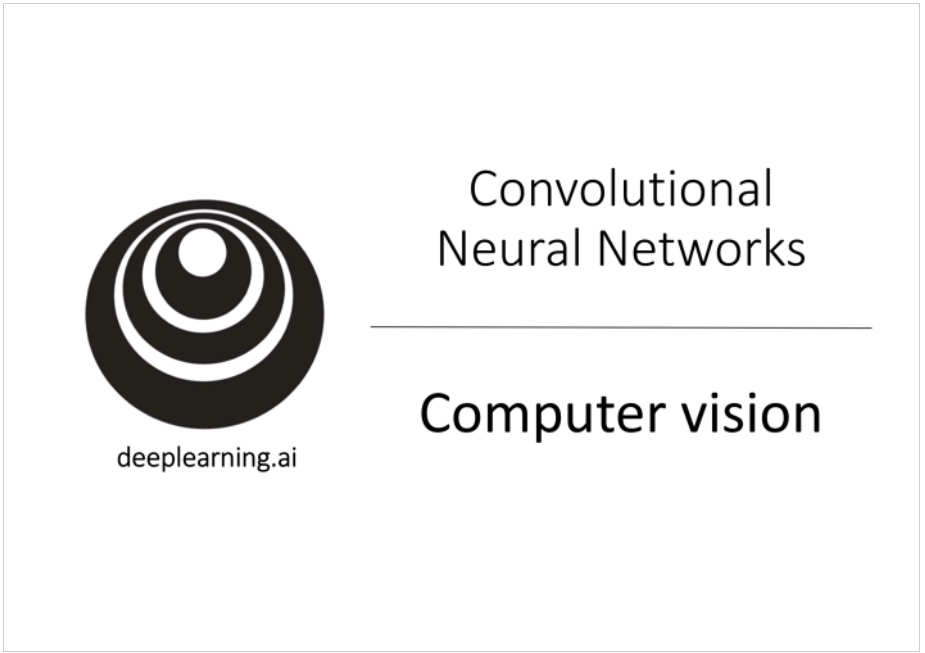
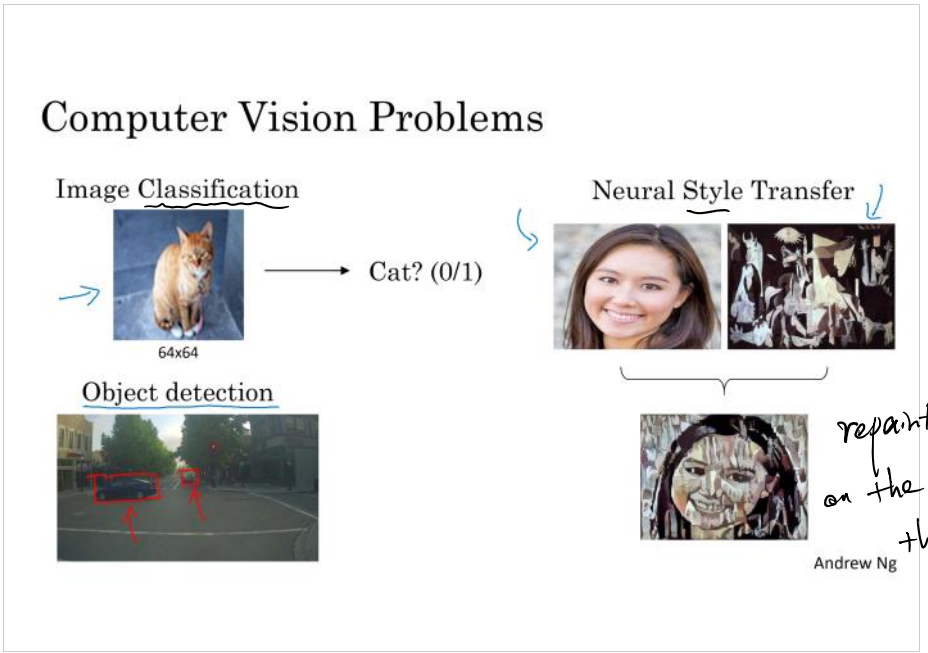
CNN

Monday, June 11, 2018 7:58 AM

intro



The literature is creative can find inspiration from CV for other algorithms.



repaint the image on the left with the style on the right

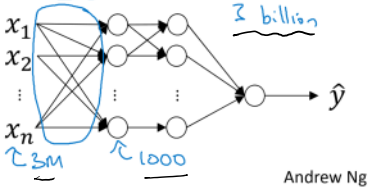
Deep Learning on large images

64x64 × 3

Cat? (0/1)

12288 ← small images.

