Convolutional Neural Networks

Pooling layers

deeplearning.ai

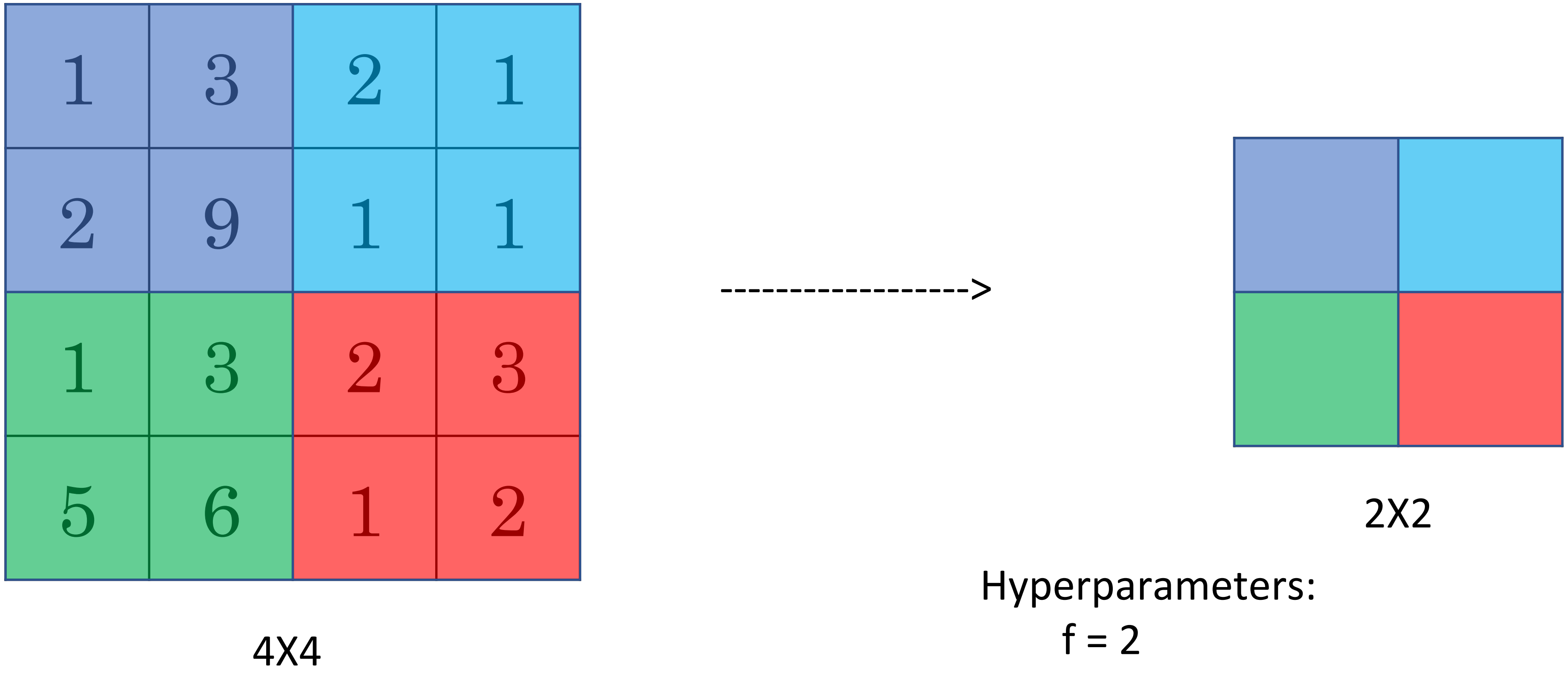
Other than convolutional layers, ConvNets often use pooling layers to reduce the size of their representation to speed up computation, as well as to make some of the features it detects a bot more robust

Pooling layer: Max pooling

So what the max operator does is really says, if this feature is detected anywhere in this filter, then keep a high number. But if this feature is not detected, so maybe this features doesn't exist in the upper right hand quadrant, then the max of all those number is still itself quite small. So that's the intuition behind max pooling.

