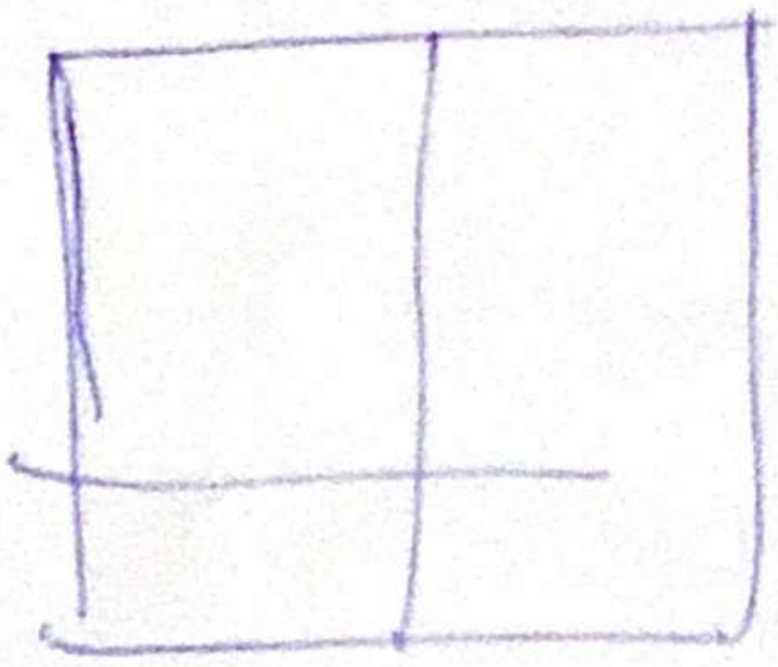
Convolutional Neural networks

Edge Detection

· vertical

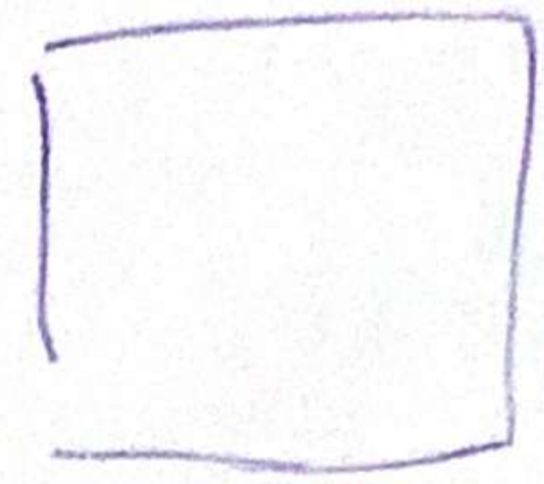


$*$

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$$

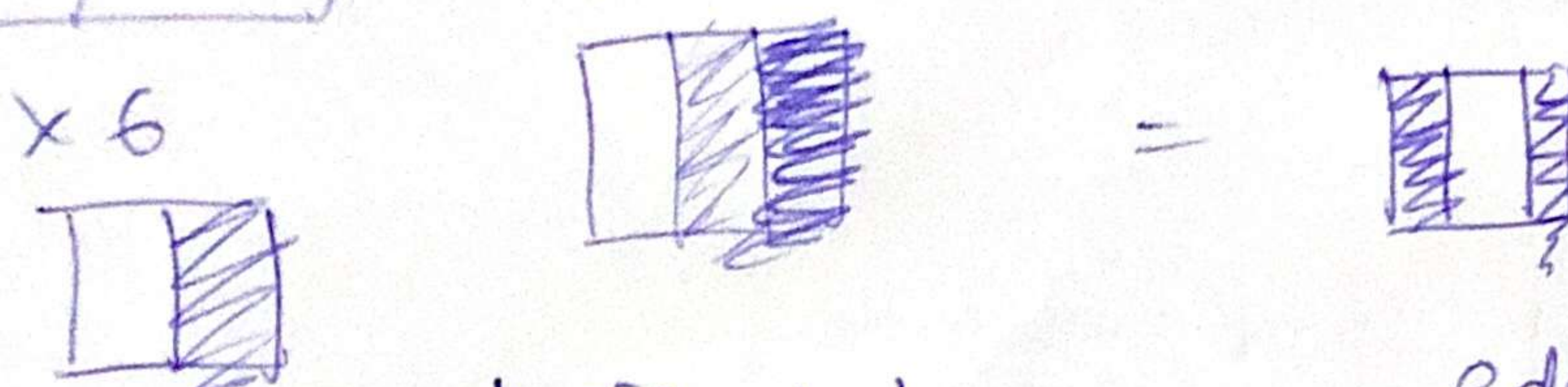
filter

$\longrightarrow$



4x4

6x6



edge.

horizontal.

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

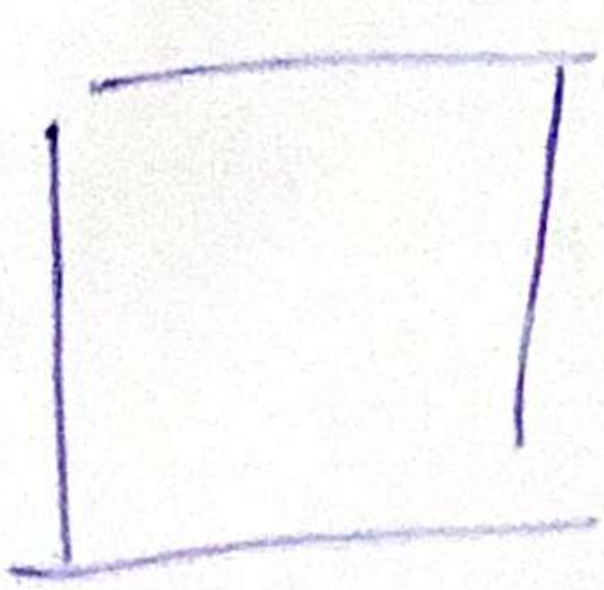
filter.

more robust filter $\longrightarrow$

$$\begin{bmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix}$$

more weight on the middle row

Padding



$*$

=.

6x6.

nxn.

3x3

fxf

4x4.

