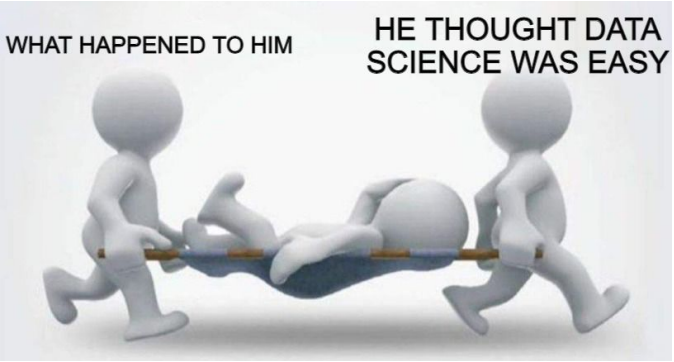
Deep Convolution Network

M.Tech. Data Science, Second Year, NMIMS

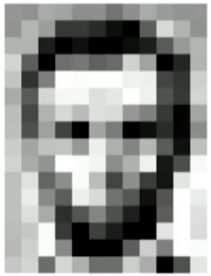
By,

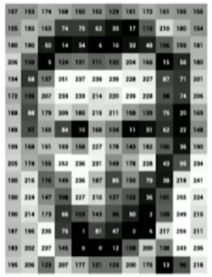
Bilal Hungund, Data Scientist, Halliburton



What Computer see?

→ Images are number





Types of images

Grayscale

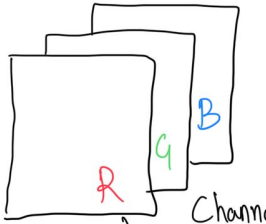
[Black and White pixels]

#channels

↑

(i, j, 1)

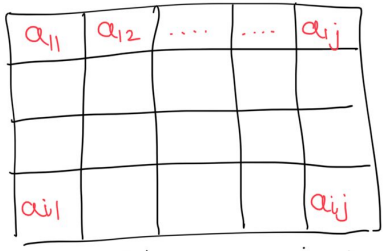
RGB



Channels

(i, j, 3)

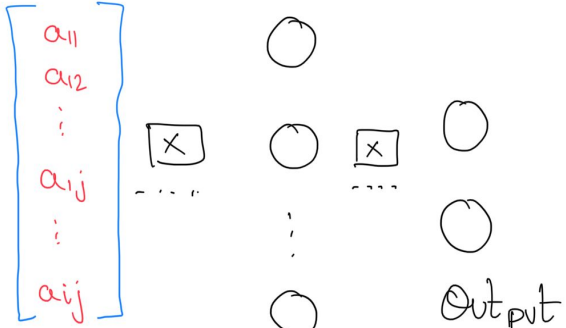
Values between (0, 255)



i×j

Traditional DNN

Flatten



Input Layer (i×j neurons)

Hidden Layers

Output Layer

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

Output without padding

4	4	4	4
4	..	..	..
:	:	:	:
:	:	:	:

4x4

Stride = 1 (step size)

With Padding

$n - 2p - f + 1$

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 $\xrightarrow[\text{padding}]{}$ 6x6

×

filter

1	-1
1	-1

4x4

Pooling

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

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

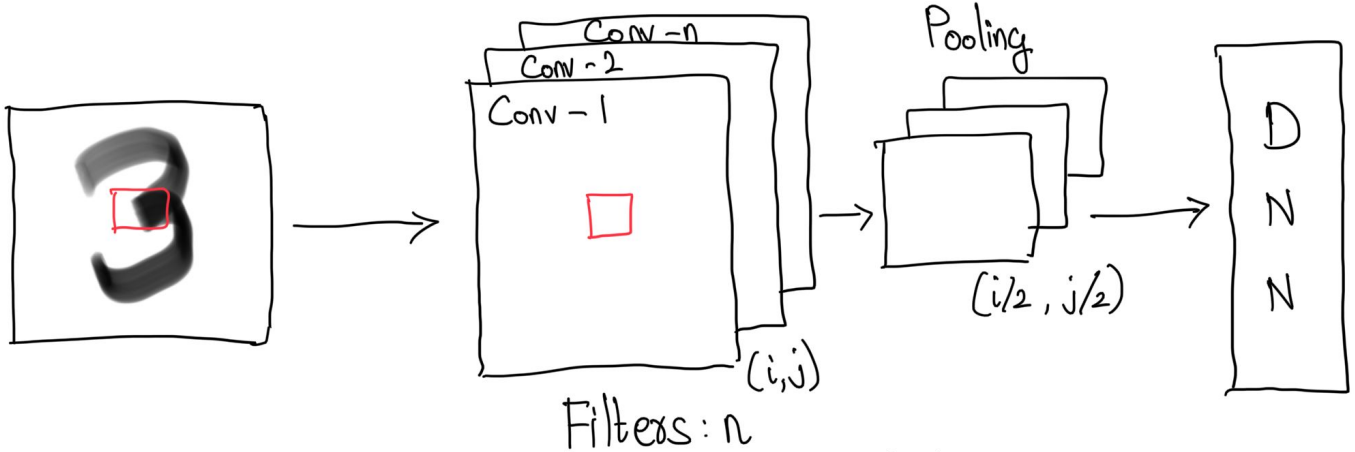
Stride = 2
(Recommended for Pooling)

Convolution Neural Network (CNN) for Classification

tf.keras.layers.Conv2D

tf.keras.activations.*

tf.keras.layers.MaxPool2D



1) Convolution: Filters to generate feature maps

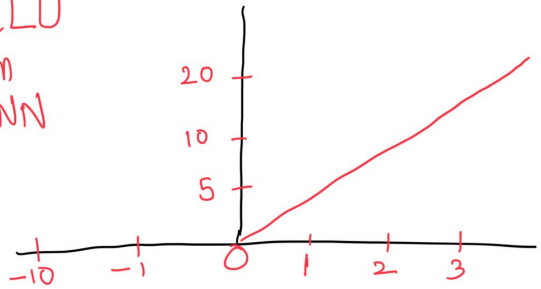
2) Non-linearity: often relu

3) Backpropagation

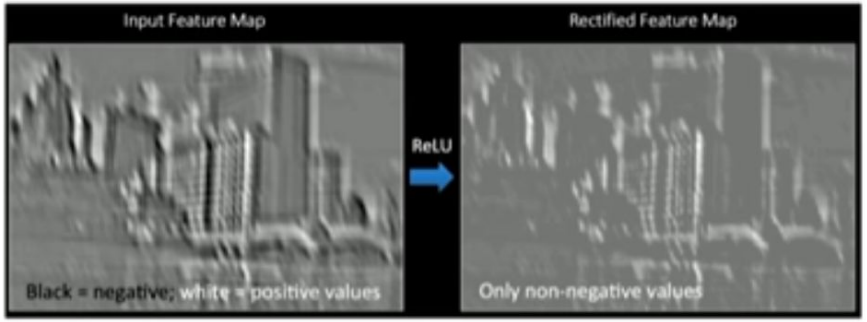
4) Pooling: Downsampling feature maps

Non-Linearity

ReLU in CNN



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



Application of CNN

- Classification
- Object Detection
- Segmentation