

الشبكات العصبية

محاضرة 7

Convolutional Neural Networks

د. فادي متوج

Computer Vision Problems

Image Classification



64x64



Cat? (0/1)

Object detection



Neural Style Transfer



Deep Learning on large images



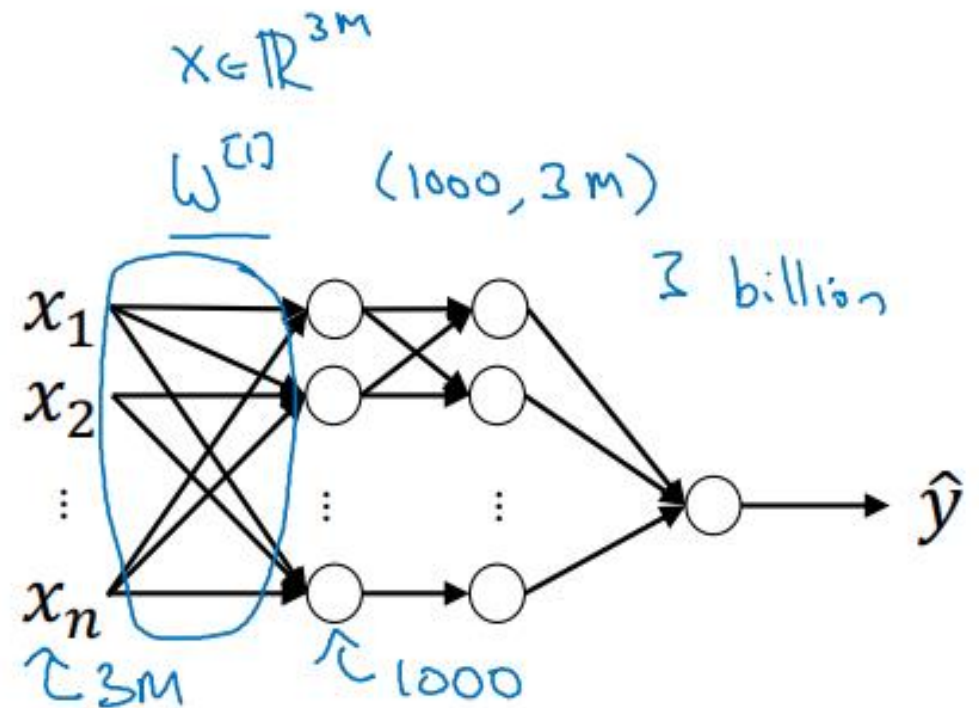
64x64 x 3

→ Cat? (0/1)

12288



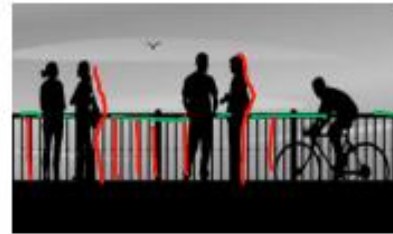
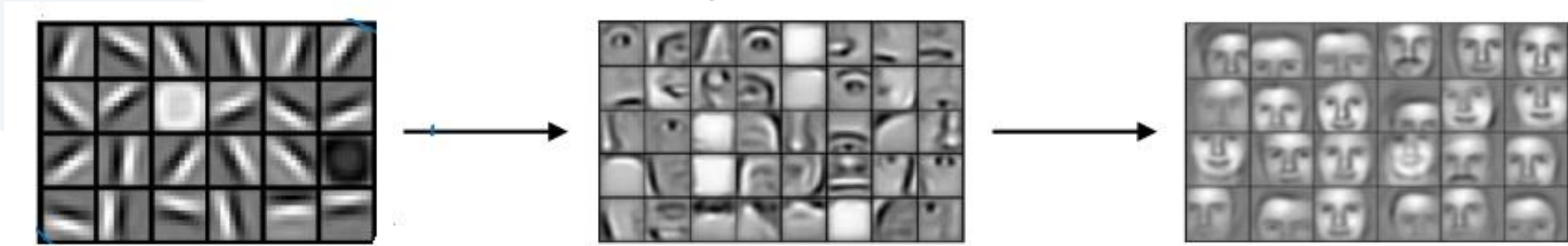
1000 x 1000 x 3
= 3 million



Convolutional Neural Networks

Edge detection example

Edge detection example



vertical edges



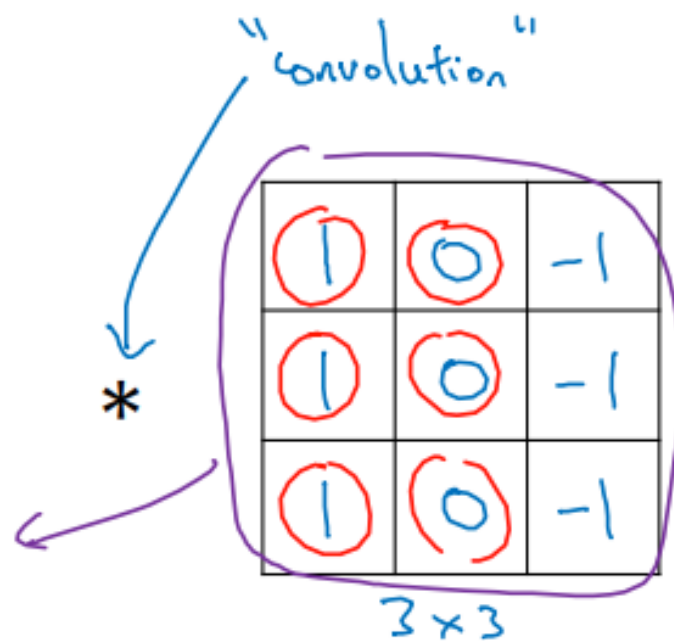
horizontal edges



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

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

6x6



=

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

4x4

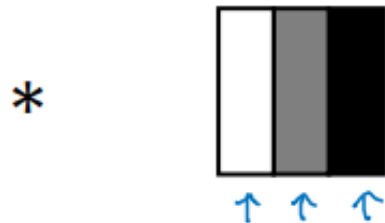
Vertical edge detection

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	<u>10</u>	<u>10</u>	<u>0</u>	0	0
10	<u>10</u>	<u>10</u>	<u>0</u>	0	0
10	<u>10</u>	<u>10</u>	<u>0</u>	0	0

6x6

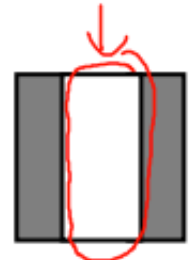


$$\begin{matrix}
 * & \downarrow \\
 \begin{matrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{matrix} & & = & & \begin{matrix} 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \\ 0 & 30 & 30 & 0 \end{matrix} \\
 & 3 \times 3 & & & \uparrow 4 \times 4
 \end{matrix}$$