1000 × 1000 × 3 = 3 million !



Andrew Ng

The input can get large

1000 × 3mil

3 billion parameters.

- hard to get enough data to prevent over-fitting
- Don't want to be stuck with small images

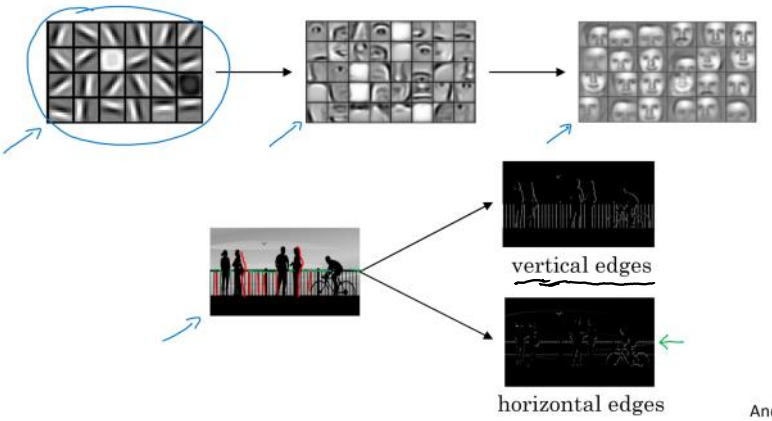
edge detect

deeplearning.ai

Convolutional Neural Networks

Edge detection example

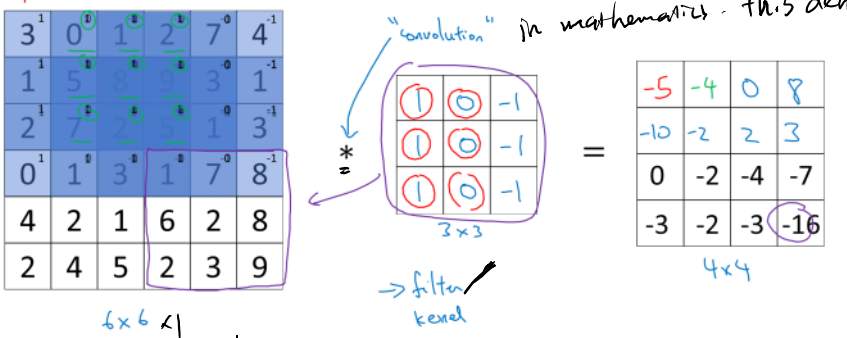
Computer Vision Problem



Andrew Ng

Vertical edge detection

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



6×6 ×1

grayscale

no separate RGB channels.

"convolution" in mathematics - this denote convolution. But in Python it also means multiplication.

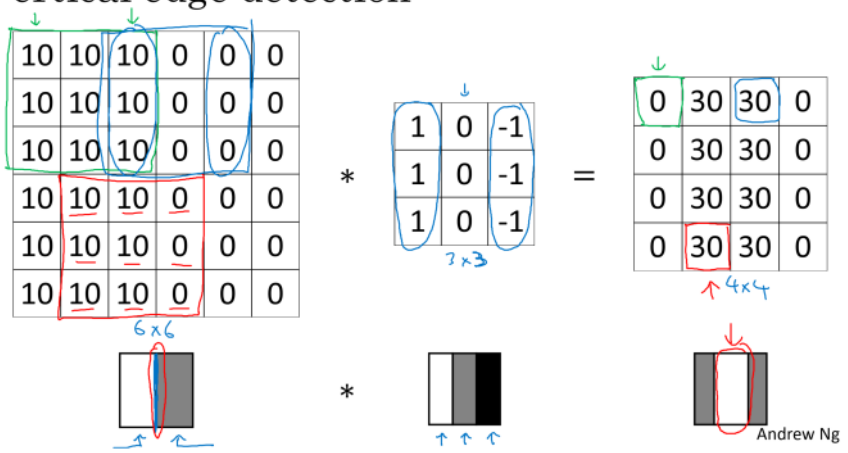
Convolve it with a 3×3 filter.

