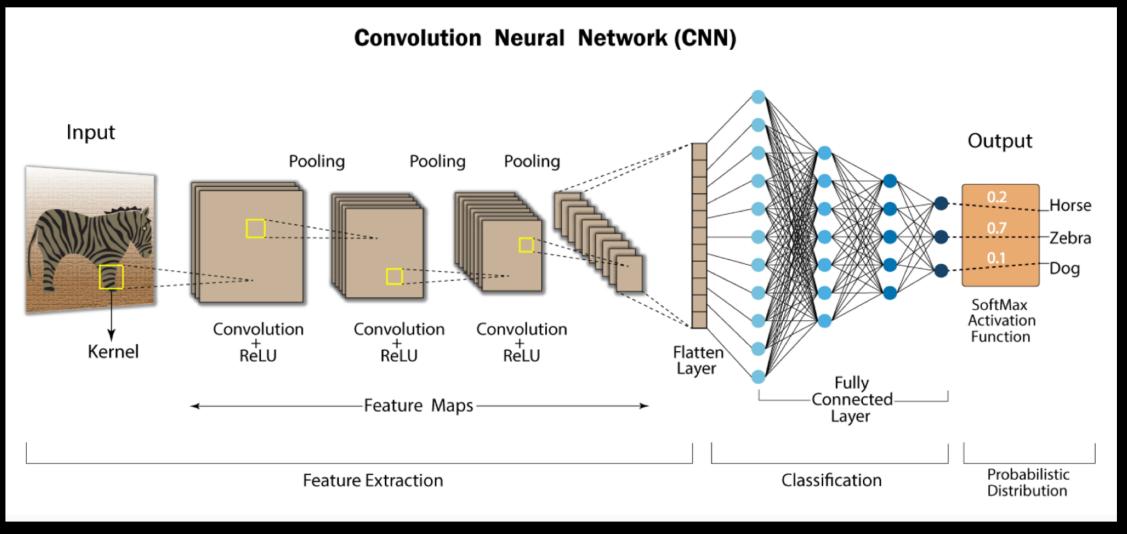
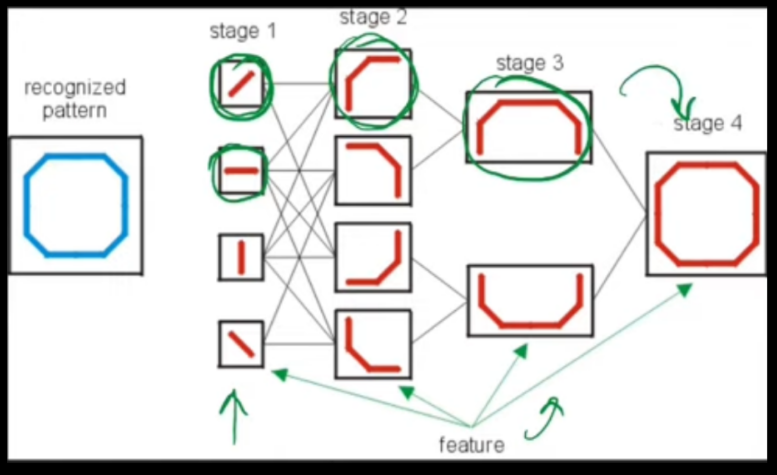
What is a CNN : Convolutional neural network, also known as Convnet, or CNNs are a special kind of neural network for processing data that has a known grid-like topology like time series (1D) or images (2D).



Why not use ANN ?

- High computation cost
- Overfitting
- Loss of imp info like Spatial arrangement of pixels.



→ How different filters try to extract patterns from the feature map (Image)

→ From simple patterns / features (edges) to complex patterns / features.