$(n-f+1) \times (n-f+1)$.

normal convolution.

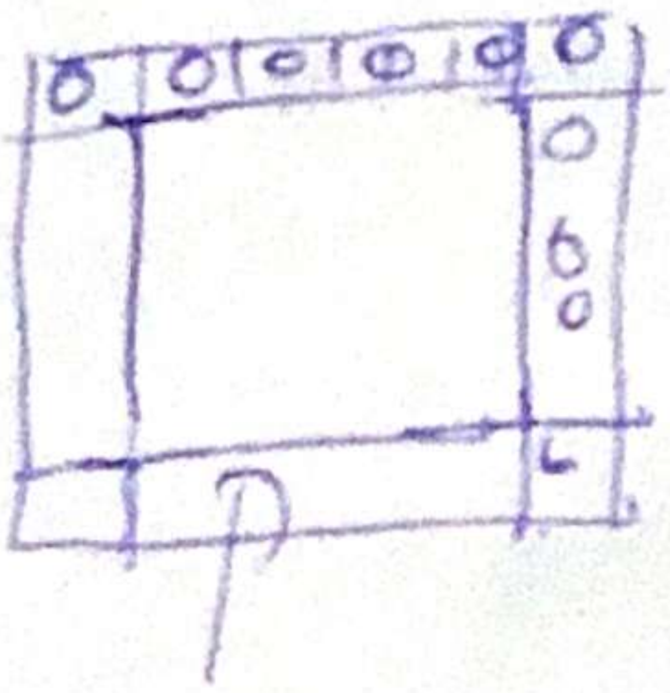
- shrinks the output
- Throw away from edge

$6\times6 \rightarrow 8\times8 \quad * \quad 3\times3 \rightarrow 6\times6$

$n\times n \qquad f\times f$

p = padding = 1

?



6×6

$(n+2p-f+1)\times(n+2p-f+1)$

$(6+2-3+1)\times(6+2-3+1)$

Valid and Same Convolutions

Valid: $n\times n * f\times f \rightarrow (n-f+1)\times(n-f+1)$

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

$n\times n \qquad (n+2p-f+1)\,(n+2p-f+1)$

$n+2p-f+1 = n$

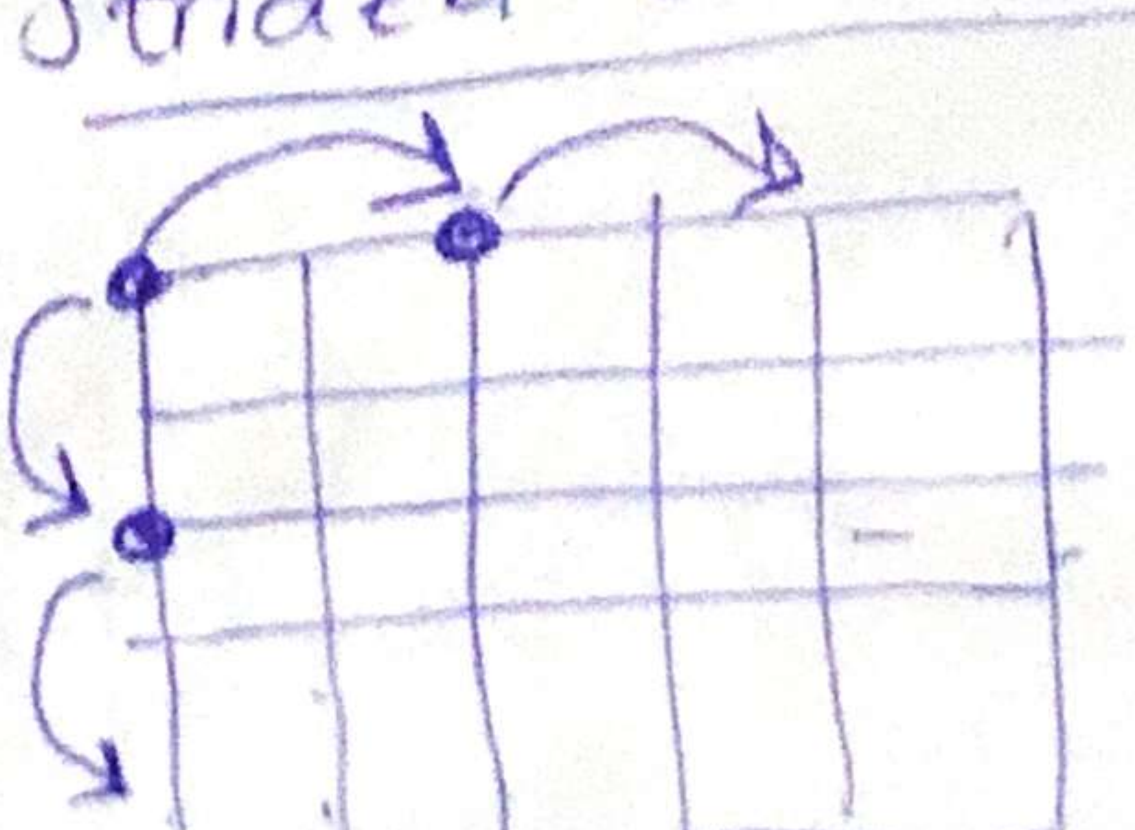
$\Rightarrow p = \frac{(f-1)}{2}$

for 3×3 kernel $\quad f=3$, $p=1$

& 5×5 kernel $\quad f=5$, $p=2$

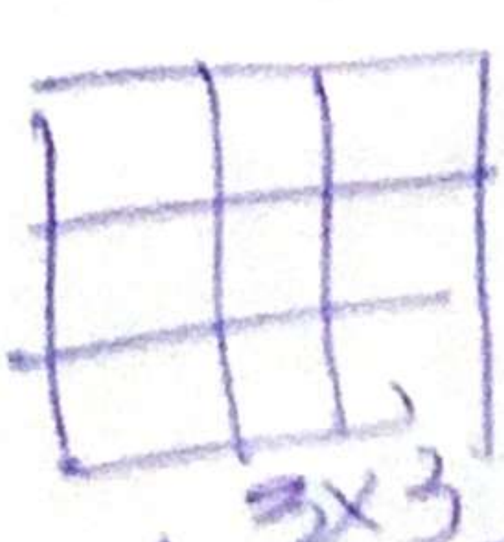
& f is usually odd.

