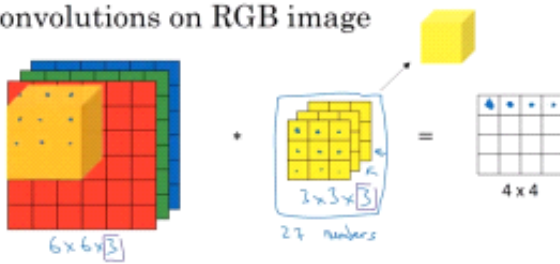
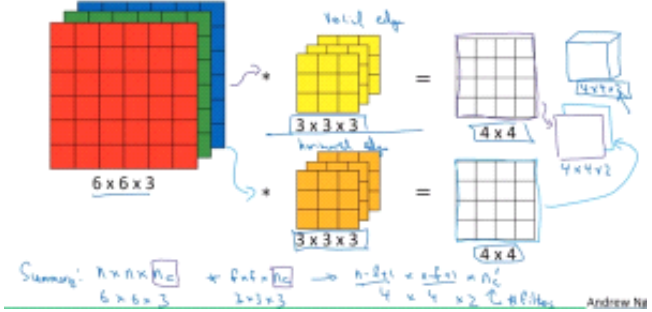


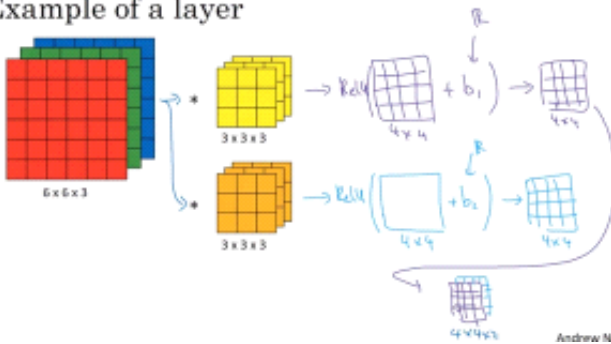
Convolutions on RGB image



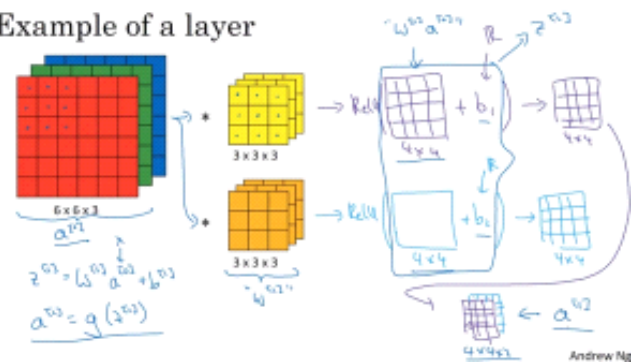
Multiple filters



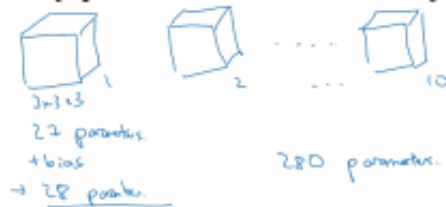
Example of a layer



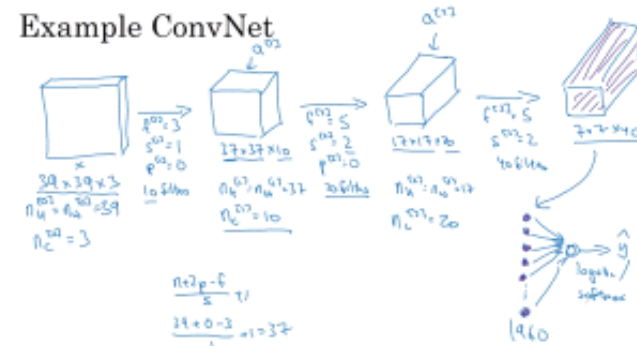
Example of a layer



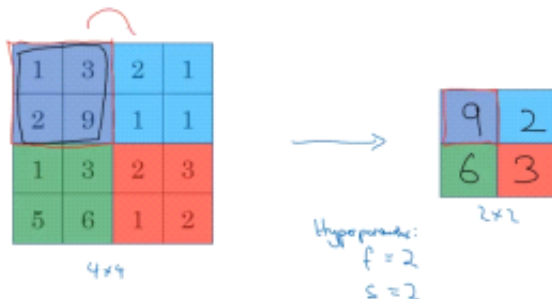
If you have 10 filters that are 3 x 3 x 3 in one layer of a neural network, how many parameters does that layer have?