Convolution operation

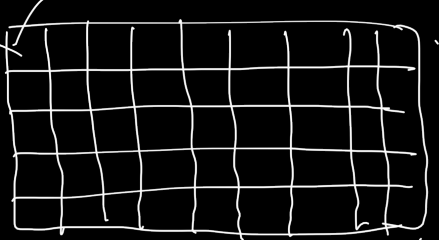
Basic of images

Pixels

(0 - 255) Black white

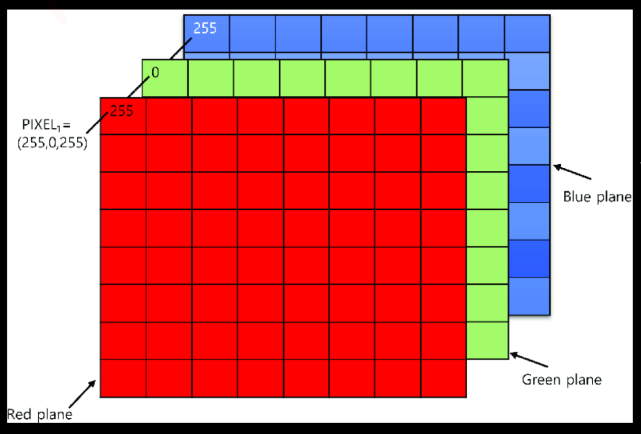
greyscale → Black & white

1 channel



(28 × 28) Pixels

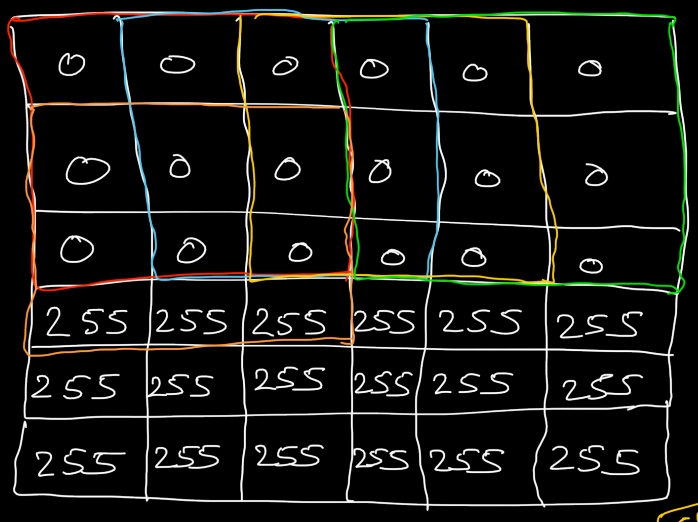
RGB → Color image — 3 channels



→ (228 × 228 × 3) pixels

Edges Detection (Convolution operation)

Edge → change in intensity



6 × 6 (low-resolution image)

Convolution operation (*)

horizontal edge detection

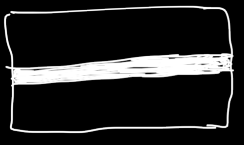
-1	-1	-1
0	0	0
1	1	1

filter / kernel matrix 3 × 3

=

0	0	0	0
255	255	255	255
255	255	255	255
0	0	0	0

4 × 4



Element wise multiplication

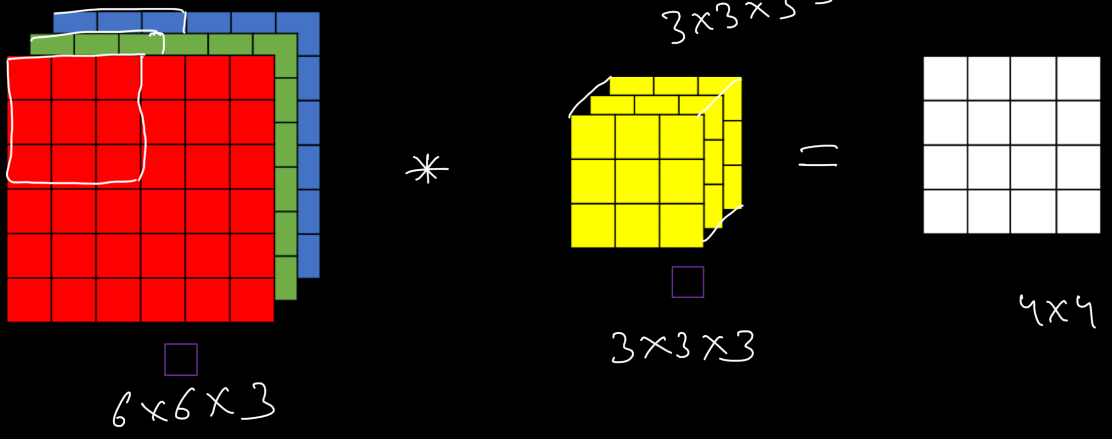
There are so many different filters for different edges but in deep learning we don't need to take care of different filter. They get learn while backprogation. We just define the matrix and rest taken care by backprogation on training.

