Deep learning

Dr. Aissa Boulmerka
a.boulmerka@centre-univ-mila.dz

2023-2024

CHAPTER 8
CONVOLUTIONAL NEURAL NETWORKS (CNNS)
“FOUNDATIONS OF CNN”

Computer Vision Problems

Image Classification

Cat? (0/1)

Object detection

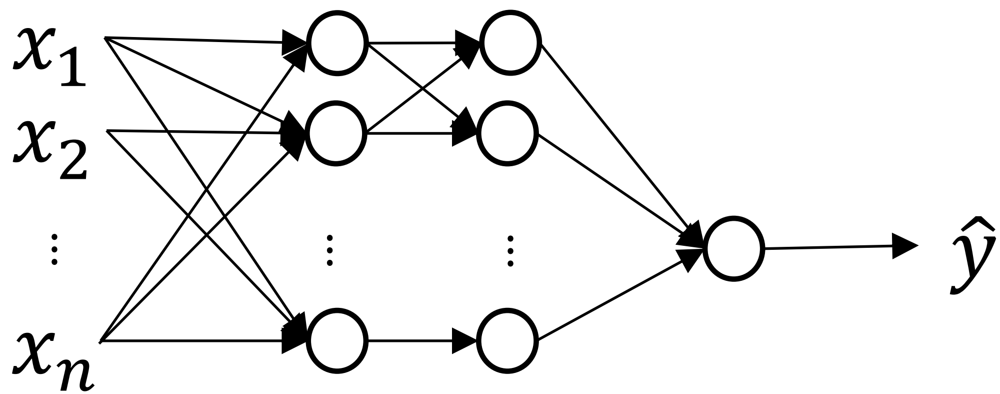
Neural Style Transfer

Deep Learning on large images

Image Classification

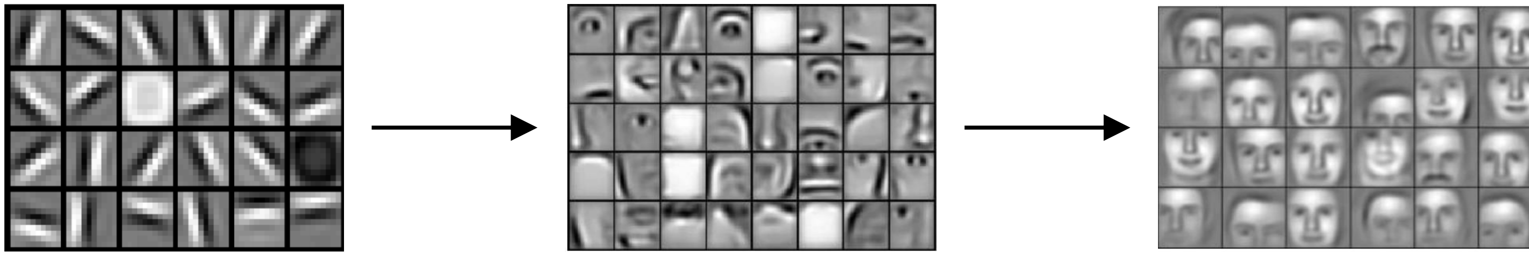
64x64x3

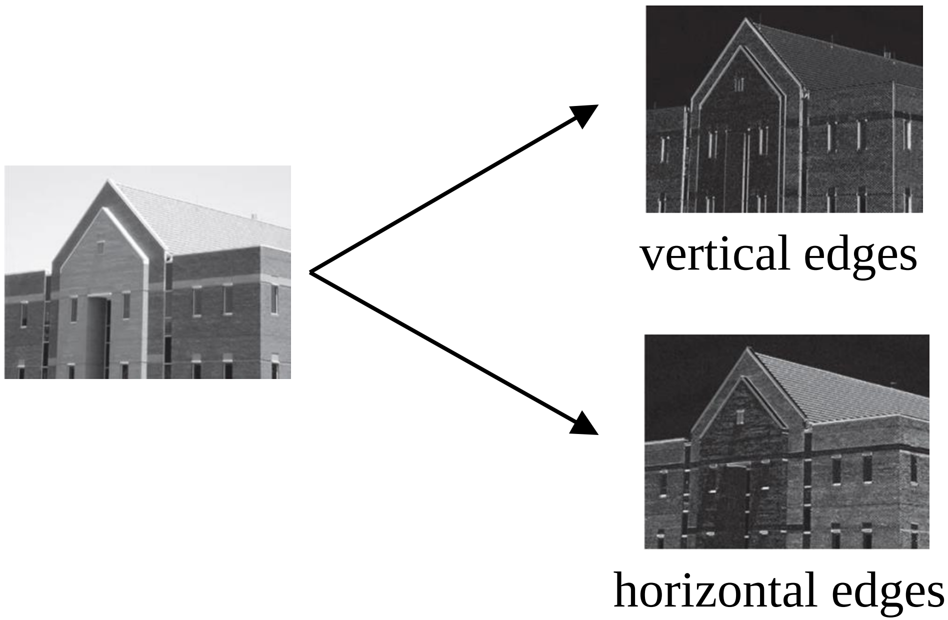
$1000 \times 1000 \times 3 = 3M$



3M 1000 $\Rightarrow$ $3B$ parameters!

