SREENIVASA INSTITUTE OF TECHNOLOGY AND MANAGEMENT STUDIES
(AUTONOMOUS)
MCA DEPARTMENT

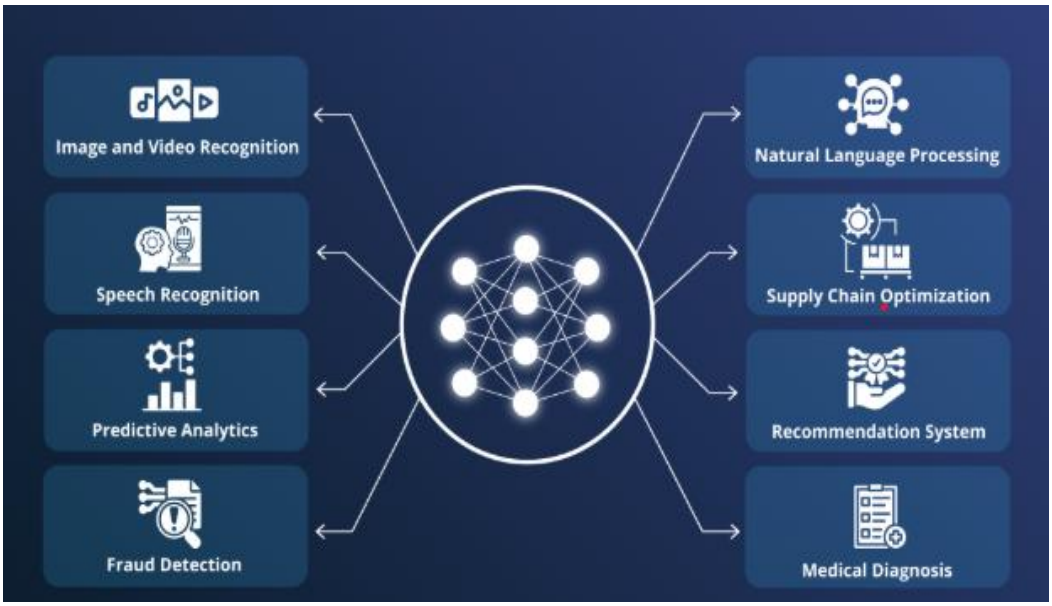
LECTURE NOTES

Subject Name: DEEP LEARNING

Year / Branch: II MCA – II SEMESTER

Regulation: R24

Prepared By: Mr. N P Gangadhar, Assistant Professor



Deep learning is the key technology behind many recent breakthroughs in artificial intelligence

—Yann LeCun

II MCA – II SEMESTER			
COURSE CODE:	**24MCA221**	**CREDITS:**	**4**
COURSE TITLE:	**DEEP LEARNING**	**L-T-P:**	**4-0-0**

PREREQUISITES: *A Course on "Machine learning", Artificial Neural Networks and Knowledge on basic linear algebra may be helpful.*

COURSE EDUCATIONAL OBJECTIVES:

CEO 1: Toacquireknowledgeaboutdifferentmathematicaltoolsfor DeepLearning.
CEO 2: Tounderstandfundamentalsofartificialneural networks.
CEO 3: ToExplore various optimizers and Regularization Techniques
CEO 4: ToLearn about ConvolutionalNeural Networks.
CEO 5:Tocomprehend Object Detection, Autoencoder and GAN Architectures

UNIT-I: MATHEMATICAL TOOLS FOR DEEP LEARNING — **Lecture Hrs:12**

Linear Algebra :Matrix, Vector,Transpose,Tensor, operations on elements, Systems of Linear equations,Rank, Norm,expressing a Matrix, Determinant, Trace, Eigen values and Eigen Vectors, Singular Value Decomposition(SVD). **Statistics** – Probability, Random Variable, Binomial Distribution, Poisson Distribution, Normal Distribution, Sampling, Central limit Theorem. **Calculus**- Derivatives, rules forderivatives,Partial derivatives.

UNIT-II: FUNDAMENTALSOFARTIFICIALNEURALNETWORK(ANN) — **Lecture Hrs:12**

Understanding the Biological Neuron, Exploring the Artificial Neuron, Early Implementation of ANN – RmcCulloch-Pits mosel of Neuron, Rosenblatt's Perceptron,Types of Activation Functions-linear function, non-linear function, softmax function. Architectures of neural network- Single layer feed forward network, Multi-layer feed forward ANN, Recurrent Neural Network, Convolutional Networks. Learning Process in ANN – Weight of Interconnection between neurons, Gradient Descent and Backpropagation.

UNIT-III: TRAININGDEEPNEURALNETWORK — **Lecture Hrs:12**

Initializing Weights- He/ Kaiming initialization, Xavier initialization. Batch, Mini-batch and stochastic gradient descent. Regularization-L1/ L2 regularization, Early stopping, Dropout regularization, data augmentation. Normalization of inputs-Batch Normalization, Batch Normas regularizer.

UNIT-IV: CONVOLUTIONAL NETWORKS — **Lecture Hrs:10**

Building blocks of CNN, Building a Convolution Neural Network, Popular CNN Architectures-LeNet-5, AlexNet, ZFNET, VGG-16, GoogleNet and ResNet Object Detection- one stage deection techniques – YOLO, SSD,Two stage object detection techniques – R-CNN, fast R-CNN, faster R-CNN, Mask R-CNN, Applications of Object Detection

UNIT-V: SEQUENCE-BASEDMODELS ANDOTHERDLARCHITECTURES — **Lecture Hrs:12**

Recurrent Neural Network – Data Preparation for RNN Vanishing Gradient problem and RNN, Applications of RNN, Types of RNN, Limitations of RNN, Longshort-termmemory (LSTM), Gated RNNs, Bidirectional RNNs. Other DL Architectures- Autoencoder, Architecture and its applications, GAN, GAN Architecture and its applications,

TEXTBOOKS:

1. *AmitKumarDas,SaptarsiGoswami,PabitraMitra,AmlanChakrabarti,"DeepLearning", Pearson Paperback,First Edition,2021.*

2. *IanGoodfellow,YoshuaBengio,AaronCourville, "DeepLearning",MITPress,2016.*

REFERENCE BOOKS:

1. *JoshPattersonandAdamGibson,"Deeplearning:Apractitioner'sapproach",O'ReillyMedia,First Edition,2017.*
2. *NikhilBuduma, "Fundamentals of Deep Learning, Designing next-generationmachineintelligence algorithms",O'Reilly,ShroffPublishers,2019.*

COURSE OUTCOMES: *On successful completion of this course, students will be able to:*		POs related to COs
CO1	Understand the mathematical background required for Deep learning.	**PO1,PO2**
CO2	Analyze the fundamental concepts of Artificial neural networks.	**PO1,PO2,PO3**
CO3	Understand and apply the deep neural networks	**PO1,PO2,PO3,PO4**
CO4	Explore the Purpose of Convolution Neural Network, Popular Architectures of CNN	**PO1,PO2,PO3,PO4,PO8**
CO5	To learn about sequence-based models like RNN, LSTM and Gated RNN and other DL architectures and use for real-world problems	**PO1,PO2,PO3,PO4,PO8**

CO-PO MAPPING(DETAILED; HIGH:3; MEDIUM:2; LOW:1)

Course	POs / COs	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8
C401: Deep Learning	C401.1	3	3	-	-	-	-	-	-
	C401.2	3	3	2	-	-	-	-	-
	C401.3	3	3	2	3	-	-	-	-
	C401.4	3	3	3	2	-	-	-	3
	C401.5	3	3	3	3	-	-	-	3
	C401	3	3	2.5	2.67				3

UNIT IV- CONVOLUTIONAL NEURAL NETWORKS

Unit Overview

This unit explains the building blocks of CNNs, how a convolutional neural network is constructed, and the major CNN architectures used in computer vision. It then introduces object detection, including one-stage and two-stage approaches, and concludes with the main applications of object detection in real-world systems. Standard course notes and framework documentation present these topics as the core foundation of modern visual AI systems.

