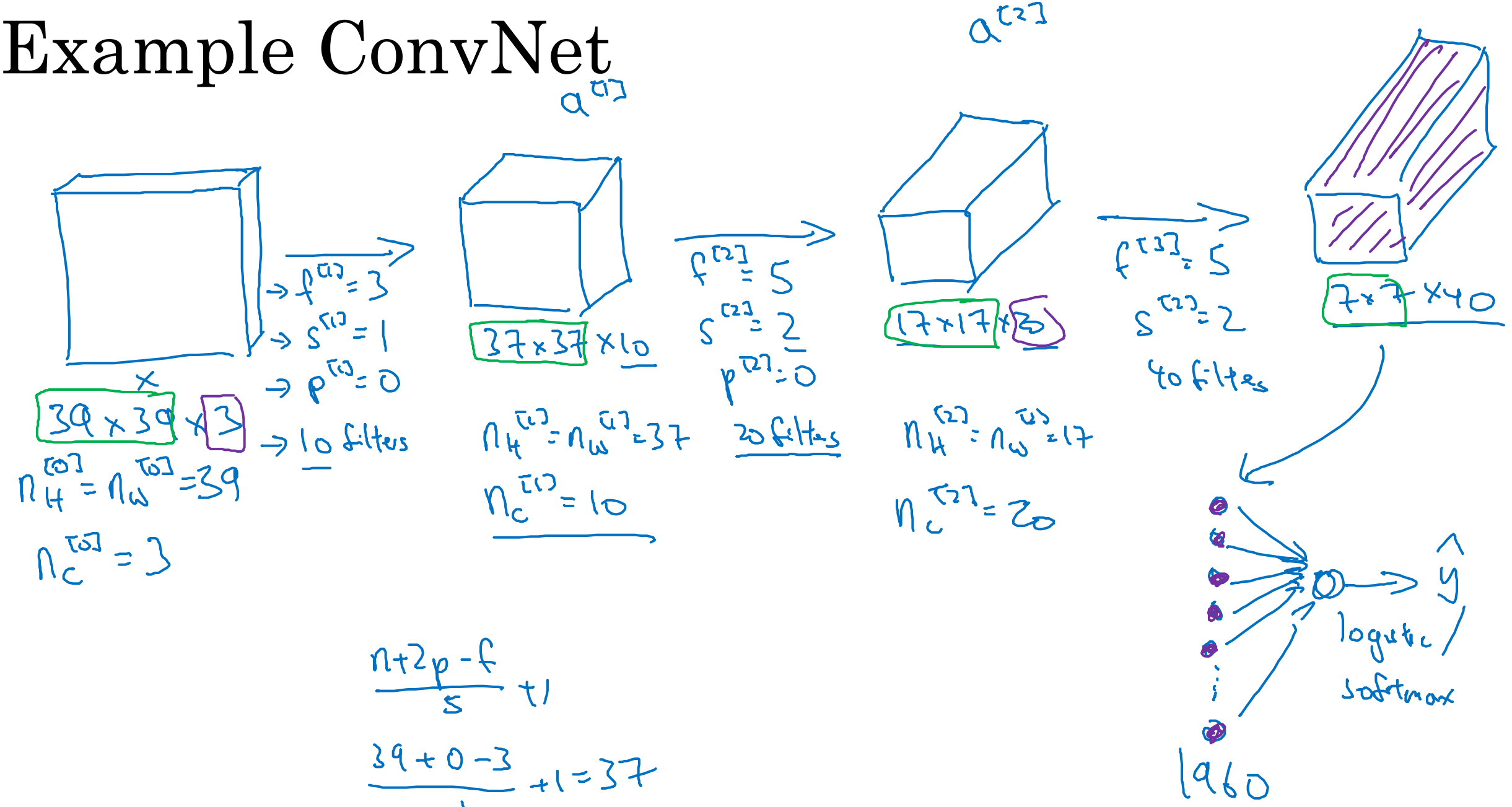
Output: $n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

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

Convolutional Neural Networks

A simple convolution network example

Example ConvNet



Types of layer in a convolutional network:

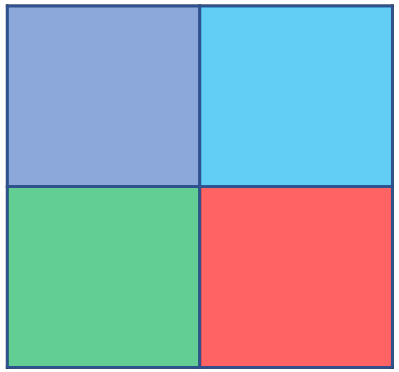
- Convolution
- Pooling
- Fully connected

Convolutional Neural Networks

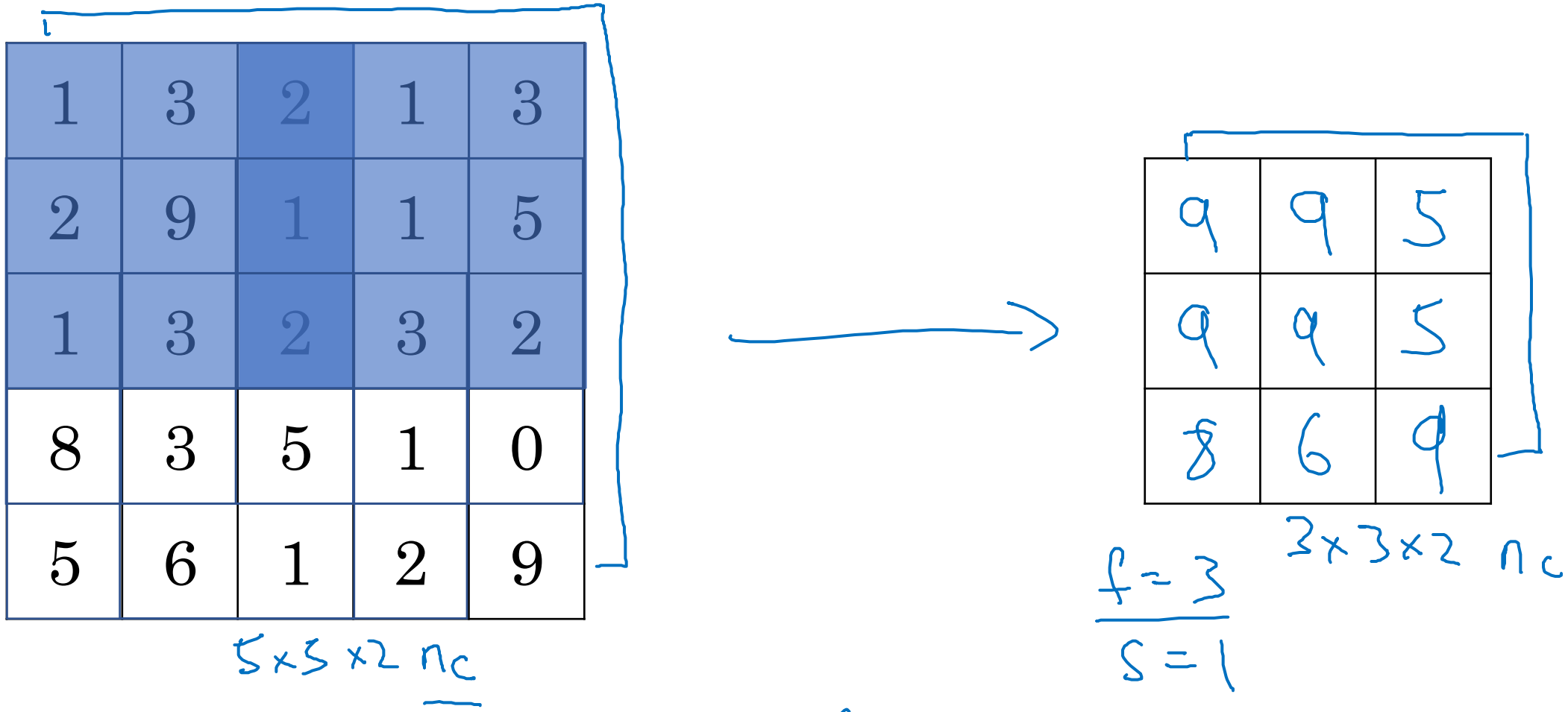