Andrew Ng

python: conv-forward

tensorflow: tf.nn.conv2d

Vertical edge detection



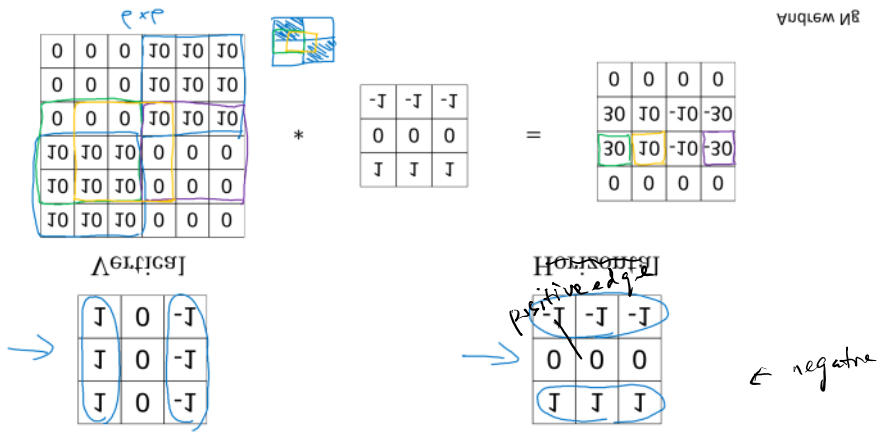
more edge

Convolutional Neural Networks

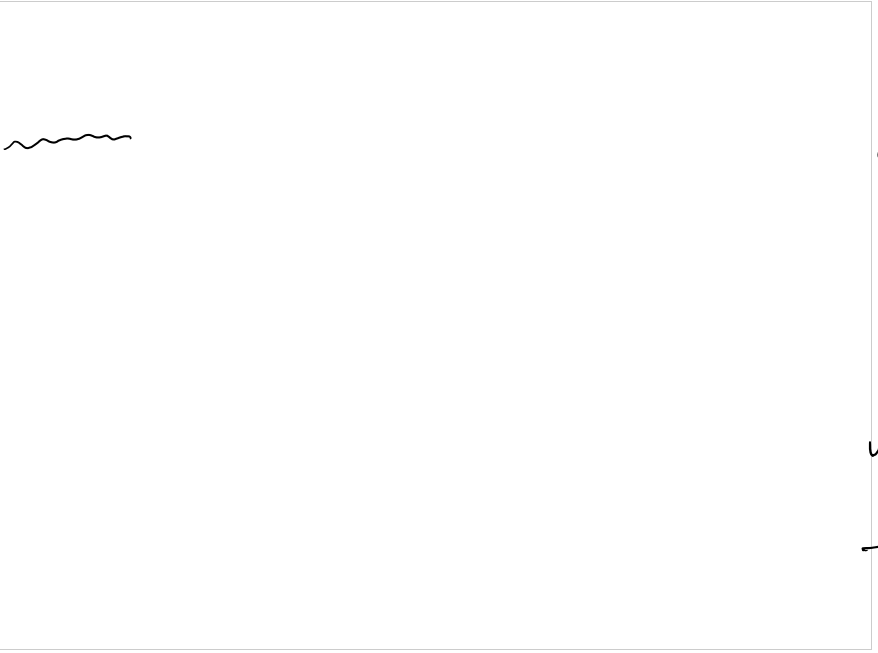
More edge detection

light to dark

dark to light.