But I have to admit, I think the main reason people use max pooling is because it's been found in a lot of experiments to work well.

Upper left hand quadrant



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

Hyperparameters:

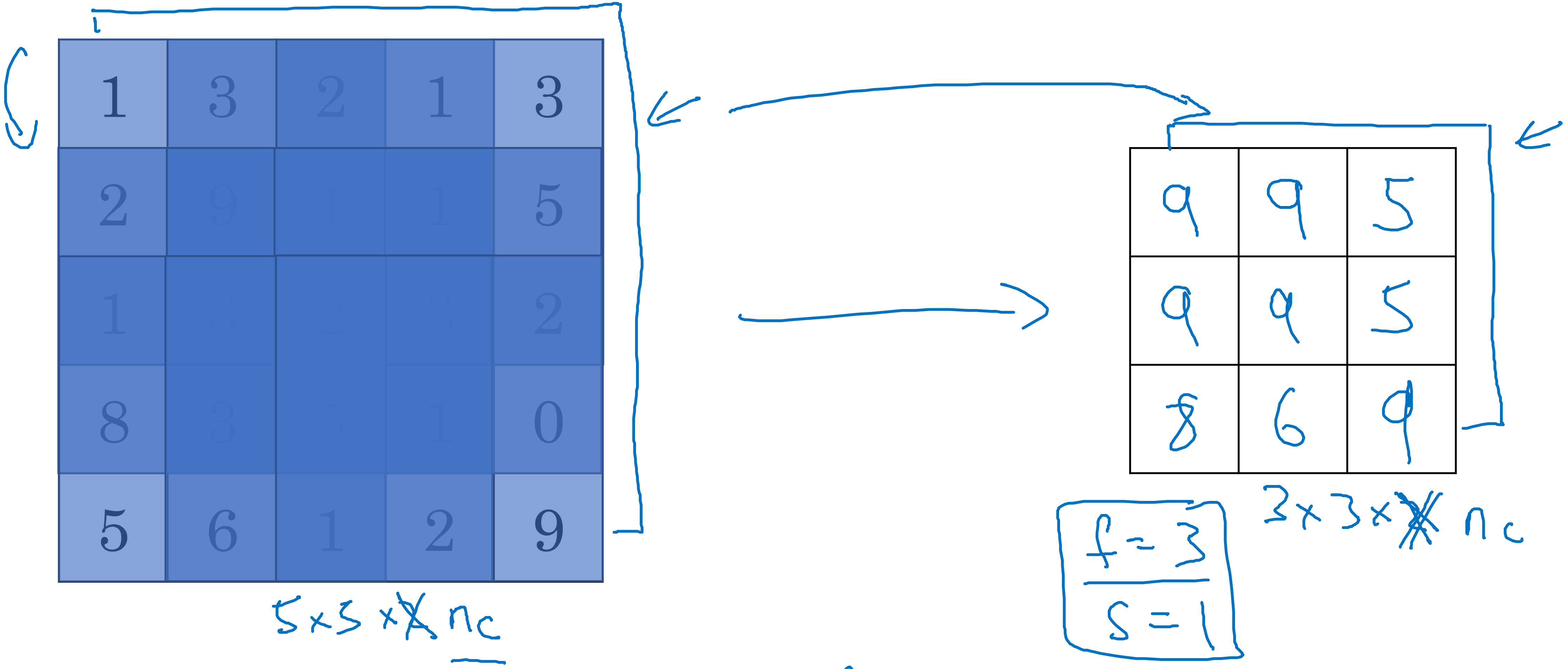
f = 2

s = 2

No parameters.!

