2/5/23

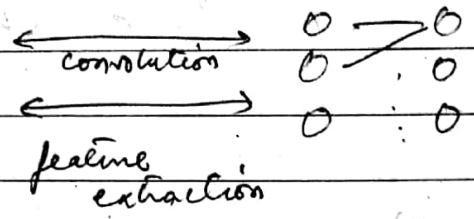
Date: ________ 20__
M T W T F S S

→ CNN

↳ used for classification

↳ Convolution Neural Network

input: cat, dog, truck, car, ⋮



features are feed into the neural network and classification is performed

- Regression = Prediction
- classification

Fourier Transform

$x_1(t) * x_2(t)$

→ Convolution

white, black

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

input ← Image 6x6x1 (grayscale)

* vertical age filter 3x3

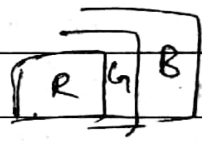
1	0	-1
2	0	-2
1	0	-1

kernel / filter = [ ]

image k saath convolution karaty hain kernel ka

kernel / filter → small size as compared to image

6 x 6 x 3: grid, 3 colors



6x6x1 * 3x3 = what should be the resultant dimension?

stride

padding

Image size		Filter size		color
n	−	f	+	1
6	−	3	+	1 = 4

↳ 4 x 4

256 x 256 x 3

min	max
0	255
0	1 — normalized

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

↳ horizontal age filter

* There's difference b/w convolution and matrix multiplication

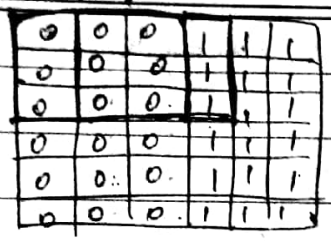
dot product

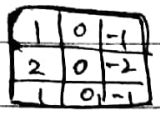
kernel window will move horizontally (stride of 1)

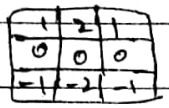
$S = 1$

Date: ____ 20__

M T W T F S S







=

0	-4	-4	0
0	-4	-4	0
0	-4	-4	0
0	-4	-4	0

4x4

→ = 0+0+0+0+0+0+0+0+0

= 0

→ = 0+0+0+0+0+0+0+(-1)+(-2)+(-1) = 0-1-2-1

= -4

→ = 0+0+0+0+0+0+0-1-2-1 = -4

→ = 1+2+1+0+0+0-1-2-1 = 4-4 = 0

Now it will slide vertically

→ = 0 → = 0+0+0+0+0+0+0-1-2-1 = -4

→ = 0

→ = 0

1 → Do with stride of 2 (column will be left) / (row will be left)

2 → Do with horizontal aye jiller

1 →

0	-4	0	
0	-4		

2 → 0+0+0+0+0 ... = 0

0+0+0+0+0+0+1+0-1

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

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

0+0+0.

0	0	0	0
0			
0			
0			

Date: ______ 20__
M T W T F S S

replace the 4x4 matrix with max and min values 0 — 255 — or 1

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

white

→ features dominant (extracted).

- 1st layer convolution
- original 6x6 to 4x4
  something is being lost

$n - f + 1 =$ we want 6..

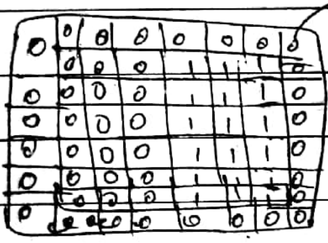
$f = 3$

$n - 3 + 1 = 6$

$n = 8$ — padding

↓ you make a border on all sides of image with value 0.

padding



size badh jayega — lekin value wohi rahegi

* 0 (pehle walay k liye → no padding)

$$\left| \frac{n + 2p - f}{s} + 1 \right| \longrightarrow \text{complete formula}$$

