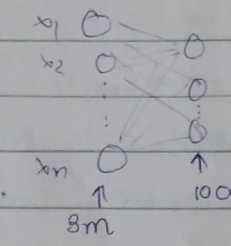
Course-4 CNN

Week-1

lets say you are using a 1000×1000 image. The total size - 3mn (3 RGB channel)

If we put in a NN,



W[1] = (1000, 3m)

, This is a very big size.

cons: More memory, more complexity

This can be solved using Convolution function

A grey scale image when given to computer,

The computer identifies vertical edges first and then horizontal edges.

consider, a 6×6 matrix representing vertical edges. This is a (2D) matrix because img is gray scale

3×1 + 1×1 + 2×1 + 0×0 + 5×0 + 7×0 + 1×-1 + 8×-1 + 2×-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
n×n

convolut *

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

3×3 f×f
filter

=

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

4×4
(n-f+1) = 4×4

The above was vertical edge detector filter.

Horizontal edge detector

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

Sobel filter

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

Scharr filter

3	0	-3
10	0	-10
3	0	-3

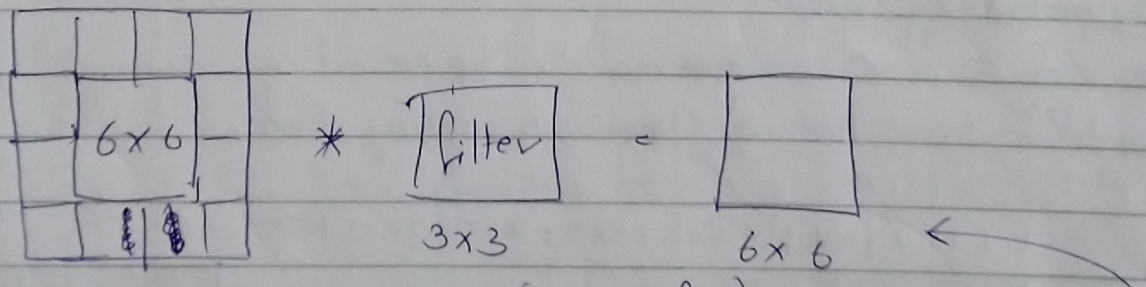
So, what we do in NN, is we consider these elements as weights and allow it to change with back prop. This allows to capture the data better than any hand coded filters.

There are two problems here.

1) Shrinking of image $6 \times 6 \longrightarrow 4 \times 4$
2) pixel at the corners are used only once, unlike a pixel in the ~~int~~ middle, thus missing some info on each step

To stop this, we do Padding.

We pad with zeros., here $p=1$ (as we pad one extra border of one pixels



"Valid" Convolution : No padding $n \times n * f \times f \longrightarrow n-f+1 \times n-f+1$

"Same" Convolution : Pad so that o/p size is same as i/p size

$n+2p-f+1 = n$ ~~to make~~

to make the size same as i/p : $n+2p-f+1 = n$

$\Rightarrow p - \frac{f-1}{2} = 1$ (here)

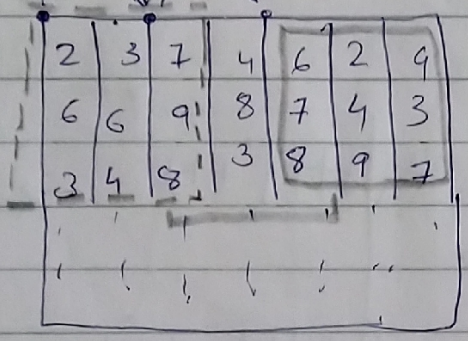
f is usually odd numbered because,

1) padding fits properly
2) You get a central pixel

Strided Convolution

Here we choose how many steps one wants to use.

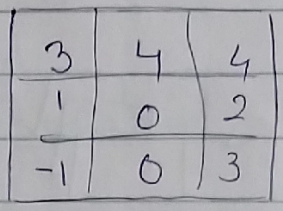
For eg.