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

$5 \times 5 \times \cancel{1}\, n_c$

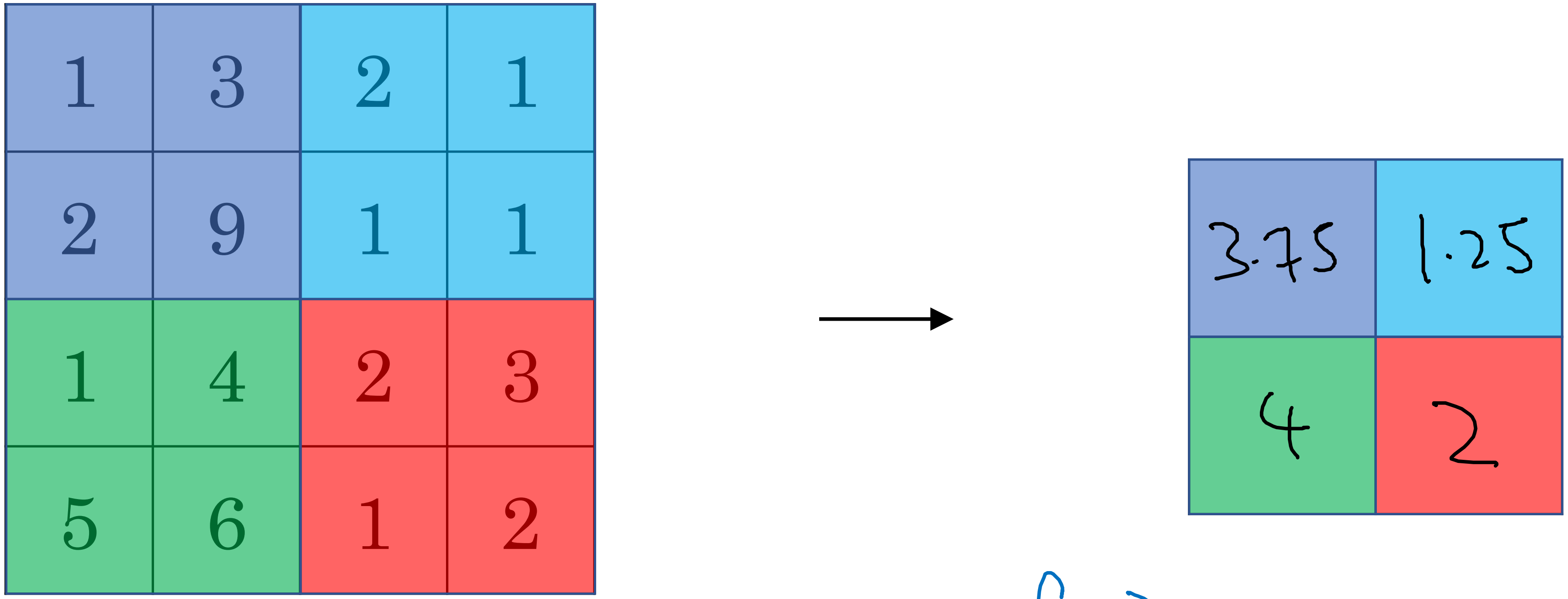
9	9	5
9	9	5
8	6	9

$3 \times 3 \times \cancel{1}\, n_c$

$\boxed{\begin{array}{c} f = 3 \\ \hline s = 1 \end{array}}$

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

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, 7X7X1000. And average over all the spacial extents to get 1X1X1000.

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

Andrew Ng

Summary of pooling

Hyperparameters:

And this has the effect of roughly shrinking the height and width by a factor of about two.

$f=2, s=2$

$f=3, s=2$

f : filter size

s : stride

Max or average pooling

$$\underline{n_H \times n_W \times n_C}$$

$$\downarrow$$

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

→ ~~p: padding~~

But for the most parts, max pooling usually does not use any padding.

No parameters to learn!

There are no parameters that backprop will adapt to max pooling. Instead there are just these hyperparameters that you set once maybe set once by hand or set using cross validation. And beyond that, you're done. It's just a fixed function that the neural network computes in one of the layers. And there is actually nothing to learn, it's just a fixed function.