Learning Outcomes

After completing this unit, students will be able to:

1. Explain the building blocks of CNNs.
2. Describe how to build a convolutional neural network.
3. Discuss major CNN architectures such as LeNet-5, AlexNet, ZFNet, VGG-16, GoogLeNet, and ResNet.
4. Explain one-stage object detection techniques such as YOLO and SSD.
5. Explain two-stage object detection techniques such as R-CNN, Fast R-CNN, Faster R-CNN, and Mask R-CNN.
6. Discuss important applications of object detection.

Importance of Studying this Unit

This unit is important because CNNs are the foundation of many modern computer-vision systems. They are used in image classification, detection, segmentation, medical imaging, robotics, autonomous driving, and surveillance. Understanding CNN architectures and object-detection methods is essential for working with real-world AI vision applications.

Key Concepts

Convolution, filter, feature map, stride, padding, activation function, pooling, fully connected layer, CNN architecture, object detection, one-stage detector, two-stage detector, region proposal, bounding box, classification score, mask prediction.

Convolutional Neural Network

- **CNN (Convolutional Neural Network)** is a class or type of deep neural network which is usually applied for image classification.
- It takes an input image, applies filters, flatten the image, and classify the image.

- Convolutional networks (LeCun, 1989), also known as convolutional neural networks or CNNs, are a specialized kind of neural network for processing data that has a known, grid-like topology.
- The first CNN was created by **Yann LeCun;** The architecture is particularly useful in image-processing applications. at the time, the architecture focused on handwritten character recognition.
- A digital image is a **binary representation** of visual data. It contains a series of **pixels** arranged in a grid-like fashion that contains pixel values to denote how bright and what color each pixel should be.

Convolutional Operation in CNNs

What is Convolution?

- **Convolution** is a mathematical operation between:
- **Input** (I) → e.g., image (2D array of pixels).
- **Kernel/Filter** (K) → small matrix that detects features (edges, textures, patterns).
- The result is a new 2D array called a **Feature Map** (activation map).

In CNNs, convolution extracts features like edges, corners, and shapes.

Mathematical Definition

For 2D convolution:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n) \cdot K(i-m, j-n)$$

where:

- $I(m, n)$ =input image pixel
- K= kernel (weights)

$S(i, j)$ =output feature map value

Cross-Correlation in CNNs

- In practice, most CNNs use **Cross-Correlation**, not pure convolution.
- Formula:

$$S(i,j) = \sum_m \sum_n I(i+m, j+n) \cdot K(m,n)$$

Difference:

- In **true convolution**, the kernel is flipped before sliding.
- In **cross-correlation**, the kernel is applied directly (no flip).

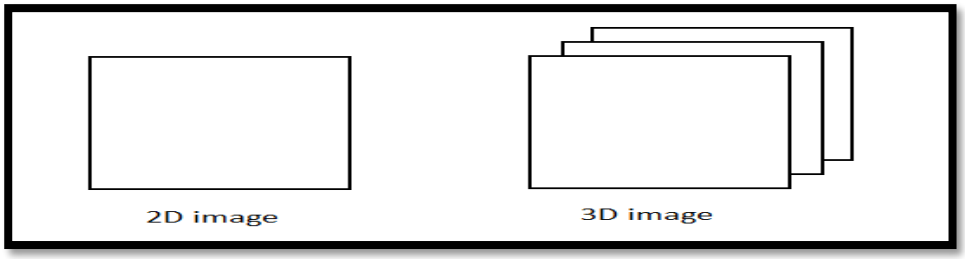
- For learning in CNNs → it doesn't matter, because kernels are learned automatically.

Building Blocks of CNN

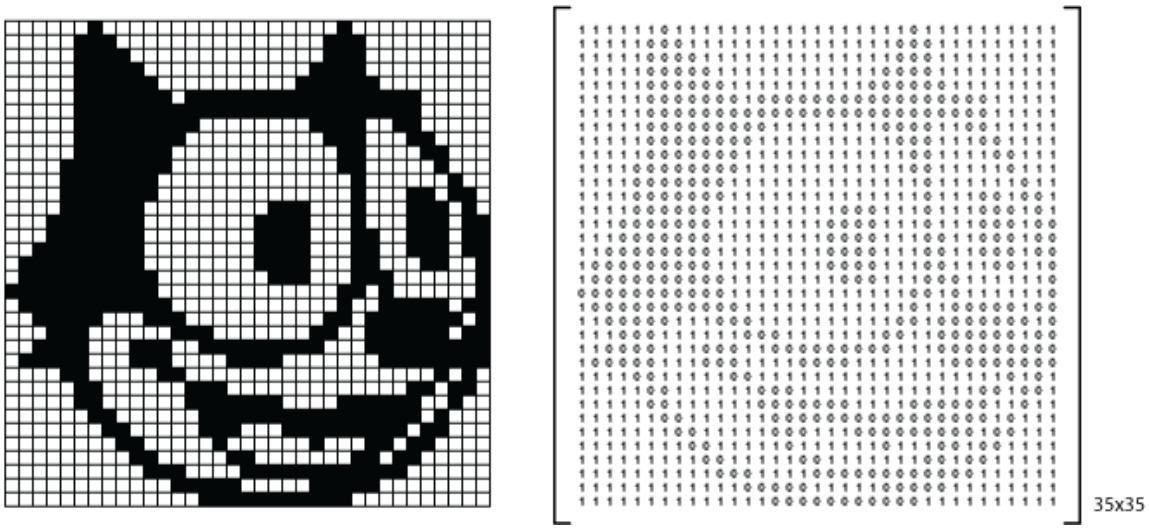
1) **Input Image**
2) **Convolution Layer**
3) **Pooling Layer**
4) **Fully Connected Input Layer(Flatten)**
5) **Fully Connected Layer**

INPUT IMAGE

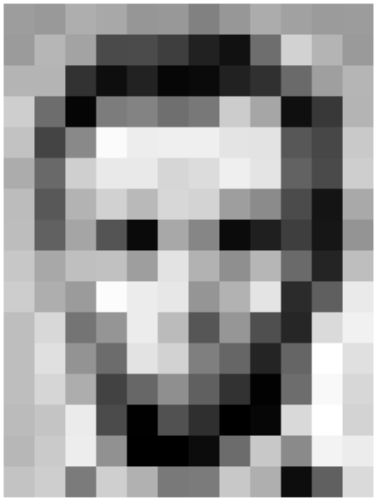
- ❖ The input image will be broken down into pixels.
- ❖ If it is a black and white image, it will only have one layer and pixels will be interpreted as 2D array with the value from 0 to 255.
- ❖ If it is colored image, it will have 3 layers (red, green, blue) and will be interpreted as 3D array.