Computer Vision Problem





Vertical edge detection

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

6 × 6

Convolution

*

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

3 × 3

Filter
Kernel

=

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

4 × 4

Vertical edge detection

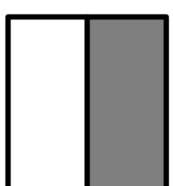
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

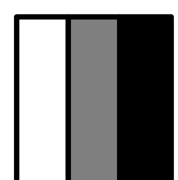
1	0	-1
1	0	-1
1	0	-1

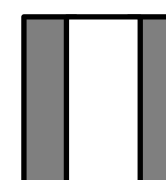
=

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



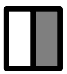
*





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



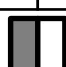
*

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

=

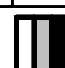
0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

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



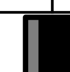
*

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



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

*

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

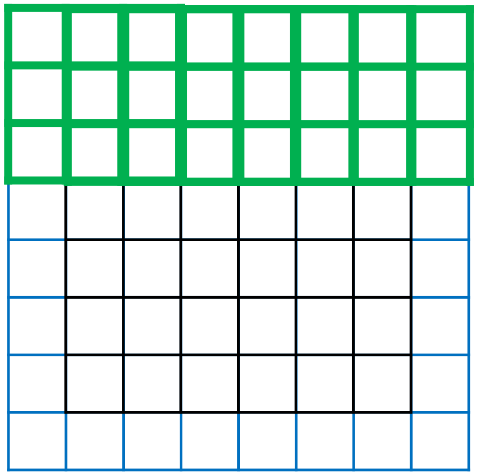
3	0	1	2	7	4
1	5	8	9	3	1
2	7	2	5	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

$*$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

$=$

Padding

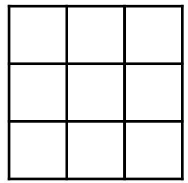


6×6

$n \times n$

$\mathbf{p = padding = 1}$

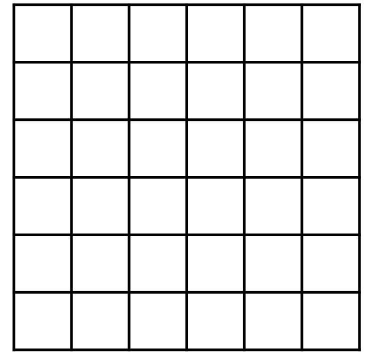
$*$



3×3

$f \times f$

$=$



6×6

with padding:

$$(n + 2p - f + 1) \times (n + 2p - f + 1)$$
$$(6 + 2 - 3 + 1) \times (6 + 2 - 3 + 1)$$
$$= 6 \times 6$$

Valid and Same convolutions

"Valid" : No padding

$$n \times n \quad * \quad f \times f \quad \rightarrow \quad (n - f + 1) \times (n - f + 1)$$

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

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

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

$$p = \frac{f - 1}{2}$$

f is usually odd

$$3 \times 3 \Rightarrow p = 1$$

$$5 \times 5 \Rightarrow p = 2$$

$$7 \times 7 \Rightarrow p = 3$$

$$\vdots$$

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

$n \times n$
Padding p

$*$

Stride s
$s = 2$

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

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

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

Summary of convolutions

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

padding p $\qquad$ stride s

Output size:

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

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

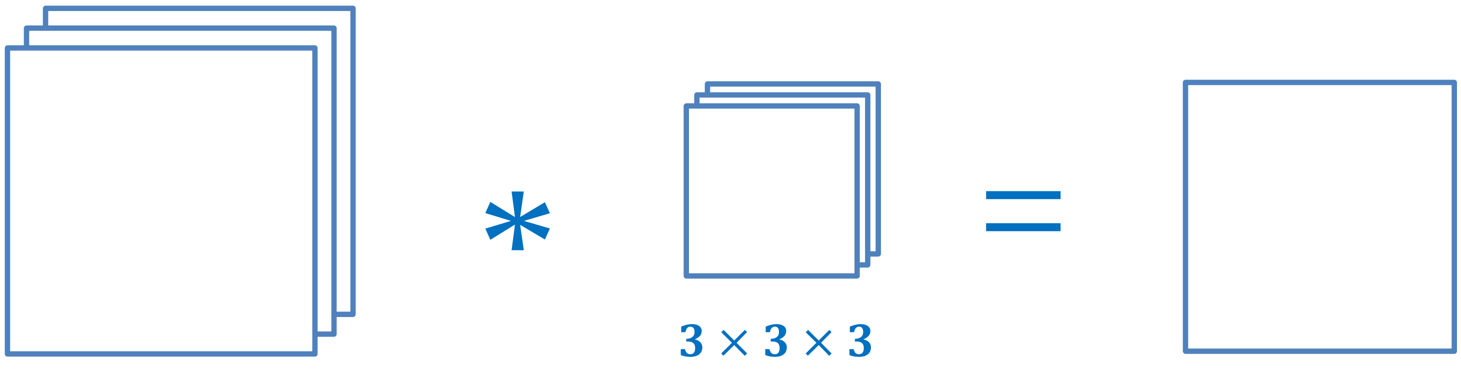
7	2	5
9	0	4
-1	1	3

=

Associativity:

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

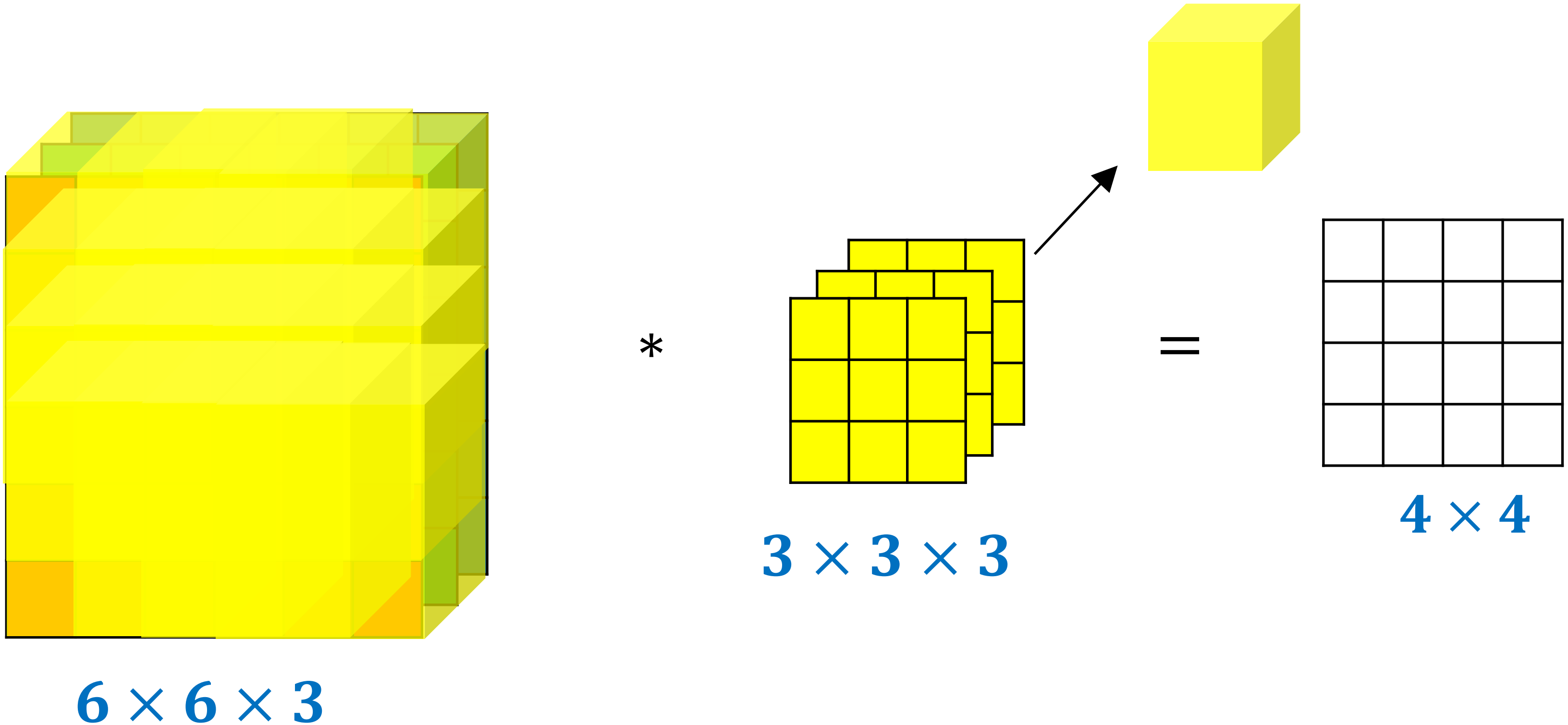
Convolutions on RGB images



