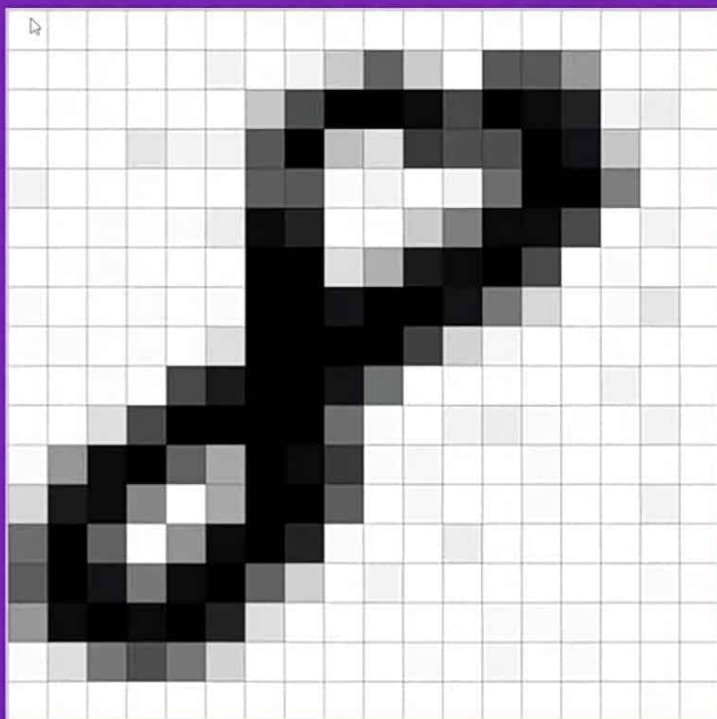


DIGITAL IMAGE

[illegible]