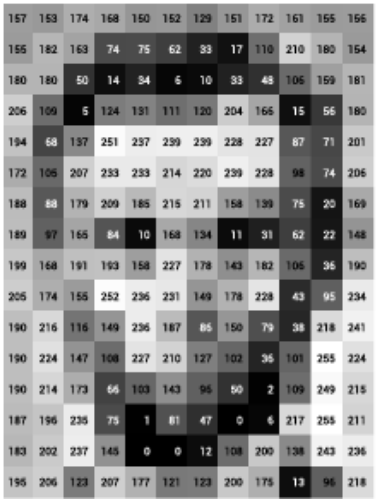


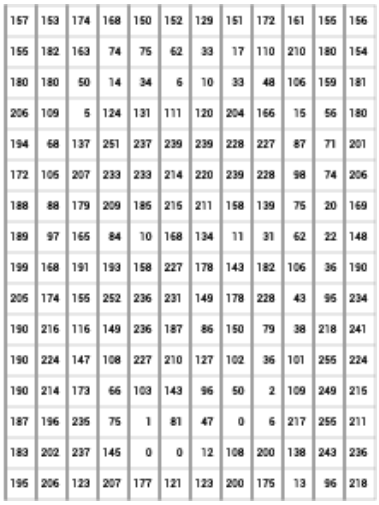
Example of 2D Black and White Image



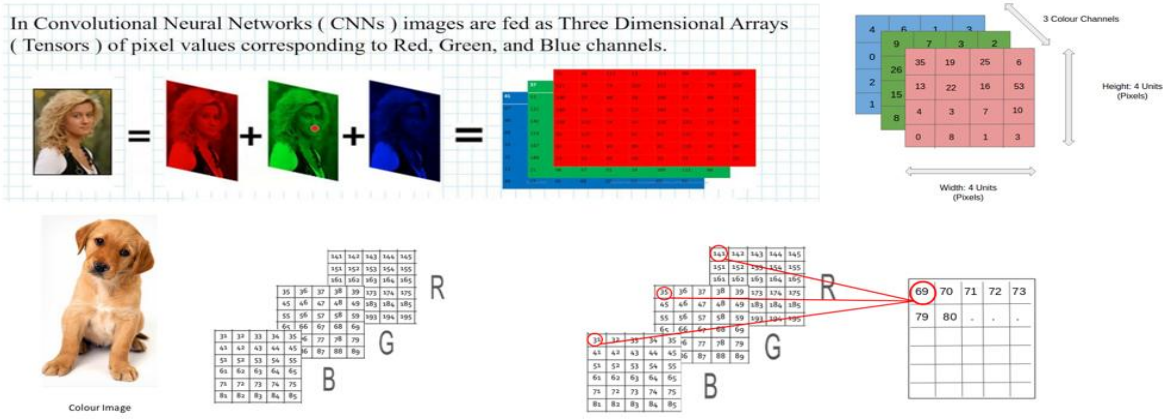
Example of 2D Gray Scale Image





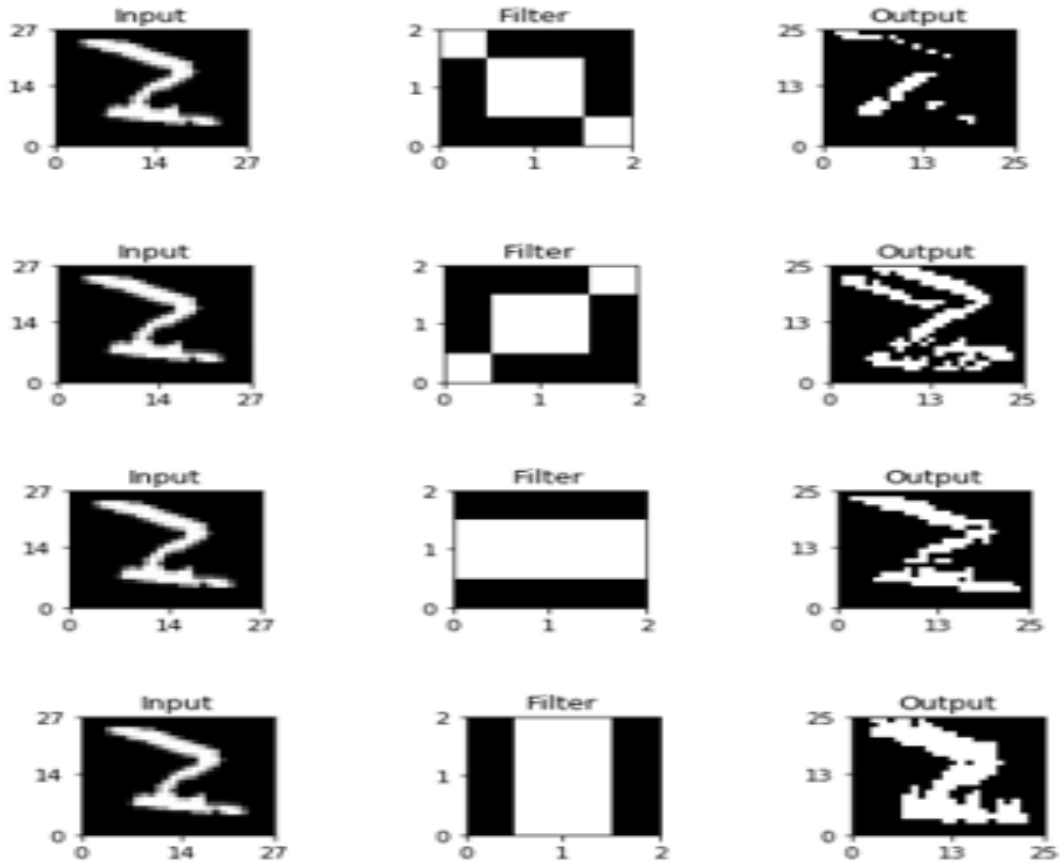


Example of 3D Colour Image



Convolution Layer

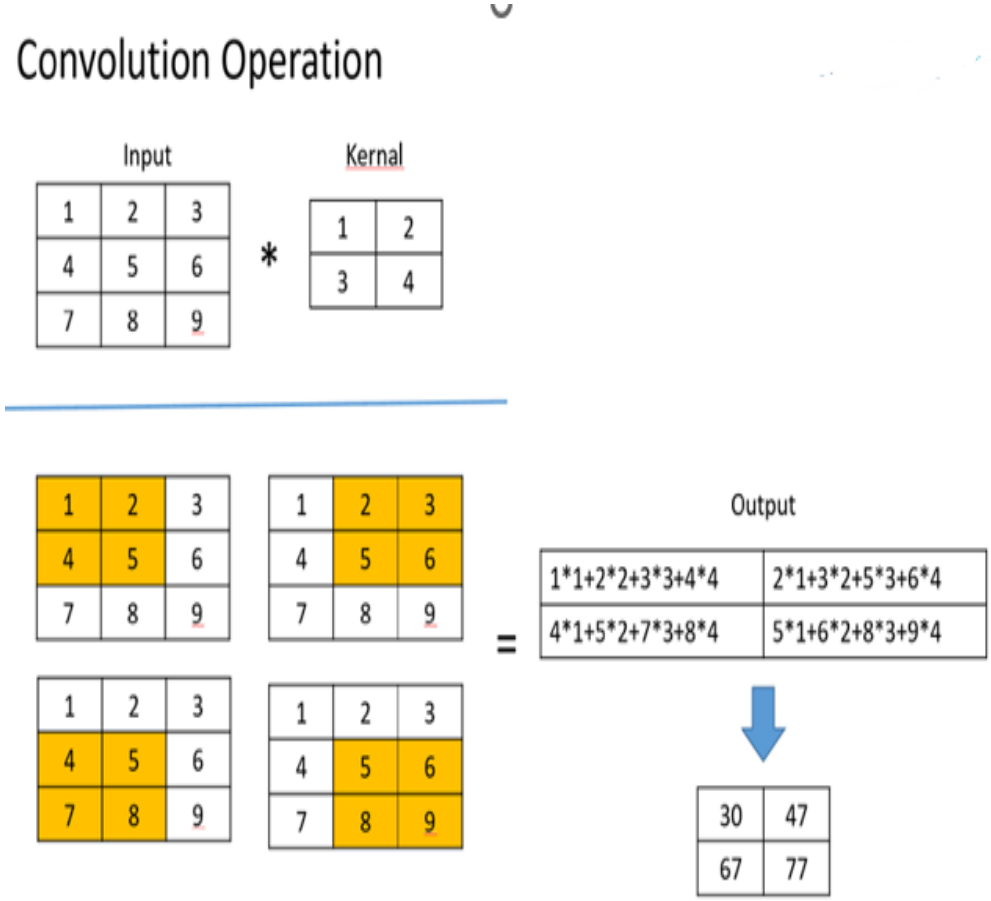
- In the convolution layer, several filters of equal size are applied, and each filter is used to recognize a specific pattern from the image, such as the curving of the digits, the edges, the whole shape of the digits, and more.
- For example, one filter might be good at finding straight lines, another might find curves, and so on. By using several different filters, the CNN can get a good idea of all the different patterns that make up the image.