Strided Convolution



7×7

$*$

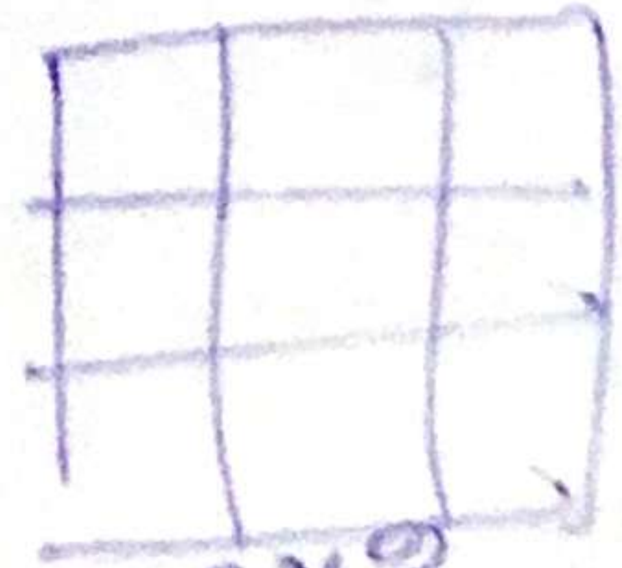


3×3

stride = 2

$p = 0$

=



3×3

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

$\left(\frac{7+0-3}{2}+1\right)$

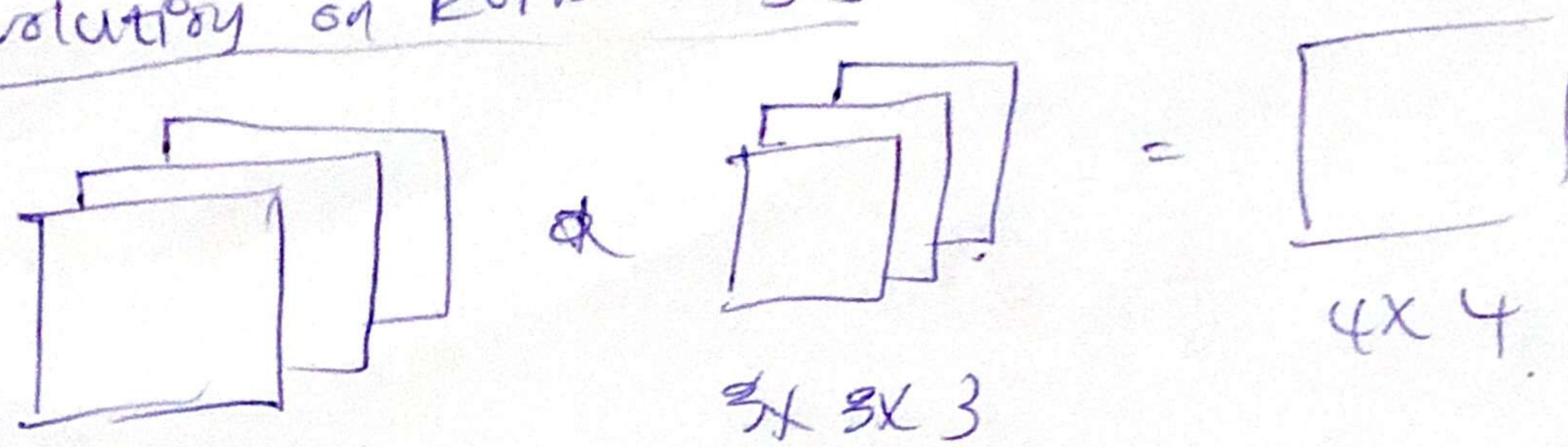
$3\times3 \Leftarrow$

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

Convolution in math textbooks:
flipping the terminal before convoluting
mathematicians call this cross co-relation.