deeplizard.com

resultant matrix shape : $(n-m+1) \times (n-m+1)$

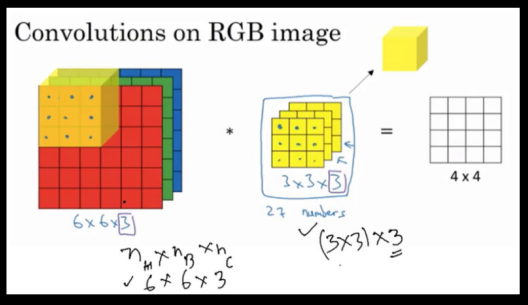
Image (28×28) $n \times n$ → filter (3×3) $m \times m$ → feature map $(28-3+1) \times (28-3+1)$ 26×26

Working with RGB Images

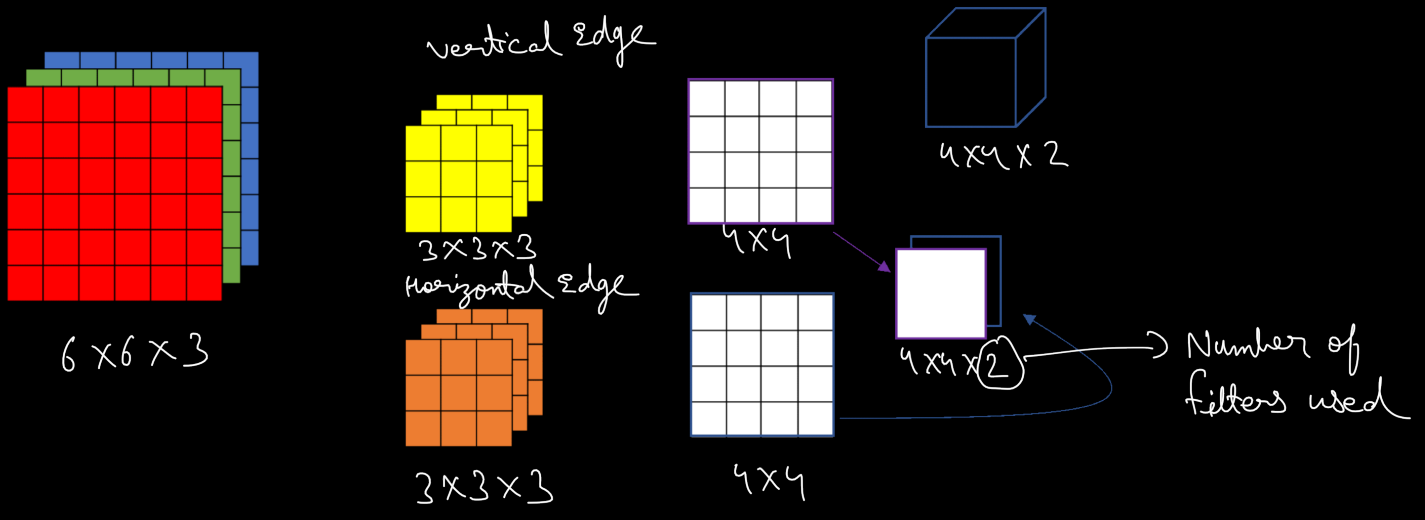


$n \times n \times c \quad * \quad m \times m \times c \rightarrow (n-m+1)(n-m+1)$

↳ Single channel feature map

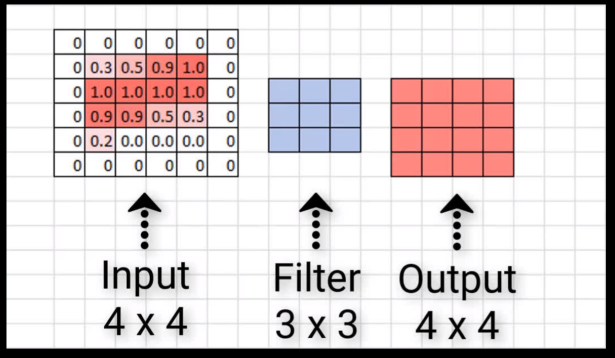


Multiple filters



What is Padding ?

(why → To get feature map with same shape as image)



$(4-3+1+2\times1) \times (4-3+1+2\times1) = 4 \times 4$

Shape of feature map after using Padding

$(n-m+1+2P)$

P → Padding value like 1, 2 ..