Using these two matrices, we can perform the convolution operation by applying the dot product, and work as follows:

1) Apply the kernel matrix from the top-left corner to the right.
2) Perform element-wise multiplication.
3) Sum the values of the products.
4) The resulting value corresponds to the first value (top-left corner) in the convoluted matrix.
5) Move the kernel down with respect to the size of the sliding window.
6) Repeat steps 1 to 5 until the image matrix is fully covered.

The dimension of the convoluted matrix depends on the size of the sliding window. The higher the sliding window, the smaller the dimension.



What is a Kernel in CNN?

- A **kernel** (also called **filter** or **convolutional matrix**) is a **small-sized matrix of numbers (weights)** used to detect features in the input data (like images).
- It is much smaller than the input image (for example, 3×3 or 5×5 instead of 28×28).
- The kernel "slides" across the image and performs a **convolution operation** to create a **feature map**.

How the Kernel Works (Step by Step)

1. **Take a small region of the image** (same size as the kernel).
2. **Multiply each pixel value** with the corresponding kernel value.
3. **Add them up** → result is a single number.
4. Place that number in the output (feature map).
5. **Slide the kernel** across the whole image (left to right, top to bottom).

   This process = **convolution**.

Input Image

252	251	246	207	90
250	242	236	144	41
252	244	228	102	43
250	243	214	59	52
248	243	201	44	54

Mark C. F. Sousa

Feature map

Kernel

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

Receptive field

X

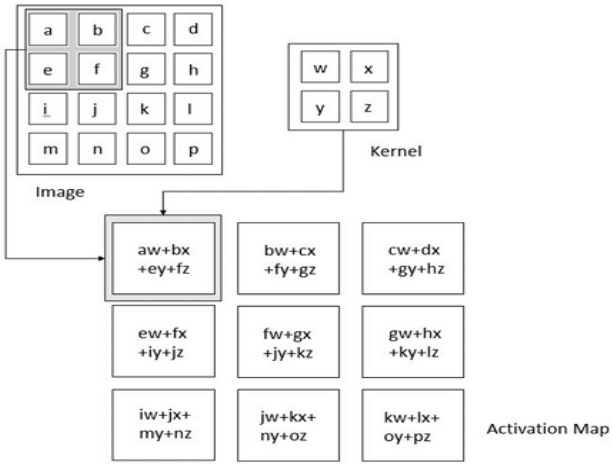
=

If we have an input of size W x W x D and Dout number of kernels with a spatial size of F with stride S and amount of padding P, then the size of output volume can be determined by the following formula:

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

Fig: Formula for Convolution Layer

This will yield an output volume of size ***Wout* x *Wout* x *Dout*.**



Note: padding is the process of adding extra pixels (usually zeros) around the border of an input image or feature map before applying convolution.

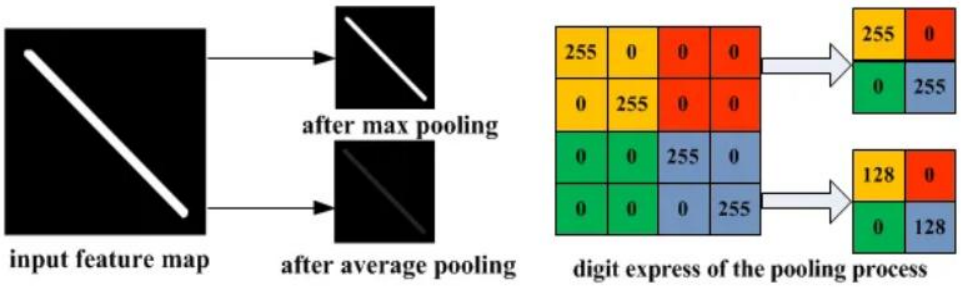
Pooling Layer

The goal of the pooling layer is to pull the most significant features from the convoluted matrix.

This is done by applying some aggregation operations, which reduce the dimension of the feature map (convoluted matrix), hence reducing the memory used while training the network.

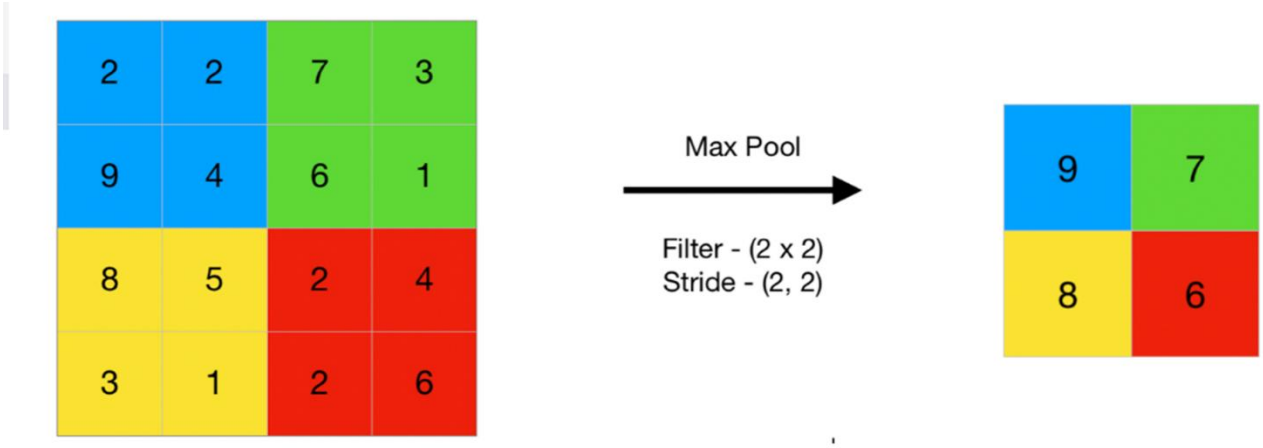
The most common aggregation functions that can be applied are:

1) Max pooling, which is the maximum value of the feature map
2) Average pooling is the average of all the values.



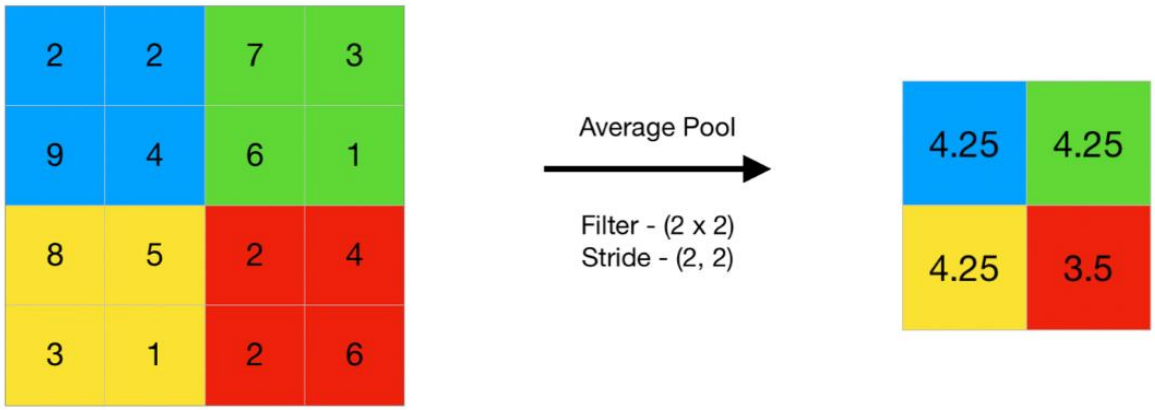
Max Pooling

- Max pooling selects the maximum element from the region of the feature map covered by the filter. Thus, the output after the max-pooling layer would be a feature map containing the most prominent features of the previous feature map.
- Max pooling layer preserves the most important features (edges, textures, etc.) and provides better performance in most cases.



