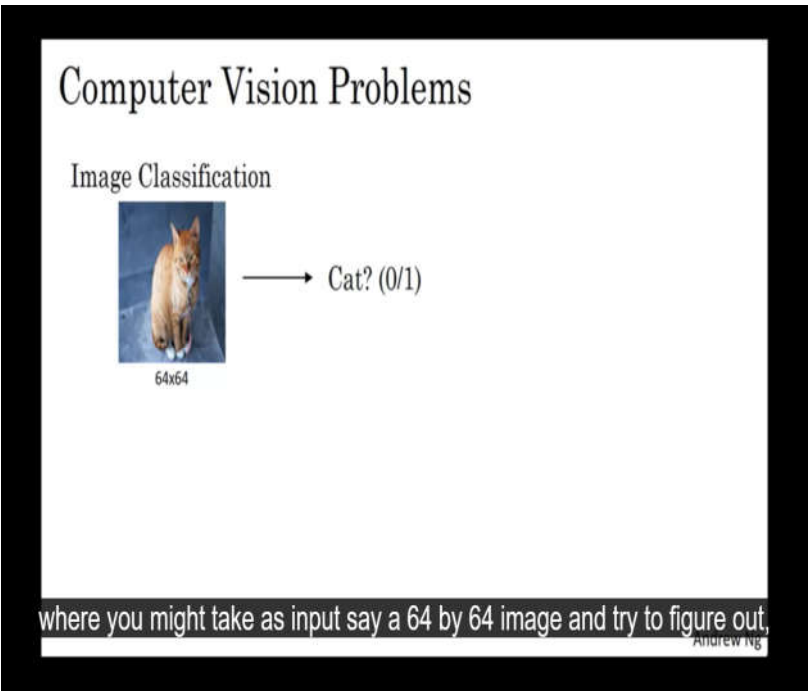
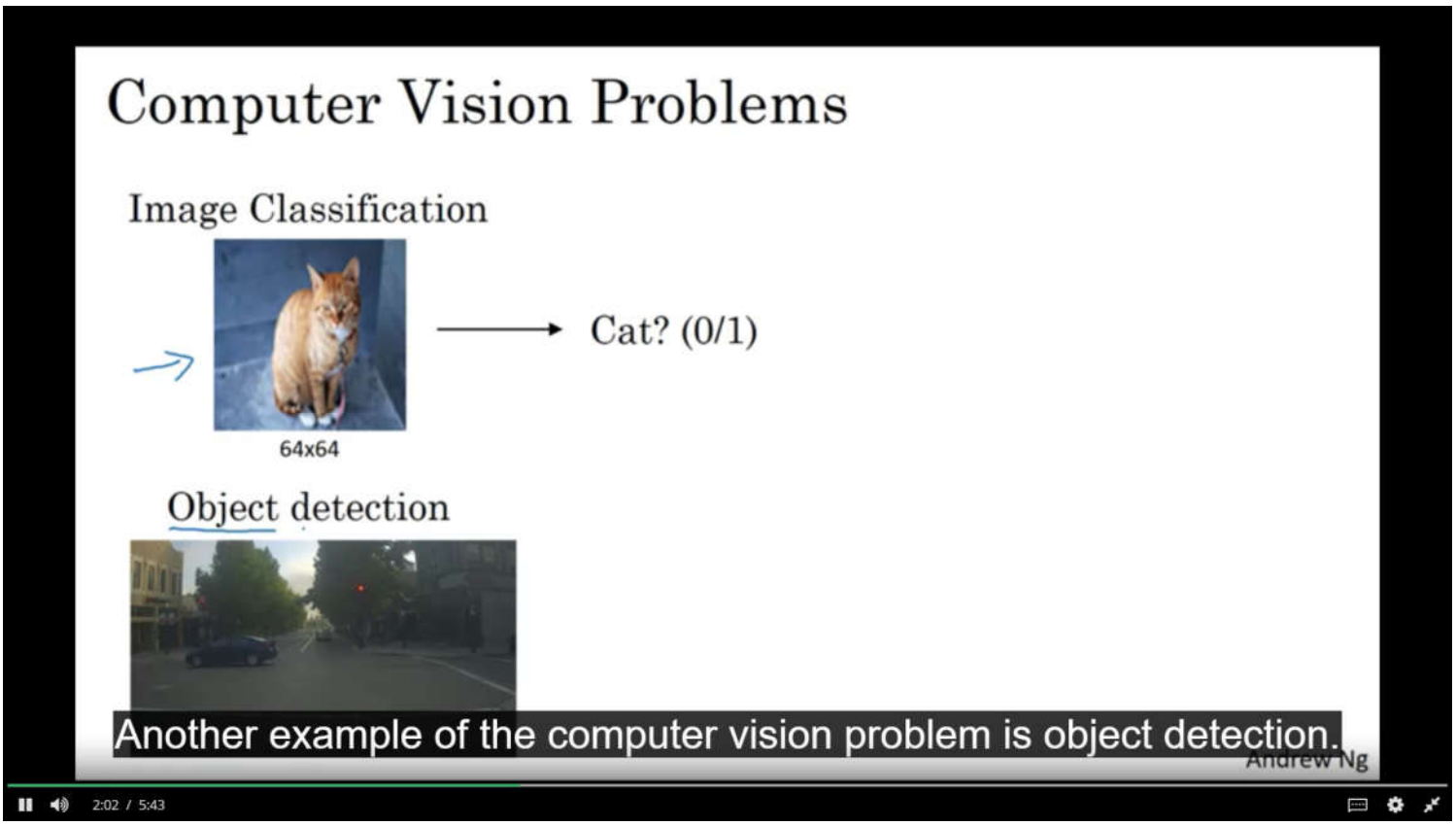
CNN -Week 1

28 April 2020 14:04





Computer Vision Problems

Image Classification

64x64

→ Cat? (0/1)

Neural Style Transfer

Object detection

and you want this picture repainted in a different style.

Andrew Ng

Computer Vision Problems

Image Classification

64x64

→ Cat? (0/1)

Neural Style Transfer

Object detection

and you end up with the image at the bottom.

Andrew Ng

Deep Learning on large images

→ Cat? (0/1)

Deep Learning on large images

64x64

→ Cat? (0/1)

One of the challenges of computer vision problems is that the inputs can get really big.

Andrew Ng

Deep Learning on large images

64x64 × 3

→ Cat? (0/1)

12288

1000 × 1000 × 3
= 3 million

If you are viewing this on a smaller screen,

Andrew Ng

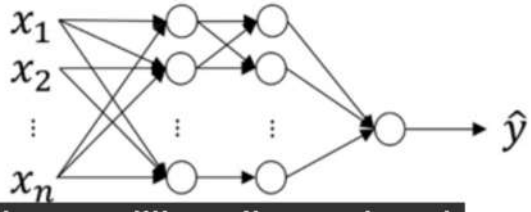
Deep Learning on large images

64x64 × 3

→ Cat? (0/1)

12288

1000 × 1000 × 3
= 3 million



then this means that X here will be three million dimensional.

Andrew Ng

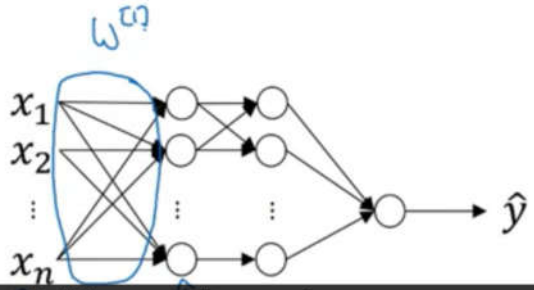
Deep Learning on large images

64x64 × 3

→ Cat? (0/1)

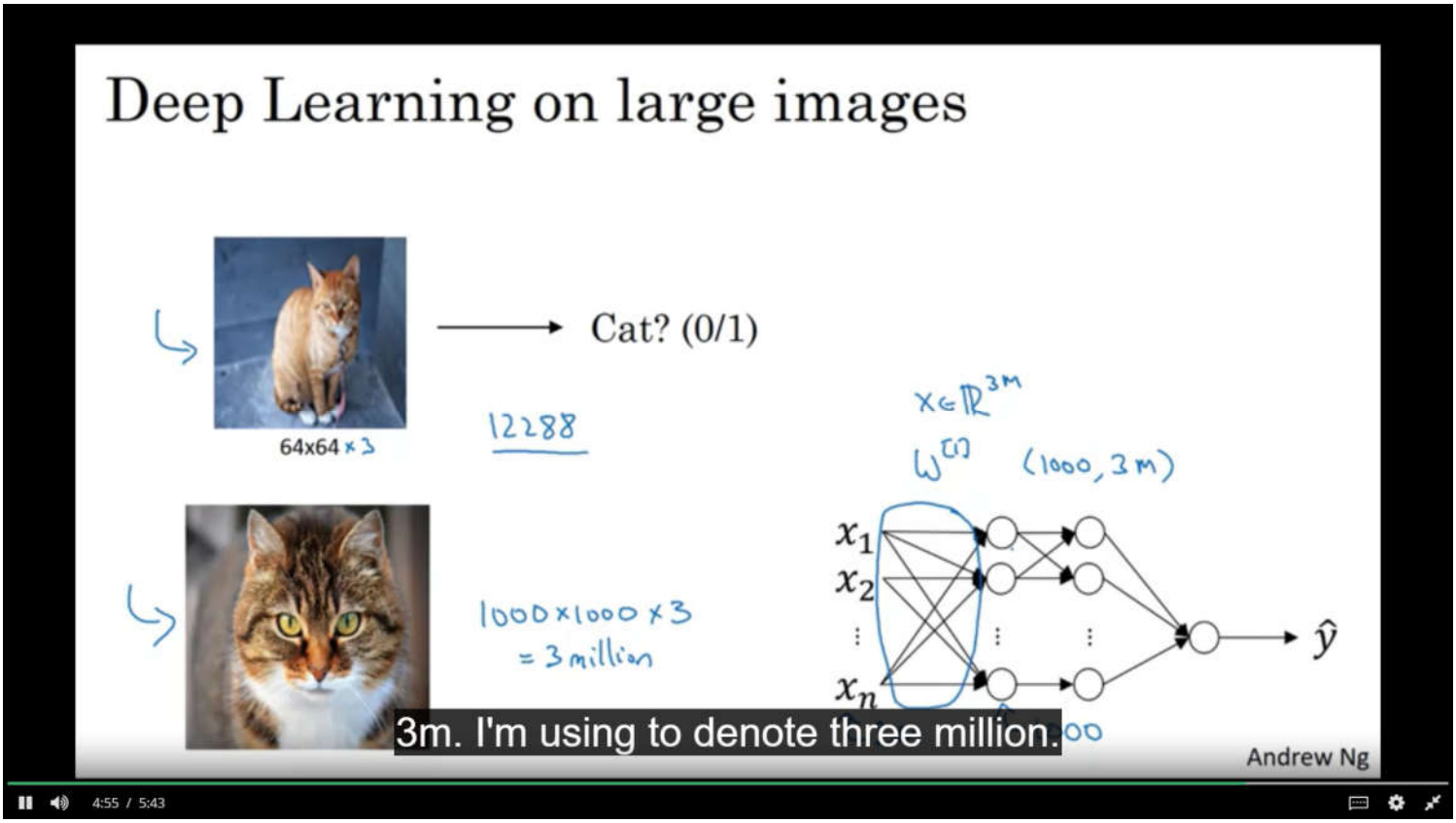
12288

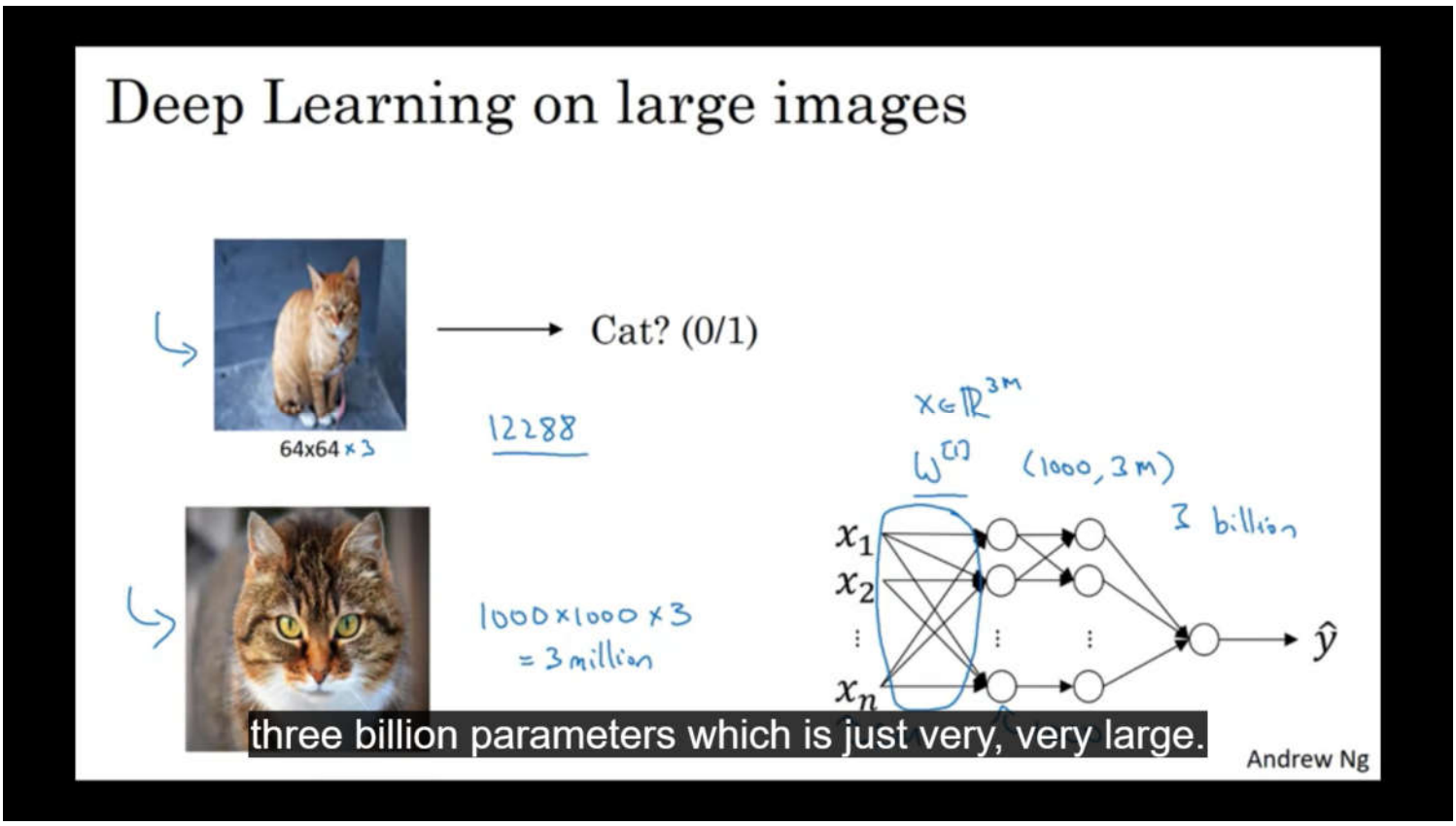
1000 × 1000 × 3
= 3 million



if you use a standard or fully connected network like we have in courses one or two.

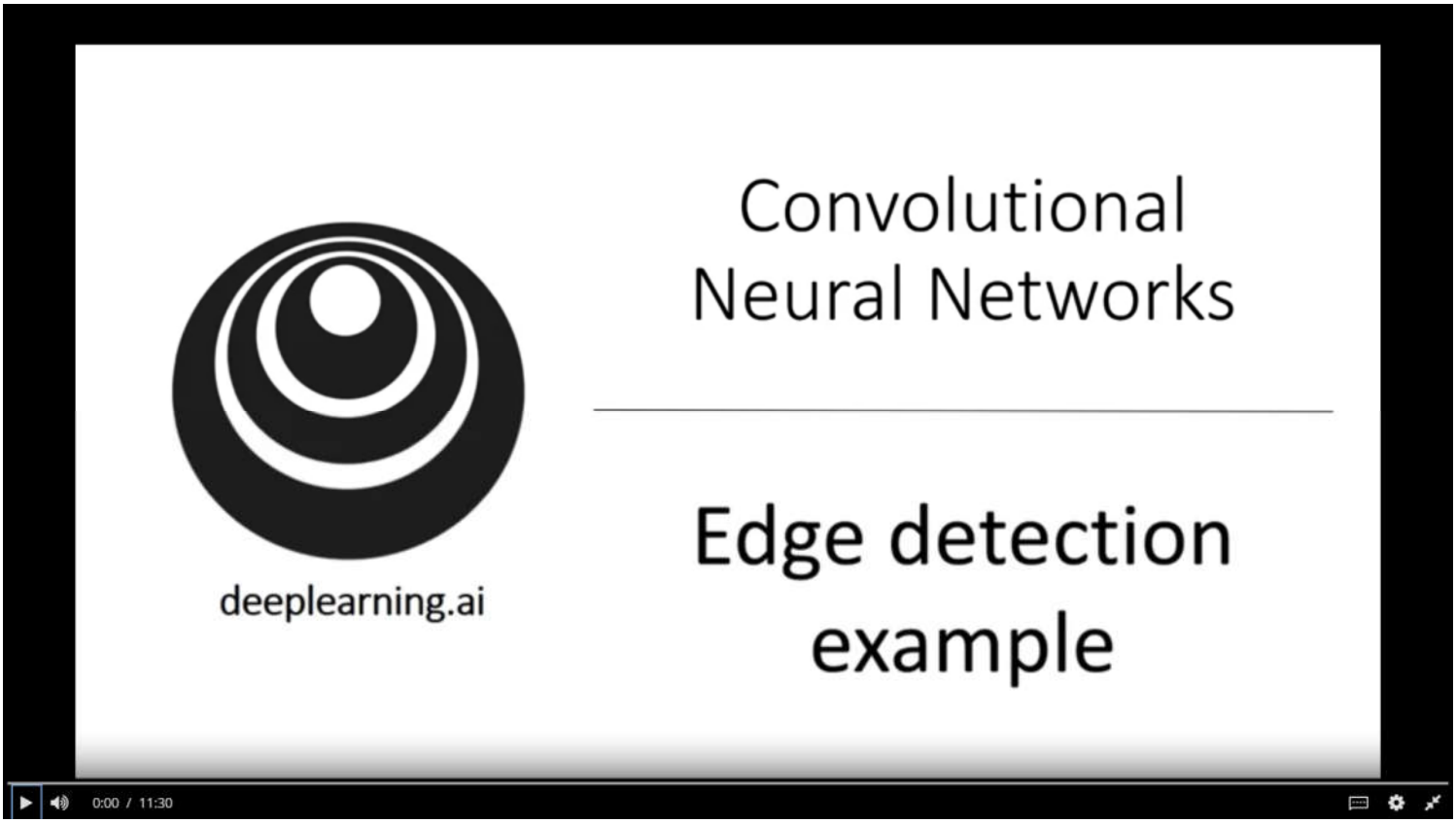
Andrew Ng

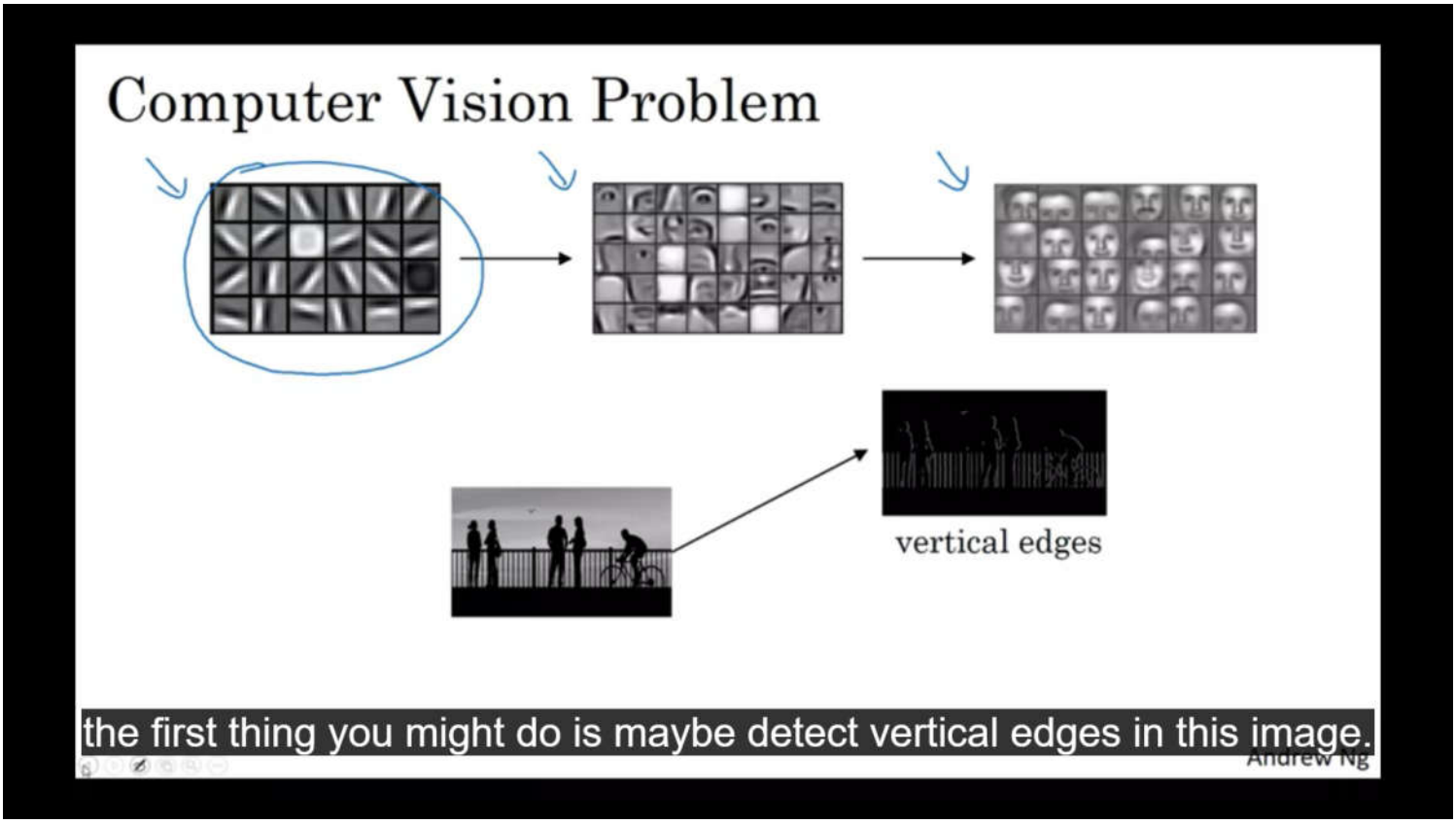

Deep Learning on large images
Cat? (0/1)
64x64 × 3
12288
1000×1000×3
= 3 million
$x \in \mathbb{R}^{3M}$
$W^{[1]}$ (1000, 3M)
x_1
x_2
x_n
$\hat{y}$
3m. I'm using to denote three million.
Andrew Ng
4:55 / 5:43


Deep Learning on large images
Cat? (0/1)
64x64 × 3
12288
1000×1000×3
= 3 million
$x \in \mathbb{R}^{3M}$
$W^{[1]}$ (1000, 3M)
3 billion
x_1
x_2
x_n
$\hat{y}$
three billion parameters which is just very, very large.
Andrew Ng

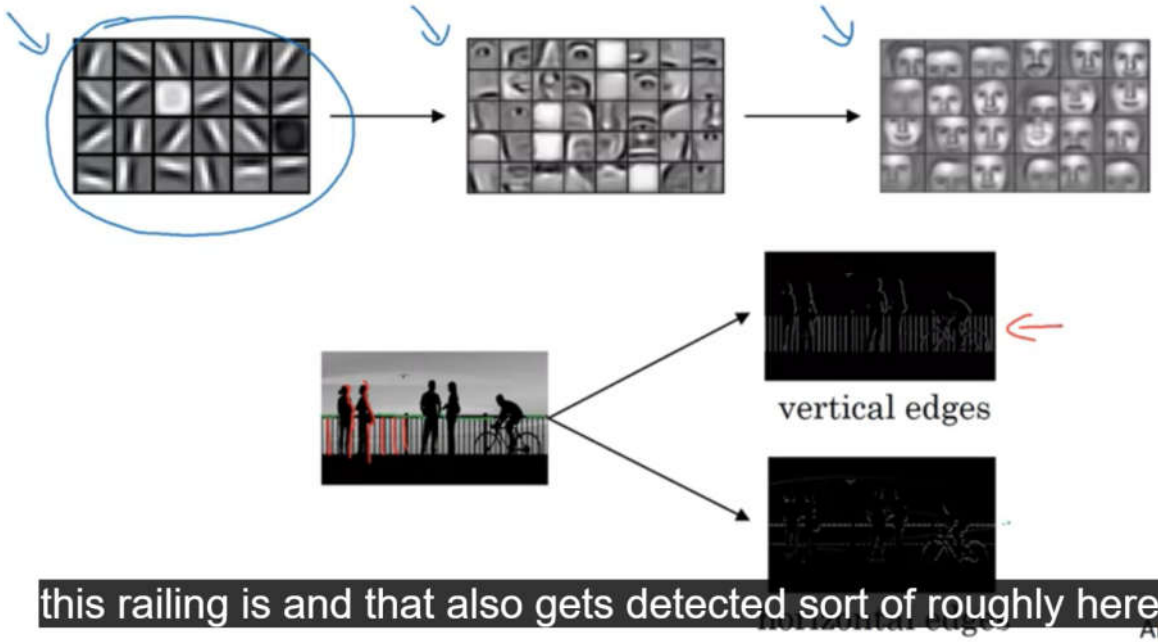
Deep
To do that, you need to better implement the convolution operation,
Andrew Ng

Deep
which is one of the fundamental building blocks of convolutional neural networks.
Andrew Ng


Convolutional
Neural Networks
deeplearning.ai
Edge detection
example
0:00 / 11:30


Computer Vision Problem
vertical edges
the first thing you might do is maybe detect vertical edges in this image.
Andrew Ng

Computer Vision Problem



this railing is and that also gets detected sort of roughly here.

Andrew Ng

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

In order to detect edges or lets say vertical edges in his image,

Andrew Ng

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

*

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

3×3 filter

1, 1, 0, 0, 0, -1, -1, -1.

Andrew Ng

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

convolve it with the 3 by 3 filter.

Andrew Ng

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

=

4×4

The way you compute this 4 by 4 output is as follows,

Andrew Ng

Vertical edge detection

3^{1}	0^{0}	1^{-1}	2	7	4
1^{1}	5^{0}	8^{-1}	9	3	1
2^{1}	7^{0}	2^{-1}	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

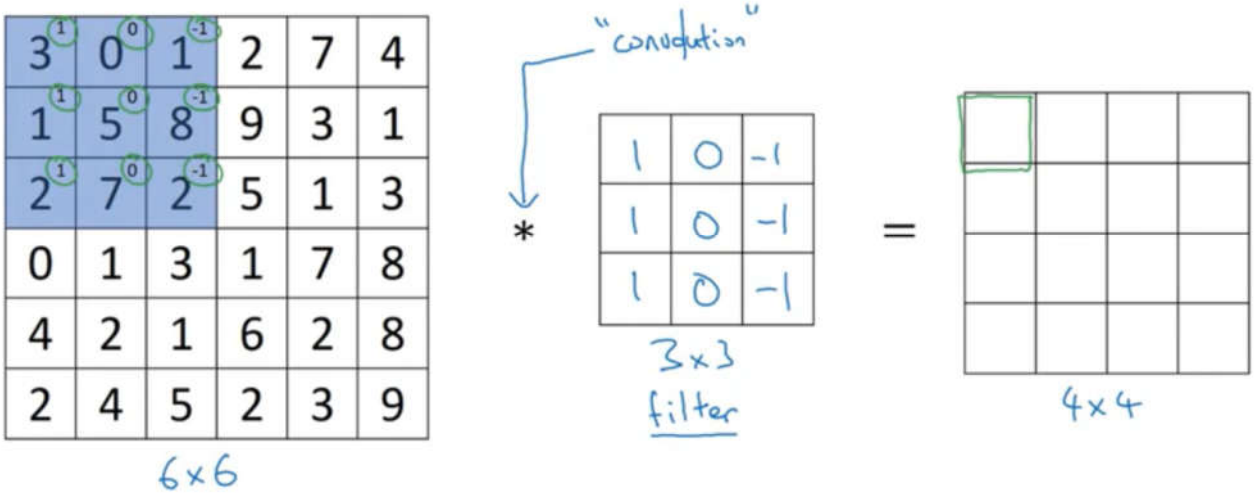
=

4×4

top of the 3 by 3 region of your original input image.

Andrew Ng

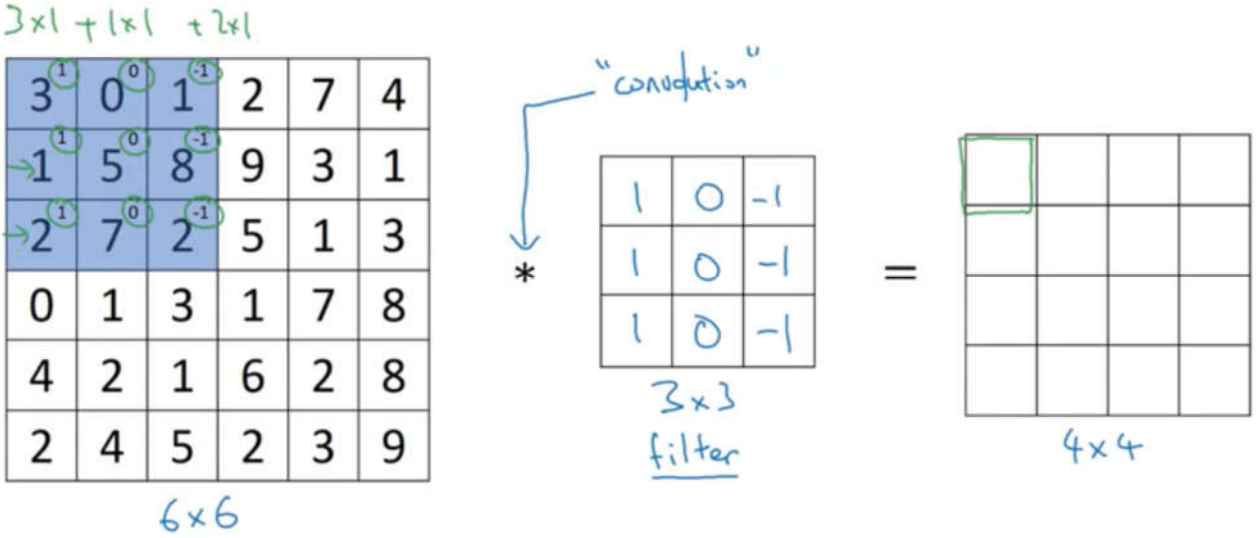
Vertical edge detection



0, 0, 0, -1, -1, -1.

Andrew Ng

Vertical edge detection



just one and then add up all of the resulting nine numbers.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1$

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

=

4×4

eight times -1, plus two times -1.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

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

6×6

"convolution"

*

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

3×3 filter

=

-5			

4×4

Let me get rid of the green marks here.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

3	0^{1}	1^{0}	2^{-1}	7	4
1	5^{1}	8^{0}	9^{-1}	3	1
2	7^{1}	2^{0}	5^{-1}	1	3
0	1	3	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6×6

$*$

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

3×3 filter

$=$

-5	-4		

4×4

you end up with negative four and so on.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

3	0	1	2	7	4
1^{1}	5^{0}	8^{-1}	9	3	1
2^{1}	7^{0}	2^{-1}	5	1	3
0^{1}	1^{0}	3^{-1}	1	7	8
4	2	1	6	2	8
2	4	5	2	3	9

6×6

$*$

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

3×3 filter

$=$

-5	-4	0	8

4×4

one down so you now have it in that position,

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

$*$

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

3×3 filter

$=$

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

4×4

Then fill in all the rest of the elements of the matrix.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

$*$

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

3×3 filter

$=$

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

4×4

A 6 by 6 matrix convolve of the 3 by 3 matrix gives you a 4 by 4 matrix.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

*

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

3×3 filter

=

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

4×4

But the matrix on the left is convenient to interpret as image,

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

*

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

3×3 filter

=

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

4×4

and the one in the middle we interpret as a filter and the one on the right,

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

*

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

3×3 filter

=

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

4×4

you can interpret that as maybe another image.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

*

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

3×3 filter

=

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

4×4

And this turns out to be a vertical edge detector,

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

$*$

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

3×3 filter

$=$

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

4×4

python: conv_forward
tensorflow: t.

there is a function tf.nn.cont2d.

Andrew Ng

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

$*$

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

3×3 filter

$=$

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

4×4

python: conv_forward
tensorflow: tf.nn.conv2d

And then other deep learning programming frameworks in the CARIS program firmware,

Vertical edge detection

$3\times1 + 1\times1 + 2\times1 + 0\times0 + 5\times0 + 7\times0 + 1\times-1 + 8\times-1 + 2\times-1 = -5$

"convolution"

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

*

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

3×3 filter

=

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

4×4

python: conv_forward
tensorflow: tf.nn.conv2d

there is a function called cont2d that implements convolution and so on.

Andrew Ng

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

Here is a simple 6 by

Andrew Ng

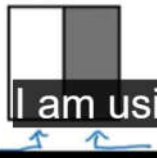
Vertical edge detection

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

6x6

$*$

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



I am using that shade of gray to denote zeros,

Andrew Ng

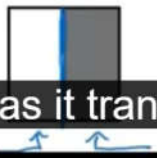
Vertical edge detection

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

6x6

$*$

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



this image as it transitions from white to black or white to darker color.

Andrew Ng

Vertical edge detection

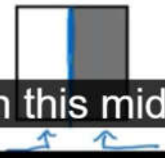
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

6x6

$*$

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

3x3



$*$

the left and then this mid tone zeroes in the middle and then darker on the right.

Andrew Ng

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

6x6

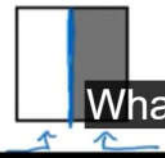
$*$

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

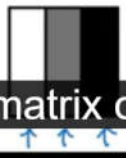
3x3

$=$

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

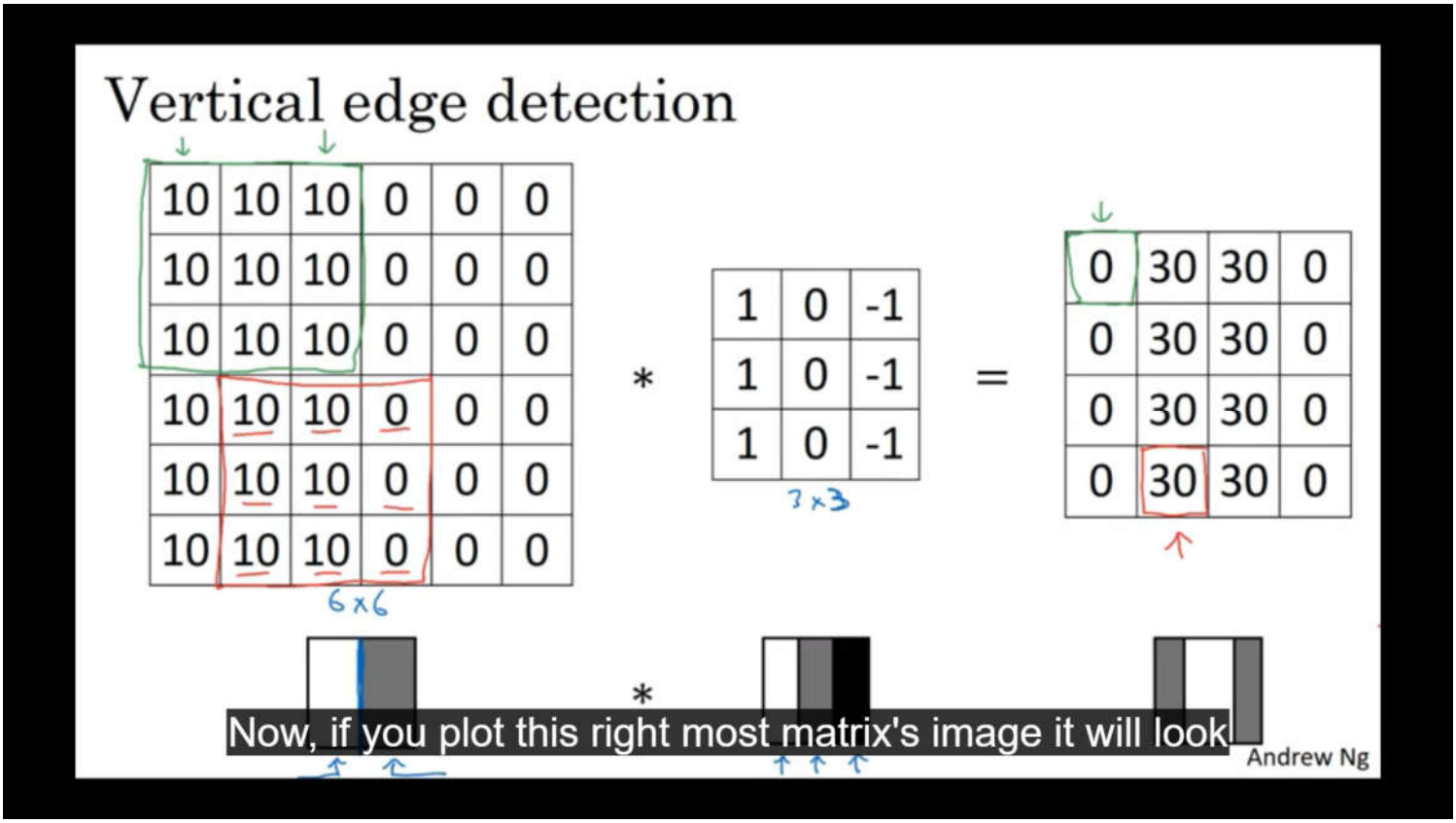


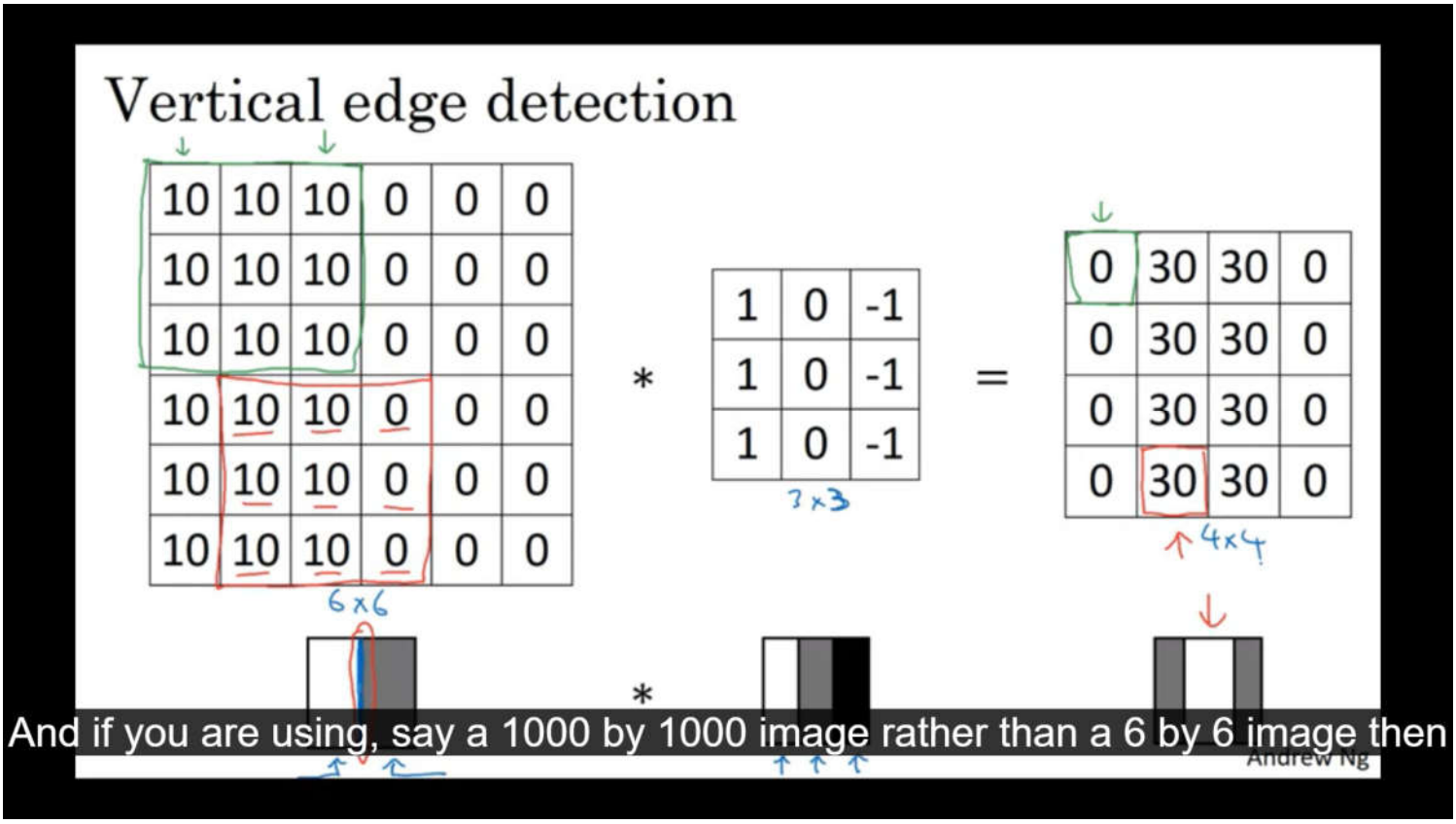
$*$

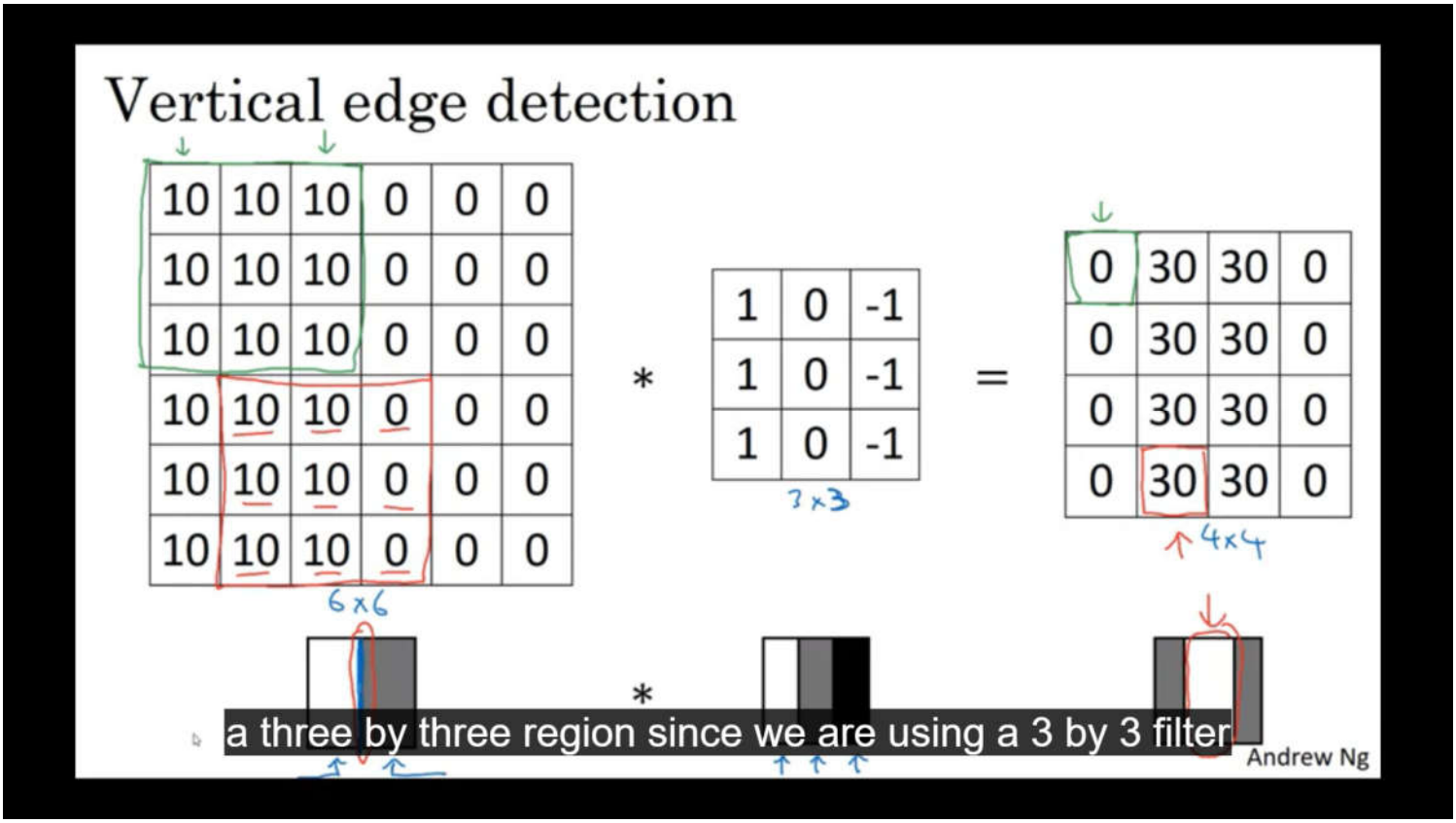


What you get is this matrix on the right.

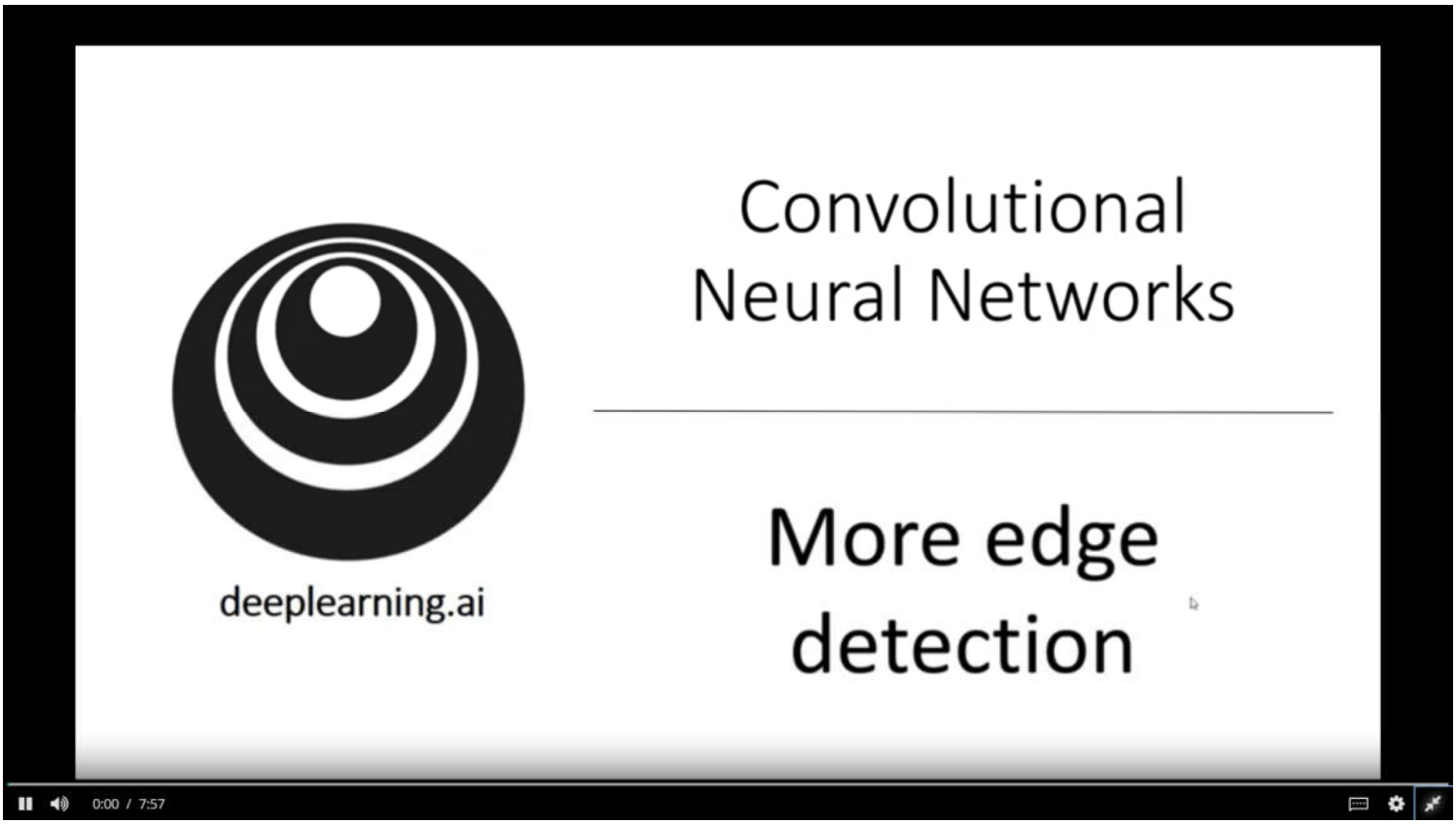
Andrew Ng


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
6x6
*
1 0 -1
1 0 -1
1 0 -1
3x3
=
0 30 30 0
0 30 30 0
0 30 30 0
0 30 30 0
Now, if you plot this right most matrix's image it will look
Andrew Ng


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
6x6
*
1 0 -1
1 0 -1
1 0 -1
3x3
=
0 30 30 0
0 30 30 0
0 30 30 0
0 30 30 0
4x4
And if you are using, say a 1000 by 1000 image rather than a 6 by 6 image then
Andrew Ng


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
6x6
*
1 0 -1
1 0 -1
1 0 -1
3x3
=
0 30 30 0
0 30 30 0
0 30 30 0
0 30 30 0
4x4
a three by three region since we are using a 3 by 3 filter
Andrew Ng

Videoo 3


Convolutional
Neural Networks
deeplearning.ai
More edge
detection
0:00 / 7:57

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

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
0	0	0	10	10	10

$*$

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

What happens in an image where the colors are flipped,

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
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

If you convolve it with
the same edge detection filter,

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
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

you can plot that as a picture
that maybe looks like that.

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
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

you could take absolute values of this output matrix.

Andrew Ng

Vertical and Horizontal Edge Detection

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

Vertical

This three by three filter we've seen allows you to detect vertical edges.

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

three by three filter will allow
you to detect horizontal edges.

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

*

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

where you have here 10s in the upper
left and lower right-hand corners.

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

*

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

darker where there are 0s, so I'm
going to shade in the darker regions, and

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

*

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

Andrew Ng

Learning to detect edges

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

So in summary, different filters allow you to find vertical and horizontal edges.

Andrew Ng

Learning to detect edges

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

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

Sobel filter

And the advantage of this is it puts a little bit more weight to the central row,

Andrew Ng

Learning to detect edges

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

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

Sobel filter

And the advantage of this is it puts a little bit more weight to the central row,

Andrew Ng

Learning to detect edges

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

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

Sobel filter

the central pixel, and this makes it maybe a little bit more robust.

Andrew Ng

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

Sch

And this is called a Scharr filter.

Andrew Ng

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	[illegible]	[illegible]

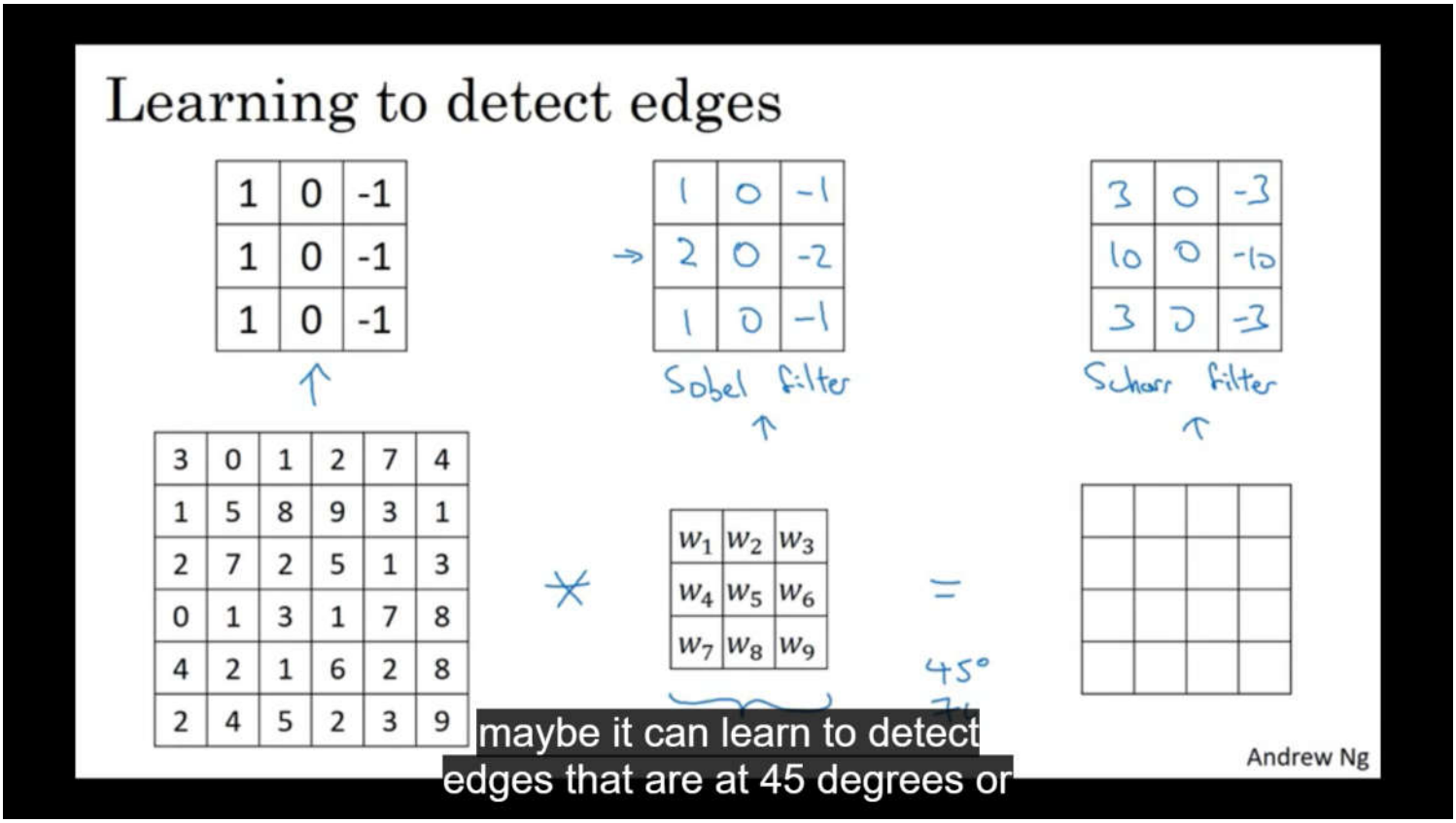
$*$

w_1	w_2	w_3
w_4	w_5	w_6
w_7	w_8	w_9

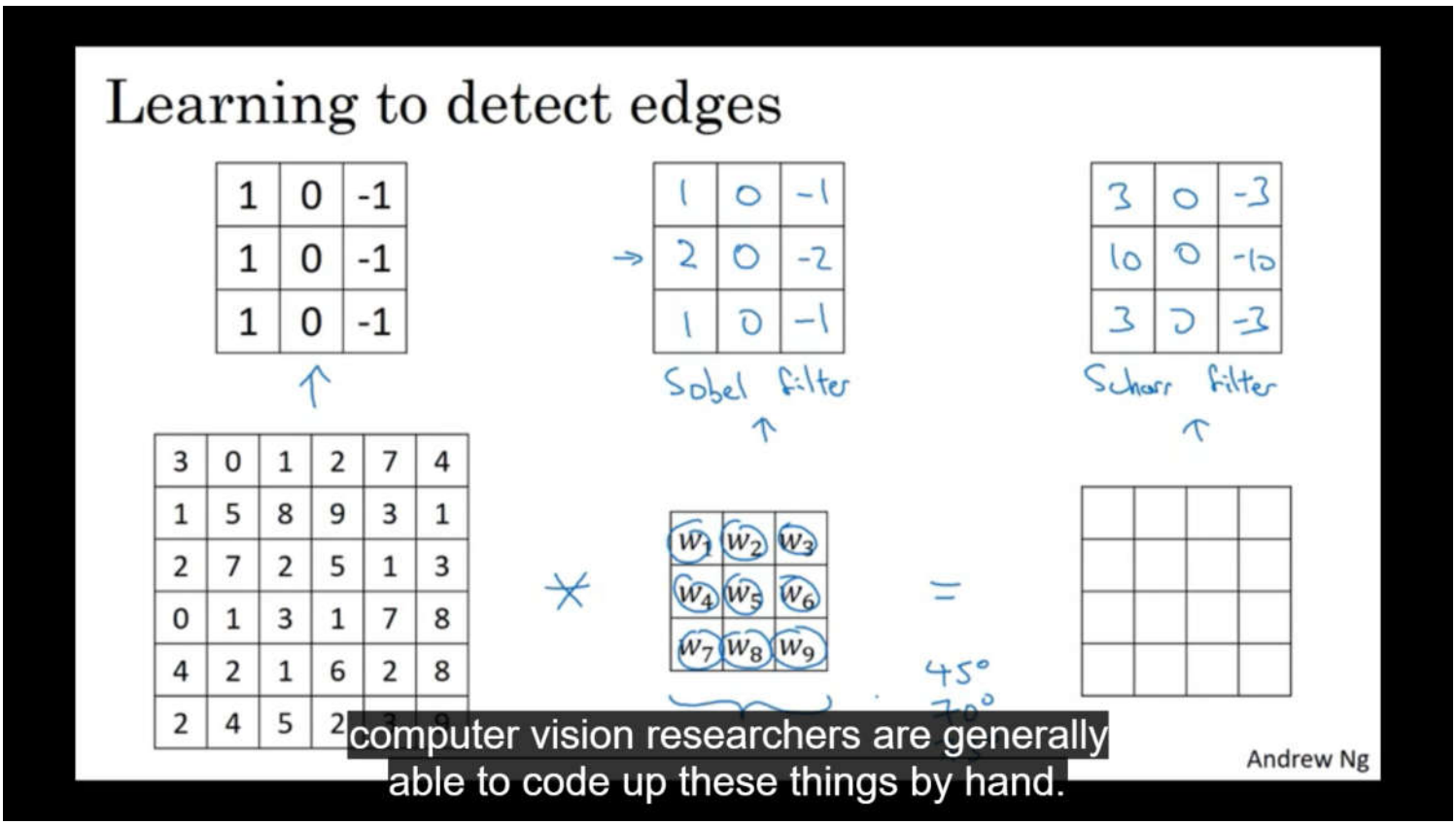
$=$

that this gives you a good edge detector.