vertical vs horizontal edge



Sobel filter: put more weights to the central weight

=> more robust

Scharr filter: another filter

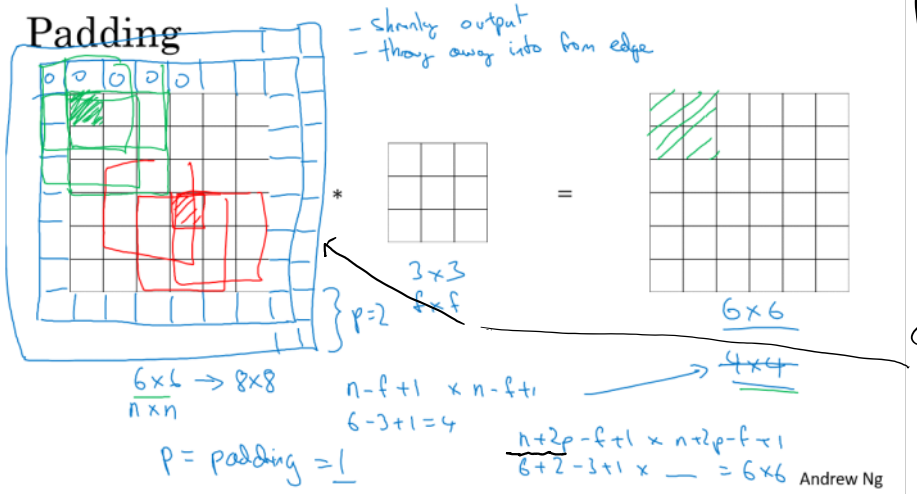
with ML, filters need not be hand-picked

it can be learned!

Treating the 9 numbers w_i, $i \in (1, 9)$ as params.

padding

Padding



Andrew Ng

nxn picture

Dimension fxf filter size

(n-f+1) x (n-f+1)

Problem ① Image shrink every time you detect edges etc.

② pixels at the corners are used less

⇒ throwing away information

Solution pad the image

p = padding

n+2p-f+1 = n

p = (f-1)/2

Valid and Same convolutions

"Valid": n×n * f×f → n-f+1 × n-f+1

6×6 * 3×3 → 4×4

no padding

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

n+2p-f+1 × n+2p-f+1

n+2p-f+1 = n ⇒ p = (f-1)/2

3×3 p = (3-1)/2 = 1

5×5 p=2

f is usually odd

1×1

3×3

5×5

7×7

Andrew Ng

Valid conv: no padding.

Same conv: pad ⇒ same size.

odd num filter convention

(1) c is possible

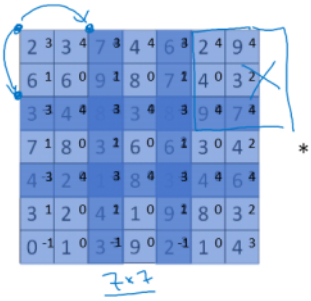
strided conv

deeplearning.ai

Convolutional Neural Networks

Strided convolutions

Strided convolution



7×7

$*$

3	4	4
1	0	2
-1	0	3

3×3

$=$

91	100	83
69	91	127
44	72	74

3×3

stride $= 2$

$\lfloor z \rfloor = \text{floor}(z)$

$n \times n$ padding p $*$ $f \times f$ stride s $s = 2$

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

floor.

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

Andrew Ng

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

Summary of convolutions

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

padding p $\quad$ stride s

Output size:

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

Andrew Ng

Technical note on cross-correlation vs. convolution

Convolution in math textbook: flip before operation.

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

*

3	4	5
1	0	2
-1	9	7

=

7	2	5
9	0	4
-1	1	3

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

Andrew Ng

this is Cross-correlation

In DL called "Convolution"

In Convention

In signal process. flipping helps on getting the associativity property.

But this doesn't matter for DL

This is just a caution about definition & notation.