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

4x4



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




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




*

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



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

6x6

*

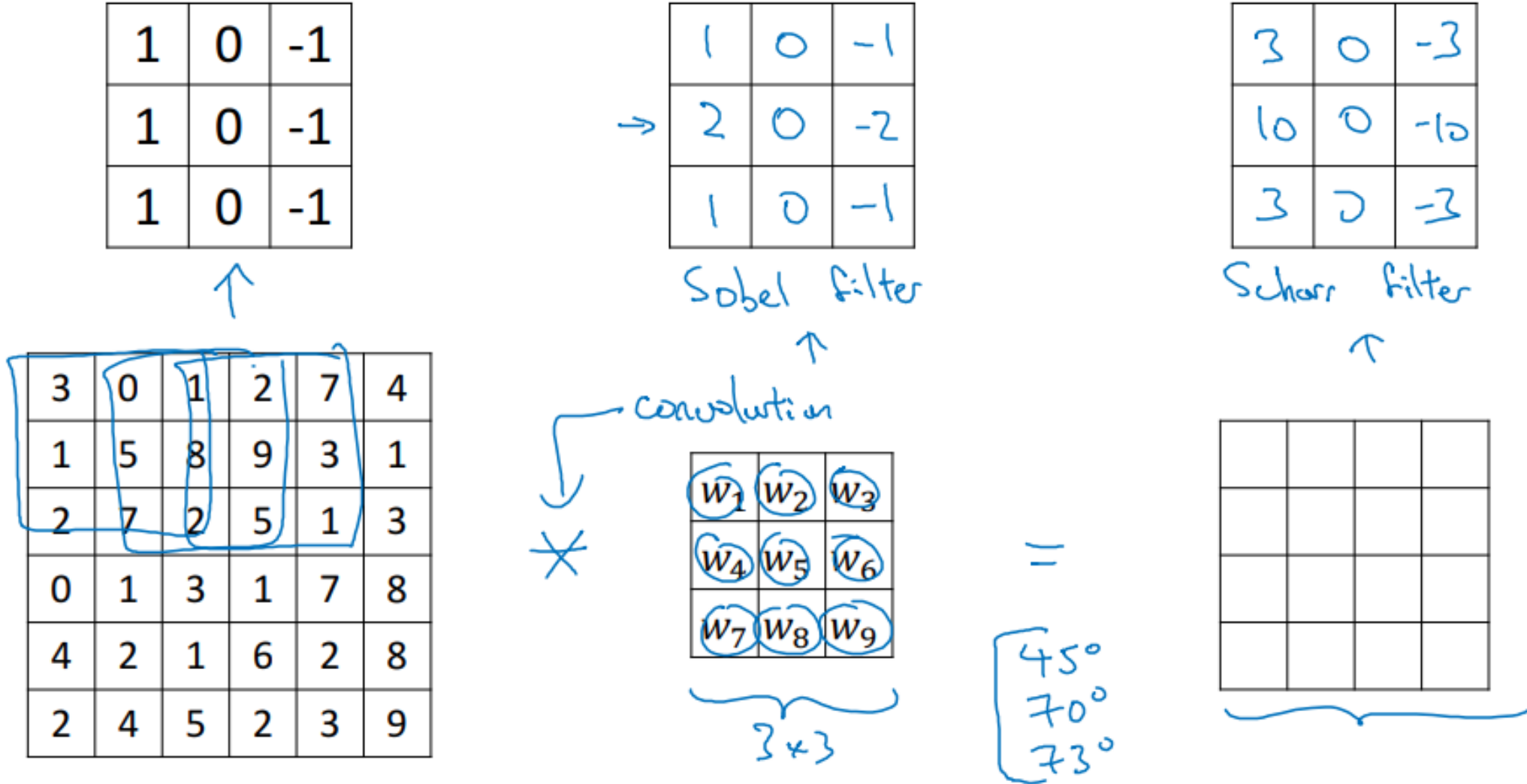
1	1	1
0	0	0
-1	-1	-1

=

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



Learning to detect edges



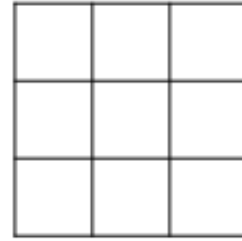
Convolutional Neural Networks

Padding



Padding

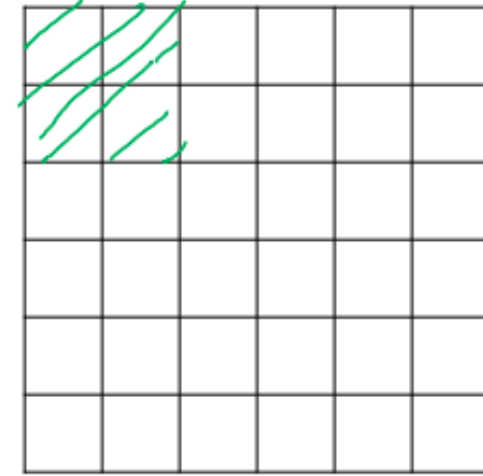
- Shrinky output
- throw away info from edge



3x3
f x f

*

=



6x6

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

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

$$6 - 3 + 1 = 4$$

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

$$6 + 2 - 3 + 1 \times \underline{\quad} = 6 \times 6$$

p = padding = 1

~~4x4~~

Valid and Same convolutions



→ no padding

“Valid”:

$$n \times n * f \times f \rightarrow \frac{n-f+1}{1} \times n-f+1$$

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

“Same”:

Pad so that output size is the same as the input size.

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

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

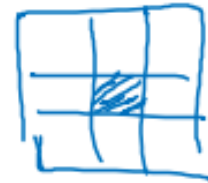
$$3 \times 3 \quad p = \frac{3-1}{2} = 1$$

$$p = 2$$

$$\left. \begin{array}{l} 5 \times 5 \\ f = 5 \end{array} \right\} p = 2$$

f is usually odd

1x1
3x3
5x5
7x7



Convolutional Neural Networks

Strided convolutions

Strided convolution



جامعة
المنارة

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

7x7

*

3	4	4
1	0	2
-1	0	3

3x3

=

91	100	83
69	91	127
44	72	74

3x3

stride = 2

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

$n \times n$ * $f \times f$
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 $f \times f$ filter

padding p stride s

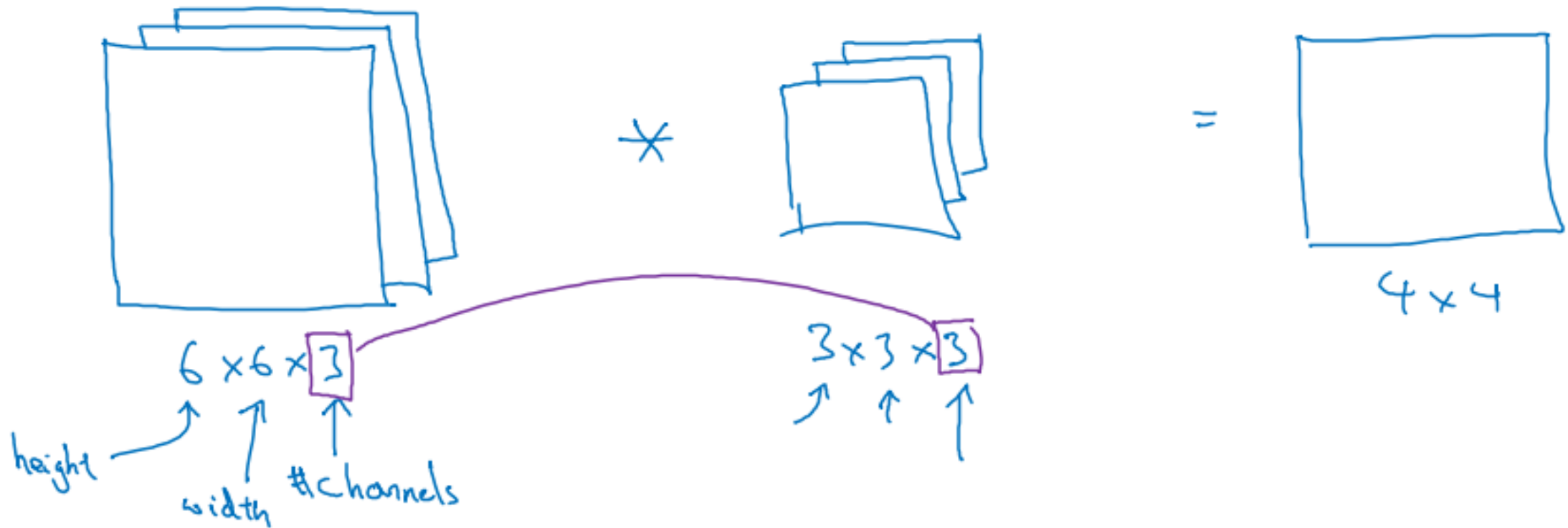
Output Size:

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

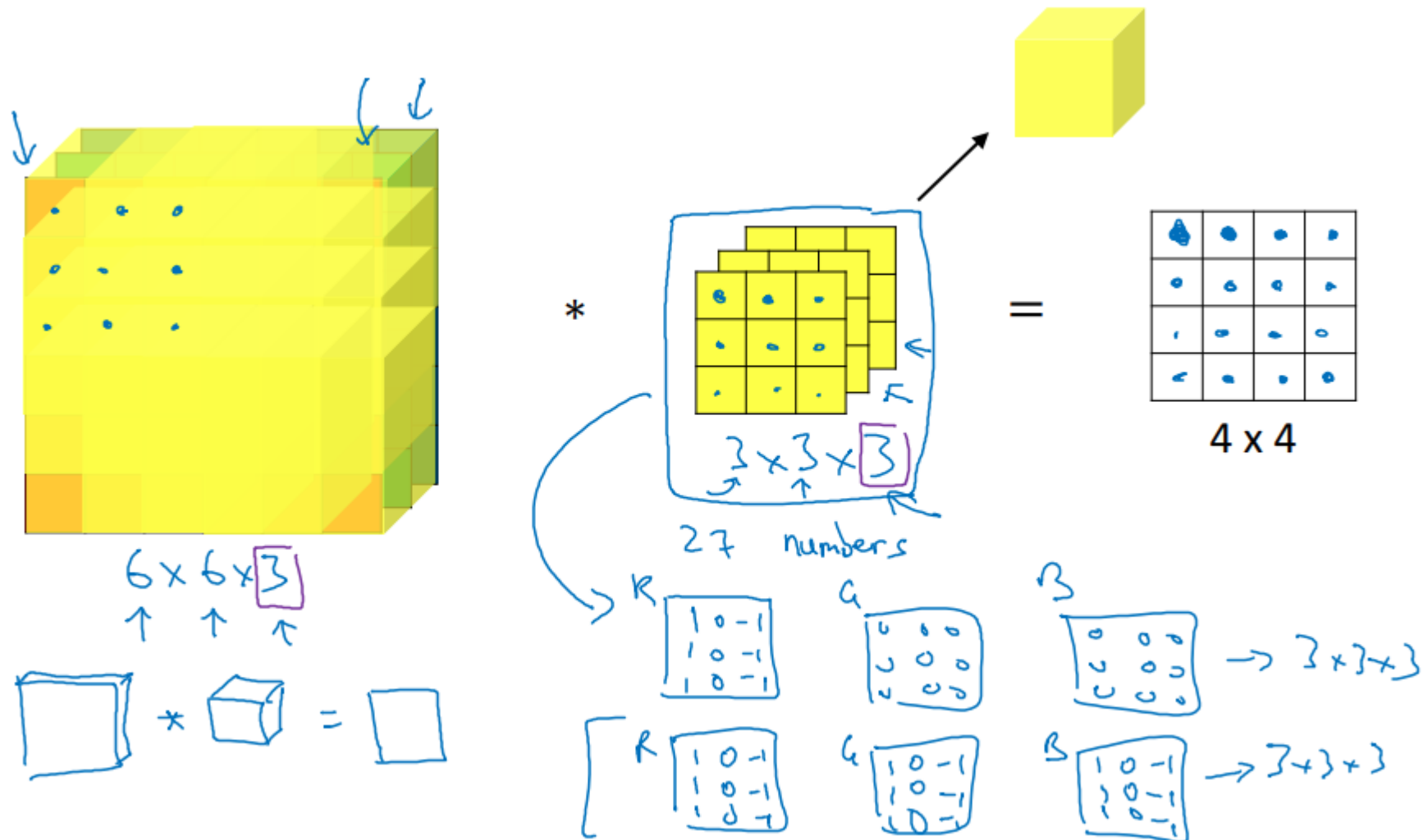
Convolutional Neural Networks

Convolutions over volumes

Convolutions on RGB images



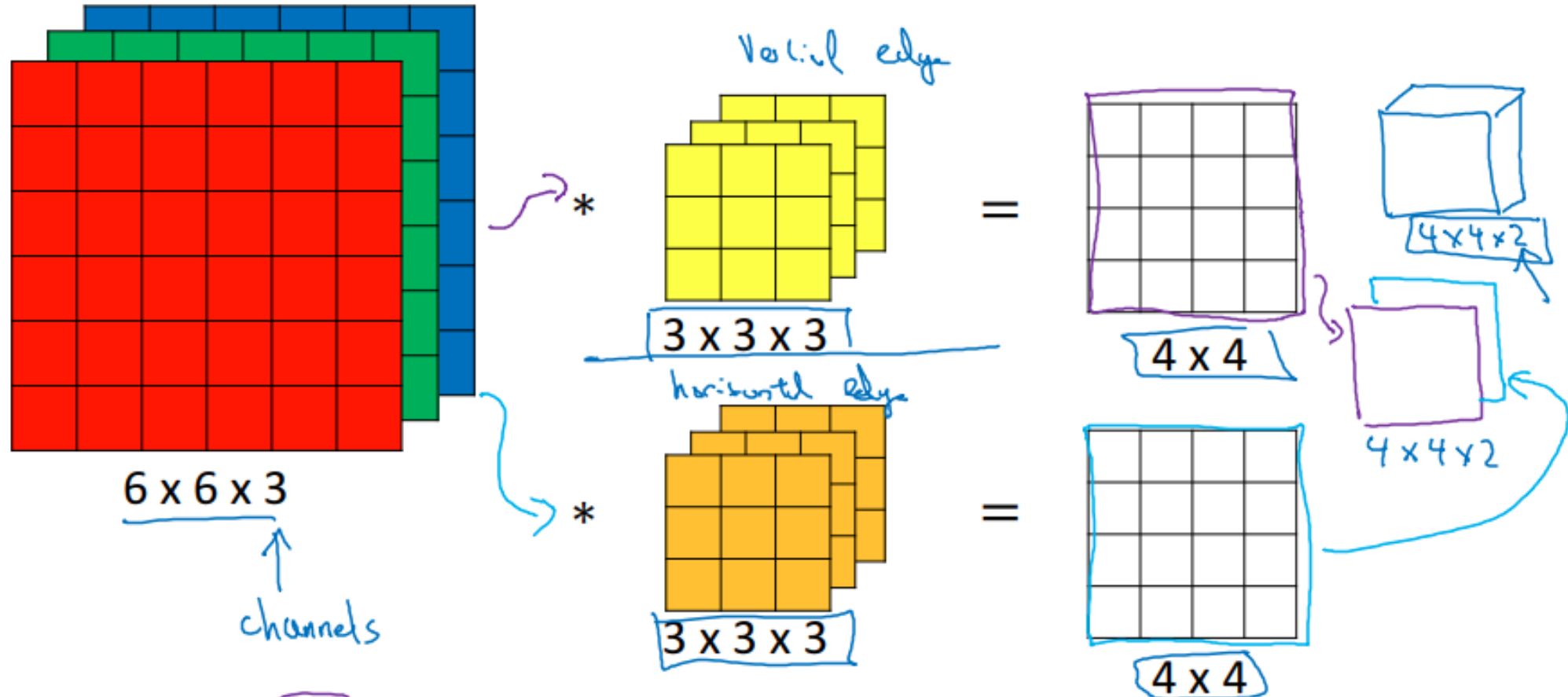
Convolutions on RGB image