7×7

$n \times n$

$*$



3×3

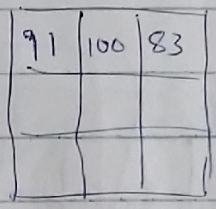
$f \times f$

stride = 2

padding $\rightarrow p$

stride $\rightarrow s$

$=$

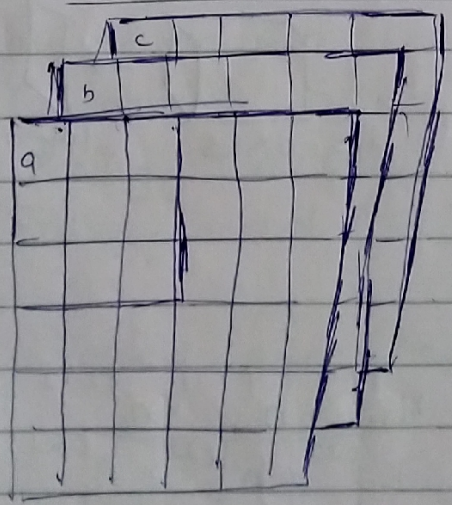


3×3

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

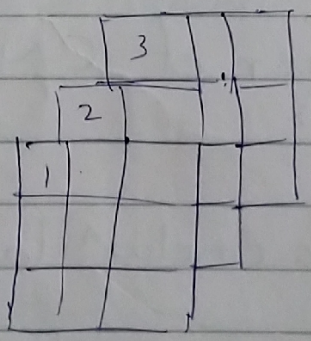
round to nearest integer

Convolution over volumes



$6 \times 6 \times 3$

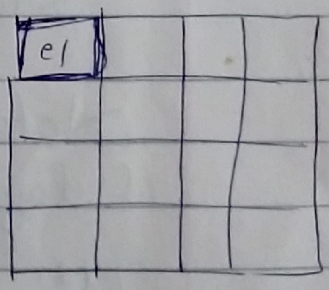
$*$



$3 \times 3 \times 3$

filter

$=$

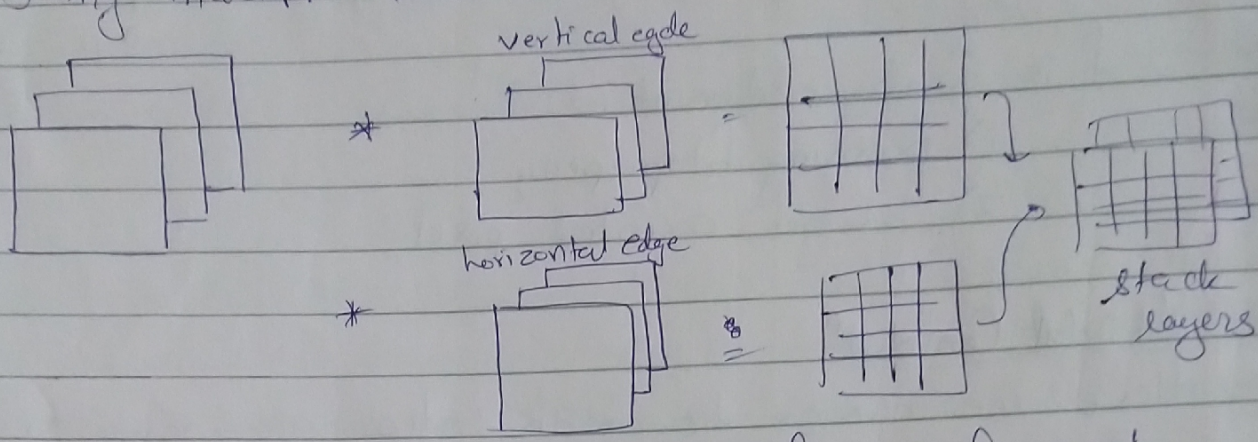


4×4

~~first~~ R ~~sec~~ G B

$e1 = a \times 1 + \ldots) + (b \times 2 + \ldots) + (c \times 3 + \ldots)$

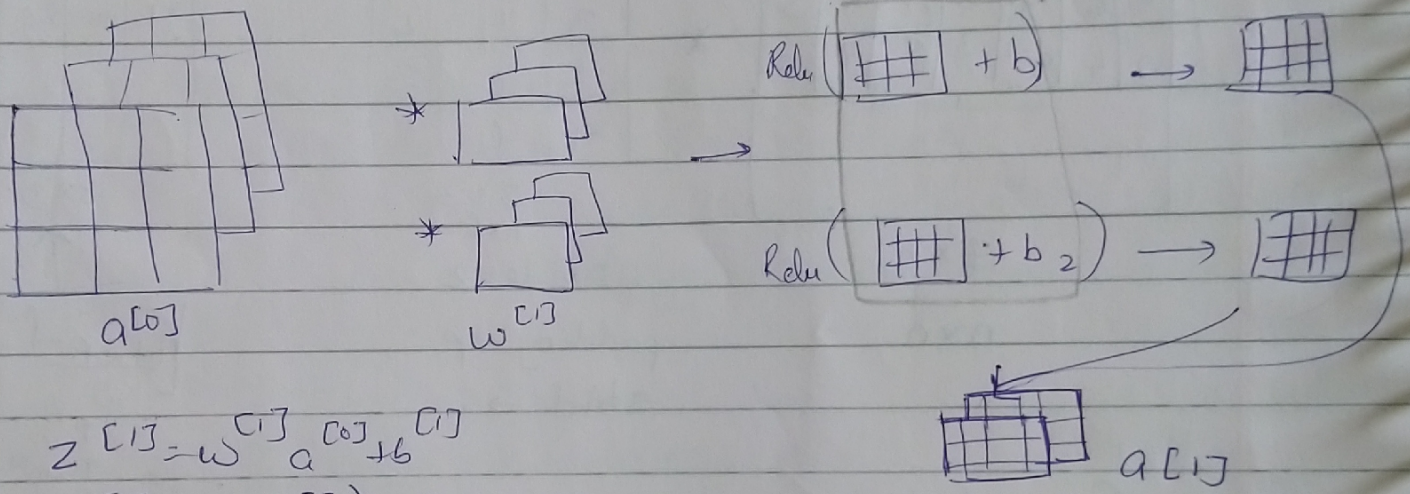
Using multiple features



$n \times n \times n_c \quad * \quad f \times f \times n_c \longrightarrow n-f+1 \times n-f+1 \times n_c'$

n_c = # channels

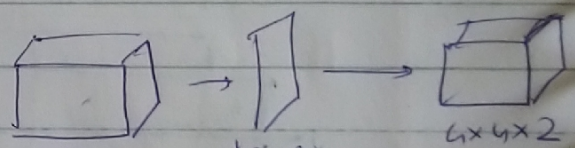
n_c' ← # filters



$z^{[1]} = w^{[1]} a^{[0]} + b^{[1]}$

$a^{[1]} = g(z^{[1]})$

One layer of CNN



Notations

If layer l is a convolut^n layer

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

$p^{[l]}$ = ~~stride~~ padding

$s^{[l]}$ = stride

$n_c^{[l]}$ = # filters

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

Activations: $a^{[l]} \longrightarrow n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$