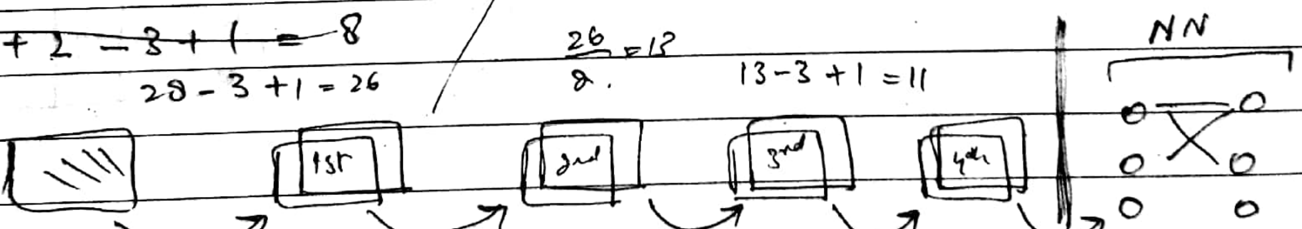
(p = padding)

1	0	1
2	0	-2
1	0	-1

layers

~~8 + 2 − 3 + 1 = 8~~

$28 - 3 + 1 = 26$ $\quad \frac{26}{2} = 13$ $\quad 13 - 3 + 1 = 11$



28x28 — I/p image

32 ⊕ 26x26 ↓ Kernels 32 (3x3)

32 ⊕ 13x13 ↓ Pooling (2x2)

64 ⊕ 11x11 ↓ Kernel 64 (3x3)

64 ⊕ 5x5 ↓ Pooling (2x2)

→ flatten → input to ~~image~~ NN

no. of filters

← Feature Extraction →

generates a big row/column vector

Max-pool
Avg-pool

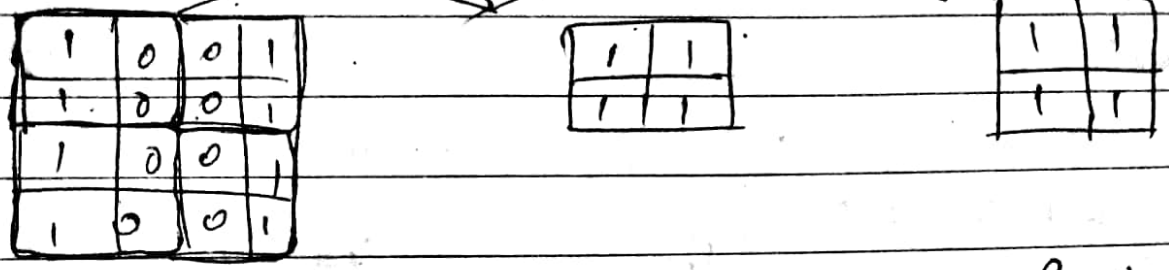
- Pooling
- Flatten

extract the max information

Date: _____ 20__
M T W T F S S

→ Max-pool — pull the max values.

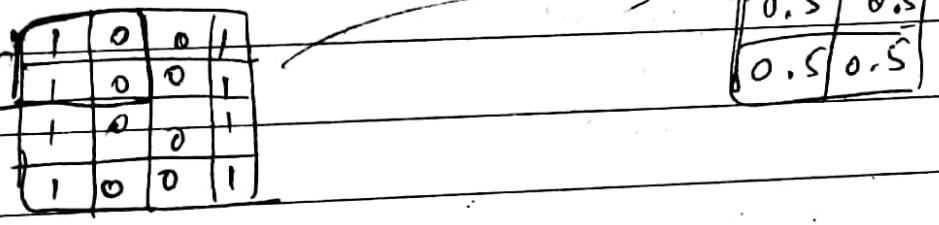
pool (2×2)



for pooling → stride of 2.

Pooling → extract the important features

→ Avg-pool — take average



$\frac{1+0+1+0}{4} = 0.5$

Stride → amount of movement

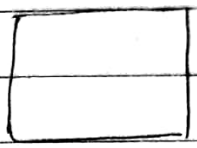
Padding → Preserve the information that is lost when padding is not used

27/10 AML CNN

Date: ____ 20__
M T W T F S S

→ 2×2 filter → scalar value output

→ striding could be 1, 2 ... skip pixels

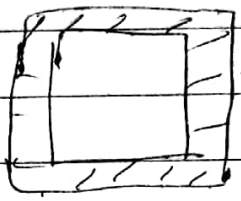


→ padding

↓

image k edges pe information hoti hai which is lost in striding

padding = 2, 3 ....