Multiple filters



جامعة
المنارة



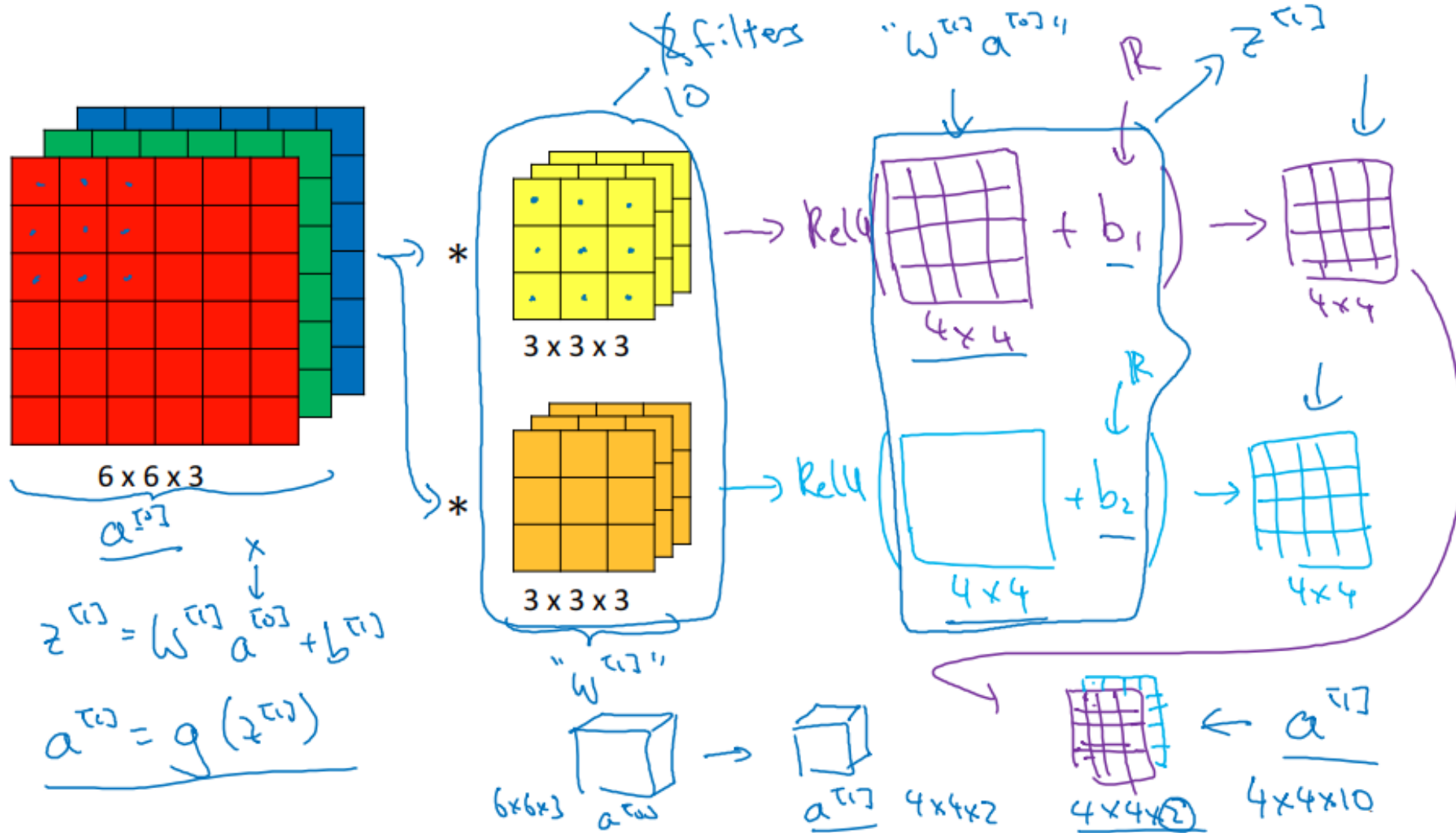
Summary: $n \times n \times n_c \times f \times f \times n_c \rightarrow \frac{n-f+1}{4} \times \frac{n-f+1}{4} \times n_c' \uparrow \# \text{ filters}$