I/p: $n_H^{[l-1]} \times n_w^{[l-1]} \times n_c^{[l-1]}$

O/p $n_H^{[l]} \times n_w^{[l]} \times n_c^{[l]}$

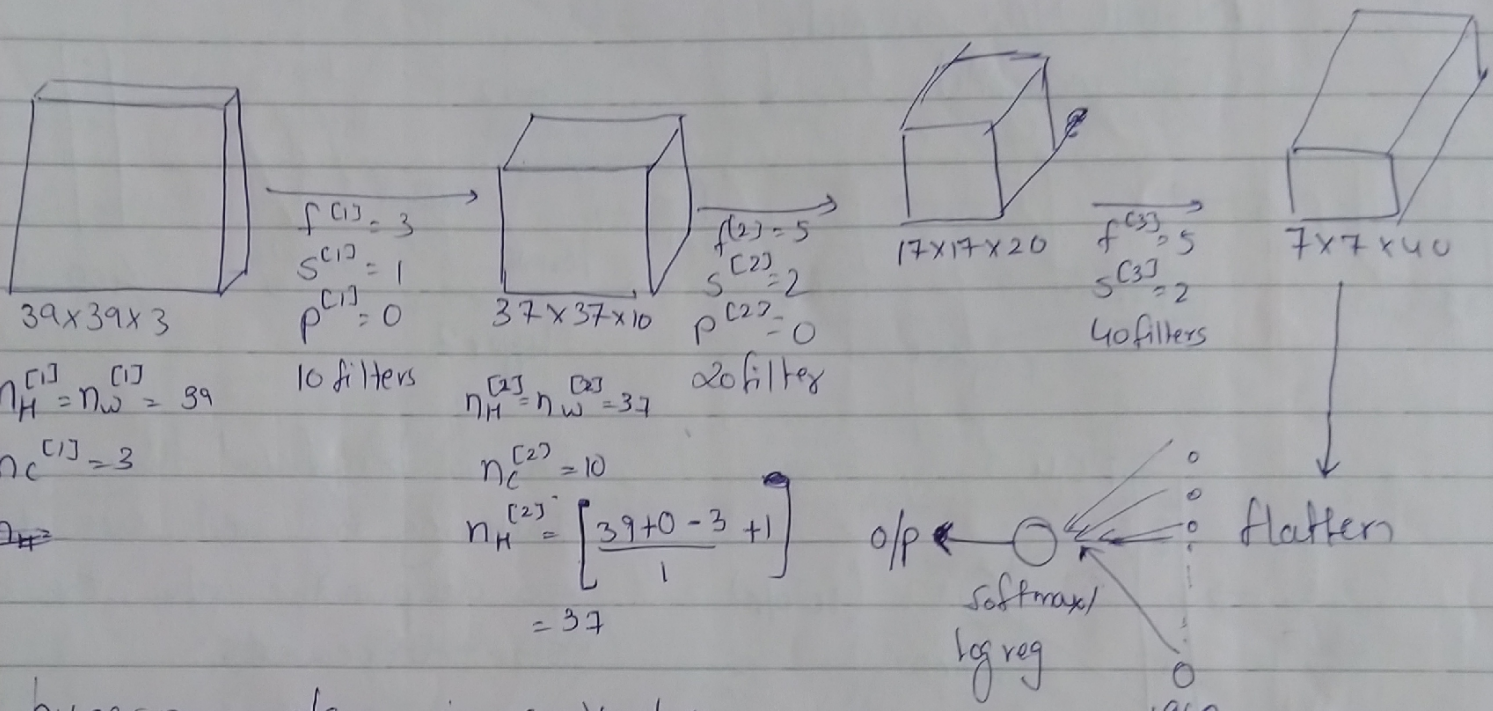
$$n_H^{[l]}, n_w^{[l]} = \left\lfloor \frac{n_H^{[l-1]} + 2p^{[l]} - f^{[l]}}{s^{[l]}} + 1 \right\rfloor$$

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

Weights: $f^{[l]} \times f^{[l]} \times n_c^{[l-1]} \times n_c^{[l]}$ bias: $n_c^{[l]} \rightarrow (1, 1, 1, n_c^{[l]})$

↑ filters in layer l

Example of ConvNet



$n_H^{[1]} = n_W^{[1]} = 39$

$n_c^{[1]} = 3$

$n_c^{[2]} = 10$

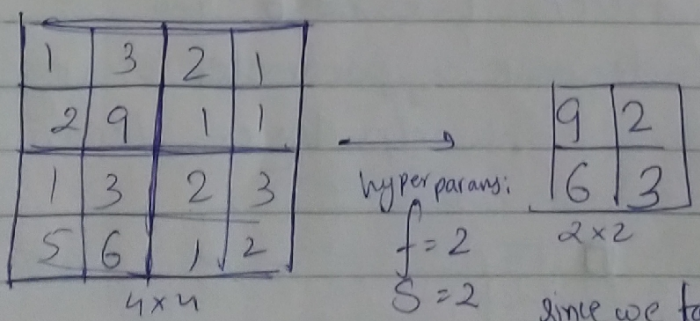
$n_H^{[2]} = \left[\frac{39+0-3}{1} + 1\right]$

