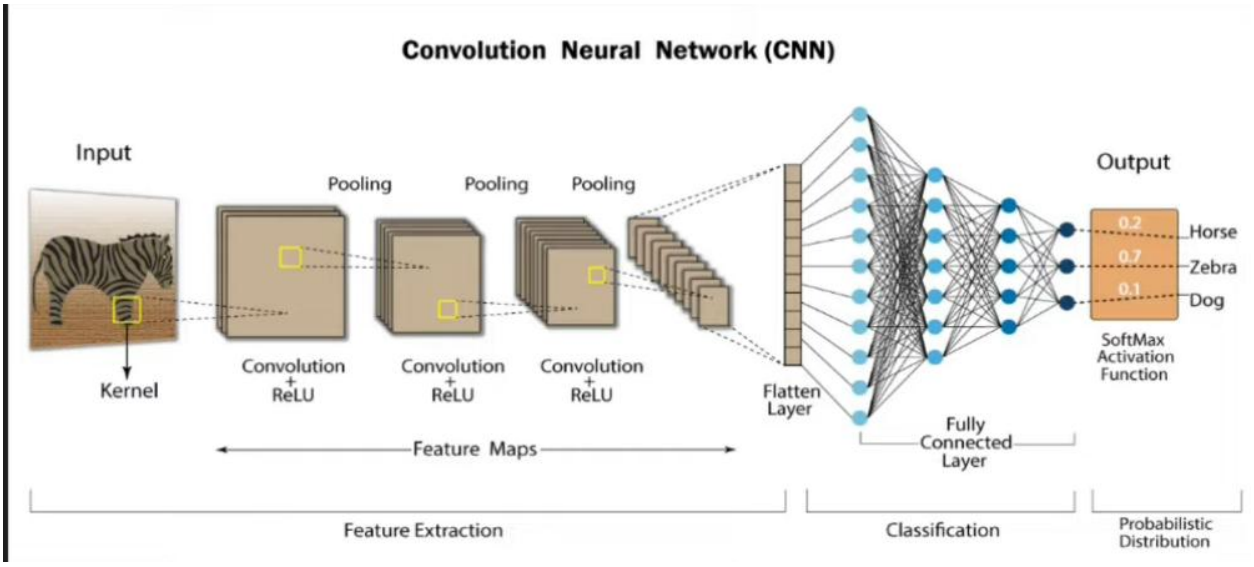
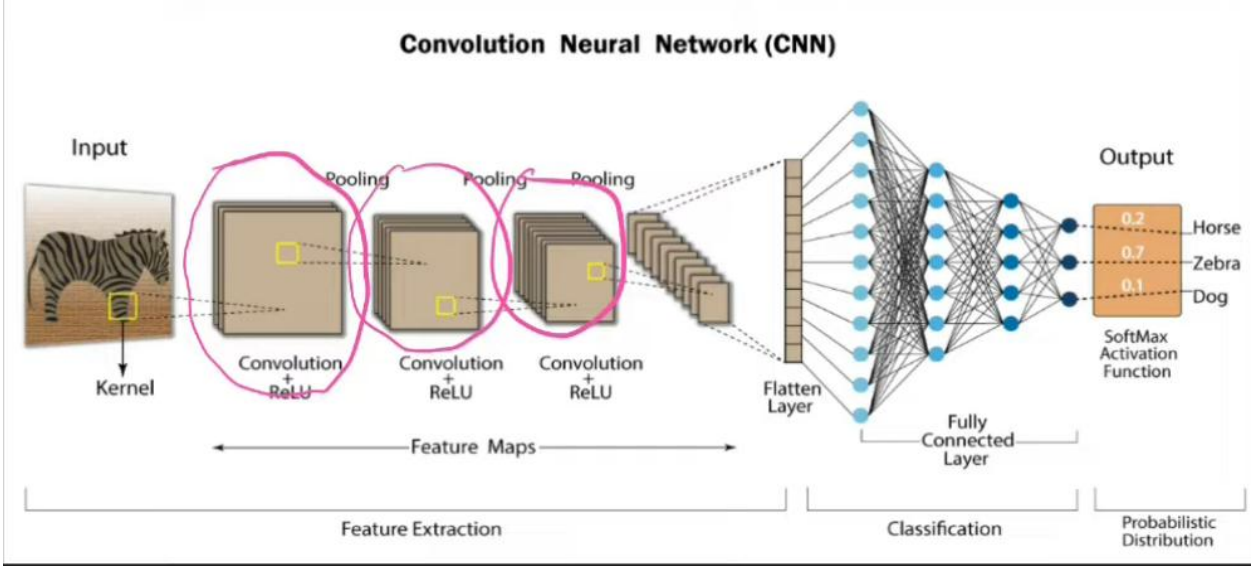
What is a CNN (Convolutional Neural Network)?