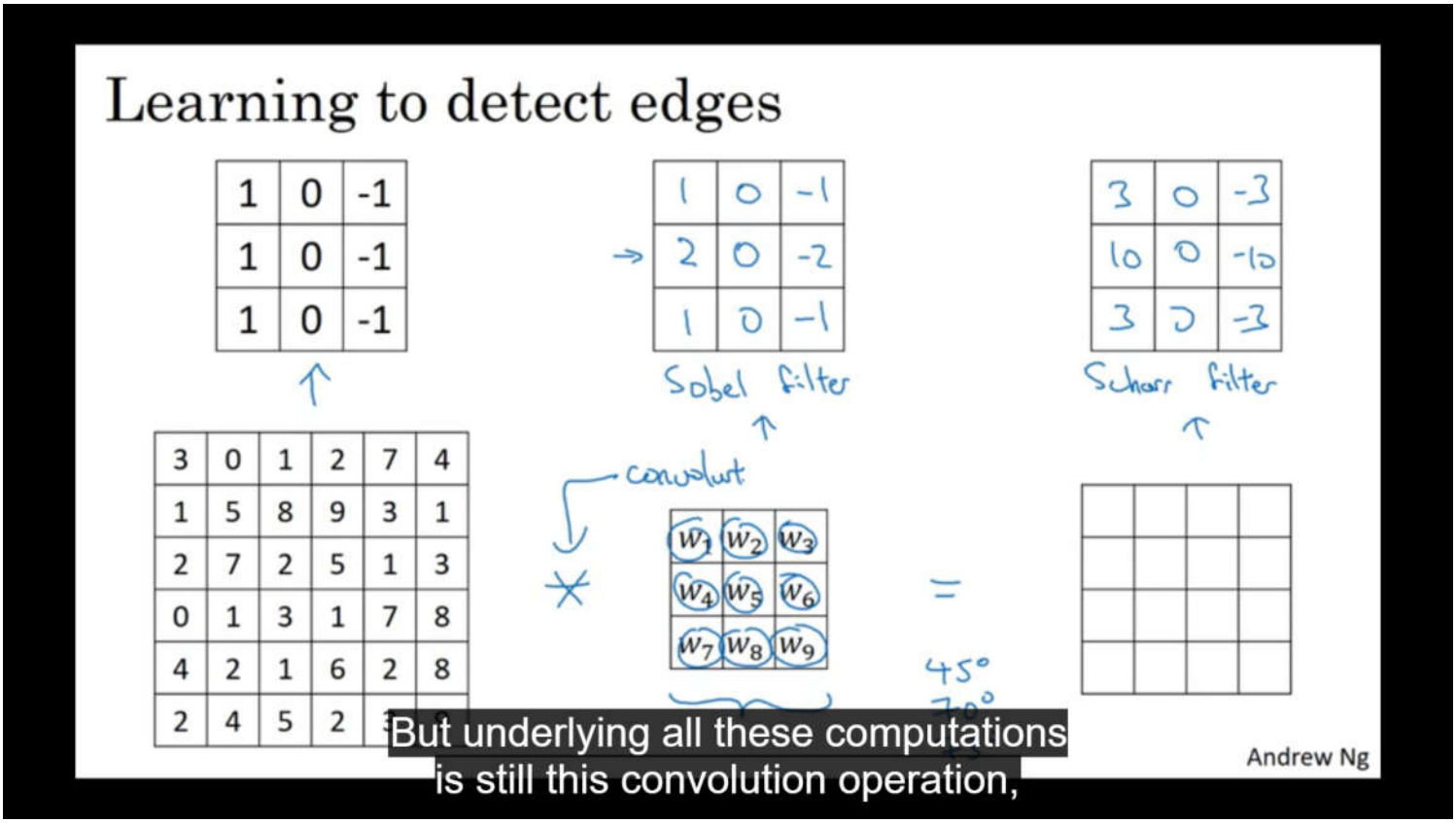
Andrew Ng

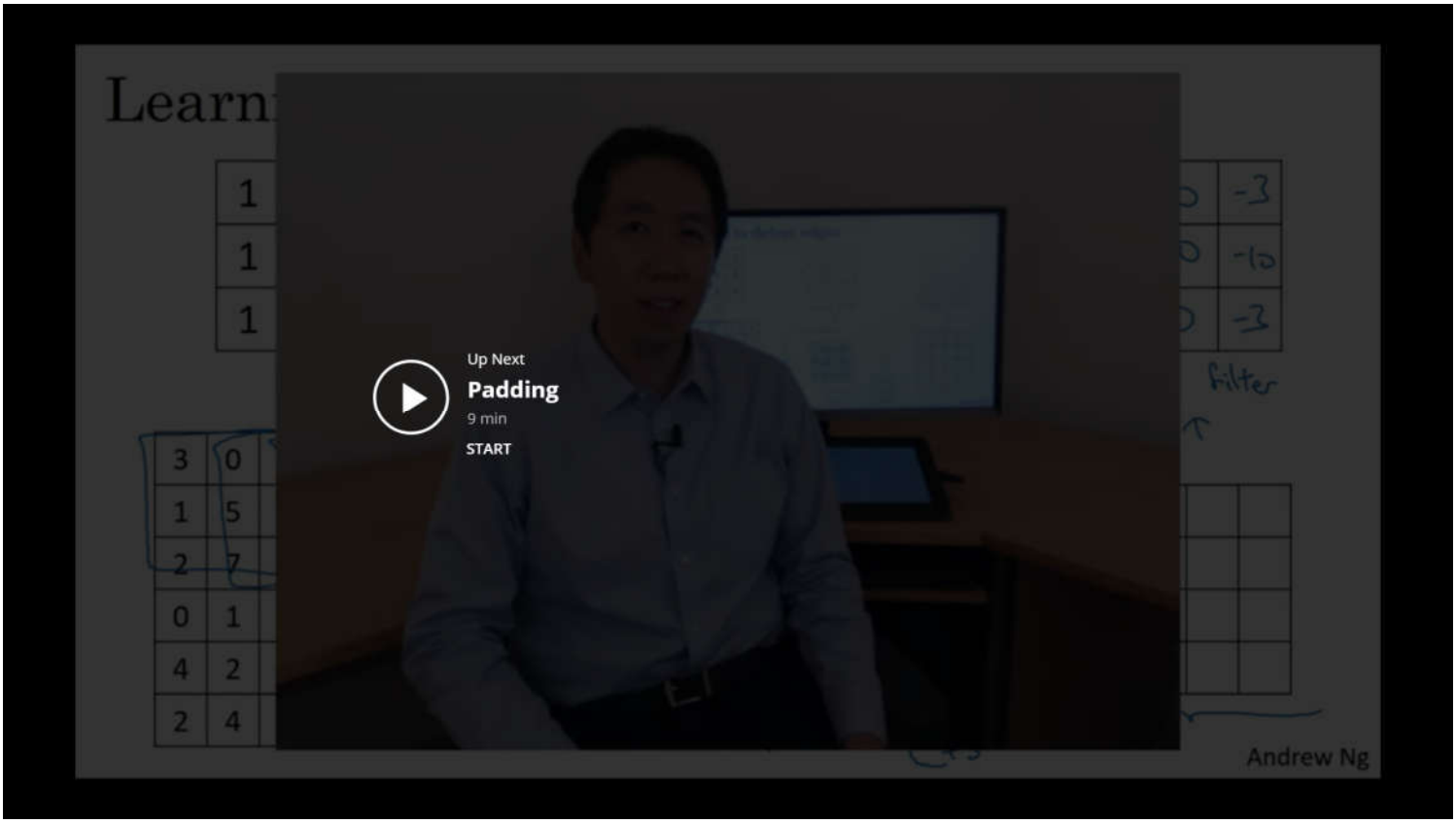


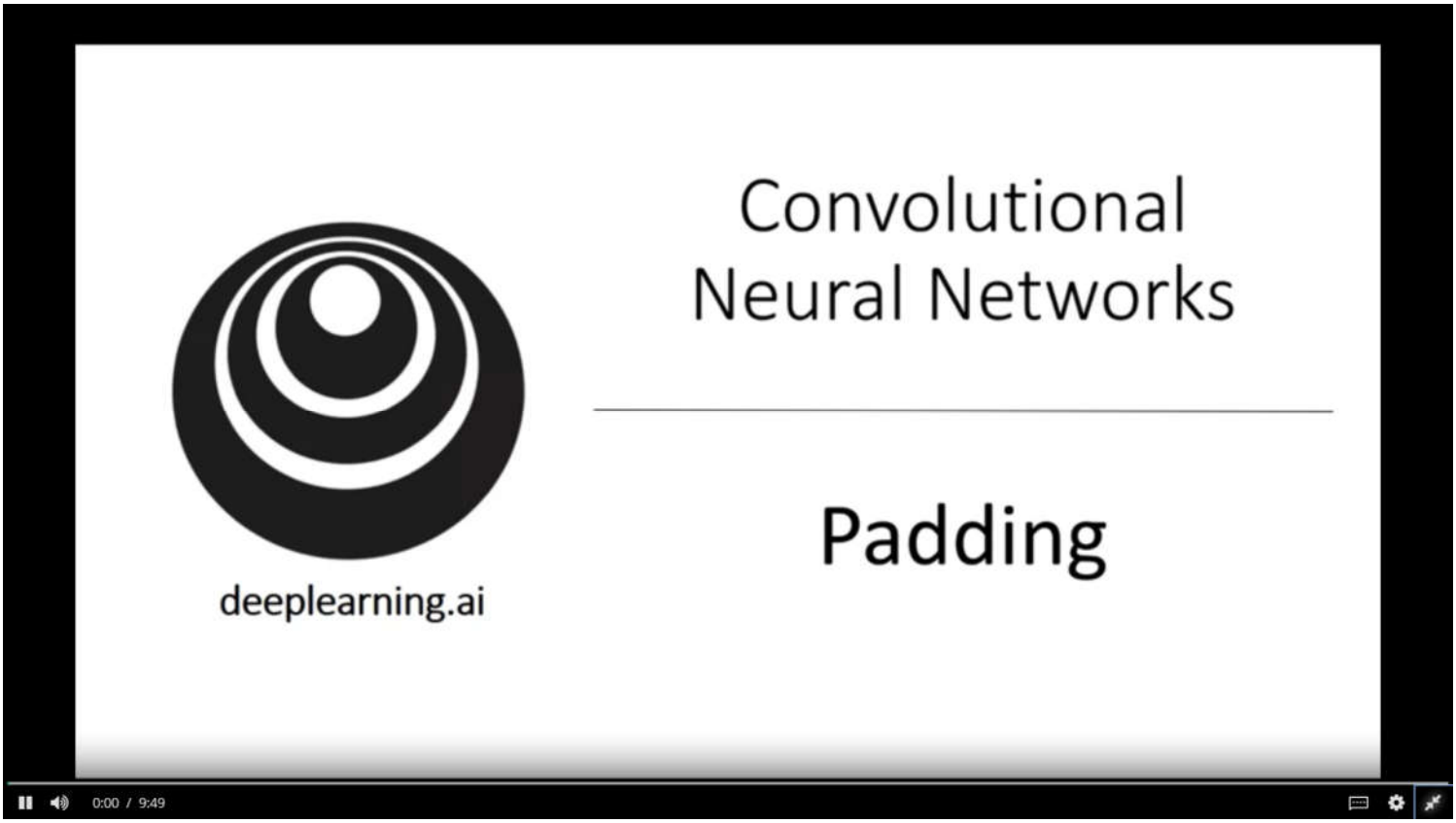
maybe it can learn to detect edges that are at 45 degrees or

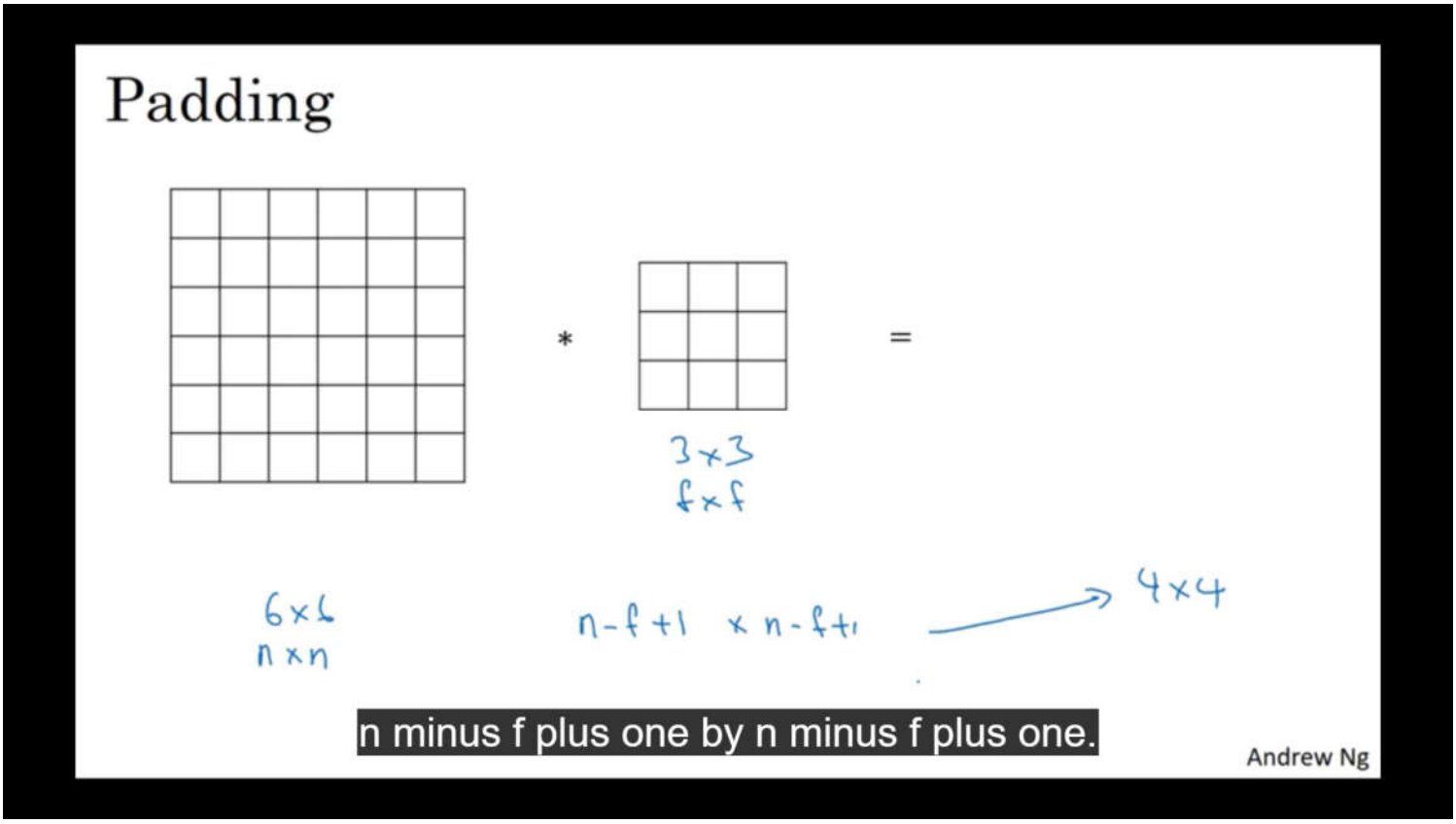


computer vision researchers are generally able to code up these things by hand.

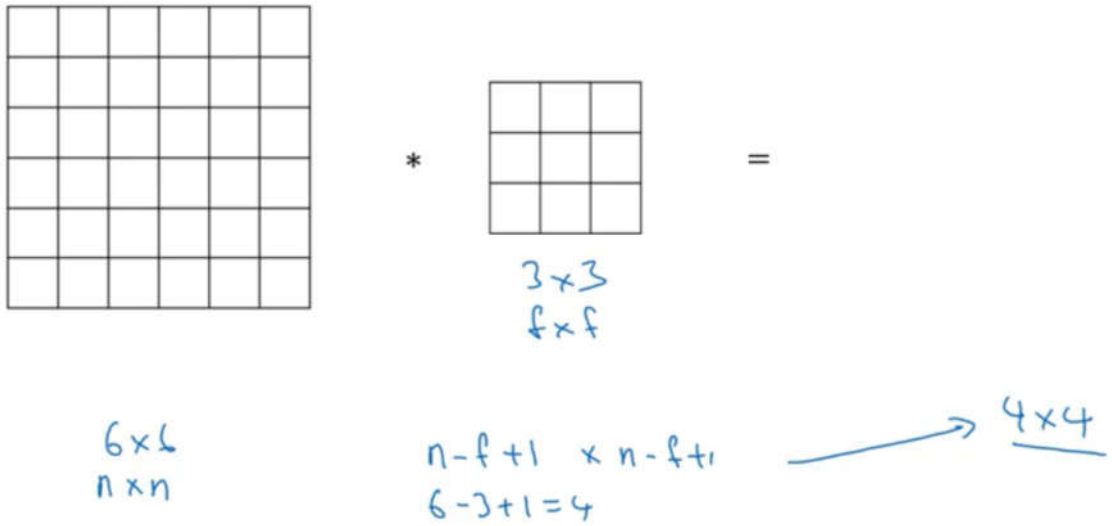

Learning to detect edges
1 0 -1
1 0 -1
1 0 -1
1 0 -1
2 0 -2
1 0 -1
Sobel filter
3 0 -3
10 0 -10
3 0 -3
Scharr filter
3 0 1 2 7 4
1 5 8 9 3 1
2 7 2 5 1 3
0 1 3 1 7 8
4 2 1 6 2 8
2 4 5 2
convolut
w_1 w_2 w_3
w_4 w_5 w_6
w_7 w_8 w_9
=
45°
But underlying all these computations
is still this convolution operation,
Andrew Ng


Up Next
Padding
9 min
START
Andrew Ng


Convolutional
Neural Networks
Padding
deeplearning.ai
0:00 / 9:49


Padding
*
=
3×3
f×f
6×6
n×n
n-f+1 × n-f+1
4×4
n minus f plus one by n minus f plus one.
Andrew Ng

Padding

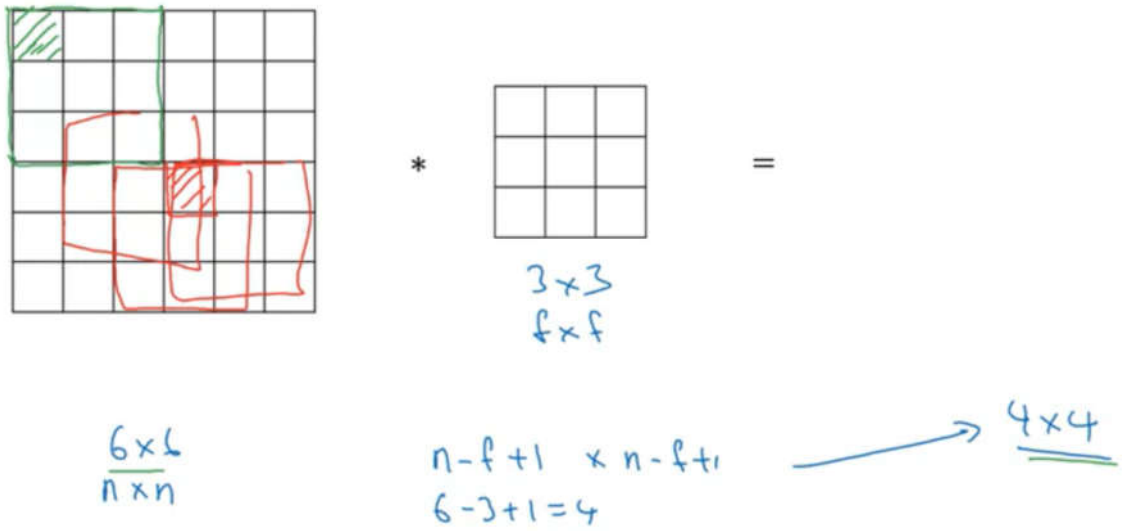


So the two downsides to this; one is that,

Andrew Ng

Padding

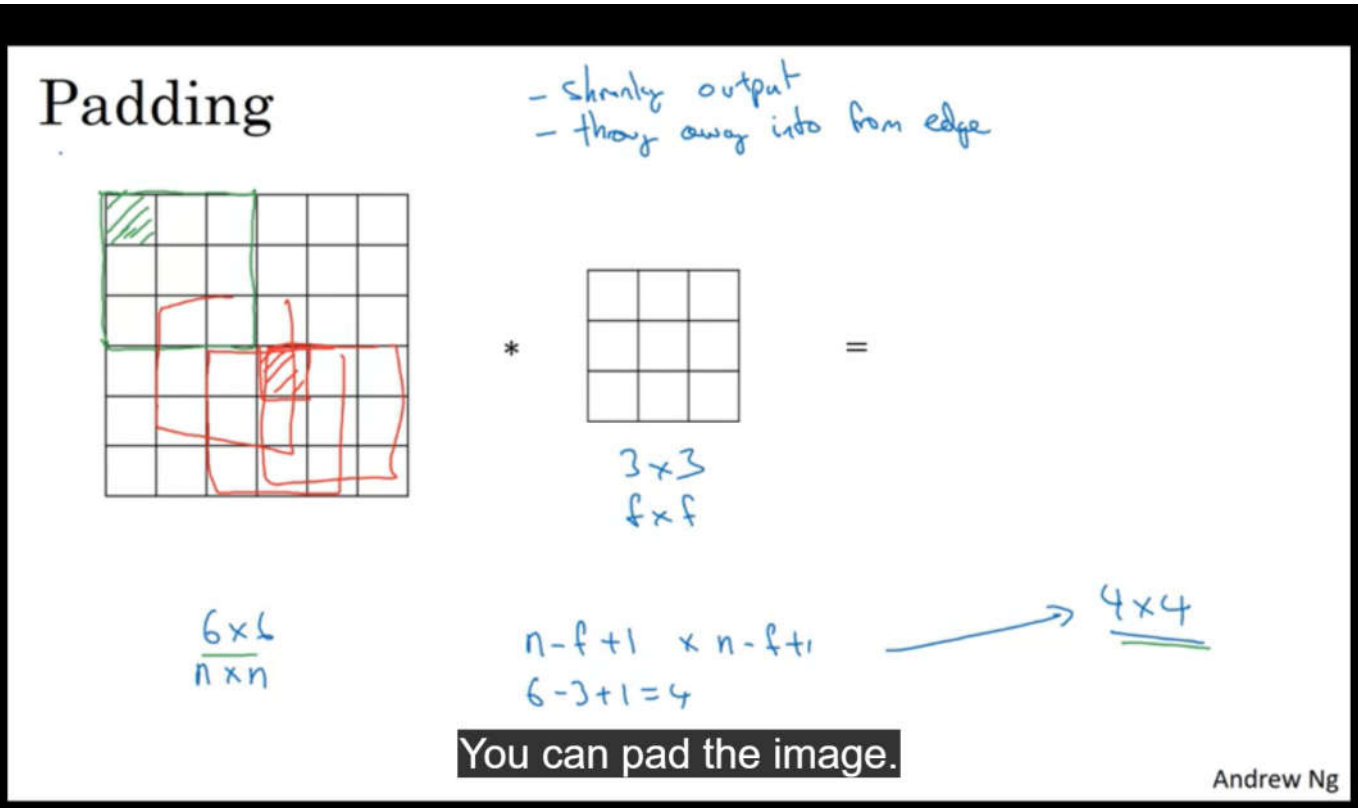
- shrinking output
- throw away info f

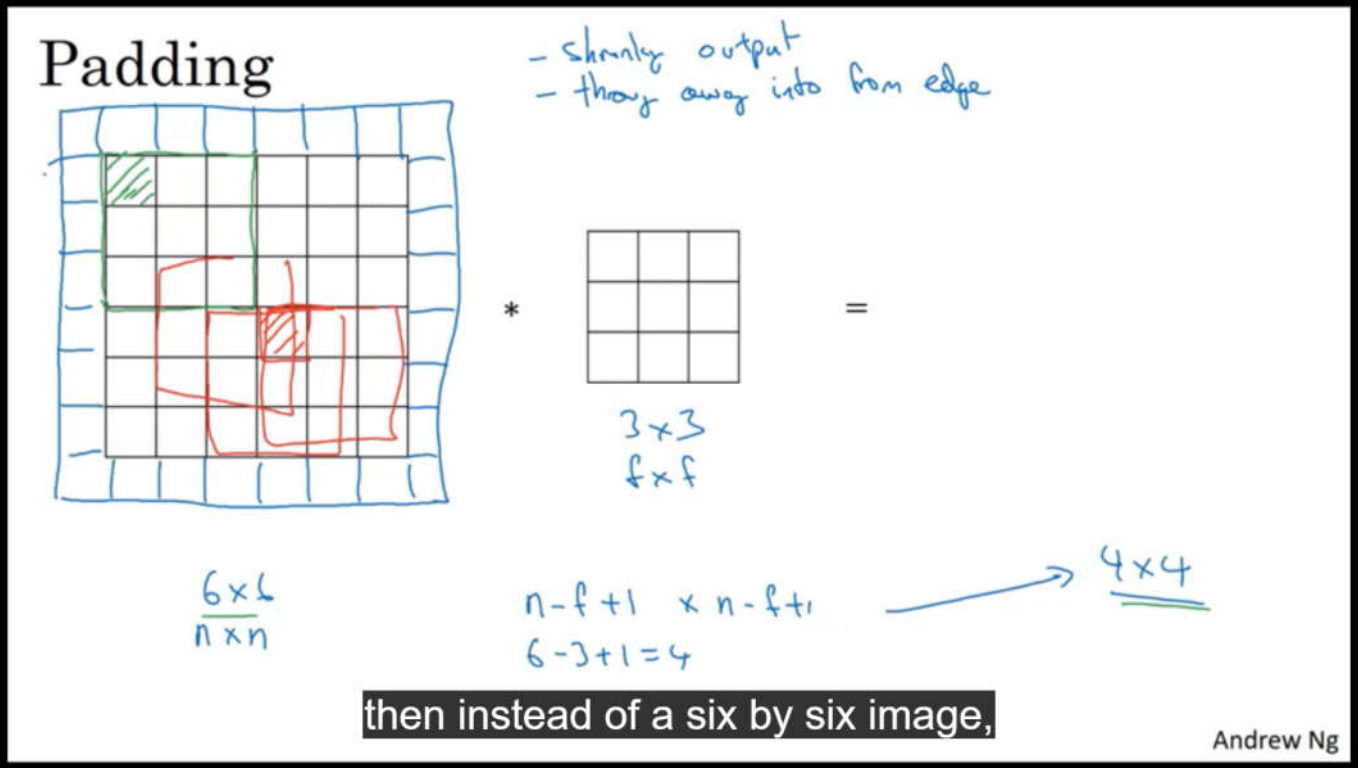


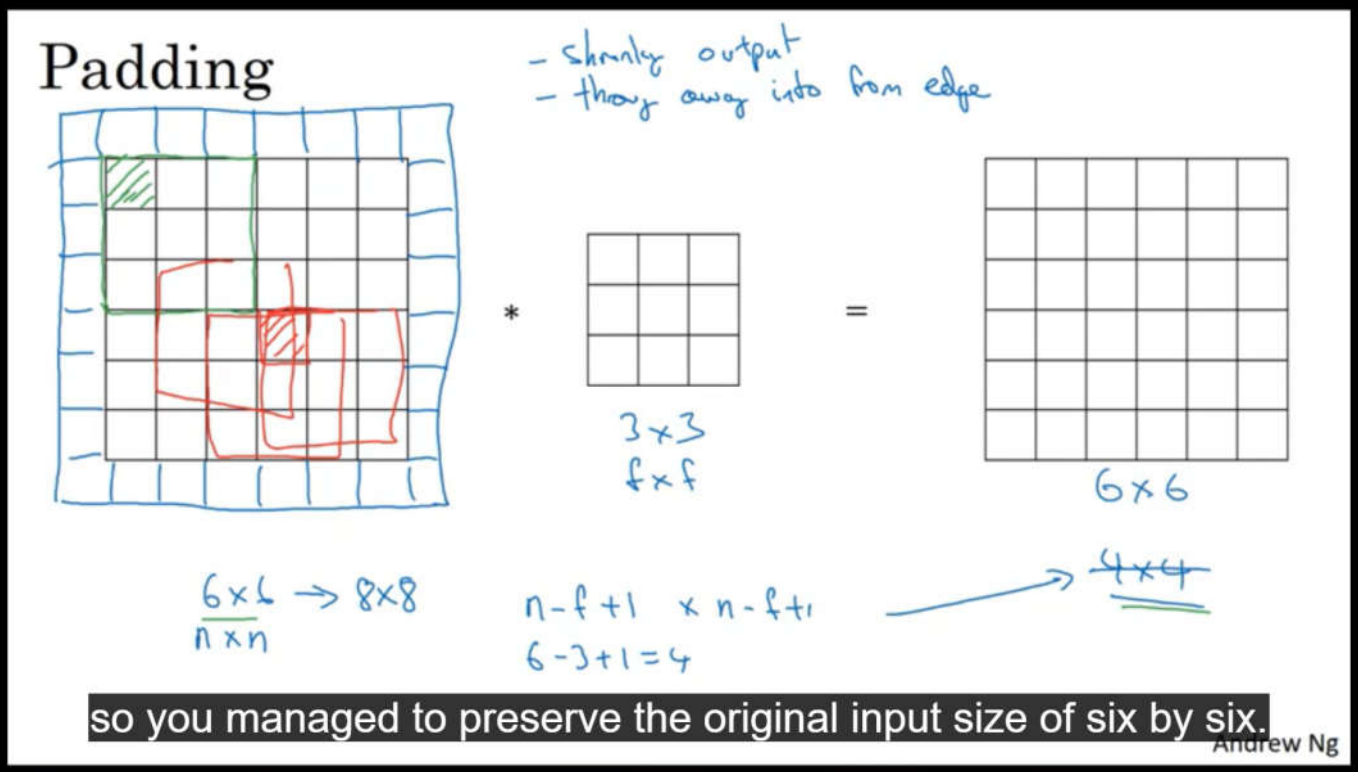
the other is throwing away a lot of the information from the edges of the image.

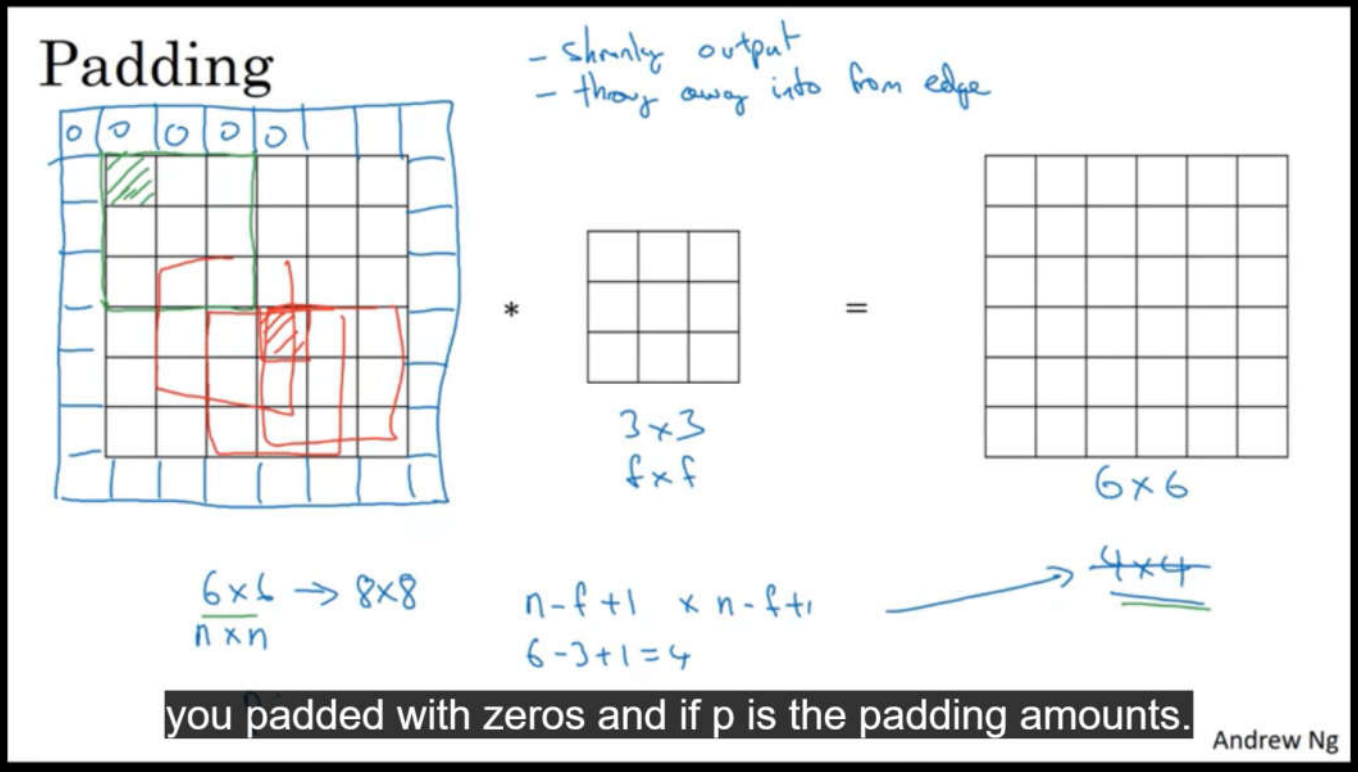
Andrew Ng

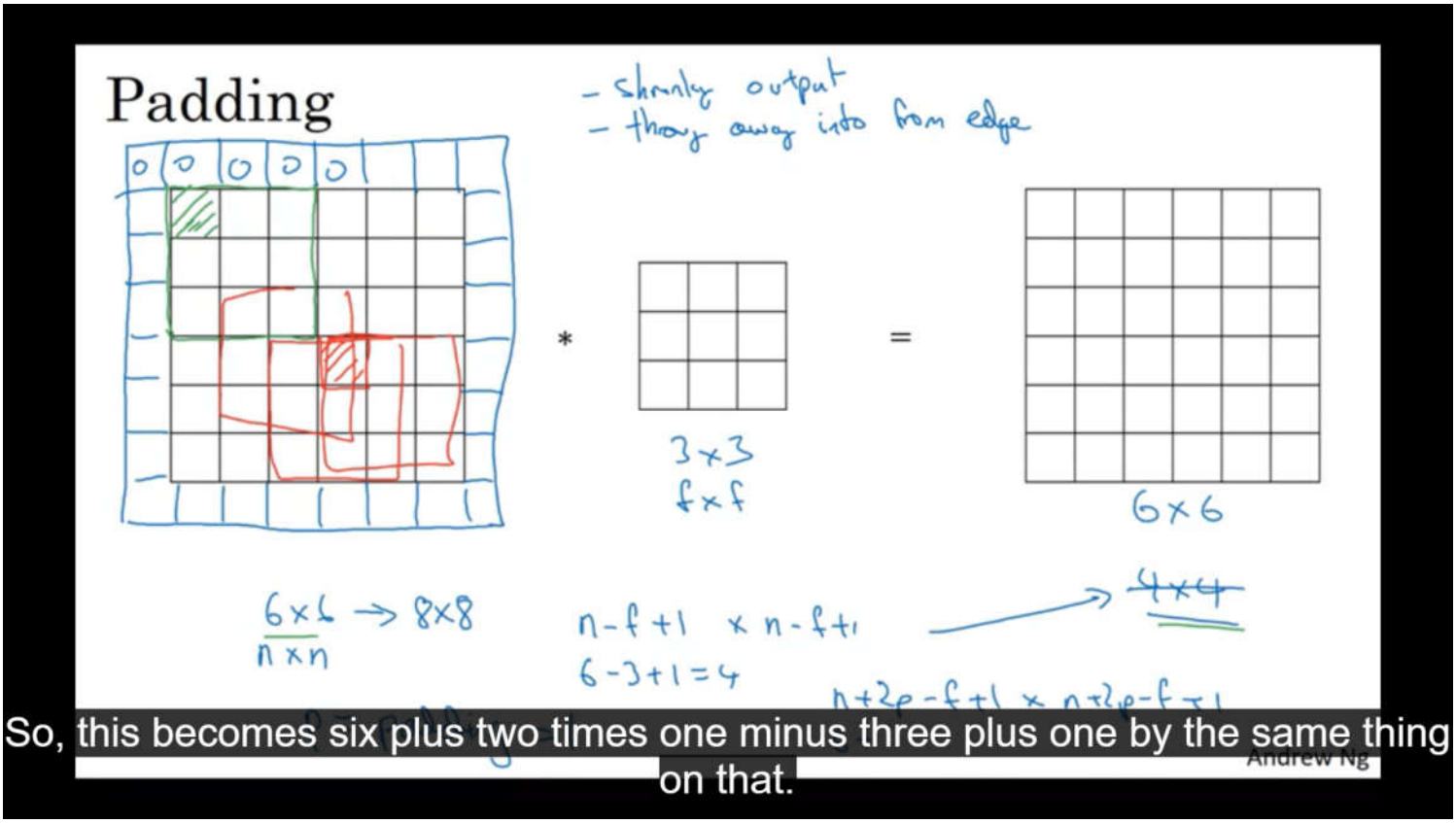
2:35 / 9:49


Padding
- shrinking output
- throw away info from edge
*
=
3×3
f×f
6×6
n×n
n-f+1 × n-f+1
6-3+1=4
4×4
You can pad the image.
Andrew Ng

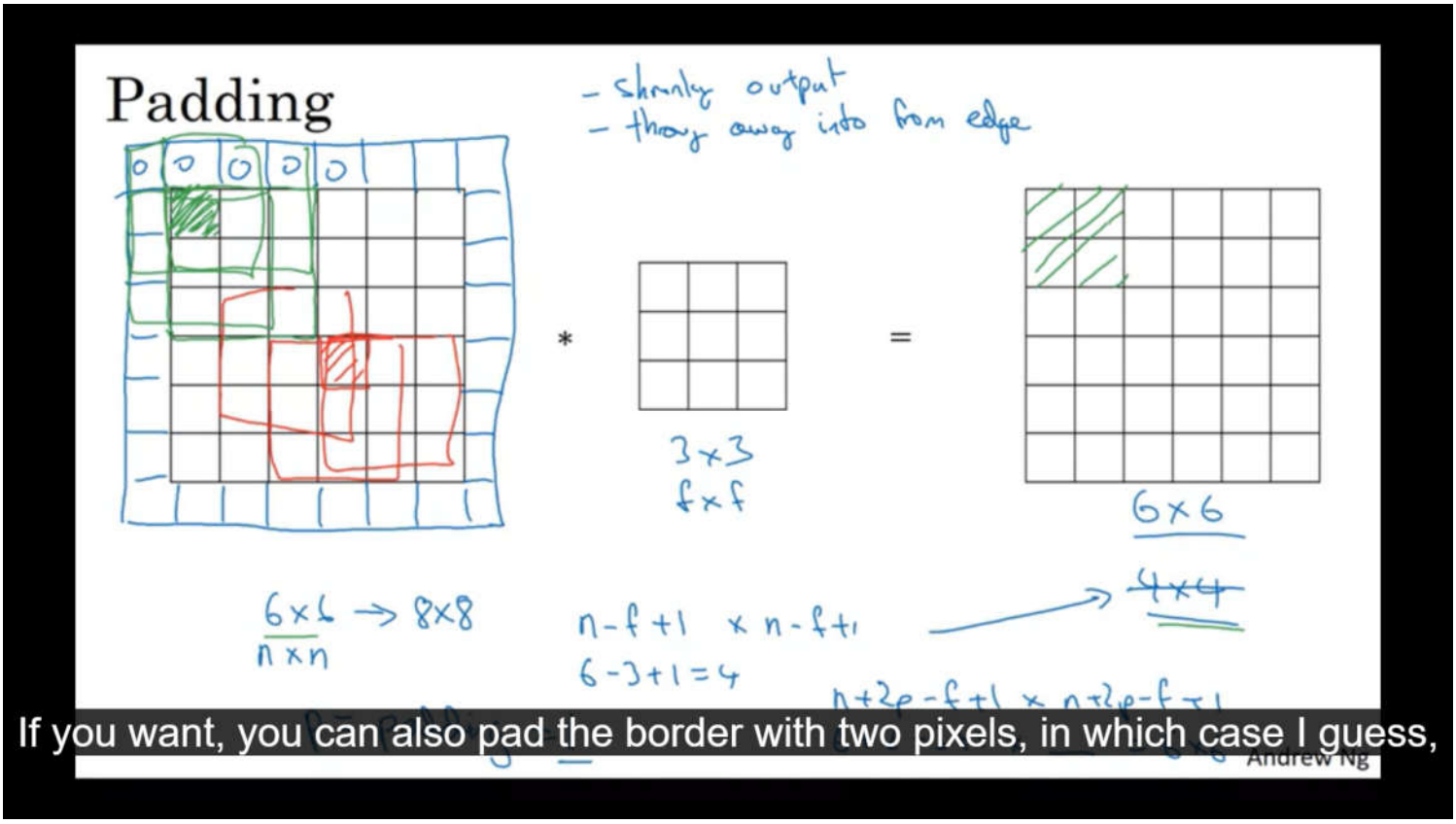

Padding
- shrinking output
- throw away info from edge
*
=
3×3
f×f
6×6
n×n
n-f+1 × n-f+1
6-3+1=4
4×4
then instead of a six by six image,
Andrew Ng


Padding
- shrinking output
- throw away info from edge
*
=
3×3
f×f
6×6
6×6 → 8×8
n×n
n-f+1 × n-f+1
6-3+1=4
4×4
so you managed to preserve the original input size of six by six.
Andrew Ng


Padding
- shrinking output
- throw away info from edge
0 0 0 0 0
*
=
3×3
f×f
6×6
6×6 → 8×8
n×n
n-f+1 × n-f+1
6-3+1=4
4×4
you padded with zeros and if p is the padding amounts.
Andrew Ng



So, this becomes six plus two times one minus three plus one by the same thing on that.



If you want, you can also pad the border with two pixels, in which case I guess,

Valid and Same convolutions

Valid convolutions and Same convolutions.

Andrew Ng

Valid and Same convolutions

no padding

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

f plus one by n minus f plus one dimensional output.

Andrew Ng

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.

the same convolution and that means when you pad,

Andrew Ng

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.

$n+2p-f+1$

then the output sizes of this dimension is xn plus 2p minus f plus one.

Andrew Ng

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.

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

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

So when f is odd,

Andrew Ng

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.

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

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

when the filter was three by three as this had happened in the previous slide,

Andrew Ng

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.

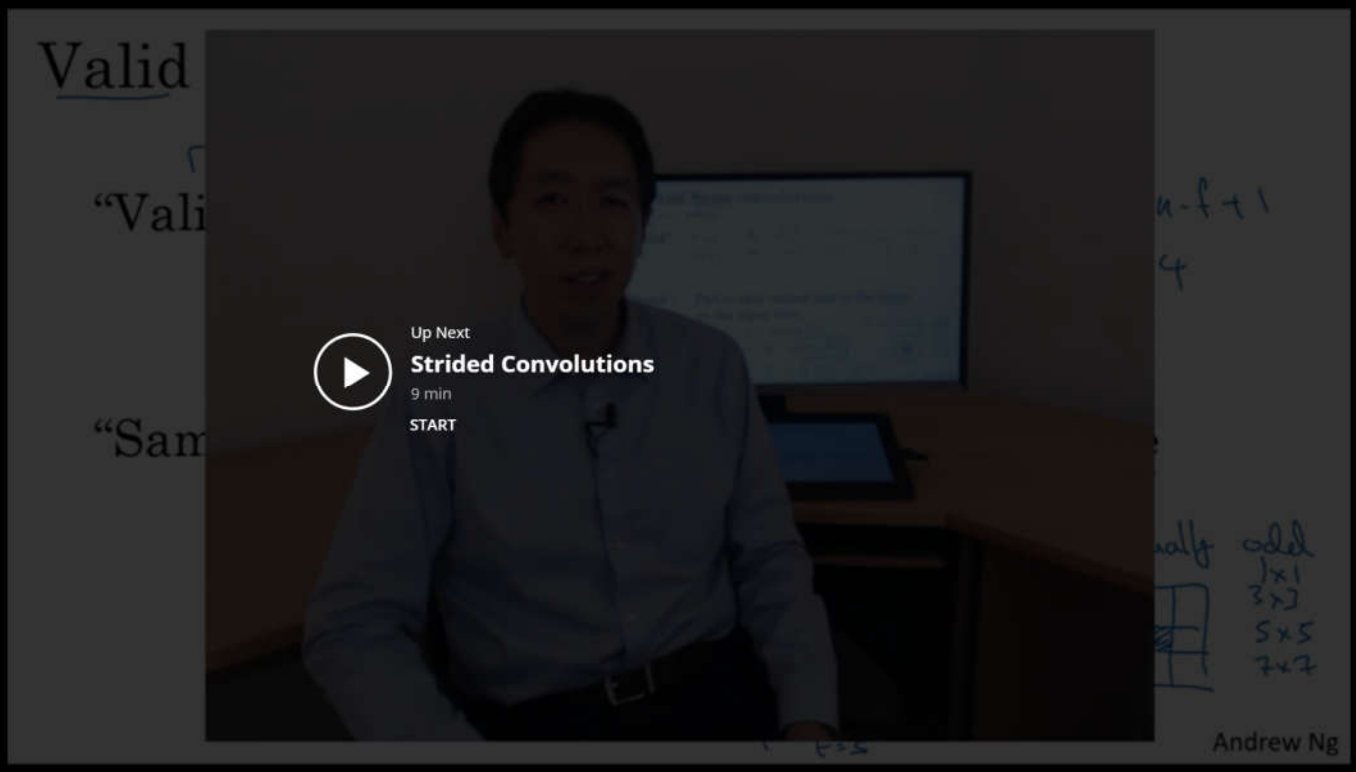
f is usually odd

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

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

then you need some asymmetric padding.

Andrew Ng



Press Esc to exit full screen

Convolutional Neural Networks

deeplearning.ai

Strided convolutions

Strided convolutions is another piece of

0:00 / 9:01

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

$*$

3	4	4
1	0	2
-1	0	3

$=$

Let's say you want to convolve this seven by seven image with this three by three filter,

Andrew Ng

0:12 / 9:01

Strided convolution

2	3	7^{3}	4^{4}	6^{4}	2	9
6	6	9^{1}	8^{0}	7^{2}	4	3
3	4	8^{-1}	3^{0}	8^{3}	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		

And then you do the usual element Y's product and summing it turns out 100.

Andrew Ng

Strided convolution

2	3	7	4	6^{3}	2^{4}	9^{4}
6	6	9	8	7^{1}	4^{0}	3^{2}
3	4	8	3	8^{-1}	9^{0}	7^{3}
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

You end up there, and that gives you 83.

Andrew Ng

Strided convolution

2	3	7	4	6	2	9
6	6	9	8	7	4	3
3^{3}	4^{4}	8^{4}	3	8	9	7
7^{1}	8^{0}	3^{2}	6	6	3	4
4^{-1}	2^{0}	1^{3}	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

Notice how we are stepping over one of the positions and then this gives you 69,

Andrew Ng

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^{3}	4^{4}	6^{4}
3	2	4	1	9^{1}	8^{0}	3^{2}
0	1	3	9	2^{-1}	1^{0}	4^{3}

7×7

$*$

3	4	4
1	0	2
-1	0	3

3×3

stride =2

$=$

91	100	83
69	91	127
44	72	74

And then for the final row 44, 72, and 74.

Andrew Ng

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^{3}	4^{4}	6^{4}
3	2	4	1	9^{1}	8^{0}	3^{2}
0	1	3	9	2^{-1}	1^{0}	4^{3}

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

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

padding p stride s

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

In our example, we have seven plus zero, minus three,

Andrew Ng

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^{3}	4^{4}	6^{4}
3	2	4	1	9^{1}	8^{0}	3^{2}
0	1	3	9	2^{-1}	1^{0}	4^{3}

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

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

padding p stride s

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

down so this notation denotes the flow of something.

Andrew Ng

Summary of convolutions

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

padding $p \quad$ stride s

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

then the output size will have this dimension.

Andrew Ng

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

cross-correlation versus convolutions and just for

Andrew Ng

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

the horizontal as well as the vertical axis so this 345102 minus 197,

Andrew Ng

Technical note on cross-correlation vs. convolution

Convolution in math textbook:

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

*

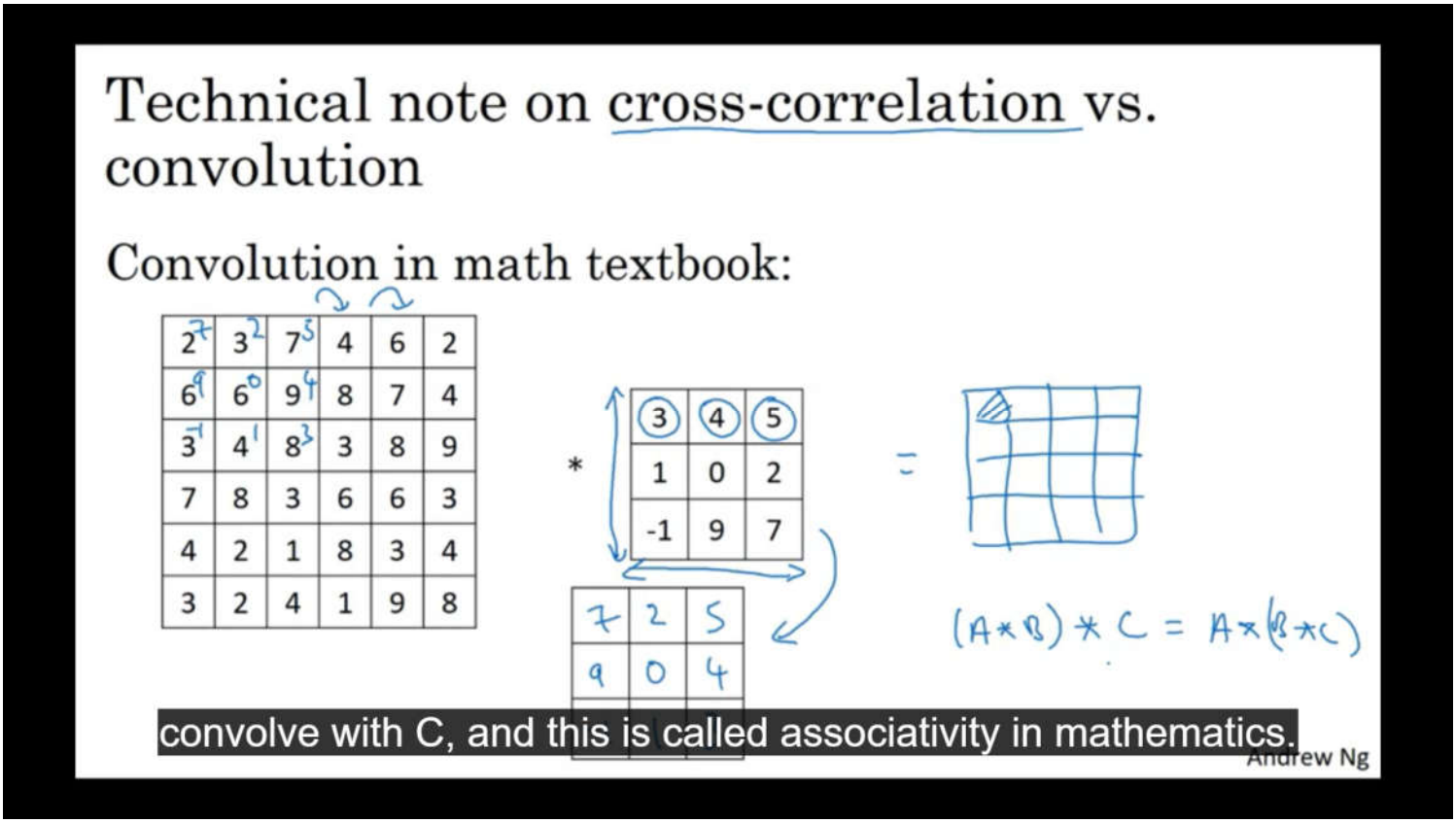
3	4	5
1	0	2
-1	9	7

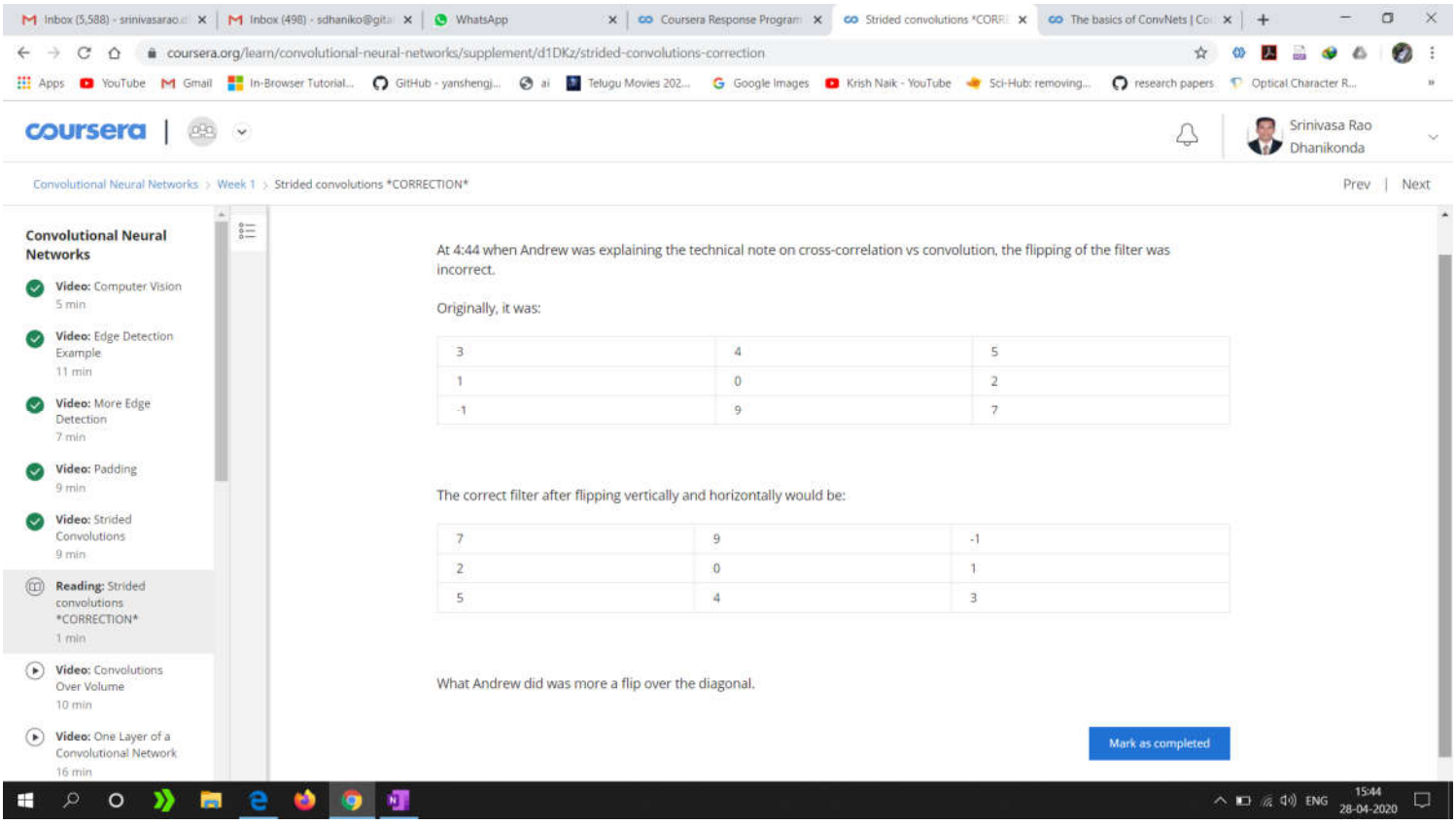
=

7	2	5
9	0	4

compute the upper left hand rows elements of the four by four output as follows.