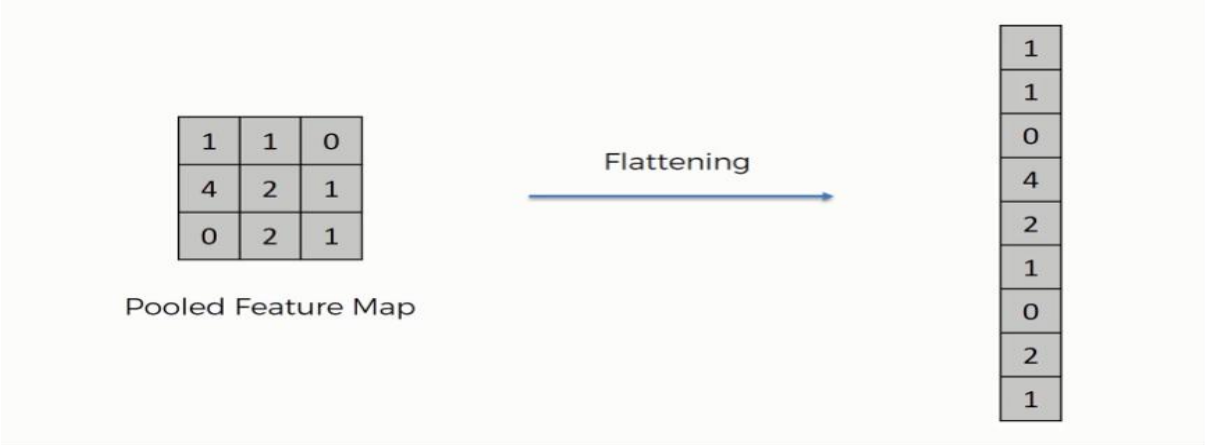
Average Pooling

- Average pooling computes the average of the elements present in the region of feature map covered by the filter.
- Thus, while max pooling gives the most prominent feature in a particular patch of the feature map, average pooling gives the average of features present in a patch.
- Average pooling provides a more generalized representation of the input. It is useful in the cases where preserving the overall context is important.



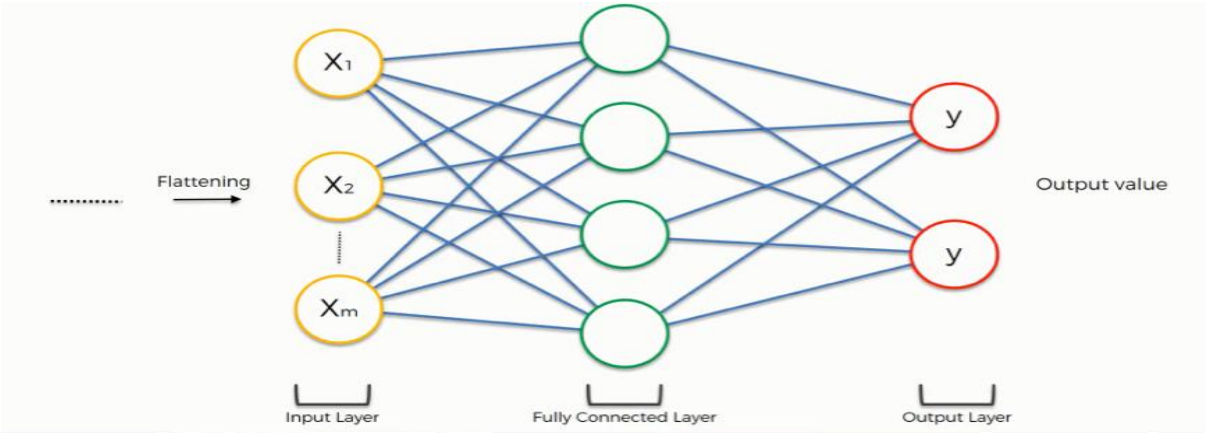
Fully Connected Input Layer(Flatten)

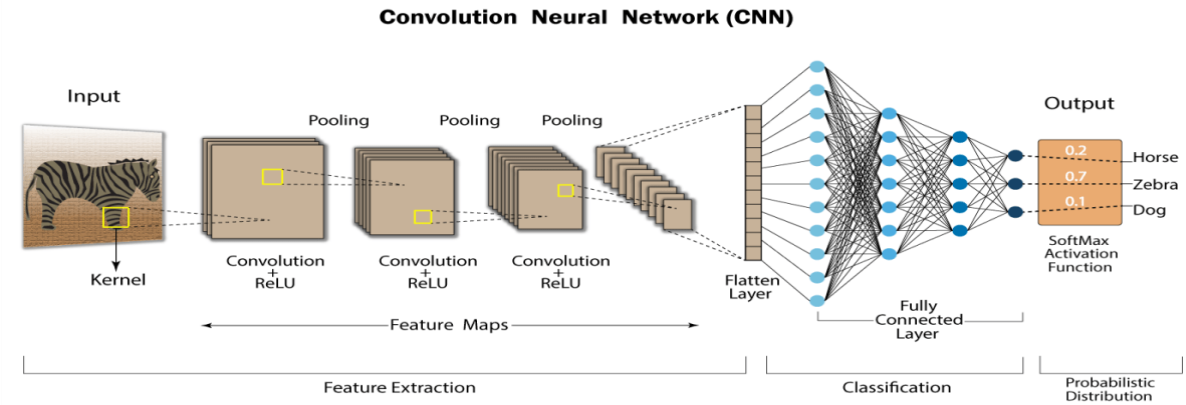
The last pooling layer flattens its feature map so that it can be processed by the fully connected layer.

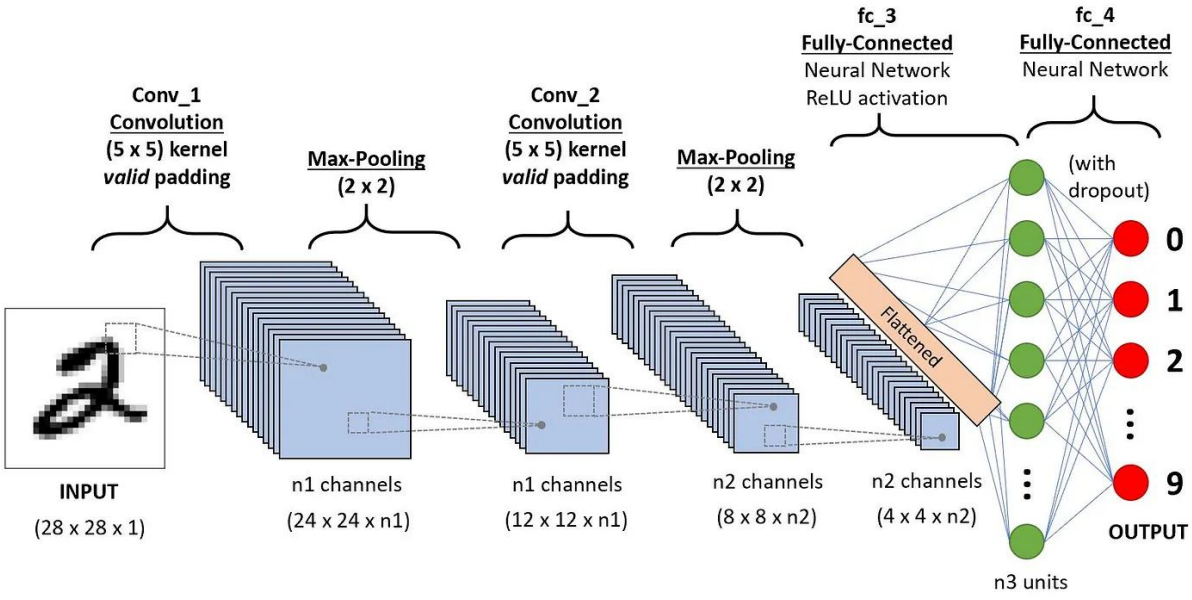


Fully Connected Layer

- The flattened matrix goes through a fully connected layer to classify the images.
- The purpose of this layer is to classify the image into a label. It takes the output of previous layer and predicts the best label by applying weights and “voting”.
- The final output will be the probabilities for each label.