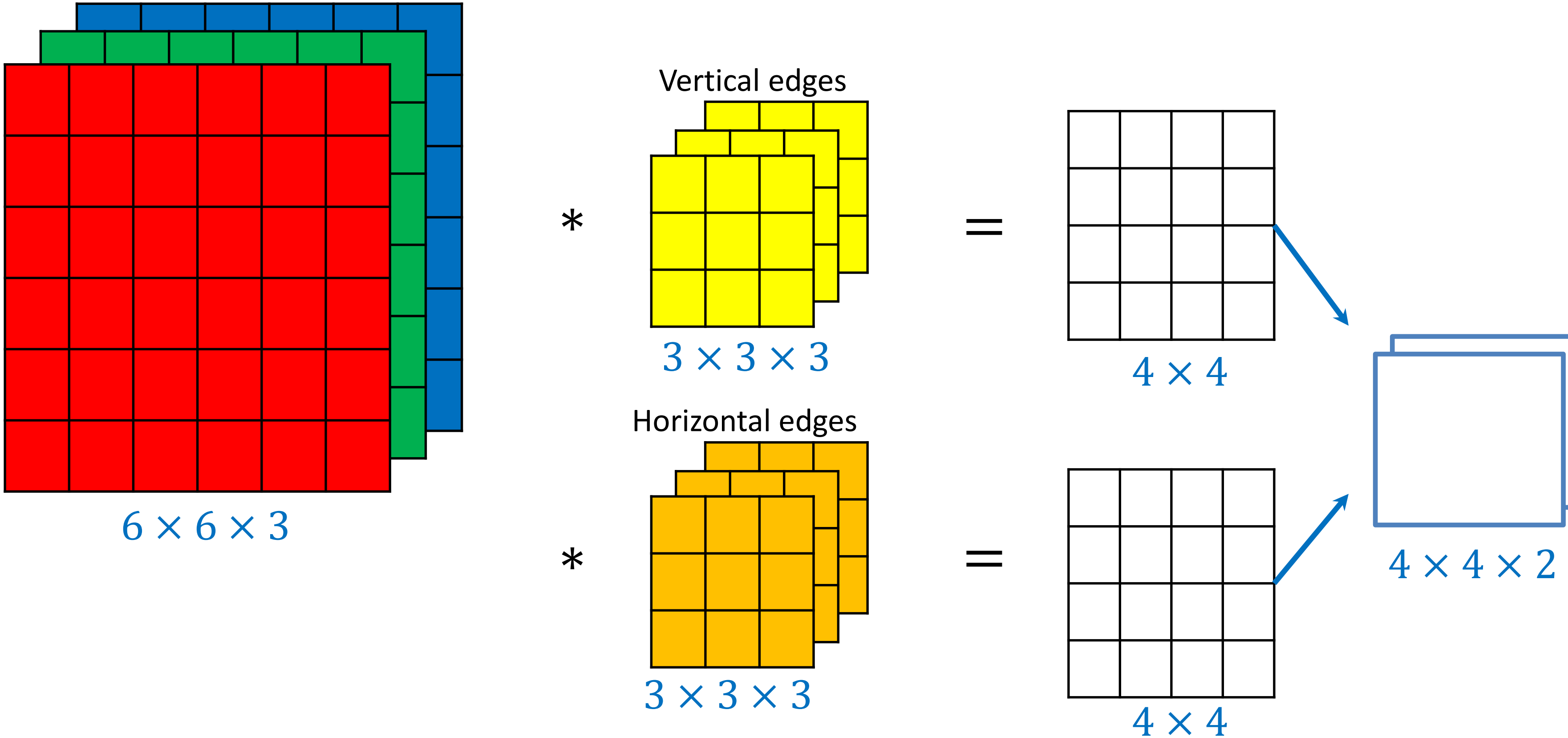
$6 \times 6 \times 3$ $\quad$ $3 \times 3 \times 3$ $\quad$ 4×4