Andrew Ng


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
(A*B)*C = A*(B*C)
convolve with C, and this is called associativity in mathematics.
Andrew Ng


coursera.org/learn/convolutional-neural-networks/supplement/d1DKz/strided-convolutions-correction
coursera
Srinivasa Rao Dhanikonda
Convolutional Neural Networks > Week 1 > Strided convolutions *CORRECTION*
Prev | Next
Convolutional Neural Networks
Video: Computer Vision 5 min
Video: Edge Detection Example 11 min
Video: More Edge Detection 7 min
Video: Padding 9 min
Video: Strided Convolutions 9 min
Reading: Strided convolutions *CORRECTION* 1 min
Video: Convolutions Over Volume 10 min
Video: One Layer of a Convolutional Network 16 min
At 4:44 when Andrew was explaining the technical note on cross-correlation vs convolution, the flipping of the filter was incorrect.
Originally, it was:
3 4 5
1 0 2
-1 9 7
The correct filter after flipping vertically and horizontally would be:
7 9 -1
2 0 1
5 4 3
What Andrew did was more a flip over the diagonal.
Mark as completed

Press Esc to exit full screen

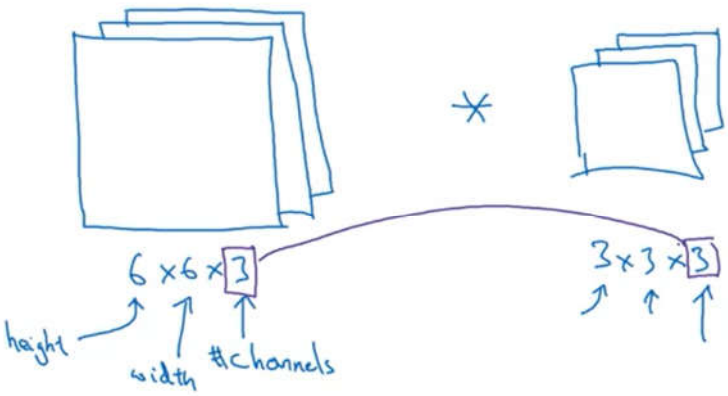
deeplearning.ai

Convolutional Neural Networks

Convolutions over volumes

0:00 / 10:44

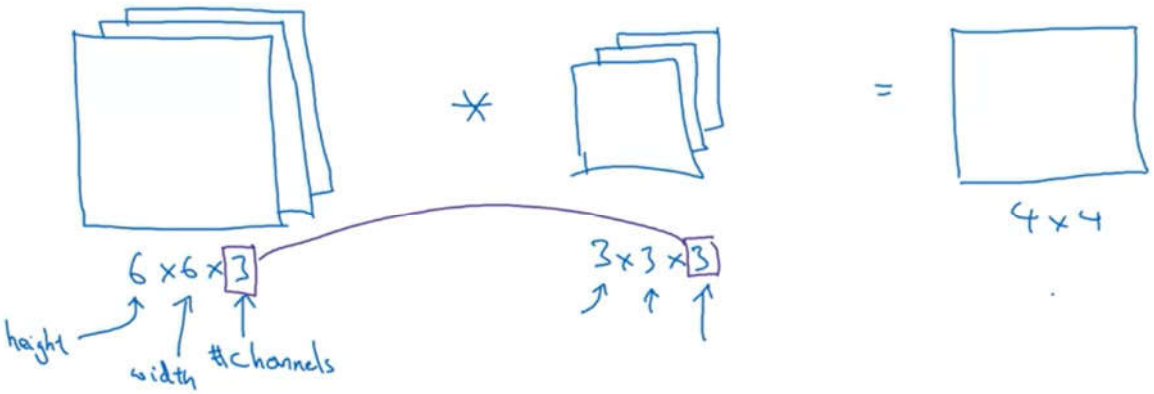
Convolutions on RGB images



so these two numbers have to be equal.

Andrew Ng

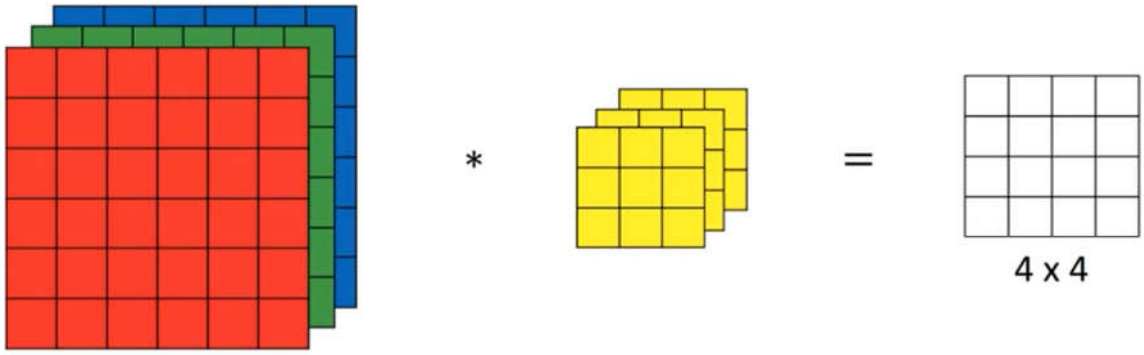
Convolutions on RGB images



but the output of this will be a four by four image.

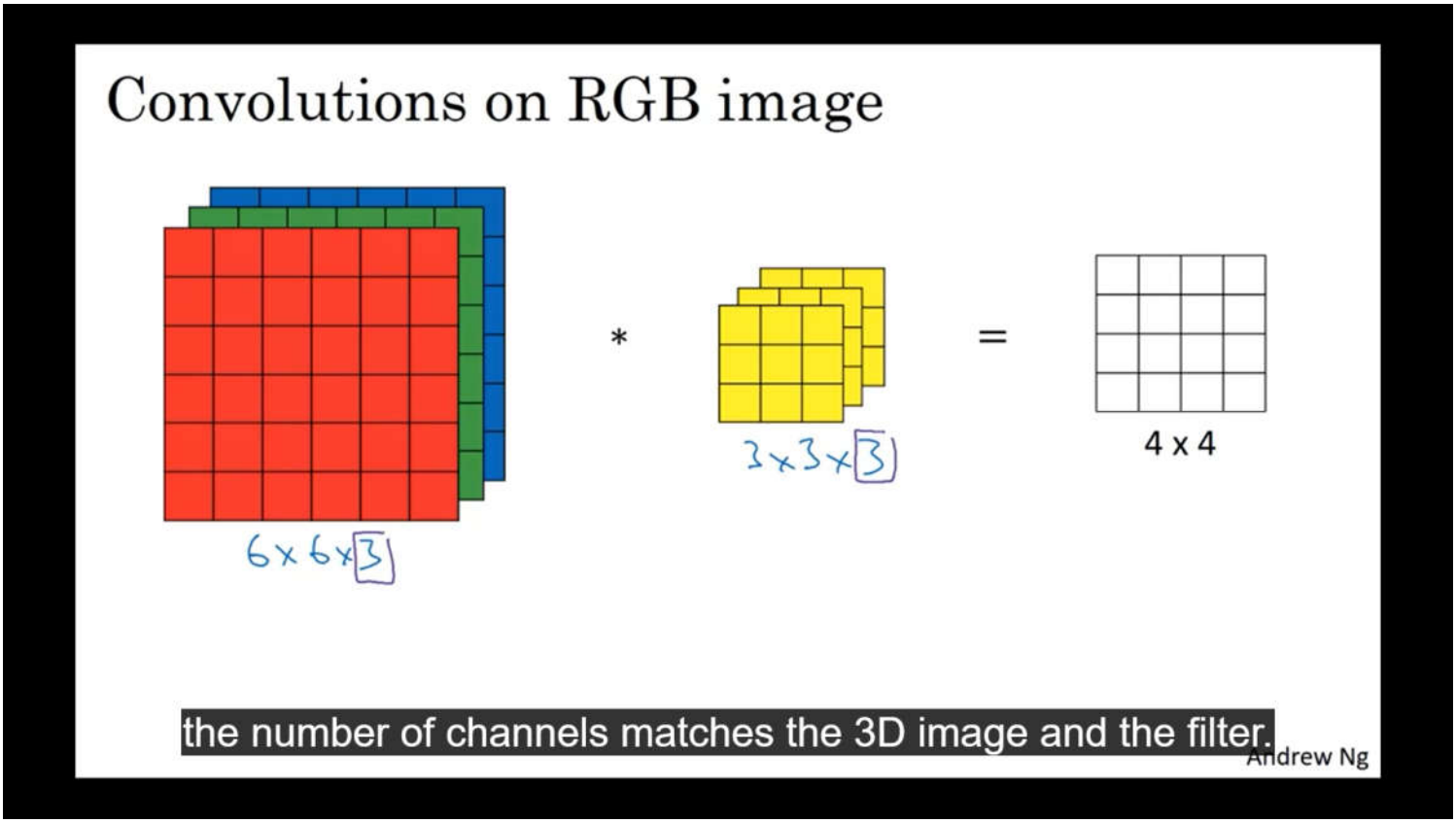
Andrew Ng

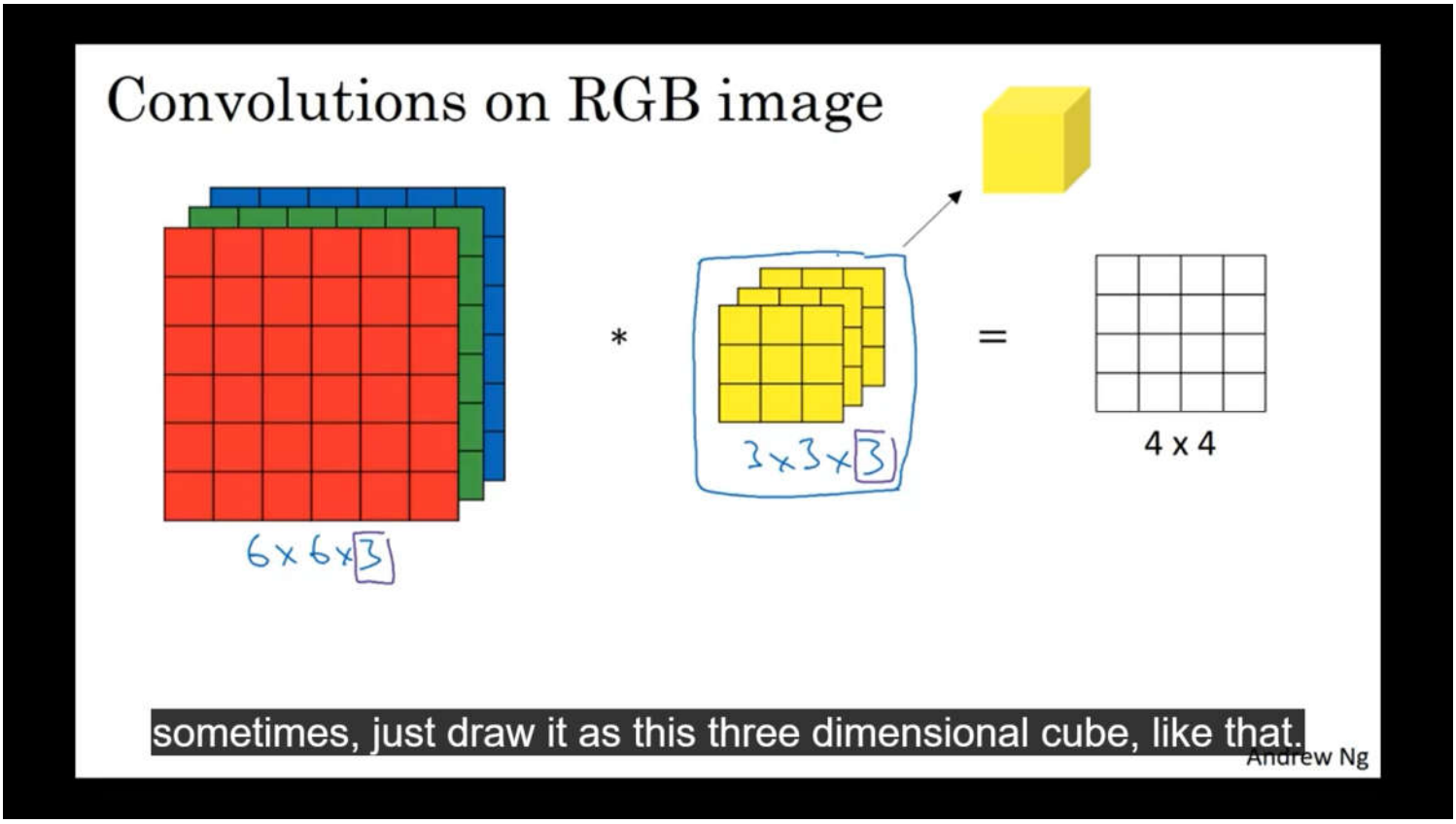
Convolutions on RGB image

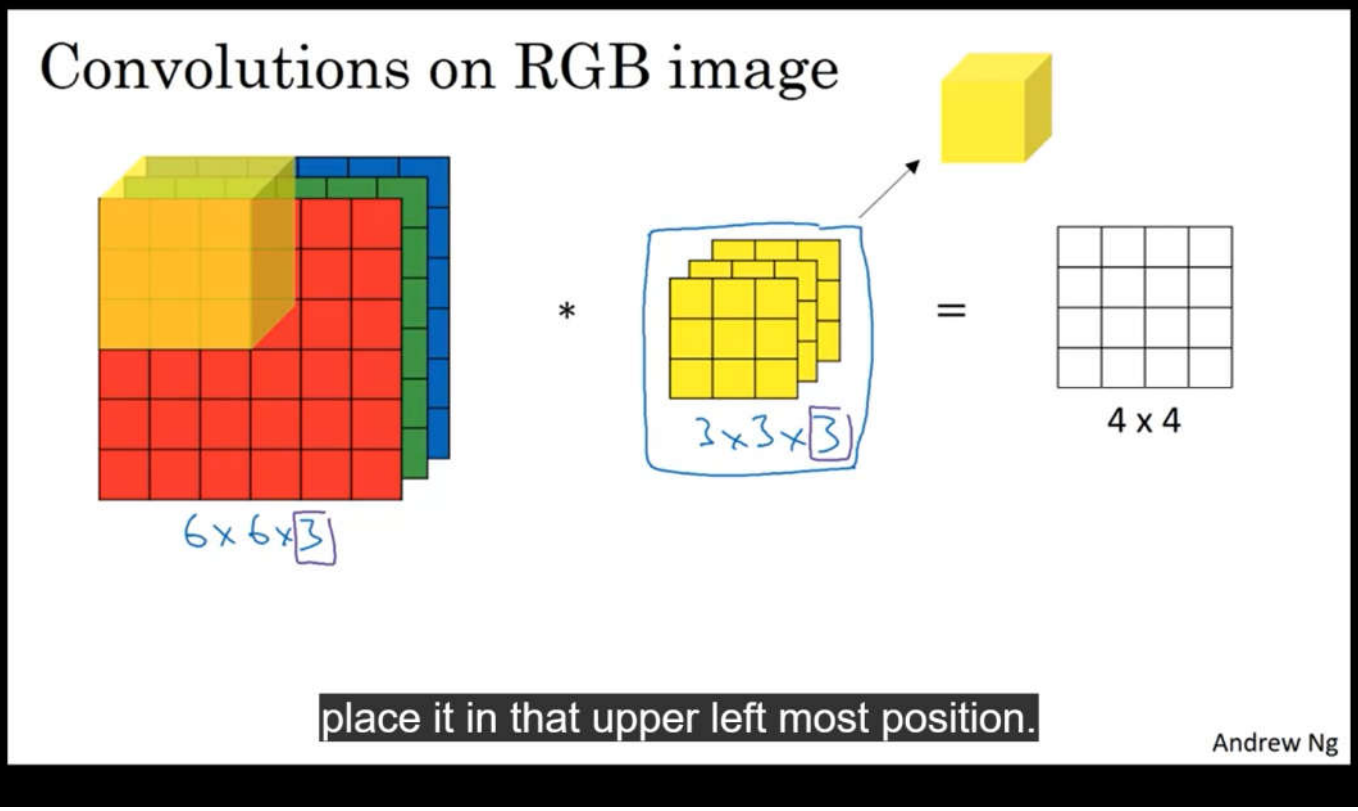


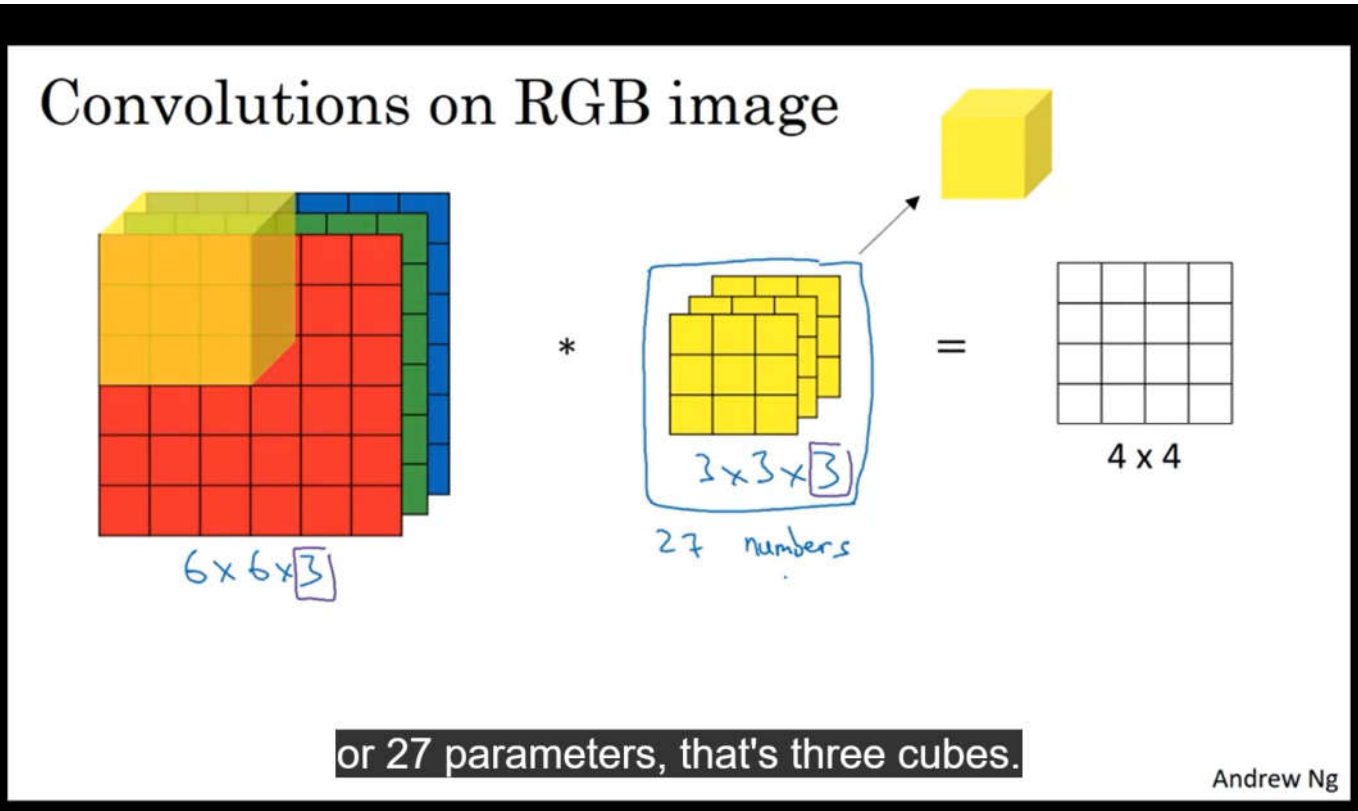
Let's go through in detail how this works but let's use a more nicely drawn image.

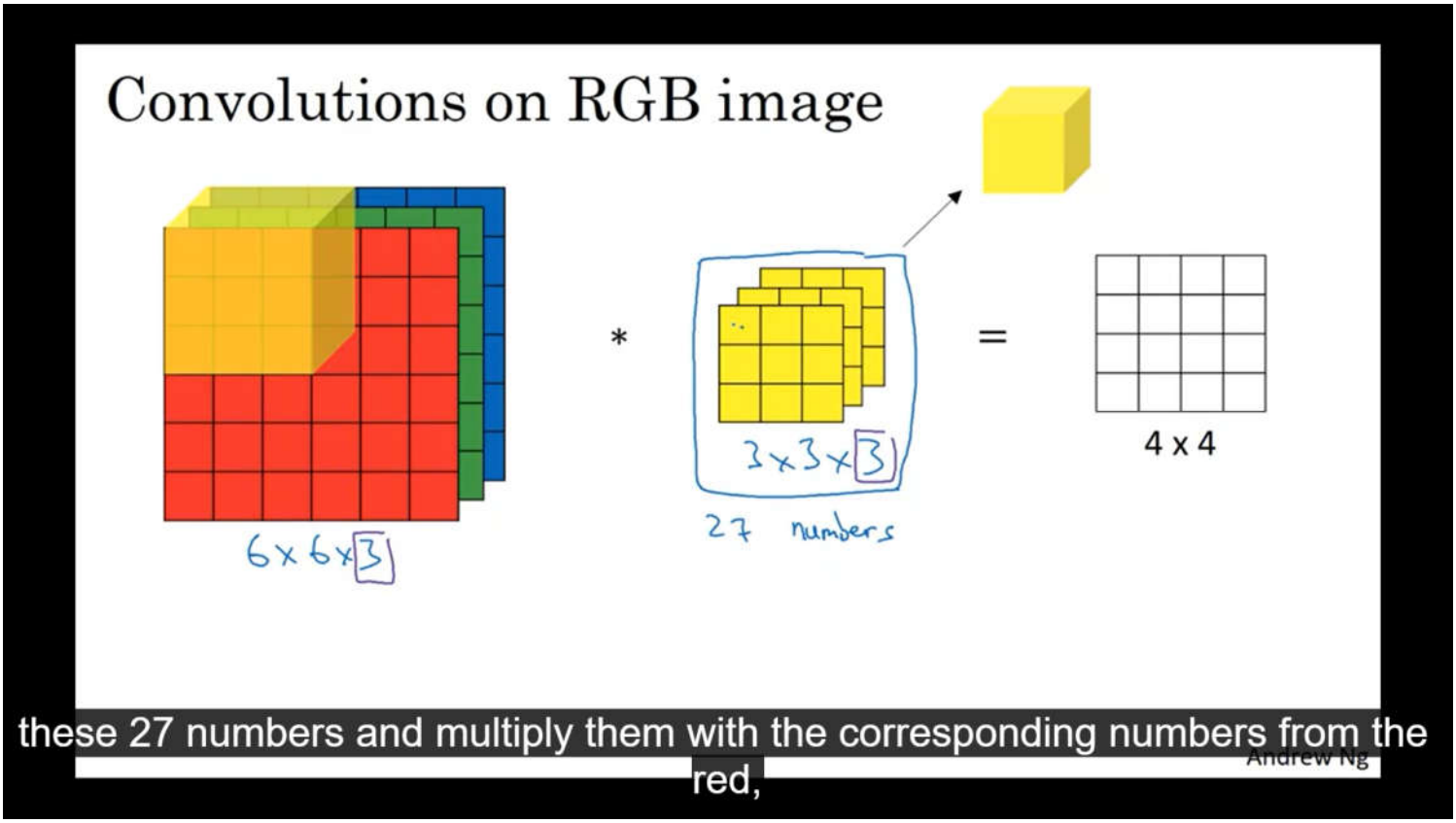
Andrew Ng

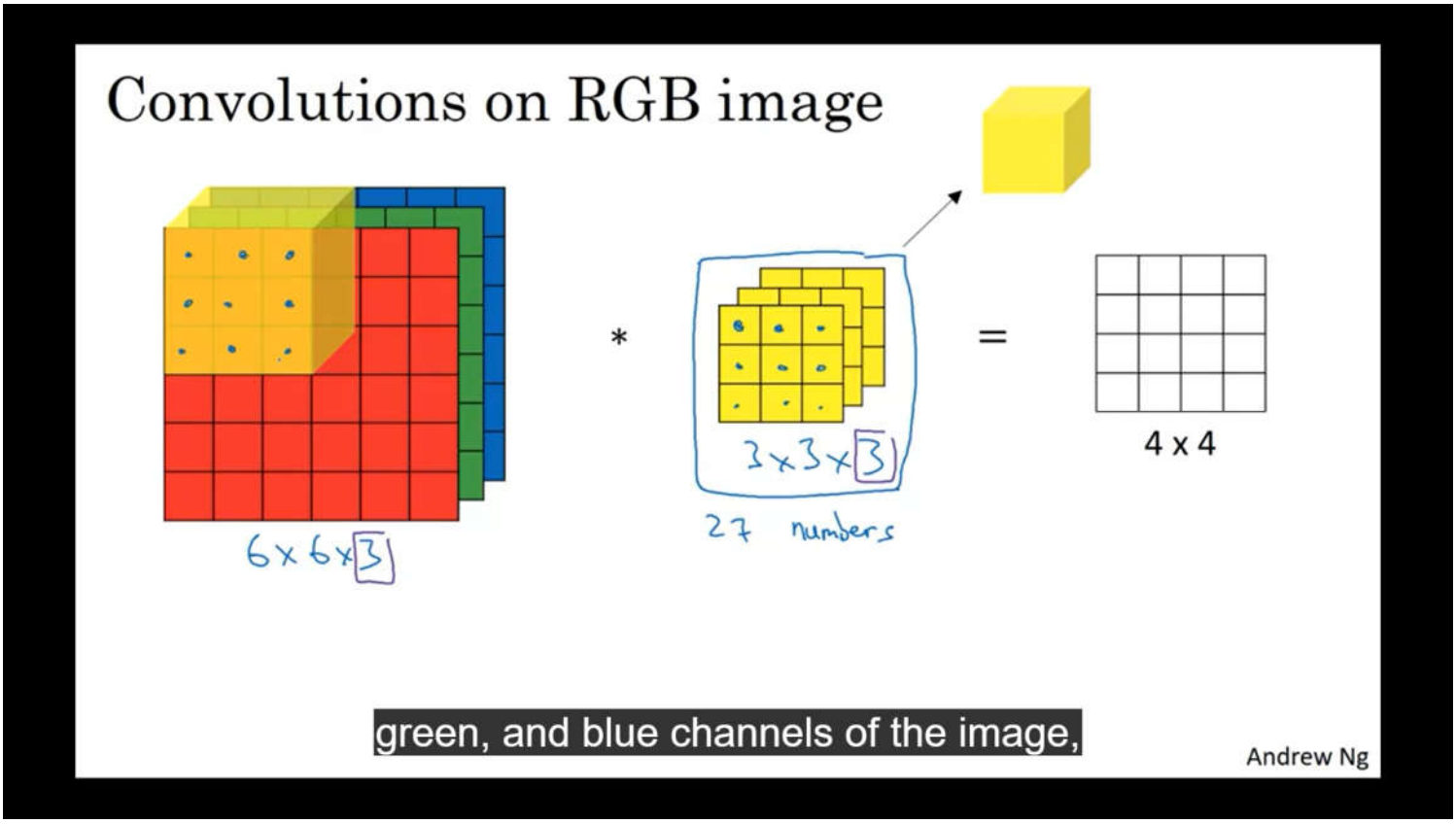

Convolutions on RGB image
*
=
3x3x3
4 x 4
6x6x3
the number of channels matches the 3D image and the filter.
Andrew Ng

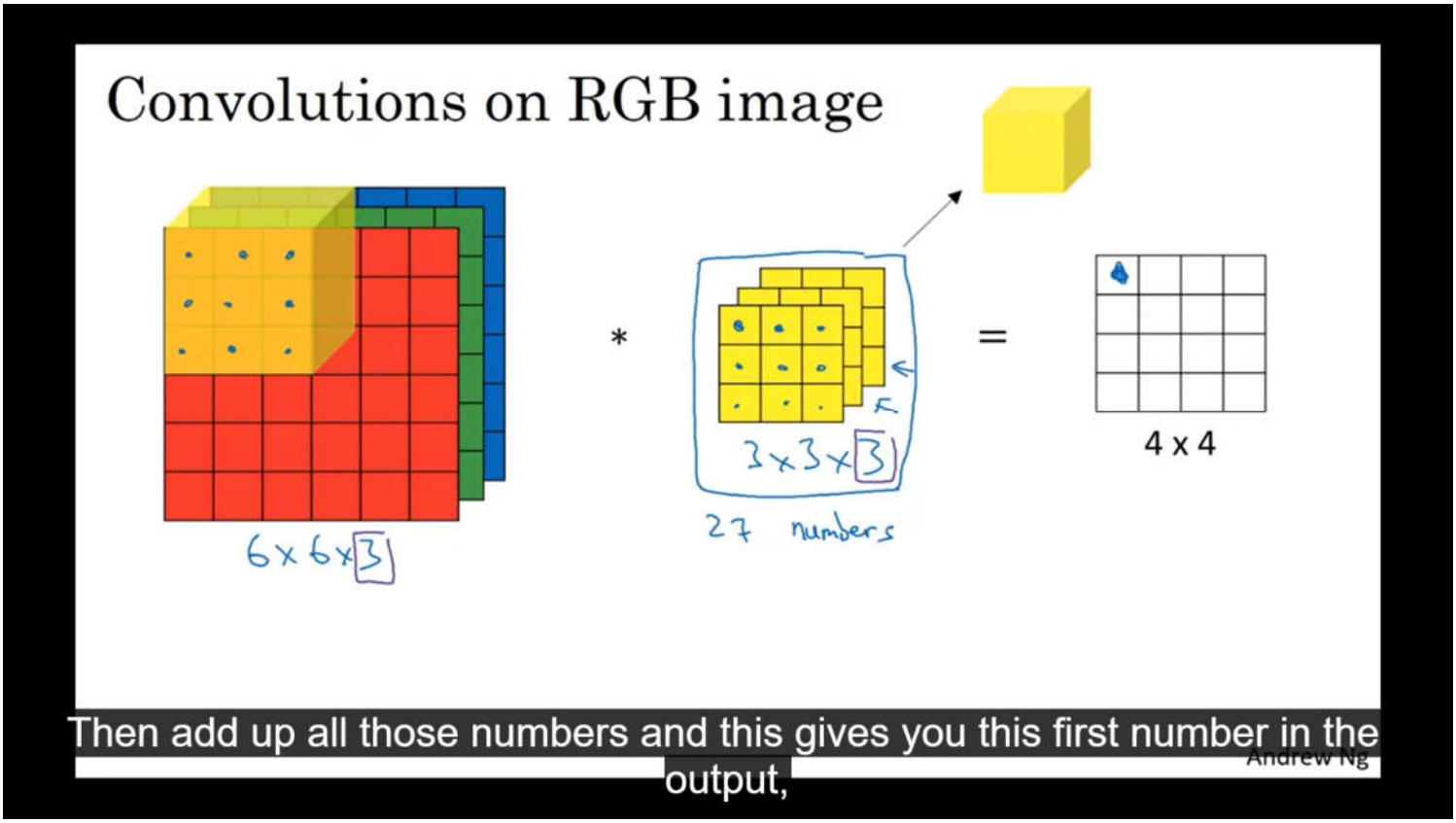

Convolutions on RGB image
*
=
3x3x3
4 x 4
6x6x3
sometimes, just draw it as this three dimensional cube, like that.
Andrew Ng

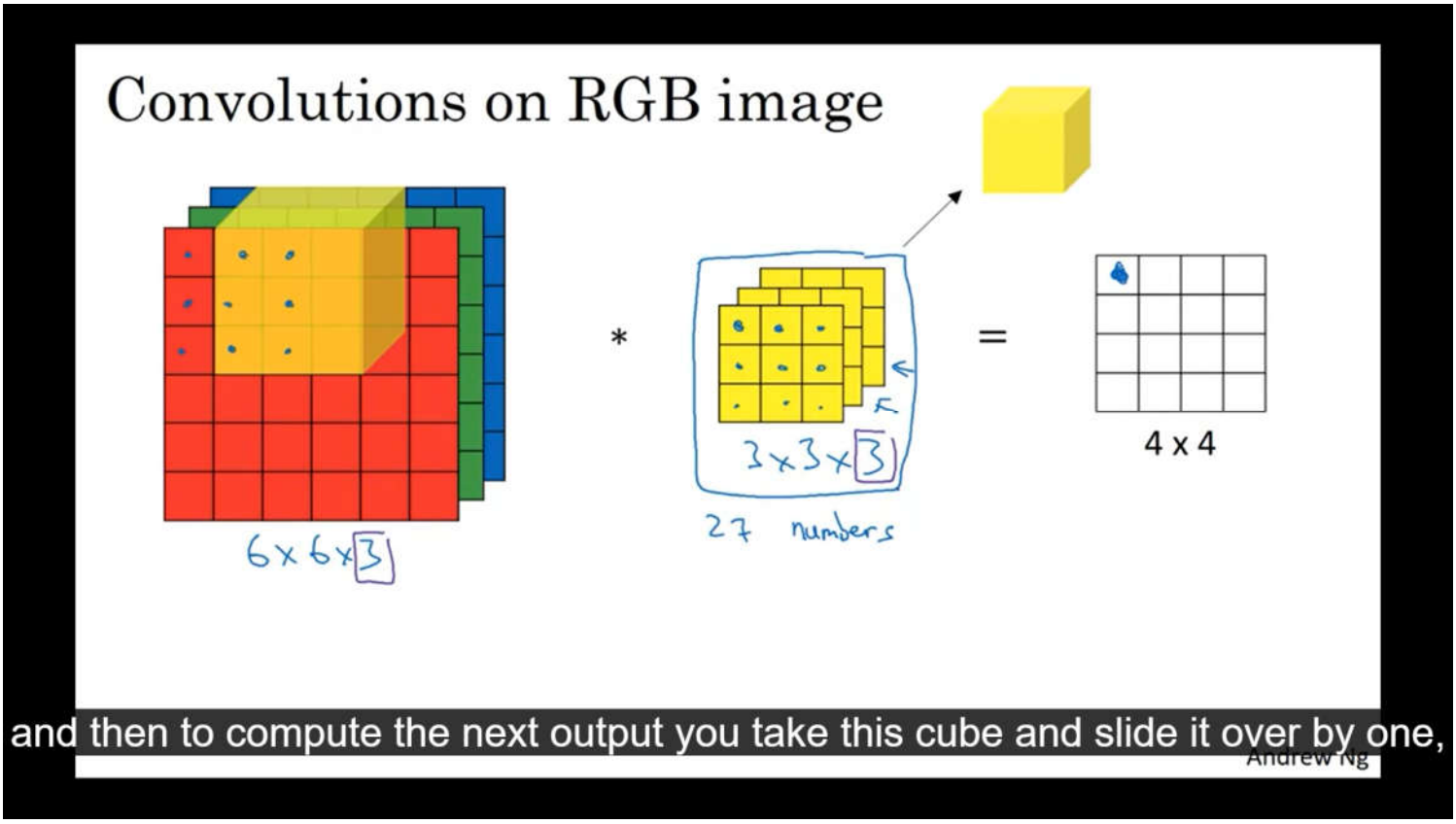

Convolutions on RGB image
*
=
4 x 4
3x3x3
6x6x3
place it in that upper left most position.
Andrew Ng

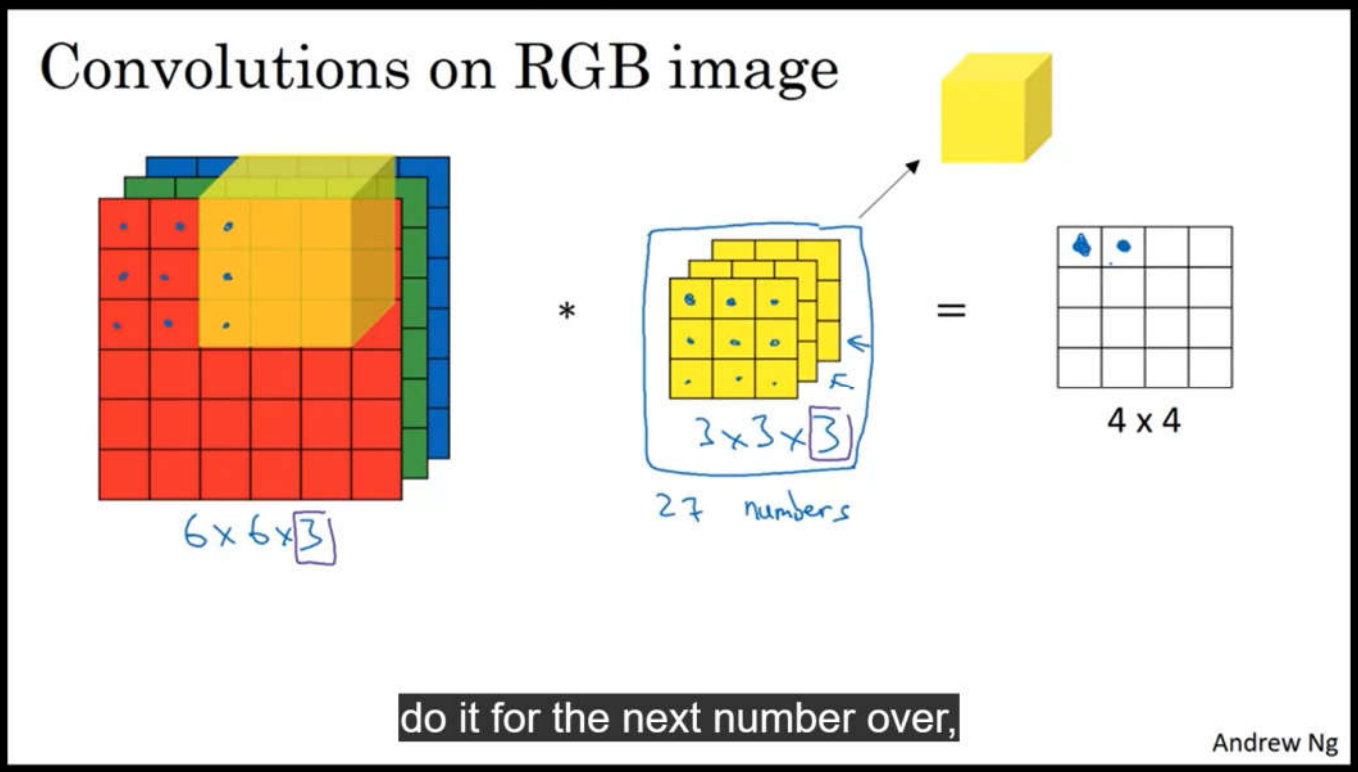

Convolutions on RGB image
*
=
4 x 4
3x3x3
27 numbers
6x6x3
or 27 parameters, that's three cubes.
Andrew Ng

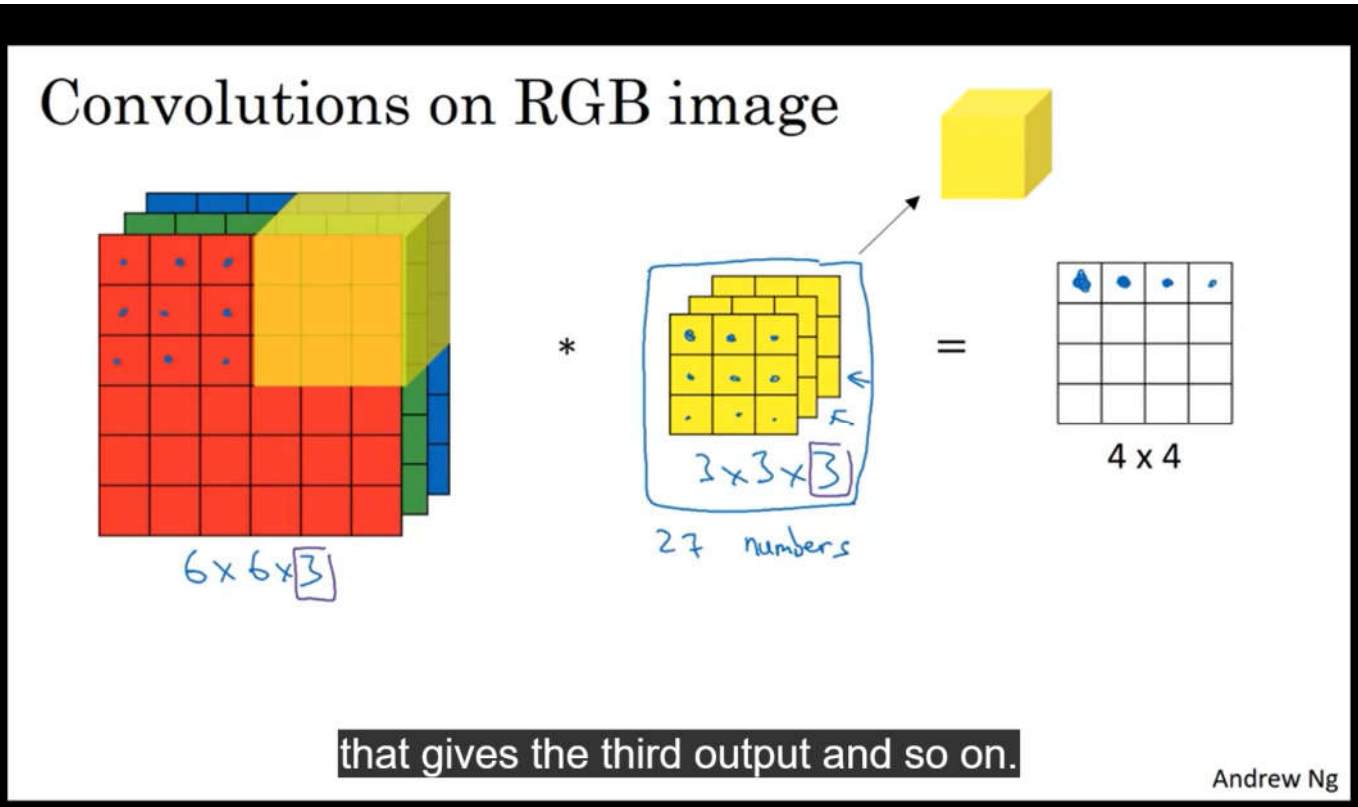

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
these 27 numbers and multiply them with the corresponding numbers from the red,
Andrew Ng

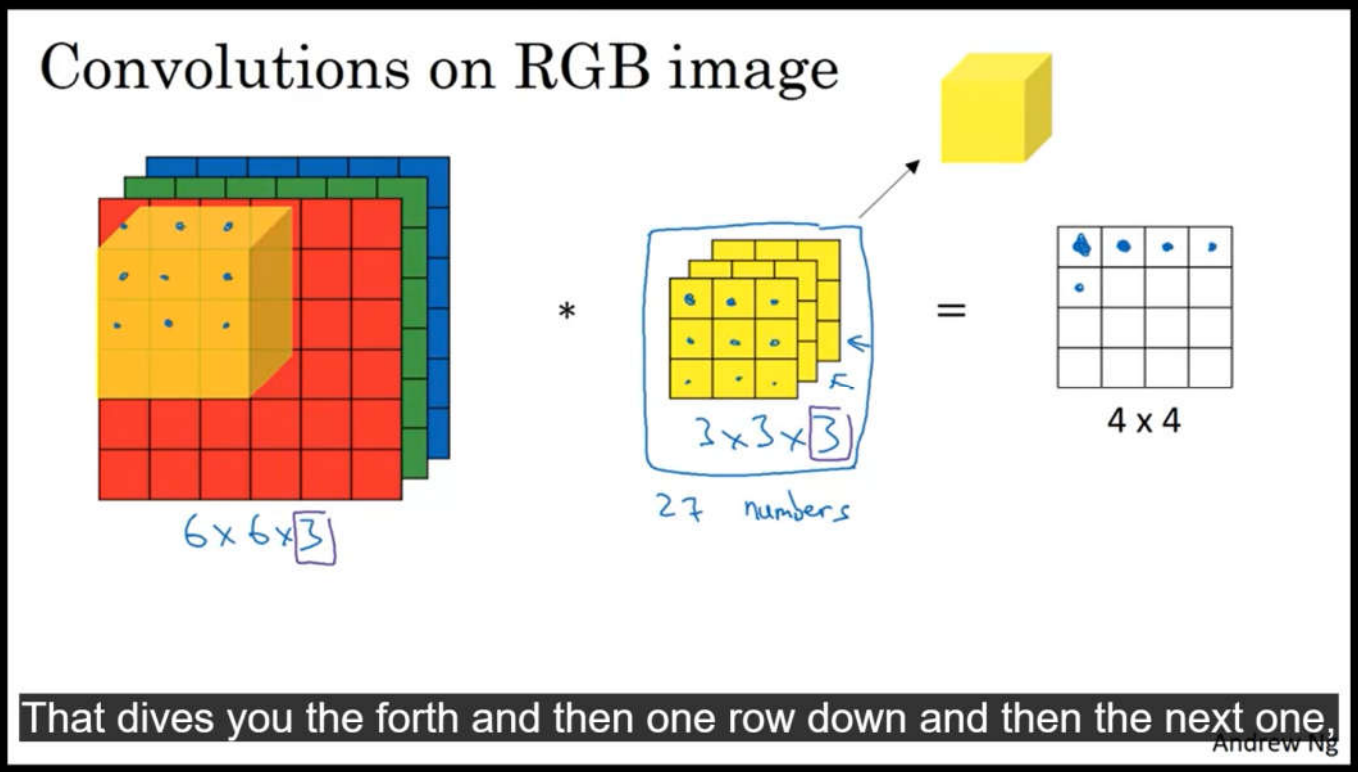

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
green, and blue channels of the image,
Andrew Ng

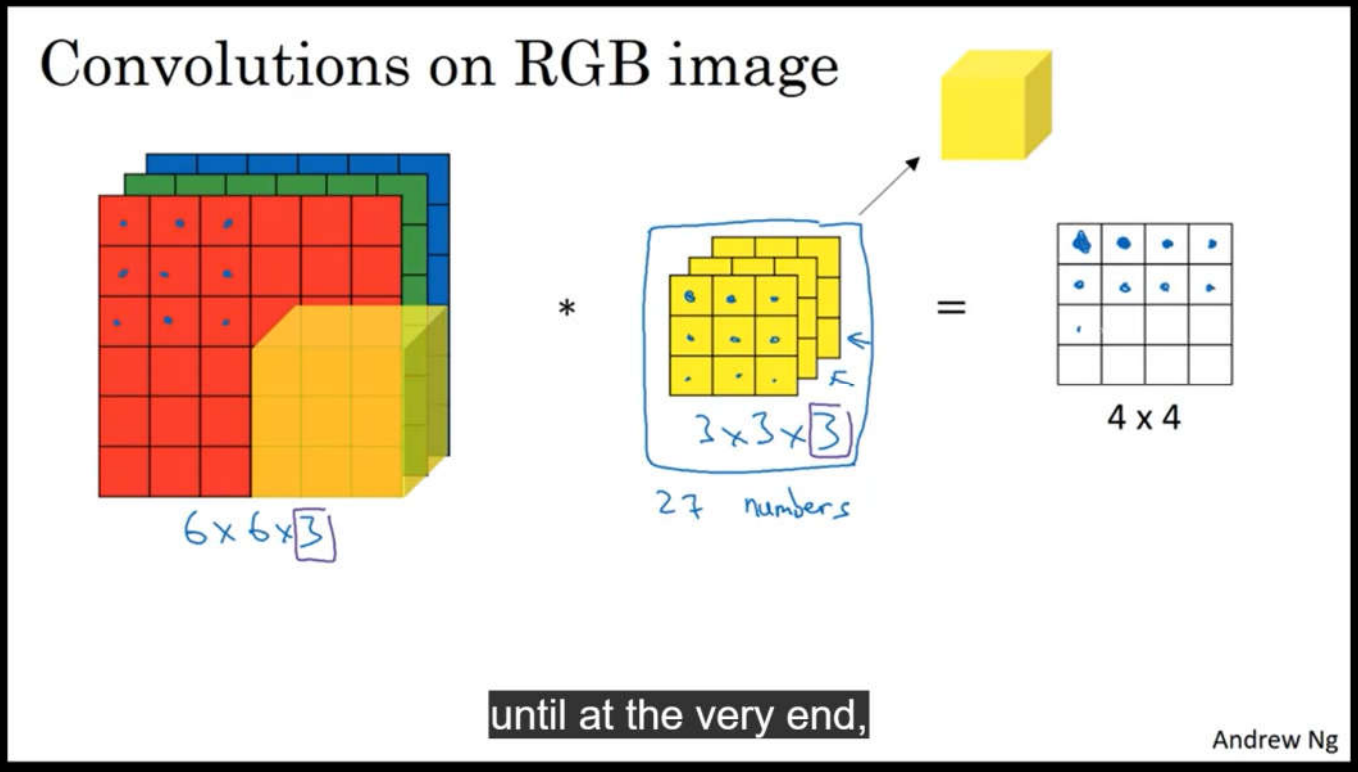

Convolutions on RGB image
*
=
4 x 4
6 x 6 x 3
3 x 3 x 3
27 numbers
Then add up all those numbers and this gives you this first number in the output,
Andrew Ng

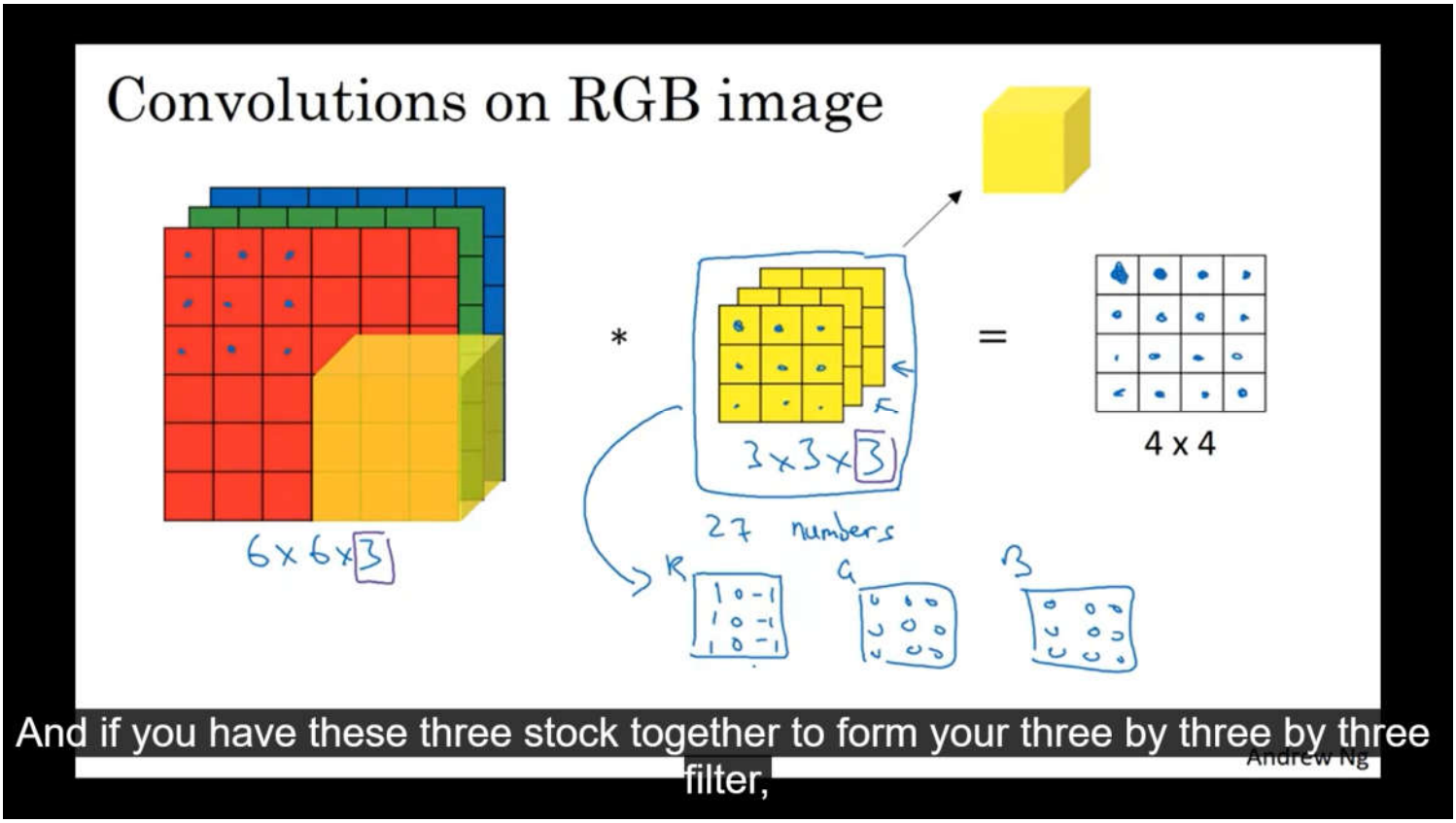

Convolutions on RGB image
*
=
4 x 4
6 x 6 x 3
3 x 3 x 3
27 numbers
and then to compute the next output you take this cube and slide it over by one,
Andrew Ng

