TECHNEEDS

Week-4
Notes

Topics Covered:

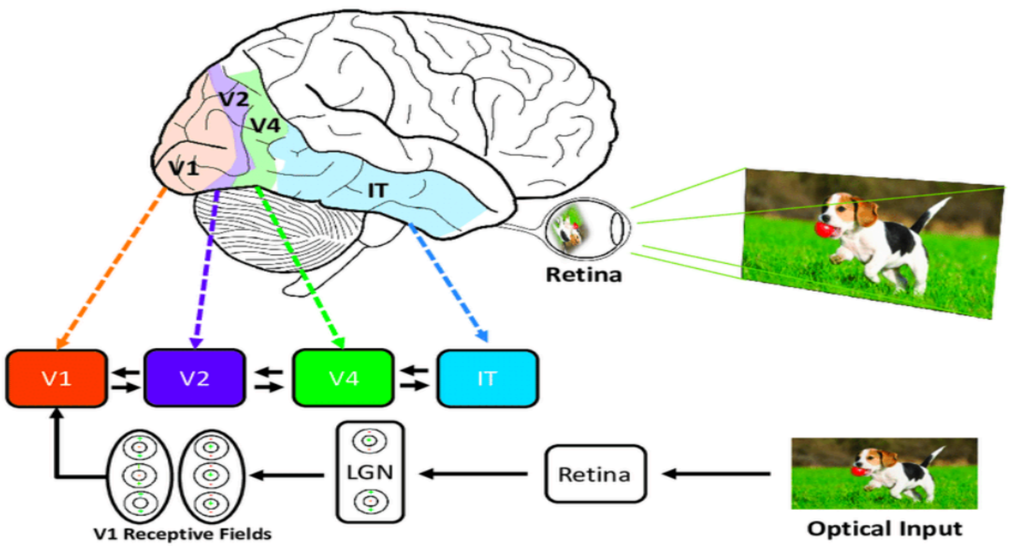
- **CNN**
- **CNN V/S ANN**
- **CONVOLUTION IN CNN**
- **PADDING IN CNN**
- **POOLING IN CNN**
- **FLATTEN IN CNN**
- **ARCHITECTURE OF CNN**
- **BACKPROPAGATION IN CNN**
- **KERAS V/S TENSORFLOW**

What is CNN?

A Convolutional Neural Network (CNN) is a type of deep learning algorithm that is particularly well-suited for image recognition and processing tasks. It is made up of multiple layers, including convolutional layers, pooling layers, and fully connected layers. The visual processing inspires the architecture of CNNs in the human brain, and they are well-suited for capturing hierarchical patterns and spatial dependencies within images.

Inspiration behind CNN?

Convolutional neural networks were inspired by the layered architecture of the human visual cortex, and below are some key similarities and differences:

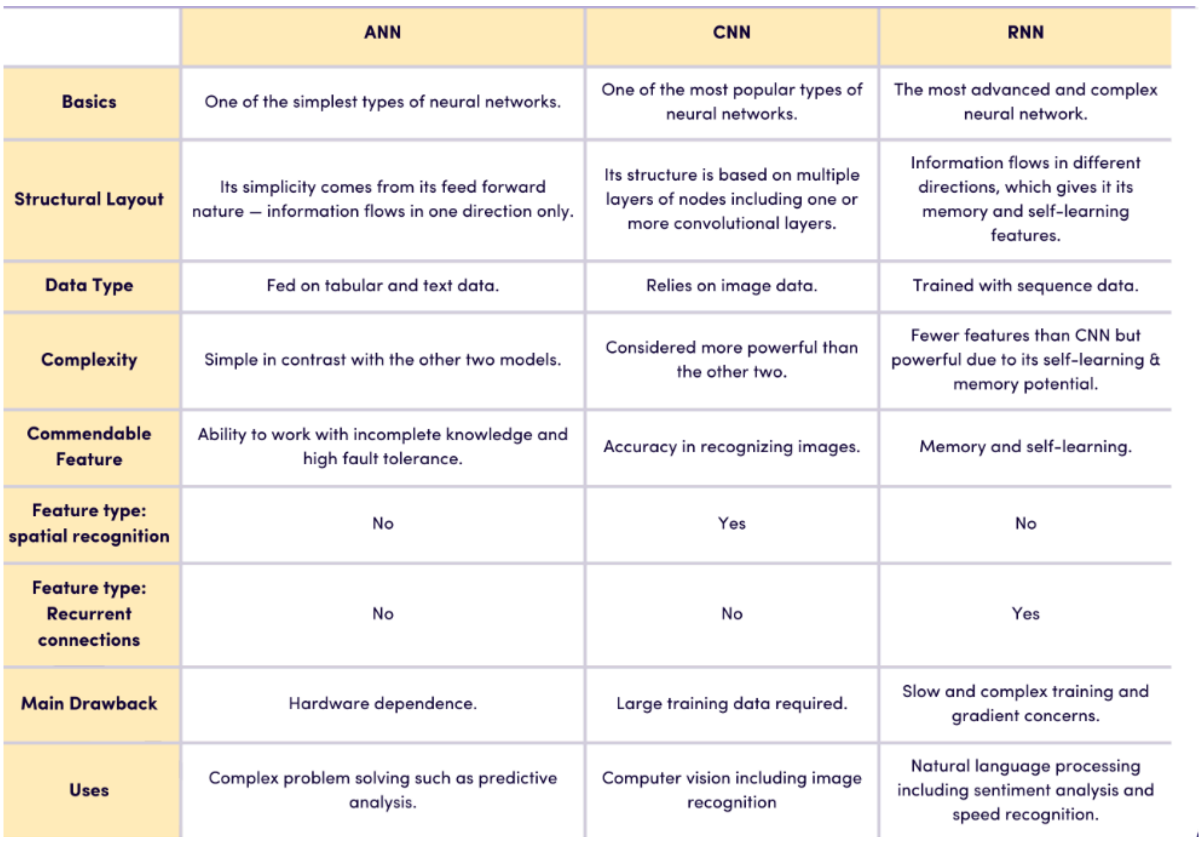


- **Hierarchical architecture:** Both CNNs and the visual cortex have a hierarchical structure, with simple features extracted in early layers and more complex features built up in deeper layers. This allows increasingly sophisticated representations of visual inputs.
- **Local connectivity:** Neurons in the visual cortex only connect to a local region of the input, not the entire visual field. Similarly, the neurons in a CNN layer are only connected to a local region of the input volume through the convolution operation. This local connectivity enables efficiency.
- **Translation invariance:** Visual cortex neurons can detect features regardless of their location in the visual field. Pooling layers in a CNN provide a degree of translation invariance by summarizing local features.
- **Multiple feature maps:** At each stage of visual processing, there are many different feature maps extracted. CNNs mimic this through multiple filter maps in each convolution layer.
- **Non-linearity:** Neurons in the visual cortex exhibit non-linear response properties. CNNs achieve non-linearity through activation functions like ReLU applied after each convolution.

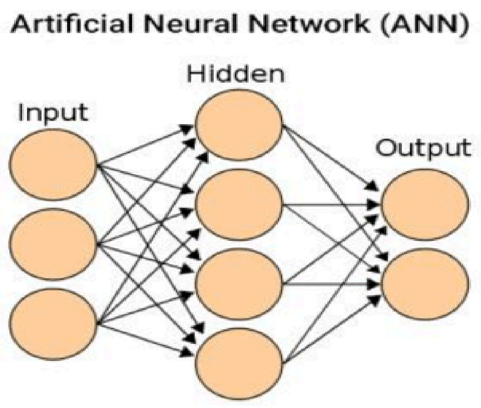
CNN V/S ANN

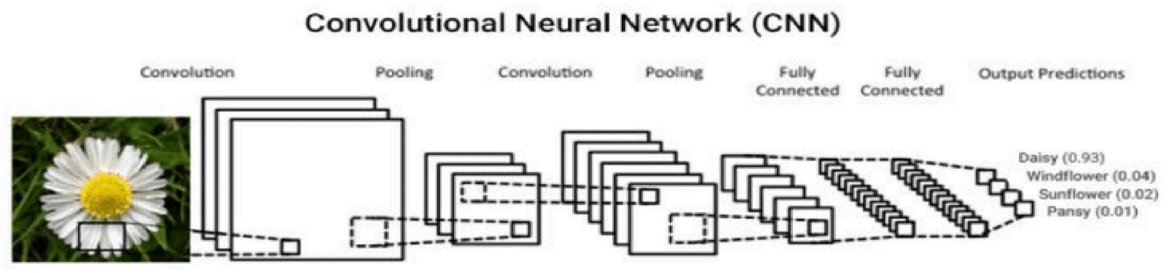
ANN is a general-purpose neural network that can be used for a wide range of tasks, including classification, regression, and pattern recognition.

