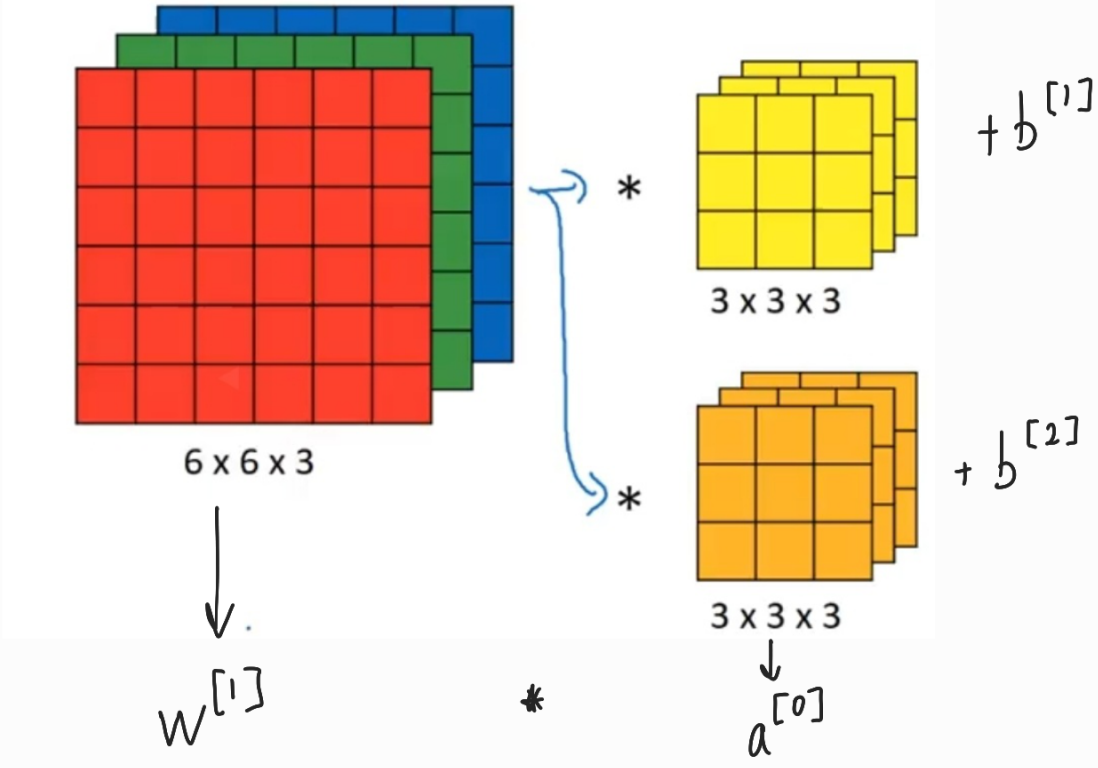
Example of a layer



Number of parameters by layer : numb_filters * (f_d1 x f_d2 x f_d3 + 1)

* CNNs are less prone to overfitting, because they use same parameters for different size images.

Summary of notation

If layer l is a convolution layer:

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

$p^{[l]}$ = padding

$s^{[l]}$ = stride

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

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

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

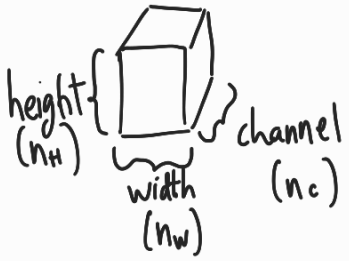
Input: $n_H^{[l-1]} \times n_W^{[l-1]} \times n_c^{[l-1]}$ ←

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

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

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

bias: $n_c^{[l]} - (1,1,1,n_c^{[l]})$ ← #filters in layer l.

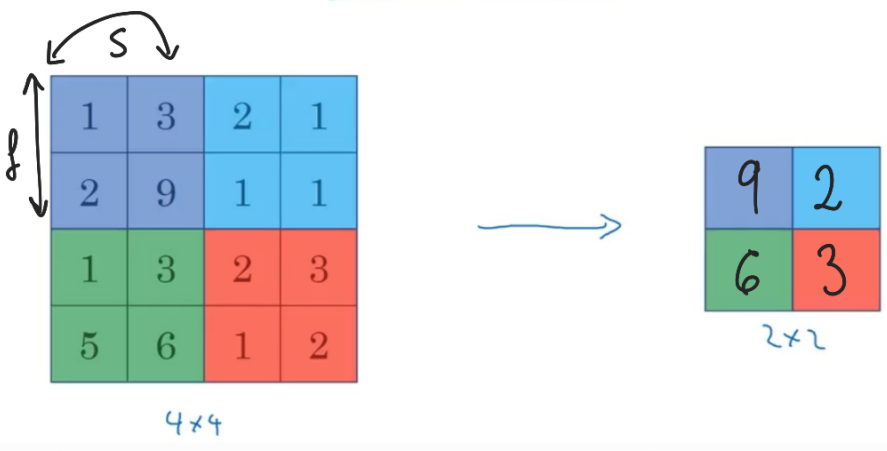


Types of layer in a convolutional network:

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

POOLING

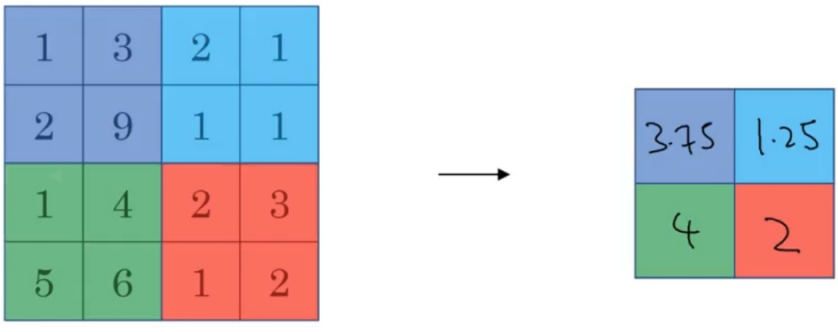
Pooling layer: Max pooling



Maximum number of each region.