DIGITAL IMAGE

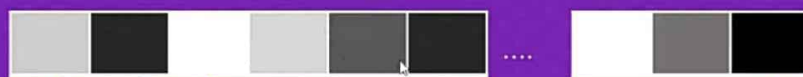


SIMPLE SOFTMAX CLASSIFICATION

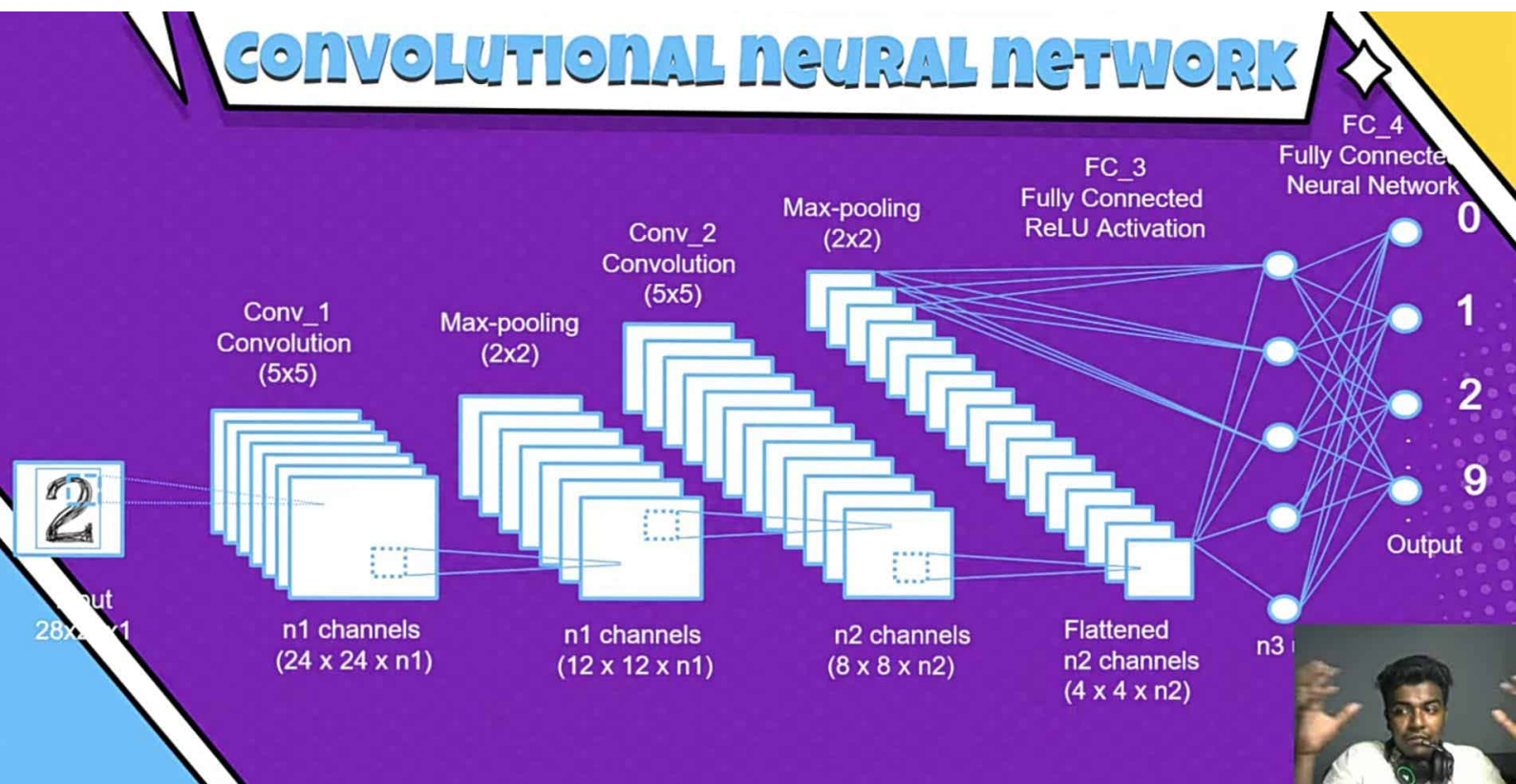


Input
28x28x1

← 784 Pixels →



CONVOLUTIONAL NEURAL NETWORK



CONVOLUTIONAL LAYER

- The Conv layer is the core building block of a Convolutional Network that does most of the computational heavy lifting
- Convolution is the first layer to extract features from an input image
- Convolution preserves the relationship between pixels by learning image features using small squares of input data
- It requires a few components,
 1. Input data
 2. Filter
 3. 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

