Convolution Neural Network (CNN/ConvNet)

- CNN is a class of deep neural networks. It is most commonly applied to image recognition, video recognition, recommendation system and natural language processing.
- It is a feedforward neural network to analyze visual images.
- It uses a special technique called convolution. Convolution is a mathematical operation on two functions that produce a third function that expresses how the shape of one is modified by other.

The three main layers of CNN are

i) Convolution Layer
ii) Pooling Layer
iii) Fully connected Layer

ReLU layer

Convolution Layer

- It is first layer of convolution network. It may be followed by additional convolution layers or pooling layers, the fully connected layer is the final layer.
- With each layer CNN increases in its complexity, identifying greater portions of the image.
- Earlier layer focus on colors and edges. As the image data progresses through the layers, it starts to recognize larger elements or shapes of the object until it finally identifies the intended object.

Convolution Layer :-

- It is core building block of CNN. where majority of occur. Components required in this layer are

i) Input data
ii) filter
iii) feature map

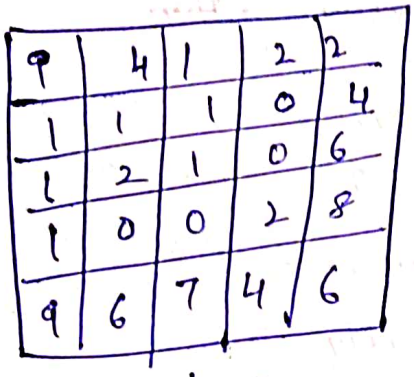
Input data - Image given as a input. Each image will be represented in the 2-D of pixels. where each pixel will be consisting of color values specific to that pixel. If the picture is black and white then gray scale is used, for color colored images RGB scale can be used. RGB Means (Red Green Blue) each pixel will be having a value corresponding to Red color, one value for green color and one value for blue color. It mean each pixel will have 3 values for 3 colors.

Say the size of the image is 200 X 200 pixels. (Image resolution)
Then for RGB scale it becomes 200 X 200 X 3. (This 3 represents color values). (12,000) 4000X3000

ii) filter (Kernel / feature detector)

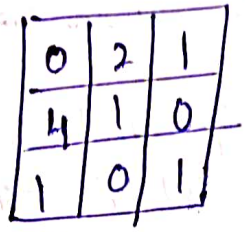
- It is 2-D array of weights, which represents part of the image. different filter sizes can be used for different images. A standard size is 3X3 matrix. The filter is applied to the part of the image and dot product is calculated between input pixel and matrix. This dot product is then fed into an output array. After this filter shifts by 'stride', repeating this process until kernel has covered entire image.

The final output from the series of dot product from input and filter is known as a feature map (activation map or convolved feature)



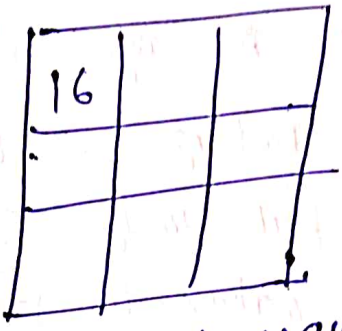
9	4	1	2	2
1	1	1	0	4
1	2	1	0	6
1	0	0	2	8
9	6	7	4	6

Input Image



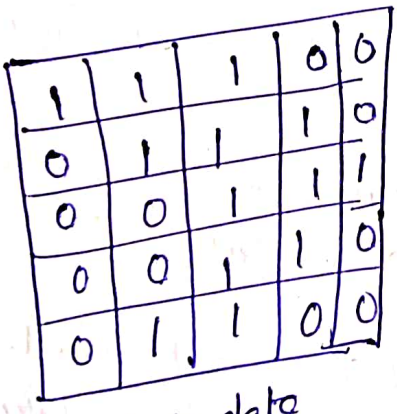
0	2	1
4	1	0
1	0	1

filter



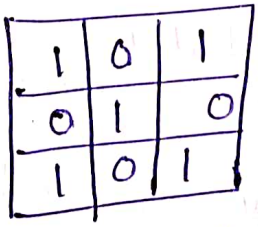
16		

Output array (feature map)