May matter when you read some papers

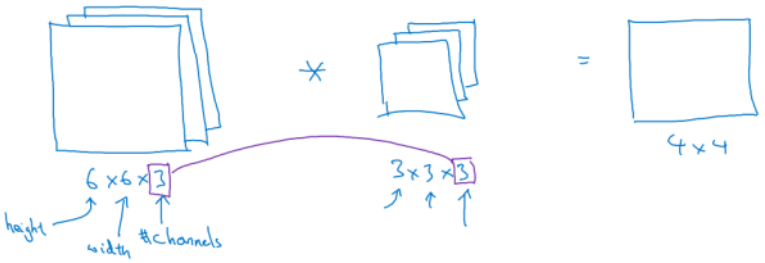
conv over volume

deeplearning.ai

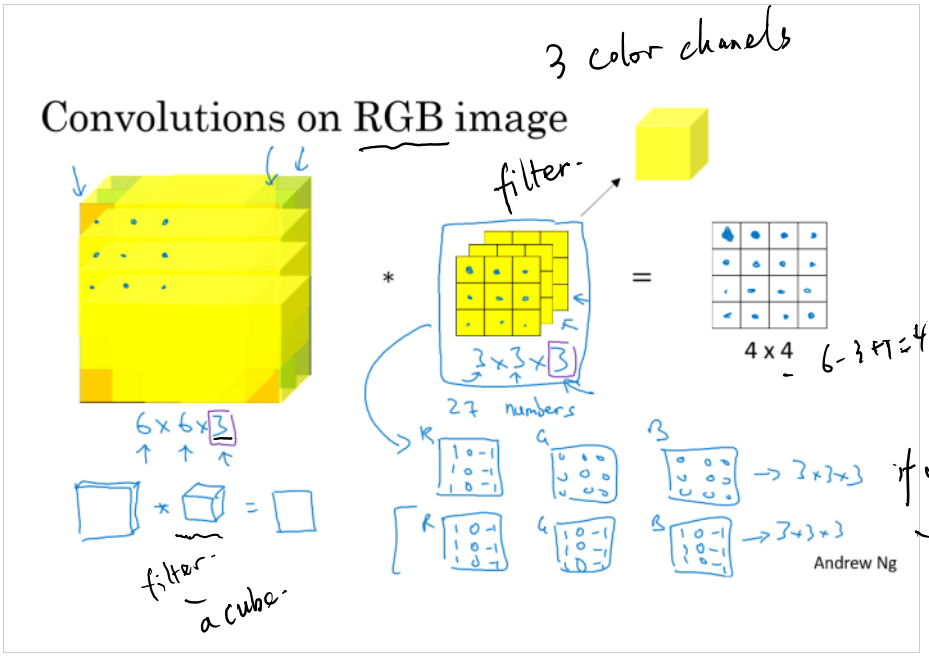
Convolutional Neural Networks

Convolutions over volumes

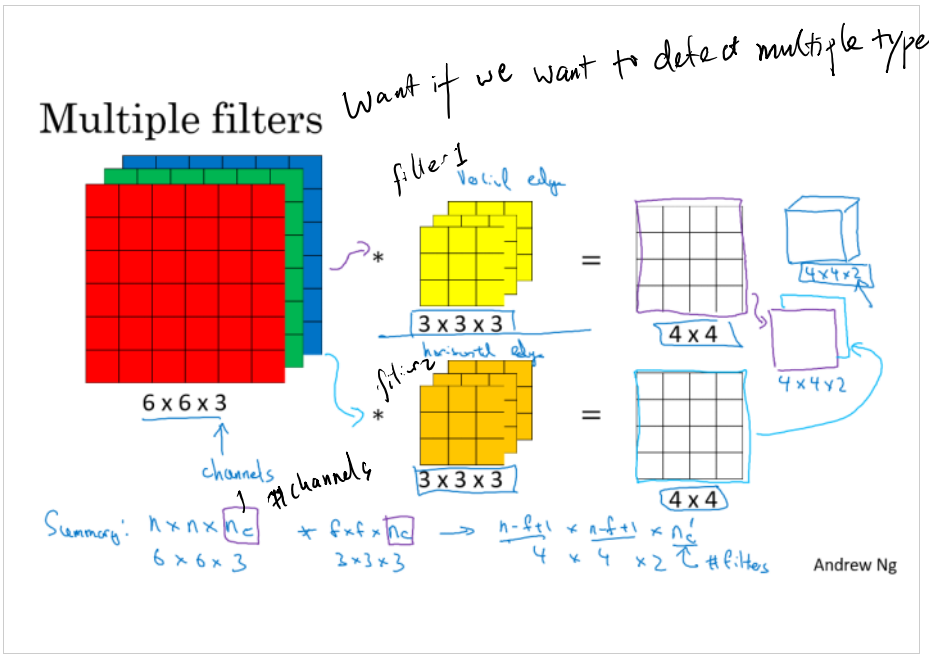
Convolutions on RGB images



Andrew Ng



if want to detect edges in the Red Channel
~ ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ in all channels.