5 x 5 - Image Matrix

*

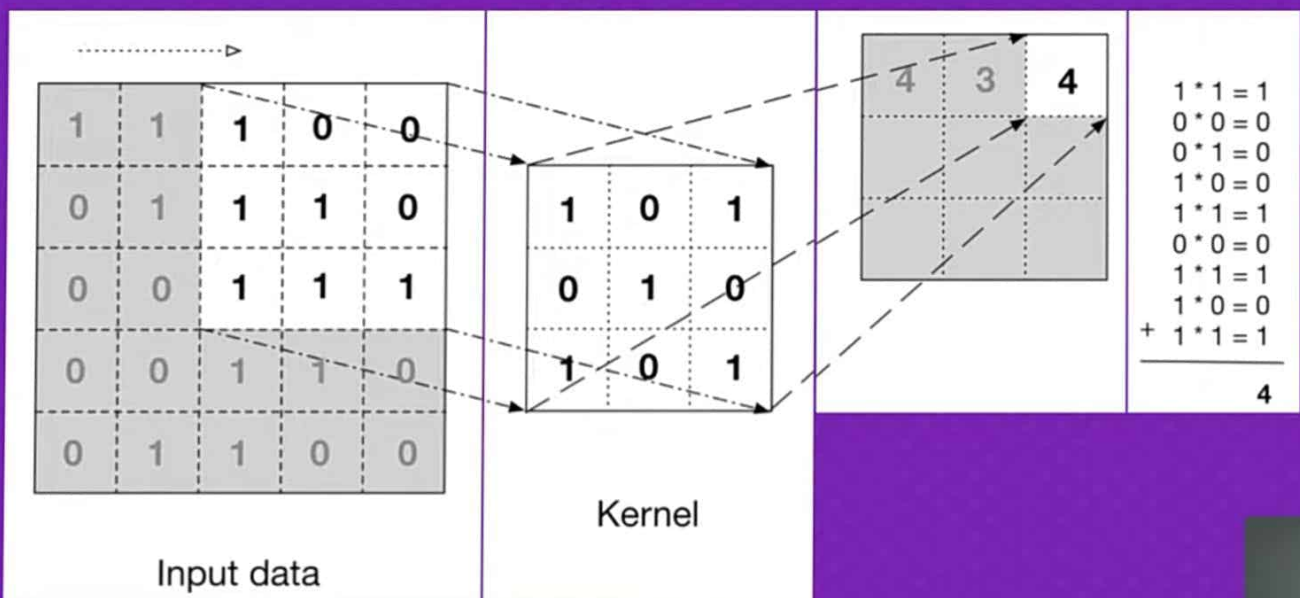
1	0	1
0	1	0
1	0	1

3 x 3 - Filter Matrix

Operation	Filter	Convolved Image
Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	
Gaussian blur (approximation)	$\frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 5 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	



CONVOLUTED FEATURE - GRAY SCALE



CONVOLUTED FEATURE - GRAY SCALE

1 _{x1}	1 _{x0}	1 _{x1}	0	0
0 _{x0}	1 _{x1}	1 _{x0}	1	0
0 _{x1}	0 _{x0}	1 _{x1}	1	1
0	0	1	1	0
0	1	1	0	0