Convolutions over Volumes

Convolution on RGB images



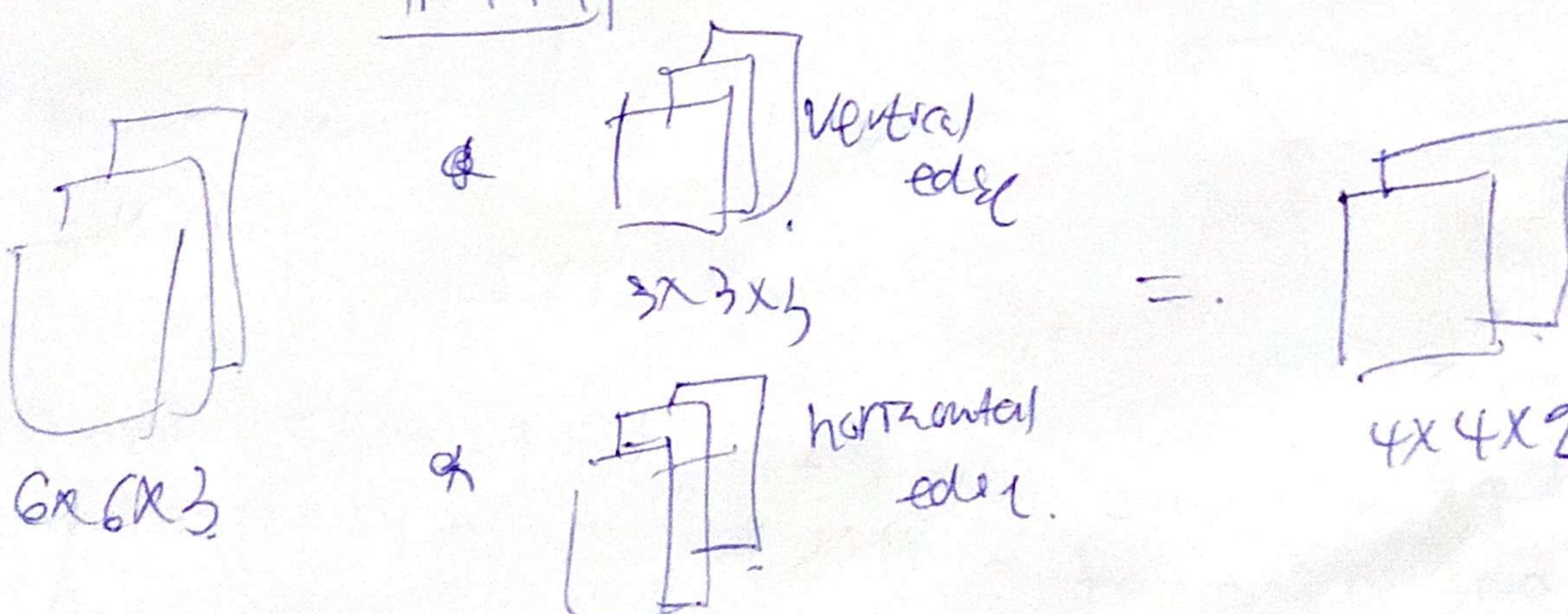
4X4

3X3X3

6x6x3
height width channels

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

filter for vertical edge.



vertical edge

3x3x3

=

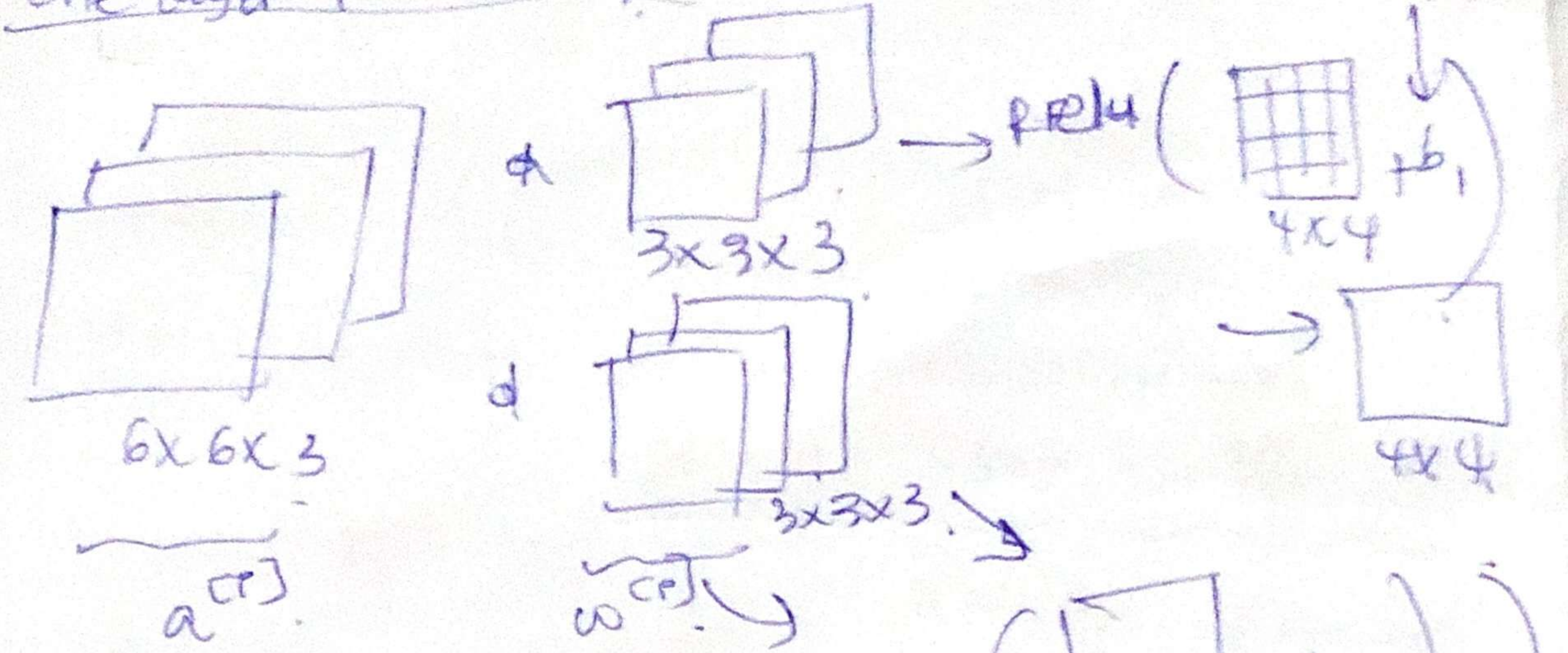
4X4X2

6x6x3

horizontal edge.