Popular CNN Architectures

LeNet (1998)

- Developed by: Yann LeCun, Corinna Cortes, and Christopher Burges.
- Designed for handwritten digit recognition (MNIST dataset).
- First successful CNN architecture.
- Considered the “Hello World” of Deep Learning.
- Foundation for modern CNNs in computer vision.
- Architecture:
- Consists of 5 convolutional layers + 2 fully connected layers.
- Uses convolution + pooling (subsampling) → fully connected layer → output.
- Max-pooling helps reduce spatial size, prevent overfitting, and improve training.
- Limitations:
- Faced vanishing gradient problem.
- Could not handle very complex images.
- Applications:
- Handwritten digit recognition.
- Traffic sign recognition.
- Early face detection tasks.

AlexNet (2012)

- Developed by: Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton.
- First large-scale CNN for image recognition.
- Won ILSVRC (ImageNet Large Scale Visual Recognition Challenge) 2012.
- Popularized CNNs and marked the beginning of the deep learning revolution.
- **Architecture:**
- Similar to LeNet but deeper and larger.
- 5 convolutional layers + max-pooling layers.
- 3 fully connected layers.
- 2 dropout layers to reduce overfitting.

- Activation: ReLU (hidden layers), Softmax (output layer).
- ~60 million parameters.
- **Key Innovations:**
- First CNN trained on large-scale dataset (ImageNet with 1.2M images).
- Used ReLU activation (faster training than sigmoid/tanh).
- Introduced Dropout (to avoid overfitting).
- Leveraged GPU computing for faster training.
- **Applications:**
- Large-scale image classification.
- Object detection and recognition

ZF Net (2013)

- **Developed by**: Matthew Zeiler and Rob Fergus.
- **Purpose & Importance**:
- **Winner of ILSVRC 2013**.
- Improvement over **AlexNet** with better hyperparameter tuning.
- Achieved higher accuracy on ImageNet classification.
- **Architecture**:
- Total of **7 layers**.
- **Convolutional + max-pooling layers** for feature extraction.
- Smaller **filter size** and **stride** in the first layer (more fine-grained features).
- Expanded size of middle convolutional layers.
- **Dropout** used before fully connected output (to reduce overfitting).
- Included **deconvolutional layers** for feature visualization and approximate inference.
- **Key Innovations**:
- Hyperparameter optimization (stride, filter size, depth).
- Visualization of intermediate feature maps via **deconvolution** → helped understand how CNNs "see."
- More efficient than AlexNet with fewer parameters.

- **Applications**:
- Image classification (ImageNet).
- Object recognition and feature visualization.

GoogLeNet (2014)

- **Developed by**: Christian Szegedy, Jeff Dean, Alexander Toshev, et al. (Google Research).
- **Winner of ILSVRC 2014** (ImageNet classification).
- Achieved **much lower error rate** than AlexNet (2012) and ZF Net (2013).
- Outperformed VGGNet (2014 runner-up) in terms of accuracy and efficiency.
- **Architecture**:
- Very deep network (**22 layers**).
- Introduced the **Inception module**, which combines multiple filter sizes (1×1, 3×3, 5×5) in parallel.
- **1×1 convolutions** used for dimensionality reduction (reduces number of parameters).
- Used **global average pooling** instead of large fully connected layers (reduces overfitting and parameters).
- Shortcut-style connections allow better flow of information between layers.
- **Key Innovations**:
- Inception module → computational efficiency with deep networks.
- Much fewer parameters compared to VGG, yet deeper and more accurate.
- Balanced **depth** (22 layers) and **efficiency**.
- **Applications**:
- **Street View House Numbers (SVHN)** digit recognition.
- Object detection and image classification in large-scale datasets.
- Roadside object recognition in real-world vision tasks.

VGGNet (2014)

- **Developed by**: Karen Simonyan and Andrew Zisserman (Oxford University, VGG group).
- **Purpose & Importance**:

- Runner-up in **ILSVRC 2014** (behind GoogLeNet).
- Known for its **simplicity and uniform design**.
- Became a strong **baseline model** for many computer vision tasks.
- **Architecture**:
- Most common variant: **VGG16** (16 weight layers), also **VGG19** (19 weight layers).
- Uses **3×3 convolutional filters** stacked multiple times.
- Input size: **224 × 224 pixels**.
- Produces **4096 convolutional features** before fully connected layers.
- Very deep (up to 19 layers) but simple → only conv + pooling layers, then FC.
- Contains **~95 million parameters**, making it computationally heavy.
- **Key Innovations**:
- Demonstrated that **deeper networks → better accuracy**.
- Used **small filters (3×3)** repeatedly instead of larger filters → more non-linearity with fewer parameters.
- Though heavy to train, its learned **deep feature representations** are transferable to other tasks.
- **Applications**:
- Image classification and object detection.
- Used in popular frameworks such as **YOLO, SSD, Faster R-CNN** for feature extraction.
- Strong baseline in many research papers and vision benchmarks.

ResNet (2015)

- **Developed by**: Kaiming He et al. (Microsoft Research).
- **Purpose & Importance**:
- **Winner of ILSVRC 2015** with **top-5 error of 3.57%** (not 15.43%).
- Proved that **very deep networks (152+ layers)** can be trained effectively.
- Solved the **vanishing gradient problem** using *Residual (skip) connections*.
- **Architecture**:
- Common versions: **ResNet-34, ResNet-50, ResNet-101, ResNet-152**.

- Introduced **residual blocks** → identity shortcut connections add input directly to output of a layer.
- Allows training of networks with **hundreds of layers**.
- **Key Innovations**:
- Residual learning enables **deeper models without performance degradation**.
- Faster convergence and higher accuracy.
- Scalable (can build deeper or lighter versions).
- **Applications**:
- Image classification, object detection, segmentation.
- NLP tasks (sentence completion, machine comprehension).
- Microsoft's machine comprehension system (2016–17).

Object Detection In CNN

Object Detection is a computer vision technique used to:

- Identify objects (classification)
- Locate objects (bounding boxes) in an image or video

For example, detecting a *car, person, and dog* in an image with rectangles around them.

Modern object detection uses Convolutional Neural Networks (CNNs) and is mainly divided into:

1. One-Stage Detection
2. Two-Stage Detection

One-Stage Object Detection

One-stage detectors predict bounding boxes and class labels directly in a single step without a separate region proposal stage.

YOLO (You Only Look Once)

❖ YOLO is a one-stage detector that processes the entire image in a single pass through the network.

❖ It divides the image into a grid and each grid cell predicts bounding boxes and class probabilities simultaneously.

❖ Because it sees the whole image at once, it's very fast — great for tasks like live video detection.

❖ Think of it as: Look at the image once, and detect all objects at once.

Advantages:

- Very fast — high frames per second (FPS).
- Good for real-time use (e.g., drones, traffic cams).
- Works best on medium and large objects (early versions had trouble with tiny objects).

SSD — Single Shot Multi Box Detector

❖ SSD is another one-stage detector that also predicts object classes and boxes in one pass, but with a slightly different strategy:

❖ Instead of a single grid, SSD uses multiple feature maps at different scales in the network, allowing it to detect objects of different sizes.

❖ At each location and scale, SSD applies default boxes (anchor boxes) with various aspect ratios and sizes.

❖ These multiple scales help it better detect smaller objects compared to early YOLO models.

Key points:

- Fast, like YOLO, but uses **multi-scale detection** to catch small and large objects.
- Often performs a good **balance of speed and accuracy**.
- Suitable for mobile/embedded environments when using lightweight backbones.

Two-Stage Object Detector:

Two-stage detectors separate object detection into two steps:

1. Stage 1 — Region Proposal: Find where objects might be in the image (generate candidate boxes).
2. Stage 2 — Classification & Refinement: For each proposed box, decide what object it contains and adjust the box to fit precisely.

Types of Two-Stage Object Detectors:

R-CNN (Region-based CNN):

❖ Introduced first in this family.