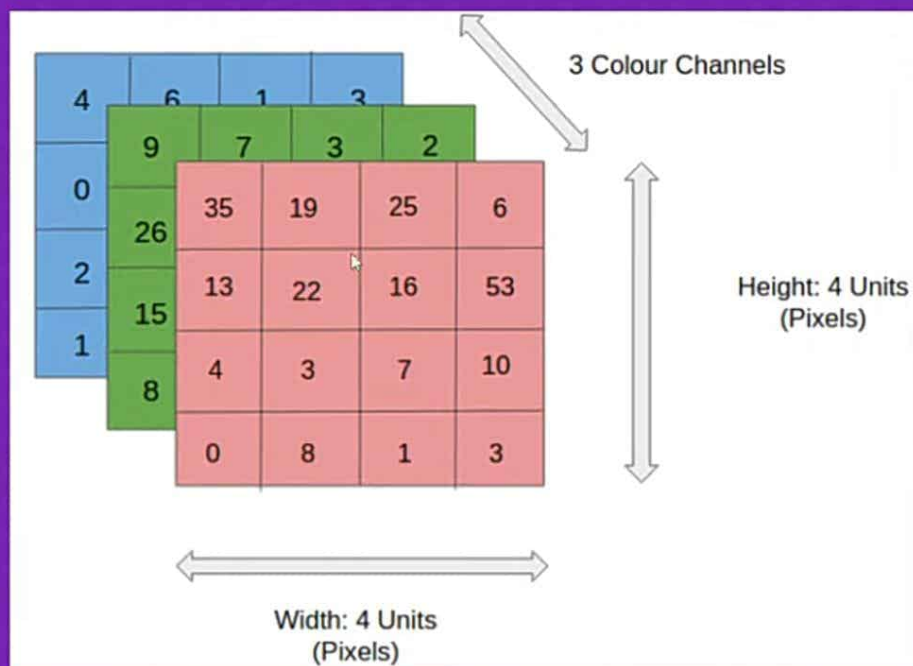
Image

4		

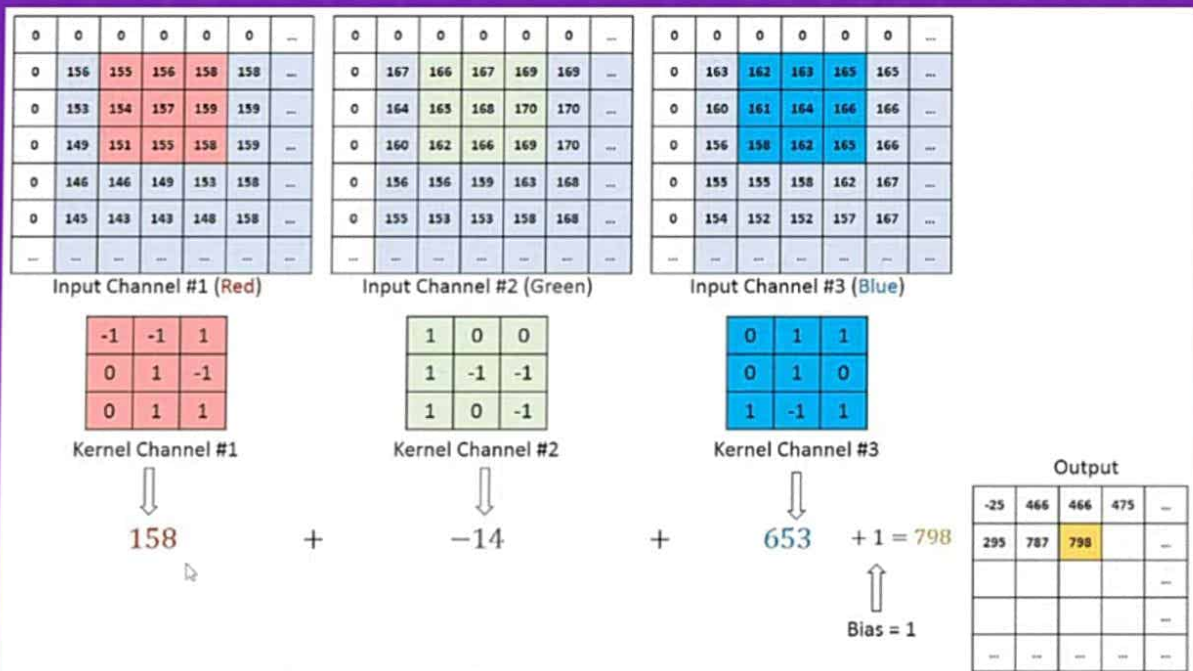
Convolved
Feature



CONVOLUTED FEATURE - RGB SCALE

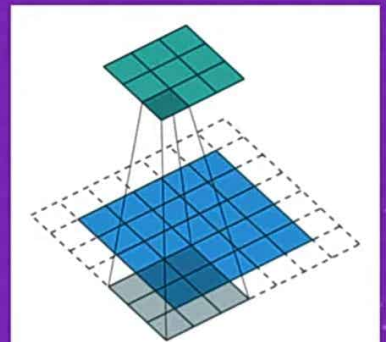


CONVOLUTED FEATURE - RGB SCALE



STRIDES

- Stride is the number of pixels shifts over the input matrix.
- When the stride is 1 then we move the filters to 1 pixel at a time. When the stride is 2 then we move the filters to 2 pixels at a time and so on



PROBLEM IN CONV.

1. Every time after convolution operation, original image size getting shrinks, in image classification task there are multiple convolution layers so after multiple convolution operation, our original image will really get small but we don't want the image to shrink every time
2. The second issue is that, when kernel moves over original images, it touches the edge of the image less number of times and touches the middle of the image more number of times and it overlaps also in the middle. So, the corner features of any image or on the edges aren't used much in the output

PADDING

Sometimes filter does not fit perfectly fit the input image.

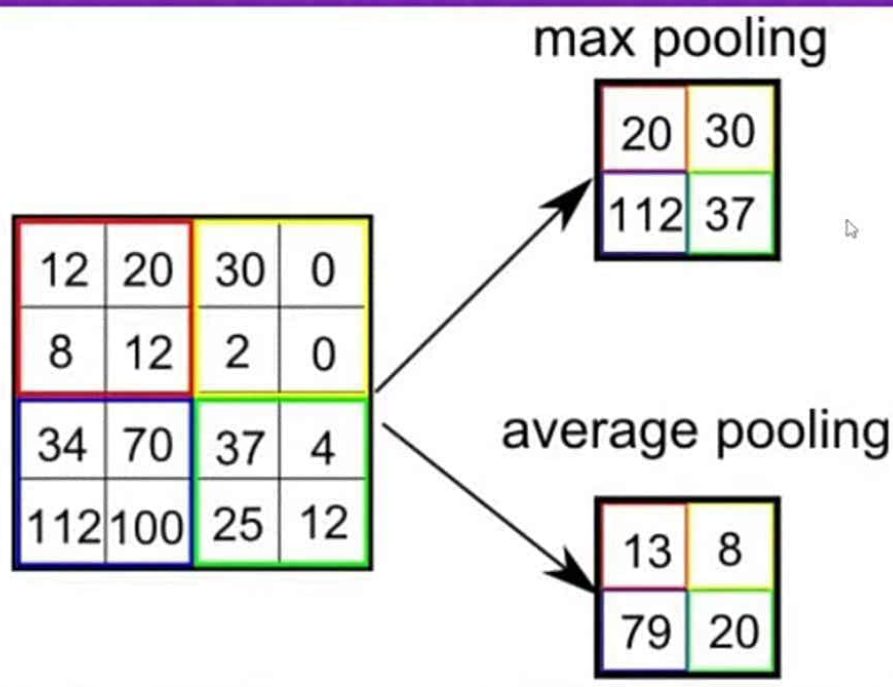
- Zero Padding: Pad the picture with zeros (zero-padding) so that it fits
- Valid padding: This is also known as no padding. Drop the part of the image where the filter did not fit. This is called valid padding which keeps only valid part of the image.
- Same padding: This padding ensures that the output layer has the same size as the input layer
- Full padding: This type of padding increases the size of the output by adding zeros to the border of the input.