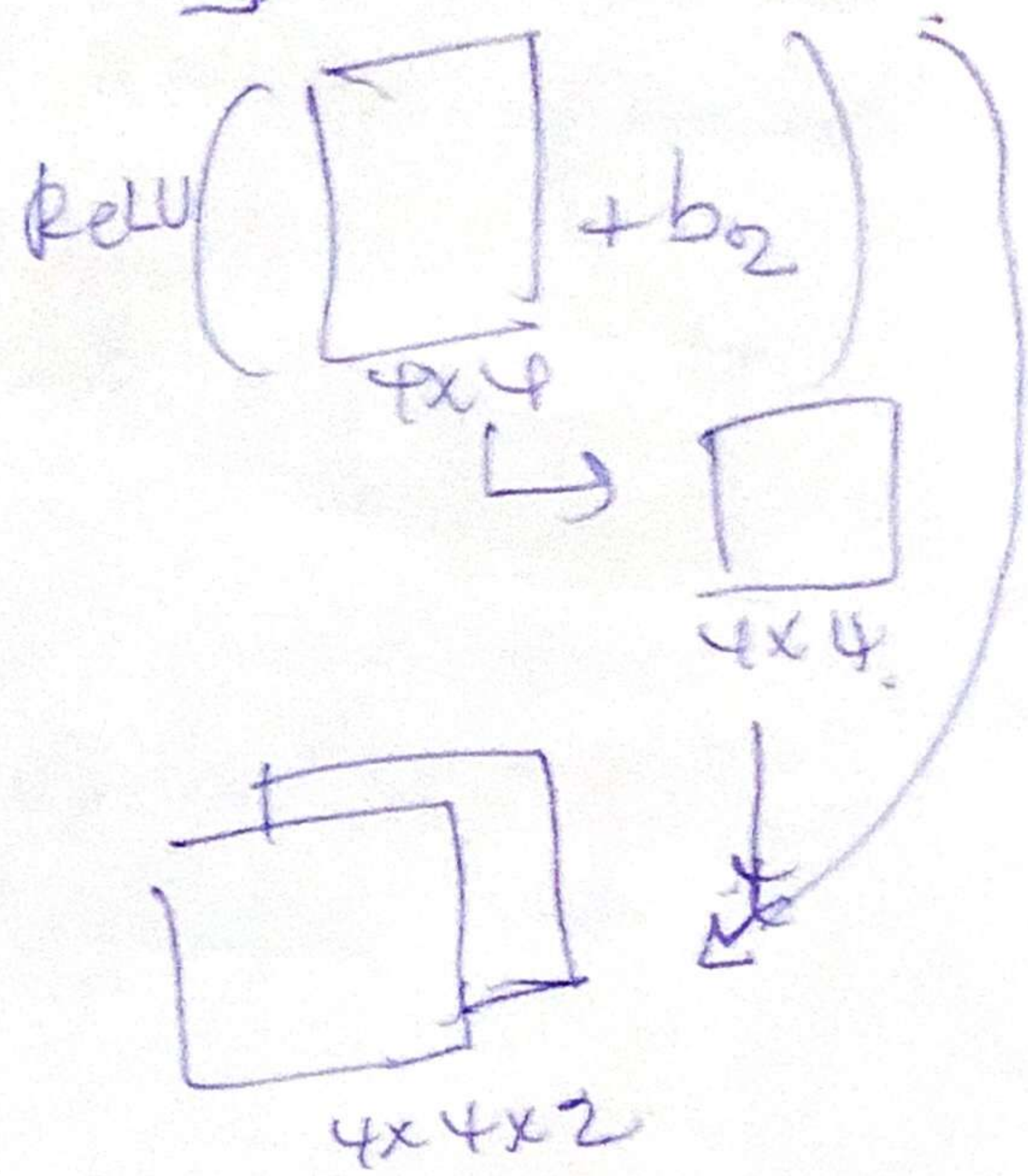
3x3x3

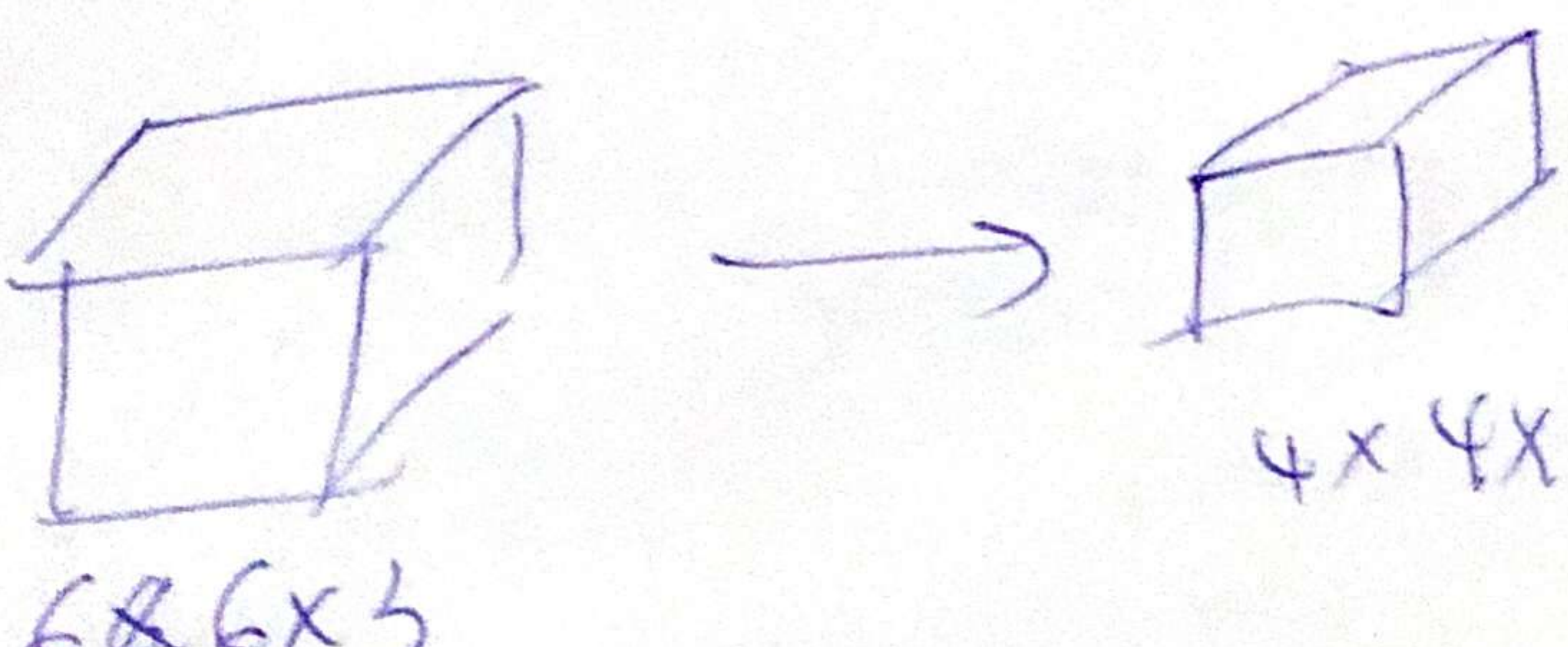
One layer of CNN



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

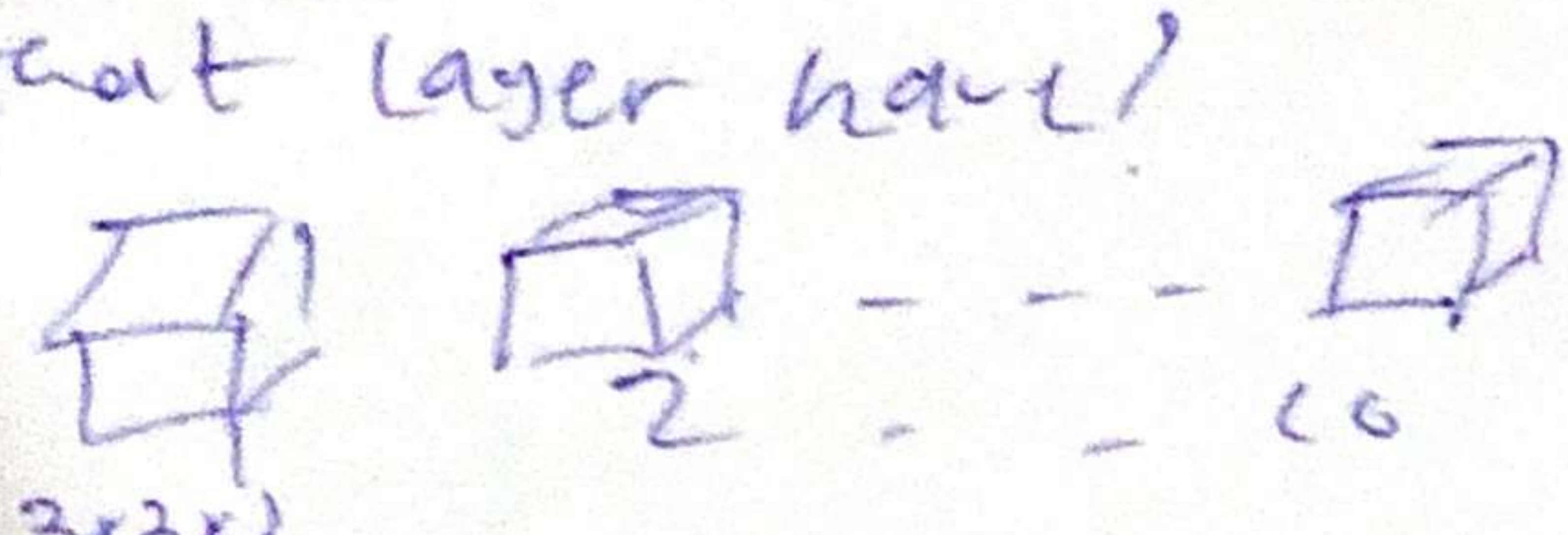
$a^{[1]} = g(z^{[1]})$



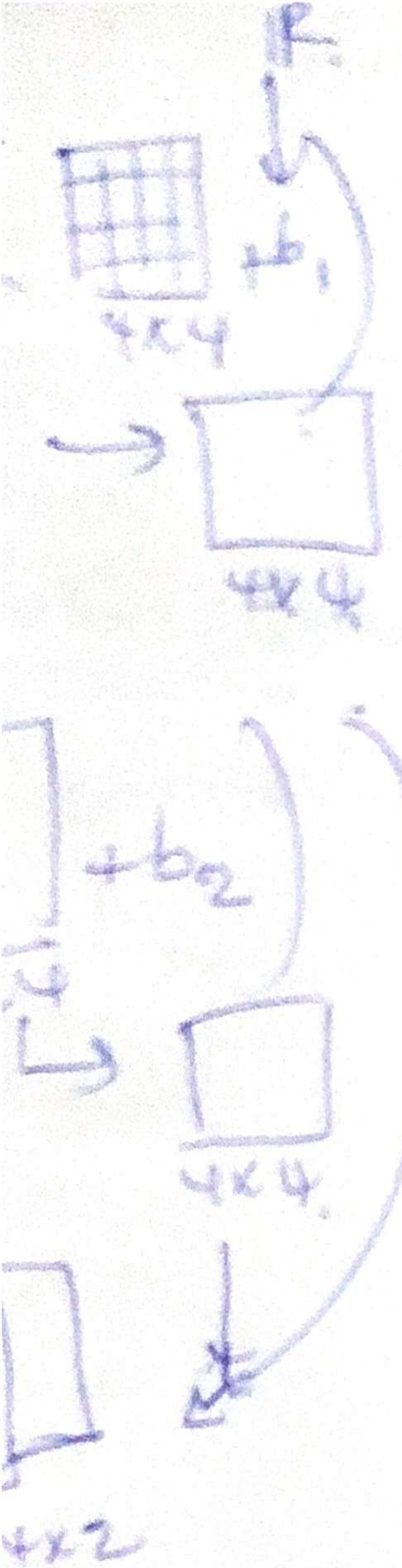


one layer of CNN

If you have 10 filters that are 3x3x3 in one layer of a NN, how many parameters does that layer have?



27 + 1 → 28 params ⟹ 28x10 = 280 parameters



ne layer of CNN

3x3x3 ...
parameters ...

... = 280 parameters

* 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 = $f^{[l]} \times f^{[l]} \times n_c^{[l-1]}$

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

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

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

Activations:

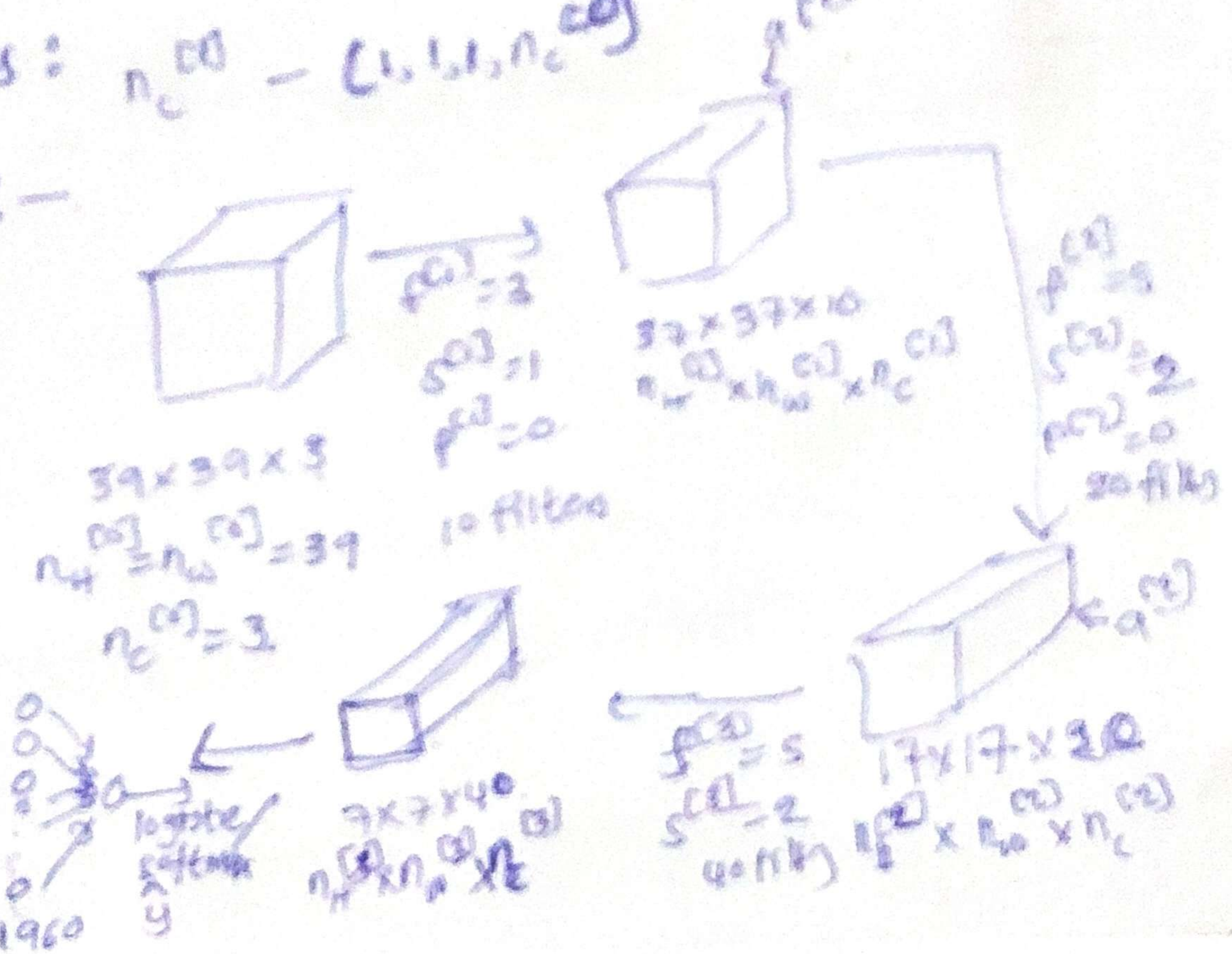
$a^{[l]} \rightarrow n_H^{[l]} \times n_W^{[l]} \times n_c^{[l]}$