Convolutional Neural Networks (CNNs), also called **ConvNets**, are a special type of neural network designed to process data that has a **grid-like structure**, such as:

- **1D data** → Time-series signals
- **2D data** → Images (pixels arranged in rows and columns)





ANN → Matrix Multiplication

CNN → Convolution (NOT simple matrix multiplication)

Core Components of a Convolutional Neural Network (CNN)

1. Convolution Layer

- Applies **filters (kernels)** over the input
- Extracts **local features** such as:
  - Edges
  - Corners
  - Textures
- Preserves **spatial relationships**

---

2. Pooling Layer

- Performs **downsampling**
- Common types:
  - **Max Pooling**
  - Average Pooling
- Benefits:
  - Reduces dimensionality
  - Lowers computation
  - Controls overfitting
  - Adds translation invariance

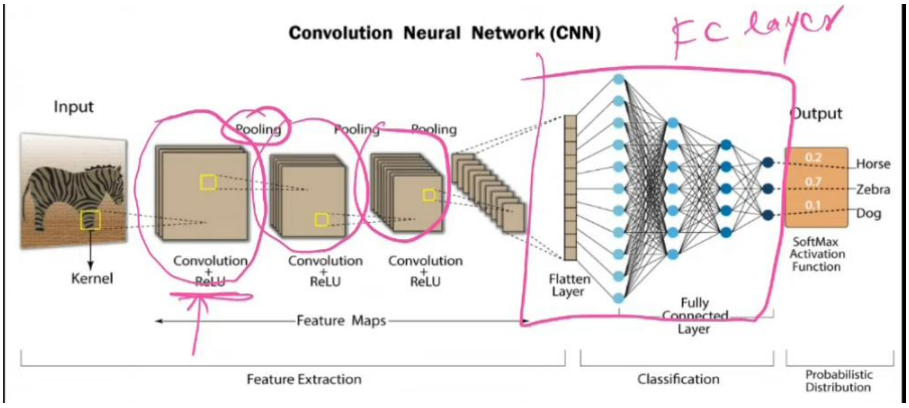
---

3. Fully Connected (FC) Layer – ANN

- Operates like a **traditional ANN**
- Takes flattened feature maps as input
- Performs **classification or regression**

CNN Workflow (One-Line)

Convolution → Pooling → Fully Connected Layer (ANN)



Convolutional Neural Networks (CNN) – Inspiration & History

Inspiration

- CNNs are inspired by the **human visual cortex**
- The visual cortex processes visual information hierarchically (edges → shapes → objects)

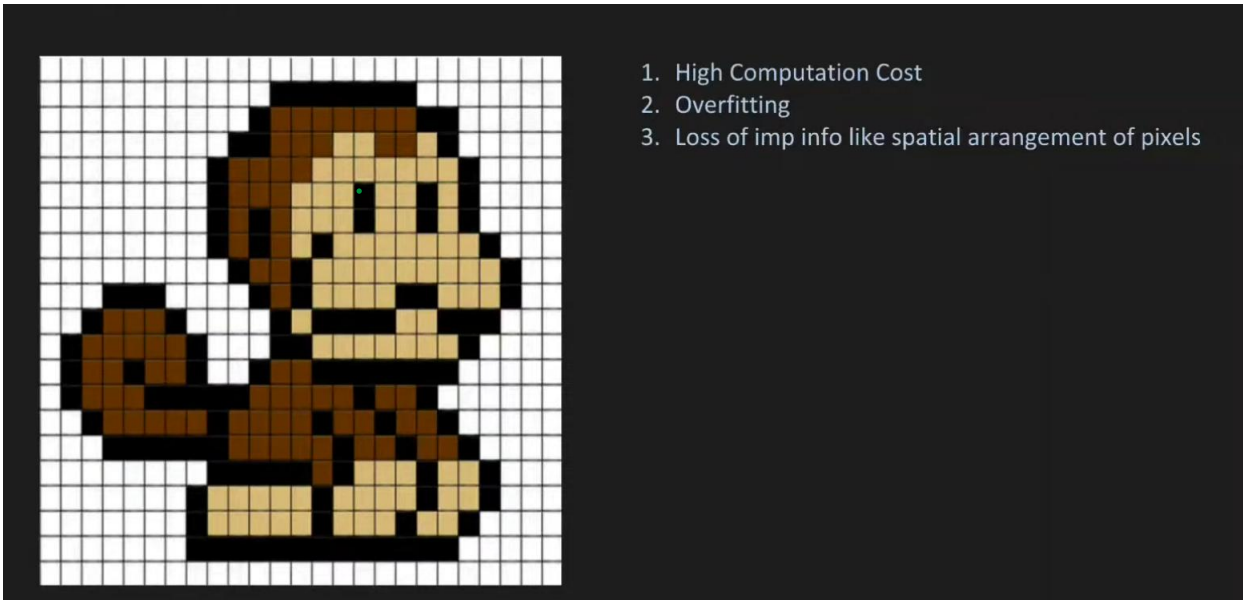
Key Milestones

- In 1998, Yann LeCun developed Convolutional Neural Networks (LeNet) at AT&T Bell Labs, which were successfully used for handwritten digit recognition on bank checks.
- **Microsoft**
    - Applied CNNs to **OCR (Optical Character Recognition)**
    - Used for **handwriting recognition**

Major Applications of CNN

- **Handwriting recognition**
- **Optical Character Recognition (OCR)**
- **Facial recognition**
- **Self-driving cars / autonomous driving**
- General **computer vision tasks**