$6 \times 6 \times 3 \times 3 \times 3 \times 3 \rightarrow 4 \times 4 \times 2$

Convolutional Neural Networks

**One layer of a
convolutional
network**

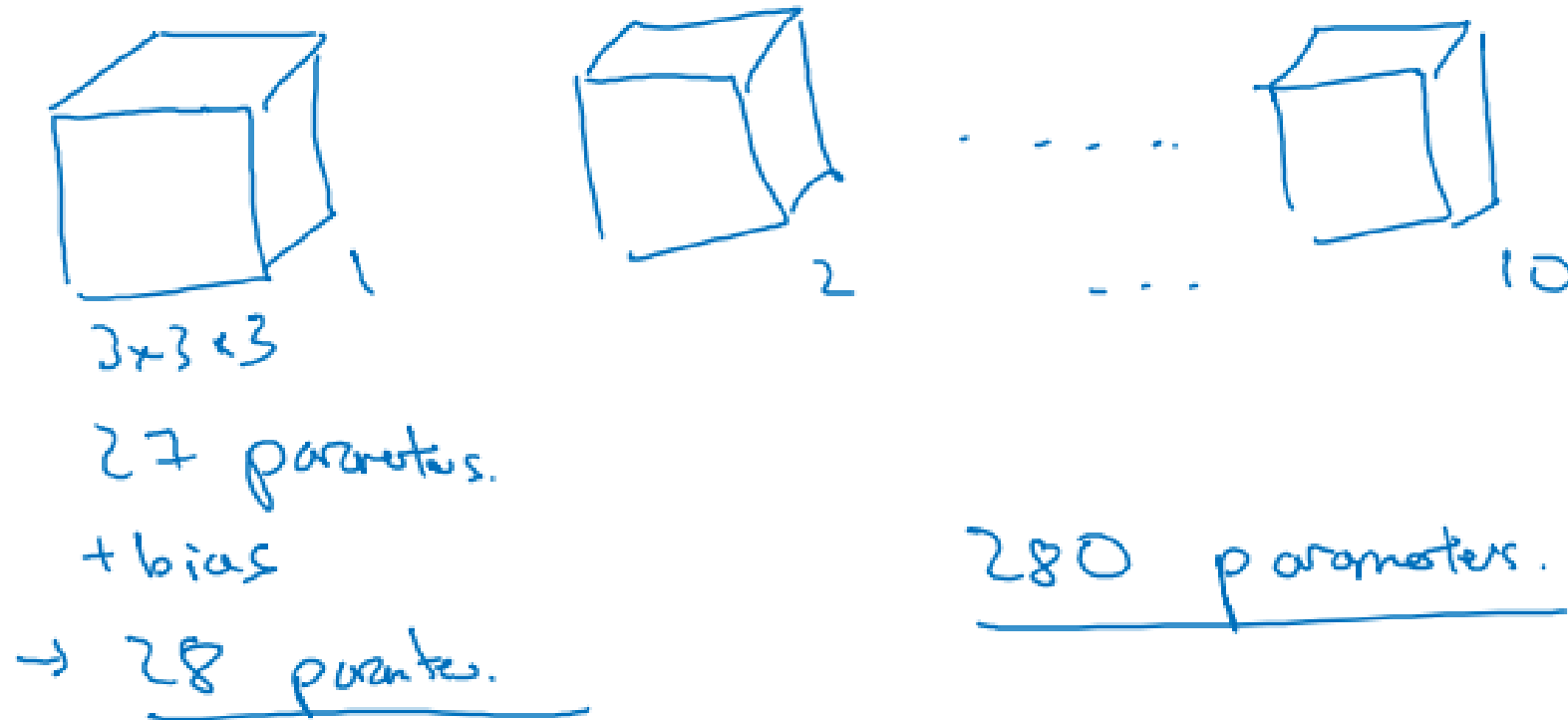
Example of a layer



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?



Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

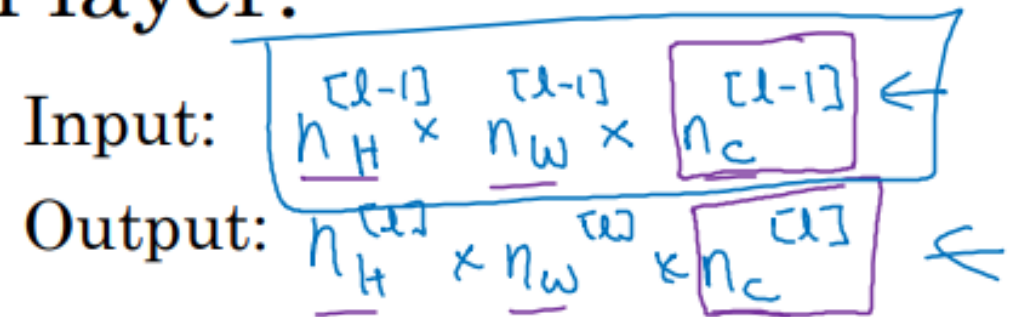
$n_c^{[l]}$ = number of filters

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



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

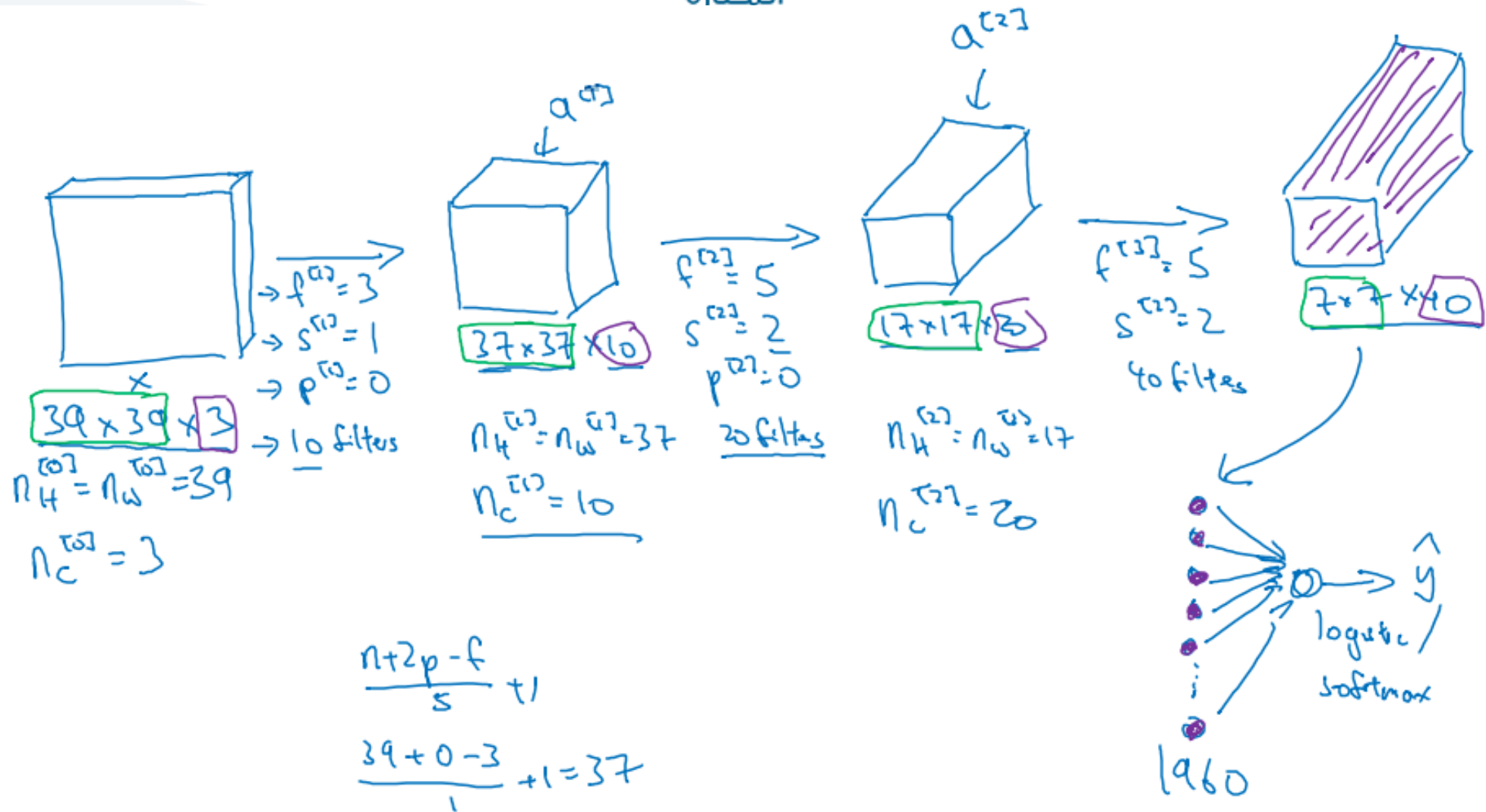
$$A^{[l]} \rightarrow n \times \underbrace{n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}}_{n_c^{[l]} \times n_H^{[l]} \times n_W^{[l]}}$$

#filters in layer l.