Height $\times$ Witdh $\times$ #channels

Convolutions on RGB image



Multiple filters

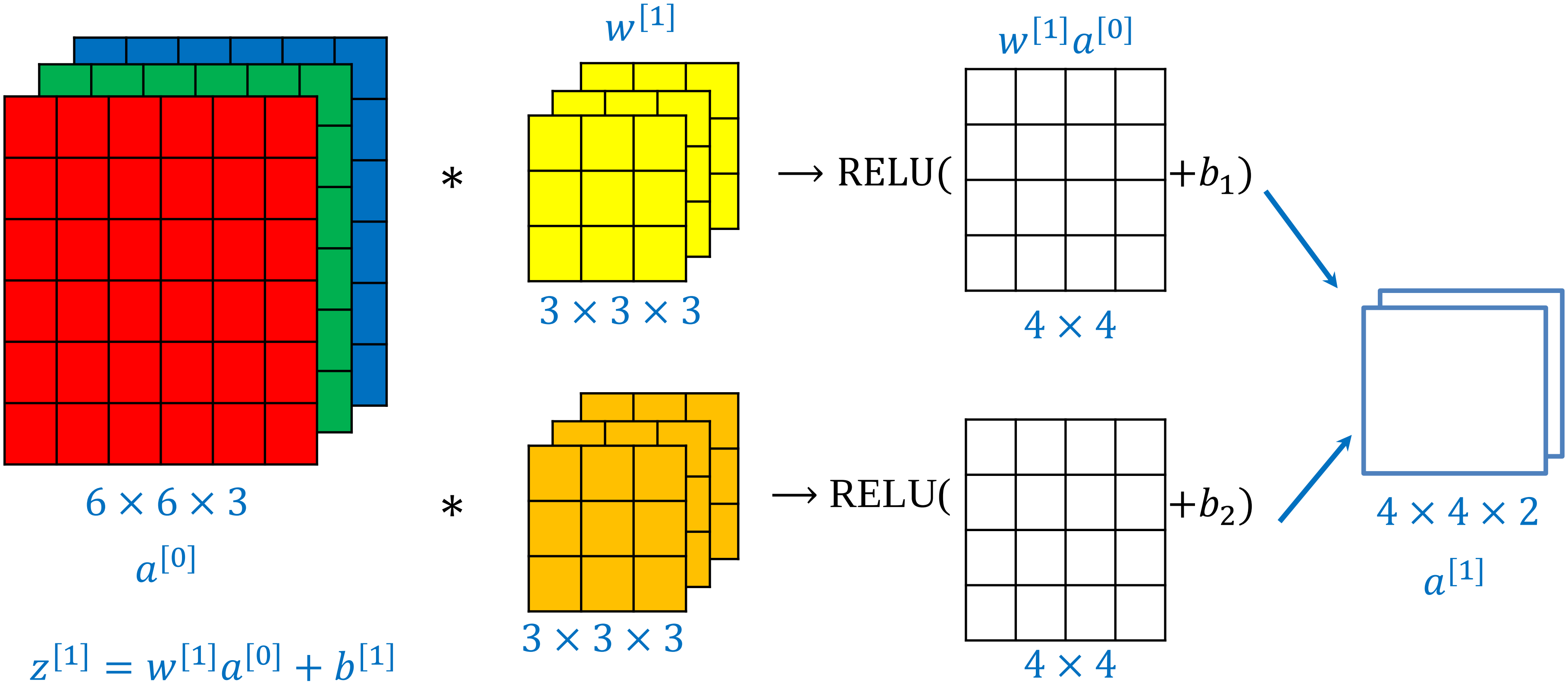


Summary: $\boldsymbol{n \times n \times nc \quad * \quad f \times f \times nc \quad \to \quad (n - f + 1) \times (n - f + 1) \times nc'}$

$\boldsymbol{6 \times 6 \times 3 \quad * \quad 3 \times 3 \times 3 \quad \to \quad 4 \times 4 \times 2}$