Why not use ANN



Why Plain ANN is NOT Ideal for Images (Motivation for CNN)

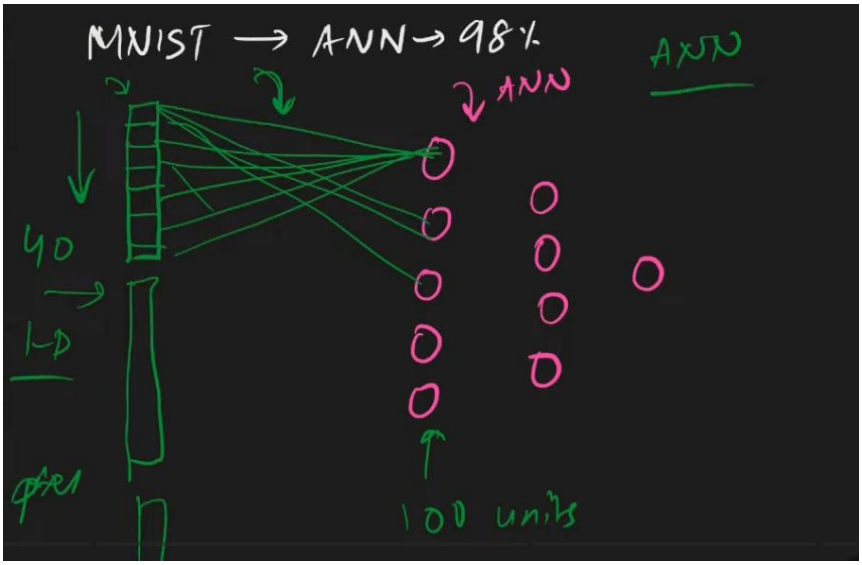
Input Representation

- Image shown is **40 × 40**
- Total input features = **1600 pixels**
- When using **ANN**, the image is **flattened into 1D**
    - Shape: 40 × 40 → 1600 × 1

Problems with ANN on Images

1. High Computational Cost ✅

- Fully connected layers connect **every input pixel to every neuron**



- Example:
  - 1600 inputs × 100 hidden units = **160,000 parameters**
- As image size increases → parameters explode

---

2. Overfitting

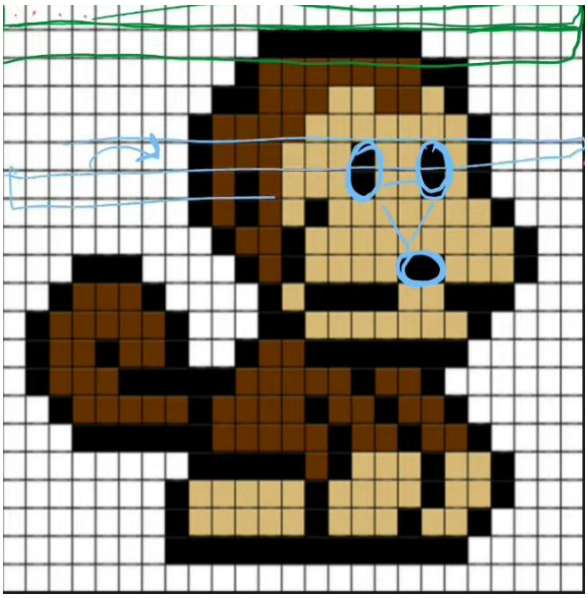
- Too many parameters
- Model memorizes training data instead of learning patterns
- Especially problematic with limited datasets (e.g., MNIST without augmentation)