Convolutional Neural Networks

A simple convolution network example

Example ConvNet



Types of layer in a convolutional network:

- Convolution
- Pooling
- Fully connected

Convolutional Neural Networks

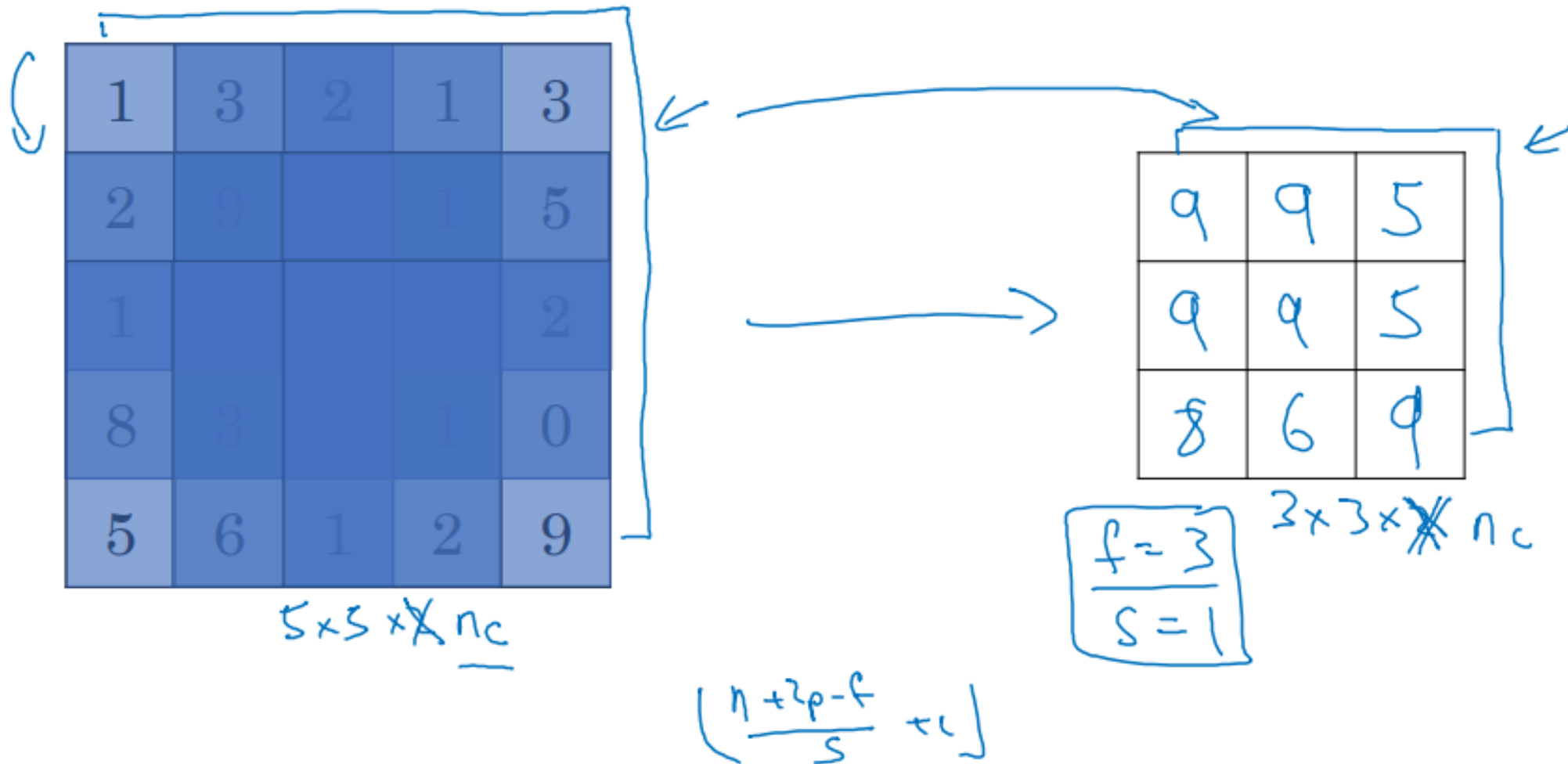
Pooling layers

Pooling layer: Max pooling

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

2	3
3	2

Pooling layer: Max pooling



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

$$f=2$$
$$s=2$$

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

Summary of pooling



Hyperparameters:

f : filter size

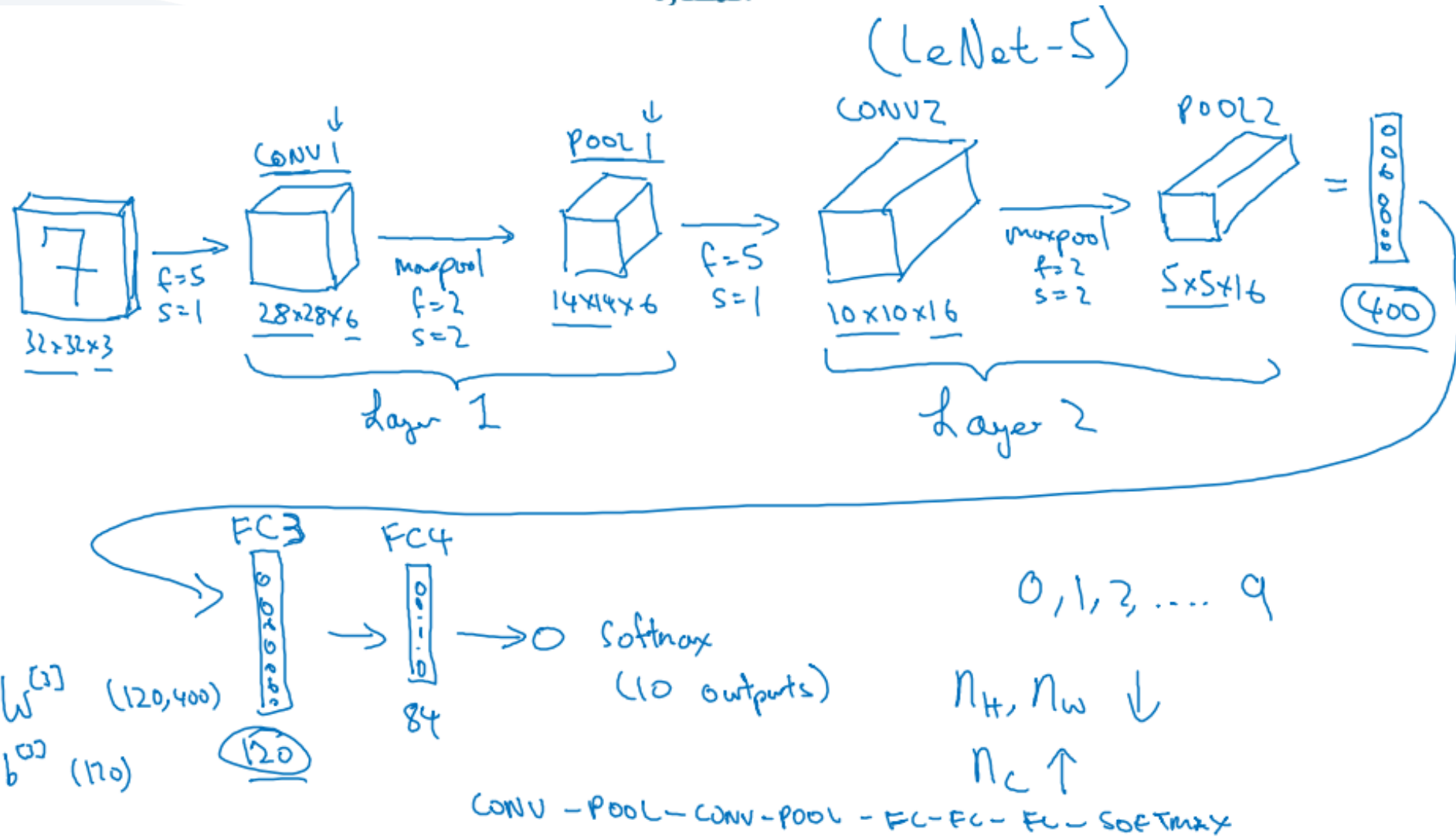
s : stride

Max or average pooling

Convolutional Neural Networks

Convolutional neural network example

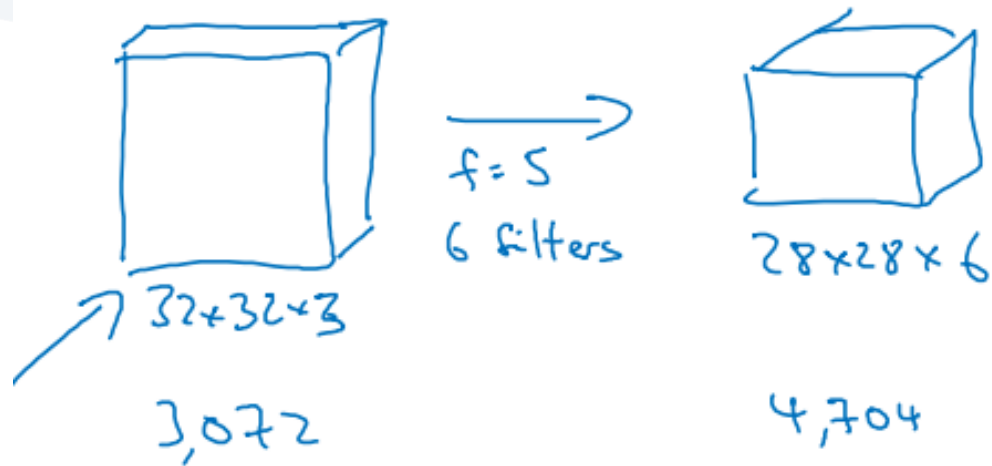
Neural network example



Neural network example

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

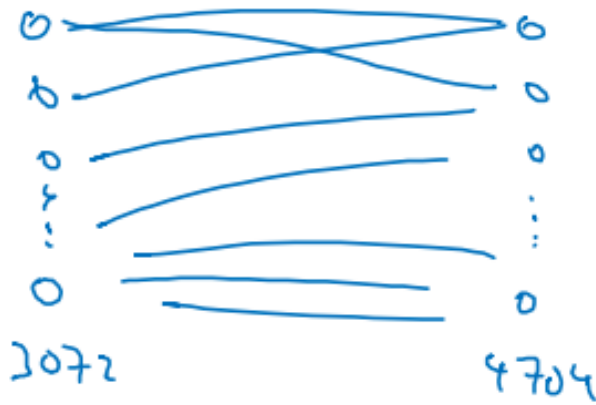
Why convolutions



$$5 \times 5 = 25$$

$$+ 26$$

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



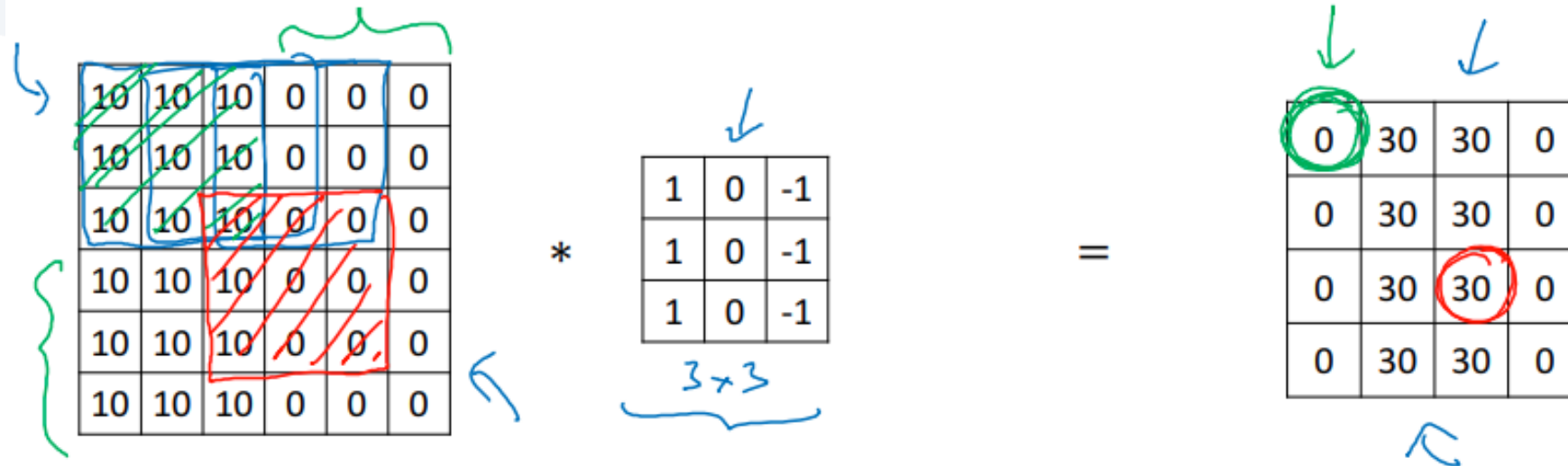
$$3,072 \times 4,704 \approx \underline{14M}$$

Why convolutions



جامعة
المنارة

translation invariance

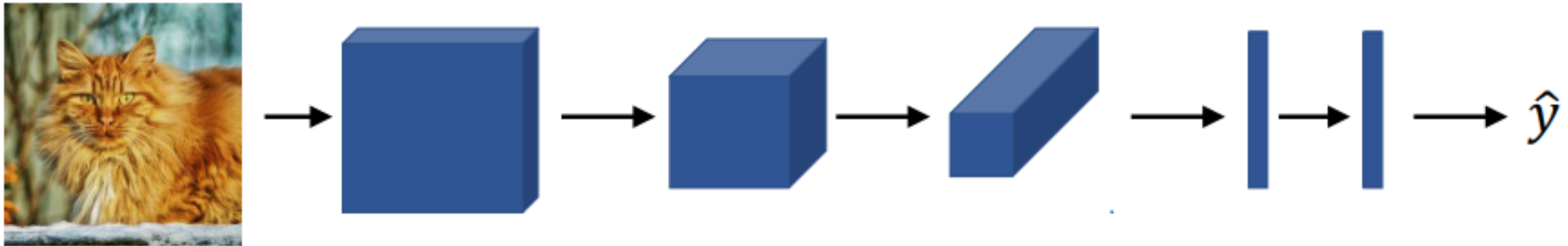


Parameter sharing: A feature detector (such as a vertical edge detector) that's useful in one part of the image is probably useful in another part of the image.

