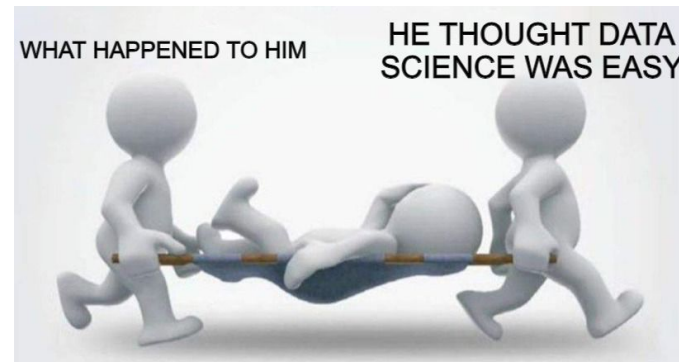


Deep Convolution Network

B.Tech. Data Science, NMIMS

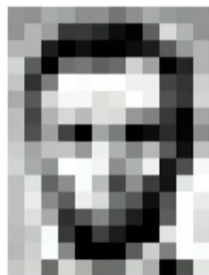
By,

Bilal Hungund, Data Scientist, Halliburton

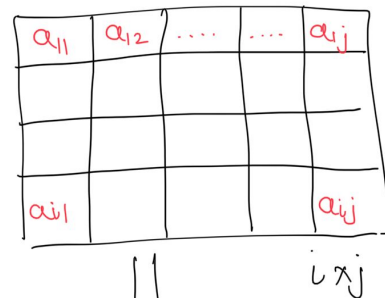


What Computer see?

→ Images are numbers



187	182	174	168	160	152	120	181	172	163	165	166
188	182	182	74	74	62	93	17	176	270	180	164
180	180	80	14	84	6	10	83	48	106	100	181
204	128	7	124	137	111	100	204	196	16	60	180
194	68	133	263	287	288	288	288	287	67	71	281
172	188	287	288	288	214	220	238	288	74	74	288
188	68	178	288	188	218	211	188	188	75	88	188
188	75	188	84	78	188	128	11	81	62	83	188
188	148	181	188	188	287	178	188	188	75	88	181
205	174	181	282	286	287	188	178	288	63	88	284
188	278	128	188	288	287	88	188	78	88	278	281
188	288	187	188	287	278	127	113	88	188	288	211
188	274	178	188	188	188	88	88	8	817	288	211
187	188	288	188	188	188	188	188	188	188	188	211
188	288	287	188	188	188	188	188	188	188	188	211
188	288	188	287	177	188	188	188	188	188	188	211

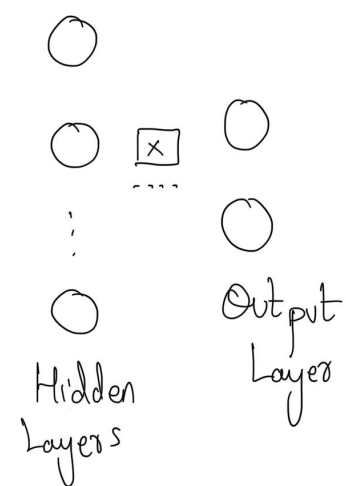


Traditional DNN

Flatten



Input Layer
($i \times j$ neurons)



Types of images

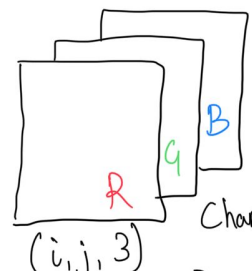
Grayscale

RGB

[Black and White pixels]

#channels

$(i, j, 1)$



Channels

$(i, j, 3)$

Values between (0, 255)

Convolution & operation

0	1	0	1	0	1
0	1	0	1	0	1
0	1	0	1	0	1
0	1	0	1	0	1
0	1	0	1	0	1
0	1	0	1	0	1

6x6 image

Filter (Weights)

-1	1	-1
2	3	2
1	1	1

3x3 Filter

$$n - f + 1$$

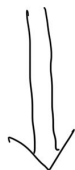
$$= 6 - 3 + 1$$

$$= 4$$

Output Size

4	4	4	4
4
:	:	:	:
:	:	:	:

4x4



Output without padding

Stride = 1
(step size)

With Padding

0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1
0	0	0	1	1	1

4x4 → 6x6
padding

Padding

Filter

1	-1
1	-1



4x4

Pooling

25	48	11	58
192	10	20	110
38	0	9	31
50	8	23	47

Stride = 2
(Recommended
for Pooling)