Pooling layers

Pooling layer: Max pooling

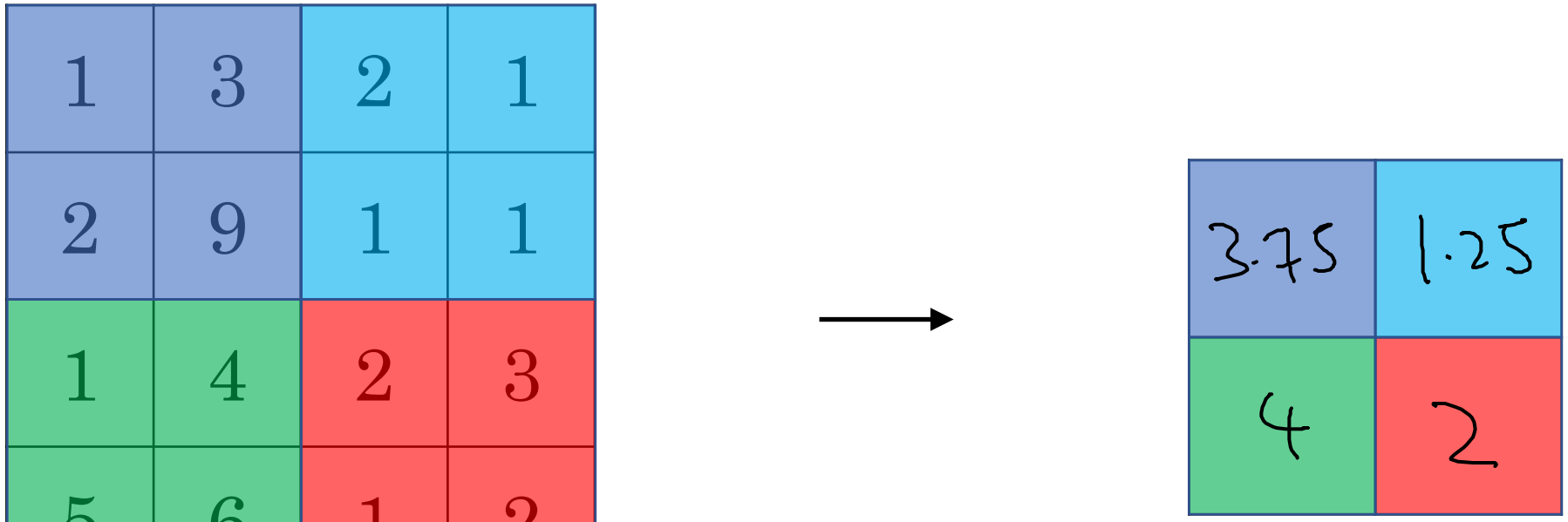
1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2