CNN (Convolutional Neural Network): CNN is a type of neural network that is commonly used for image recognition and computer vision tasks.



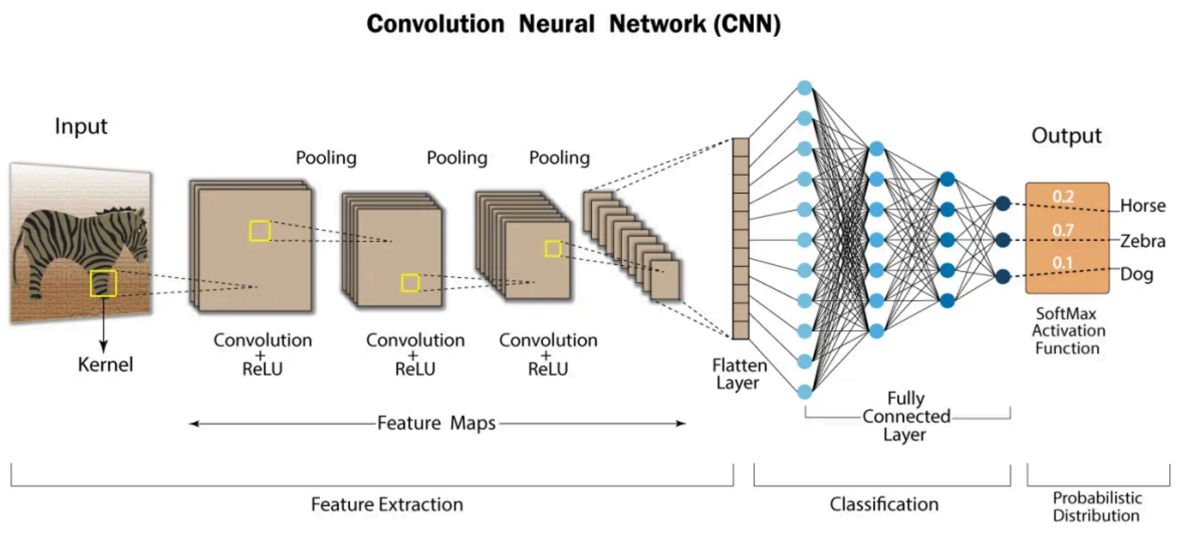
	ANN	CNN	RNN
Basics	One of the simplest types of neural networks.	One of the most popular types of neural networks.	The most advanced and complex neural network.
Structural Layout	Its simplicity comes from its feed forward nature — information flows in one direction only.	Its structure is based on multiple layers of nodes including one or more convolutional layers.	Information flows in different directions, which gives it its memory and self-learning features.
Data Type	Fed on tabular and text data.	Relies on image data.	Trained with sequence data.
Complexity	Simple in contrast with the other two models.	Considered more powerful than the other two.	Fewer features than CNN but powerful due to its self-learning & memory potential.
Commendable Feature	Ability to work with incomplete knowledge and high fault tolerance.	Accuracy in recognizing images.	Memory and self-learning.
Feature type: spatial recognition	No	Yes	No
Feature type: Recurrent connections	No	No	Yes
Main Drawback	Hardware dependence.	Large training data required.	Slow and complex training and gradient concerns.
Uses	Complex problem solving such as predictive analysis.	Computer vision including image recognition	Natural language processing including sentiment analysis and speed recognition.





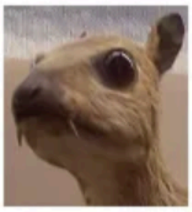
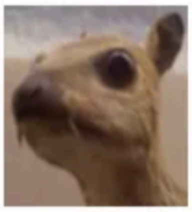
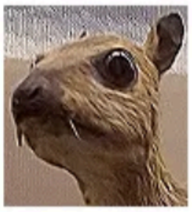
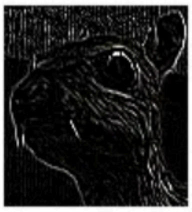
CONVOLUTION IN CNN

Convolution is a mathematical operation that allows the merging of two sets of information. In the case of CNN, convolution is applied to the input data to filter the information and produce a feature map. This filter is also called a kernel, or feature detector, and its dimensions can be, for example, 3x3..