Example ConvNet



Pooling layer: Max pooling



Pooling layer: Average pooling

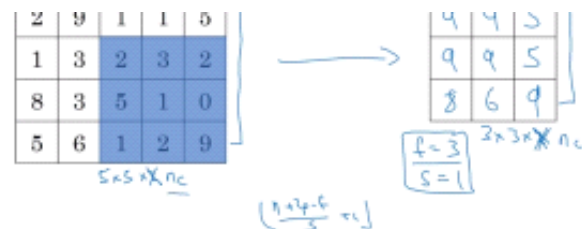
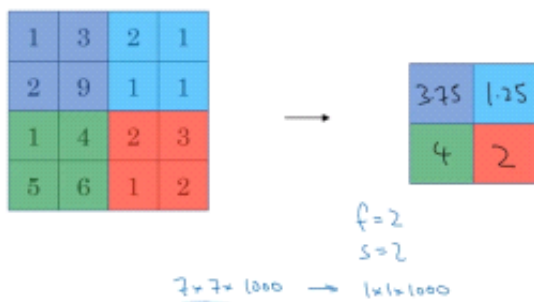
Types of layer in a convolutional network:

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

Pooling layer: Max pooling



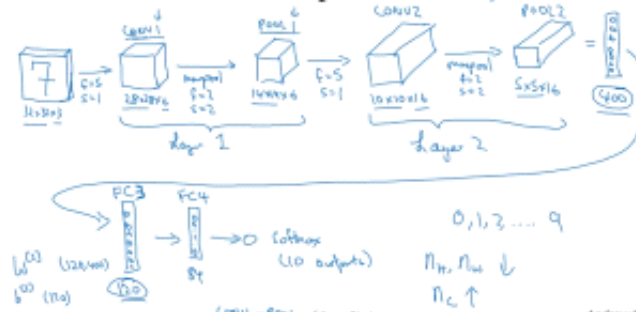
Pooling layer: Average pooling



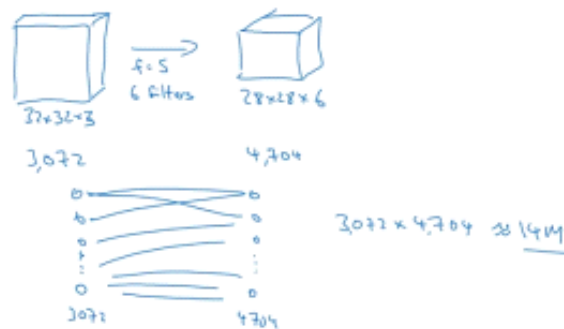
Neural network example

	Activation shape	Activation Size	# parameters
Input:	(32,32,3)	3,072	0
CONV1 (f=5, s=1)	(28,28,8)	6,272	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

Neural network example (LeNet-5)



Why CNN



Convolutions has less parameters as parameter sharing is done

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

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

Putting it together

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



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

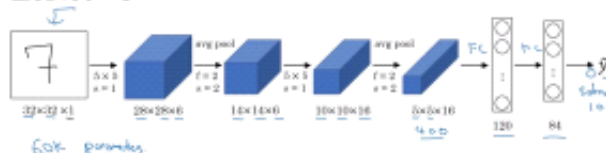
Use gradient descent to optimize parameters to reduce J

Classic networks:

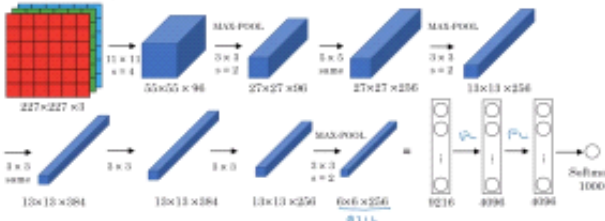
- LeNet-5 ←
- AlexNet ←
- VGG ←

RESNET
INCEPTION

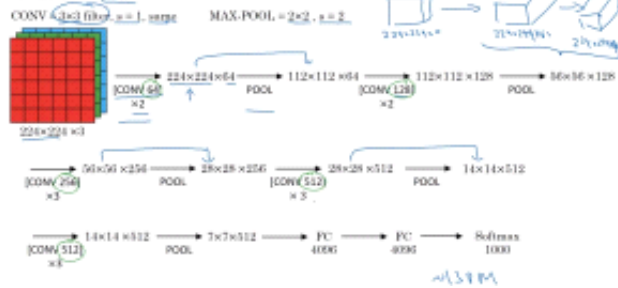
LeNet - 5



AlexNet

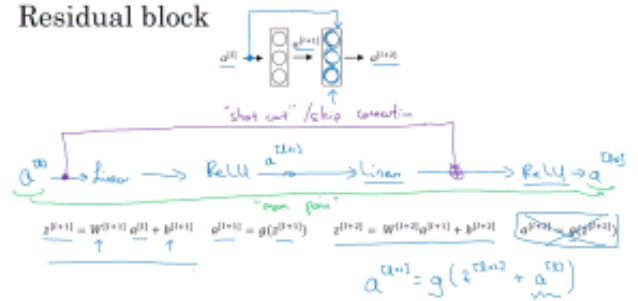


VGG - 16



RESNET

Residual block



ResNet