Stride

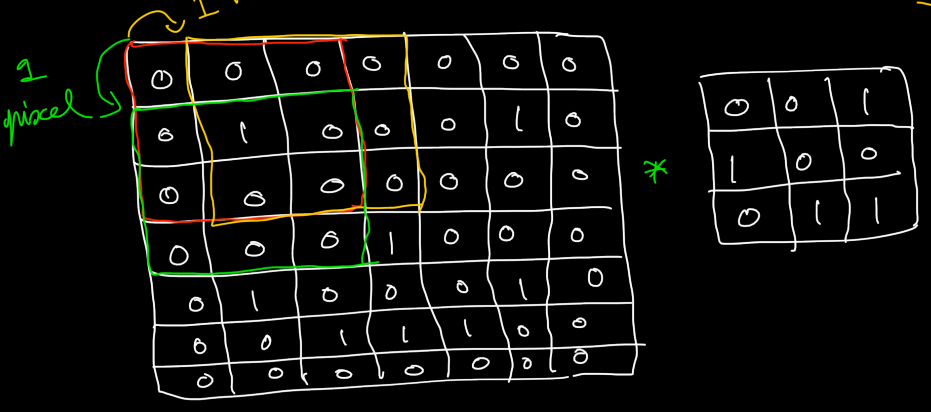
$$\left[\frac{n+2p-m}{s}+1\right]$$

S → Stride

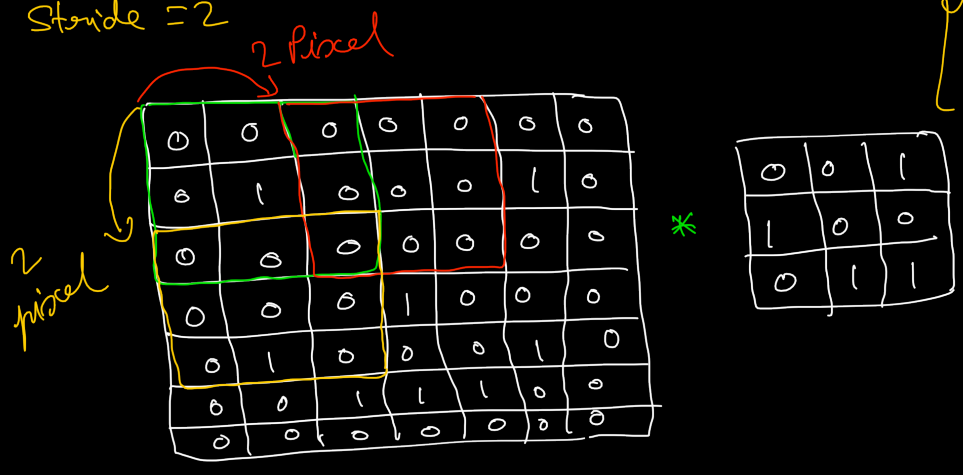
P → Padding

resulting feature map Shape

Stride = 1

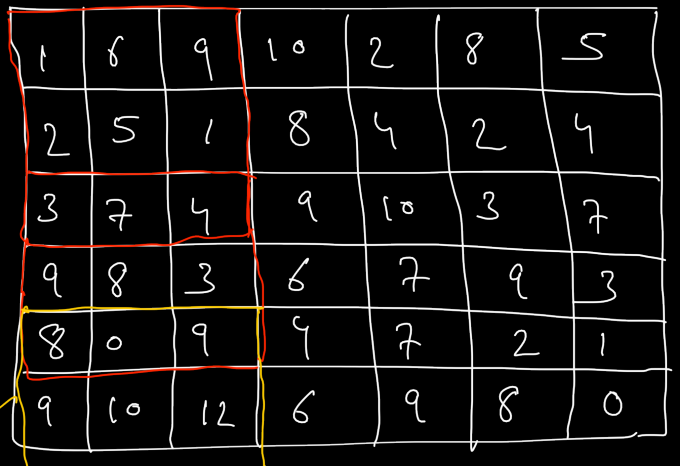


Stride = 2



when we use Stride value more than 1 then we say it Strided Convolution.