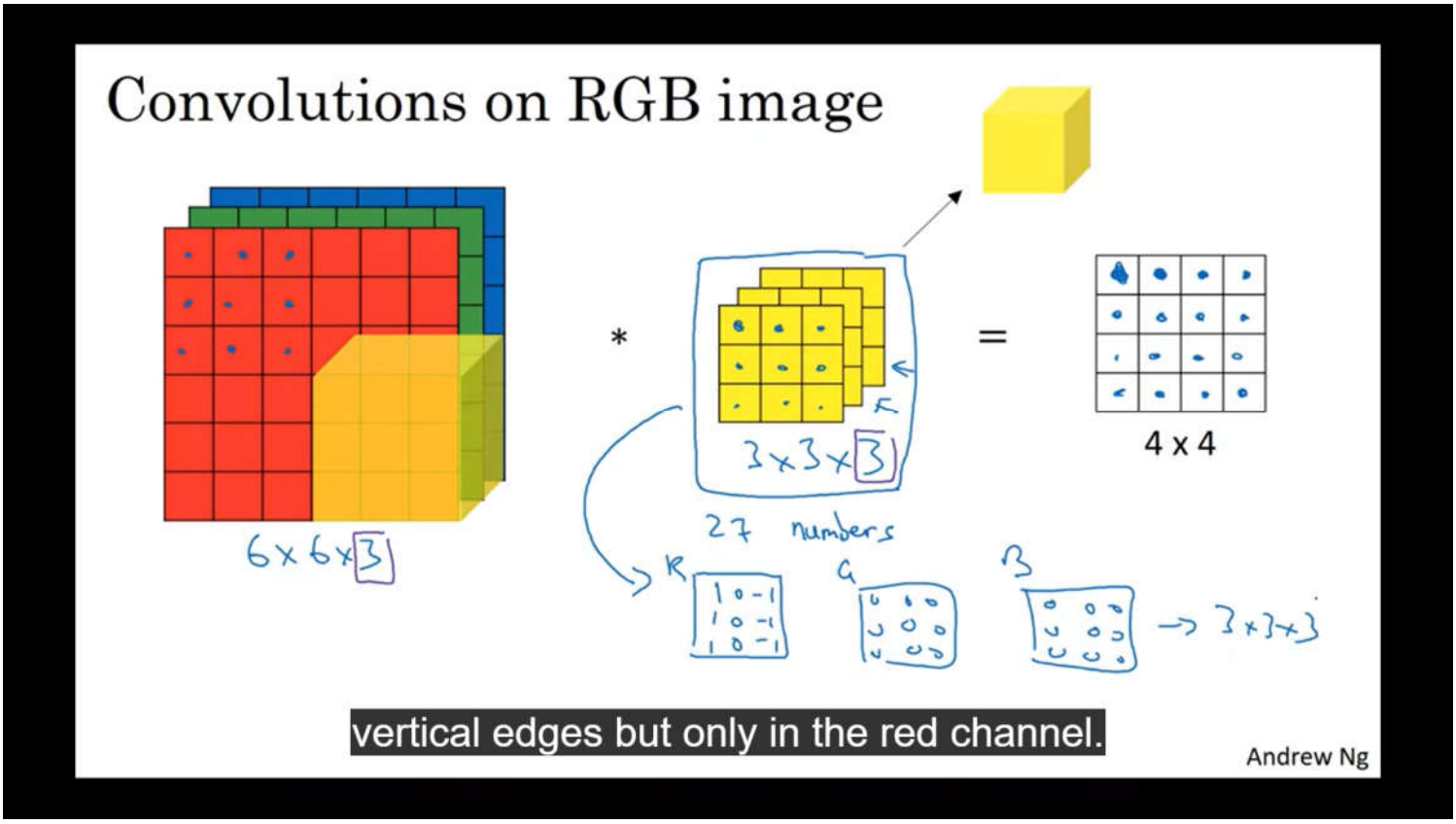

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
do it for the next number over,
Andrew Ng

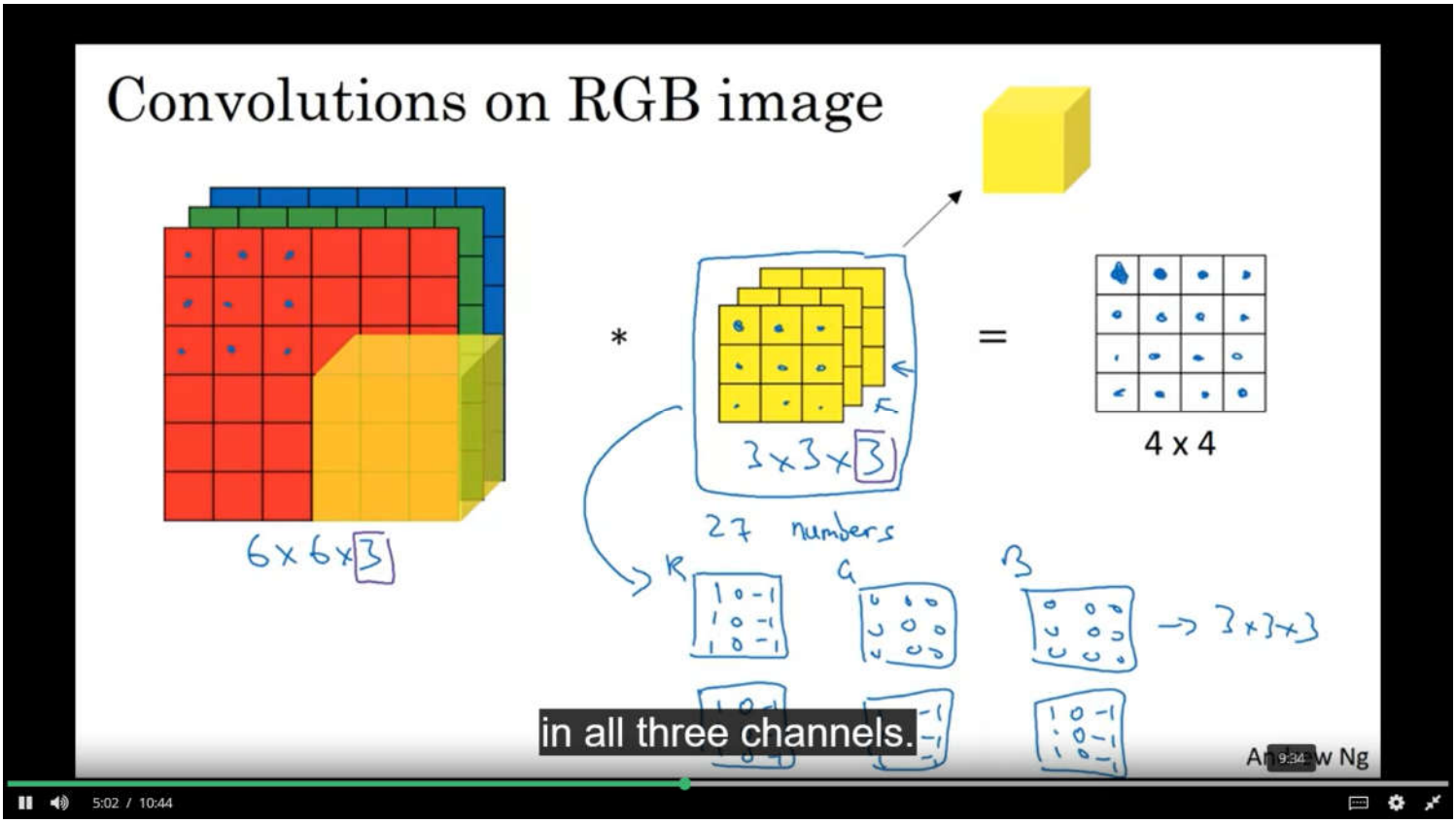

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
that gives the third output and so on.
Andrew Ng

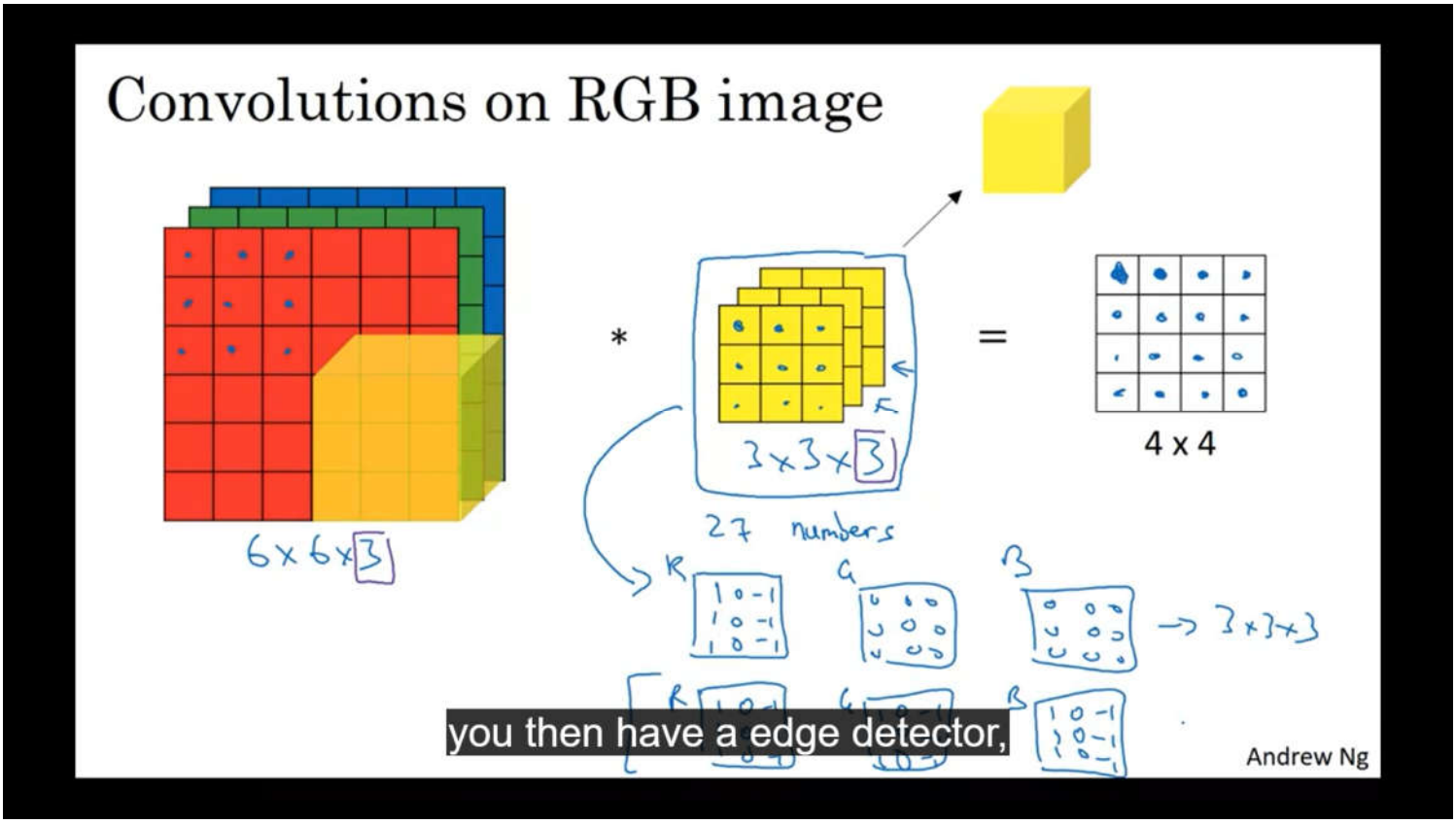

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
That dives you the forth and then one row down and then the next one,
Andrew Ng

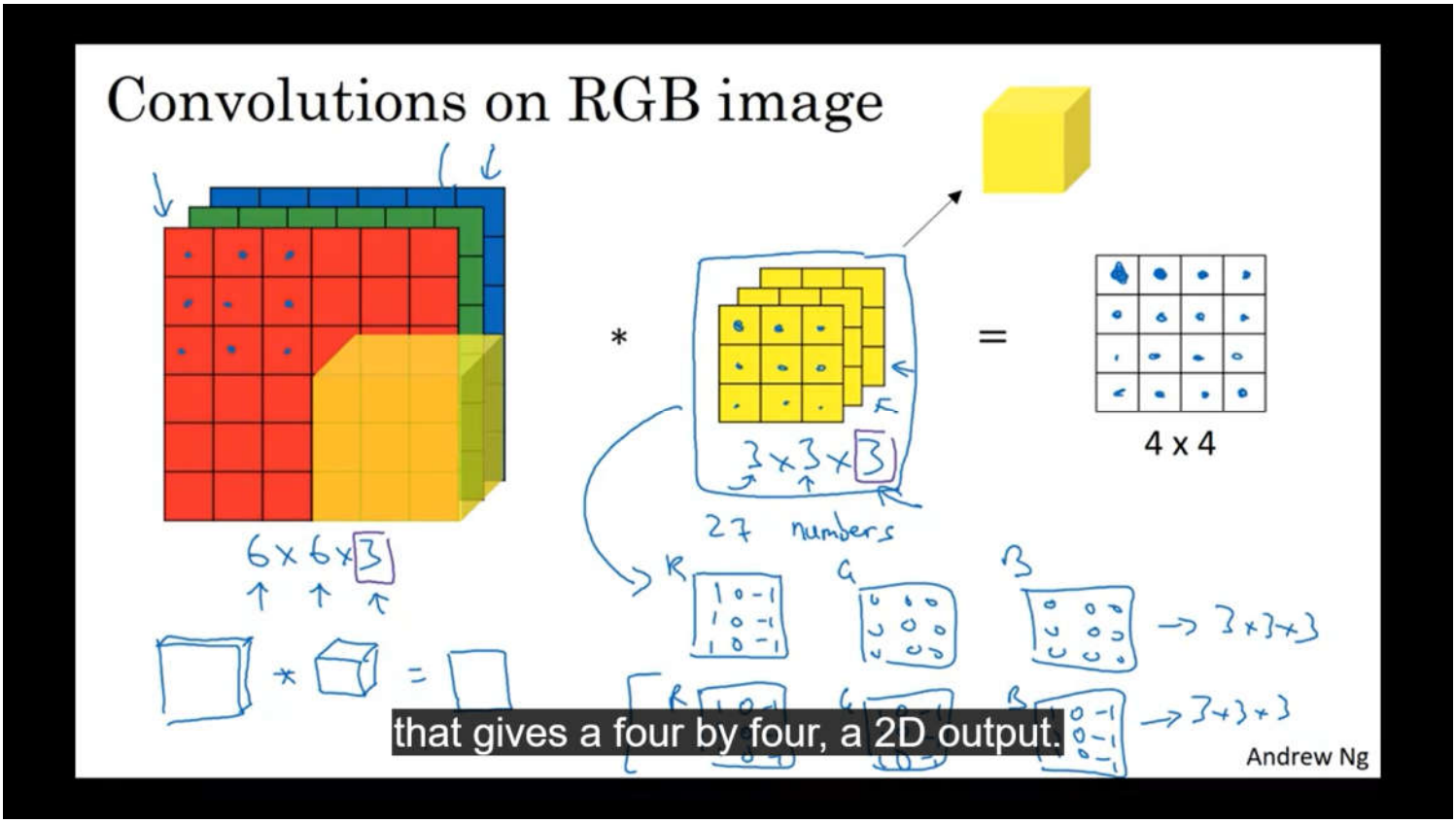

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
until at the very end,
Andrew Ng

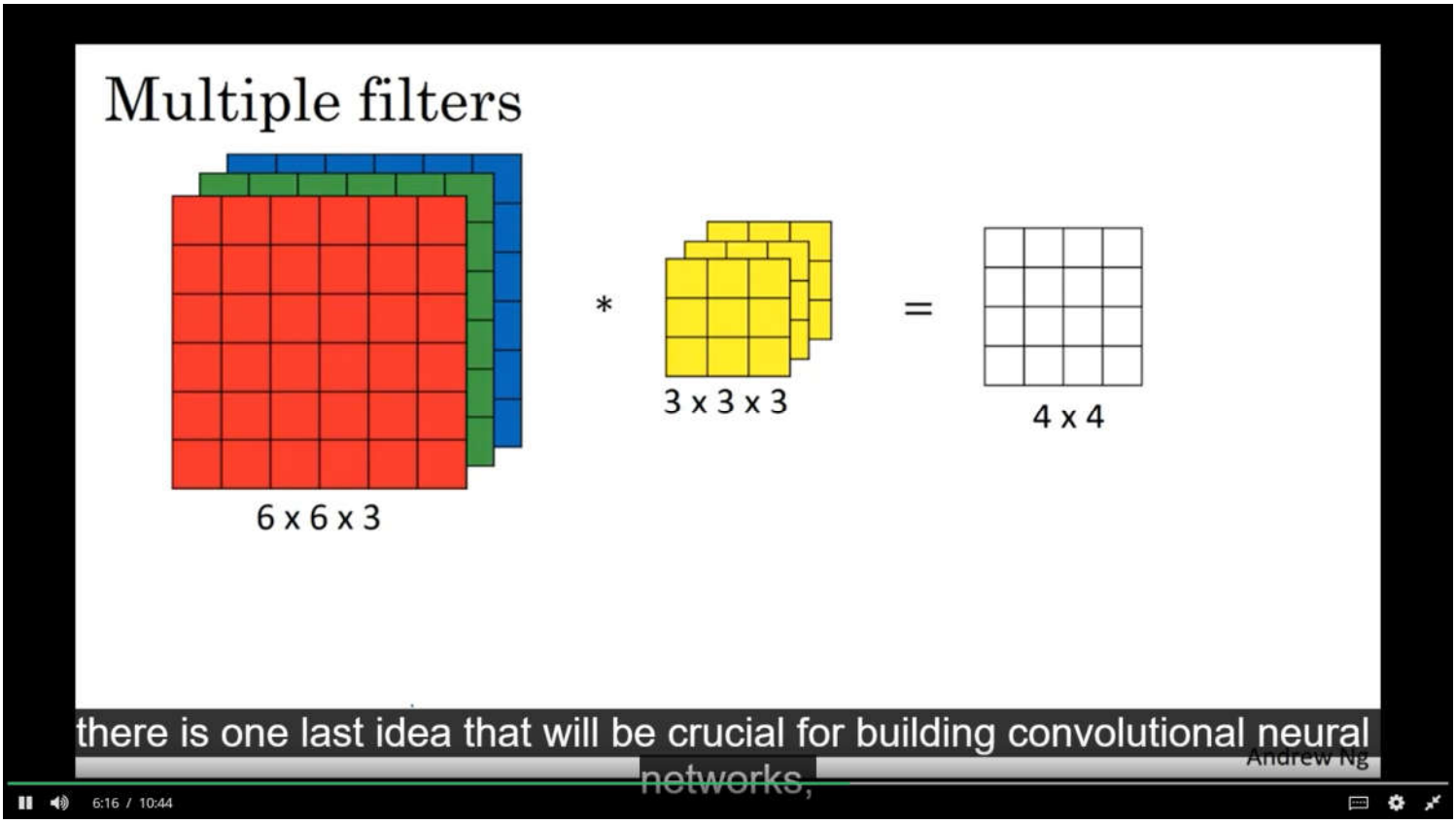

Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
R
G
B
And if you have these three stock together to form your three by three by three filter,
Andrew Ng


Convolutions on RGB image
6 x 6 x 3
*
3 x 3 x 3
27 numbers
=
4 x 4
R
G
B
→ 3 x 3 x 3
vertical edges but only in the red channel.
Andrew Ng

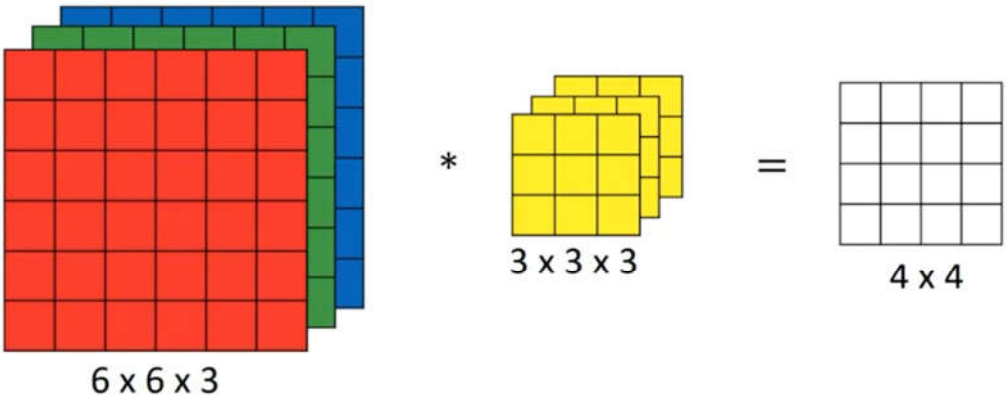

Convolutions on RGB image
*
=
4 x 4
6x6x3
3x3x3
27 numbers
R
G
B
-> 3x3x3
in all three channels.
Andrew Ng
5:02 / 10:44


Convolutions on RGB image
*
=
4 x 4
6x6x3
3x3x3
27 numbers
R
G
B
-> 3x3x3
you then have a edge detector,
Andrew Ng


Convolutions on RGB image
6 x 6 x 3
*
=
4 x 4
3 x 3 x 3
27 numbers
that gives a four by four, a 2D output.
Andrew Ng


Multiple filters
6 x 6 x 3
*
3 x 3 x 3
=
4 x 4
there is one last idea that will be crucial for building convolutional neural networks,
Andrew Ng
6:16 / 10:44

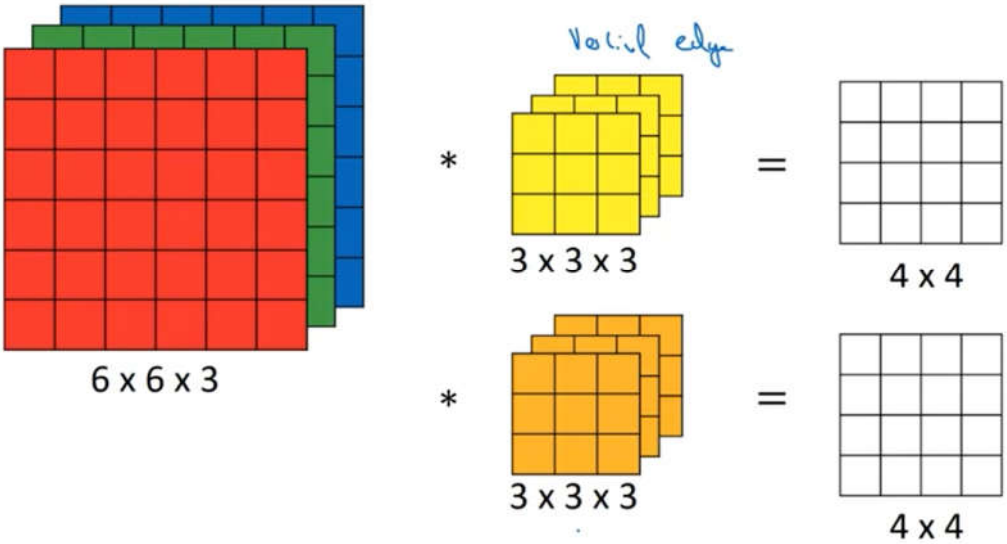
Multiple filters



What if we wanted to detect vertical edges and horizontal edges

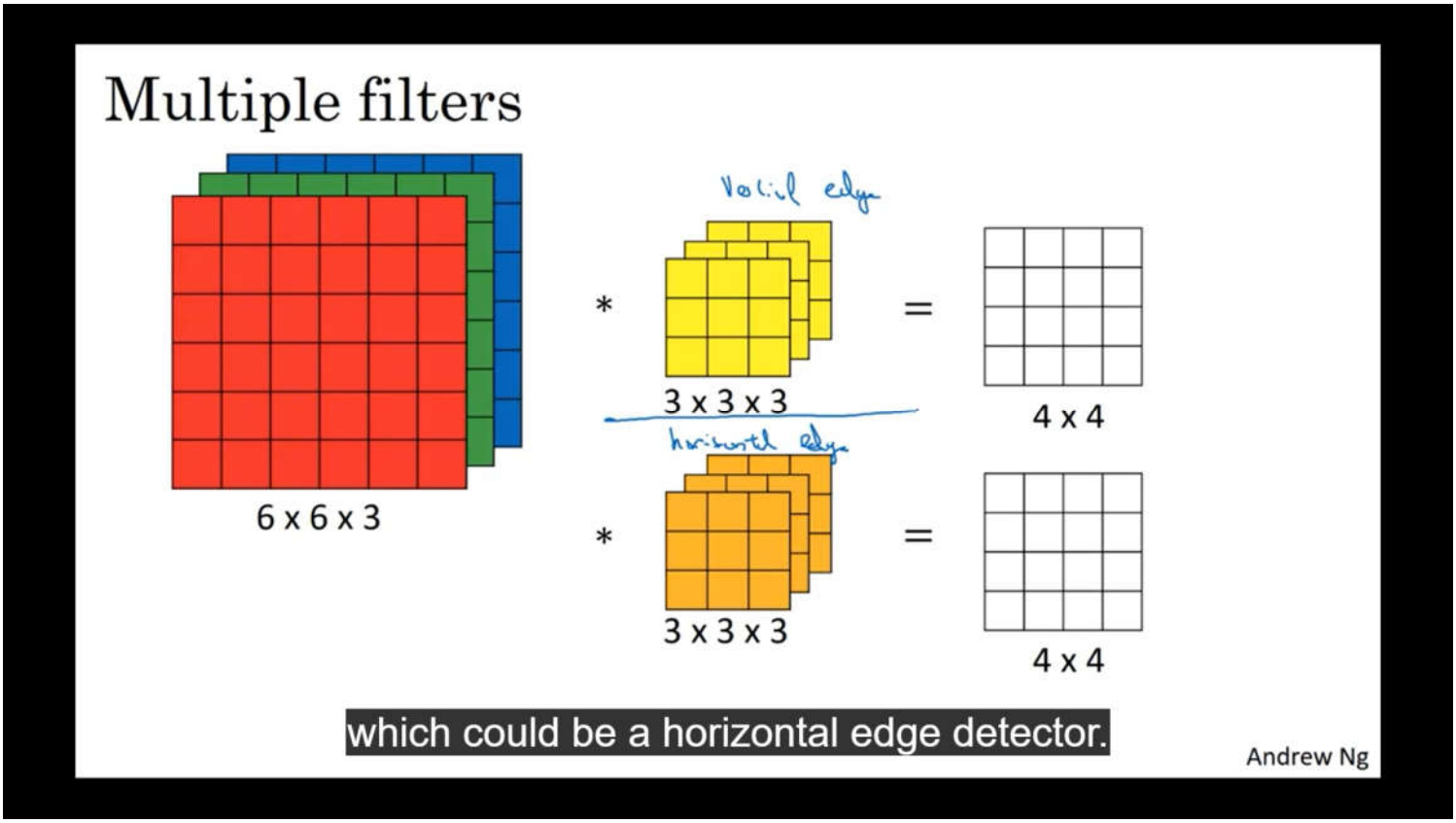
Andrew Ng

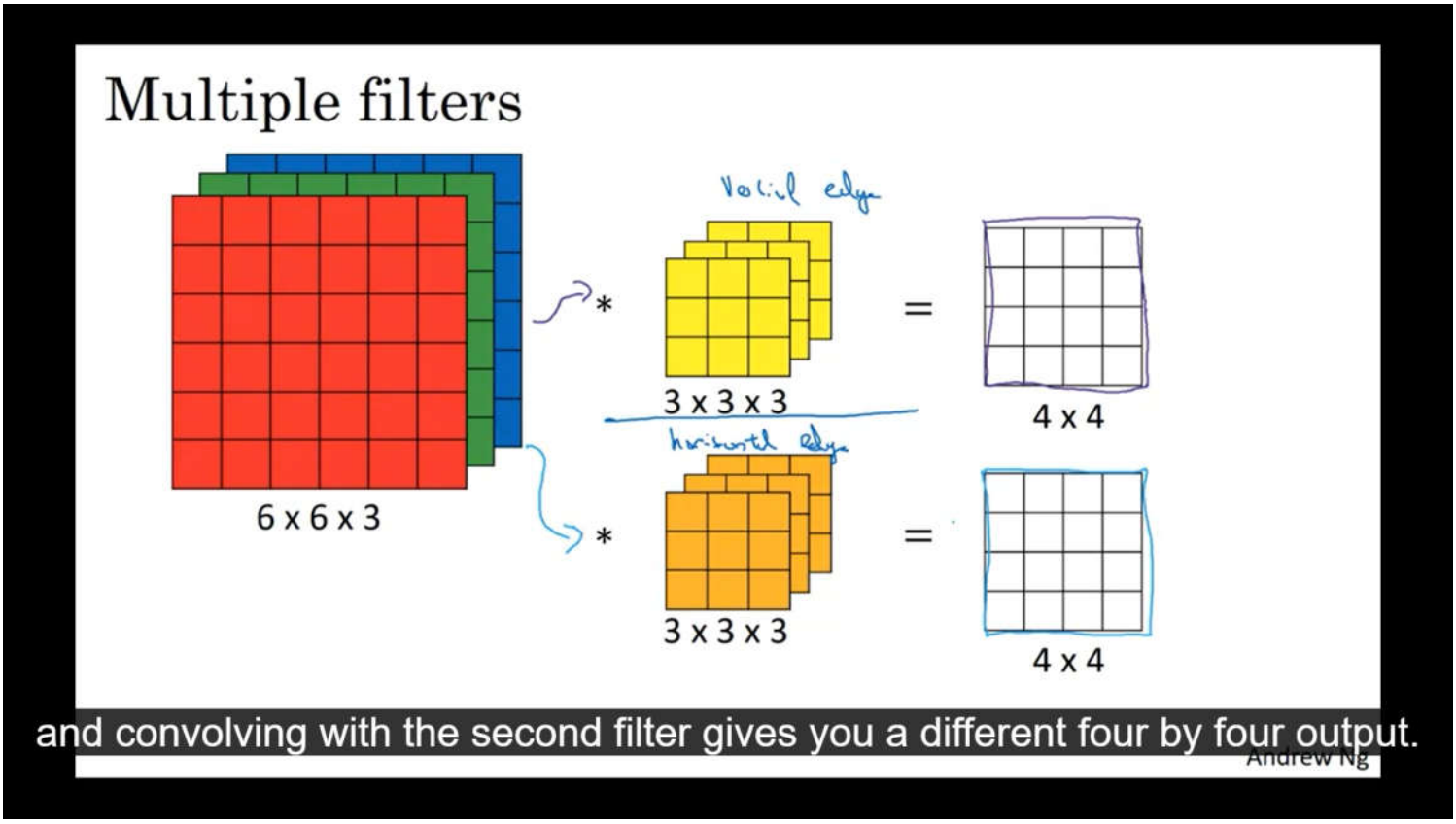
Multiple filters

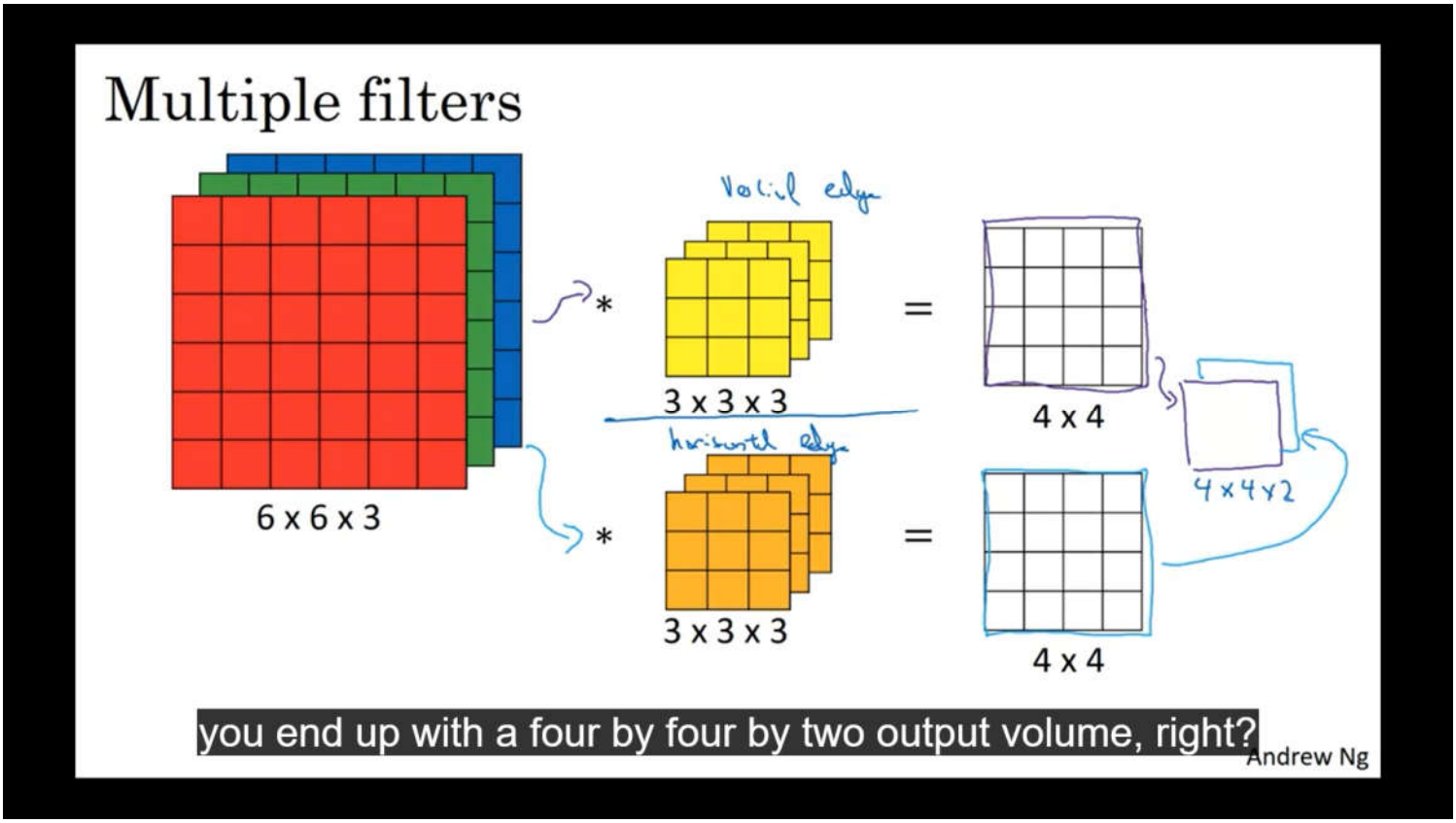


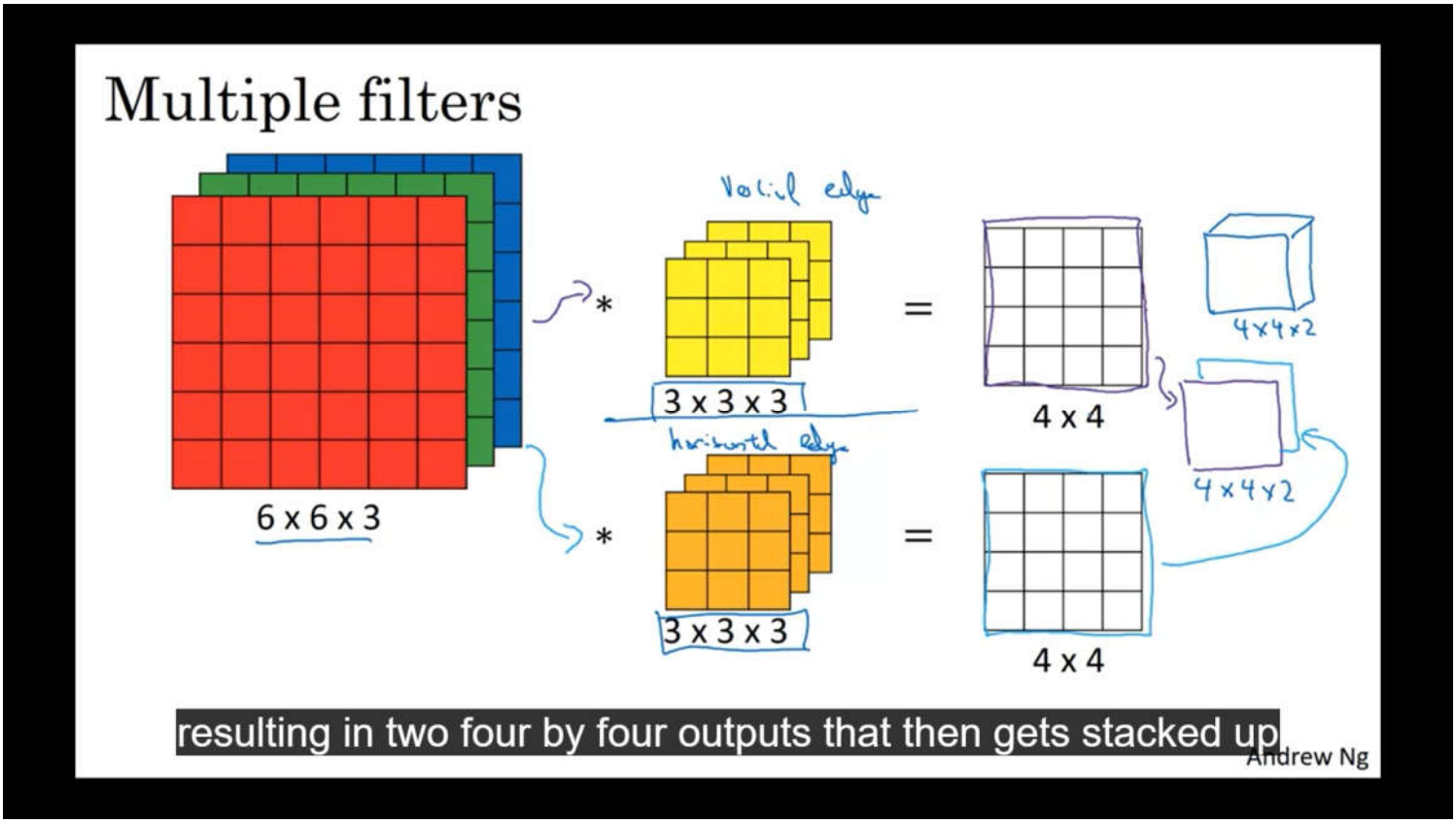
Now, maybe a second filter may be denoted by this orange-ish color,

Andrew Ng

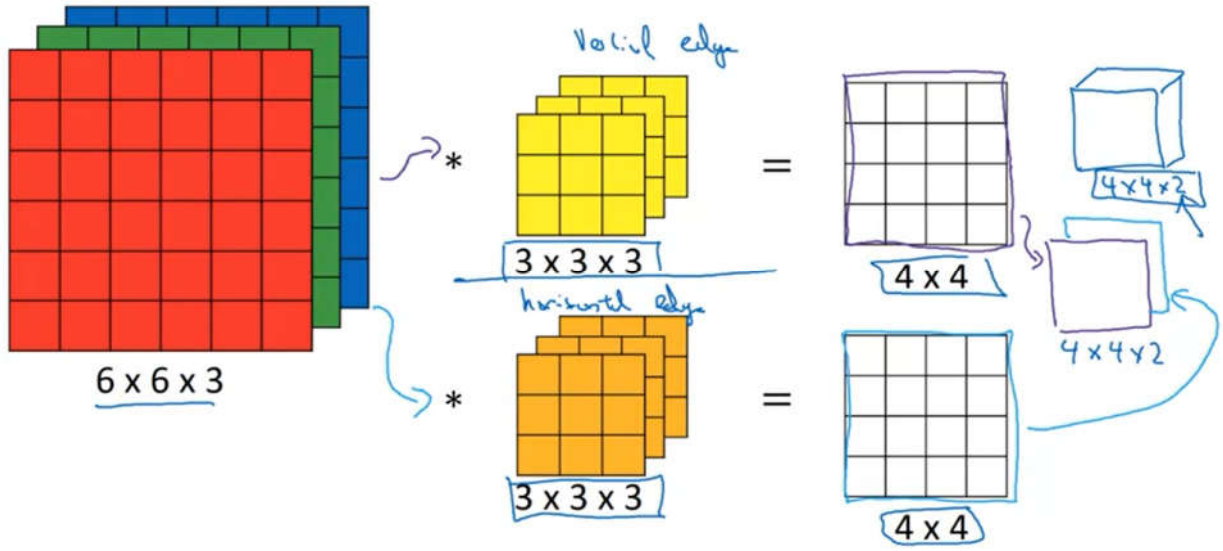

Multiple filters
6 x 6 x 3
*
3 x 3 x 3
=
4 x 4
*
3 x 3 x 3
=
4 x 4
which could be a horizontal edge detector.
Andrew Ng


Multiple filters
6 x 6 x 3
*
3 x 3 x 3
=
4 x 4
*
3 x 3 x 3
=
4 x 4
and convolving with the second filter gives you a different four by four output.
Andrew Ng


Multiple filters
6 x 6 x 3
*
3 x 3 x 3
=
4 x 4
4 x 4 x 2
*
3 x 3 x 3
=
4 x 4
you end up with a four by four by two output volume, right?
Andrew Ng


Multiple filters
6 x 6 x 3
*
3 x 3 x 3
=
4 x 4
4 x 4 x 2
4 x 4 x 2
*
3 x 3 x 3
=
4 x 4
resulting in two four by four outputs that then gets stacked up
Andrew Ng

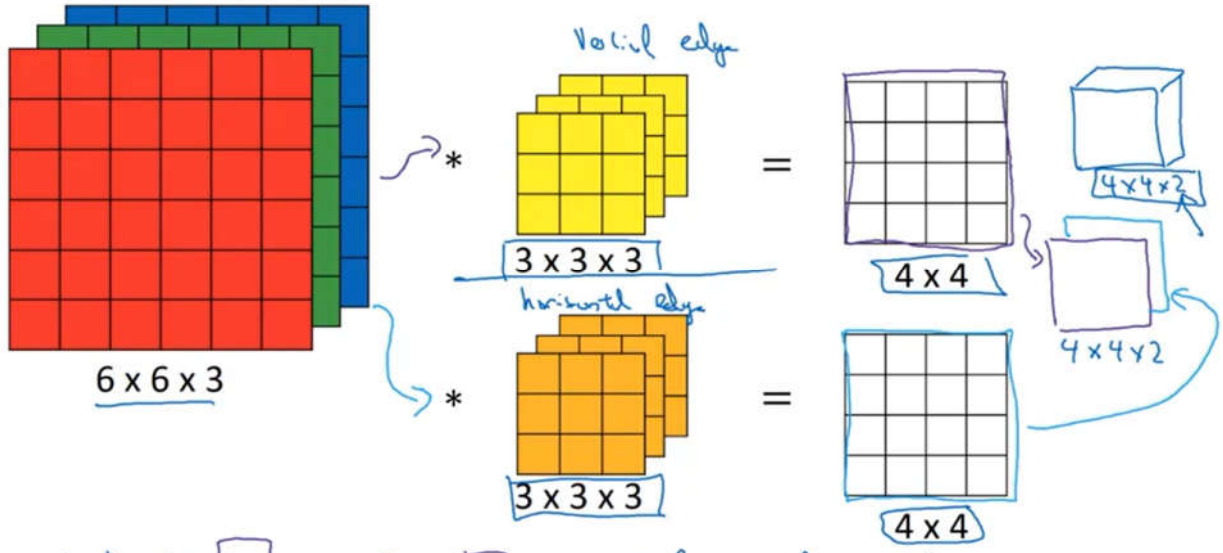
Multiple filters



And the two here comes from the fact that we used two different filters.

Andrew Ng

Multiple filters

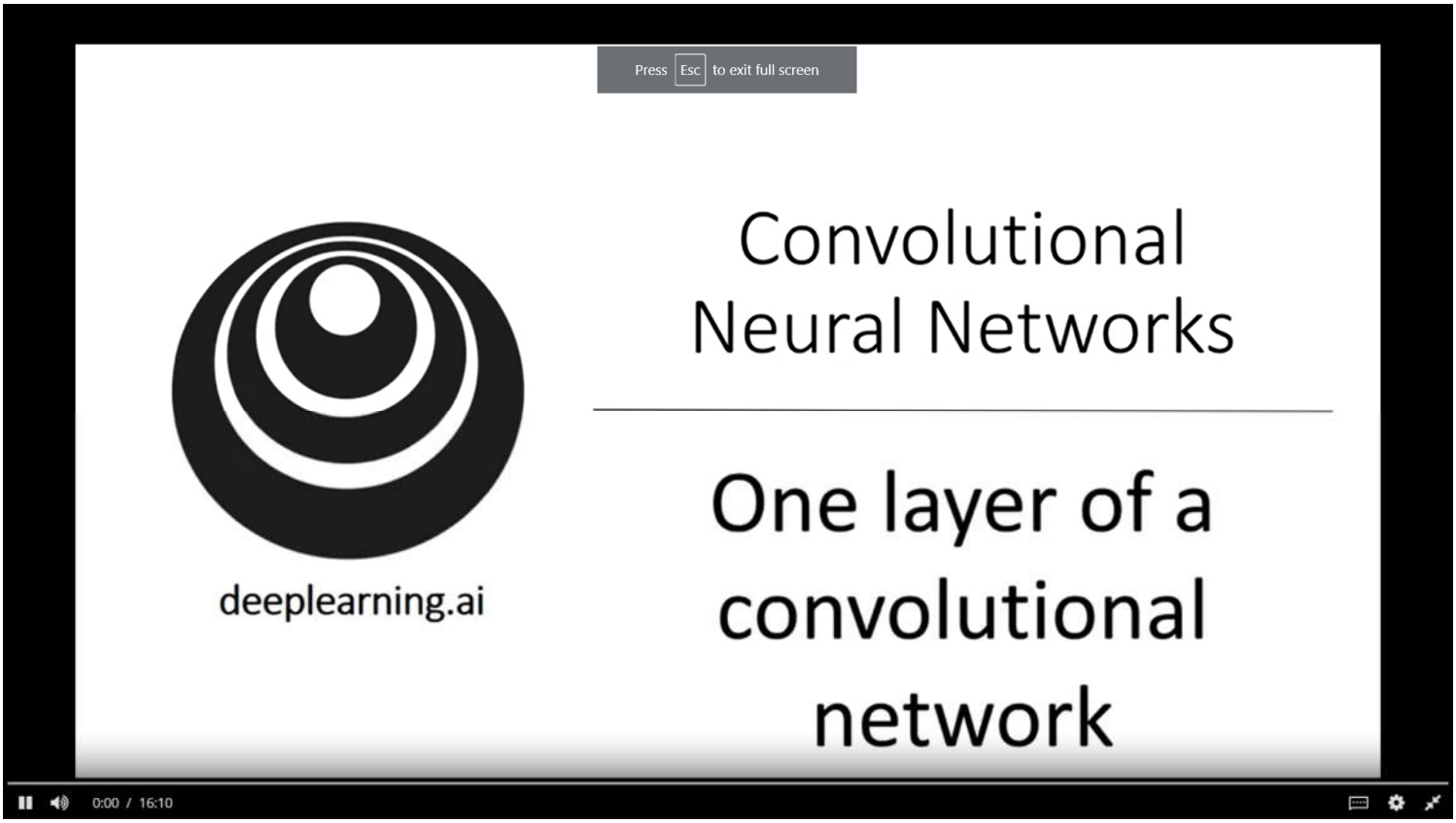


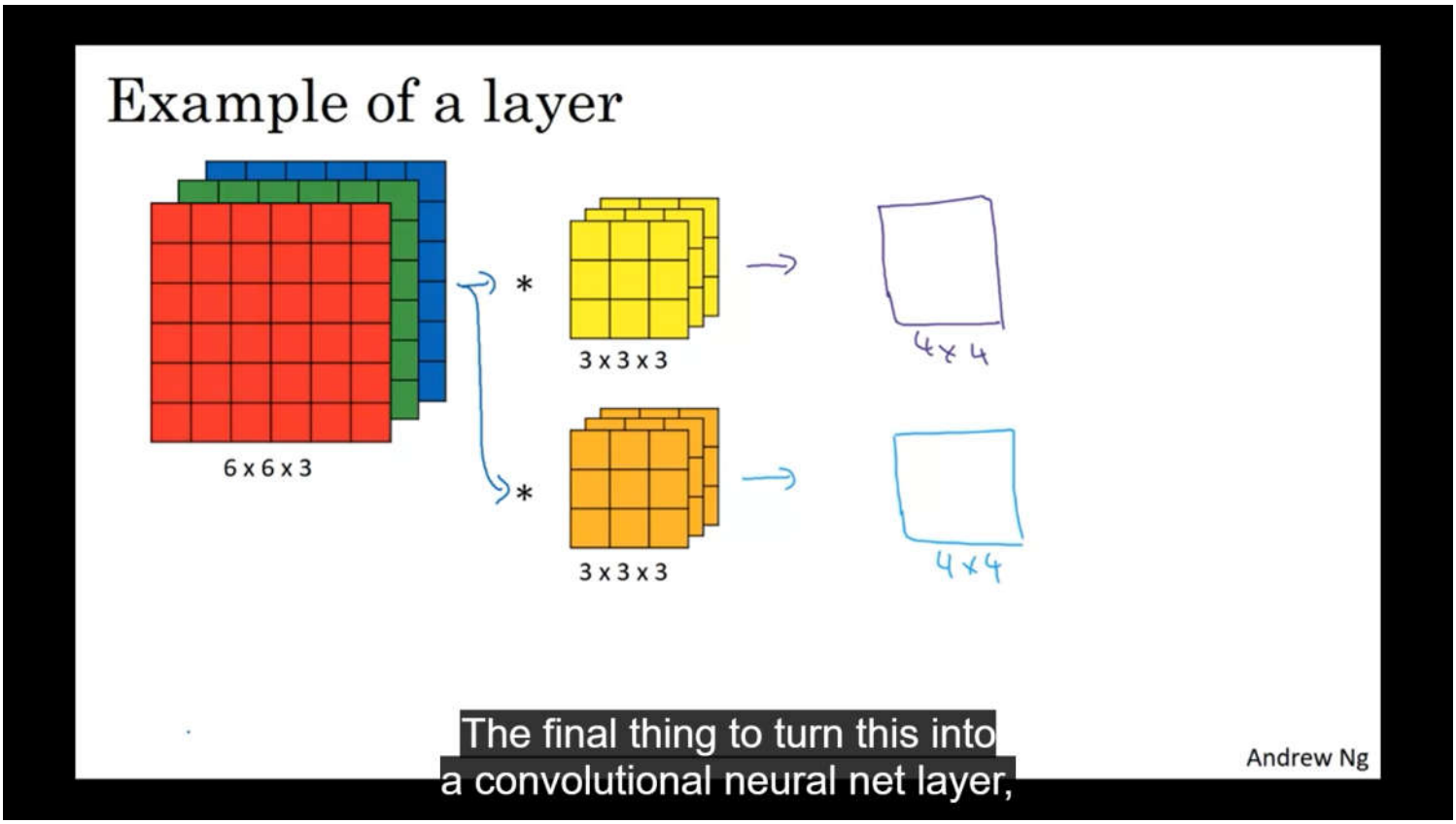
Summary: $n \times n \times \boxed{n_c}$ $\quad * \quad f \times f \times \boxed{n_c} \quad \rightarrow \quad n-f+1 \times n-f+1 \times n_c'$

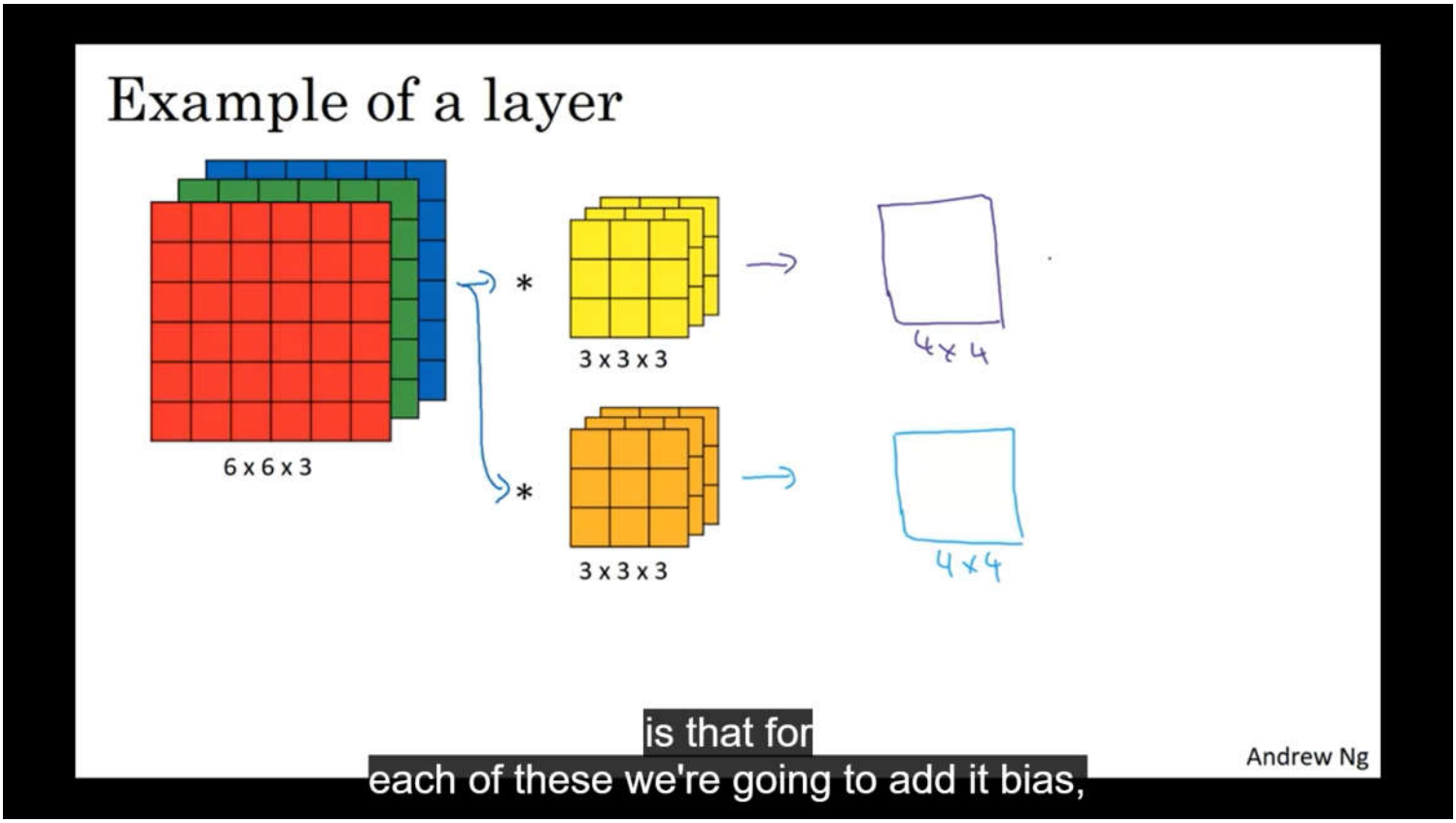
$6 \times 6 \times 3 \qquad 3 \times 3 \times 3$