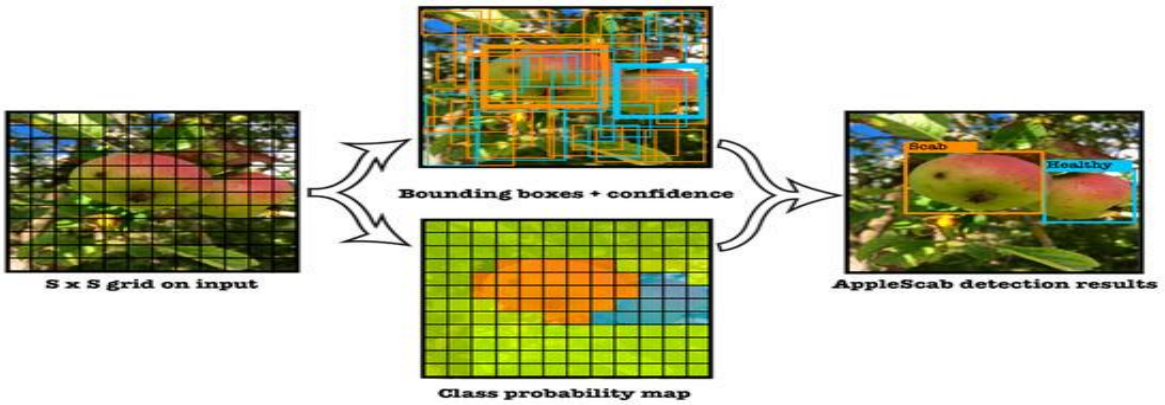
❖ Uses a separate algorithm (like *Selective Search*) to propose ~2000 possible object regions from an image.

- ❖ Each region is then cropped, resized, and passed through a CNN to extract features.
- ❖ After feature extraction, it uses a classifier (originally SVM) to decide what object is present and adjusts the bounding box.

Example

In the fruit bowl image:

- Box might be proposed around an apple.
- CNN extracts features of that box (shape, color patterns).
- SVM determines this box contains an *apple* and not *background*.
- A bounding box regressor slightly expands/shifts the box to fit the apple better



Fast R-CNN

- ❖ Improvement over R-CNN that shares computations.
- ❖ Instead of processing each region separately, the whole image is passed once through a CNN to create a feature map.
- ❖ Then *RoI Pooling* extracts fixed-size features for all region proposals from that shared feature map.
- ❖ It replaces SVM with a softmax classifier and trains the network end-to-end

Example

Fruit photo:

- Only one CNN pass processes the full basket.
- ROI Pooling extracts features for each fruit proposal from the shared map.
- The network outputs *apple* , *banana*, etc., and refines the boxes.

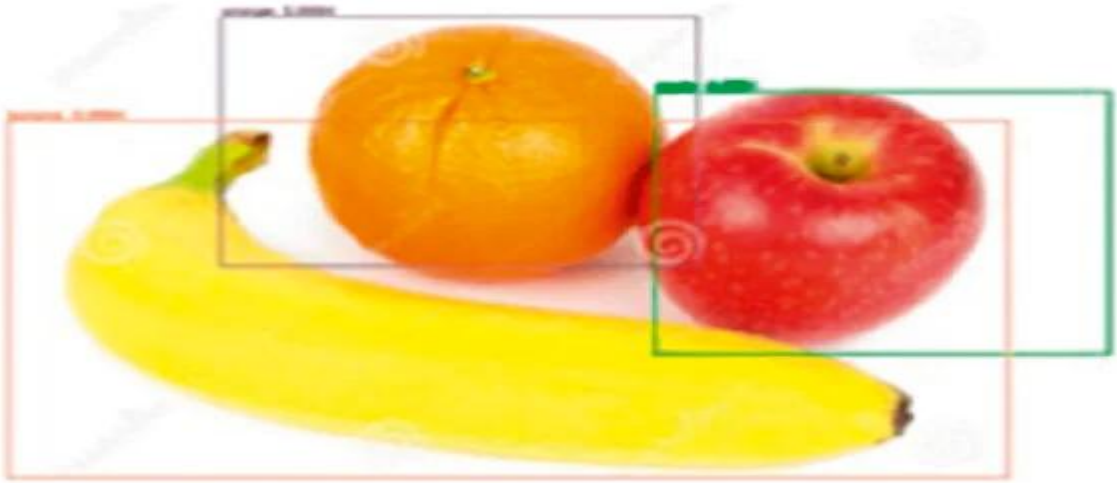
Faster R-CNN

- ❖ Major breakthrough in two-stage detection.
- ❖ Integrates a Region Proposal Network (RPN) inside the CNN to generate object proposals automatically — no more separate slow proposal algorithm.
- ❖ RPN shares features with the detection network, speeding up the whole pipeline.
- ❖ Stage 1: RPN proposes candidate regions.
- ❖ Stage 2: A Fast R-CNN detector classifies and refines those regions.

Example:

Fruit bowl:

- RPN proposes likely fruit regions (e.g., one for apple, one for banana).
- On the shared feature map, these proposals are classified and refined.
- Output might be: “Apple at (x1,y1,x2,y2)”, “Banana at (x3…y3)”.



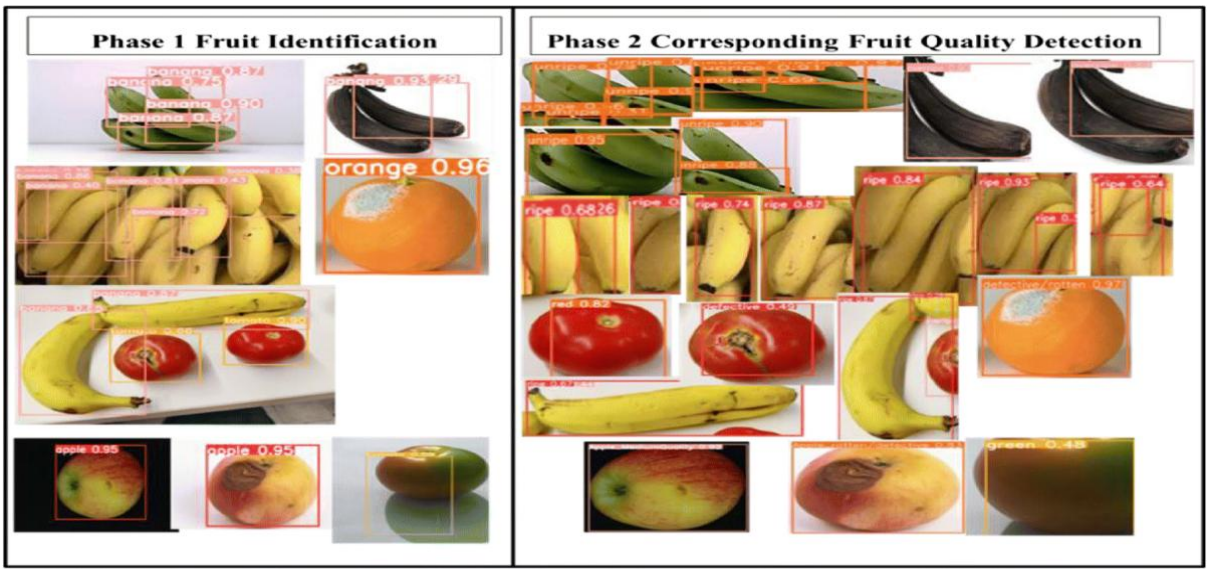
Mask R-CNN

- ❖ Extends Faster R-CNN to also produce pixel-level masks (instance segmentation).
- ❖ In addition to class and bounding box prediction, it has a third branch that outputs a binary mask (pixel-by-pixel shape) for each detected object.
- ❖ Uses an improved *RoIAlign* layer to align features precisely.
- ❖ So instead of just "where and what," Mask R-CNN gives you shape as well as location and label for each instance.