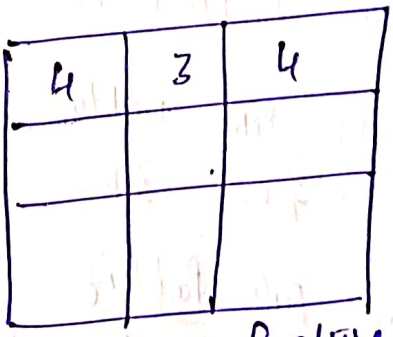
1	1	1	0	0
0	1	1	1	0
0	0	1	1	1
0	0	1	1	0
0	1	1	0	0

Input data



1	0	1
0	1	0
1	0	1

Kernel



4	3	4

Convolved feature

Wt in the feature detector remain fixed as it moves across the image, which is known as parameter sharing. Some parameters, like the weight value, adjust during training through the process of back propagation and gradient descent

- The number of filters affects the depth of the output. for eg. 3 distinct filters would yield 3 different feature map, creating a depth of three.

- Stride is the distance (or number of pixels) that Kernel moves over the input matrix. Stride value of 2 or greater is rare. A large stride yields a smaller output.

- Zero Padding - is usually used when the filters don't fit the input image. This sets all elements that falls outside of input matrix to zeros, producing a larger or equally sized output. ~~There~~ There are three types of padding :-

  i) Valid Padding - Also known as no padding. In this case, the last convolution is dropped if dimensions don't align.

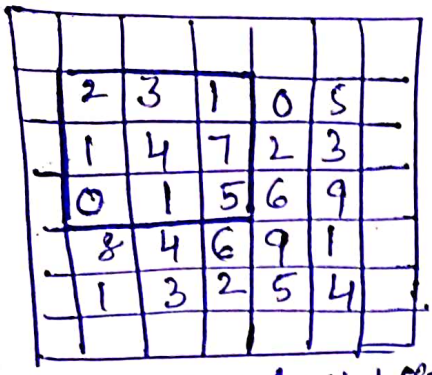
  ii) Same Padding : This padding ensures that output layer has same size as the input layer.

  iii) Full Padding - This type of padding increases the size of output by adding zeros to the border of input.

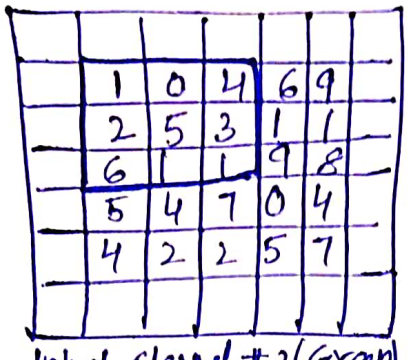
After each convolution operation, CNN applies a Rectified Linear Unit (ReLU) transformation to feature maps introducing non-linearity to the Model.

convolution with multiple channels (RGB)

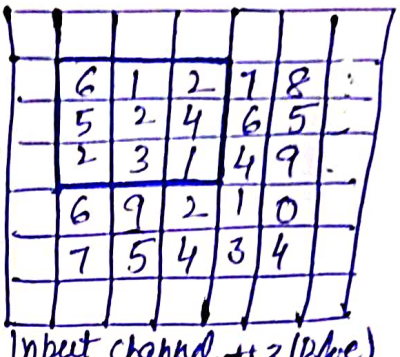
eg.