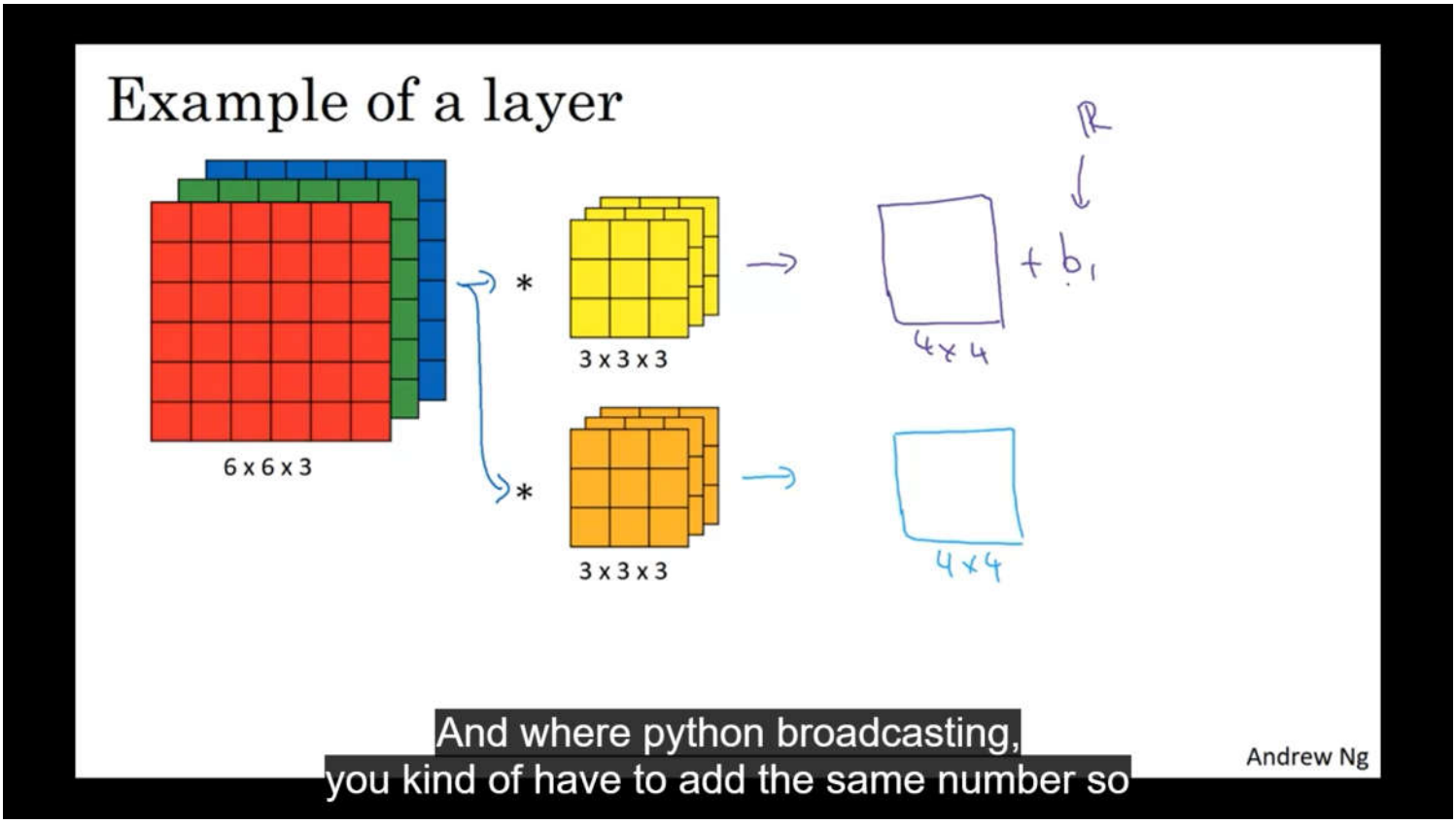
filters

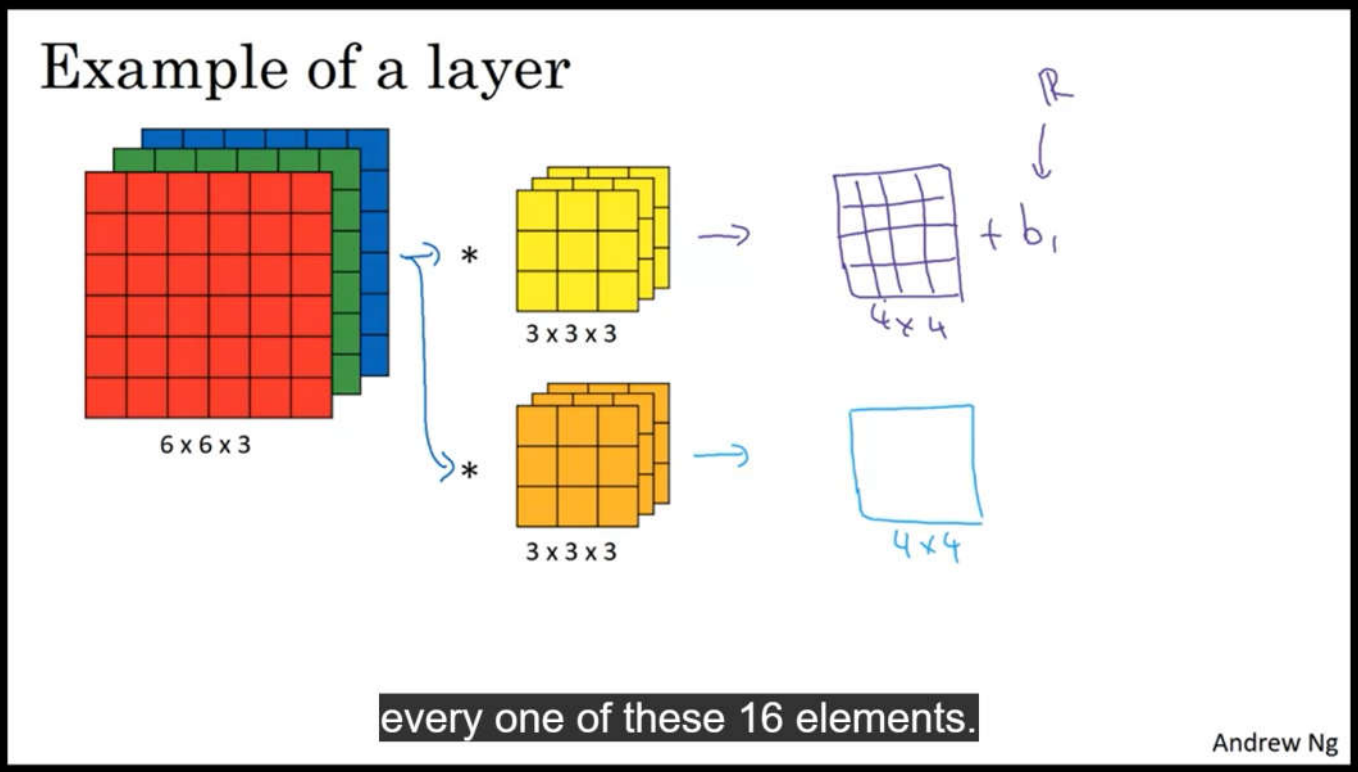
Andrew Ng

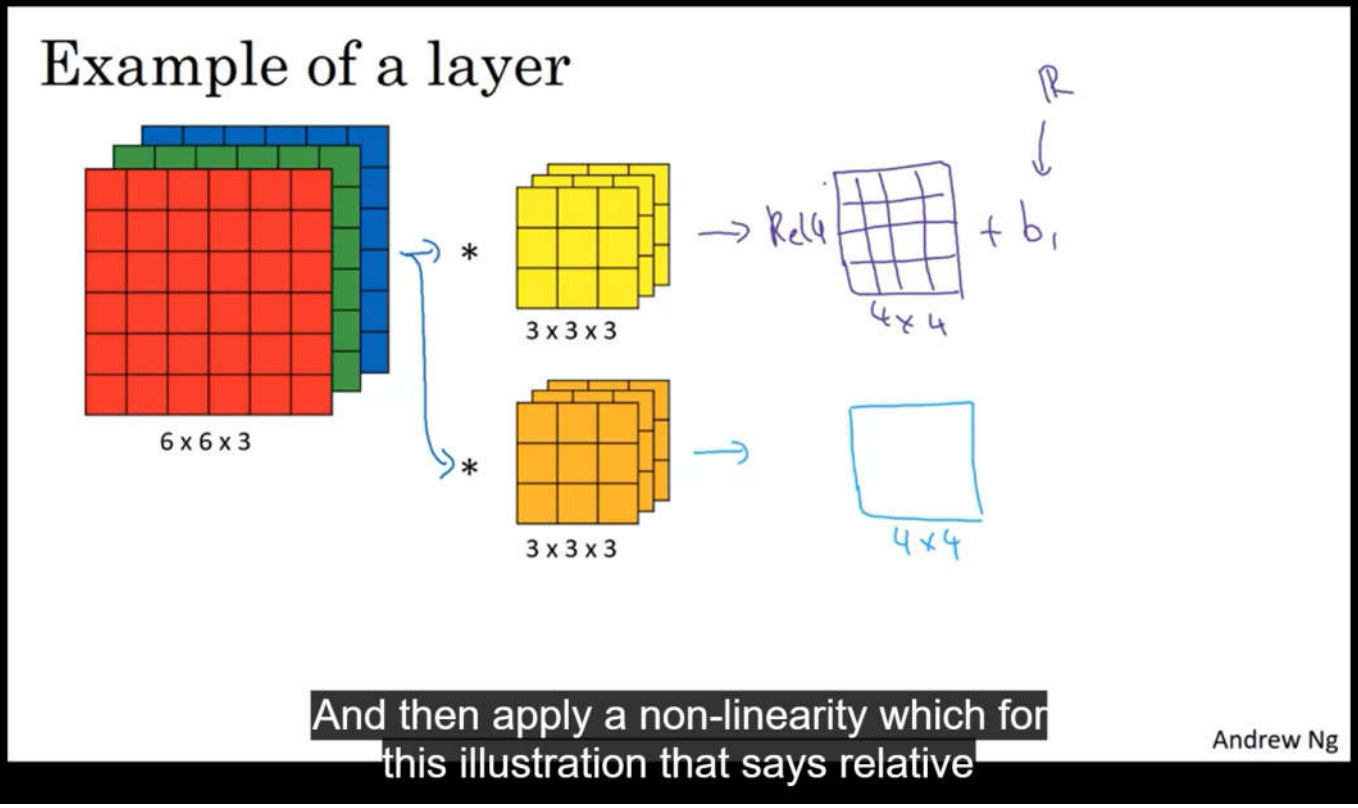

Press Esc to exit full screen
Convolutional Neural Networks
deeplearning.ai
One layer of a convolutional network
0:00 / 16:10

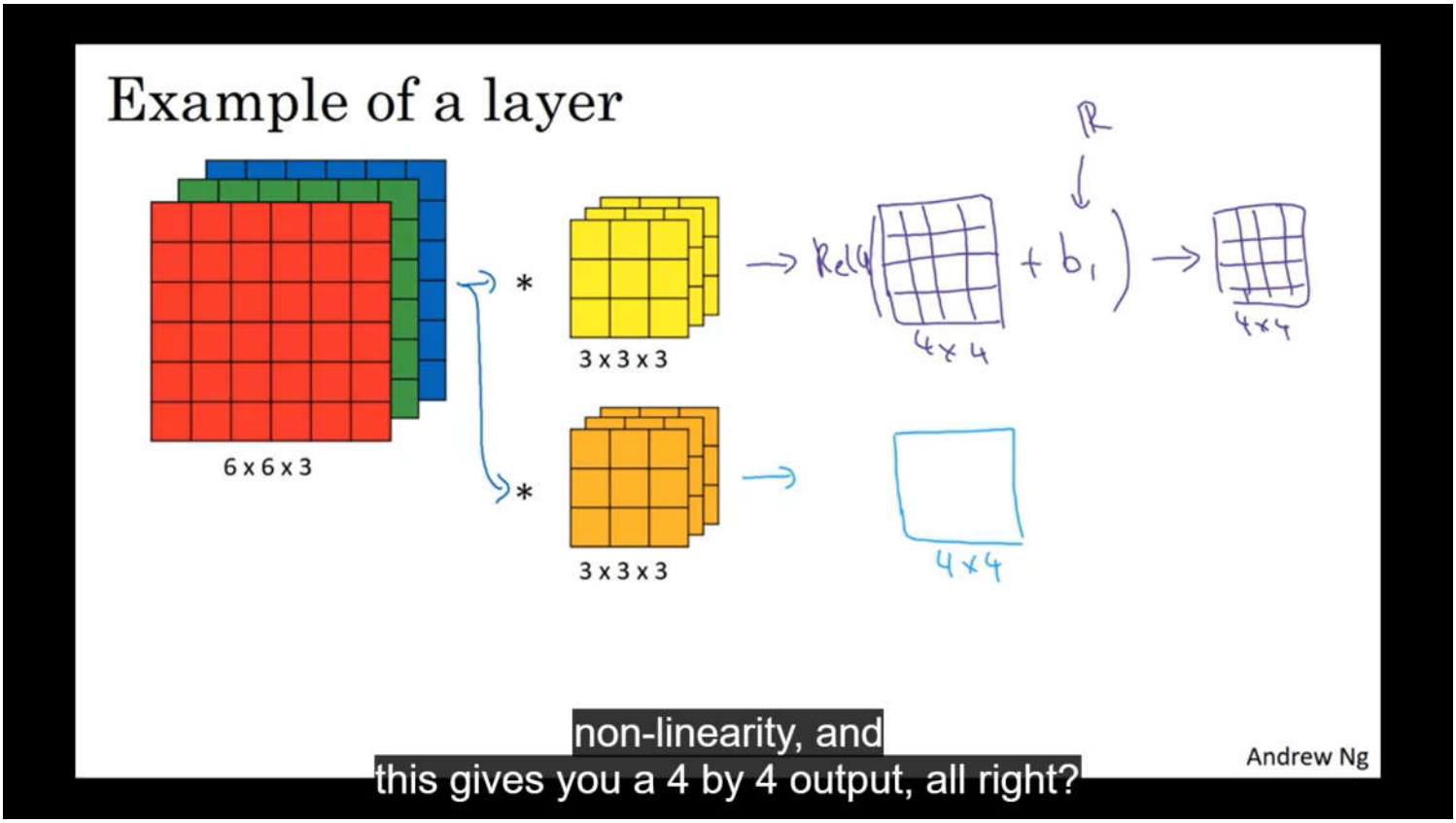

Example of a layer
6 x 6 x 3
*
3 x 3 x 3
4x4
*
3 x 3 x 3
4x4
The final thing to turn this into
a convolutional neural net layer,
Andrew Ng


Example of a layer
6 x 6 x 3
*
3 x 3 x 3
4x4
*
3 x 3 x 3
4x4
is that for
each of these we're going to add it bias,
Andrew Ng

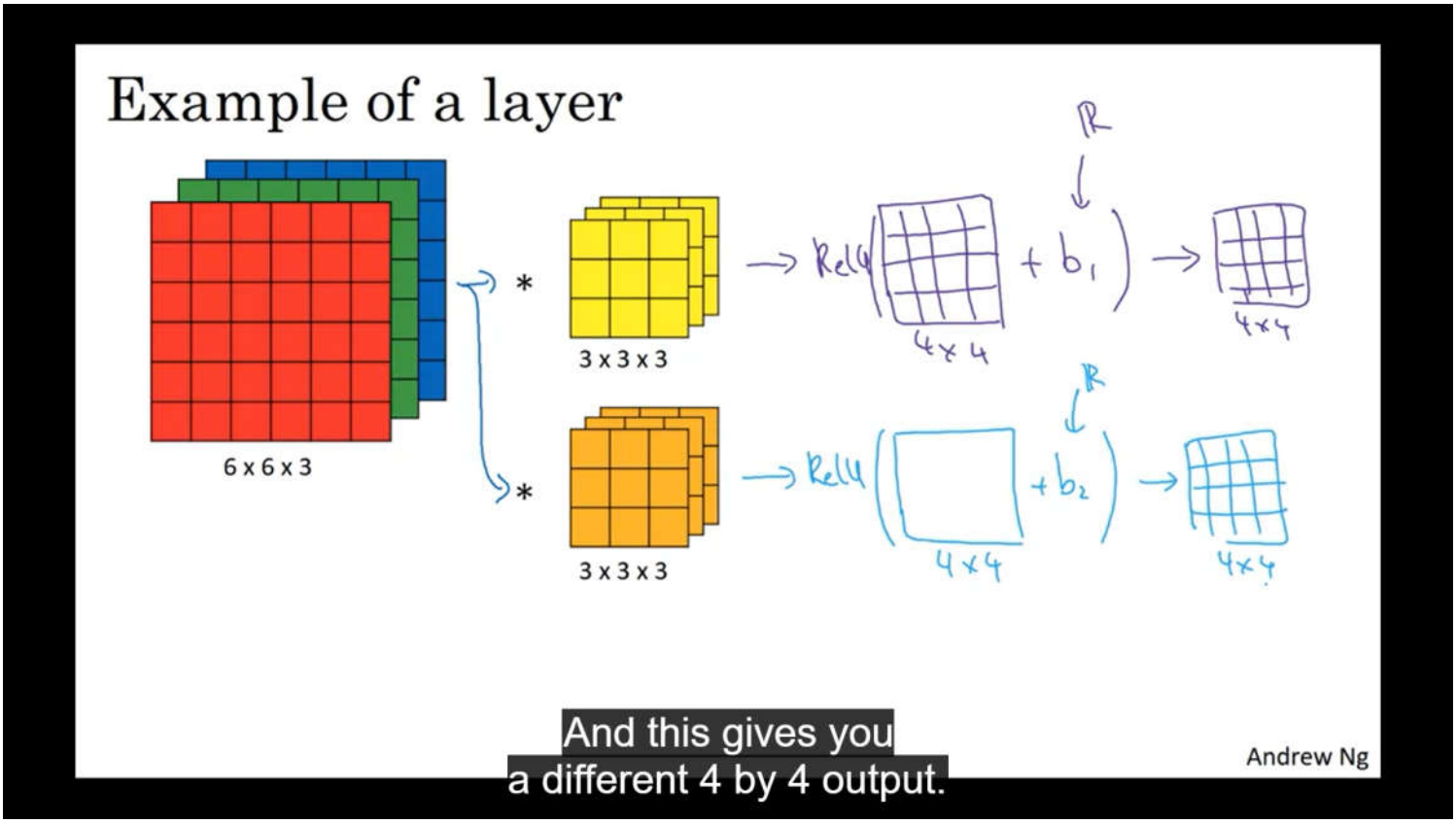

Example of a layer
6 x 6 x 3
*
3 x 3 x 3
4x4
+ b1
*
3 x 3 x 3
4x4
And where python broadcasting,
you kind of have to add the same number so
Andrew Ng


Example of a layer
6 x 6 x 3
*
3 x 3 x 3
*
3 x 3 x 3
4x4
+ b1
4x4
every one of these 16 elements.
Andrew Ng

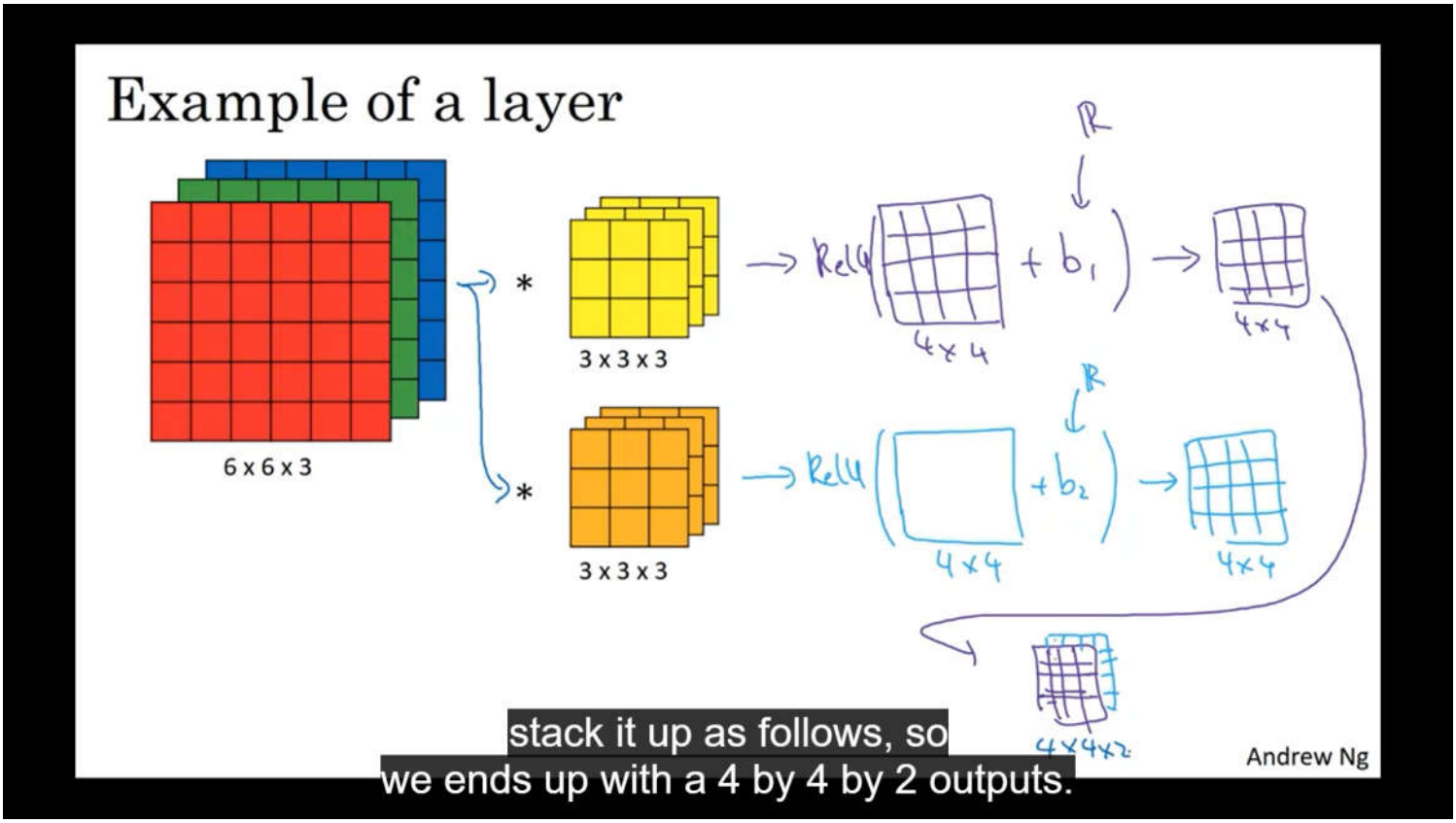

Example of a layer
6 x 6 x 3
*
3 x 3 x 3
*
3 x 3 x 3
Relu
4x4
+ b1
4x4
And then apply a non-linearity which for
this illustration that says relative
Andrew Ng



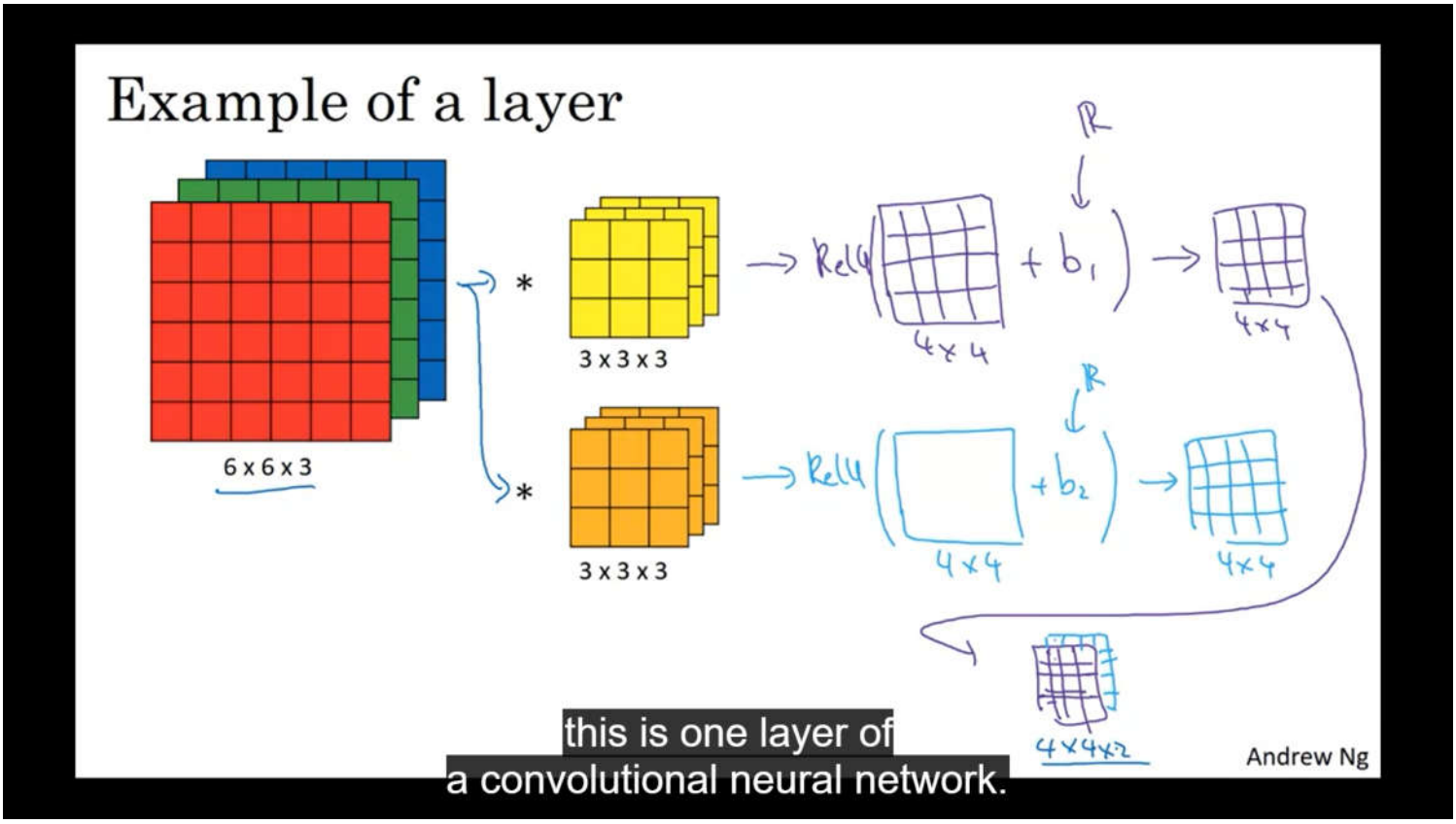
non-linearity, and
this gives you a 4 by 4 output, all right?



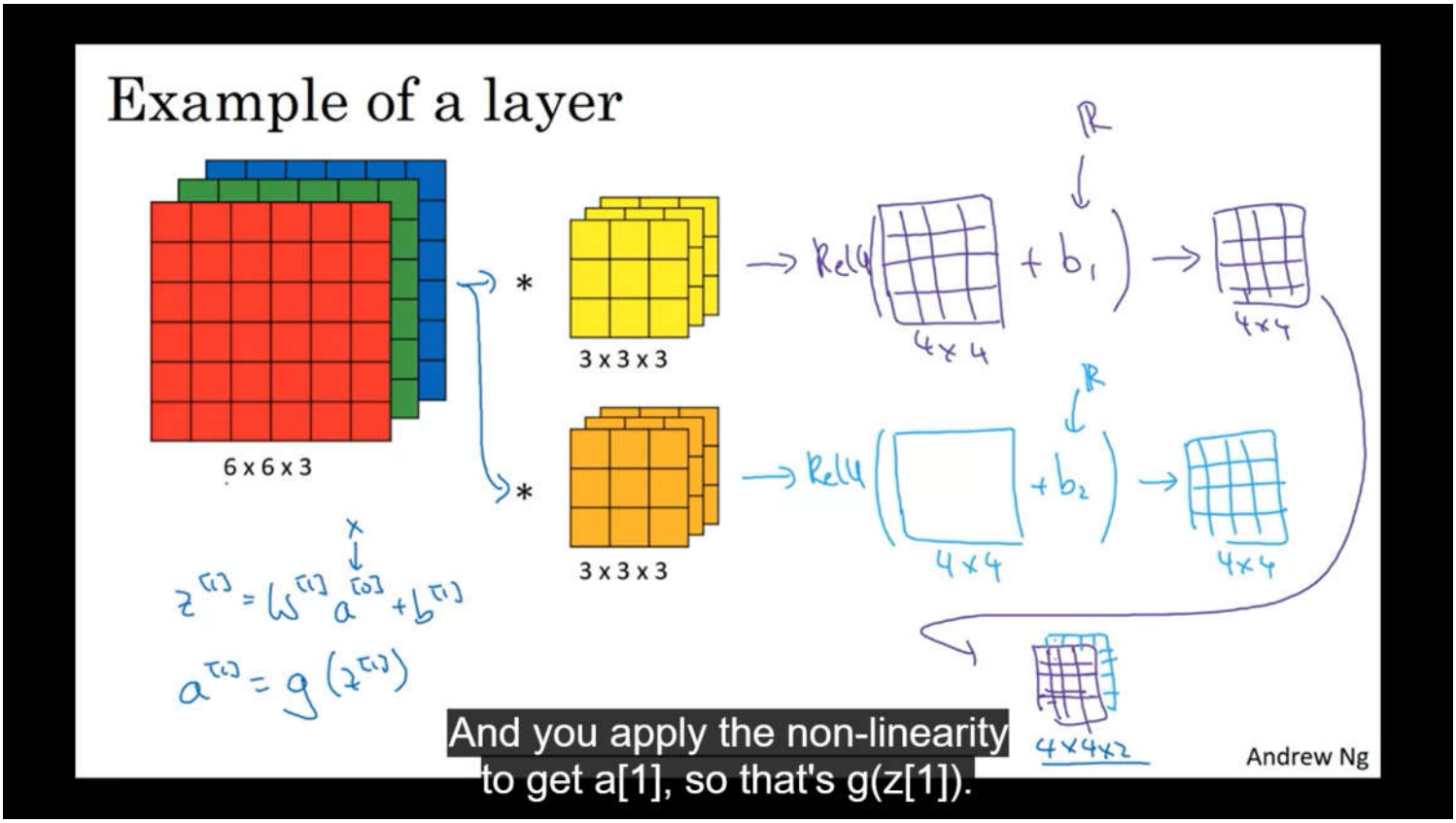
And this gives you
a different 4 by 4 output.

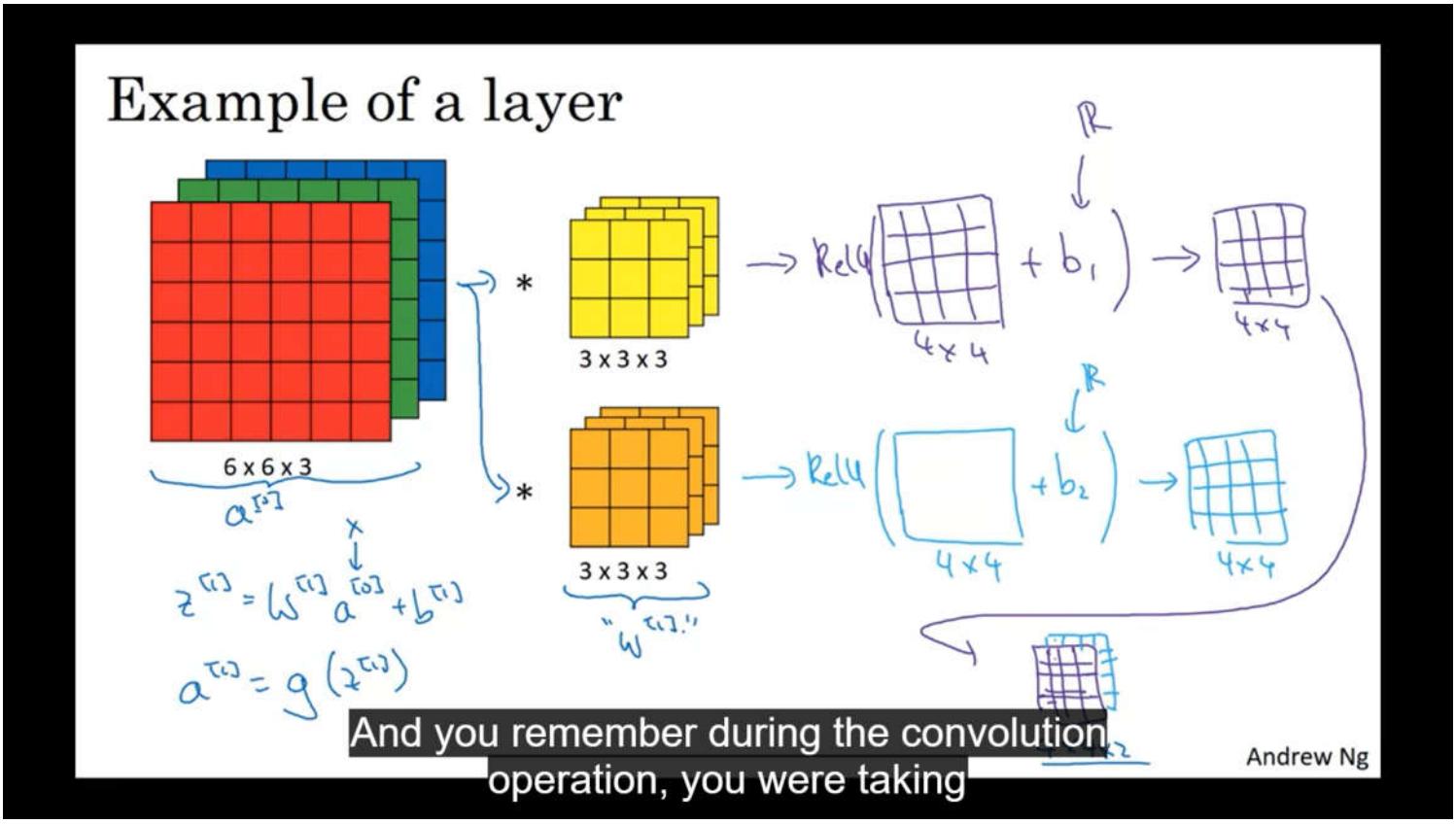


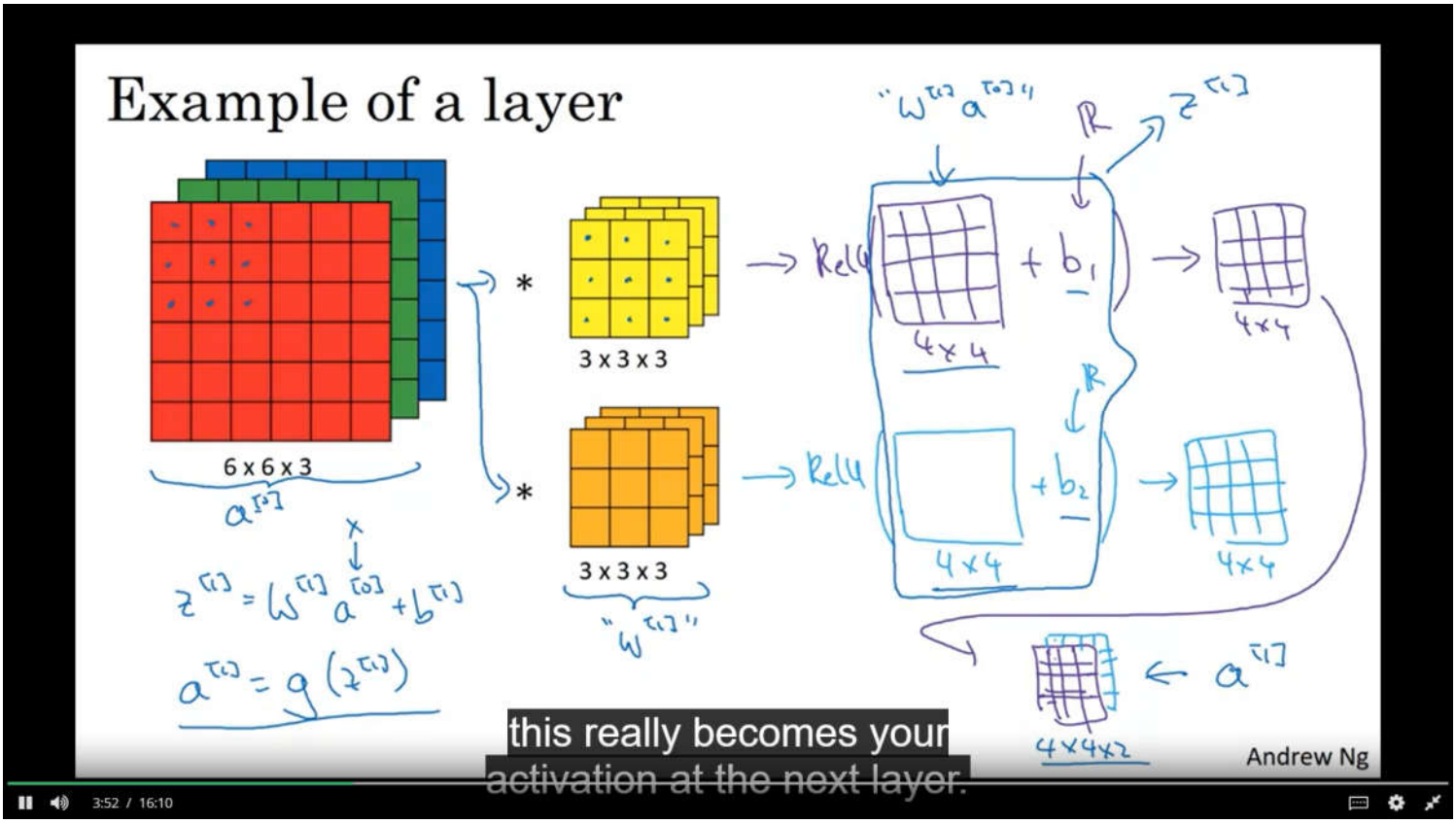
stack it up as follows, so
we ends up with a 4 by 4 by 2 outputs.

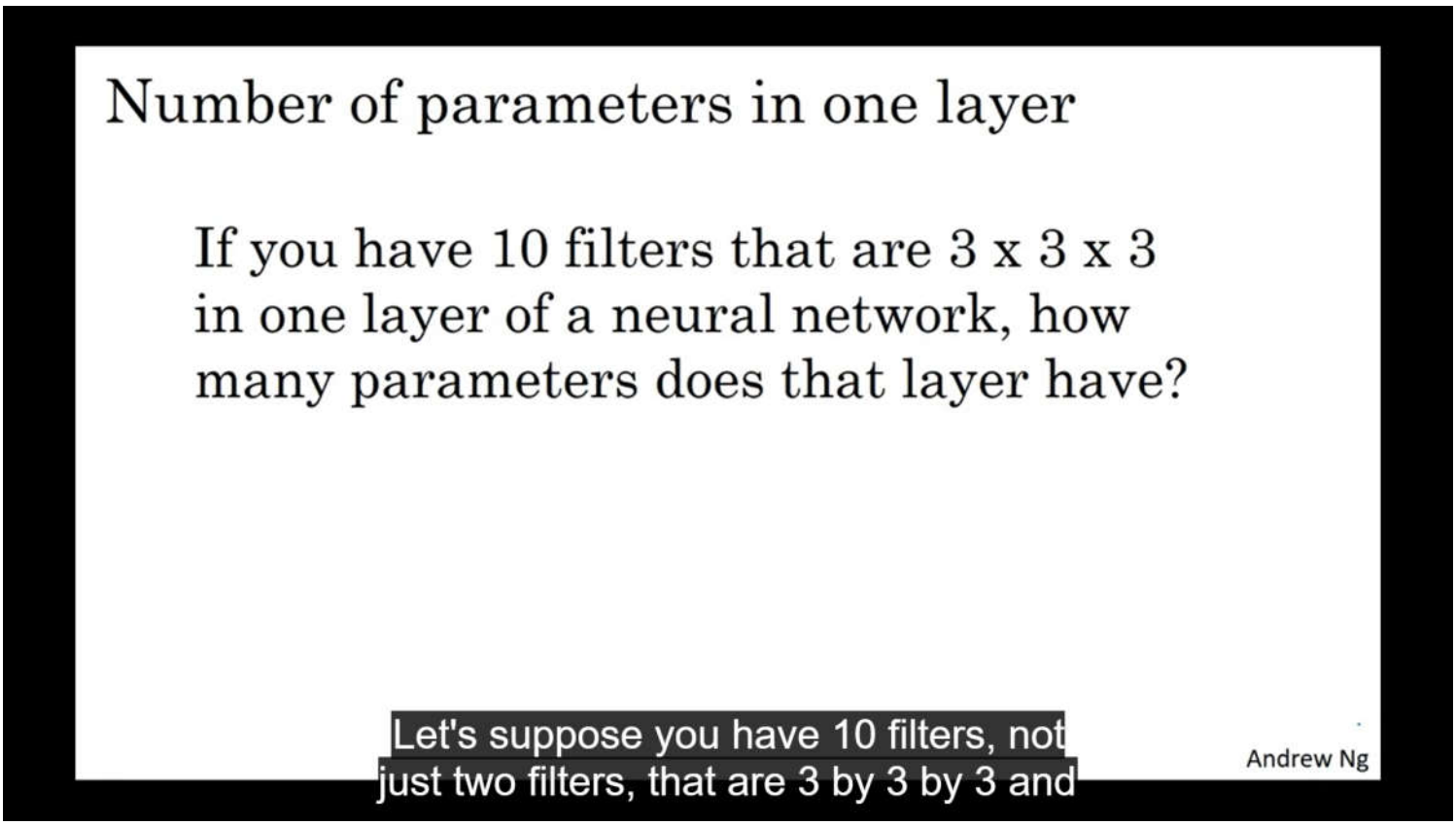


this is one layer of
a convolutional neural network.


Example of a layer
6 x 6 x 3
3 x 3 x 3
3 x 3 x 3
ReLU
4x4
4x4
4x4
4x4
4x4x2
Andrew Ng
And you apply the non-linearity
to get a[1], so that's g(z[1]).

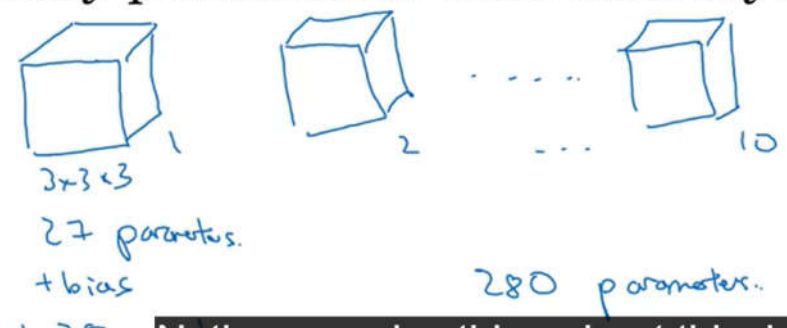

Example of a layer
6 x 6 x 3
3 x 3 x 3
3 x 3 x 3
ReLU
4x4
4x4
4x4
4x4
Andrew Ng
And you remember during the convolution
operation, you were taking


Example of a layer
6 x 6 x 3
3 x 3 x 3
3 x 3 x 3
4x4
4x4
4x4
4x4
4x4x2
Andrew Ng
this really becomes your
activation at the next layer.
3:52 / 16:10


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?
Andrew Ng
Let's suppose you have 10 filters, not just two filters, that are 3 by 3 by 3 and

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?

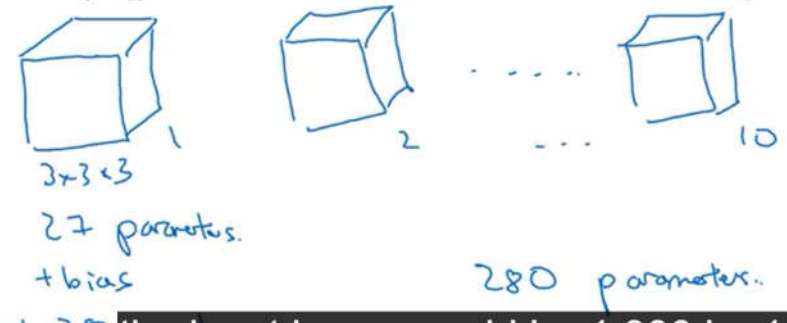


Notice one nice thing about this, is that no matter how big the input image is,

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?

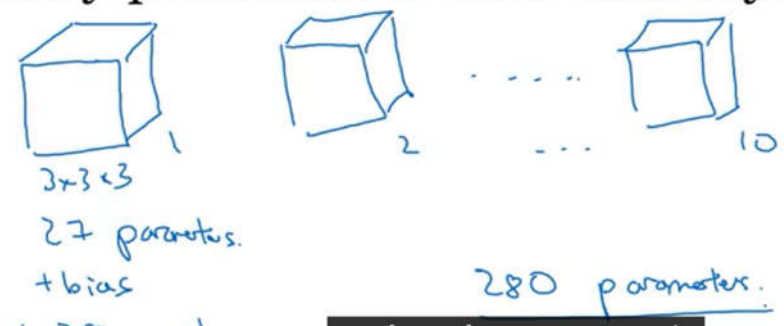


the input image could be 1,000 by 1,000 or 5,000 by 5,000,

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?



makes less prone to overfitting then if you could.

Andrew Ng

Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

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

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

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

while we know from the previous examples that that's equal

Andrew Ng

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:

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

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

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

Next, how about the size of this filter?

Andrew Ng

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

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

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

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

Which is why each filter is going to be f(l) by f(l) by nc(l-1).

Andrew Ng

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

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

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

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

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

And when you are using a vectorized implementation or batch gradient

Andrew Ng

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

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

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

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

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

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

If say you're using bash grading decent and

Andrew Ng

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

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

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

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

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

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

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

but that's the dimension of one filter.

Andrew Ng

13:17 / 16:10

Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

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

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

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

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

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

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

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

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

Andrew Ng

← #filters in layer l.

Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

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

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

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: $n_c^{[l]}$

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

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

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

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

← #filters in layer l.

it's just a vector of this dimension.

Andrew Ng

Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

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

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

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

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

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

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

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

← #filters in layer l.

Andrew Ng

Subtitles

✓ Subtitles Off

Chinese (Simplified)

Chinese (Traditional)

English

French

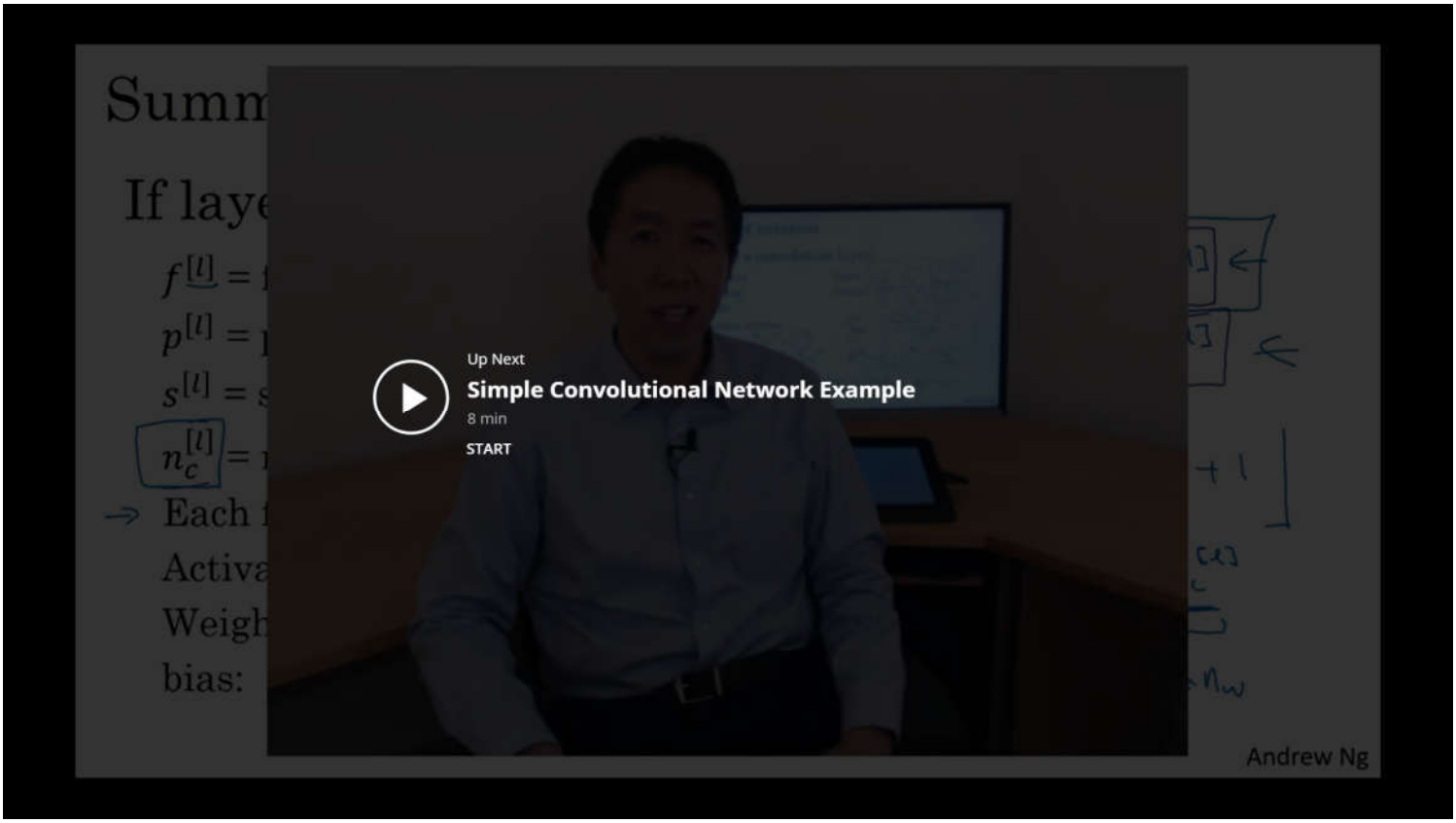
Japanese

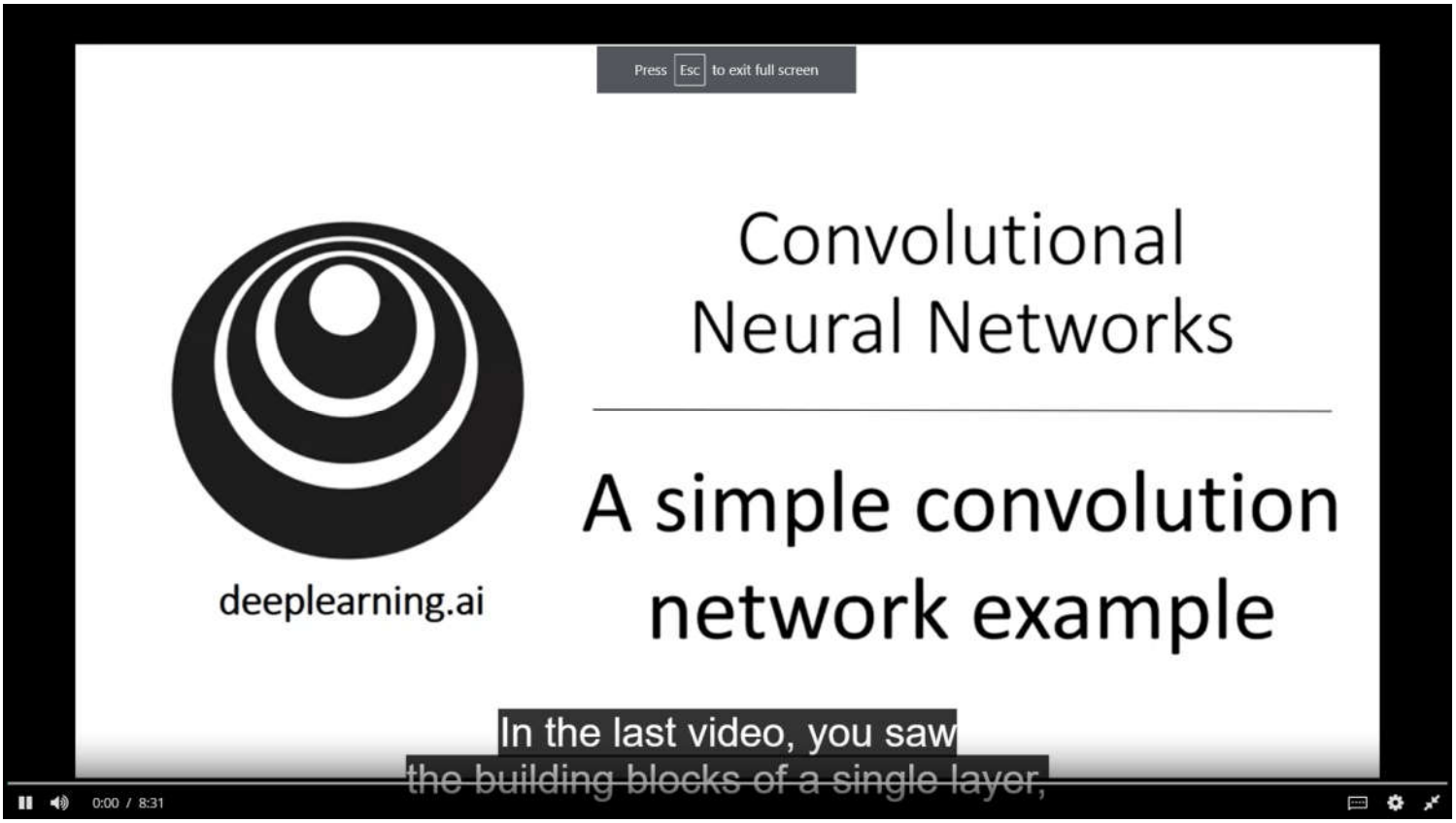
Korean

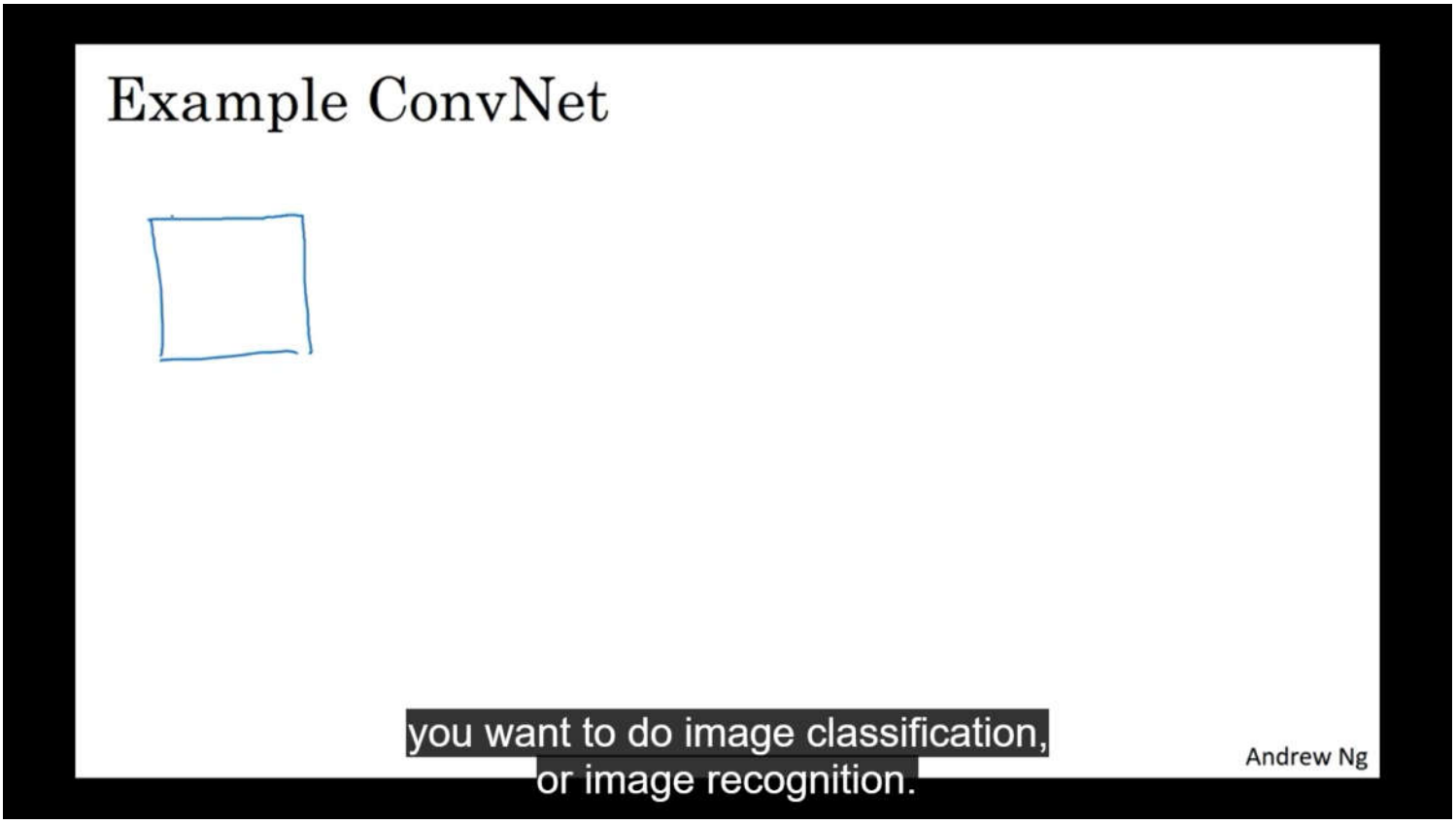
Spanish

Turkish

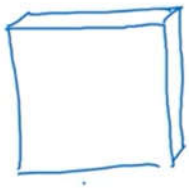
14:10 / 16:10


Up Next
Simple Convolutional Network Example
8 min
START
Andrew Ng


Press Esc to exit full screen
Convolutional
Neural Networks
A simple convolution
network example
deeplearning.ai
In the last video, you saw
the building blocks of a single layer,
0:00 / 8:31


Example ConvNet
you want to do image classification,
or image recognition.
Andrew Ng

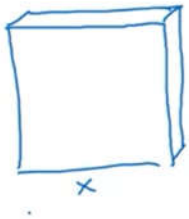
Example ConvNet



Where you want to take as input an image,
x, and decide is this a cat or not, 0 or

Andrew Ng

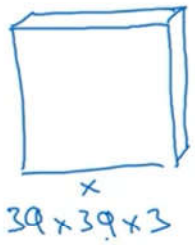
Example ConvNet



For the sake of this example,
I'm going to use a fairly small image.