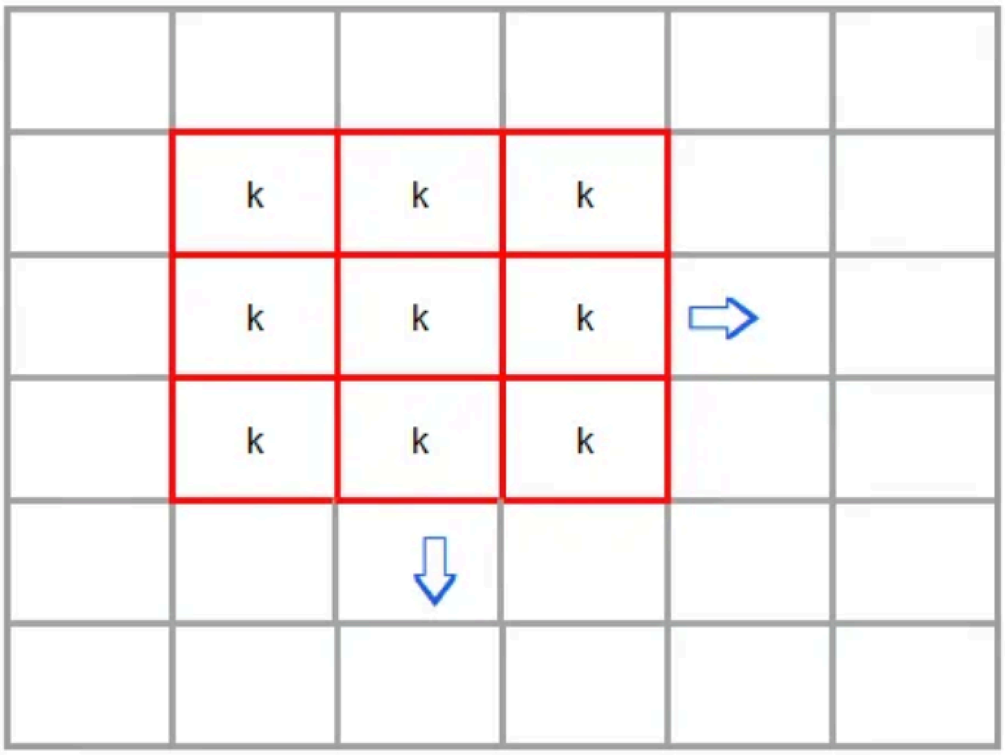
Convolution Kernels

A kernel is a small 2D matrix whose contents are based upon the operations to be performed. A kernel maps on the input image by simple matrix multiplication and addition, the output obtained is of lower dimensions and therefore easier to work with.

Original	Gaussian Blur	Sharpen	Edge Detection
$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$
			

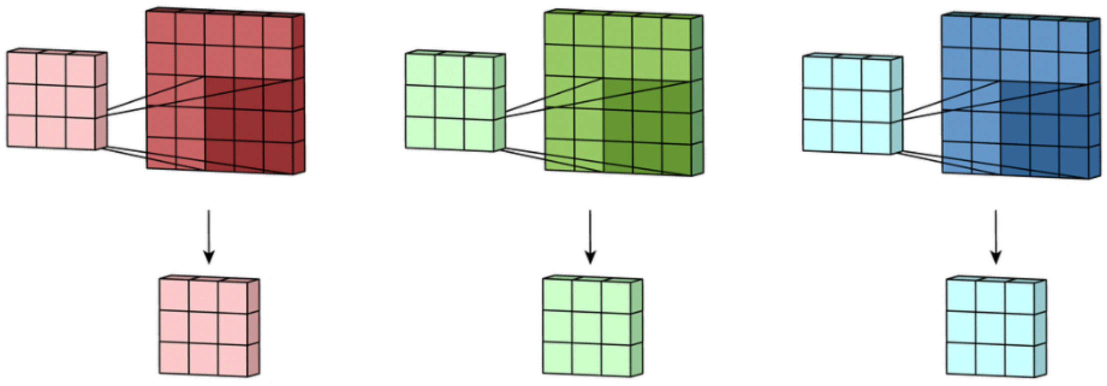
Kernel types

The shape of a kernel is heavily dependent on the input shape of the image and architecture of the entire network, mostly the size of kernels is (MxM) i.e a square matrix. The movement of a kernel is always from left to right and top to bottom.