Pooling layer: Max pooling



Pooling layer: Average pooling



Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

$n_H \times n_w \times n_c$

$\downarrow$

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

$\times n_c$

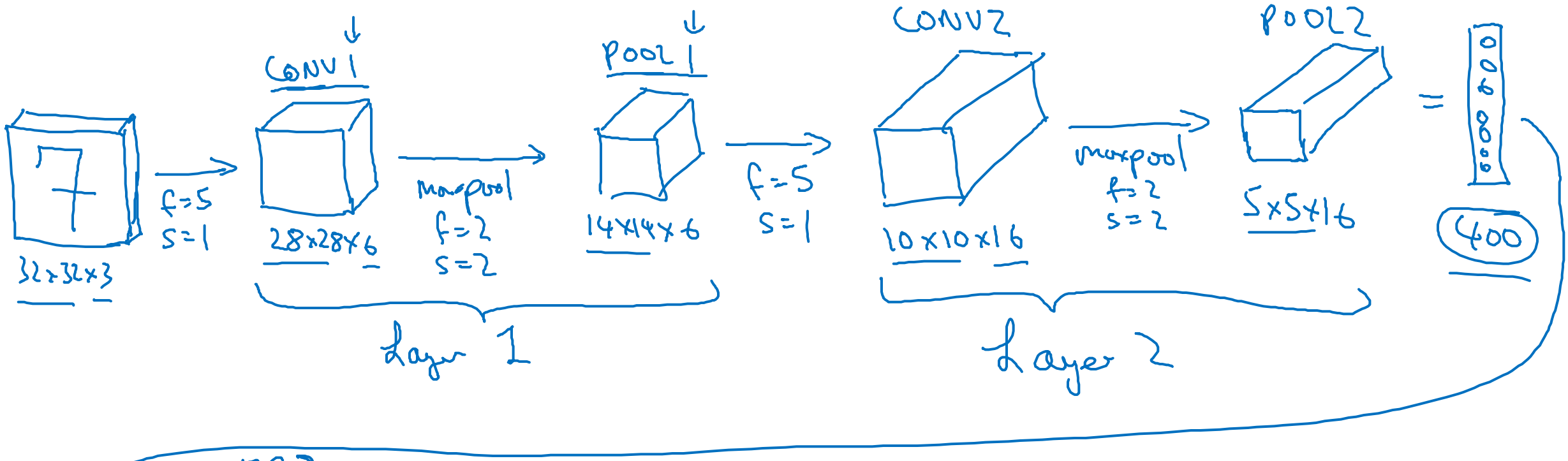
~~p : padding~~

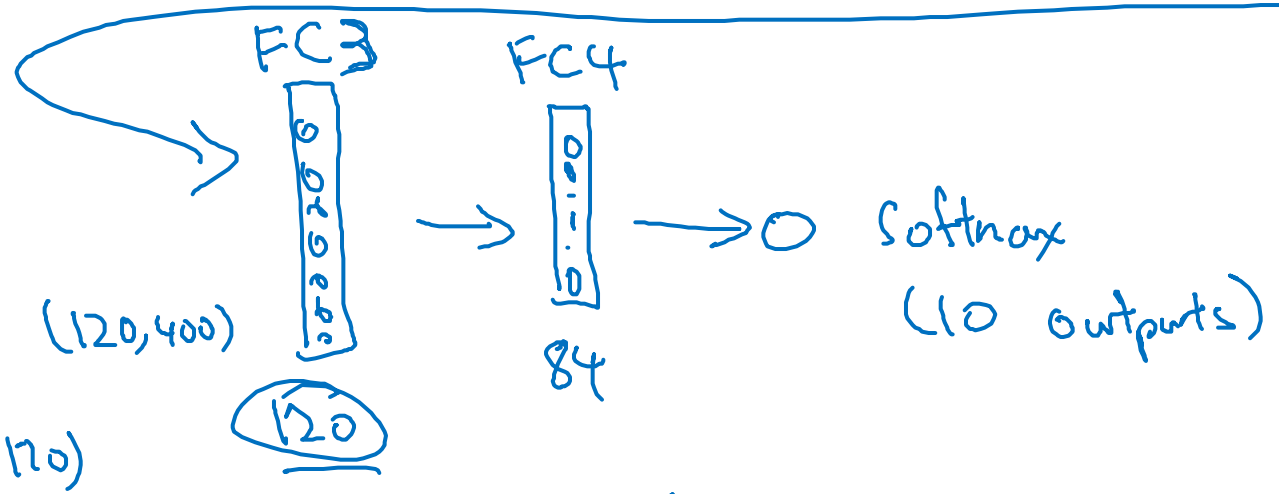
No parameters to learn!

Convolutional
Neural Networks

Convolutional neural network example

Neural network example (LeNet-5)