Input channel #1 (Red)



Input channel #2 (Green)



Input channel #3 (Blue)

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

2 -3 +1 +0 +4 -7 +0 +1 +5

= 7 7

+

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

1+ 0+0+2-5-3+6+0-1

= 0 0

+

0	1	1
0	0	1
1	-1	1

0+1+2+0+0+4+2-3+1

= 7 7

+

15		

(7+0+7+1) = 15

↑

Bias

Here = 3x3x3 (Kernel)

Image is 5x5x3

Pooling Layer -

- It is also known as down sampling. It conducts dimensionality reduction, reducing the number of parameters to the input. The filter is swiped across all the inputs. In this case filter (Kernel) applies an aggregation function to the values within respective field, populating the output array. There are 2 main types of pooling :-

i) **Max Pooling**
As the filter moves across the input it selects the pixel with the maximum value send to the output array. It is more used as compared to average pooling.

3	2	2	1
6	9	8	4
7	1	4	2
0	5	9	1

Max Pooling with 2×2 filter and stride=2 →

9	8
7	9

ii) **Average Pooling**
As filter moves across the input, it calculates the average value within the respective field to send to output array.

6	4	5	2
1	9	1	7
3	9	6	5
8	0	2	2

Average Pooling with 2×2 filter and stride=2 →

5	2.5
5	3.75

Benifits of Pooling Layer
- It reduces complexity
- " Improves efficiency
- Limit the risk of overfitting

Fully Connected Layer

Pixel values of input layer (image) are not directly connected to output layer. in partially connected layers (convolution Layer) In fully connected layer, each node of output layer connects directly to a node in previous layer. This layer performs classification task

- ReLU function is used in convolution and pooling layers. Fully connected layer usually leverage a softmax activation function to classify the inputs appropriately, producing a probability from 0 to 1.

Rectified feature Map

1	4	2	7
2	6	8	5
3	4	0	7
1	2	3	1

Max pooling with 2x2 filter and stride = 2

Pooled feature Map

6	8
4	7

max (3, 4, 1, 2) = 4

Image size = 1920 × 1080 × 3

first layer Neurons = 1920 × 1080 × 3 ≈ 6 Millions

Hidden " " = Let say we keep it ~ 4 millions

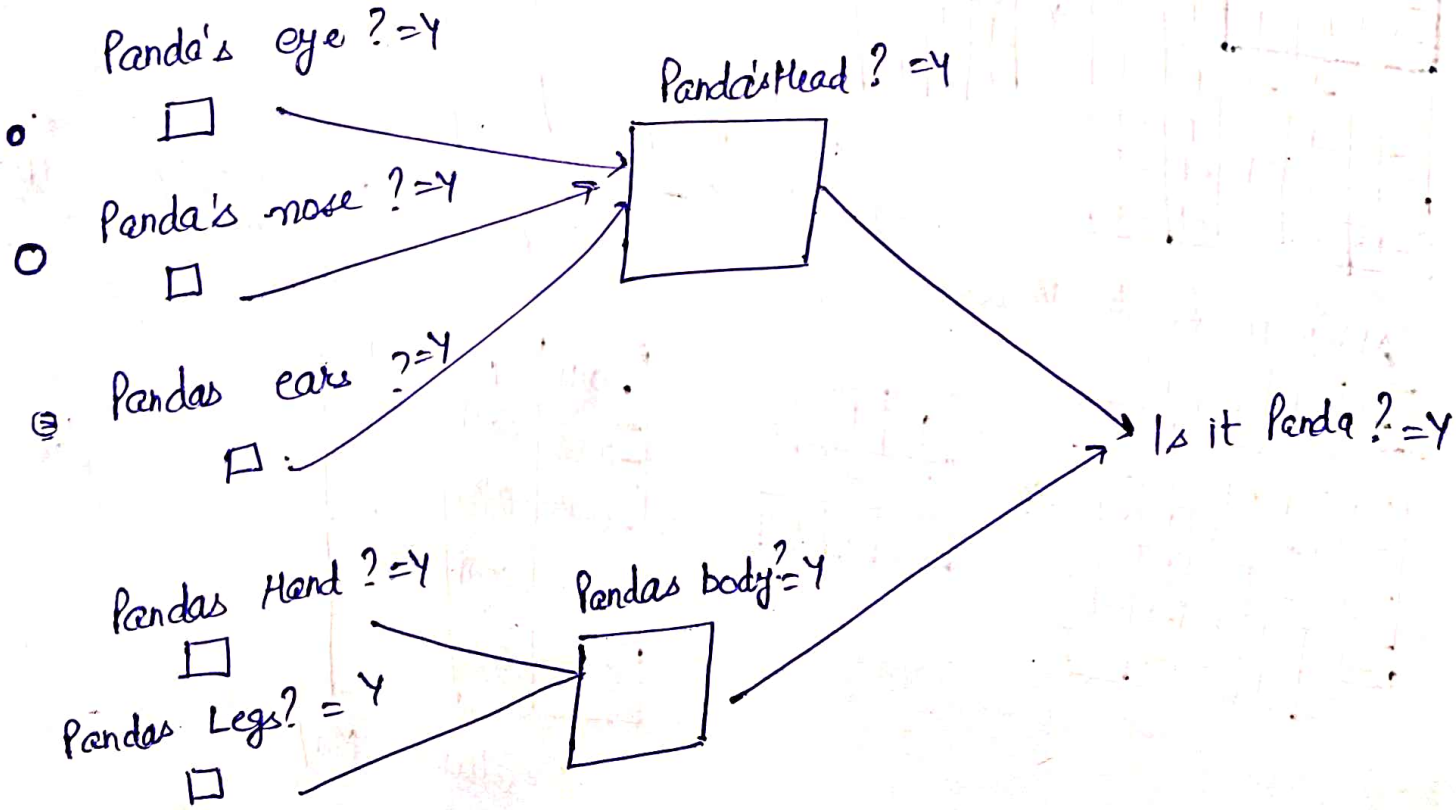
Weights between input and hidden layer = 6 × 4
= 24 millions