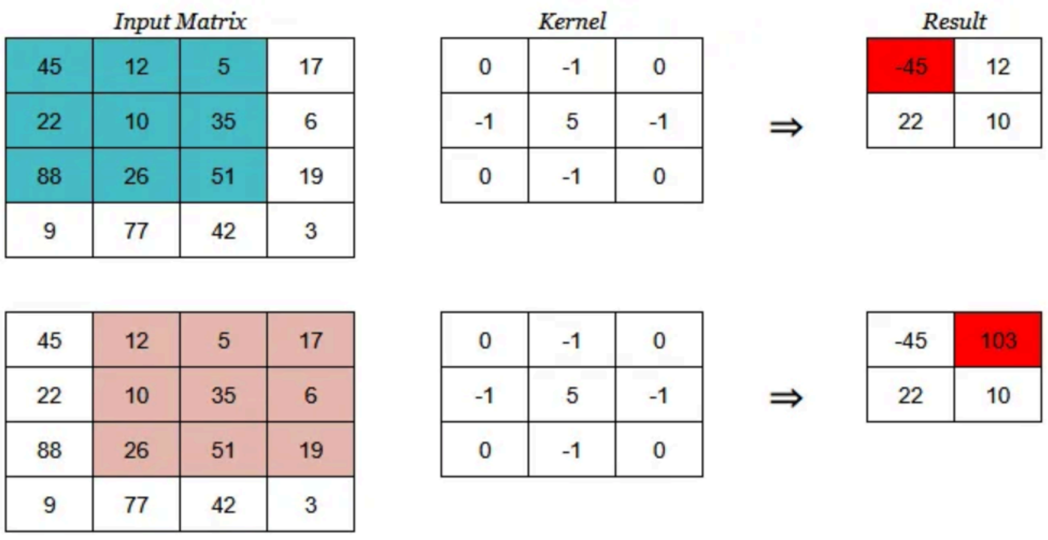
Kernel movement

Stride defines by what step does to kernel move, for example stride of 1 makes kernel slide by one row/column at a time and stride of 2 moves kernel by 2 rows/columns.



Multiple kernels aka filters with stride=1

Convolution in Action



Input Matrix

45	12	5	17
22	10	35	6
88	26	51	19
9	77	42	3

Kernel

0	-1	0
-1	5	-1
0	-1	0

⇒

Result

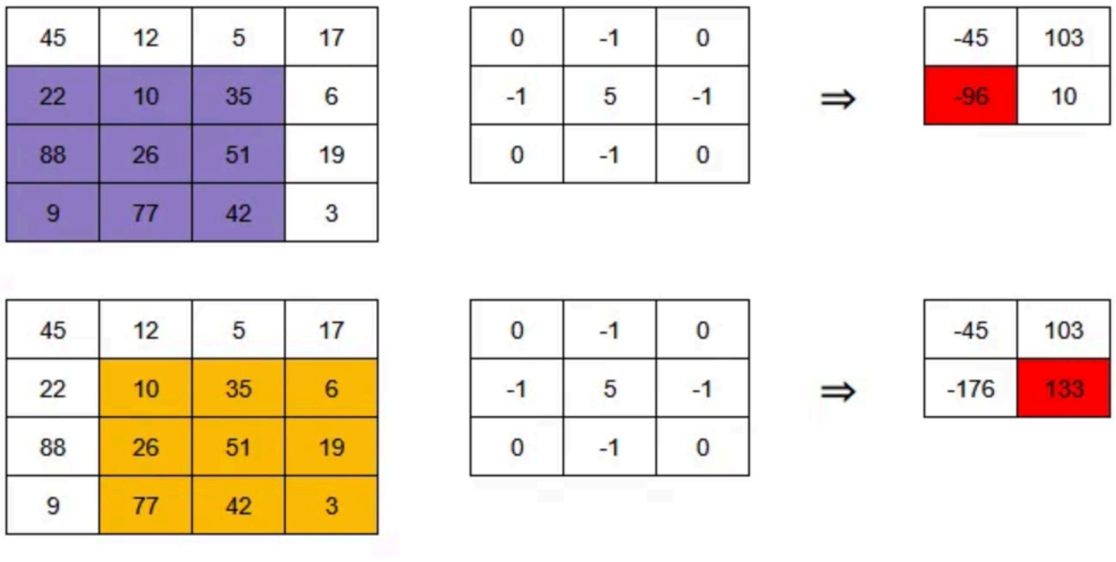
-45	12
22	10

45	12	5	17
22	10	35	6
88	26	51	19
9	77	42	3

0	-1	0
-1	5	-1
0	-1	0

⇒

-45	103
22	10



Convolution in action

Here the input matrix has shape 4x4x1 and the kernel is of size 3x3 since the shape of input is larger than the kernel, we are able to implement a sliding window protocol and apply the kernel over entire input. First entry in the convoluted result is calculated as:

*45*0 + 12*(-1) + 5*0 + 22*(-1) + 10*5 + 35*(-1) + 88*0 + 26*(-1) + 51*0 = -45*

PADDING IN CNN

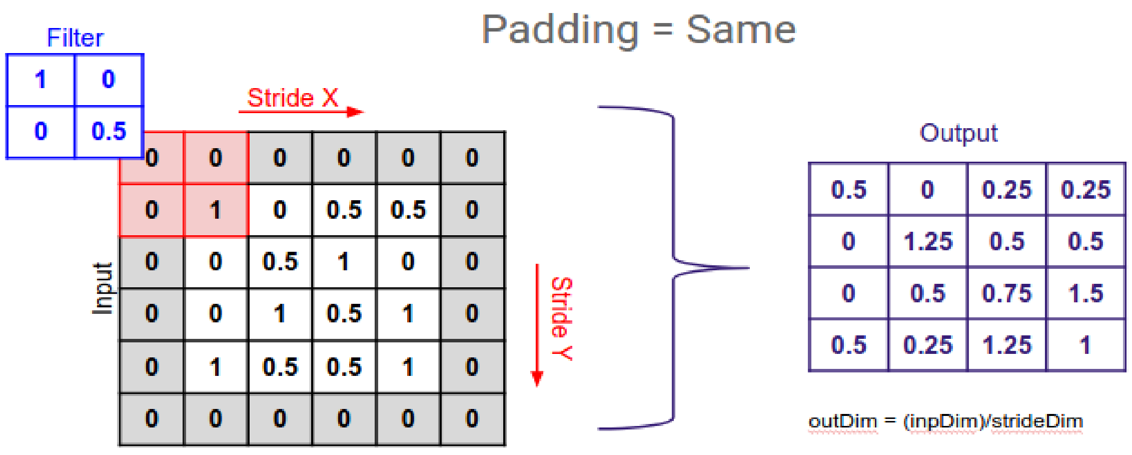
Padding in Convolutional Neural Networks (CNNs) plays a crucial role in shaping the behavior of convolution operations. Understanding padding is fundamental for anyone delving into the realm of deep learning and computer vision. In this comprehensive guide, we will explore what padding is in CNNs, its types, and the significance of each type. So, let’s dive deep into the world of padding in CNNs.

Types of Padding in CNN

In CNNs, various padding types are used, each with its purpose. So, let’s explore them one by one:

1. **Same Padding**: Also, the same padding adds zeros around the input image. To keep the output size equal to the input size after convolution.
2. **Valid Padding in CNN**: Valid padding, also called ‘no padding,’ doesn’t add any extra pixels to the input image. Causing the output feature map to shrink compared to the input. It is handy when you only want the convolution. Portion to focus on the actual data without any additional padding.
3. **Causal Padding:** Causal padding is mainly used in tasks like natural language processing (NLP) and time-series analysis. It adds padding only to the left side of the input sequence. Also, ensuring that each output only depends on the current and past inputs, not the future ones.
4. **Memory Foam Carpet Padding (Non-standard):** This type of padding is used under carpets, even though it’s not directly related to

CNNs. Its mention here shows how padding is used in various contexts. Not just in deep learning, it highlights how the term applies universally.