Neural network example

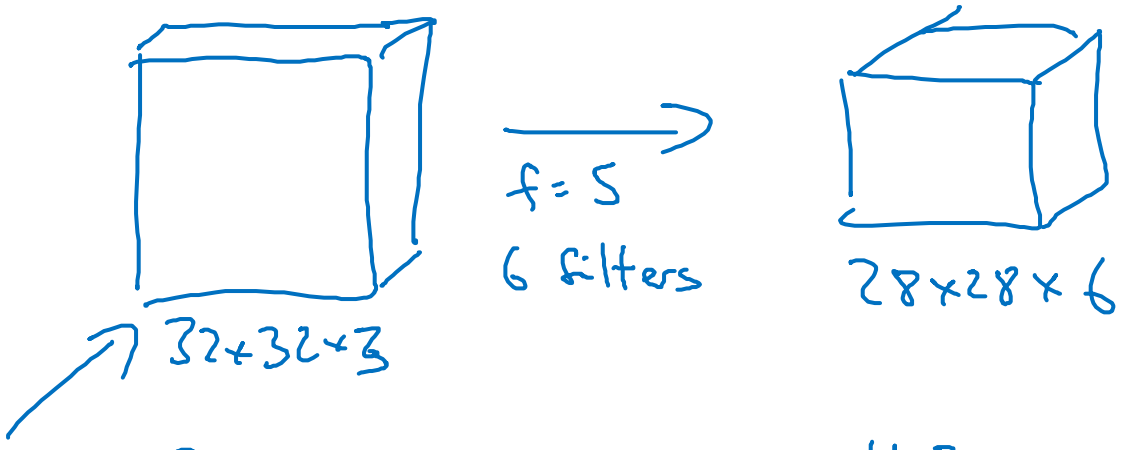
	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	3,072	0
CONV1 (f=5, s=1)	(28,28,8)	6,272	456
POOL1	(14,14,8)	1,568	0
CONV2 (f=5, s=1)	(10,10,16)	1,600	1516
POOL2	(5,5,16)	400	0
FC3	(120,1)	120	48120
FC4	(84,1)	84	10164
Softmax	(10,1)	10	850

Andrew Ng

Convolutional Neural Networks

Why convolutions?

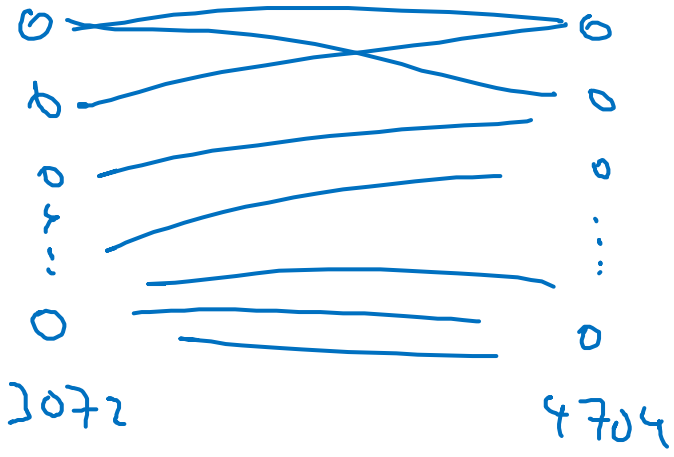
Why convolutions



5 x 5 — 25

26

6 x 26 = 156 parameters



3,072 x 4,704 ≈ 14M

Why convolutions

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

3×3

=

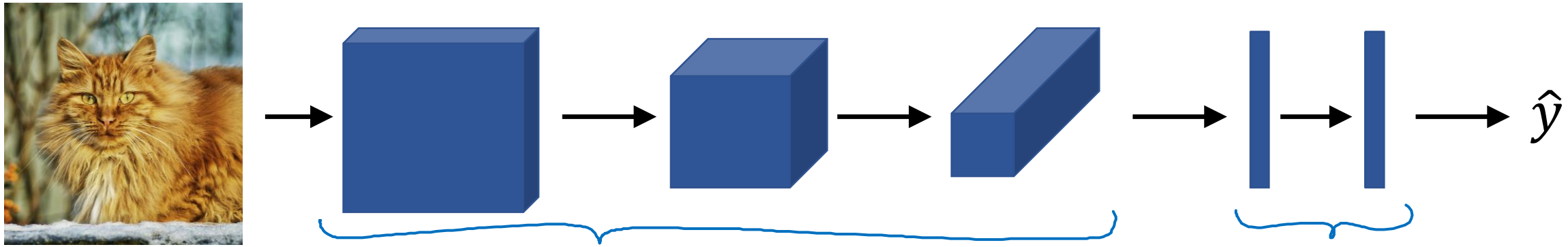
0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

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.

Putting it together

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



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