Example:

Fruit bowl:

- Faster R-CNN might detect "apple at box X", "banana at box Y".
- Mask R-CNN additionally outputs pixel masks like "mask for apple shape" and "mask for banana shape"



Applications of Object Detection in CNN

1.Autonomous Vehicles & Smart Transportation

CNNs are the "eyes" of self-driving cars. They process real-time video feeds to ensure safe navigation.

- Obstacle Detection: Identifying pedestrians, cyclists, and other vehicles to avoid collisions.
- Traffic Sign Recognition: Detecting and interpreting speed limits, stop signs, and traffic lights.
- Lane Detection: Helping vehicles stay within road boundaries.
- Smart Infrastructure: Cities use CNNs in traffic cameras to monitor congestion and automatically detect accidents or traffic violations.

2. Healthcare & Medical Diagnostics

Precision is critical in medicine, and CNN-based models (like Faster R-CNN) are now standard tools for radiologists.

- Tumor & Lesion Detection: Automatically spotting anomalies in X-rays, MRI scans, and CT scans that might be missed by the human eye.
- Cell Analysis: Detecting and counting specific cell types (e.g., cancer cells or white blood cells) in microscopic imagery.
- Surgical Assistance: Real-time detection of organs and surgical tools during robotic-assisted surgeries.

3. Security & Surveillance

Legacy CCTV systems required constant human monitoring; CNNs have made surveillance proactive.

- Intrusion Detection: Identifying unauthorized persons or vehicles in restricted zones.
- Crowd Management: Counting people in stadiums or public squares to prevent overcrowding and ensure safety.
- Facial Recognition: Detecting faces in a crowd to match against databases for law enforcement or access control.

4. Retail & Inventory Management

The "Just Walk Out" shopping experience is powered entirely by object detection.

- Automated Checkout: Systems detect which items a customer picks up and adds them to a virtual cart.
- Shelf Monitoring: Robots or fixed cameras detect "out-of-stock" items or misplaced products on retail shelves.
- Shrinkage Prevention: Detecting suspicious behavior or un-scanned items at self-checkout kiosks.

5. Industrial Automation & Agriculture

CNNs are driving the "Industry 4.0" and "AgTech" revolutions.

- Quality Control: Detecting surface defects, cracks, or missing components on high-speed manufacturing assembly lines.
- Precision Farming: Drones equipped with CNNs fly over fields to detect pests, diseases, or nutrient deficiencies in specific plants.

- Robotic Sorting: Sorting waste for recycling or picking ripe fruit in automated greenhouses.

6.Sports Analytics & "Player Tracking"

Professional sports leagues (FIFA, NBA, NFL) use CNNs to turn video into actionable data.

- Performance Metrics: Models track the x, y, z coordinates of every player and the ball/puck simultaneously to calculate running speed, fatigue levels, and heat maps.
- Automated Officiating: "Semi-automated offside technology" uses CNNs to detect the exact moment a ball is kicked and the limb positions of players to make millisecond-accurate calls.
- Broadcasting Enhancements: Real-time overlays (like the "yellow line" in football or shot probabilities in basketball) are generated by detecting objects and mapping them to a 3D field model.

7. Wildlife Conservation & Anti-Poaching

CNNs are being deployed in remote areas to protect biodiversity without human interference.

- Satellite Population Counts: High-resolution satellite imagery uses CNNs to count large animals like elephants or whales across thousands of square miles to monitor migration and health.
- Smart Camera Traps: Instead of recording 24/7, AI-powered cameras only "wake up" and alert rangers when they detect specific species or unauthorized human intruders (poachers).
- Species Identification: Research models (like YOLO-WildASM) are now accurate enough to distinguish between individual animals based on unique coat patterns or scars.

8. Augmented Reality (AR) & Virtual Try-Ons

CNNs provide the "spatial awareness" needed for immersive digital experiences.

- Occlusion Handling: For AR to look real, digital objects must go *behind* physical ones. CNNs detect the boundaries of furniture or people so a virtual character can realistically hide behind a real sofa.
- Virtual Fashion: Retail apps use CNNs to detect specific body parts (wrists for watches, feet for shoes, or faces for glasses) to accurately "anchor" a digital 3D model onto the user's body in real-time.

9. Disaster Response & Search and Rescue (SAR)

In emergencies, speed saves lives. CNNs process drone and satellite feeds faster than any human team.

- Damage Assessment: After an earthquake or hurricane, CNNs automatically detect collapsed buildings, flooded roads, and broken bridges to help rescue teams prioritize routes.
- Human Detection in Wildfires: Thermal cameras mounted on drones use CNNs to find the heat signatures of survivors through thick smoke where visual light is useless.

10. Waste Management & Circular Economy

To solve the global recycling crisis, CNNs are automating the "dirty work" of sorting.

- Automated Sorting Belts: High-speed CNNs identify plastic types (PET vs. HDPE), metals, and paper on conveyor belts, triggering air jets to "shoot" the correct item into its designated bin.
- Ocean Cleanup: Autonomous underwater vehicles (AUVs) use object detection to identify and collect plastic debris from the ocean floor while avoiding marine life.

Unit Highlights

- CNNs are built from layers such as **convolution, activation, pooling, and fully connected layers**.
- Popular CNN architectures include **LeNet-5, AlexNet, ZFNet, VGG-16, GoogLeNet, and ResNet**.
- Object detection predicts both **object class** and **object location**.
- **YOLO** and **SSD** are one-stage detection methods.
- **R-CNN, Fast R-CNN, Faster R-CNN, and Mask R-CNN** are two-stage methods.
- **Mask R-CNN** adds mask prediction to Faster R-CNN.
- Object detection is used in vehicles, security, robotics, medicine, agriculture, and retail.

S.No	Questions
UNIT IV- CONVOLUTIONAL NEURAL NETWORKS	
PART-A (Two Marks Questions)	
1.	List the building blocks of a Convolutional Neural Network (CNN).
2.	What is the role of convolution layer in CNN?
3.	What is pooling? Mention its types.
4.	Define fully connected layer in CNN
5.	What are the key features of LeNet-5 architecture?
6.	Mention two important contributions of AlexNet.

7.	What is the purpose of small filters in VGG-16?
8.	What is the main idea behind GoogleNet (Inception Network)?
9.	What is residual learning in ResNet?
10.	Define one-stage object detection. Give examples.
11.	What is YOLO?
12.	What is SSD in object detection?
13.	Define two-stage object detection.
14.	What is Mask R-CNN used for?
15.	Mention two applications of object detection.
PART-B (Ten Marks Questions)	
1.	Explain the convolution operation in detail. Derive the output size formula considering padding and stride, and illustrate with an example.
2.	Describe the complete architecture of a Convolutional Neural Network for image classification, explaining the role of each layer.
3.	Analyze the effect of stride, padding and pooling on feature map dimensions and model performance
4.	Compare classical CNN architectures such as LeNet-5, AlexNet, VGG-16, GoogLeNet and ResNet with respect to depth, innovation and performance improvements.
5.	Explain the concept of residual learning in ResNet. How do residual connections help in training very deep networks?
6.	Differentiate between one-stage and two-stage object detection frameworks with suitable examples.
7.	Explain the working principle of R-CNN, Fast R-CNN and Faster R-CNN. Provide a comparative analysis
8.	Describe the YOLO object detection algorithm. Explain how it performs detection as a regression problem.
9.	Explain the SSD (Single Shot MultiBox Detector) algorithm and discuss its advantages over R-CNN based methods
10.	Discuss real-world applications of CNN and object detection models in areas such as healthcare, autonomous driving and surveillance.

TEXT BOOKS:

1. *Amit Kumar Das, Saptarsi Goswami, Pabitra Mitra, Amlan Chakrabarti, "Deep Learning", Pearson Paperback, First Edition,2021.*

REFERENCE BOOKS:

1. *Josh Patterson and Adam Gibson, "Deep learning: Apractitioner's approach ",O' Reilly Media, First Edition,2017.*
2. *Nikhil Buduma, "Fundamentals of Deep Learning, Designing next generation machine intelligence algorithms",O'Reilly,ShroffPublishers,2019.*