$= 37$

hyper parameters in convNet:
- Strides
- Padding
- # filters

Types of layer in convolutional network:
- Convolution (CONV)
- Pooling (POOL)
- Fullyconnected (FC)

Pooling Layer: Max Pooling



Since we take two strides

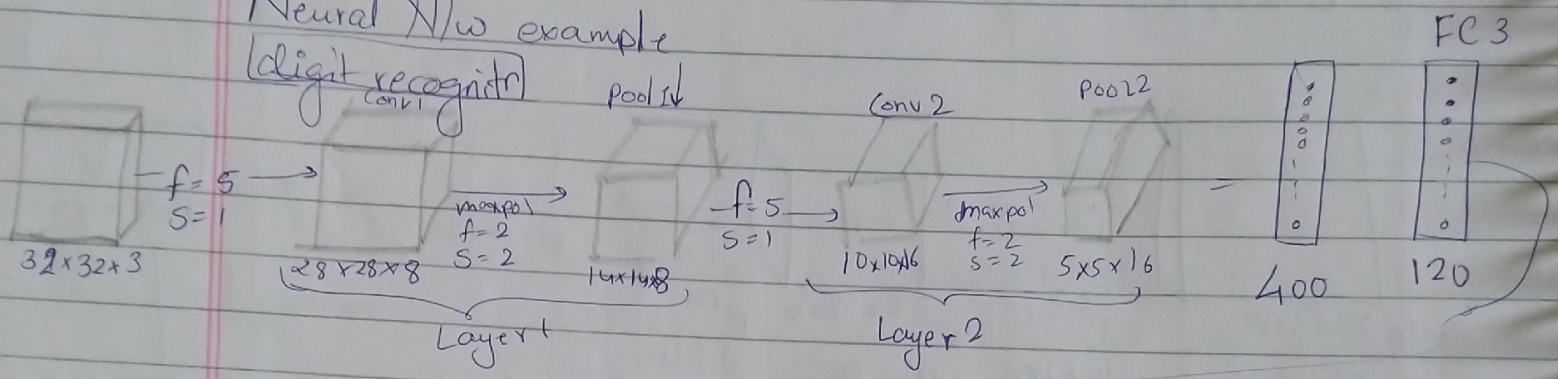
It doesn't have any params to learn, once you fix the hyperparameter

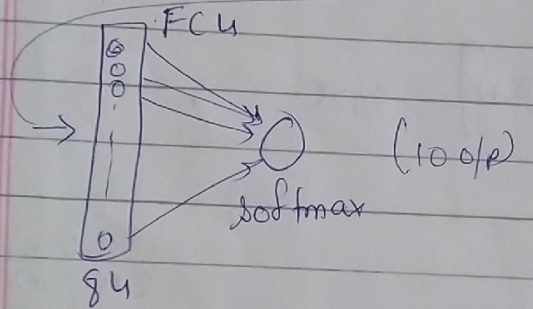
Another pooling technique, less used is Average pooling

Pooling doesn't use padding

Neural N/w example

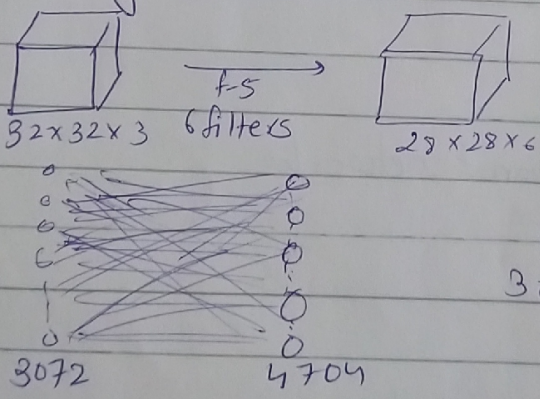
Digit recognition





This is beneficial than just flattening ing and use it. As, we go deep in the n/w, the activatn size decreases

'Why convolution'



Parameters

$3072 \times 4704 = 14M$

Wheras in conv net all one has to do, is adjust the params of filters. $(5 \times 5 \times 3 + 1) \times 6 = 456$

(1) Reason of this decrease is parameter sharing.

(2) Sparsity of connections: each o/p layer depends on a small no. of i/p's