Special case

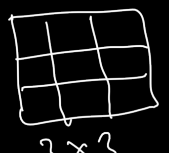


(6 x 7) { rows & Columns are not Equal }

Here Convolution will not happen.

filter = (3, 3)

Stride = 2 then Shape of resulting feature map



$$\text{Row} = \frac{n-m}{s}+1$$

$$= \frac{6-3}{2}+1$$

$$= (1.5)+1 \quad \text{(floor)}$$

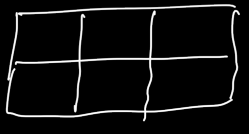
$$= 1+1 = 2$$

$$\text{Column} = \frac{n-m}{s}+1$$

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

$$= 2+1 = 3$$

feature map = (2, 3)

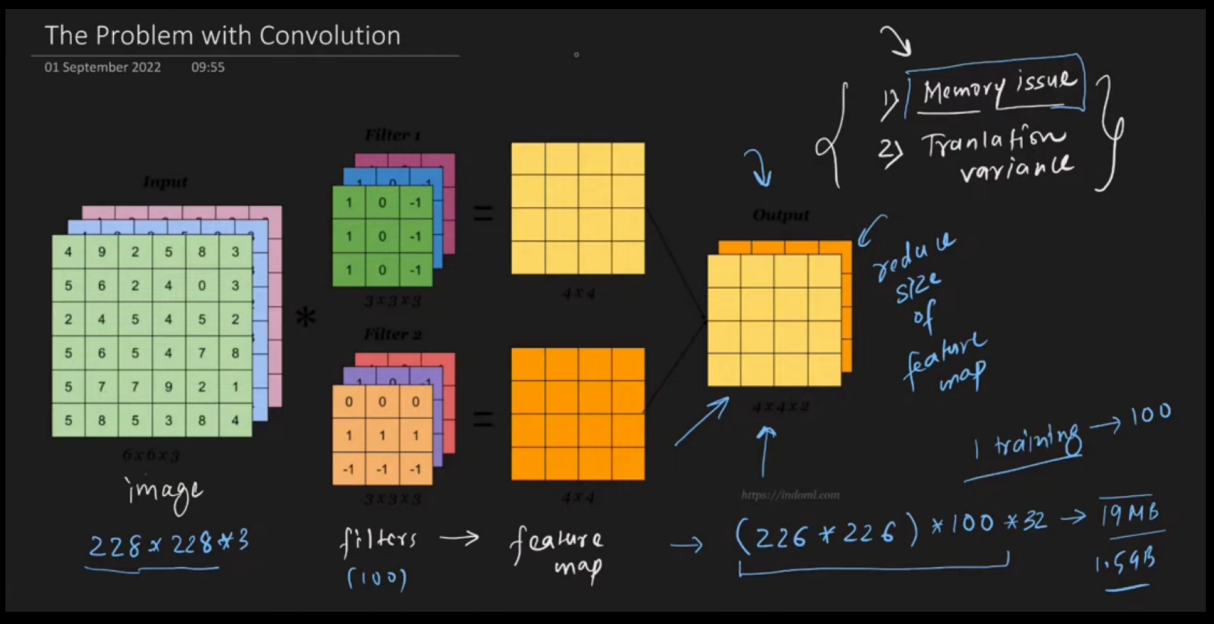


→ Why Strides are required ?

(i) when we need high level features

(ii) faster computation

Pooling Layer in CNN

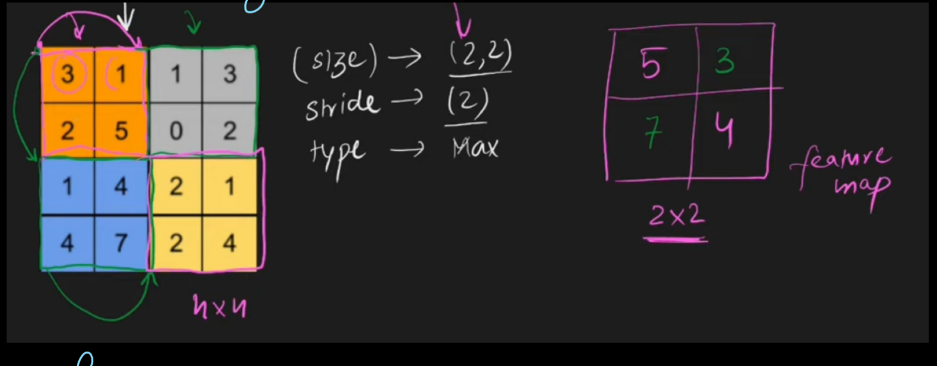


Pooling → down sample our feature map → features become location independent (Translation invariant)