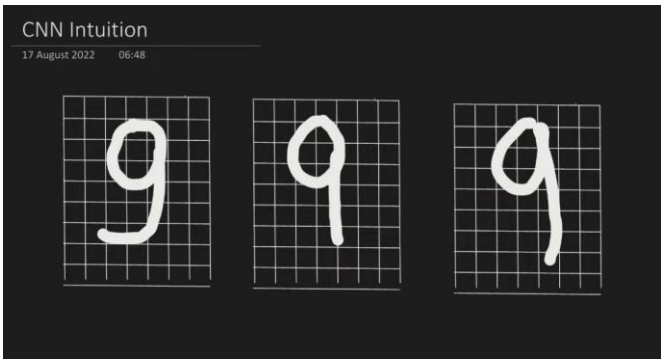
---

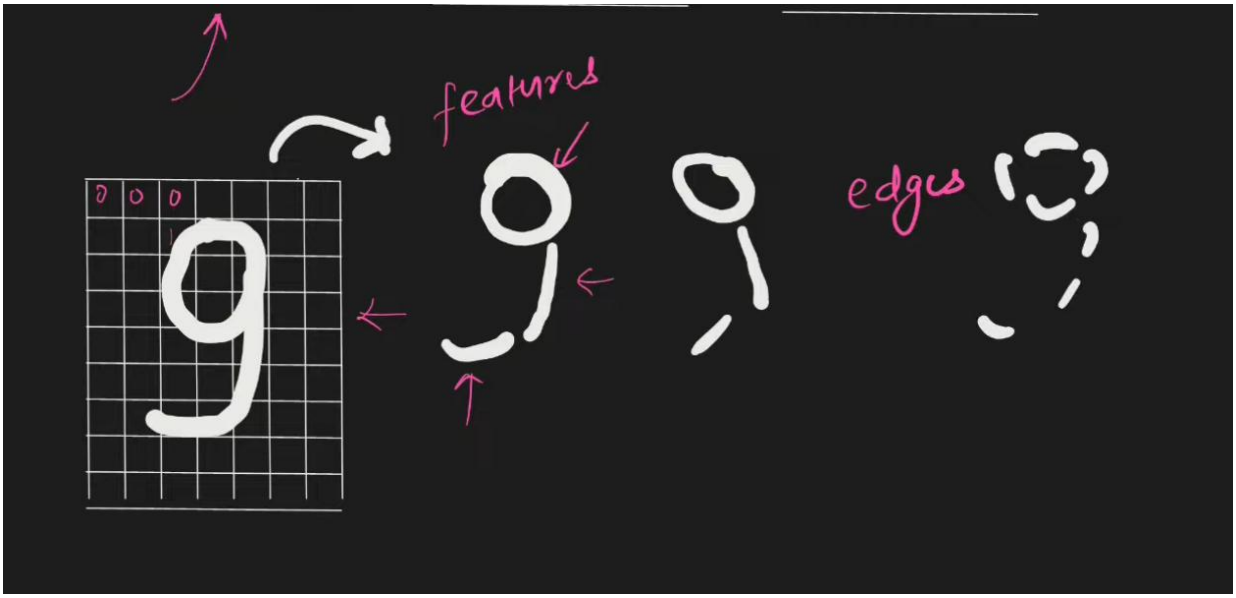
3. Loss of Important Spatial Information

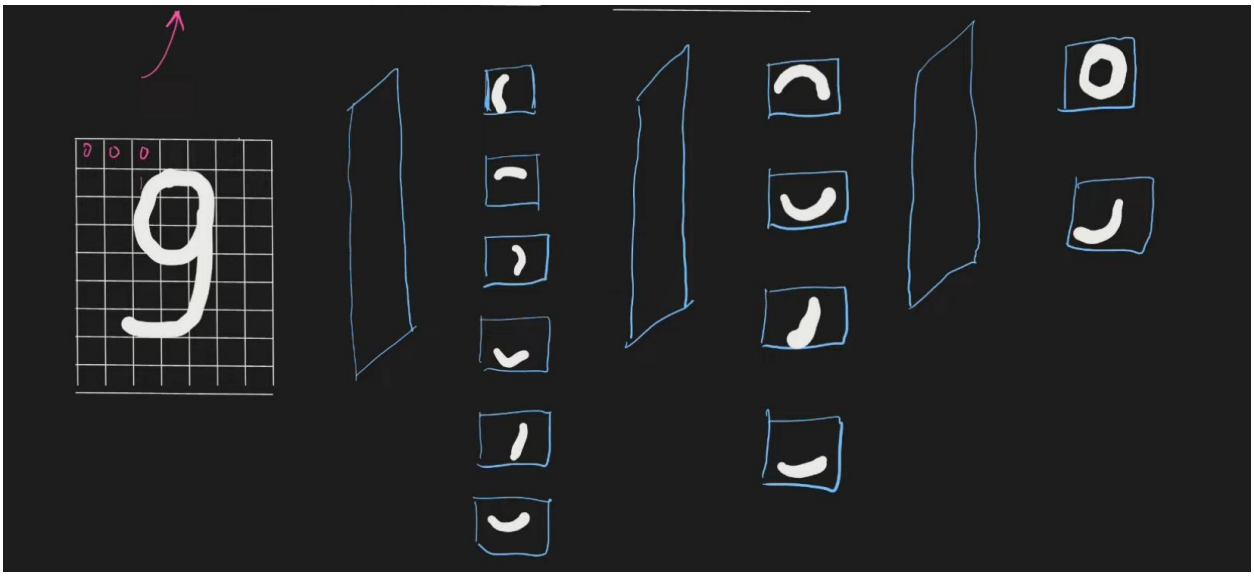
- Flattening destroys **spatial relationships**
- ANN does **NOT know**:

    - Which pixels are neighbors
    - Shape, edges, or orientation
- Example:
    - Eye pixels being close together is lost after flattening






features
edges



What This Diagram Explains: Feature Extraction in CNN

Step 1: Input Image

- Input is a handwritten digit **"9"**
- Represented as a **2D grid of pixels**
- Each pixel has an intensity value

Step 2: First Convolution Layer – Low-Level Features

- Small **filters (kernels)** slide over the image
- Each filter detects a **simple local feature**, such as:
    - Short vertical strokes
    - Small curves
    - Edges

📌 Output:

- Multiple **feature maps**
- Each map highlights **where a specific feature appears**

Examples shown in the diagram:

- Small curved strokes
- Short line segments

Step 3: Deeper Convolution Layer – Mid-Level Features

- Takes feature maps from the previous layer as input
- Combines simple features into **more meaningful patterns**, such as:
    - Larger curves
    - Partial loops