→ pooling

10 filters mean 10 features

depth of the picture → RGB value → channels

↓ 3 channels

↓ we normally keep it 1

single channel

32, (25, 25, 8)

$$\frac{n + 2p - f}{s} + 1$$

n → image size, p → padding, f → filter, s → stride

* padding, striding → hyper params

weights filters value → learning parameters

different filters give you different outcomes

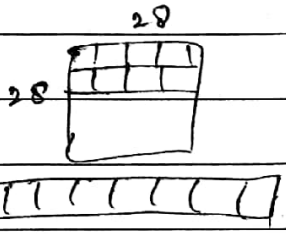
eg: 28×28

↳ fully connected — how many neurons?

28 × 28 → flatten → 784

↓

first row k saath next row concatenate kardi and so on



complex function → non linearity

simple function → linearity

~~complexity~~

Date: ____20__
M T W T F S S

→ Avg pool

→ Pooling Layer
- downsampler → makes representation smaller and manageable

→ Max pool
↳ convolution matrix ko pool kardiya
↳ goal is to downsample
↳ no information is lost → max information is preserved

invariance → actual information change na ho
↳ eg: translation, rotation
↳ Δ ▷

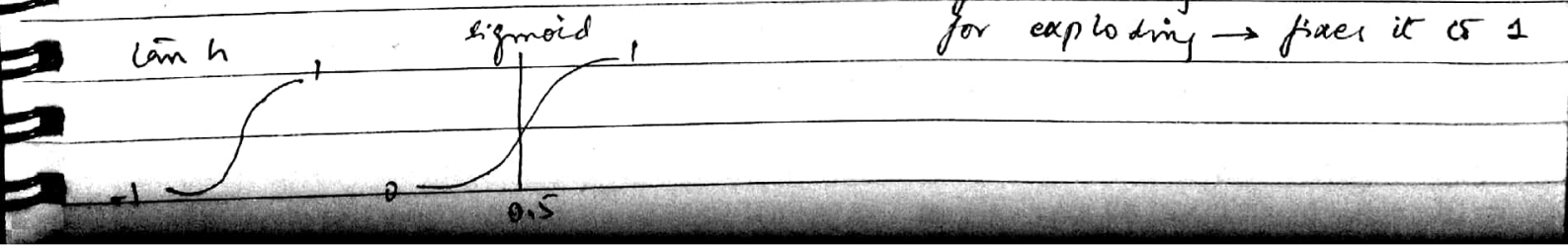
→ FC Layer
pool ka result → flatten → FC
↳ the value in a cell is provided to every neuron
activation function → RELU
↓
why?
hidden layers pe ziada tar RELU use horaha hota hai
- vanishing gradient problem
- Exploding gradient problem
} → Problems in Backpropogation
↳ to learn weights
optimize weights

gradient close to 0 jisse weights update nahi horahay hotay

→ gradient boost ziada bach jaata hai
only 0 means 0, 0.1 means 1 ⟹ RELU
0.4 is 0 → sigmoid

comes these two problems
↑
RELU
↓
for vanishing
for exploding → fixes it to 1

relu



Date: ____20__
M T W T F S S

One layer is ~~or~~ Convolutional layer + RELU + Max pooling
↳ 24 filters

Feature Selection → DT is also a feature selection technique
↳ select some features and drop the rest → based on their importance

Feature Extraction
↳ ~~new~~ transform features
actual coordinates change

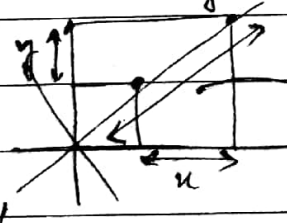
Har feature main different gains hotay hain
don't drop features → e.g. both features are important but we want to keep only one

So, we transform both features into one

e.g. # of rooms	# of toilets	Price

transform into size of the house

samples vectors hotay hain
→ Dimensional reduction
↳ jis column ka variance zyada hai → ilt important features



transform
variance on x-axis is larger than y-axis
var(x) > var(y)

change coordinates

→ Exploding problem
↳ drop bhi karsaktay hain

~~regularizer~~ regularizer → non linearity ko learn karna
tranquilizer → reduce overfitting