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

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

bias: $n_c^{[l]}$ — $(1, 1, 1, n_c^{[l]})$

eg:—

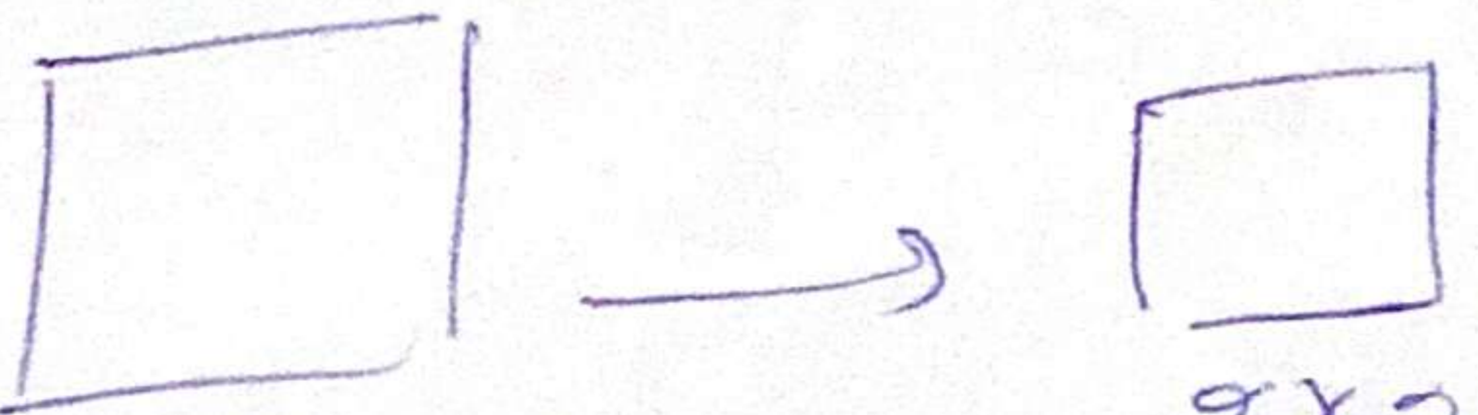


Types of CNN
- Convolution
- Pooling
- Fully connected

Pooling Layers

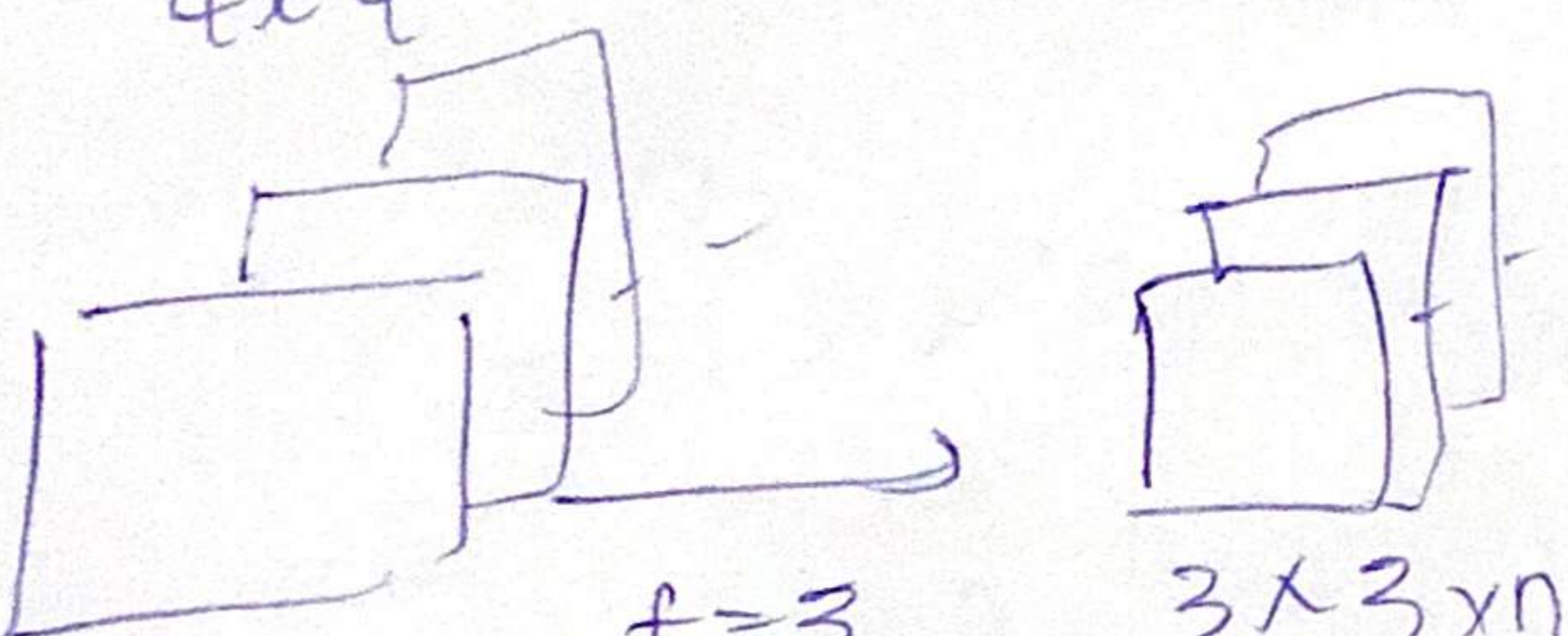
Hyperparameters $f = 2$, $s = 2$

1) Max pooling
2) Average pooling



4X4 2X2

f: filter size
s: stride
do not use any padding ($p=0$)

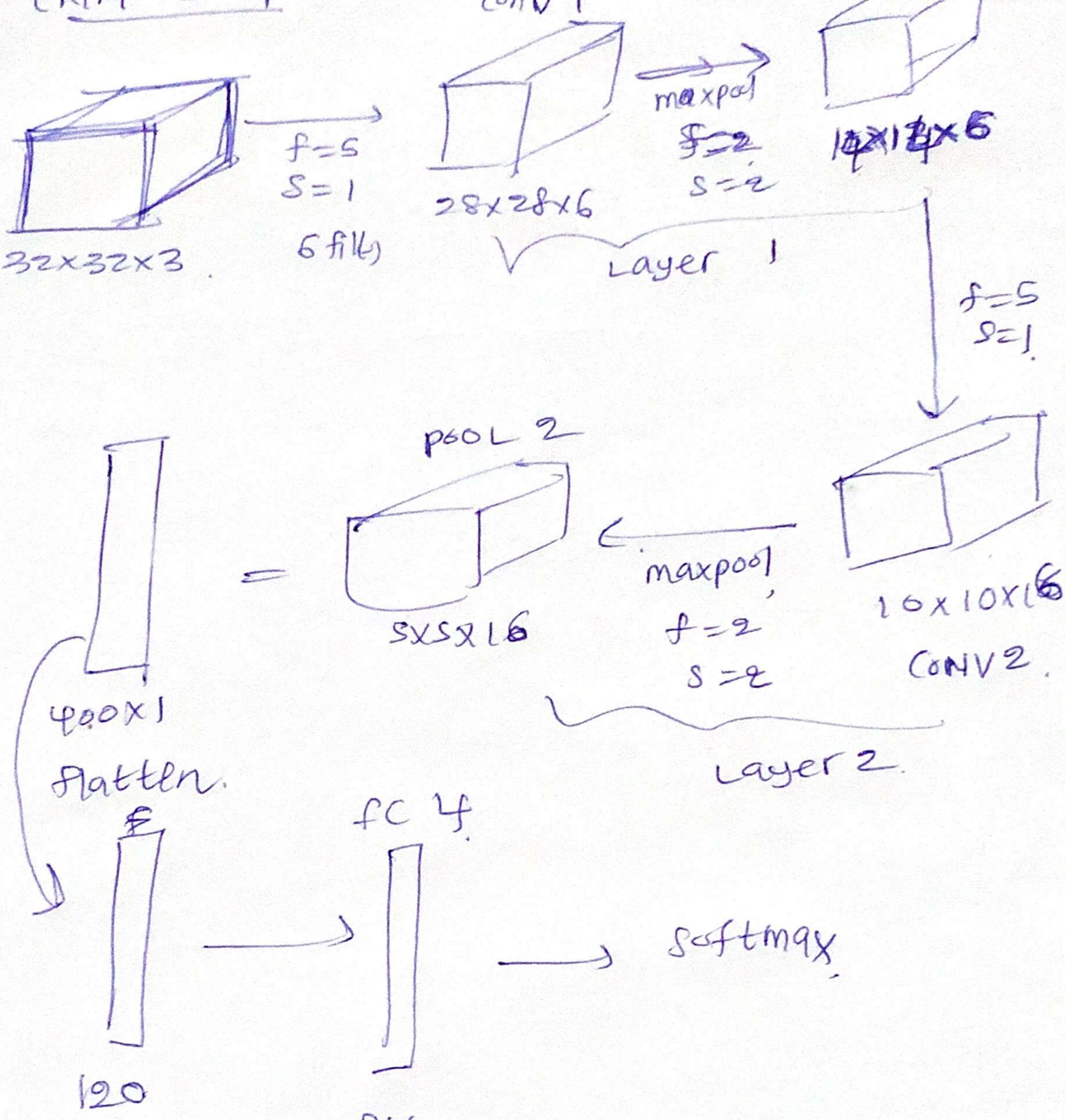


$5 \times 5 \times n_c$ $f=3$, $s=1$ $3 \times 3 \times n_c$

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

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

CNN example



FC3

[fully connected]

$W^{[3]}$ (120, 400)

$b^{[3]}$ (120).