- Stroke combinations

📌 Output:

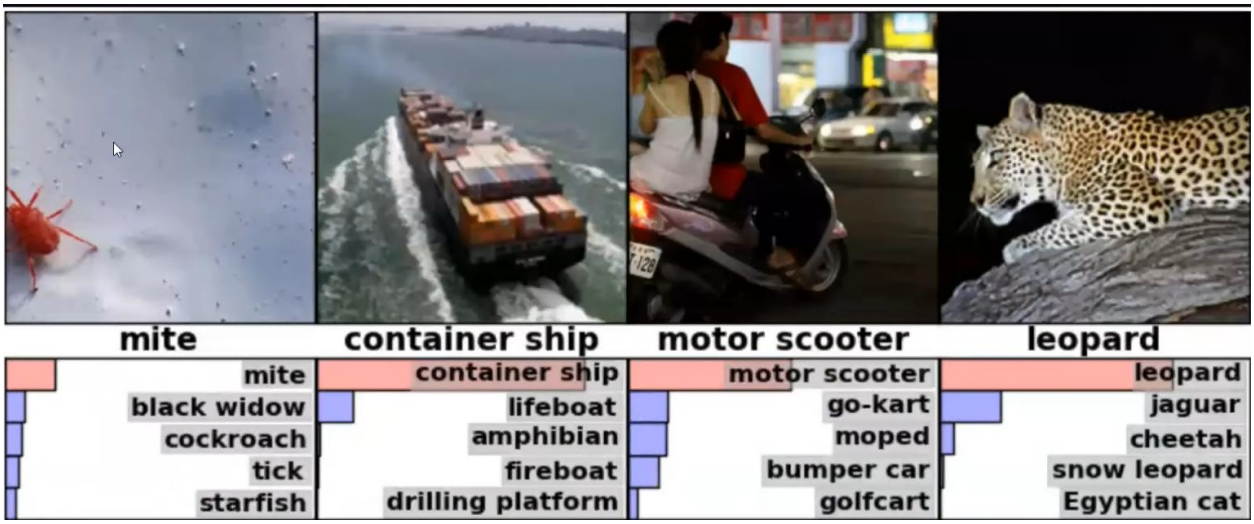
- Fewer but **richer feature maps**
- Clearer shape components of the digit

Step 4: Deeper Convolution Layer – High-Level Features

- Learns **complex structures**
- For digit “9”, this includes:
  - Circular loop at the top
  - Tail/curve at the bottom

📌 Output:

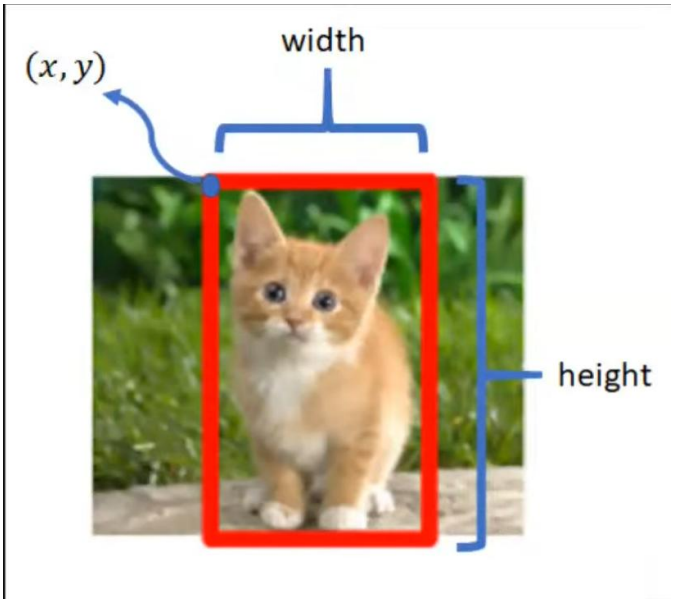
- Feature maps representing **distinct parts of the digit**



Object localization is the task of:

1. **Identifying the object in an image**, and
2. **Determining its exact location** using a **bounding box**.

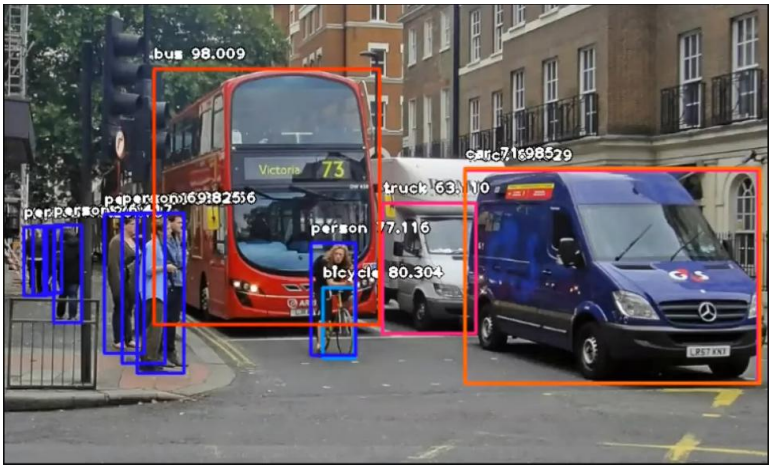
In the image, the object is a **cat**, and the red rectangle represents its **bounding box**.