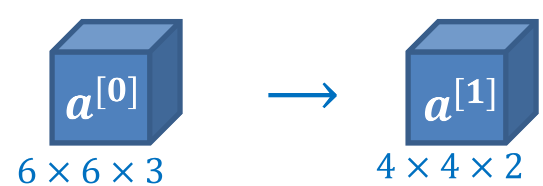
 # filters

Example of a layer



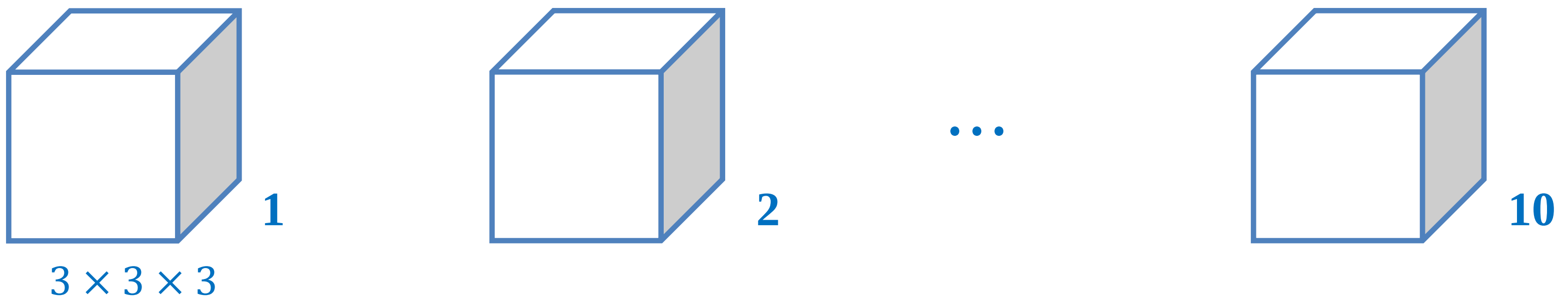
$$z^{[1]} = w^{[1]}a^{[0]} + b^{[1]}$$

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



Number of parameters in one layer

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



27 parameters + 1 bias
=> **28 parameters**

<u>280 parameters</u>

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

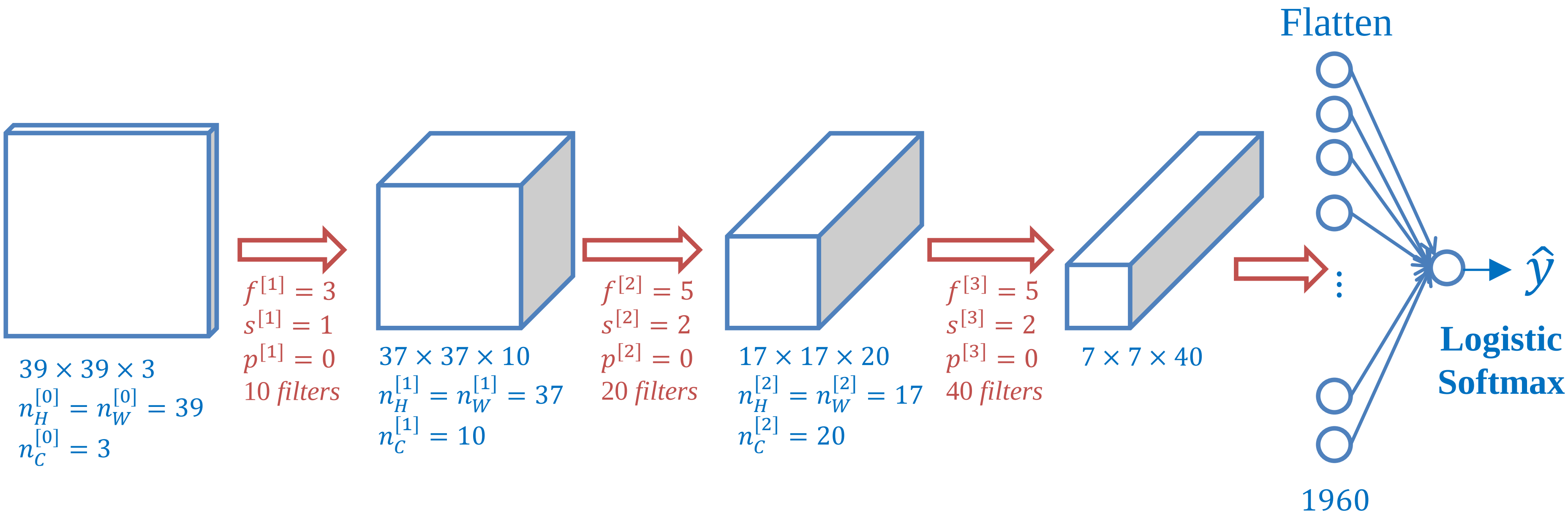
Bias : $n_c^{[l]}$

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

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

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

Example ConvNet



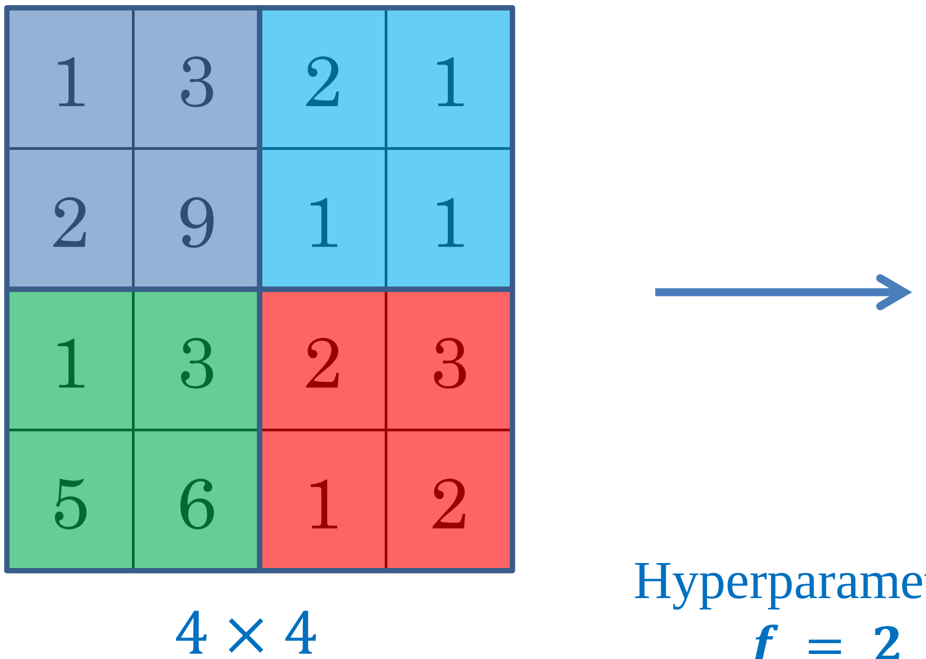
Types of layer in a convolutional network

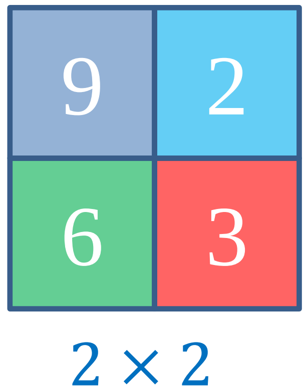
- Convolution (CONV)

- Pooling (POOL)

- Fully connected (FC)

