Unit - IV Convolutional Neural Networks

4.1 Introduction to Convolutional Neural Networks (CNNs)

A **Convolutional Neural Network (CNN)** is a class of deep learning models specially designed to process **grid-like data**, such as images. CNNs automatically learn **spatial features** like edges, textures, and shapes using convolution operations.

Key Features:

- Local connectivity
- Parameter sharing
- Translation invariance
- Fewer parameters than fully connected networks

Architecture of CNN

A CNN consists of multiple layers arranged to extract increasingly complex features.

1. Input Layer

- Accepts raw image data (e.g., height × width × channels).
- Example: 224 × 224 × 3 (RGB image)

2. Convolutional Layer

- Applies **filters (kernels)** to extract features.
- Produces **feature maps**.

Purpose: Detect edges, corners, textures, and patterns.

3. Activation Function (ReLU)

- Introduces non-linearity.
- Removes negative values.

Purpose: Improves learning of complex patterns.

4. Pooling Layer

- Reduces spatial dimensions.
- Common types: **Max Pooling**, **Average Pooling**.

Purpose: Reduces computation and overfitting.

5. Fully Connected Layer

- Flattens feature maps into a vector.
- Performs final classification.

6. Output Layer

- Uses **Softmax** (multi-class) or **Sigmoid** (binary).
- Produces final prediction.

Typical CNN Flow

Input → Convolution → ReLU → Pooling → Fully Connected → Output

Applications of CNN

1. Image Classification

- Identifying objects in images
- Example: Face recognition, handwritten digit recognition

2. Object Detection

- Locating and classifying multiple objects
- Example: Autonomous vehicles, surveillance

3. Medical Image Analysis

- Disease detection from X-rays, MRI, CT scans

4. Natural Language Processing

- Sentence classification
- Text feature extraction

5. Video Analysis

- Action recognition
- Motion detection

6. Facial Recognition

- Biometric authentication
- Emotion detection

4.2 Padding, Strided convolution, Convolution over volume, Pooling

Padding

Definition:

Padding is the process of adding extra pixels (usually zeros) around the border of an input image before applying convolution.

Purpose:

- Prevents loss of edge information
- Controls output size
- Allows deeper networks

Types:

- **Valid padding:** No padding
- **Same padding:** Output size equals input size

Strided Convolution

Definition:

Strided convolution moves the convolution filter across the input by **more than one pixel at a time**.

Purpose:

- Reduces spatial dimensions
- Acts as an alternative to pooling

Effect:

- Larger stride → smaller output size
- Faster computation

Convolution over Volume

Definition:

Convolution over volume refers to applying filters to **multi-channel inputs** (e.g., RGB images with depth).

Explanation:

- A filter spans the **entire depth** of the input
- Produces a **2D feature map**
- Multiple filters create multiple feature maps (output volume)

Example:

- Input: 32 × 32 × 3
- Filter: 5 × 5 × 3
- Output: Feature map

Pooling

Definition:

Pooling is a down-sampling operation that reduces the spatial size of feature maps while retaining important features.

Types:

- **Max Pooling:** Takes maximum value
- **Average Pooling:** Takes average value

Purpose:

- Reduces computation
- Controls overfitting
- Provides translation invariance

4.3 Case studies: LeNet, AlexNet, VGGNet, ResNet, GoogleNet etc.

LeNet-5 (1998)

Proposed by: Yann LeCun

Purpose: Handwritten digit recognition

Key Features:

- One of the earliest CNNs
- Uses convolution, pooling, and fully connected layers
- Small and shallow network

Architecture:

- Input → Conv → Pool → Conv → Pool → FC → Output

Significance:

Foundation of modern CNN architectures.

AlexNet (2012)

Proposed by: Alex Krizhevsky

Purpose: ImageNet classification

Key Features:

- First deep CNN to win ImageNet
- Uses **ReLU** activation
- Introduced **dropout** and **data augmentation**
- GPU-based training

Architecture:

- 5 Convolutional layers + 3 Fully Connected layers

Significance:

Started the deep learning revolution in computer vision.

VGGNet (2014)

Proposed by: Visual Geometry Group (Oxford)

Key Features:

- Very deep network (VGG-16, VGG-19)
- Uses only **3×3 convolution filters**
- Simple and uniform architecture

Advantages:

- Strong feature extraction
- Easy to understand and implement

Disadvantages:

- Very large number of parameters
- High memory and computation cost

GoogleNet / Inception (2014)

Proposed by: Google

Key Features:

- Introduced **Inception modules**
- Uses multiple filter sizes (1×1, 3×3, 5×5) in parallel
- Efficient parameter usage

Advantages:

- Reduced computation
- Deeper but efficient network

Significance:

Improved performance with fewer parameters.

ResNet (2015)

Proposed by: Microsoft Research

Key Features:

- Introduced **residual connections (skip connections)**
- Solves vanishing gradient problem
- Enables very deep networks (ResNet-50, 101, 152)

Core Idea:

$H(x)=F(x)+x$

Advantages:

- Faster convergence
- Higher accuracy
- Very deep architectures possible