Andrew Ng

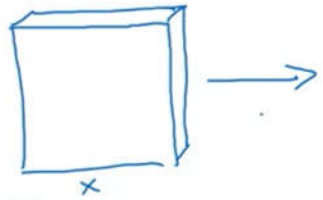
Example ConvNet



This choice just makes some of the numbers work out a bit better.

Andrew Ng

Example ConvNet



$39 \times 39 \times 3$

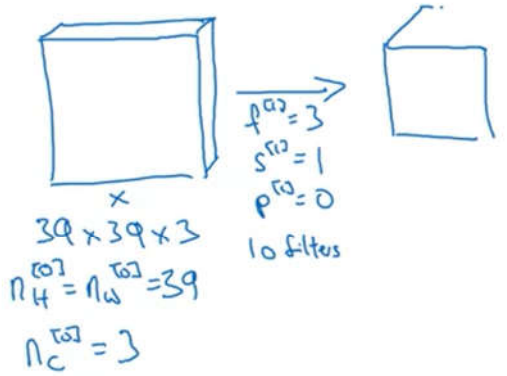
$n_H^{[0]} = n_W^{[0]} = 39$

$n_C^{[0]} = 3$

Let's say the first layer uses a set of 3 by 3 filters

Andrew Ng

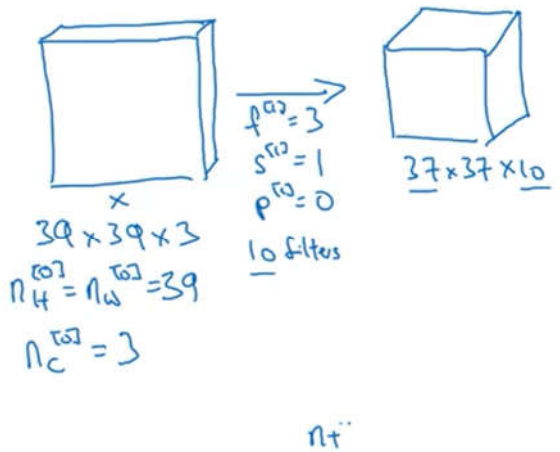
Example ConvNet



Then the activations in this next layer of

Andrew Ng

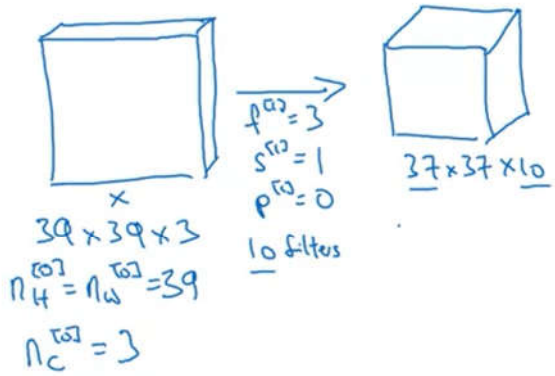
Example ConvNet



And 37 comes from this formula

Andrew Ng

Example ConvNet



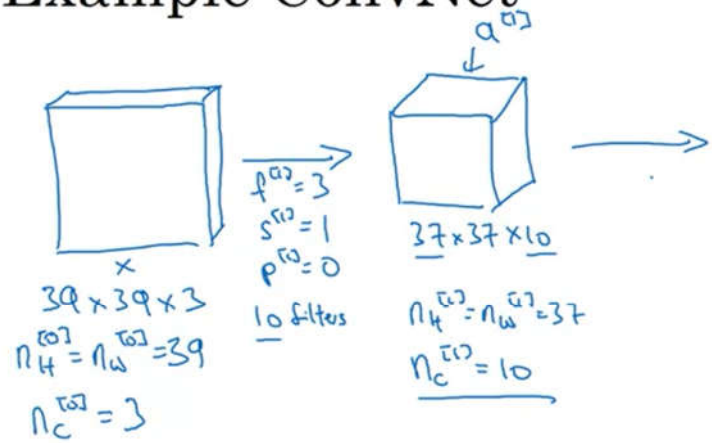
$$\frac{n+2p-f}{s}+1$$

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

So in our notation you would have nh[1] = nw[1] = 37 and

Andrew Ng

Example ConvNet

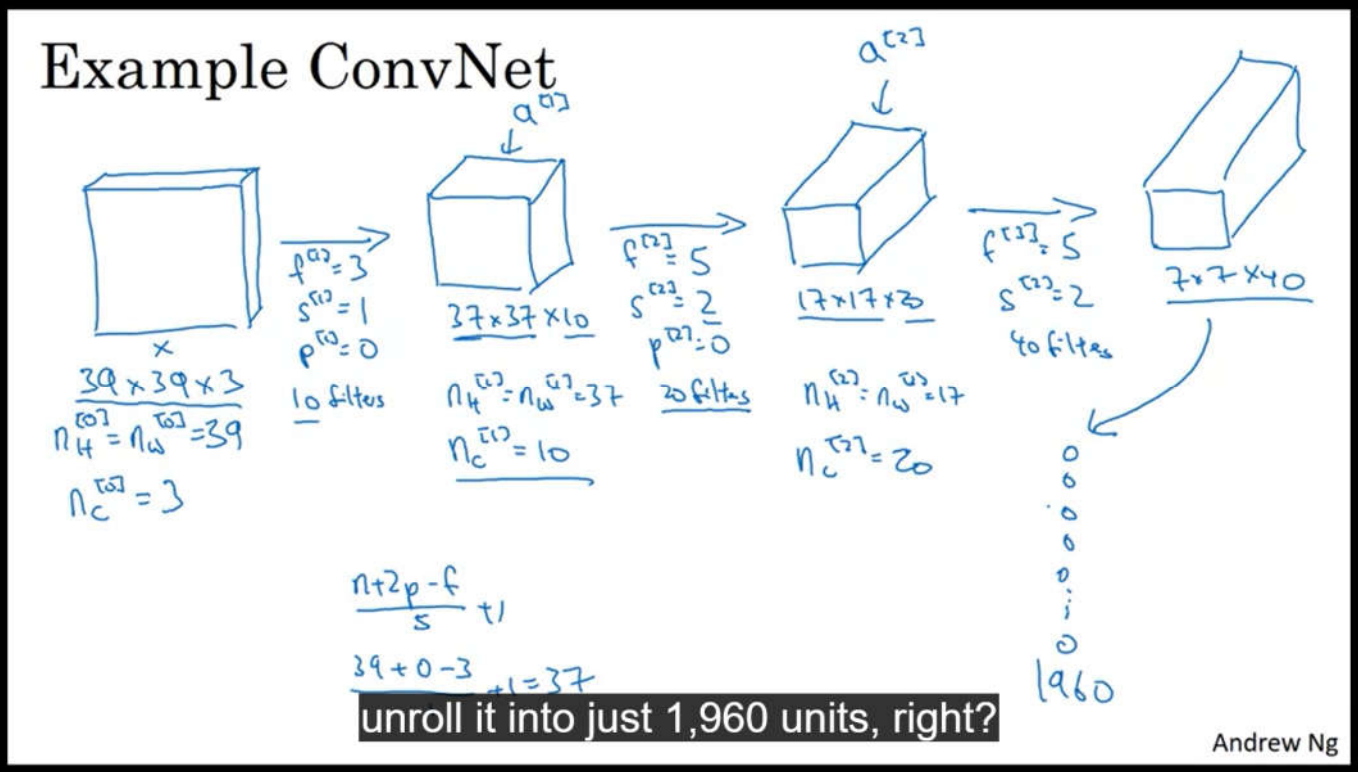


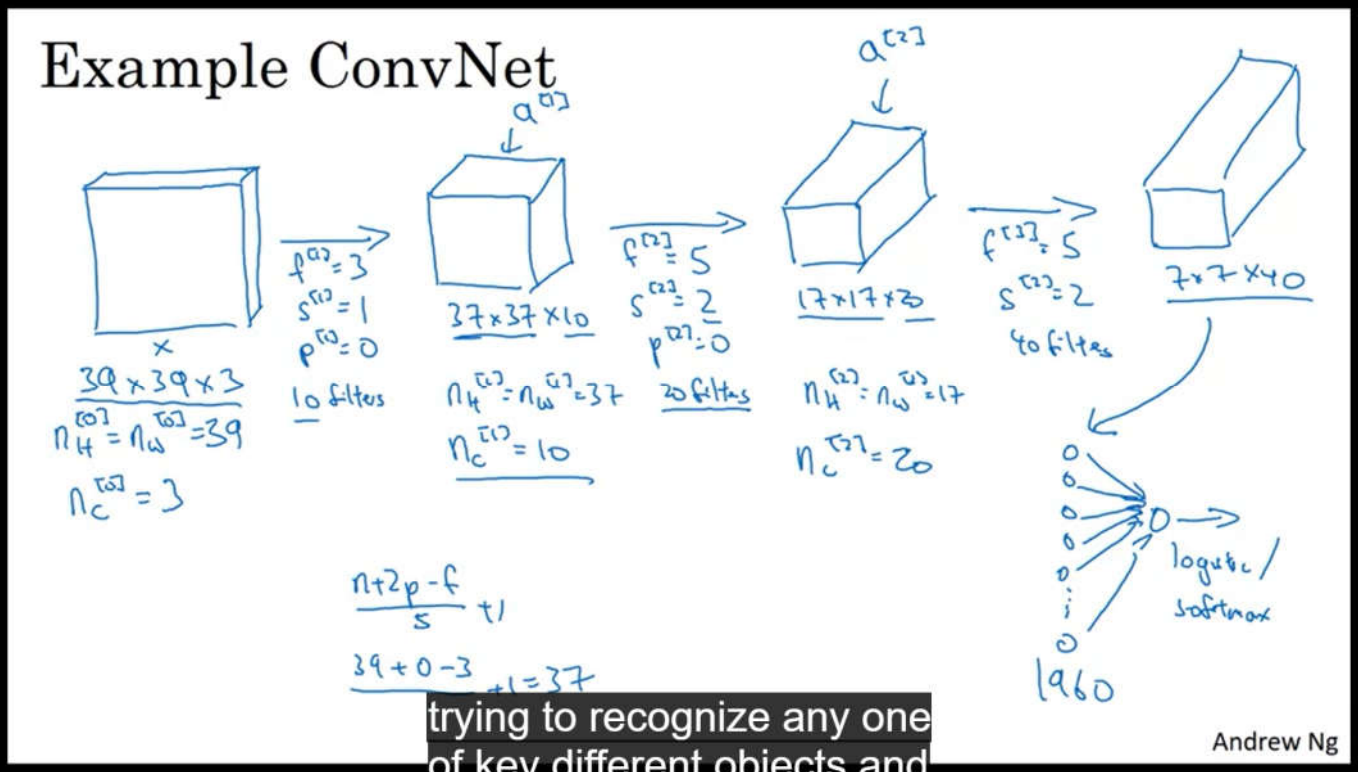
$$\frac{n+2p-f}{s}+1$$

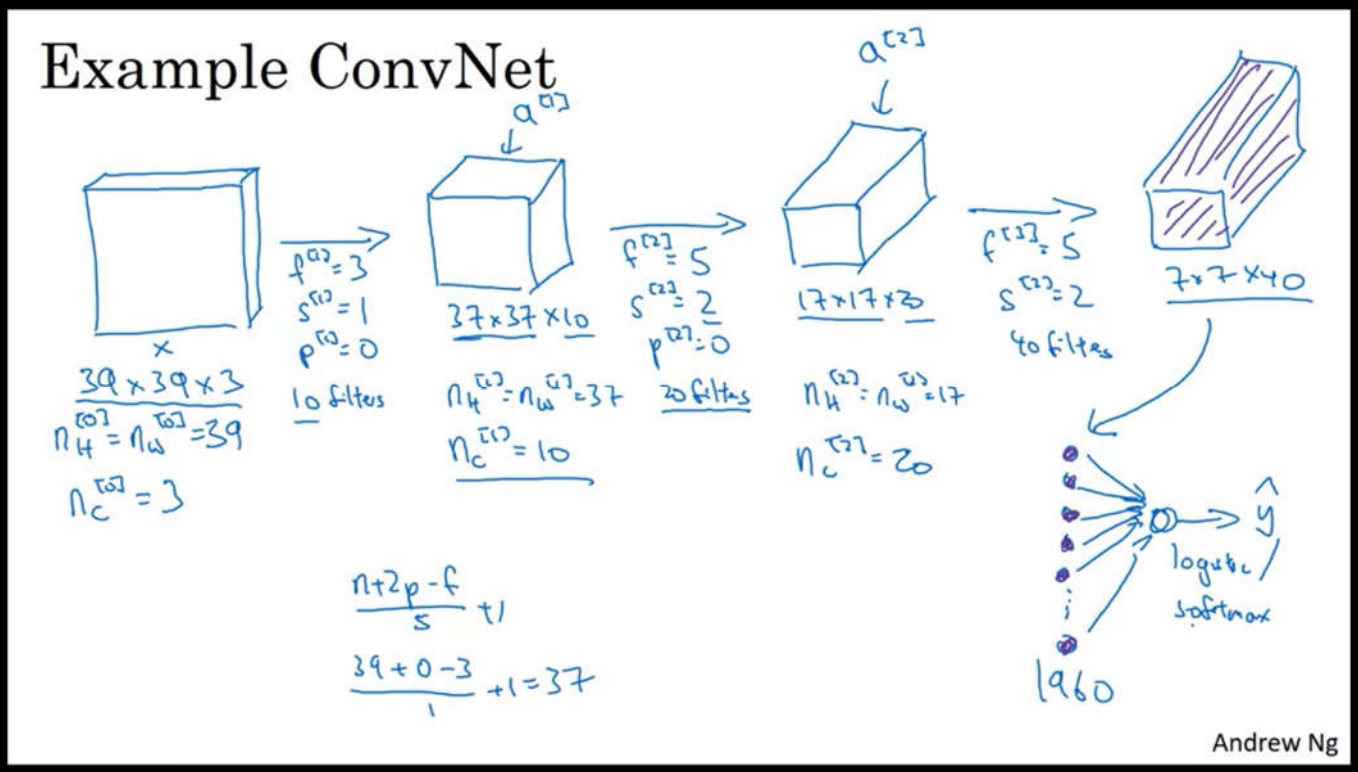
$$\frac{39+0-3}{1}+1=37$$

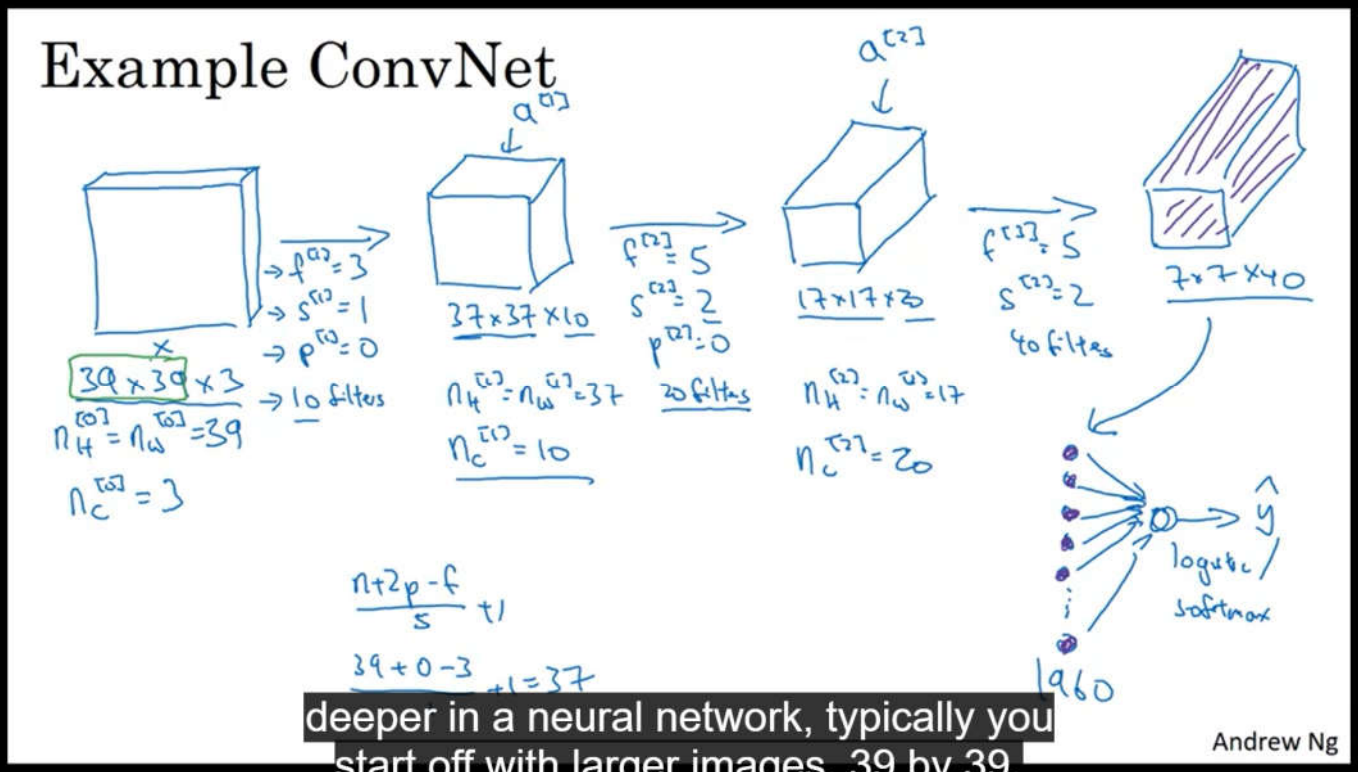
Let's say you now have another convolutional layer and

Andrew Ng


Example ConvNet
x
39 x 39 x 3
$n_H^{[0]} = n_W^{[0]} = 39$
$n_c^{[0]} = 3$
$f^{[1]} = 3$
$s^{[1]} = 1$
$p^{[1]} = 0$
10 filters
$a^{[1]}$
37 x 37 x 10
$n_H^{[1]} = n_W^{[1]} = 37$
$n_c^{[1]} = 10$
$f^{[2]} = 5$
$s^{[2]} = 2$
$p^{[2]} = 0$
20 filters
$a^{[2]}$
17 x 17 x 20
$n_H^{[2]} = n_W^{[2]} = 17$
$n_c^{[2]} = 20$
$f^{[3]} = 5$
$s^{[3]} = 2$
40 filters
7 x 7 x 40
1960
$\frac{n+2p-f}{s} + 1$
$\frac{39+0-3}{} + 1 = 37$
unroll it into just 1,960 units, right?
Andrew Ng

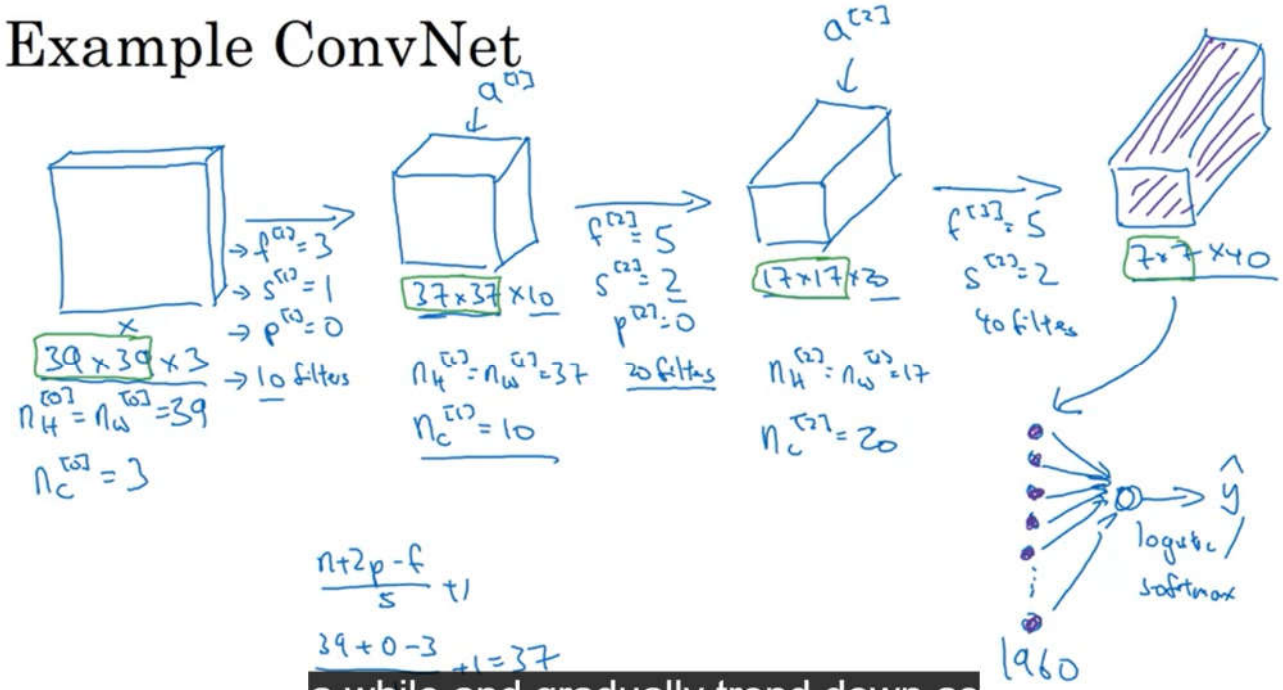

Example ConvNet
x
39 x 39 x 3
$n_H^{[0]} = n_W^{[0]} = 39$
$n_c^{[0]} = 3$
$f^{[1]} = 3$
$s^{[1]} = 1$
$p^{[1]} = 0$
10 filters
$a^{[1]}$
37 x 37 x 10
$n_H^{[1]} = n_W^{[1]} = 37$
$n_c^{[1]} = 10$
$f^{[2]} = 5$
$s^{[2]} = 2$
$p^{[2]} = 0$
20 filters
$a^{[2]}$
17 x 17 x 20
$n_H^{[2]} = n_W^{[2]} = 17$
$n_c^{[2]} = 20$
$f^{[3]} = 5$
$s^{[3]} = 2$
40 filters
7 x 7 x 40
1960
logistic / softmax
$\frac{n+2p-f}{s} + 1$
$\frac{39+0-3}{} + 1 = 37$
trying to recognize any one
of key different objects and
Andrew Ng





deeper in a neural network, typically you start off with larger images, 39 by 39.

Example ConvNet



a while and gradually trend down as you go deeper in the neural network.

Andrew Ng

Types of layer in a convolutional network:

- Convolution (CONV)

And that's what we've been using in the previous network.

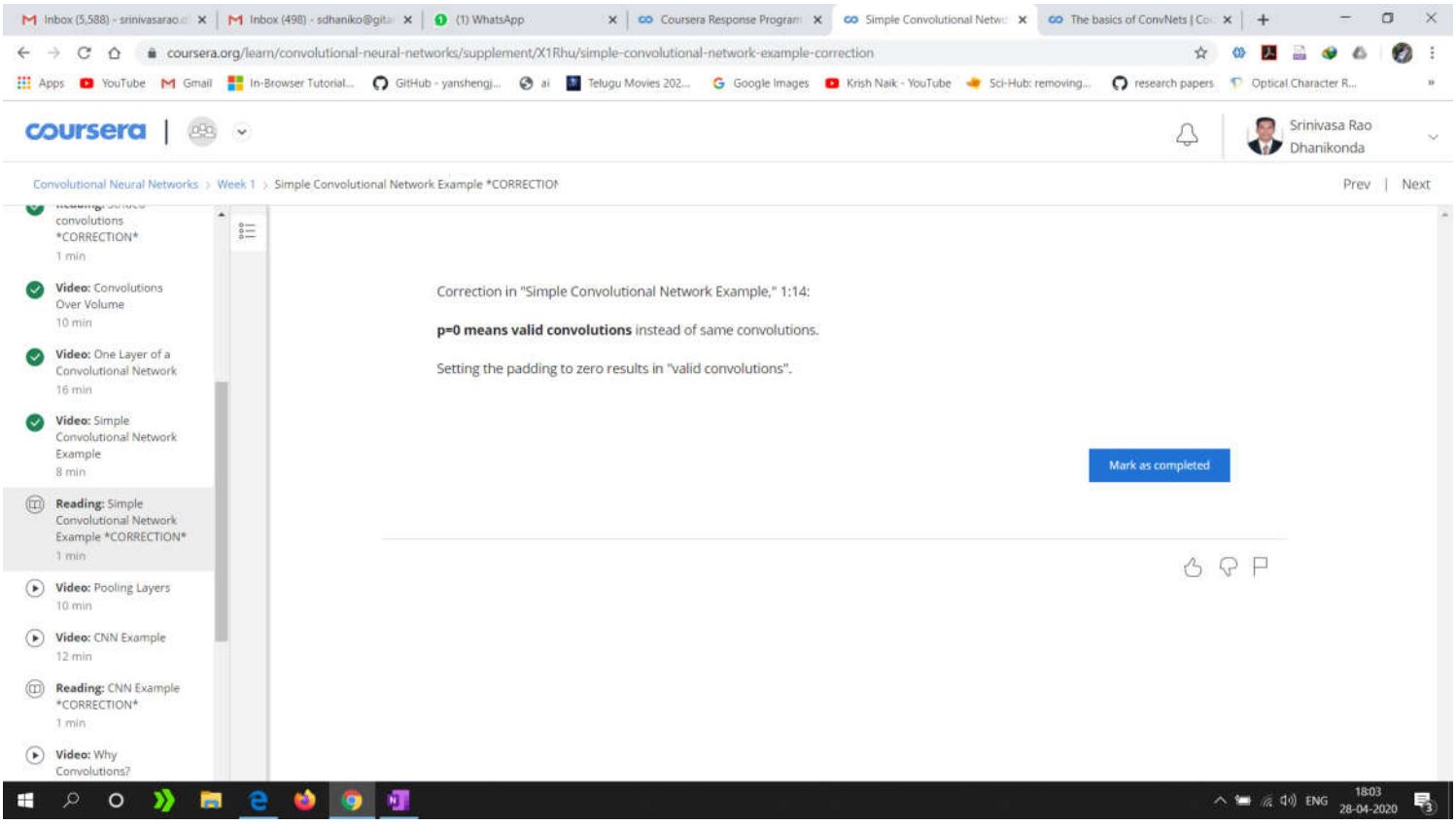
Andrew Ng

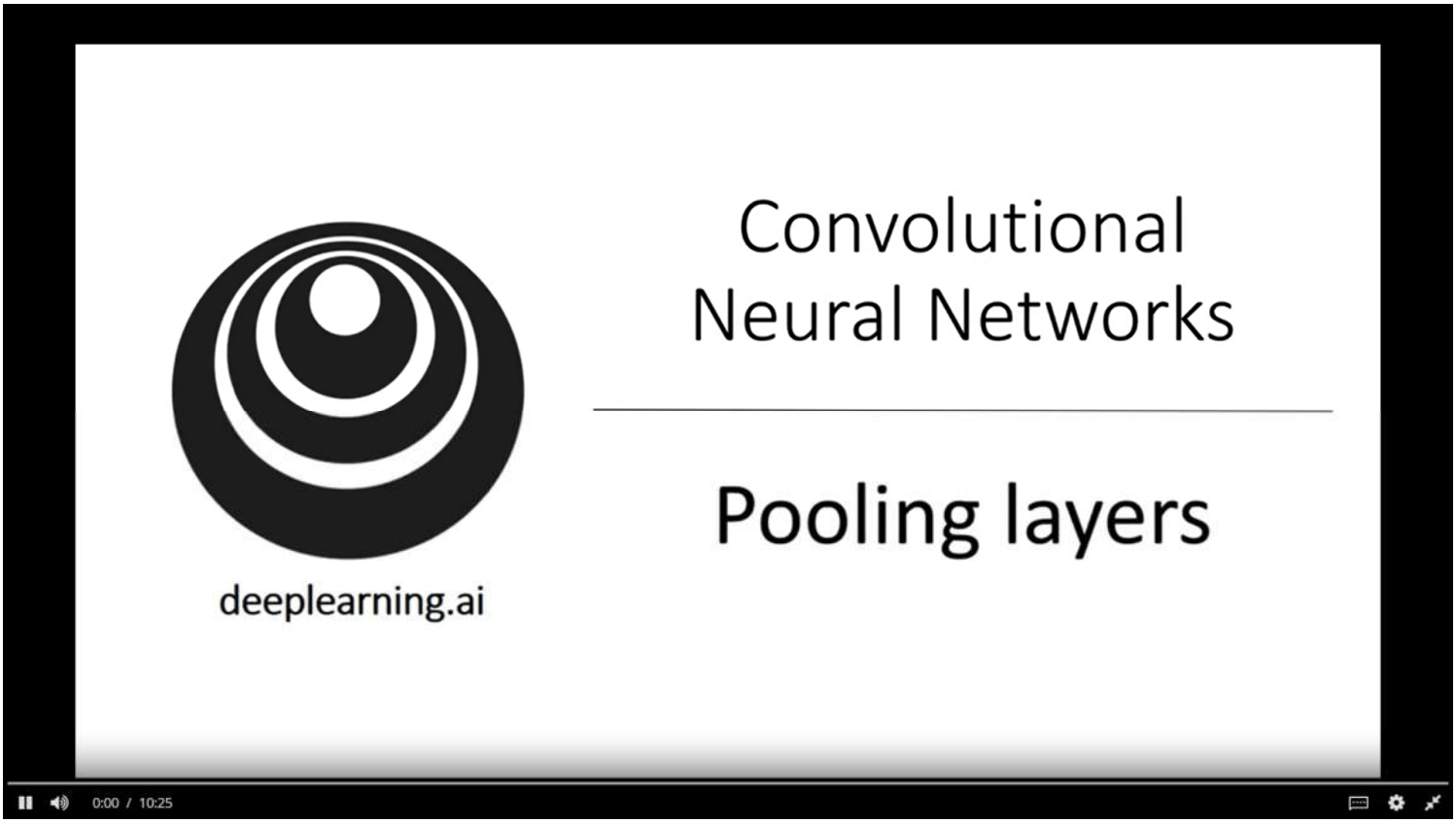
Types of layer in a convolutional network:

- Convolution (CONV)
- Pooling (POOL)
- Fully connected (FC)

And then the last is a fully connected layer called FC.

Andrew Ng




Convolutional
Neural Networks
Pooling layers
deeplearning.ai
0:00 / 10:25

Pooling layer: Max pooling

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

and then we'll talk about why you might want to do this.

Andrew Ng

Pooling layer: Max pooling

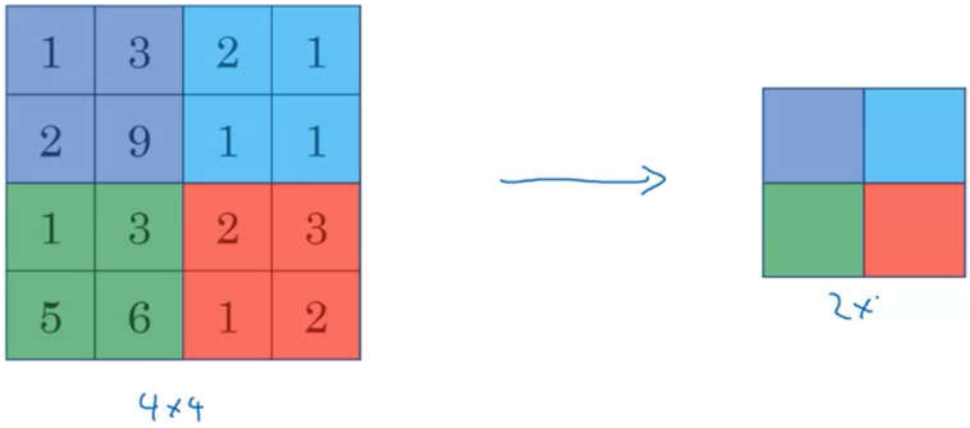
1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

4×4

Take your four by four input and break it into

Andrew Ng

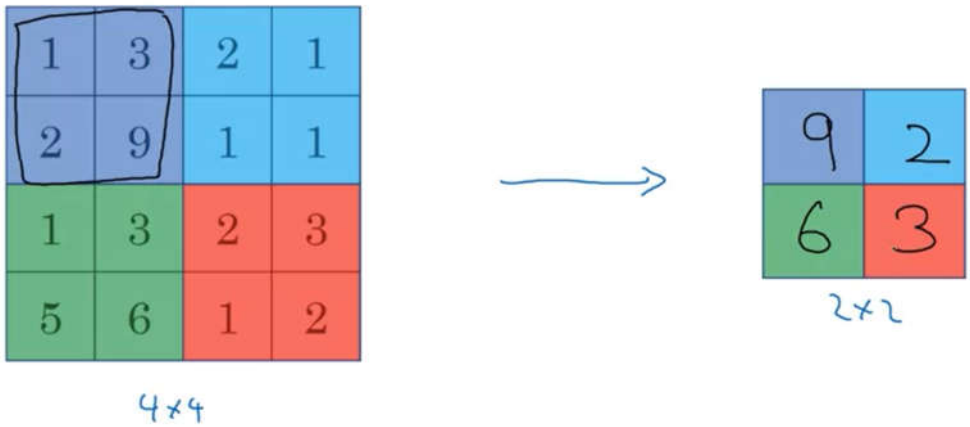
Pooling layer: Max pooling



which is two by two,

Andrew Ng

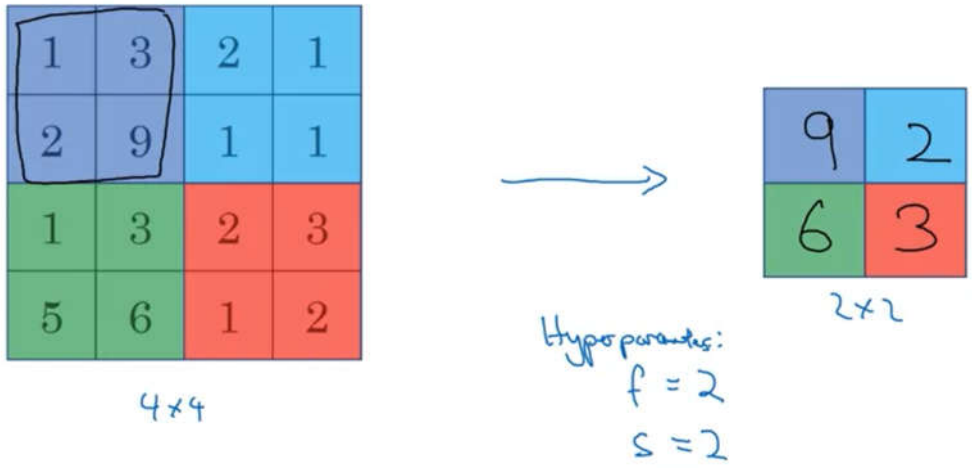
Pooling layer: Max pooling



and lower right, the biggest number is three.

Andrew Ng

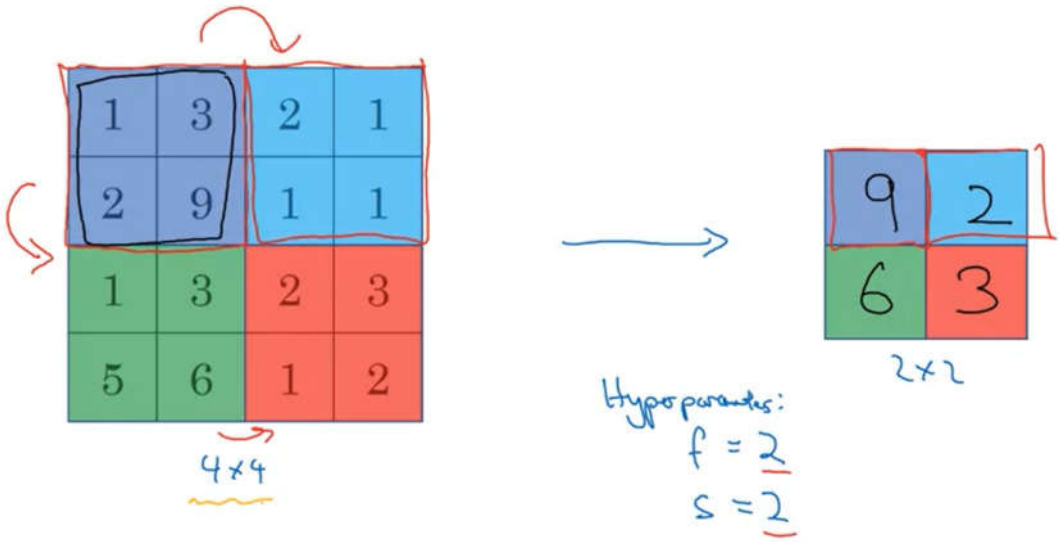
Pooling layer: Max pooling



max pooling because we start from this filter size.

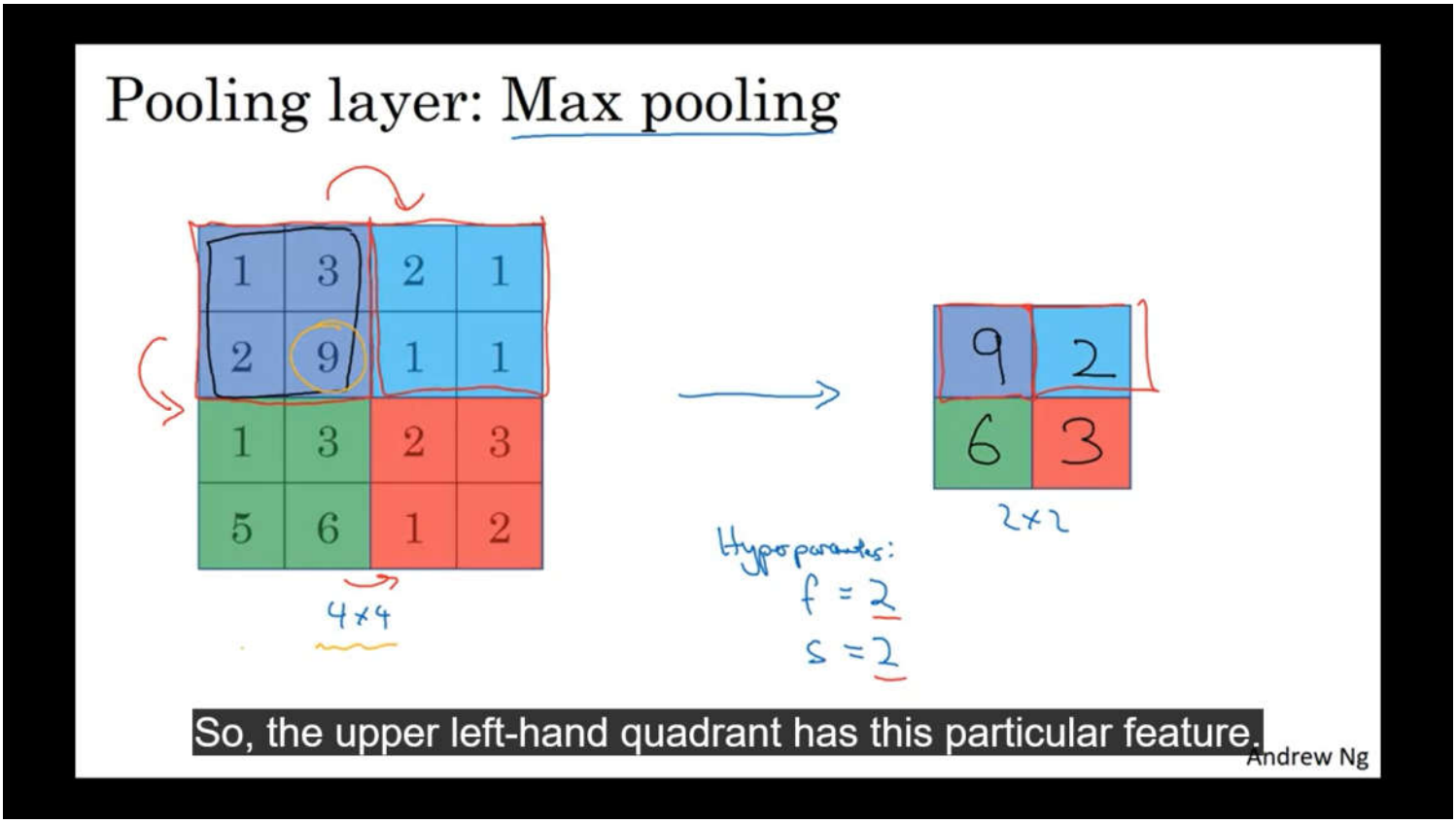
Andrew Ng

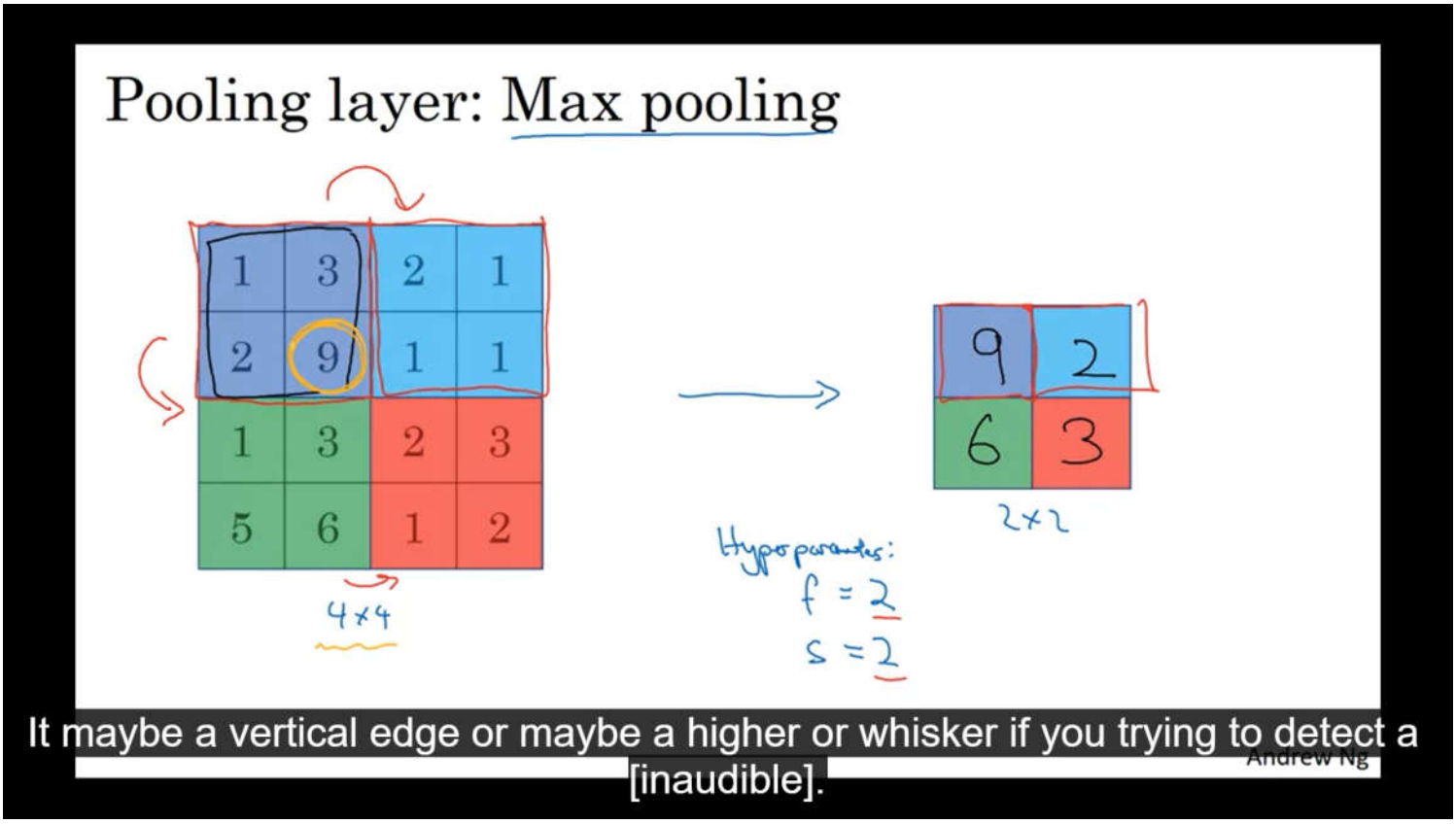
Pooling layer: Max pooling

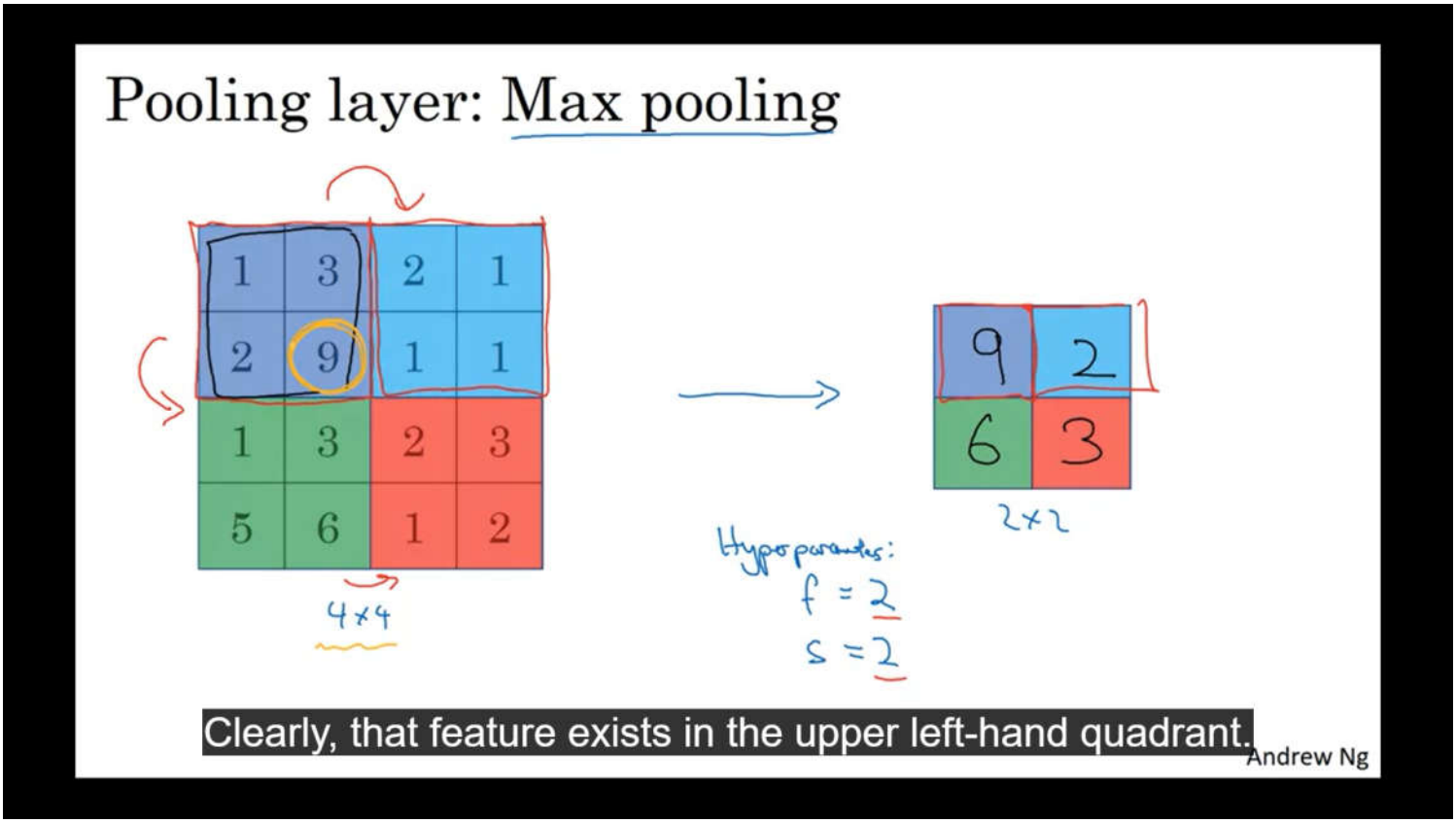


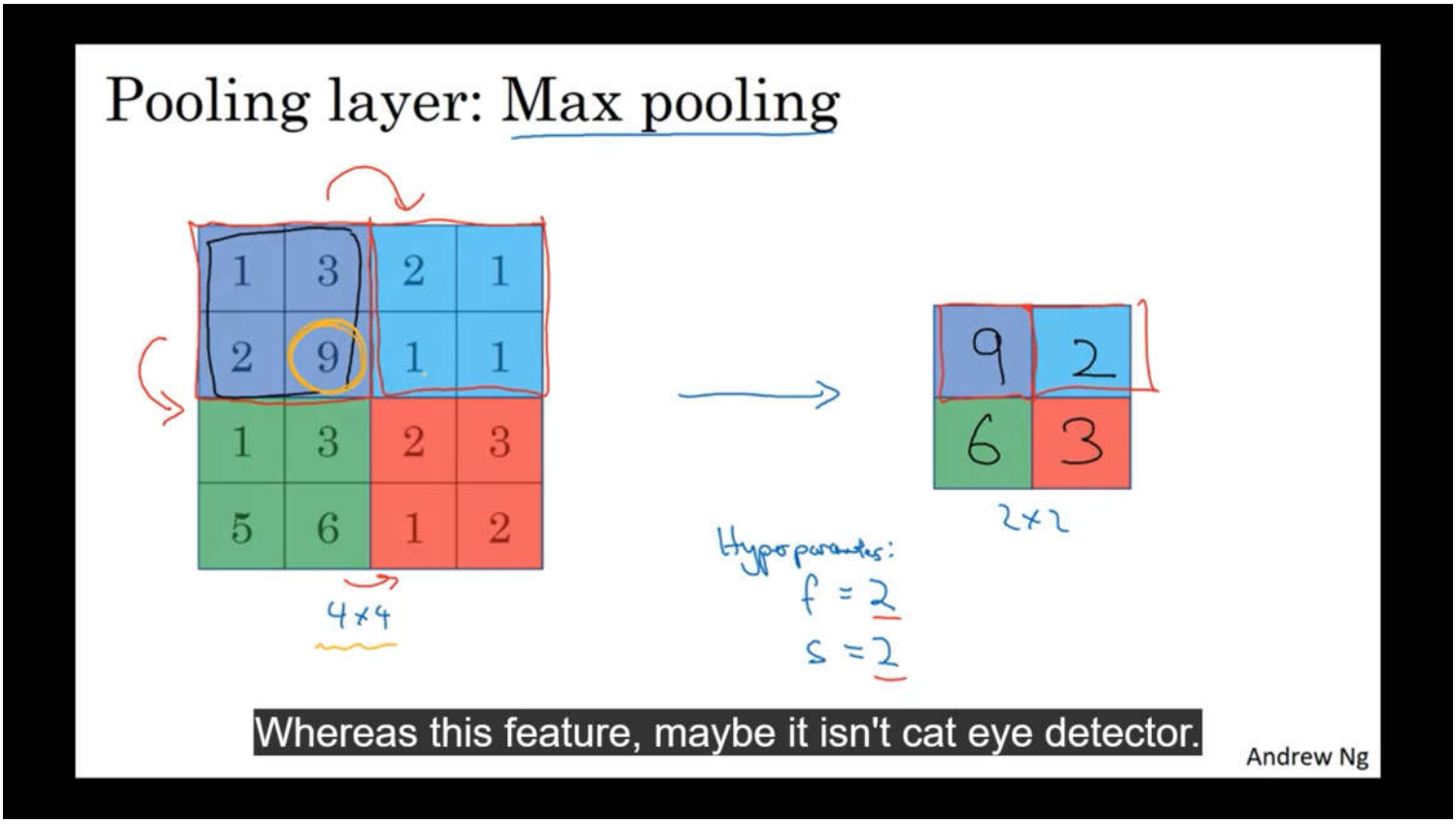
If you think of this four by four region as some set of features,