- Max Pooling
- Min Pooling
- Avg Pooling
- L2 Pooling
- Global Pooling

feature map (non-linear)

Max Pooling



feature map

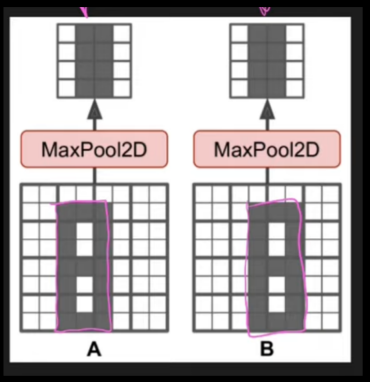
what max-pooling is doing ? In small receptive field only taking dominant features and ignoring others features.

low level details → Eliminate → translation invariant

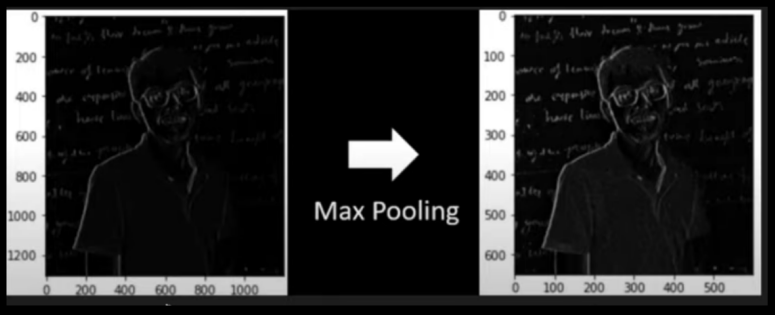
Advantage of Pooling

(1) reduced Size

(2) Translation invariance



(3) Enhanced features
(only in case of Max Pooling)



(4) No need of training for Pooling
It is a aggregation operation like min, max, avg.

Types of Pooling

(i) Max Pooling
(ii) Avg Pooling
(iii) Min Pooling
(iv) Global Pooling

- Global max pooling
- Global avg Pooling

Global Pooling : It will Consider a feature map as a single matrix and from that feature map it will take max or avg depending upon type of pooling.

3	1	1	3
2	5	0	2
0	7	1	3
4	6	2	4

Global max Pooling → 7

→ Global Pooling can be used at the end in place of flatten layer.

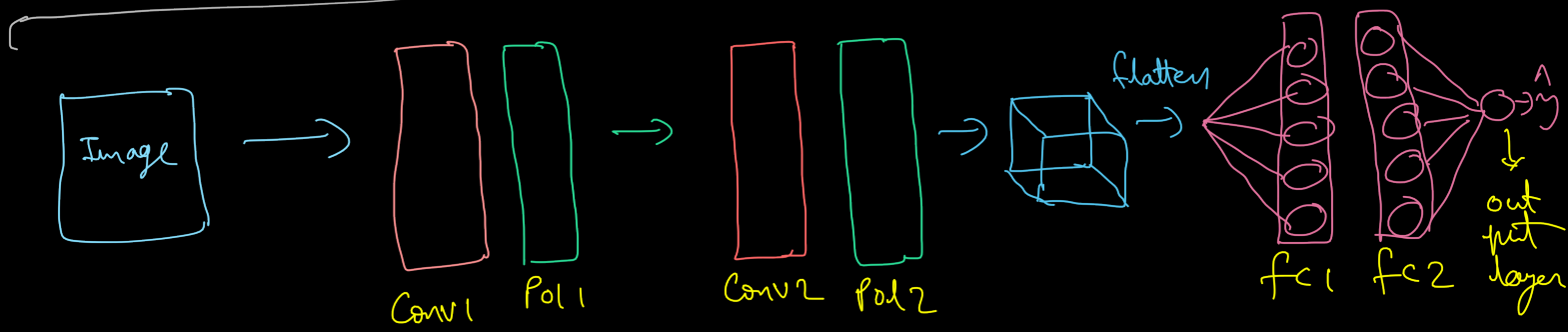
Disadvantage of Pooling

(i) Information loss

(ii) from Pooling translation invariant comes, which is not good in every case such as image Segmentation.