Pooling layer: Max pooling

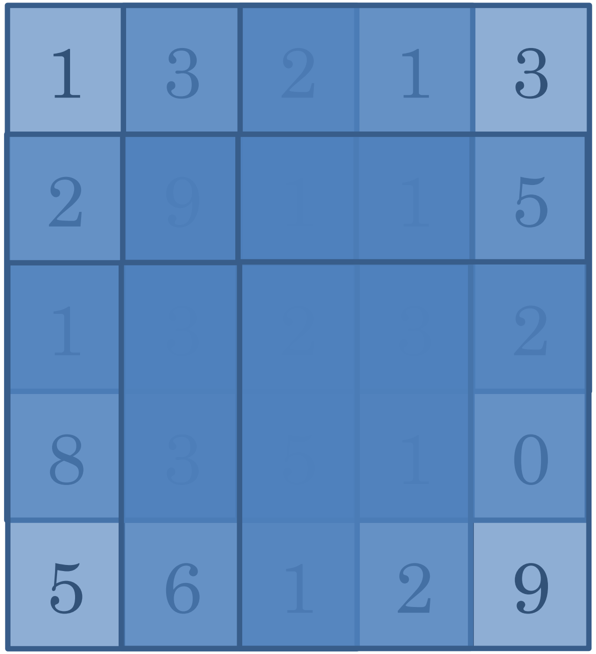




Hyperparameters:

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

Pooling layer: Max pooling



$5 \times 5 \times 2$

9	9	5
9	9	5
8	6	9

$3 \times 3 \times 2$

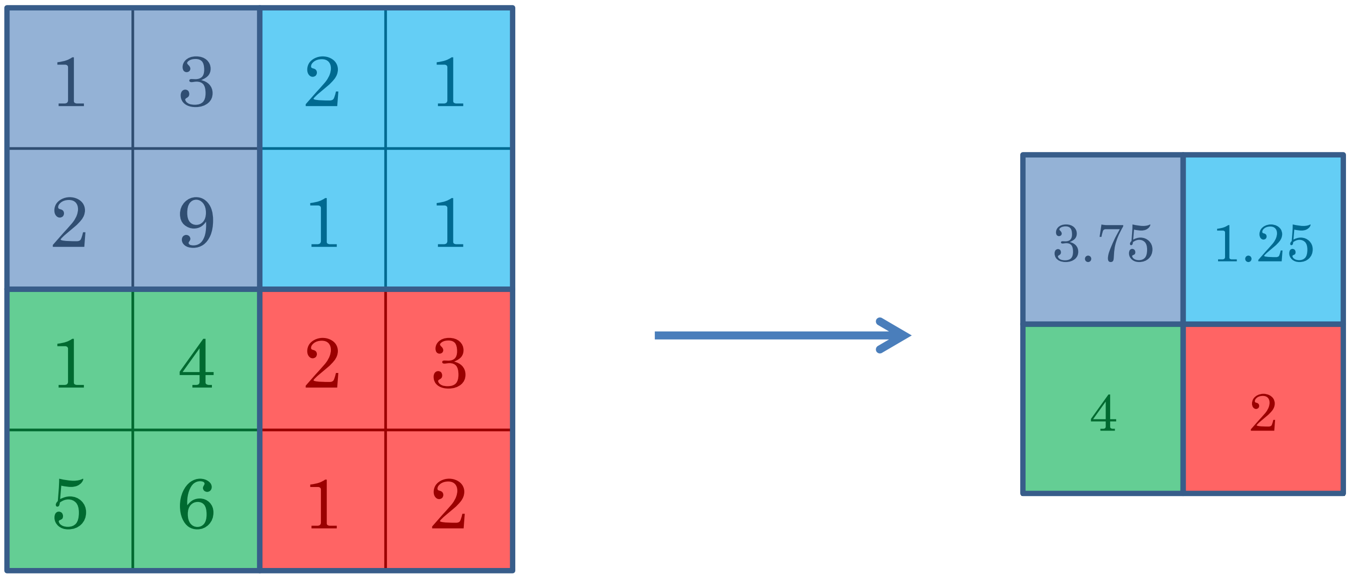
$$\left\lfloor \frac{n-f}{s} + 1 \right\rfloor$$

Hyperparameters:

$f = 3$

$s = 1$

Pooling layer: Average pooling



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

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

$$n_H \times n_W \times n_C$$

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

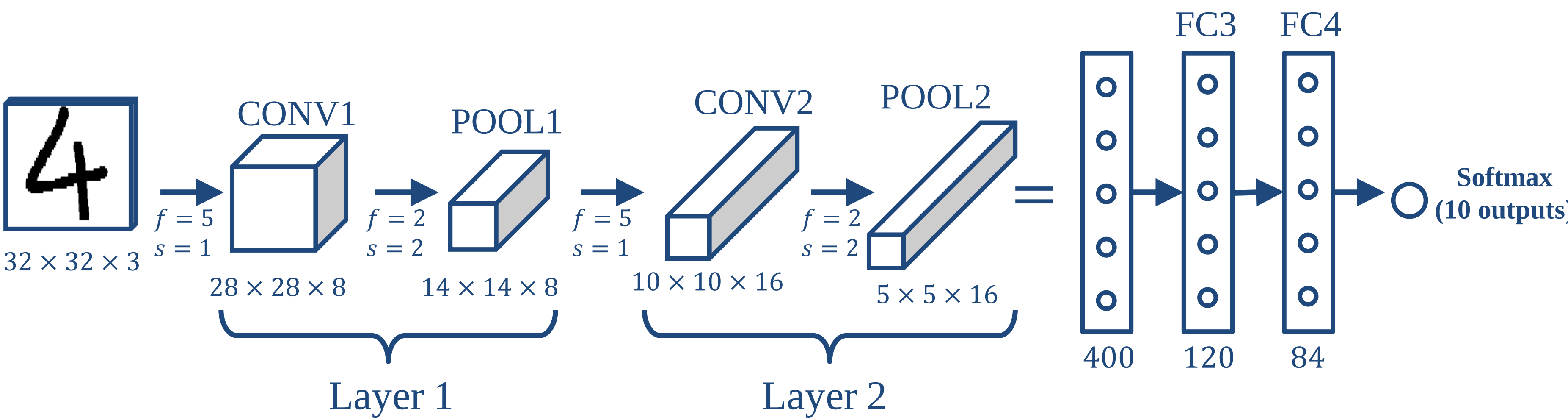
No parameters to learn!