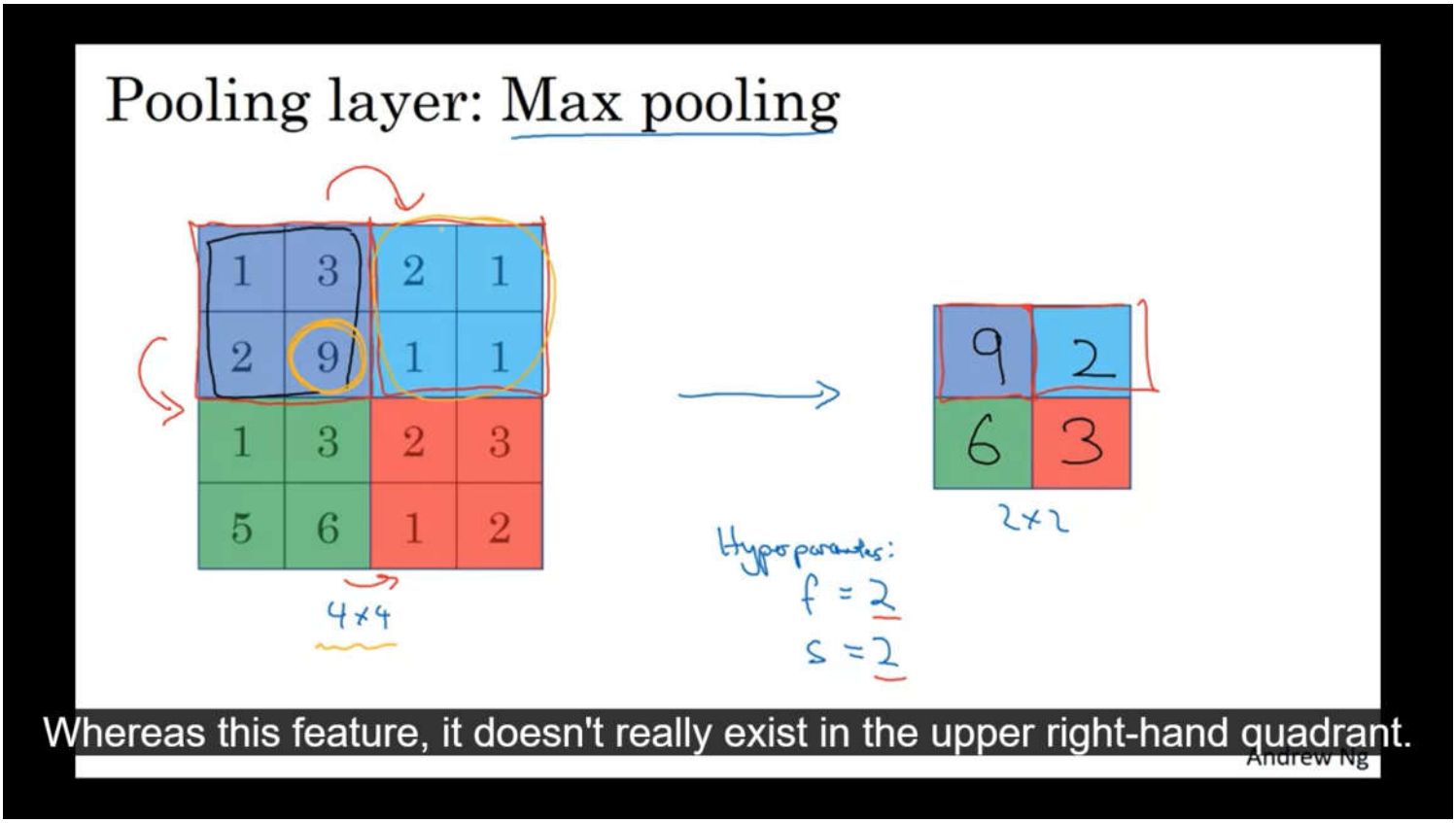
Andrew Ng

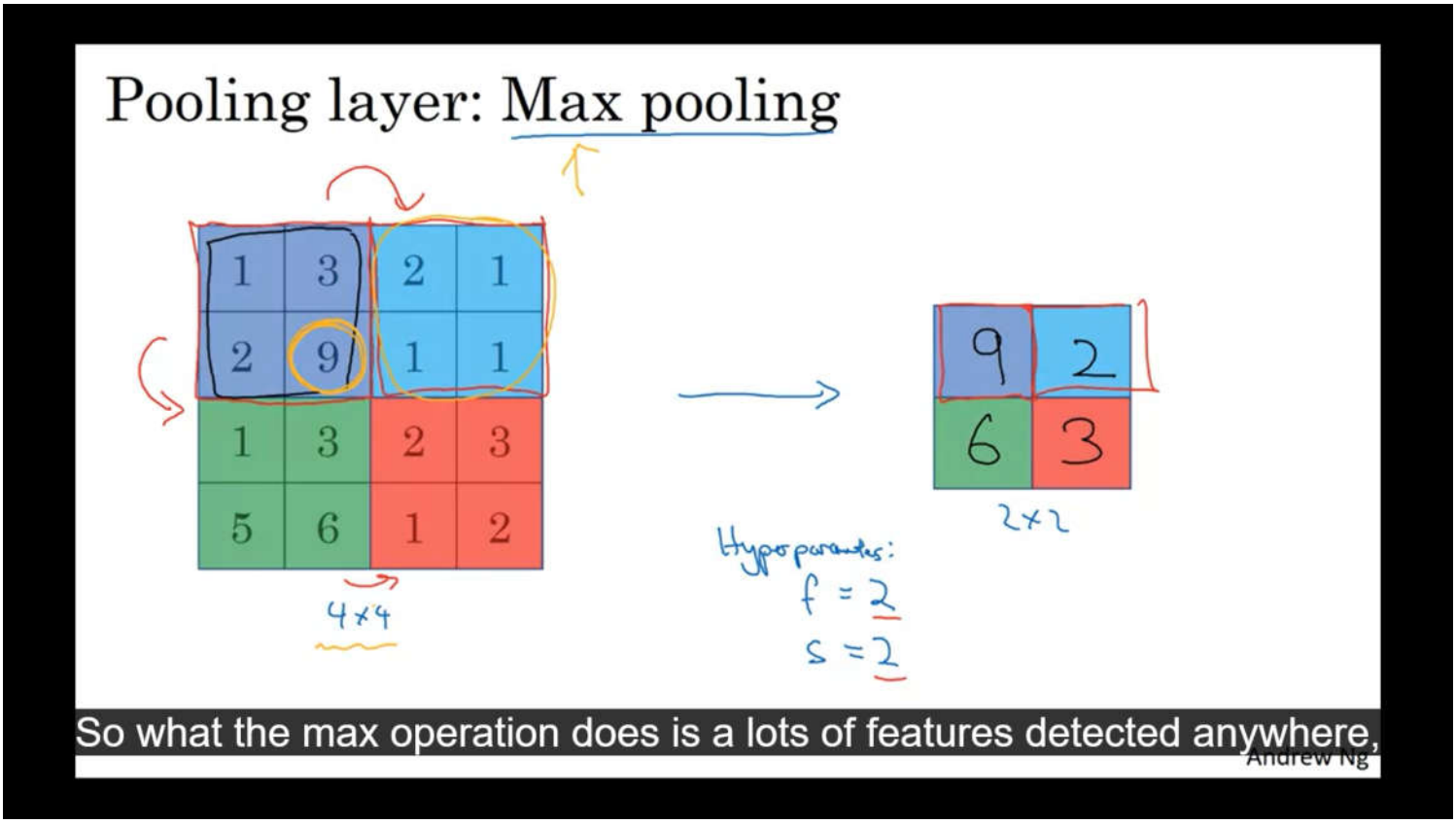

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
So, the upper left-hand quadrant has this particular feature.
Andrew Ng

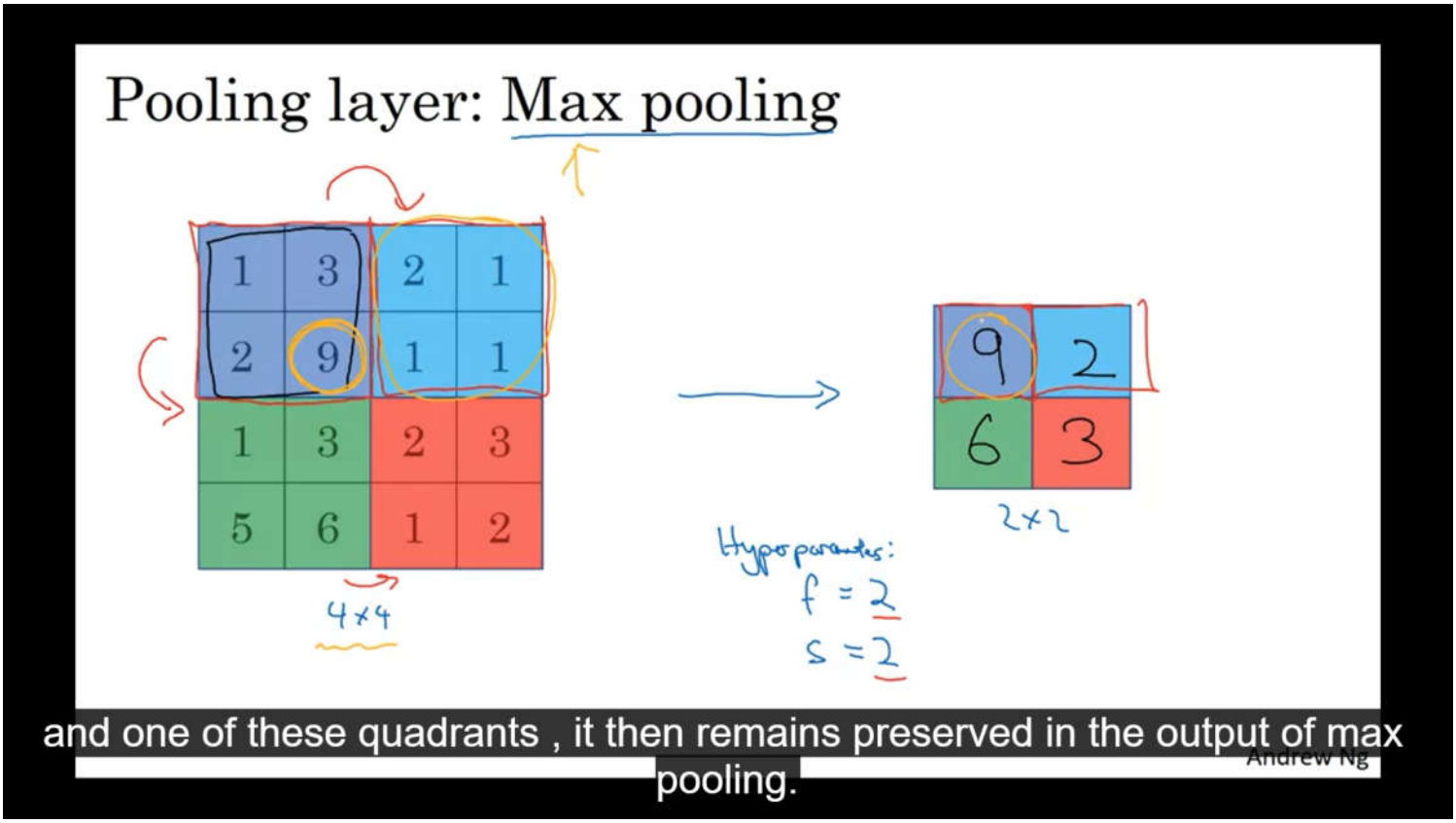

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
It maybe a vertical edge or maybe a higher or whisker if you trying to detect a [inaudible].
Andrew Ng

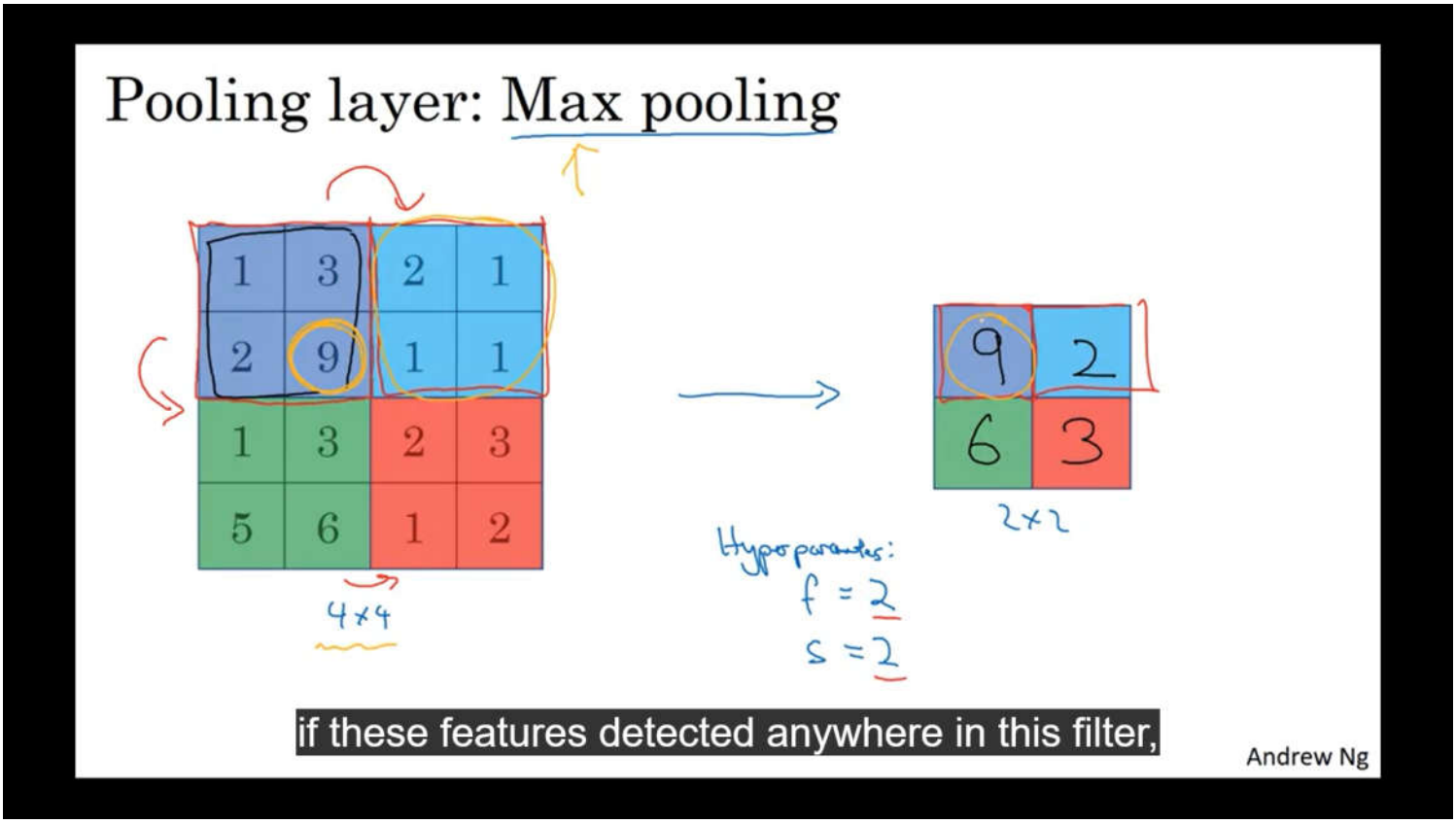

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
Clearly, that feature exists in the upper left-hand quadrant.
Andrew Ng

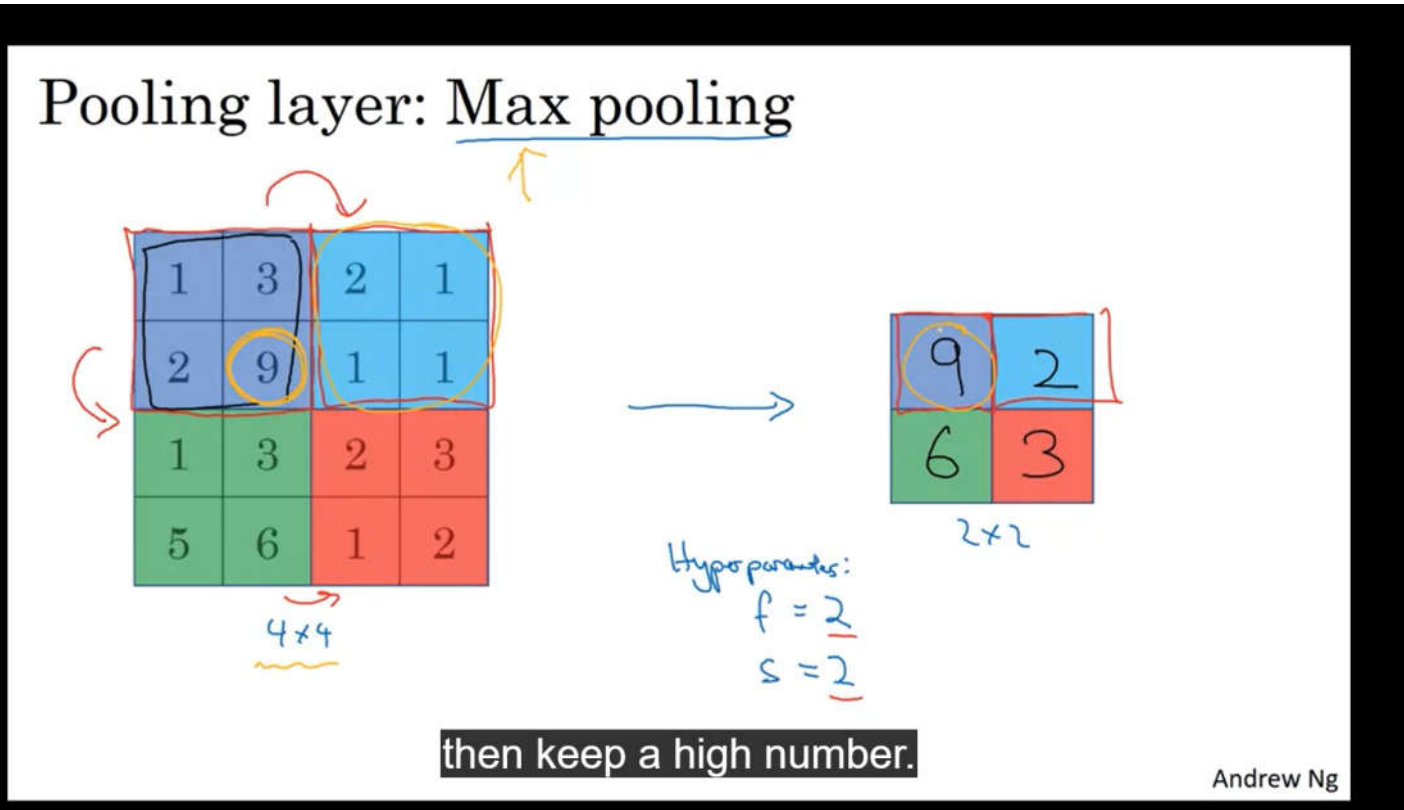

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
Whereas this feature, maybe it isn't cat eye detector.
Andrew Ng

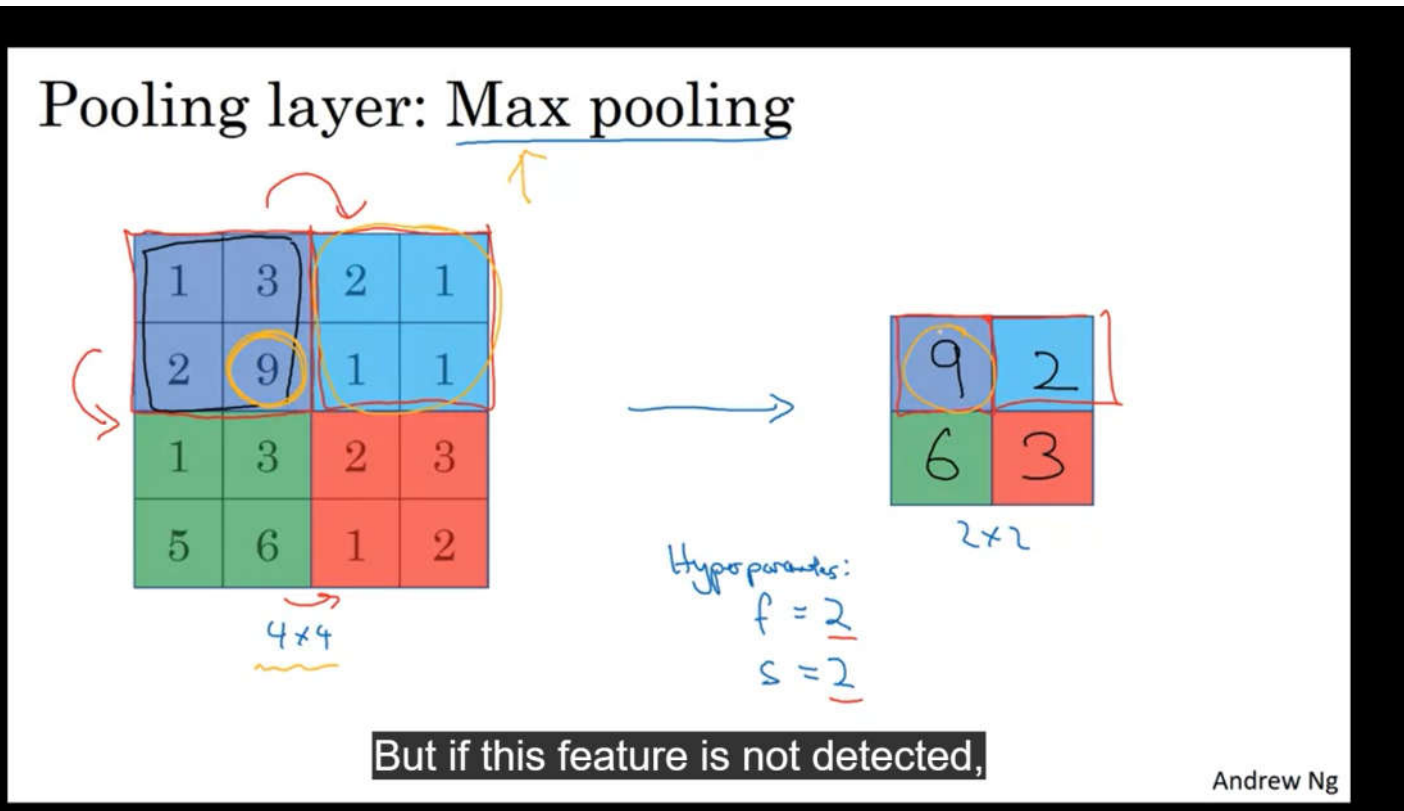

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
Whereas this feature, it doesn't really exist in the upper right-hand quadrant.
Andrew Ng

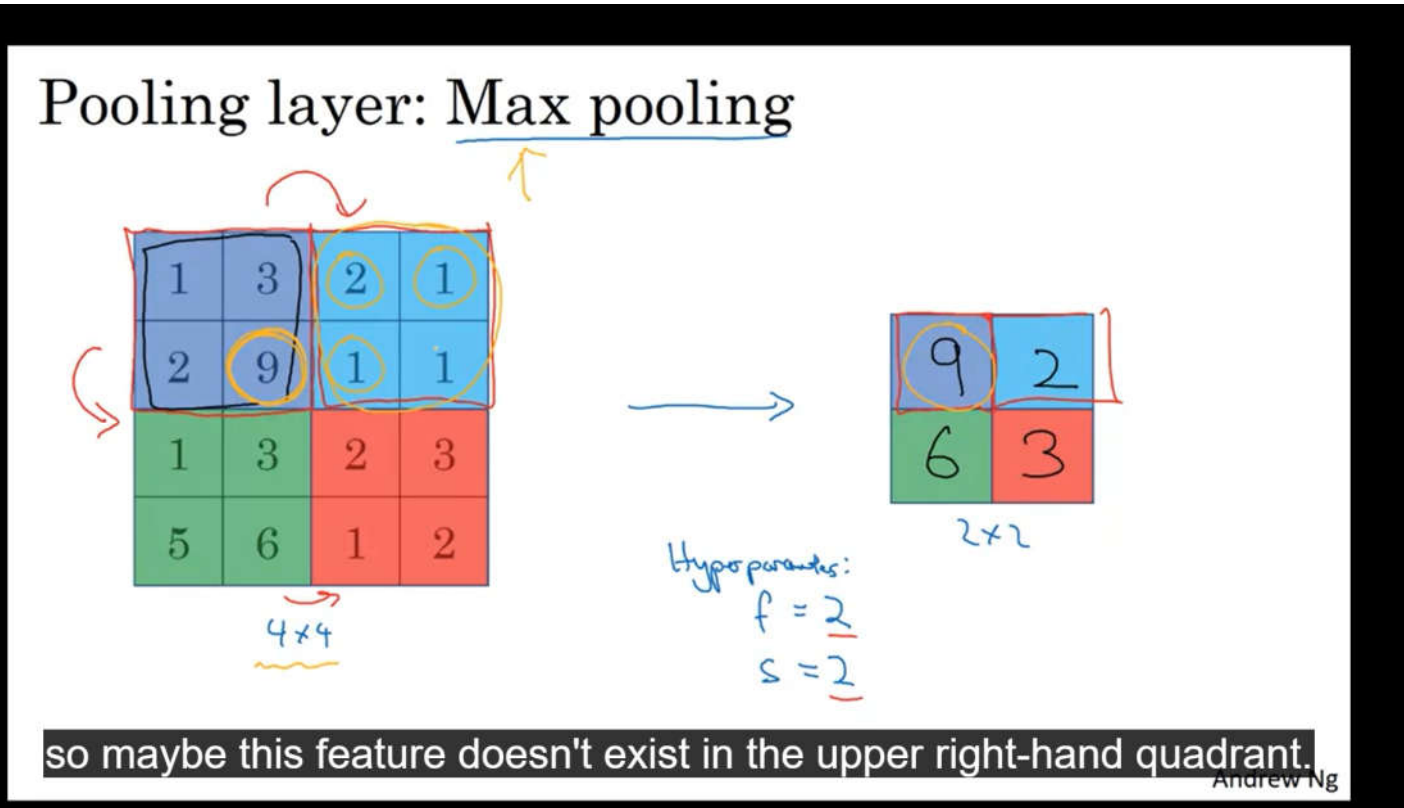

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
So what the max operation does is a lots of features detected anywhere,
Andrew Ng

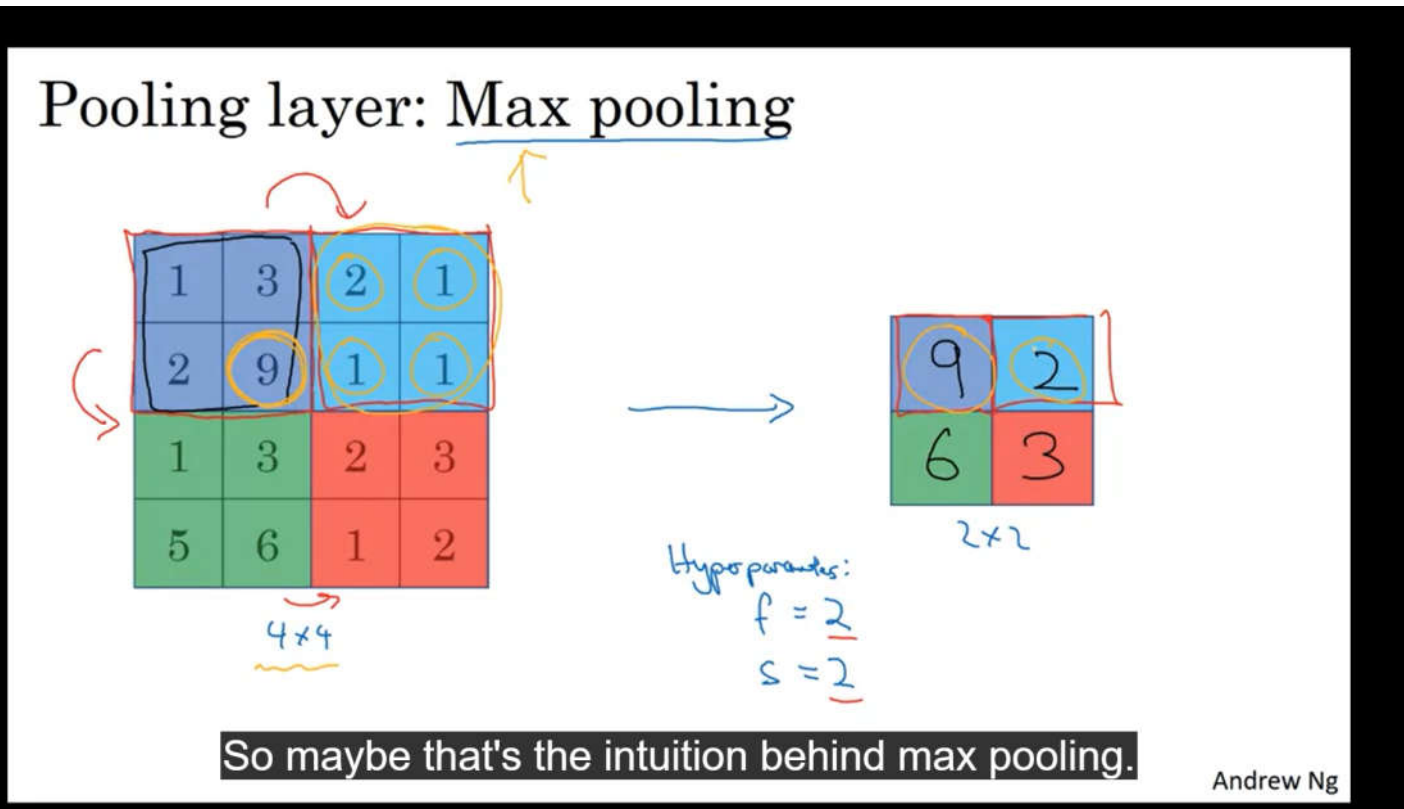

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
and one of these quadrants , it then remains preserved in the output of max pooling.
Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
if these features detected anywhere in this filter,
Andrew Ng

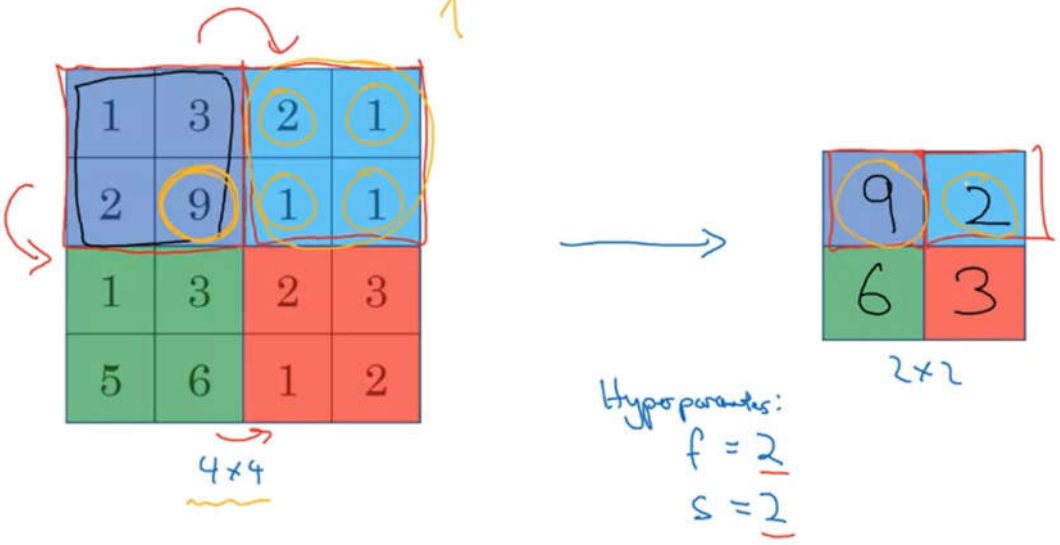

Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
then keep a high number.
Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
But if this feature is not detected,
Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
so maybe this feature doesn't exist in the upper right-hand quadrant.
Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
So maybe that's the intuition behind max pooling.
Andrew Ng

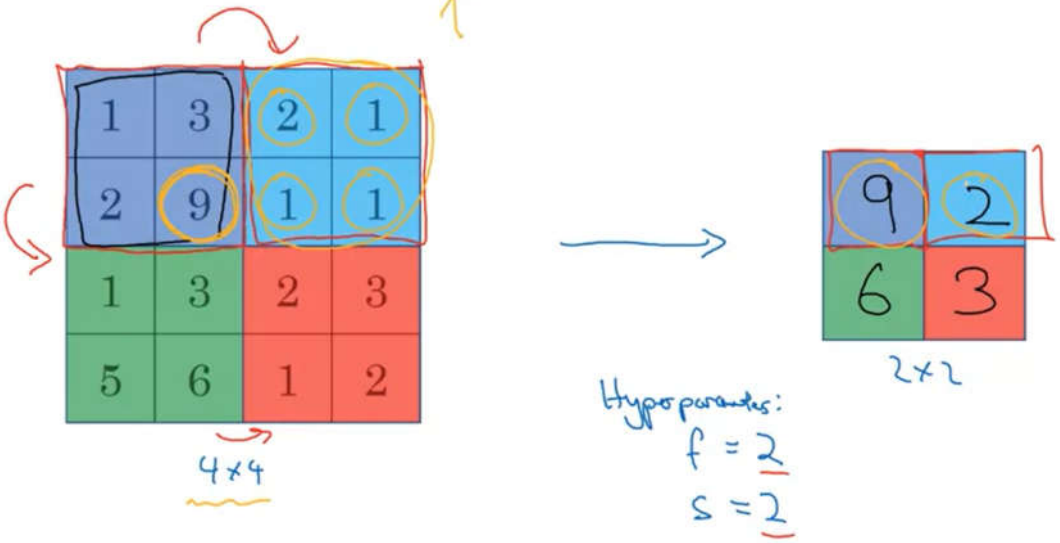
Pooling layer: Max pooling



because it's been found in a lot of experiments to work well.

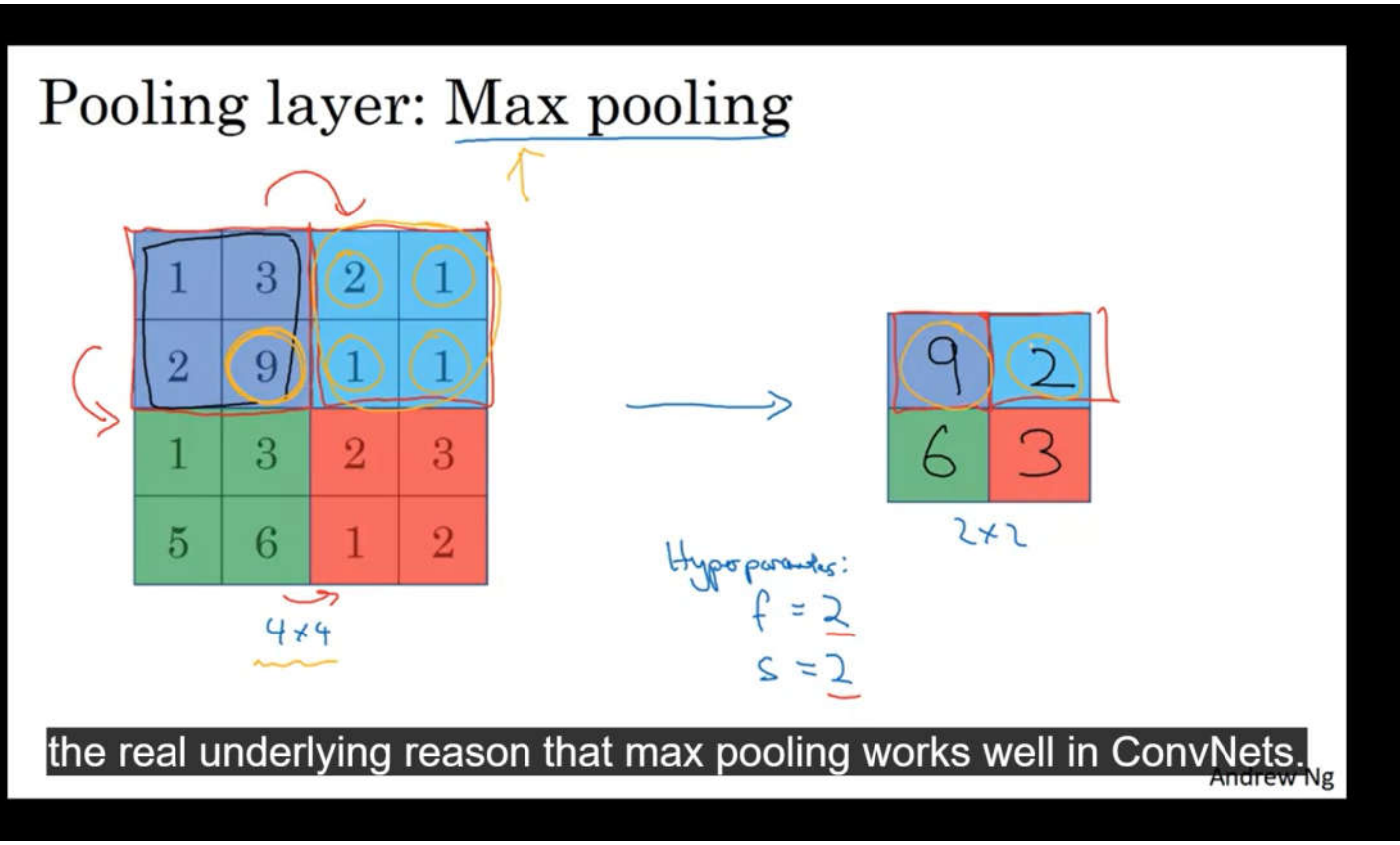
Andrew Ng

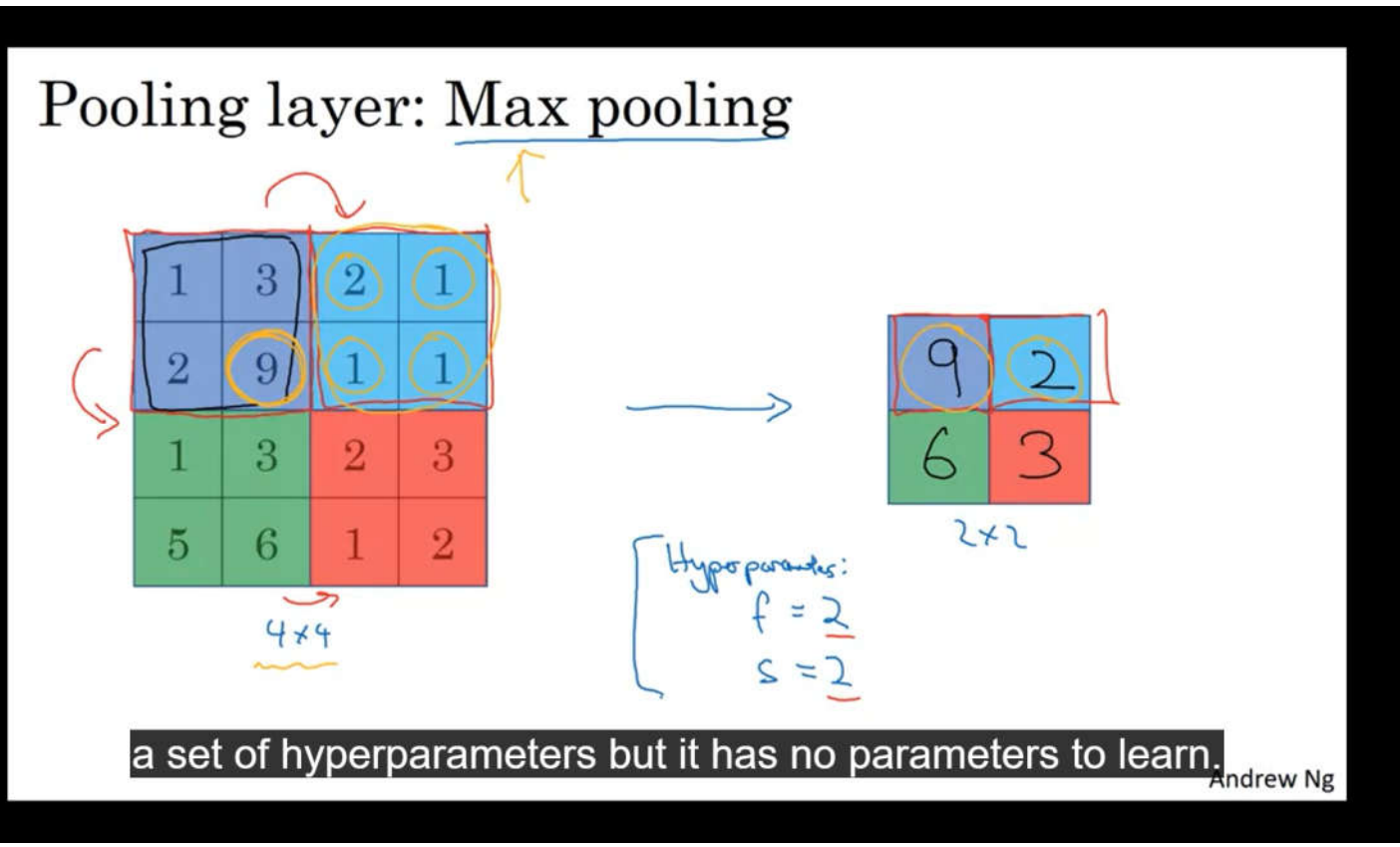
Pooling layer: Max pooling



I don't have anyone knows if that's

Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
the real underlying reason that max pooling works well in ConvNets.
Andrew Ng


Pooling layer: Max pooling
1 3 2 1
2 9 1 1
1 3 2 3
5 6 1 2
4×4
9 2
6 3
2×2
Hyperparameters:
f = 2
s = 2
a set of hyperparameters but it has no parameters to learn.
Andrew Ng

Pooling layer: Max pooling

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

Let's go through an example with some different hyperparameters.

Andrew Ng

Pooling layer: Max pooling

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

5x5

f = 3

s = 1

So in this case, the output size is going to be three by three.

Andrew Ng

Pooling layer: Max pooling

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

5x5

9	9	5
9	9	5
8	6	9

3x3

$f = 3$

$s = 1$

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

Andrew Ng

Pooling layer: Max pooling

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

5x5

9	9	5
9	9	5
8	6	9

3x3

$f = 3$

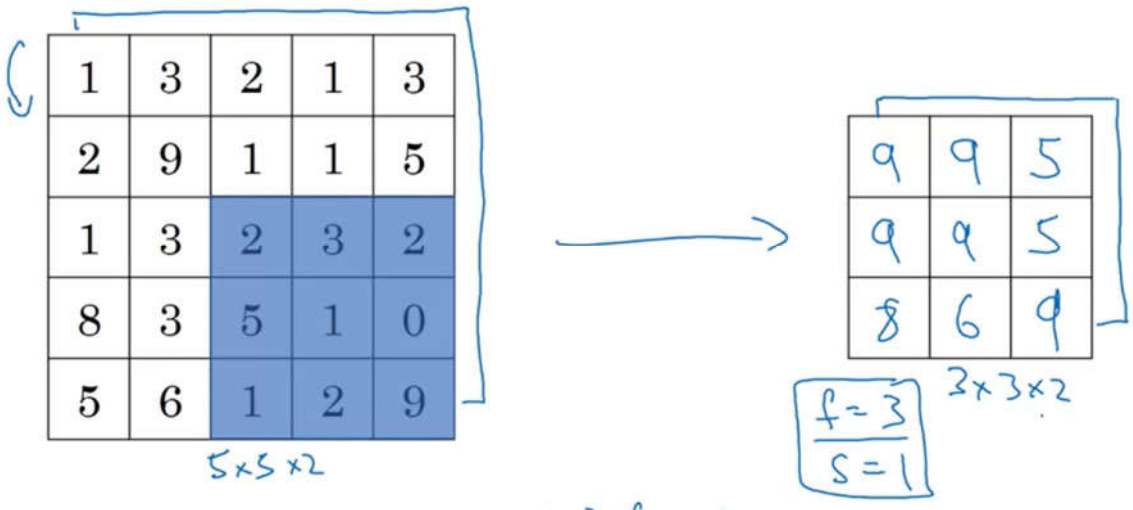
$s = 1$

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

So for example, if you have five by five by two,

Andrew Ng

Pooling layer: Max pooling

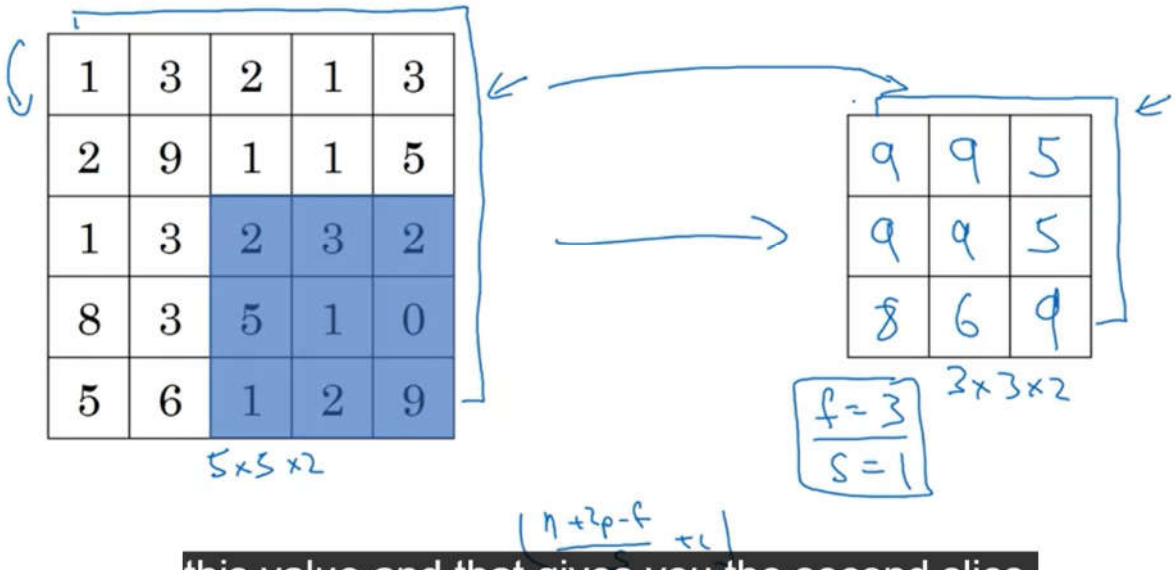


$\left(\frac{n+2p-f}{s}+1\right)$

then the output will be three by three by two and the way you compute

Andrew Ng

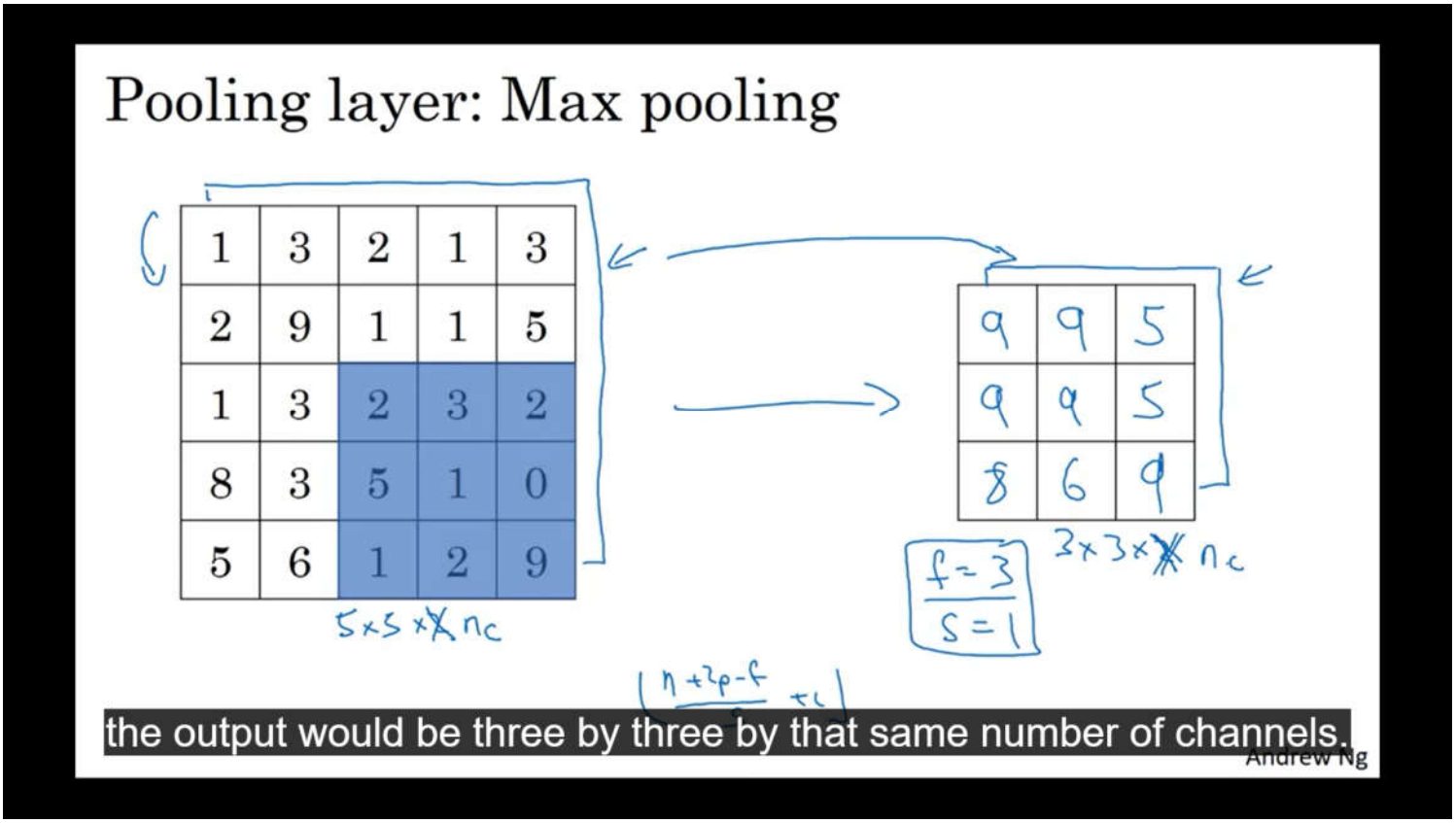
Pooling layer: Max pooling

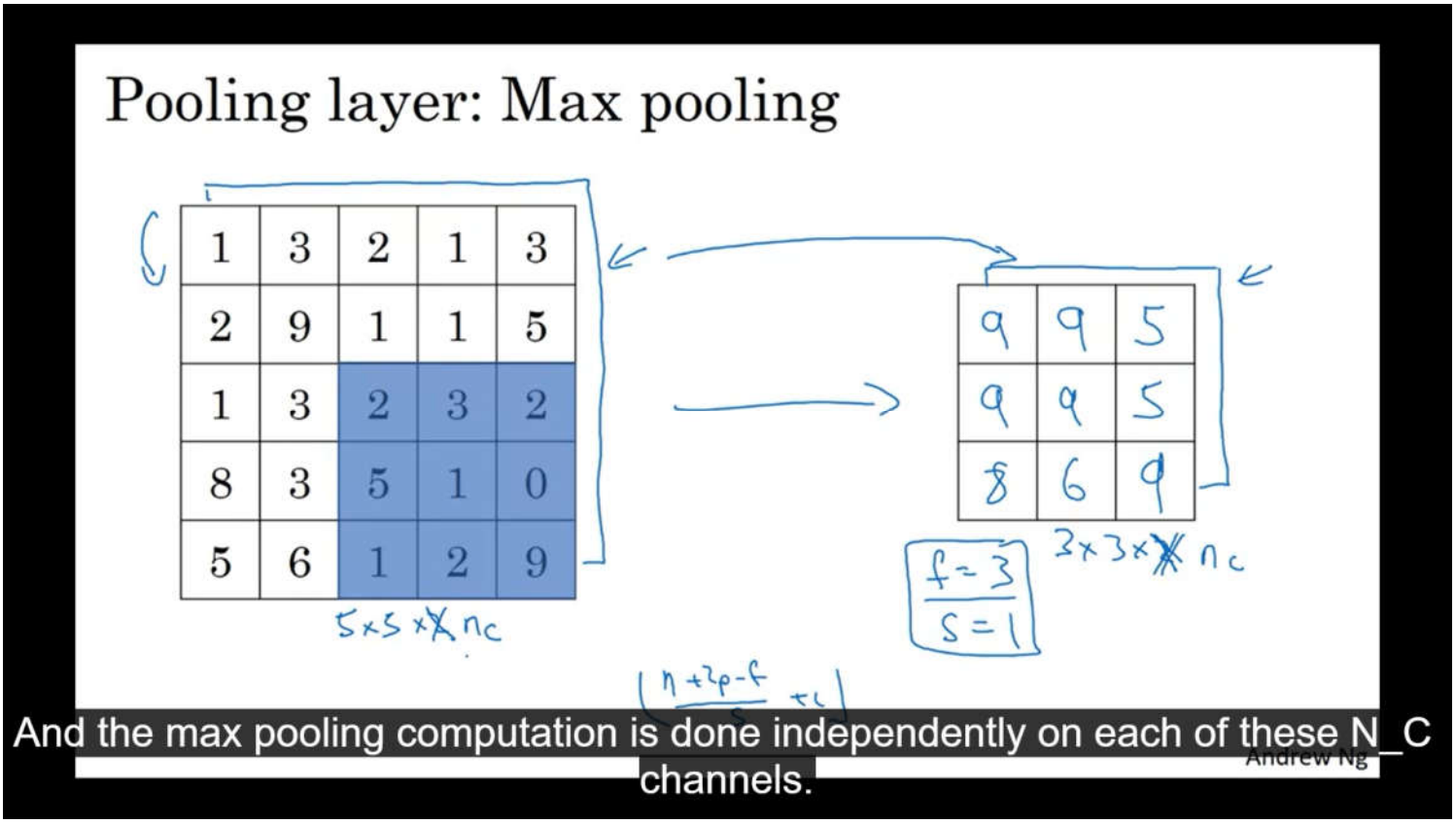


$\left(\frac{n+2p-f}{s}+1\right)$

this value and that gives you the second slice.

Andrew Ng


Pooling layer: Max pooling
1 3 2 1 3
2 9 1 1 5
1 3 2 3 2
8 3 5 1 0
5 6 1 2 9
9 9 5
9 9 5
8 6 9
f = 3
s = 1
the output would be three by three by that same number of channels.
Andrew Ng


Pooling layer: Max pooling
1 3 2 1 3
2 9 1 1 5
1 3 2 3 2
8 3 5 1 0
5 6 1 2 9
9 9 5
9 9 5
8 6 9
f = 3
s = 1
And the max pooling computation is done independently on each of these N_C channels.
Andrew Ng

Pooling layer: Average pooling

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

→

but I'll mention briefly which is average 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

and four and two.

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

$f = 2$
$s = 2$

s equals two, we can choose other hyperparameters as well.

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

$f = 2$
$s = 2$

You might use average pooling to collapse your representation from say,

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

So just to summarize,

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size $f=2, s=2$

s : stride

Max or average pooling

the height and width of the representation by a factor of two.

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

f=2, s=2

f=3, s=2

→ p: padding.

although there is one exception that we'll see next week as well.

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

f=2, s=2

f=3, s=2

~~→ p: padding.~~

usually, it does not use any padding.

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

$f=2, s=2$

$f=3, s=2$

~~p: padding~~

And the input of max pooling is that you input a volume of size that,

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

$f=2, s=2$

$f=3, s=2$

~~p: padding~~

$$n_H \times n_W \times n_C$$

$$\downarrow$$

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

So the number of input channels is equal to the number of output channels

Andrew Ng

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

$f=2, s=2$

$f=3, s=2$

~~p: padding~~

No parameters to learn!