Disadvantage of using ANN for image classification :-

i) Too much computations
ii) Treats local pixels same as pixels far apart
iii) sensitive to location of an object in an image.

- Different filters are applied to identify different parts of image (Assume the picture contains 'Panda' animal. So different filters are applied to identify ears, nose, eyes, hands, legs, body of the Panda.

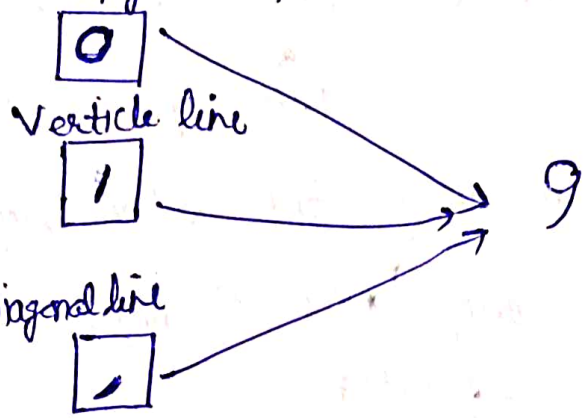
filter for - eye
- nose
- ears

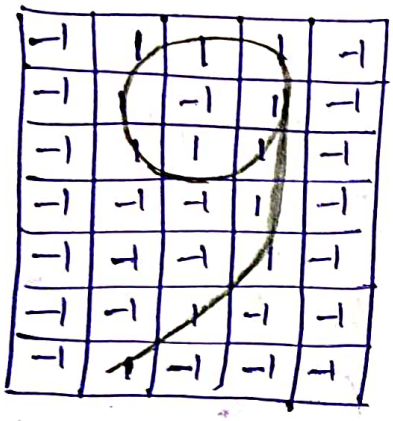


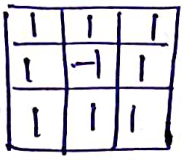
To identify number 9 (Handwritten)