Neural network example

$$g_k(\tilde{x}) = \frac{\exp(-\tilde{\mathbf{w}}_k{}^T \tilde{x})}{\sum_j \exp(-\tilde{\mathbf{w}}_j{}^T \tilde{x})}$$

$$g_k(\tilde{x}) \in [0,1]$$

(LeNet-5)



CONV-POOL-CONV-POOL-FC-FC-SOFTMAX

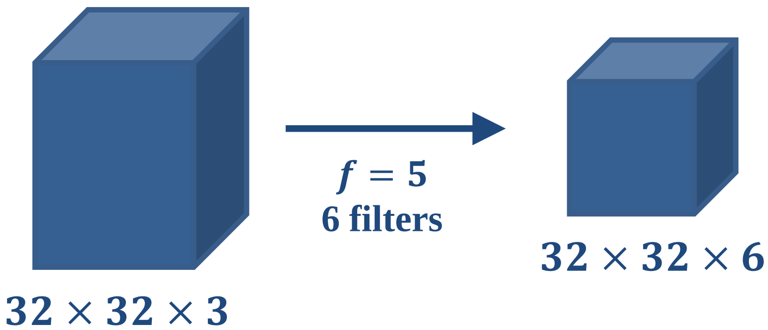
Neural network example

	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	3,072	0

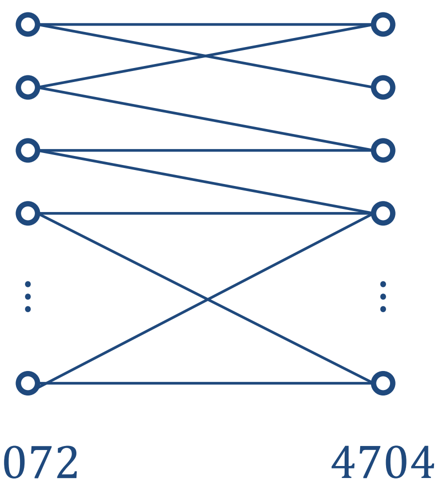
Neural network example

	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	3,072	0
CONV1 (f=5, s=1)	(28,28,8)	6,272	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	48,120
FC4	(84,1)	84	10,164
Softmax	(10,1)	10	850

Why convolutions



$$5 \times 5 = 25 + 1$$
$$= 26 \; parameters \; per \; filter$$
$$6 \times 26 = 156 \; parameters$$



$$3072 \times 4704 \approx 14M$$

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

$=$

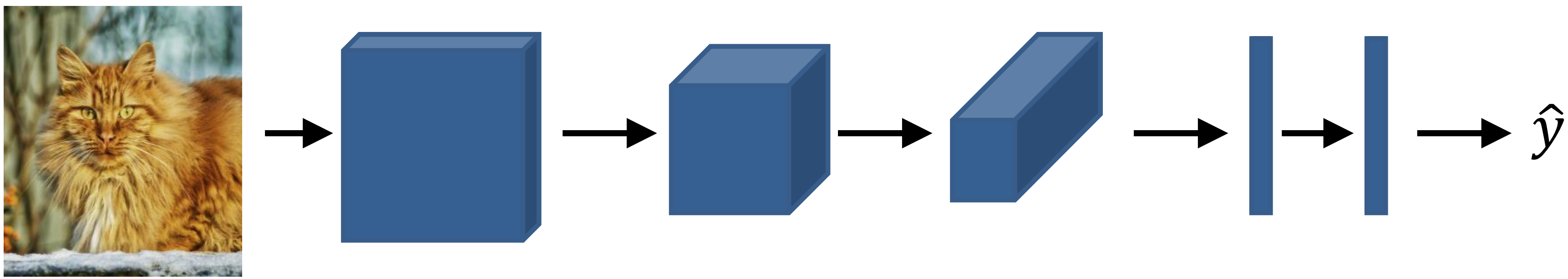
0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

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

References

- Andrew Ng. Deep learning. Coursera.
- Geoffrey Hinton. Neural Networks for Machine Learning.
- Kevin P. Murphy. Probabilistic Machine Learning An Introduction. MIT Press, 2022.
- MIT Deep Learning 6.S191 (http://introtodeeplearning.com/)