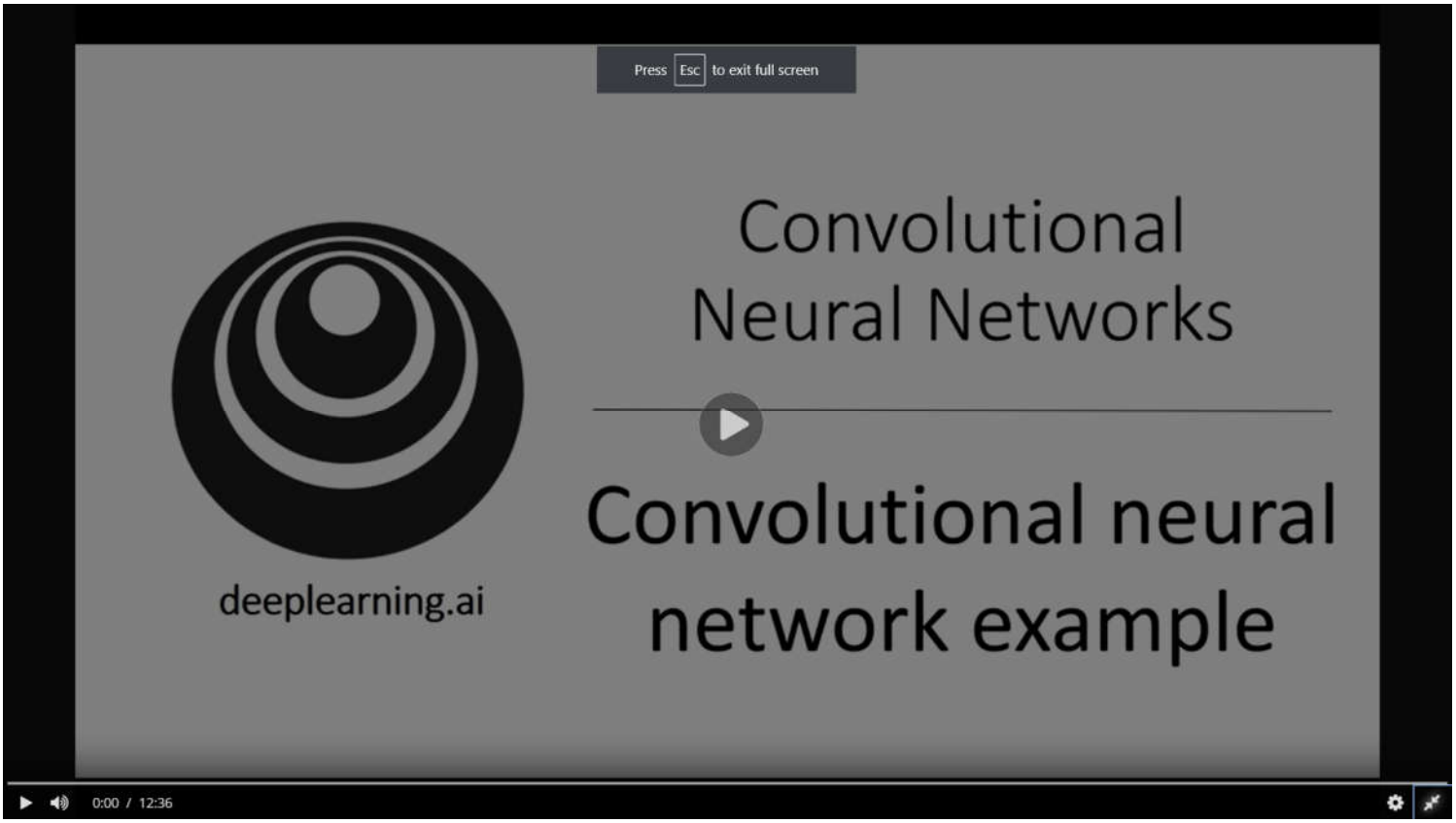
$$n_H \times n_W \times n_C$$

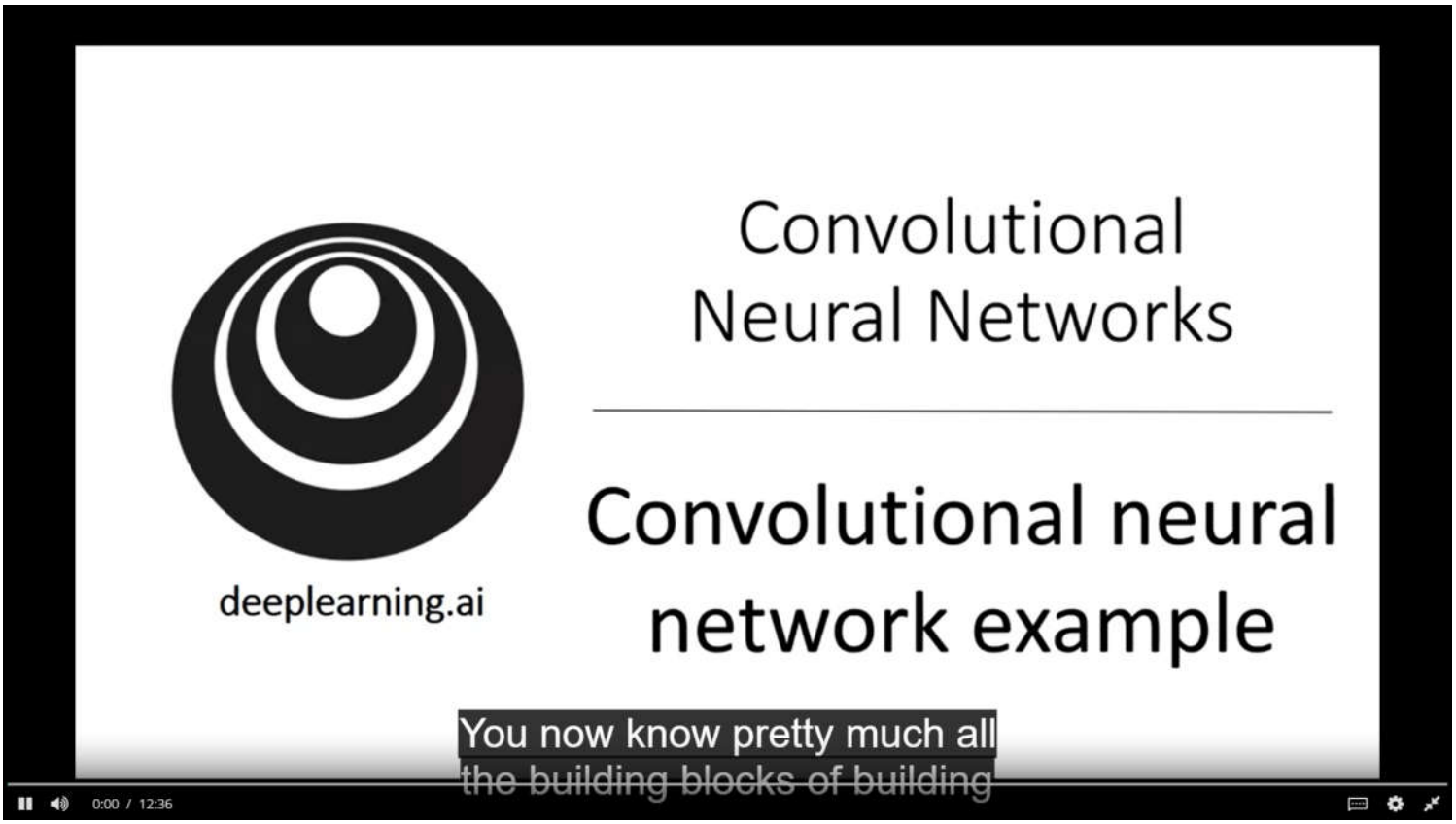
$$\downarrow$$

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

Its just a fixed function that the neural network computes in one of the layers,

Andrew Ng


Press Esc to exit full screen
Convolutional
Neural Networks
deeplearning.ai
Convolutional neural
network example
0:00 / 12:36


Convolutional
Neural Networks
deeplearning.ai
Convolutional neural
network example
You now know pretty much all
the building blocks of building
0:00 / 12:36

Neural network example

it's an RGB image and maybe you're trying to do handwritten digit recognition.

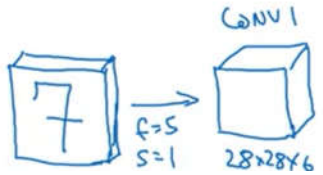
Andrew Ng

Neural network example

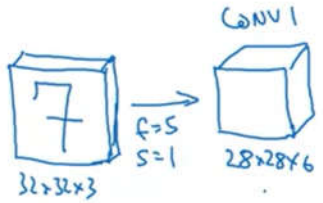
it's actually quite similar to one of the classic neural networks called LeNet-5,

Andrew Ng

Neural network example (LeNet-5)



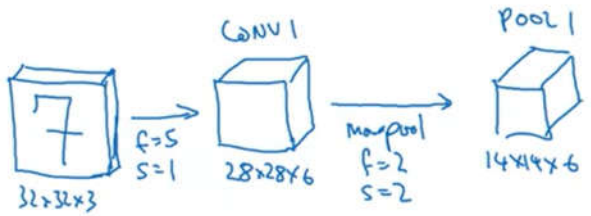
Neural network example (LeNet-5)



maybe a real non-linearity,
and that's the conv 1 output.

Andrew Ng

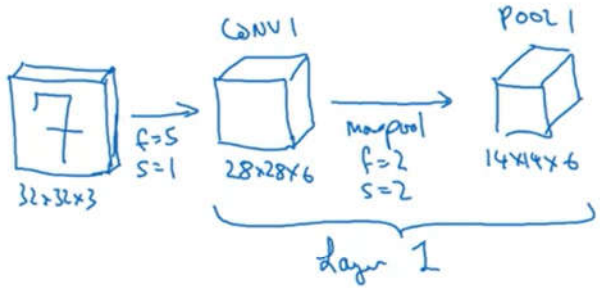
Neural network example (LeNet-5)



So, it turns out that in the literature
of a ConvNet there are two

Andrew Ng

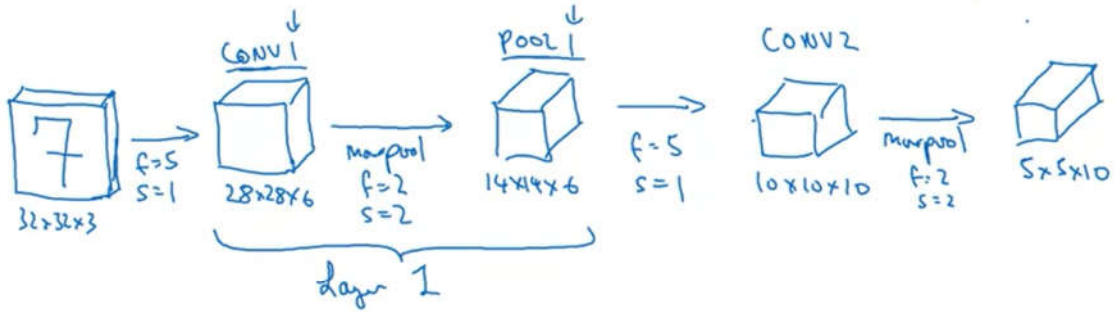
Neural network example (LeNet-5)



will be to call they convey layer as
a layer and the pool layer as a layer.

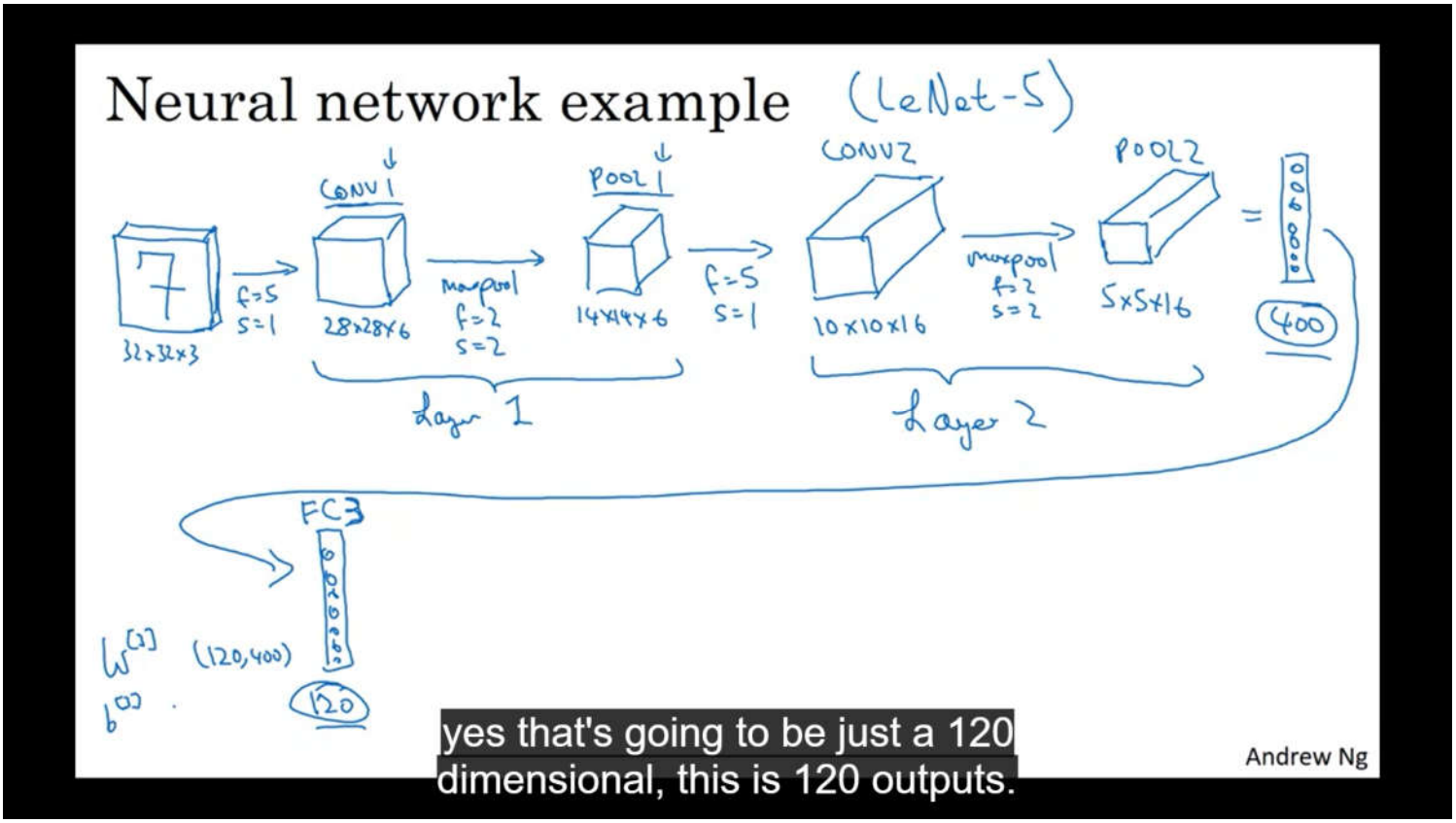
Andrew Ng

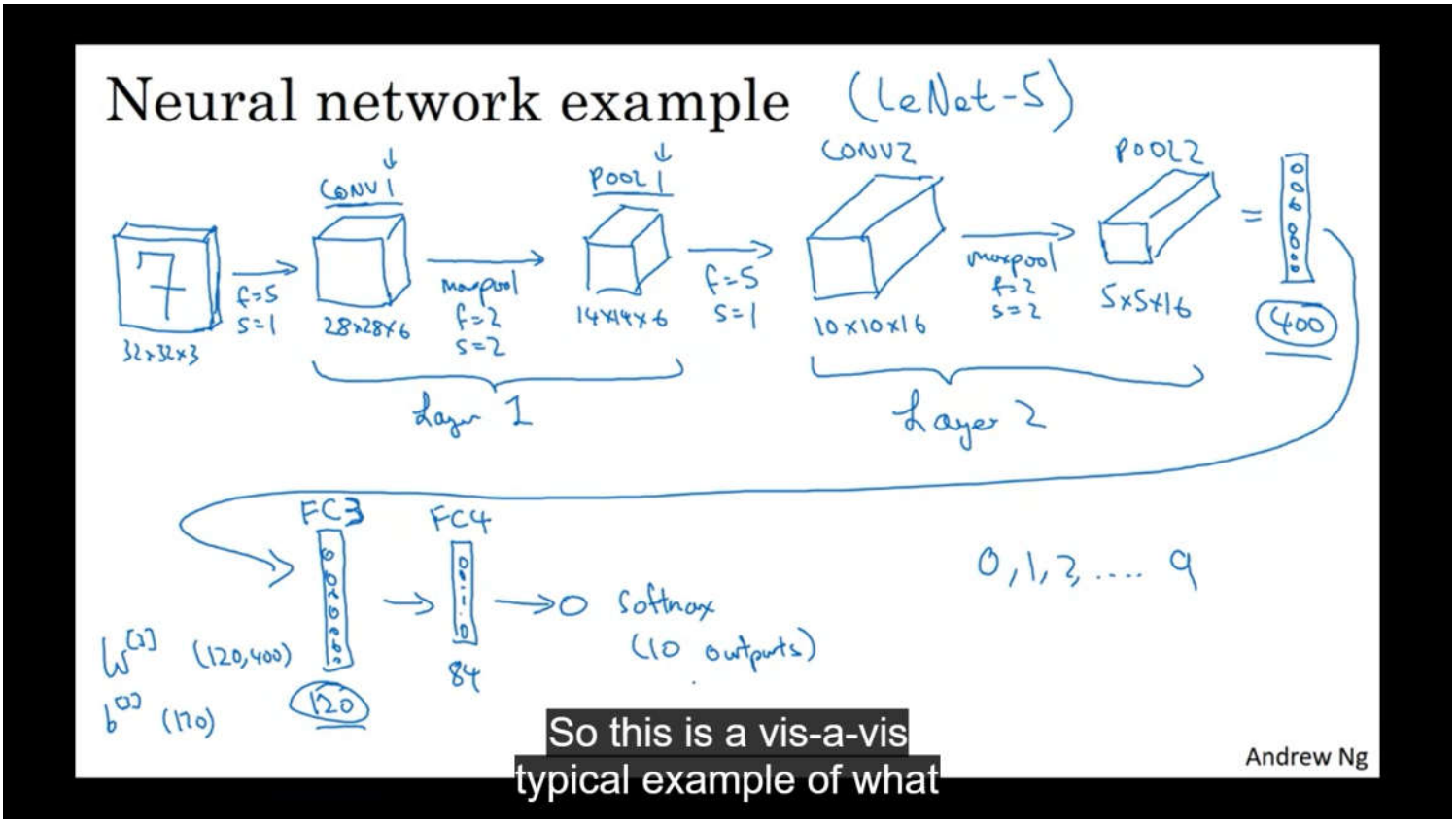
Neural network example (LeNet-5)



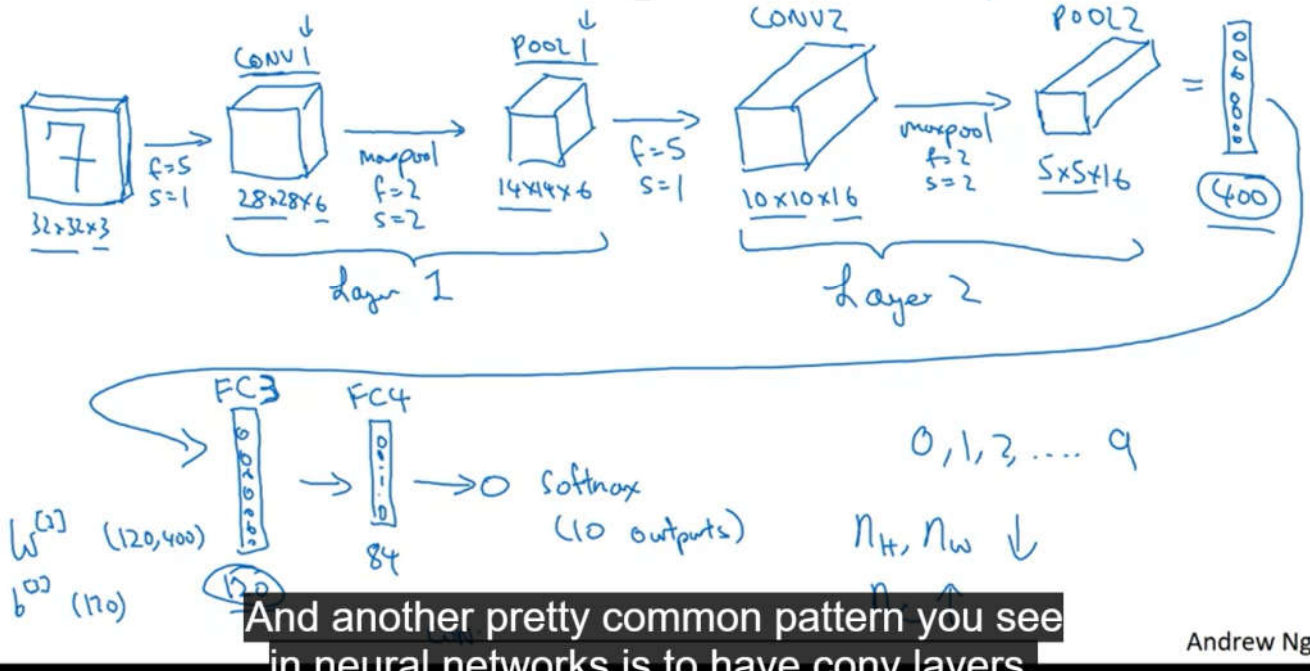
width by a factor of 2, so
you're left with 5 x 5 x 10.

Andrew Ng


Neural network example
(LeNet-5)
CONV1
POOL1
CONV2
POOL2
7
32x32x3
f=5
s=1
28x28x6
maxpool
f=2
s=2
14x14x6
f=5
s=1
10x10x16
maxpool
f=2
s=2
5x5x16
=
400
Layer 1
Layer 2
FC3
120
W[3] (120,400)
b[3]
yes that's going to be just a 120
dimensional, this is 120 outputs.
Andrew Ng


Neural network example
(LeNet-5)
CONV1
POOL1
CONV2
POOL2
7
32x32x3
f=5
s=1
28x28x6
maxpool
f=2
s=2
14x14x6
f=5
s=1
10x10x16
maxpool
f=2
s=2
5x5x16
=
400
Layer 1
Layer 2
FC3
FC4
84
0
Softmax
(10 outputs)
0, 1, 2, ... 9
120
W[3] (120,400)
b[3] (120)
So this is a vis-a-vis
typical example of what
Andrew Ng

Neural network example (LeNet-5)



And another pretty common pattern you see in neural networks is to have conv layers,

Andrew Ng

Neural network example

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

the activation size, and the number of parameters in this network.

Andrew Ng

Neural network example

	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	3,072 $a^{[0]}$	0

Well it's really 32 x 32 x 3.

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

the activation sizes of these different layers.

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

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

First, notice that the max pooling layers don't have any parameters.

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

maybe go down gradually as you go deeper in the neural network.

Andrew Ng

16 min

Video: Simple Convolutional Network Example
8 min

Reading: Simple Convolutional Network Example *CORRECTION*
1 min

Video: Pooling Layers
10 min

Video: CNN Example
12 min

Reading: CNN Example *CORRECTION*
1 min

Video: Why Convolutions?
9 min

Reading: Why Convolutions? *CORRECTION*
1 min

Practice questions

Quiz: The basics of ConvNets

Here is a screenshot of that slide:

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

Here are the 5 typos:

1. 208 should be (5*5*3 + 1) * 8 = 608

coursera.org/learn/convolutional-neural-networks/supplement/4WJgu/cnn-example-correction

coursera | Srinivasa Rao Dhanikonda

Convolutional Neural Networks > Week 1 > CNN Example *CORRECTION*

Prev | Next

16 min

Video: Simple Convolutional Network Example
8 min

Reading: Simple Convolutional Network Example *CORRECTION*
1 min

Video: Pooling Layers
10 min

Video: CNN Example
12 min

Reading: CNN Example *CORRECTION*
1 min

Video: Why Convolutions?
9 min

Reading: Why Convolutions? *CORRECTION*
1 min

Practice questions

Quiz: The basics of ConvNets

FC4	(84,1)	84	10,081
Softmax	(10,1)	10	841

Andrew Ng

Here are the 5 typos:

1. 208 should be (5*5*3 + 1) * 8 = 608

2. 416 should be (5*5*8 + 1) * 16 = 3216

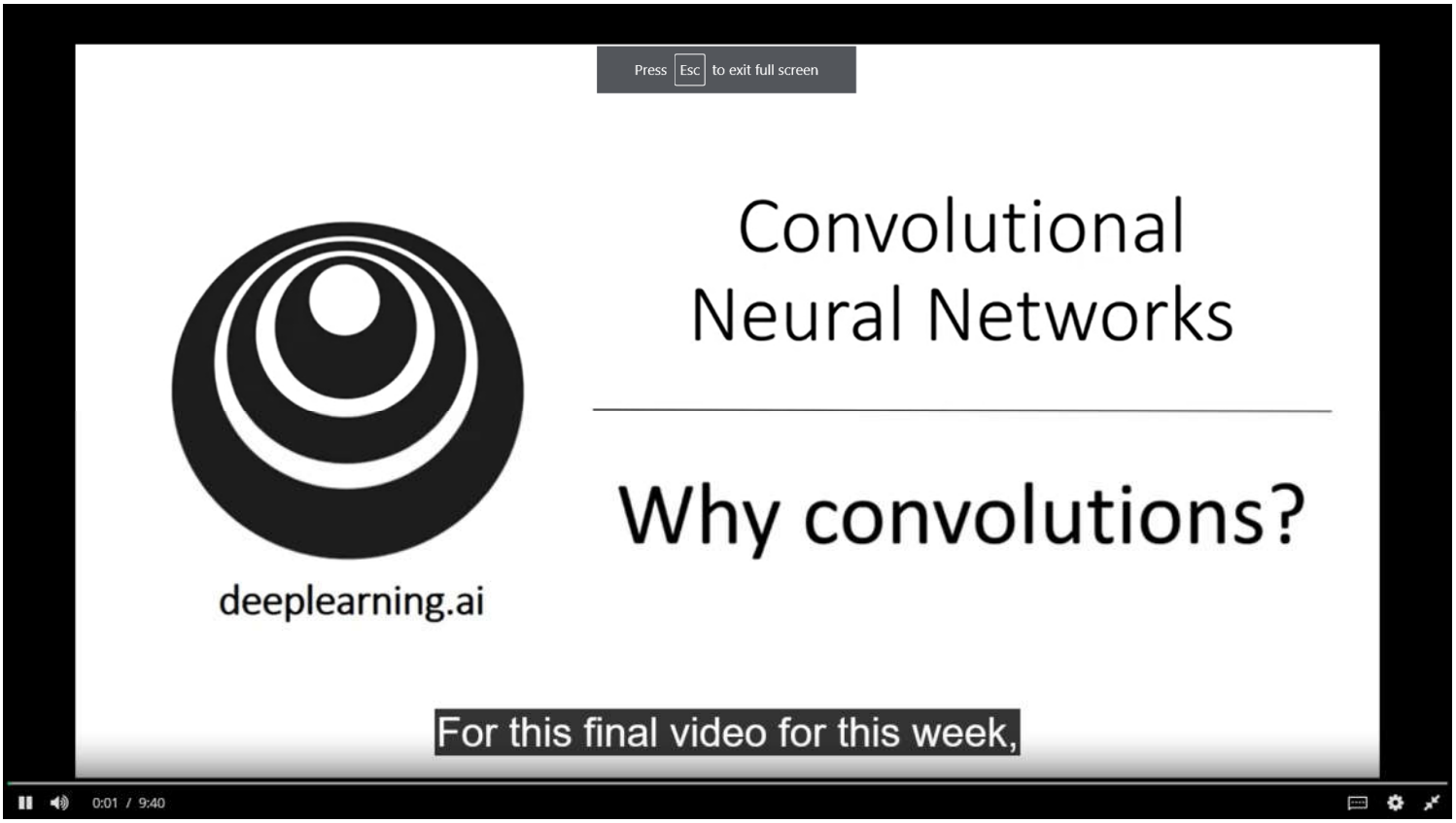
3. In the FC3, 48001 should be 400*120 + 120 = 48120, since the bias should have 120 parameters, not 1

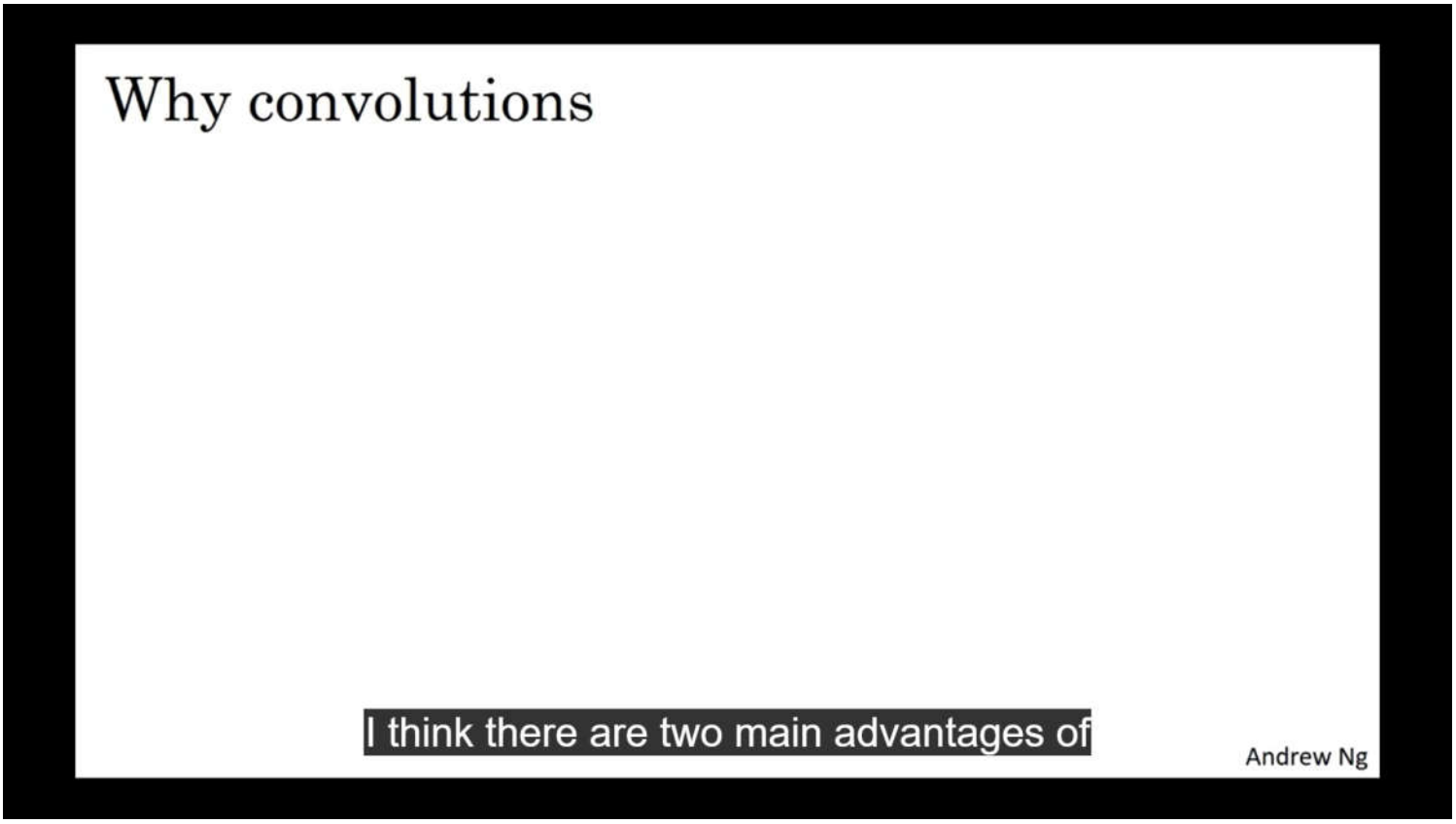
4. Similarly, in the FC4, 10081 should be 120*84 + 84 (not 1) = 10164

(Here, the bias is for the fully connected layer. In fully connected layers, there will be one bias for each neuron, so the bias become In FC3 there were 120 neurons so 120 biases.)

5. Finally, in the softmax, 841 should be 84*10 + 10 = 850

Mark as completed


Press Esc to exit full screen
Convolutional
Neural Networks
Why convolutions?
deeplearning.ai
For this final video for this week,
0:01 / 9:40

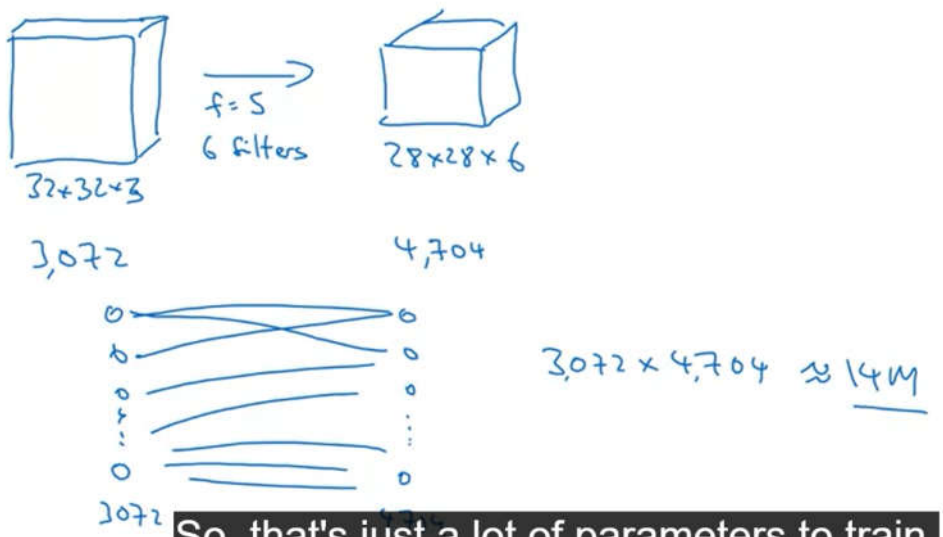

Why convolutions
I think there are two main advantages of
Andrew Ng

Why convolutions

And the advantages are parameter sharing and sparsity of connections.

Andrew Ng

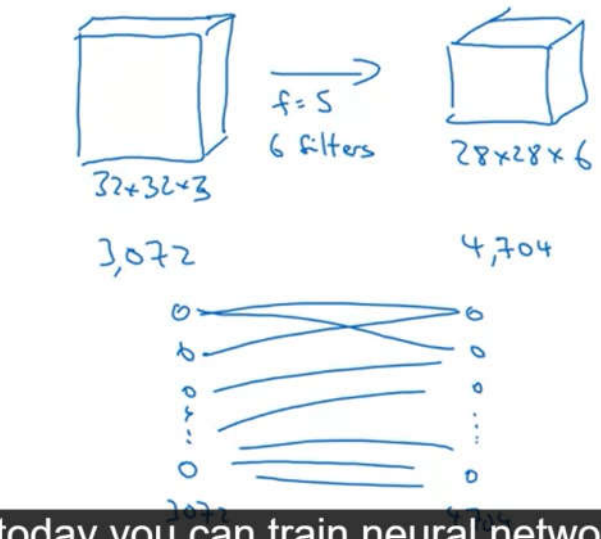
Why convolutions



So, that's just a lot of parameters to train.

Andrew Ng

Why convolutions

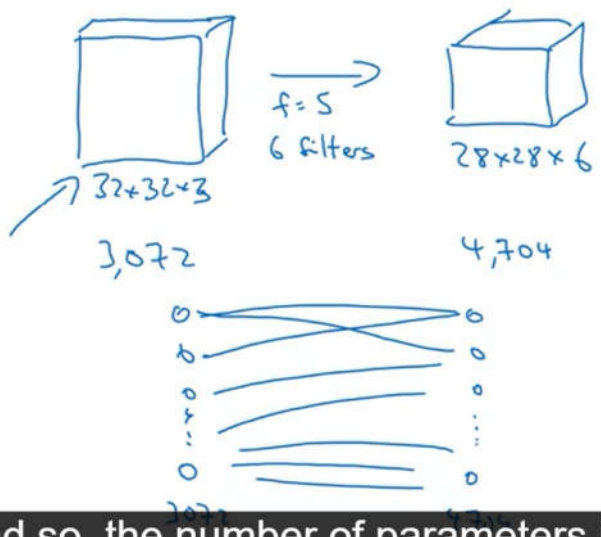


$3{,}072 \times 4{,}704 \approx 14M$

And today you can train neural networks with even more parameters than 14 million,

Andrew Ng

Why convolutions



$5 \times 5 = 25$

26

$6 \times 26 = 156$ parameters

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

And so, the number of parameters in this conv layer remains quite small.

Andrew Ng

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.

And parameter sharing is motivated by the observation

Andrew Ng

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.

Andrew Ng

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.

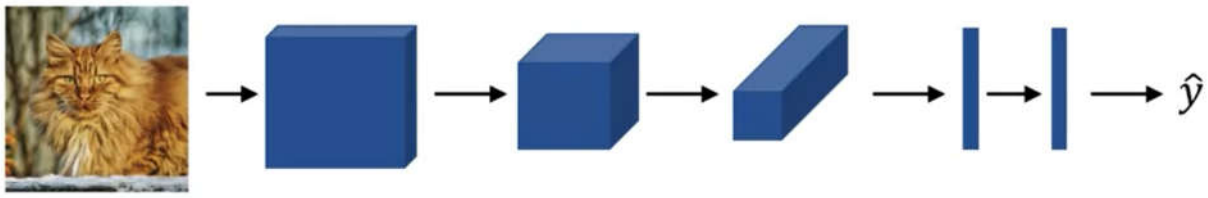
Sparsity of connections: In each layer, each output value de

a neural network has a lot fewer parameters which allows it

Andrew Ng

Putting it together

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

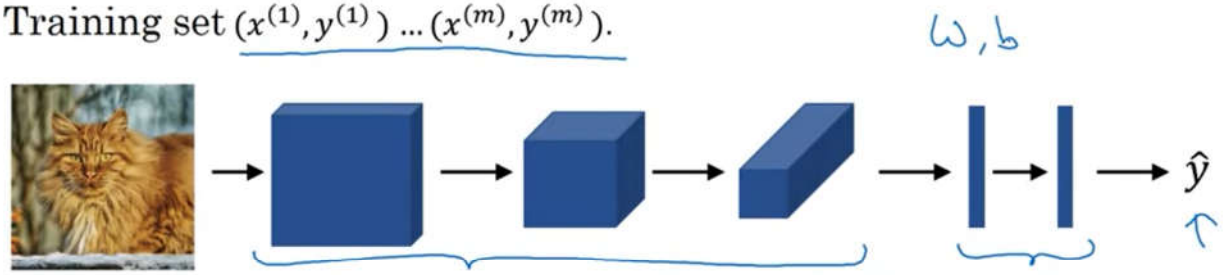


Finally, let's put it all together and see how you can train one of these networks.

Andrew Ng

Putting it together

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



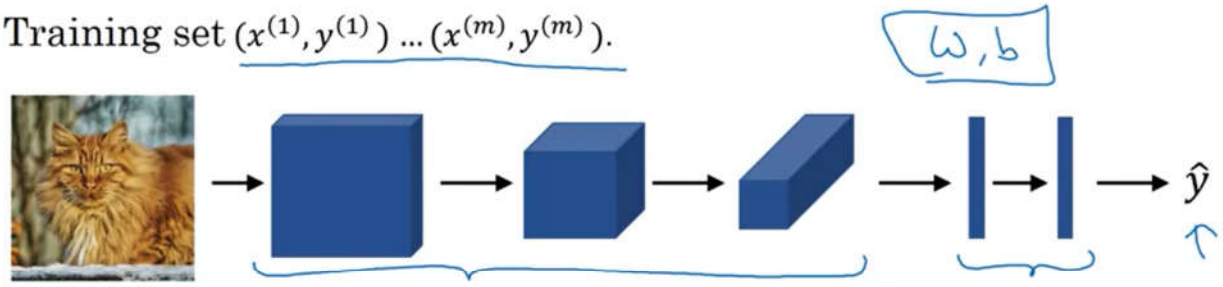
Cost $J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$

lets you define a cost function similar to what we have seen in the previous courses,

Andrew Ng

Putting it together

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

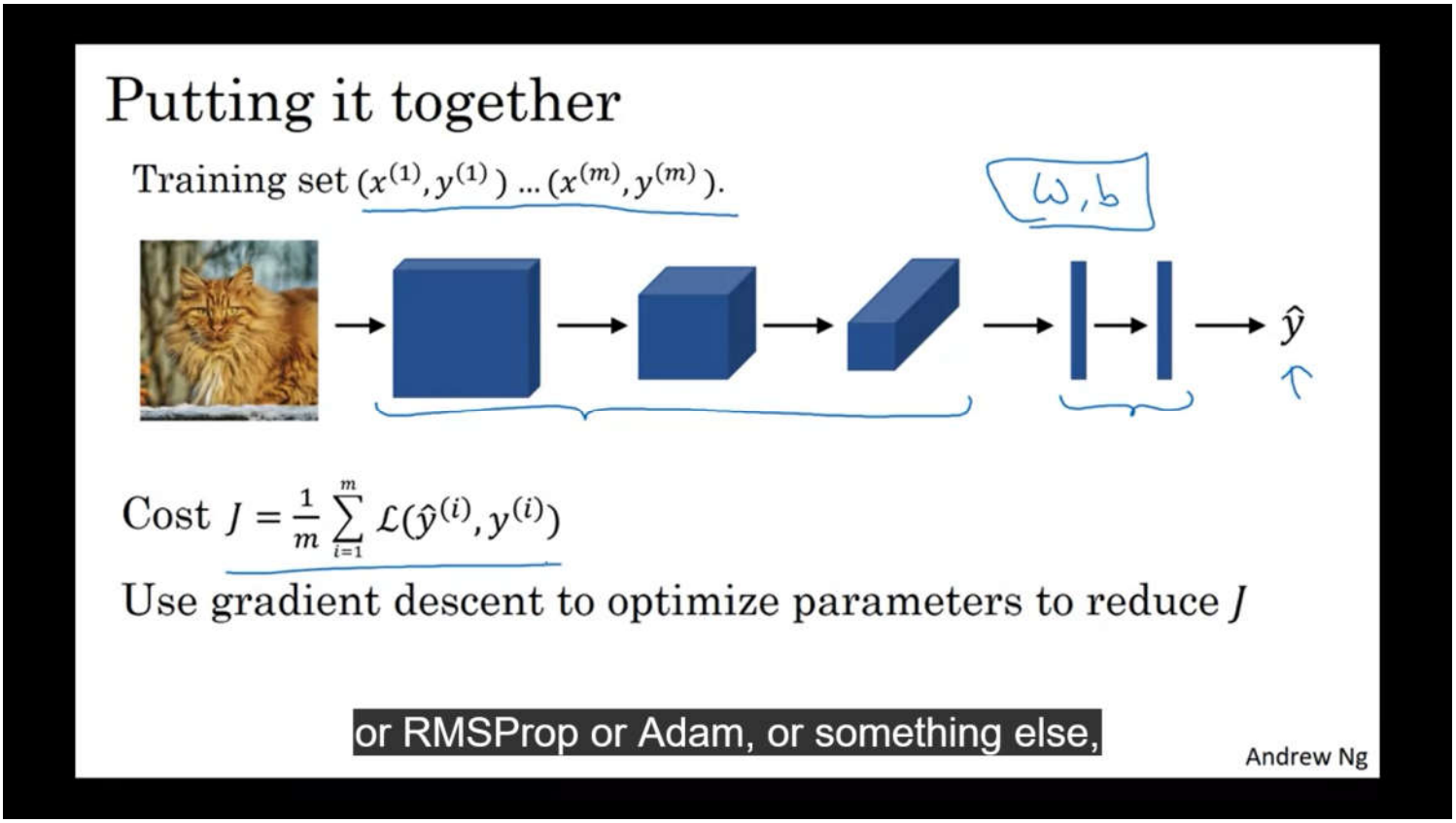


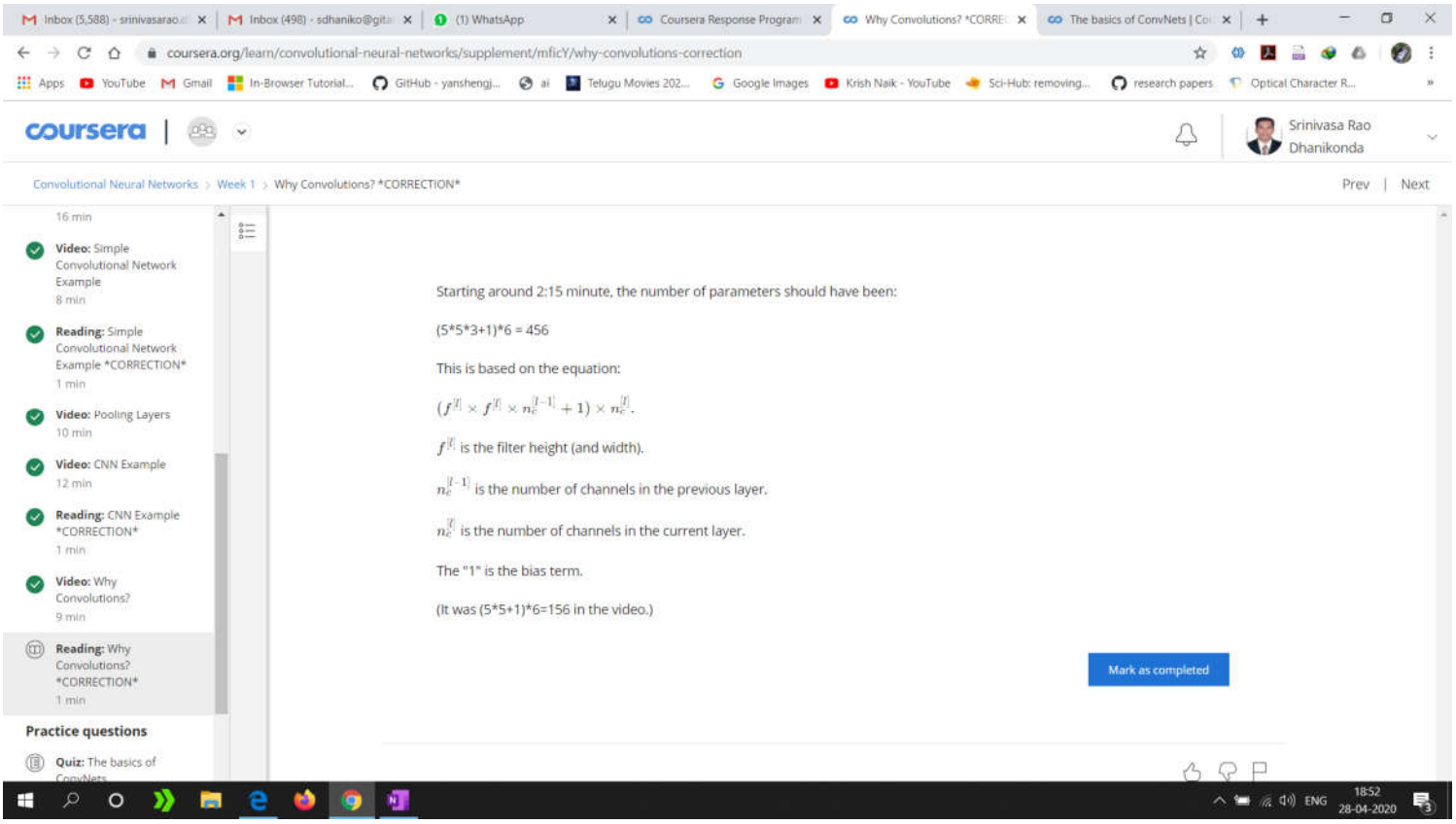
Cost $J = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$

Use gradient descent to optimize parameters to reduce J

all you need to do is then use gradient descents or some of

Andrew Ng


Putting it together
Training set $(x^{(1)}, y^{(1)}) \ldots (x^{(m)}, y^{(m)})$.
w, b
$\hat{y}$
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
or RMSProp or Adam, or something else,
Andrew Ng


coursera.org/learn/convolutional-neural-networks/supplement/mficY/why-convolutions-correction
coursera
Srinivasa Rao Dhanikonda
Convolutional Neural Networks > Week 1 > Why Convolutions? *CORRECTION*
Prev | Next
Video: Simple Convolutional Network Example 8 min
Reading: Simple Convolutional Network Example *CORRECTION* 1 min
Video: Pooling Layers 10 min
Video: CNN Example 12 min
Reading: CNN Example *CORRECTION* 1 min
Video: Why Convolutions? 9 min
Reading: Why Convolutions? *CORRECTION* 1 min
Practice questions
Quiz: The basics of ConvNets
Starting around 2:15 minute, the number of parameters should have been:
(5*5*3+1)*6 = 456
This is based on the equation:
$(f^{[l]} \times f^{[l]} \times n_c^{[l-1]} + 1) \times n_c^{[l]}$.
$f^{[l]}$ is the filter height (and width).
$n_c^{[l-1]}$ is the number of channels in the previous layer.
$n_c^{[l]}$ is the number of channels in the current layer.
The "1" is the bias term.
(It was (5*5+1)*6=156 in the video.)
Mark as completed

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets
Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

The basics of ConvNets

TOTAL POINTS 10

1. What do you think applying this filter to a grayscale image will do? (1 point)

$$\begin{bmatrix} 0 & 1 & -1 & 0 \\ 1 & 3 & -3 & -1 \\ 1 & 3 & -3 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

- ○ Detect image contrast
- ◉ Detect vertical edges
- ○ Detect 45 degree edges
- ○ Detect horizontal edges

2. Suppose your input is a 300 by 300 color (RGB) image, and you are not using a convolutional network. If the first hidden layer has 100 neurons, each one fully connected to the input, how many parameters does this hidden layer have (including the bias parameters)? (1 point)

- ○ 9,000,001
- ○ 9,000,100

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets
Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

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

- ○ Detect image contrast
- ◉ Detect vertical edges
- ○ Detect 45 degree edges
- ○ Detect horizontal edges

2. Suppose your input is a 300 by 300 color (RGB) image, and you are not using a convolutional network. If the first hidden layer has 100 neurons, each one fully connected to the input, how many parameters does this hidden layer have (including the bias parameters)? (1 point)

- ○ 9,000,001
- ○ 9,000,100
- ○ 27,000,001
- ◉ 27,000,100

3. Suppose your input is a 300 by 300 color (RGB) image, and you use a convolutional layer with 100 filters that are each 5x5. How many parameters does this hidden layer have (including the bias parameters)? (1 point)

- ○ 2501
- ○ 2600

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ○ 27,000,001
- ◉ 27,000,100

3. Suppose your input is a 300 by 300 color (RGB) image, and you use a convolutional layer with 100 filters that are each 5x5. How many parameters does this hidden layer have (including the bias parameters)? (1 point)

- ○ 2501
- ○ 2600
- ○ 7500
- ◉ 7600

4. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, using a stride of 2 and no padding. What is the output volume? (1 point)

- ○ 16x16x32
- ○ 16x16x16
- ○ 29x29x16
- ○ 29x29x32

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ○ 2600
- ○ 7500
- ◉ 7600

4. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, using a stride of 2 and no padding. What is the output volume? (1 point)

- ○ 16x16x32
- ○ 16x16x16
- ○ 29x29x16
- ◉ 29x29x32

5. You have an input volume that is 15x15x8, and pad it using "pad=2." What is the dimension of the resulting volume (after padding)? (1 point)

- ○ 17x17x10
- ○ 17x17x8
- ○ 19x19x12
- ○ 19x19x8

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ○ 16x16x16
- ○ 29x29x16
- ◉ 29x29x32

5. You have an input volume that is 15x15x8, and pad it using "pad=2." What is the dimension of the resulting volume (after padding)? (1 point)

- ○ 17x17x10
- ○ 17x17x8
- ○ 19x19x12
- ◉ 19x19x8

6. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, and stride of 1. You want to use a "same" convolution. What is the padding? (1 point)

- ○ 1
- ○ 2
- ○ 3
- ○ 7

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ◉ 19x19x8

6. You have an input volume that is 63x63x16, and convolve it with 32 filters that are each 7x7, and stride of 1. You want to use a "same" convolution. What is the padding? (1 point)

- ○ 1
- ○ 2
- ◉ 3
- ○ 7

7. You have an input volume that is 32x32x16, and apply max pooling with a stride of 2 and a filter size of 2. What is the output volume? (1 point)

- ○ 32x32x8
- ○ 16x16x16
- ○ 16x16x8
- ○ 15x15x16

8. Because pooling layers do not have parameters, they do not affect the backpropagation (derivatives) (1 point)

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ○ 2
- ◉ 3
- ○ 7

7. You have an input volume that is 32x32x16, and apply max pooling with a stride of 2 and a filter size of 2. What is the output volume? (1 point)

- ○ 32x32x8
- ◉ 16x16x16
- ○ 16x16x8
- ○ 15x15x16

8. Because pooling layers do not have parameters, they do not affect the backpropagation (derivatives) calculation. (1 point)

- ○ True
- ○ False

9. In lecture we talked about "parameter sharing" as a benefit of using convolutional networks. Which of the following statements about parameter sharing in ConvNets are true? (Check all that apply.) (1 point)

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets

Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

- ○ 32x32x8
- ◉ 16x16x16
- ○ 16x16x8
- ○ 15x15x16

8. Because pooling layers do not have parameters, they do not affect the backpropagation (derivatives) calculation. (1 point)

- ○ True
- ◉ False

9. In lecture we talked about "parameter sharing" as a benefit of using convolutional networks. Which of the following statements about parameter sharing in ConvNets are true? (Check all that apply.) (1 point)

- ☐ It reduces the total number of parameters, thus reducing overfitting.
- ☐ It allows parameters learned for one task to be shared even for a different task (transfer learning).
- ☐ It allows a feature detector to be used in multiple locations throughout the whole input image/input volume.

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt

The basics of ConvNets
Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

9. In lecture we talked about "parameter sharing" as a benefit of using convolutional networks. Which of the following statements about parameter sharing in ConvNets are true? (Check all that apply.) — 1 point

- [x] It reduces the total number of parameters, thus reducing overfitting.
- [ ] It allows parameters learned for one task to be shared even for a different task (transfer learning).
- [x] It allows a feature detector to be used in multiple locations throughout the whole input image/input volume.
- [ ] It allows gradient descent to set many of the parameters to zero, thus making the connections sparse.

10. In lecture we talked about "sparsity of connections" as a benefit of using convolutional layers. What does this mean? — 1 point

- (●) Each activation in the next layer depends on only a small number of activations from the previous layer.
- ( ) Regularization causes gradient descent to set many of the parameters to zero.
- ( ) Each filter is connected to every channel in the previous layer.
- ( ) Each layer in a convolutional network is connected only to two other layers

coursera.org/learn/convolutional-neural-networks/exam/Nugx8/the-basics-of-convnets/attempt?redirectToCover=true

The basics of ConvNets
Graded Quiz • 30 min

Due May 4, 1:29 PM +0630

Congratulations! You passed!
TO PASS 80% or higher

Keep Learning

GRADE
100%

The basics of ConvNets

LATEST SUBMISSION GRADE
100%

1. What do you think applying this filter to a grayscale image will do? — 1 / 1 point

$$\begin{bmatrix} 0 & 1 & -1 & 0 \\ 1 & 3 & -3 & -1 \\ 1 & 3 & -3 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

✓ Correct

2. Suppose your input is a 300 by 300 color (RGB) image, and you are not using a convolutional network. If the first hidden layer has 100 neurons, each one fully connected to the input, how many parameters does this hidden layer have (including the bias parameters)? — 1 / 1 point