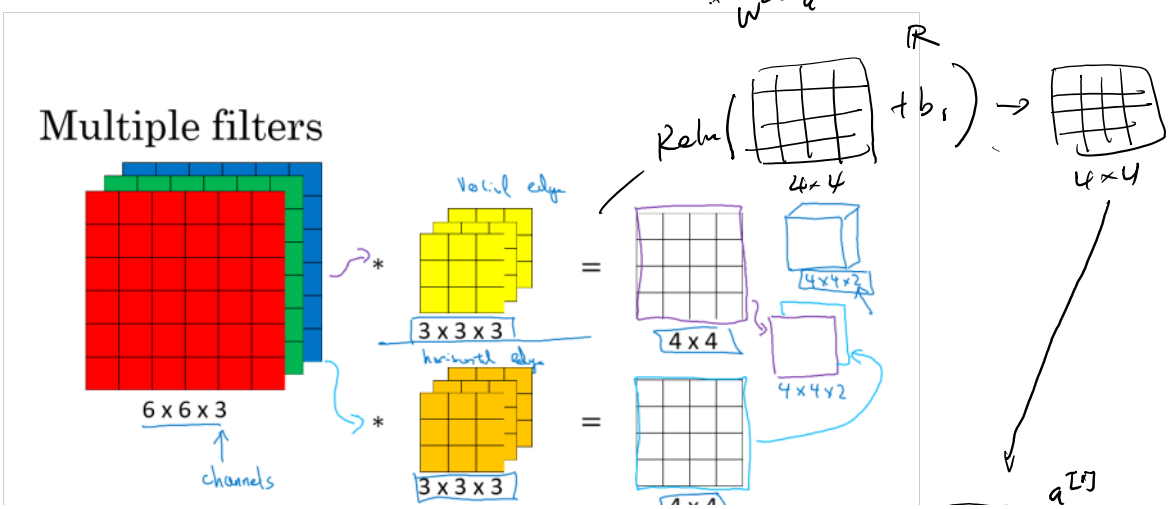


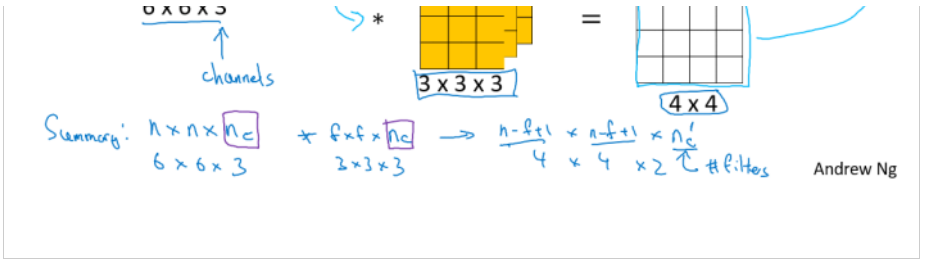
maybe: filter 1 – vertical edge detector

filter 2 = horizontal edge detector

n_c: # of channels
Depth of the 3D volume

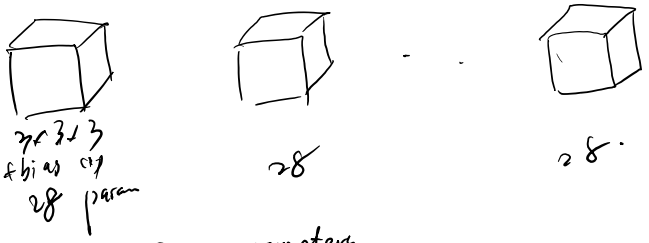
one layer of CNN





If we have 10 filters that are 3×3×3 in one layer.

How many param ?



280 parameters.

10 filters that detect different edges

can work for even large images

⇒ significantly less parameter

than not using CNN (described in the intro)

⇒ avoid overfitting problem.

If layer l is a convolution layer

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

$p^{[l]}$ = padding size

$s^{[l]}$ = stride

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

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

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

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

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

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

Activation: $a^{[l]} \to n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

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

Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$ ← # of filters in layer l

bias: $(1, 1, 1, n_c^{[l]})$ ← 每个 channel 有一个 bias.

Caution: in open source implementation, no convention about the order of dimensions.

Some frameworks put channel first.