25	48
192	10

11	58
20	110

38	0
50	8

9	31
23	47

Pooling
⇒



Max Pooling

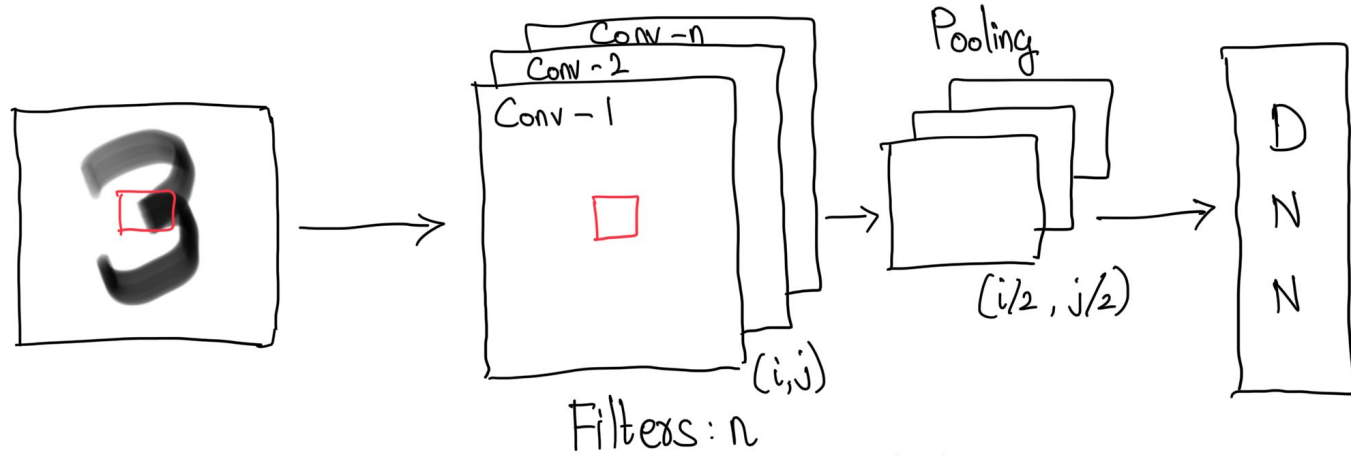
192	110
50	47

69	50
22	28

Average Pooling

Convolution Neural Network (CNN) for Classification

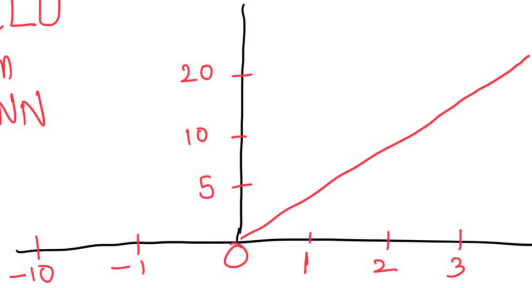
 `tf.keras.layers.Conv2D`
 `tf.keras.activations.*`
 `tf.keras.layers.MaxPool2D`



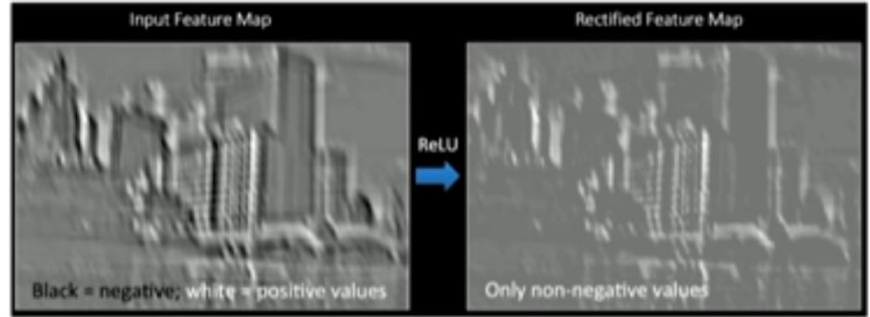
- 1) Convolution: Filters to generate feature maps
- 2) Non-linearity: often relu
- 3) Backpropagation
- 4) Pooling: Downsampling feature maps

Non-Linearity

ReLU
in
CNN



$$g(z) = \max(0, z)$$



Application of CNN

- Classification
- Object Detection
- Segmentation