Object detection is a computer vision task that:

1. **Identifies multiple objects in an image**
2. **Classifies each object**
3. **Locates each object using a bounding box**

👉 Unlike object localization, **object detection handles multiple objects at once**.



Confidence Score

- The number next to each label (e.g., 98.009)
- Represents the **model's confidence** that the detected object belongs to that class
- Higher value = higher certainty

How CNNs Are Used in Object Detection

Modern object detection models (YOLO, SSD, Faster R-CNN):

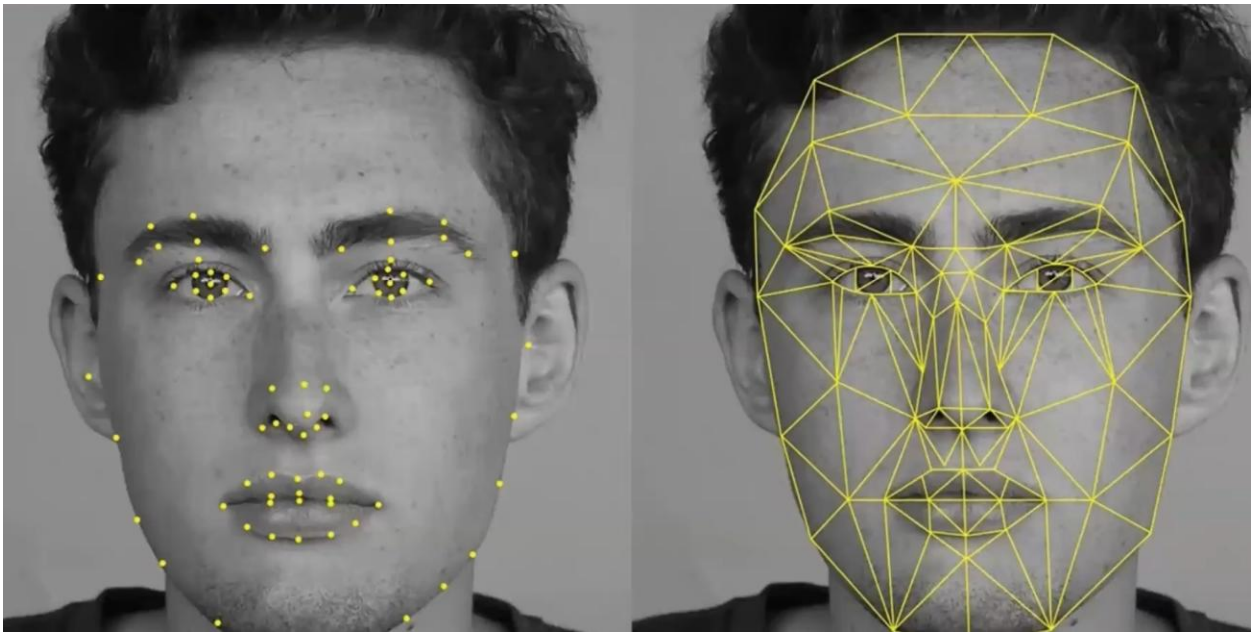
- Use **CNNs for feature extraction**
- Predict:
    - **Class probabilities**
    - **Bounding box coordinates**
- Perform detection in **one stage or two stages**

What is Face Recognition?

Face recognition is the task of:

- **Identifying or verifying a person** using their face

👉 It answers **“Who is this person?”**

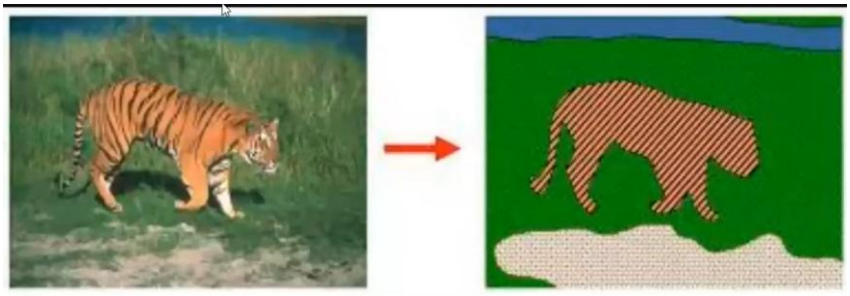


What is Image Segmentation?

Image segmentation is the task of:

- **Dividing an image into meaningful regions**
- Assigning a **label to every pixel** in the image

👉 Instead of drawing a box, segmentation colors **each pixel**.



Left Image

- Original image of a **tiger**

Right Image

- Image is **segmented**
- Different colors show different regions:
    - **Tiger (object)**
    - **Grass (background)**
    - **Ground**

📌 Every pixel belongs to a class.

How CNN Helps

1. Image is input to a **CNN**
2. CNN learns:
    - Shapes
    - Edges
    - Textures
3. CNN predicts a **label for every pixel**

Real-Life Applications

- Self-driving cars (road, pedestrian, vehicle)
- Medical imaging (tumor detection)
- Satellite images
- Photo editing tools

What is Black & White Image Colorization?