POOLING LAYER

- Pooling layers, also known as downsampling, conducts dimensionality reduction, reducing the number of parameters in the input.
- Similar to the convolutional layer, the pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights.
- Instead, the kernel applies an aggregation function to the values within the receptive field, populating the output array.
- They help to reduce complexity, improve efficiency, and limit risk of overfitting
- Max pooling: As the filter moves across the input, it selects the pixel with the maximum value to send to the output array
- Average pooling: As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

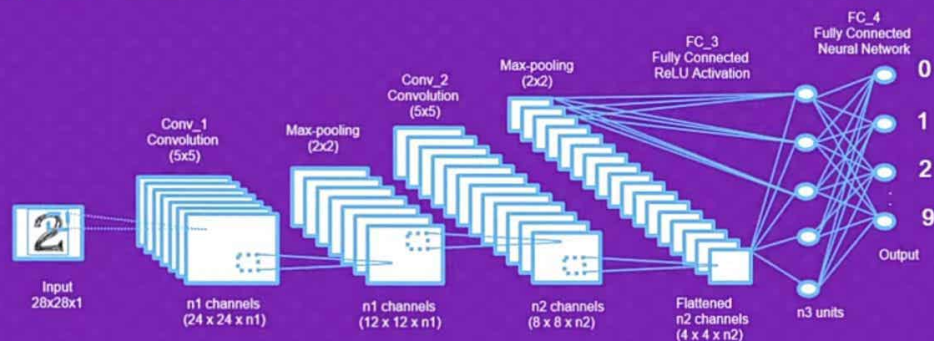


POOLING LAYER



FULLY CONNECTED LAYER

- The layer we call as FC layer, we flattened our matrix into vector and feed it into a fully connected layer like a neural network
- In the above diagram, the feature map matrix will be converted as vector (x_1, x_2, x_3, \dots). With the fully connected layers, we combined these features together to create a model. Finally, we have an activation function such as softmax or sigmoid to classify the outputs as cat, dog, car, truck etc.,



DEEP LEARNING TERMINOLOGY - 1



Tensor Processing Unit (TPU)

- An application-specific integrated circuit (ASIC) that optimizes the performance of machine learning workloads.
- These ASICs are deployed as multiple TPU chips on a TPU device.

DEEP LEARNING TERMINOLOGY - 2



Transfer learning

- Transferring information from one machine learning task to another.
- For example, in multi-task learning, a single model solves multiple tasks, such as a deep model that has different output nodes for different tasks.
- Transfer learning might involve transferring knowledge from the solution of a simpler task to a more complex one, or involve transferring knowledge from a task where there is more data to one where there is less data.
- Most machine learning systems solve a single task.
- Transfer learning is a baby step towards artificial intelligence in which a program can solve multiple tasks.



DEEP LEARNING TERMINOLOGY - 3



Filters

- A filter in a CNN is like a weight matrix with which we multiply a part of the input image to generate a convoluted output.
- Let's assume we have an image of size 28×28 . We randomly assign a filter of size 3×3 , which is then multiplied with different 3×3 sections of the image to form what is known as a convoluted output.
- The filter size is generally smaller than the original image size. The filter values are updated like weight values during backpropagation for cost minimization.