$f=2$
$s=2$

Pooling layer: Average pooling



Mean of each region.

$f=2$, $s=2$

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

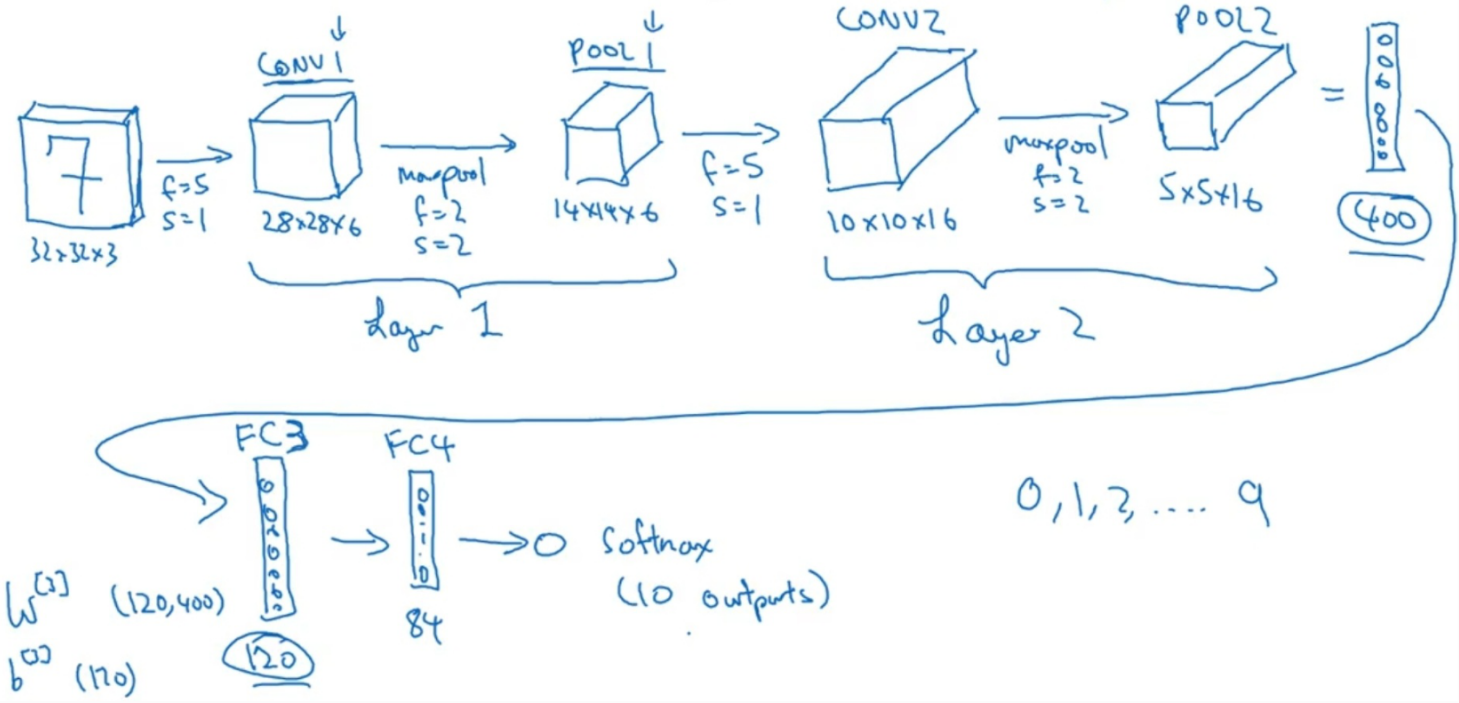
$n_H \times n_w \times n_c$

$\downarrow$

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

$\times n_c$

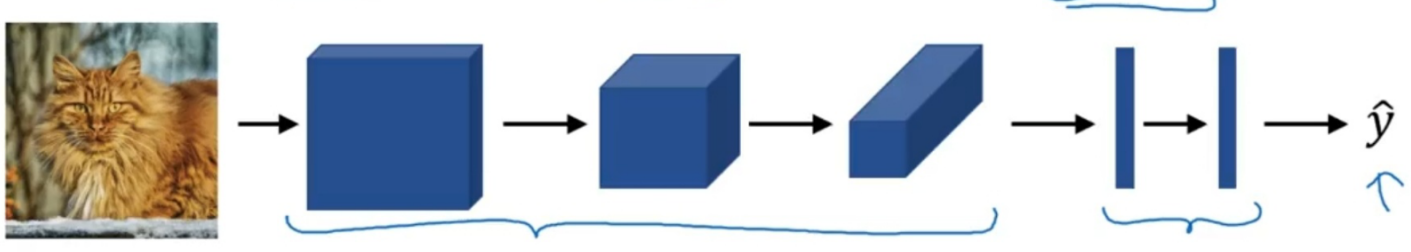
Neural network example (LeNet-5)



Usually, as it gets larger, we see this pattern ⇒ $n_H, n_W \downarrow$ $n_C \uparrow$

Putting it together

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



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

Reminder:

The formulas relating the output shape of the convolution to the input shape are:

$$n_H = \left\lfloor \frac{n_{H_{prev}} - f + 2 \times pad}{stride} \right\rfloor + 1$$

$$n_W = \left\lfloor \frac{n_{W_{prev}} - f + 2 \times pad}{stride} \right\rfloor + 1$$

n_C = number of filters used in the convolution

Computational cost = #filter params * #filter positions * # of filters