9

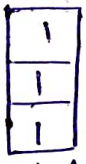
Loopy circle pattern





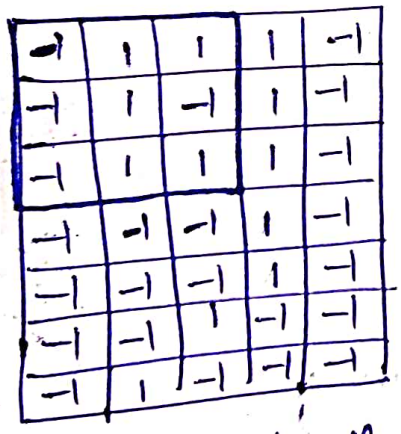


Loopy Pattern filter

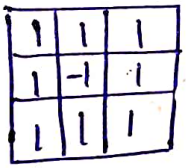


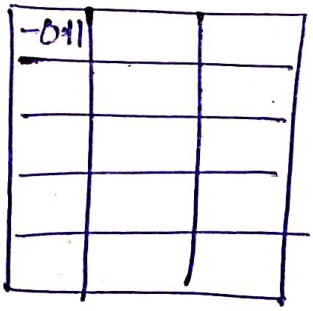
Verticle line filter

Diagonal line filter



*

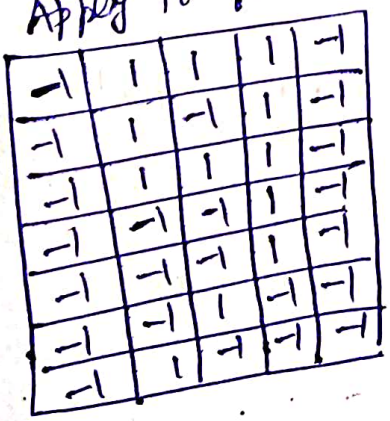




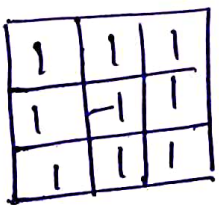
$-1+1+1-1-1-1-1+1+1 = -1 \rightarrow -1/9 = 0.11$

(Average calculation)