Image colorization is the process of:

- Converting a **black-and-white (grayscale) image**
- Into a **color image**

👉 The left image is **original (black & white)**
👉 The right image is the **colorized output**

What is Pose Estimation?

Pose estimation is a computer vision task that:

- Detects the **position of the human body**
- Identifies **key body points** such as:
    - Head
    - Shoulders
    - Elbows
    - Knees
    - Ankles
- Connects these points to show **body posture or movement**

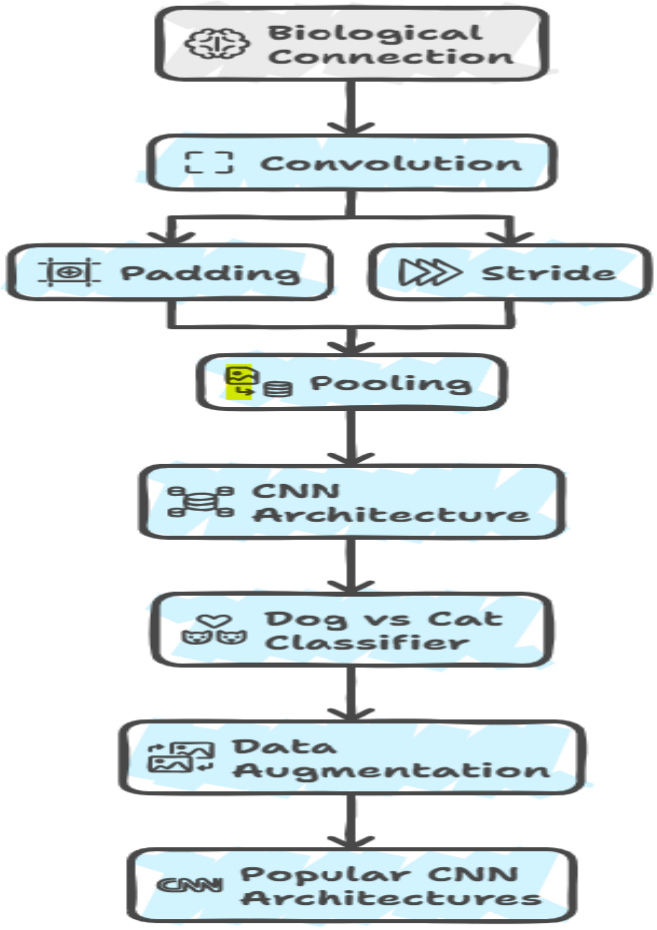
How CNN Helps

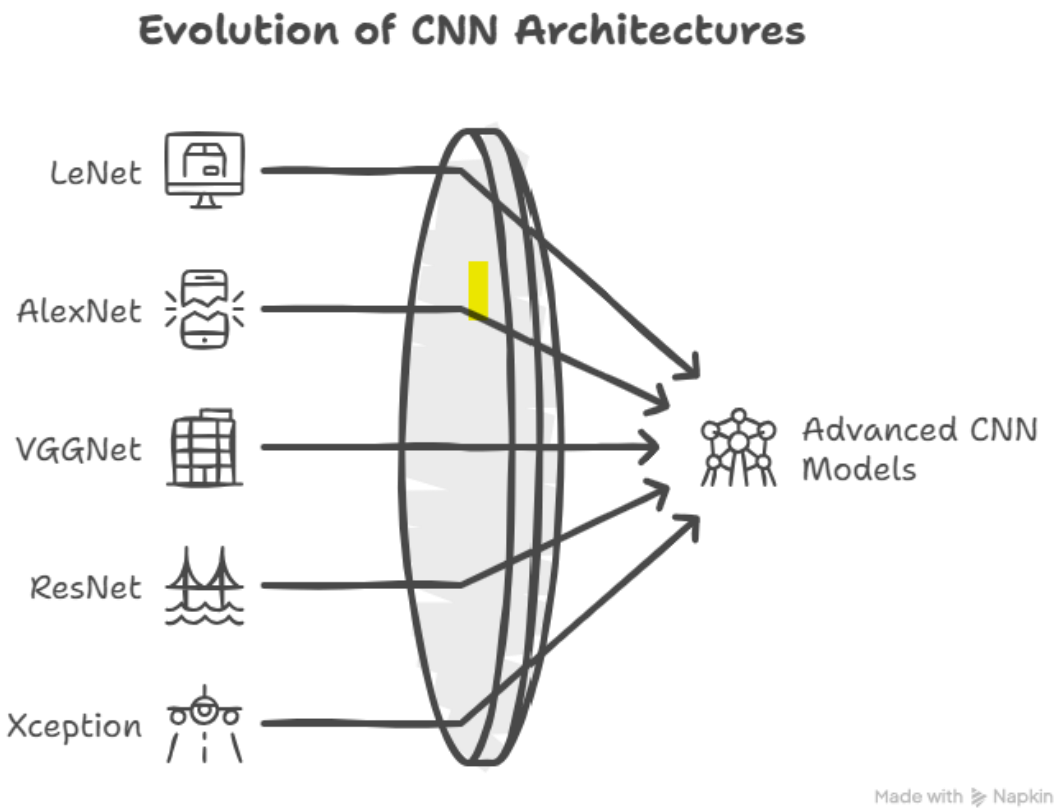
1. Image or video frame is given to a **CNN**
2. CNN detects **important body points**
3. CNN connects these points
4. Final output shows the **human pose**