CNN Architecture

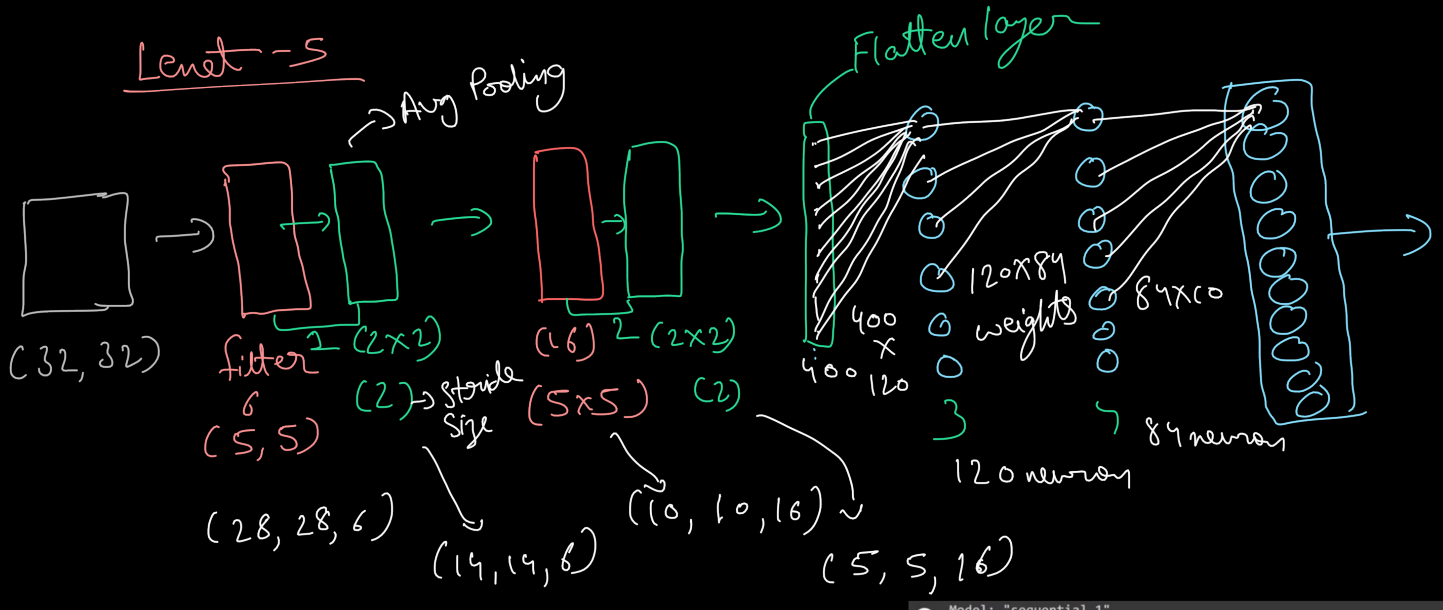
a dummy CNN architecture



# Conv layer	Stride	Fc nodes	activation fn	Batch Normalization
# filter size	Padding	Fc layers	dropout	. . . . .
# No of filter	Pooling			

By changing these things we can create different CNN architecture

Lenet-5




```
model = Sequential()

model.add(Conv2D(6,kernel_size=(5,5),padding='valid', activation='tanh', input_shape=(32,32,1)))
model.add(AveragePooling2D(pool_size=(2, 2), strides=2, padding='valid'))

model.add(Conv2D(16,kernel_size=(5,5),padding='valid', activation='tanh'))
model.add(AveragePooling2D(pool_size=(2, 2), strides=2, padding='valid'))

model.add(Flatten())

model.add(Dense(120, activation='tanh'))
model.add(Dense(84, activation='tanh'))
model.add(Dense(10, activation='softmax'))
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 28, 28, 6)         156
_________________________________________________________________
average_pooling2d_2 (Averag  (None, 14, 14, 6)         0
ePooling2D)
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 10, 10, 16)        2416
_________________________________________________________________
average_pooling2d_3 (Averag  (None, 5, 5, 16)          0
ePooling2D)
_________________________________________________________________
flatten_1 (Flatten)          (None, 400)               0
_________________________________________________________________
dense_3 (Dense)              (None, 120)               48120
_________________________________________________________________
dense_4 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_5 (Dense)              (None, 10)                850
```