Apply it for all values



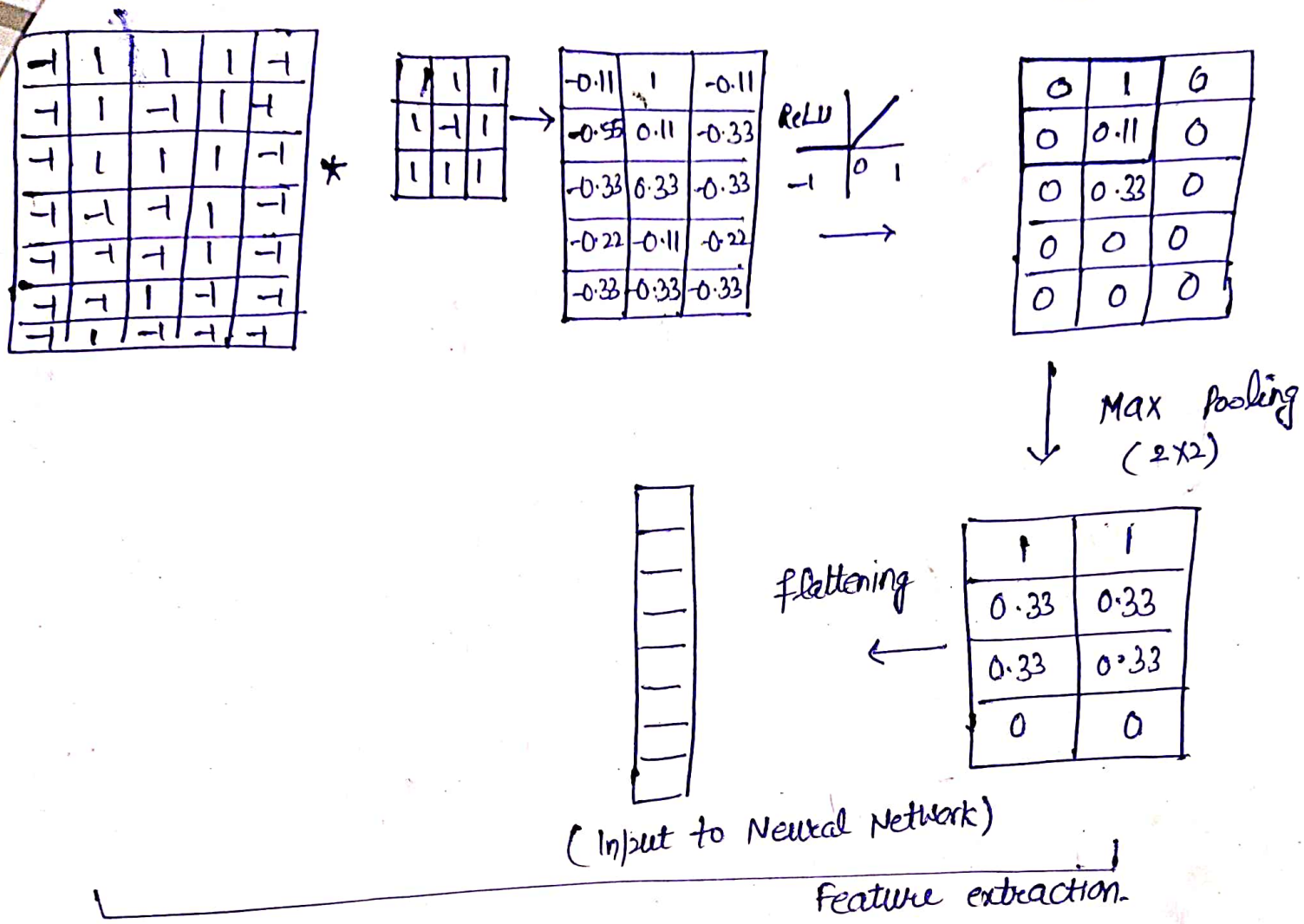
*

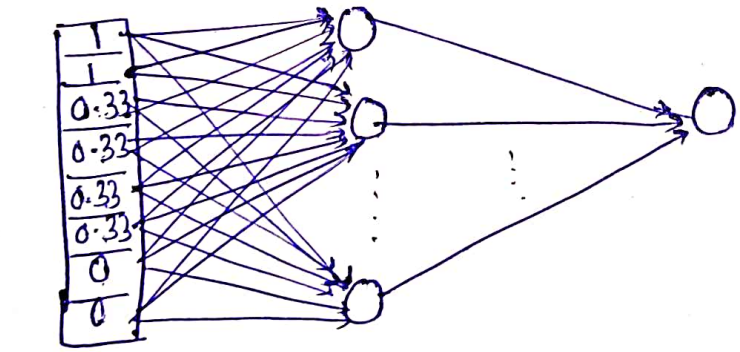


-0.11	1	-0.11
-0.53	0.11	-0.33
-0.33	0.33	-0.33
-0.22	-0.11	-0.22
-0.33	-0.33	-0.33

Feature Map

2(i)

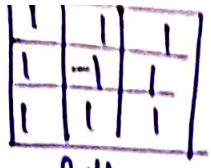
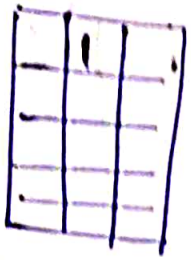




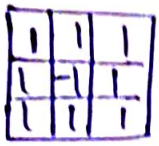
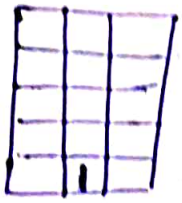
Input Layer

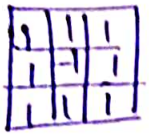
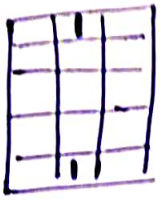
Is this Panda ?