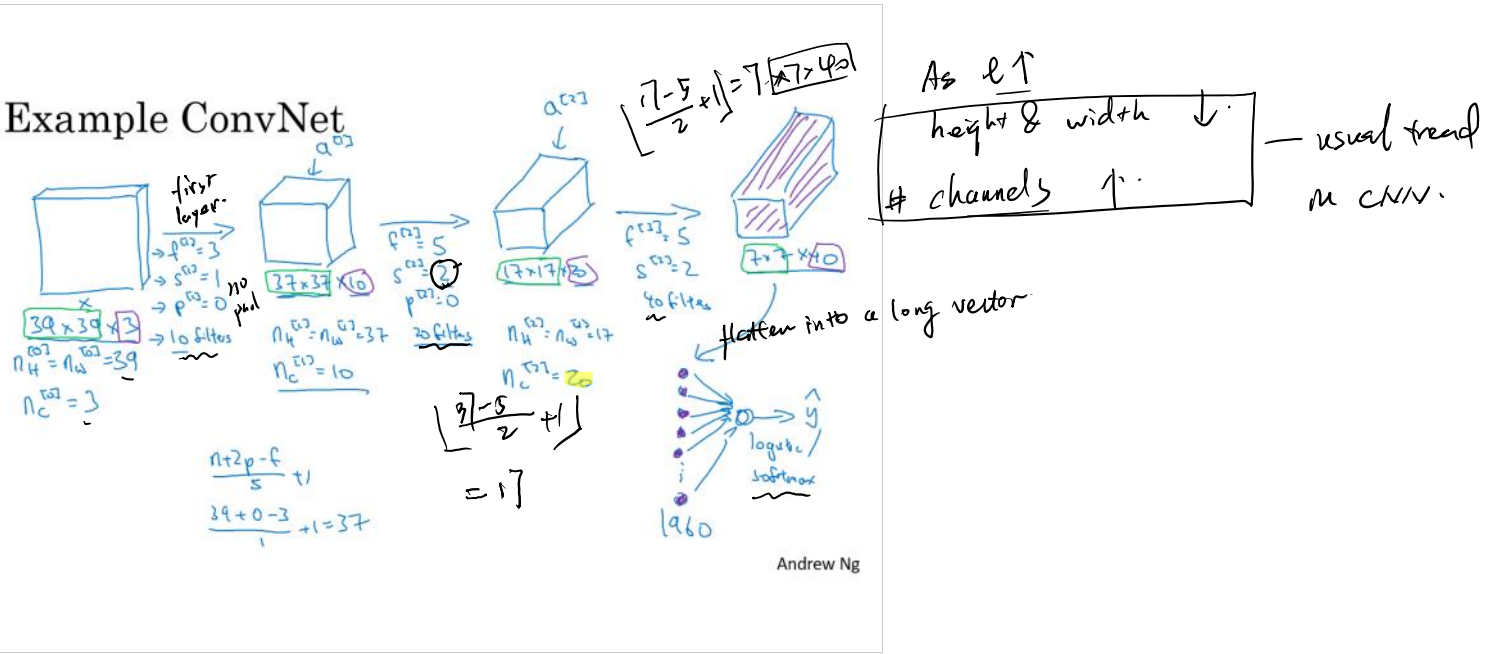
---

single conv net

deeplearning.ai

Convolutional Neural Networks

A simple convolution network example



As $\ell \uparrow$
height & width ↓.
channels ↑.
— usual trend in CNN.

Types of layer in a convolutional network:

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

Andrew Ng

pooling

Convolutional Neural Networks

Pooling layers

Pooling layer: Max pooling

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

9	2
6	3

(large value can be viewed as) detecting some feature.

Max Pooling: if this feature is detected in this region → keep the large number

(Intuition)

No proof that this is the real underlying mechanism

Andrew Ng

But why is it used? proved to work well.

Pooling layer: Max pooling

也是作用于每一个 filter，有 filter size f

stride s

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

$5 \times 5 \times n_c$

9	9	5
9	9	5
8	6	9

$3 \times 3 \times n_c$

$f = 3$, $s = 1$

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

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

$f = 2$
$s = 2$

$7 \times 7 \times 1000 \rightarrow 1 \times 1 \times 1000$

Andrew Ng

Max pooling is used more often than avg pooling.