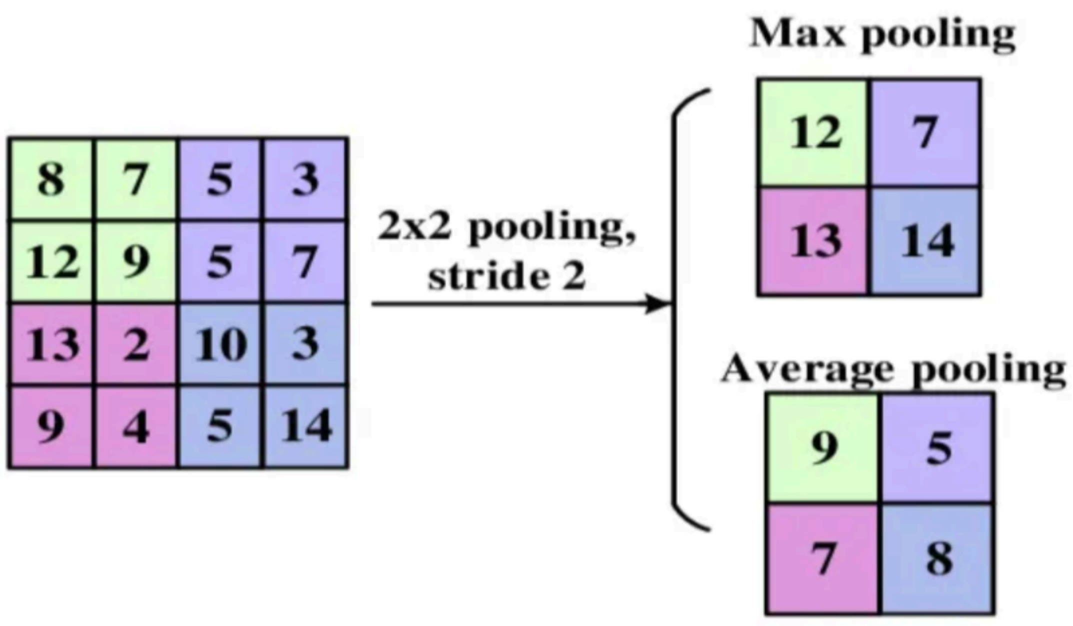


0	0	0	0	0	0
0	1	0	0.5	0.5	0
0	0	0.5	1	0	0
0	0	1	0.5	1	0
0	1	0.5	0.5	1	0
0	0	0	0	0	0

0.5	0	0.25	0.25
0	1.25	0.5	0.5
0	0.5	0.75	1.5
0.5	0.25	1.25	1

POOLING IN CNN

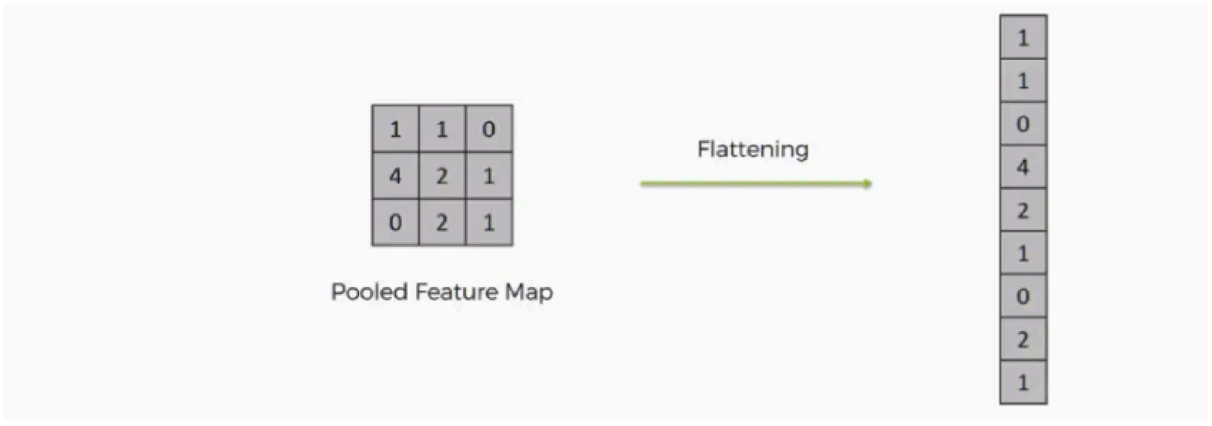
Pooling in convolutional neural networks is a technique for generalizing features extracted by convolutional filters and helping the network recognize features independent of their location in the image.