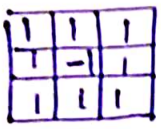
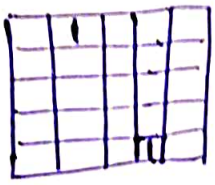
classification

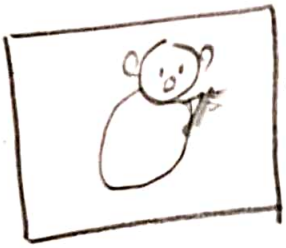
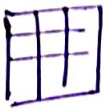
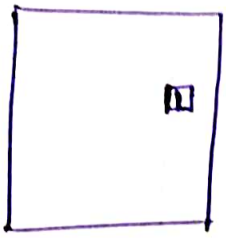
9 *  = 

Loopy Pattern detector

6 *  = 

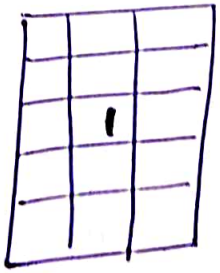
8 *  = 

96 *  = 

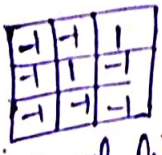
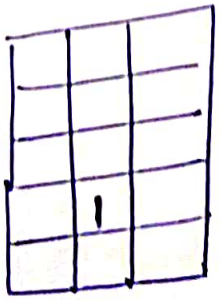
 *  = 

Hand detector

extra examples

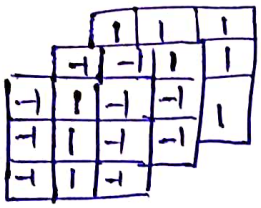
9 *  = 

Verticle line detector

9 *  = 

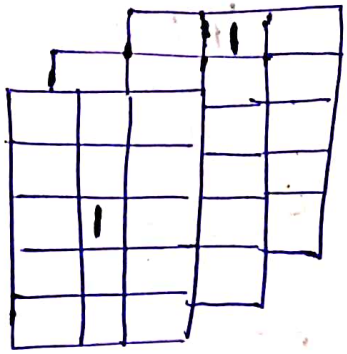
Diagonal line detector.

9 *

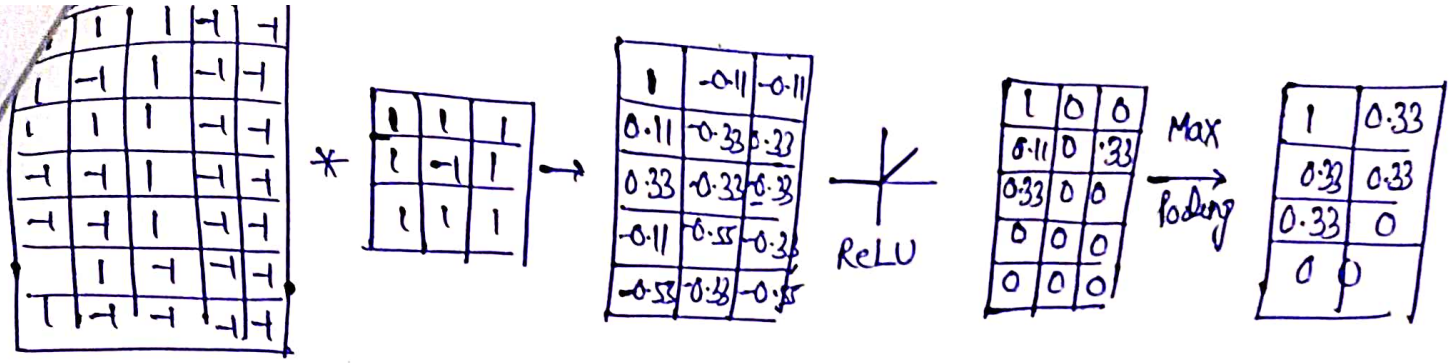


Filters

=



Feature Map



shifted 9 at different Position

Example of Average Pooling

5	1	3	4
8	2	9	2
1	3	0	1
2	2	2	0

Average Pooling →

4	4.5
2	0.75

↓ Max Pooling

8	9
3	2