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

Putting it together

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



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

Use gradient descent to optimize parameters to reduce J

Case studies



Classic networks:

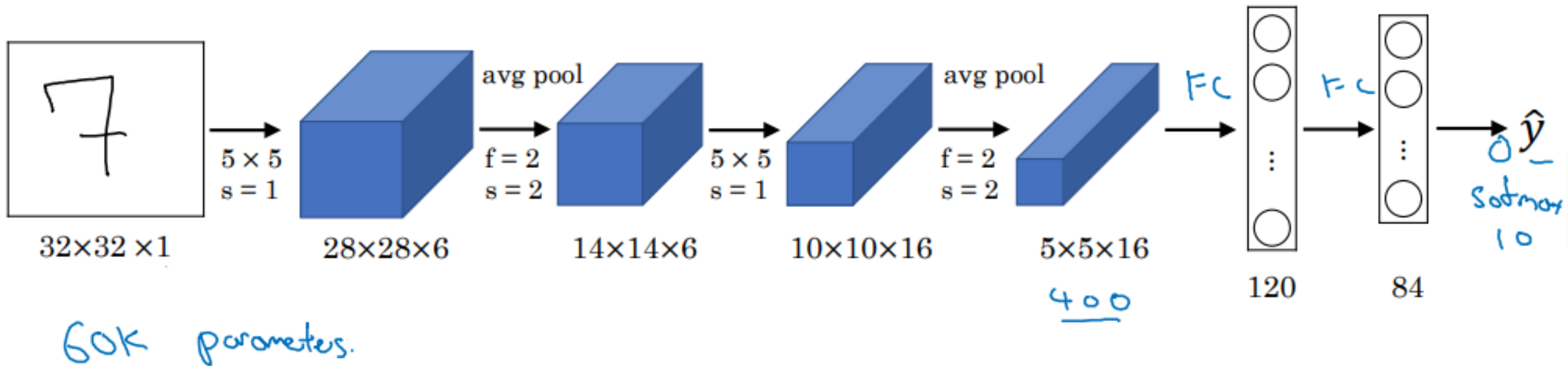
- LeNet-5
- AlexNet
- VGG

ResNet

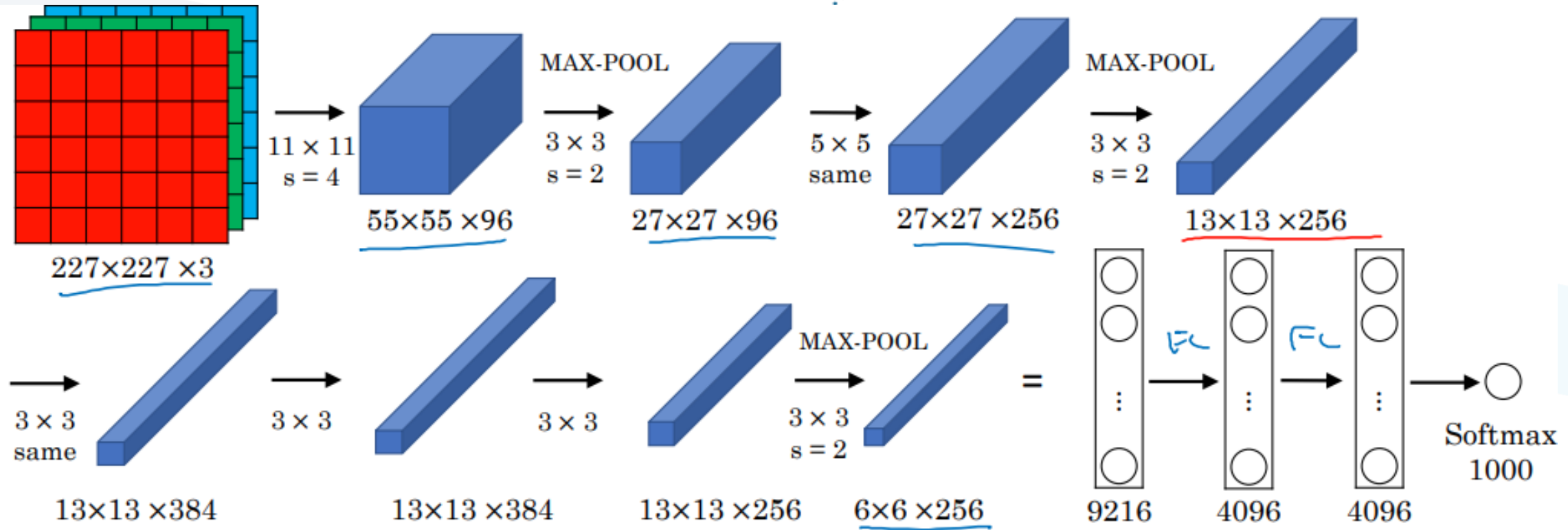
Case studies

Classic networks

LeNet - 5



AlexNet



- Similar to LeNet, but much bigger.

- ReLU

- Multiple GPUs.

- Local Response Normalization (LRN)

9216



~60M parameters

[Krizhevsky et al., 2012. ImageNet classification with deep convolutional neural networks]

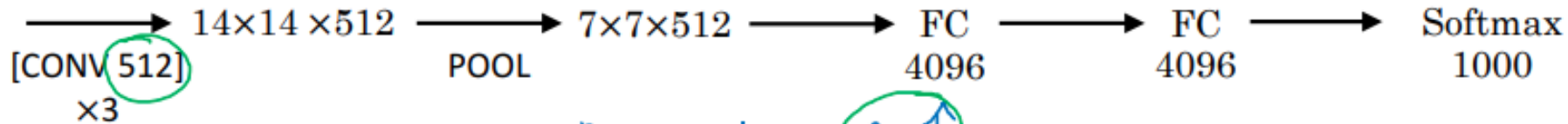
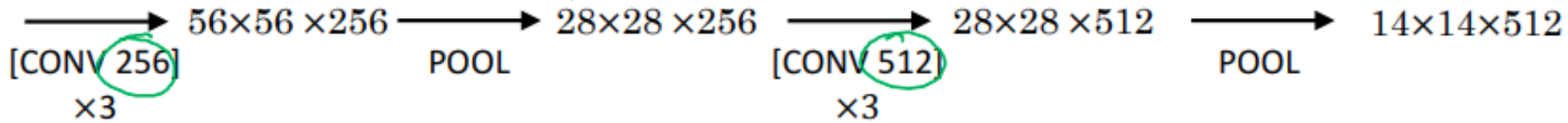
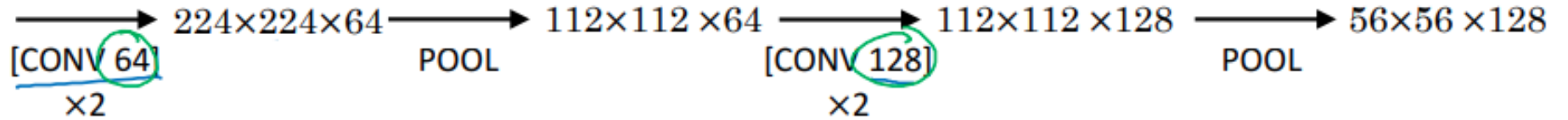
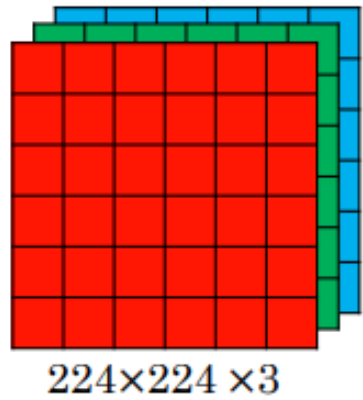
VGG - 16



جامعة

CONV = 3x3 filter, s = 1, same

MAX-POOL = 2x2, s = 2



$n_H, n_W \downarrow$

$n_C \uparrow$

~138M

[Simonyan & Zisserman 2015. Very deep convolutional networks for large-scale image recognition]