8	7	5	3
12	9	5	7
13	2	10	3
9	4	5	14

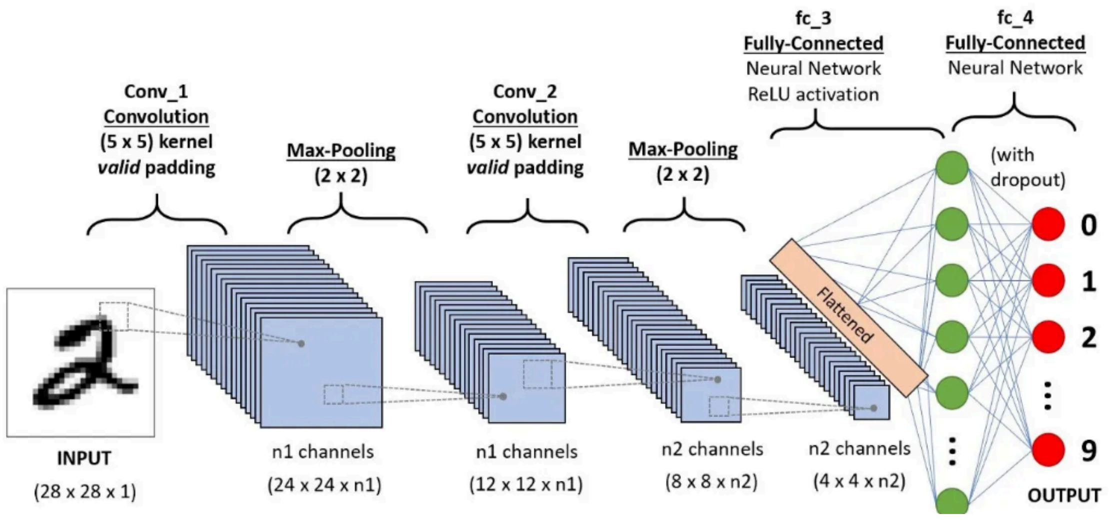
FLATTEN IN CNN

Flattening is used to convert all the resultant 2-Dimensional arrays from pooled feature maps into a single long continuous linear vector. The flattened matrix is fed as input to the fully connected layer to classify the image.



ARCHITECTURE OF CNN:

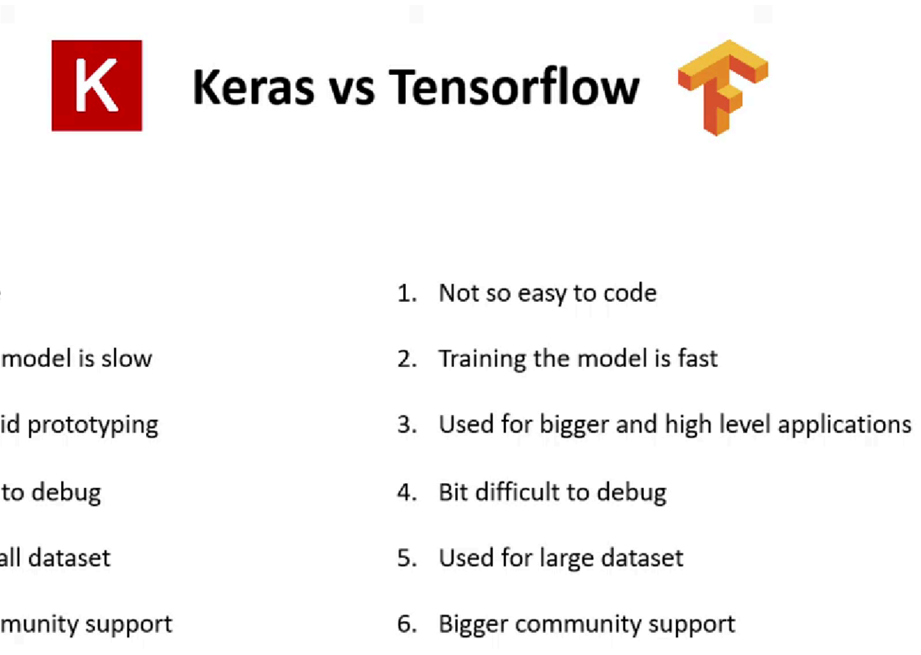
A convolutional neural network (CNN), is a network architecture for deep learning which learns directly from data. CNNs are particularly useful for finding patterns in images to recognize objects. They can also be quite effective for classifying non-image data such as audio, time series, and signal data.



BACKPROPAGATION IN CNN

Refer to this pdf:
Backpropagation

KERAS V/S TENSORFLOW


Keras vs Tensorflow

1. Easy to code	1. Not so easy to code
2. Training the model is slow	2. Training the model is fast
3. Used for rapid prototyping	3. Used for bigger and high level applications
4. Lesser need to debug	4. Bit difficult to debug
5. Used for small dataset	5. Used for large dataset
6. Smaller community support	6. Bigger community support

Refer to this link for implementation:

CNN :CNN Implementation