CNN Architecture

1. What is a CNN?

- CNN (Convolutional Neural Network) is a type of neural network designed for image data.
- It automatically learns features from images and classifies them.
- Unlike regular neural networks, CNNs use convolutional layers to extract features.

2. Why Use CNNs?

Traditional neural networks struggle with large image data because:

- They treat every pixel as independent, ignoring spatial relationships.
- They require too many parameters, leading to inefficiency.

CNNs solve this by using:

- **Local connections** – Focus on small regions of the image.
- **Shared weights** – Reduce the number of parameters.
- **Efficient feature extraction** – Makes CNNs ideal for image processing.

3. CNN Workflow

The CNN pipeline consists of four main steps:

1. **Input Data** – Images or video frames.
2. **Preprocessing** – Resize, normalize, or transform images for consistency.
3. **Feature Extraction** – Learn patterns like edges, shapes, and objects.
4. **Classification** – Use extracted features to predict the class (e.g., cat, dog, digit).

4. CNN Architecture Overview

A CNN consists of:

- **Input Layer** – Accepts raw image data.
- **Convolutional Layers** – Extract features like edges, textures, and shapes.
- **Fully Connected Layers** – Classify the extracted features.
- **Output Layer** – Produces predictions (e.g., probabilities for each class).

5. Key Components of CNN

(a) Convolutional Layers

Purpose: Detect patterns (edges, shapes, textures).
How it works:

- A **filter/kernel** slides over the image, performing element-wise multiplication and summing results.
- This produces a **feature map** that highlights specific patterns.

- **Depth**: Each filter detects one feature; multiple filters create multiple feature maps.
- **Pooling**: Reduces the size of feature maps while retaining important information (e.g., max pooling keeps the most prominent feature).

(b) Activation Functions

- Adds non-linearity so the model can learn complex patterns.
- **Common activation**: ReLU (Rectified Linear Unit) – Converts negative values to zero.

(c) Fully Connected Layers

- After feature extraction, feature maps are **flattened** into a vector.
- This vector is passed through fully connected layers for classification.
- These layers work like traditional neural networks.

(d) Output Layer

- Produces the final prediction (e.g., class probabilities).
- Uses **softmax** for multi-class classification or **sigmoid** for binary classification.

6. How CNN Learns Features

Layer-by-Layer Learning:

- **Early Layers** – Detect simple features like edges and lines.
- **Middle Layers** – Combine simple features into shapes (e.g., circles, corners).
- **Deeper Layers** – Detect complex features like objects (e.g., faces, cars).

Example: Face Recognition

- **Layer 1**: Detects edges.
- **Layer 2**: Detects eyes, nose, and mouth.
- **Layer 3**: Combines these into a full face.

7. Why Replace Manual Feature Extraction?

- Before CNNs, humans manually extracted features (e.g., SIFT, HOG).
- CNNs automate this process, learning better and more complex features.

8. Advantages of CNNs

- **Efficient** – Shared weights reduce the number of parameters.
- **Scalable** – Works well with large datasets.
- **Robust** – Handles variations in images (e.g., rotation, scaling).

9. Example: Digit Classification

- **Input**: Image of a digit (e.g., "3").
- **Feature Extraction**:
  - Early layers detect edges and curves.

    - Deeper layers combine these into the shape of "3".
- **Classification**: Fully connected layers decide if the image is "3" or "7".

10. Key Terms to Remember

- **Feature Map** – Output of a filter applied to an image.
- **Filter/Kernel** – Small matrix used to detect patterns.
- **Pooling –** Reduces feature map size (e.g., max pooling).
- **ReLU** – Activation function that introduces non-linearity.
- **Flattening** – Converts feature maps into a 1D vector for classification.