In very deep NN, may need to use avg pooling to collapse representation. $7 \times 7 \times 1000$

$\Rightarrow 1 \times 1 \times 1000$

Summary of pooling

Hyperparameters:

- f : filter size — $f=2, s=2$ / $f=3, s=2$
- s : stride
- Max or average pooling

~~$\Rightarrow$ p : padding~~

No parameters to learn!

$n_H \times n_W \times n_C$

$\downarrow$

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

usually no padding.

Andrew Ng

cnn example

deeplearning.ai

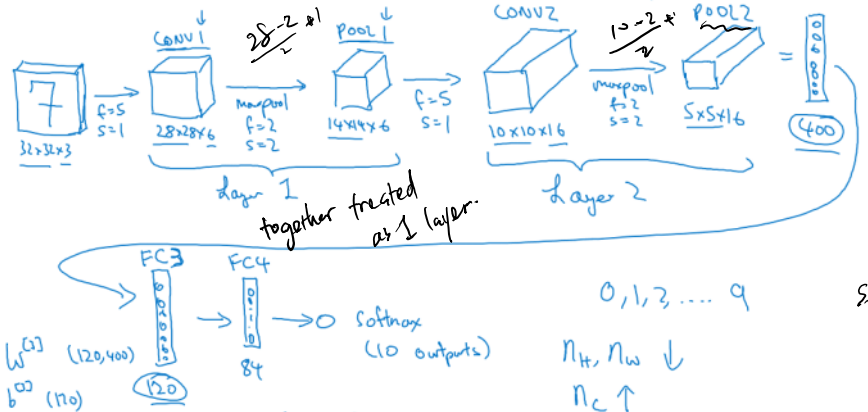
Convolutional Neural Networks

Convolutional neural network example

Hand-written digit recognition

LeCun

Neural network example (LeNet-5) not exactly



Andrew Ng

Notation thing: counting layers that have weights

search for hyperparameters in the literature

Don't blindly search for own hyperparam

usually. channel (n_c) ↑ => full connected

Neural network example

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

Andrew Ng

- Max Pooling layer have no params.

Fully connected layers have lots of parameters.

lots of convnet share similar structure

putting layers together requires insight

— It'll be good to build on prior work.

Why convolution

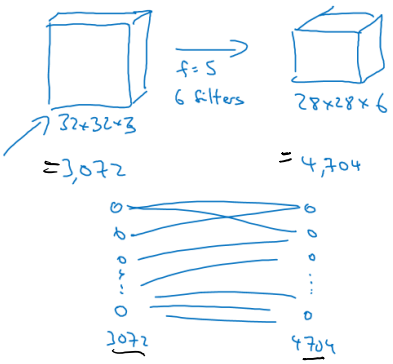
why conv

Convolutional Neural Networks

Why convolutions?

deeplearning.ai

Why convolutions



5×5 − 25
26
6 × 26 = 156 parameters

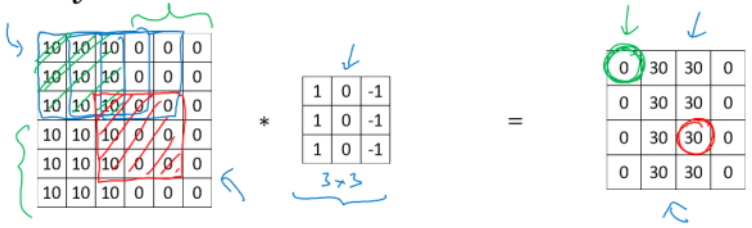
3,072 × 4,704 ≈ 14M

If ~~too~~ fully connected to start.
⇒ too many parameters !

Andrew Ng

Why convolutions

translation invariance



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.

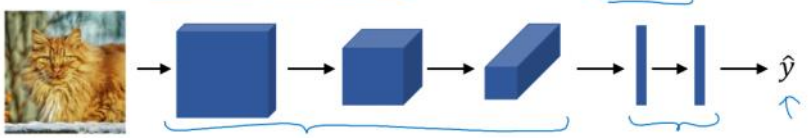
Sparsity of connections: In each layer, each output value depends only on a small number of inputs.

← How this work(s)
local features.
output depends on only a small # of input

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

Andrew Ng

Andrew Ng

✔ 1 / 1 points

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)?

7600 -- 所以自动乘以三?

✔ 1 / 1 points

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)?

27000100

✔ 1 / 1 points

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

False!