Real-Life Applications

- Fitness apps (exercise posture)
- Sports analysis
- Dance & motion tracking
- Gaming & AR/VR
- Human activity recognition

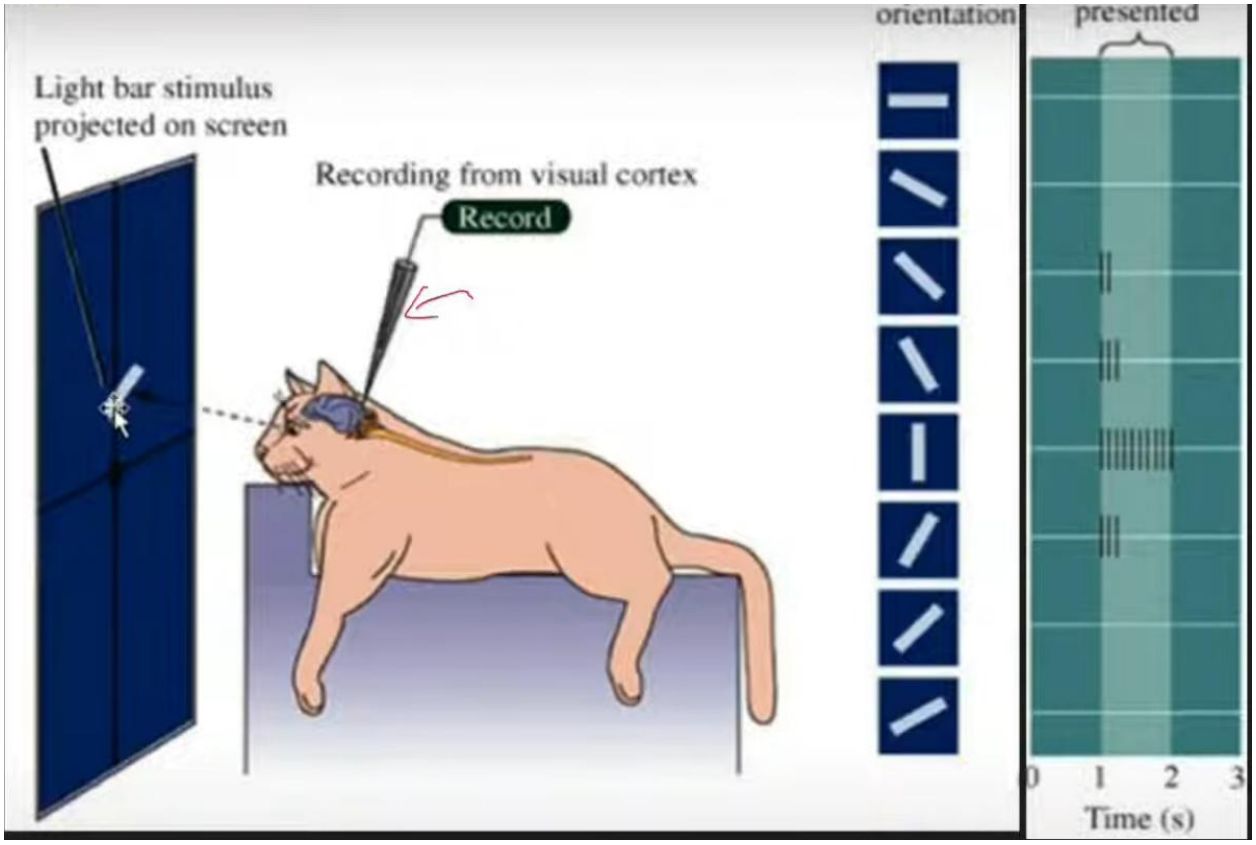
CNN Learning Roadmap




Evolution of CNN Architectures
LeNet
AlexNet
VGGNet
ResNet
Xception
Advanced CNN Models
Made with Napkin

Hubel & Wiesel's Experiment (1960s)

- In the **1960s**, **David Hubel and Torsten Wiesel** conducted famous experiments on **cats**
- They studied how a **cat's brain processes visual information**
- Their work focused on the **visual cortex**



What They Discovered

They found that:

- Neurons in the visual cortex **do not see the whole image**
- Each neuron responds only to a **small region** of the image
- Some neurons respond to:

- **Edges**
- **Lines**
- **Orientation (vertical, horizontal, slanted)**

Types of Cells They Identified

1. Simple Cells

Detect **basic features**, especially:

- **Edges**
- **Orientation (direction of lines)**

Have a **small receptive field**

- Respond to a small area of the image

Respond best to a **preferred stimulus**

2. Complex Cells

Detect the **same features**, but:

- Work over a **larger area**
- Are less sensitive to exact position

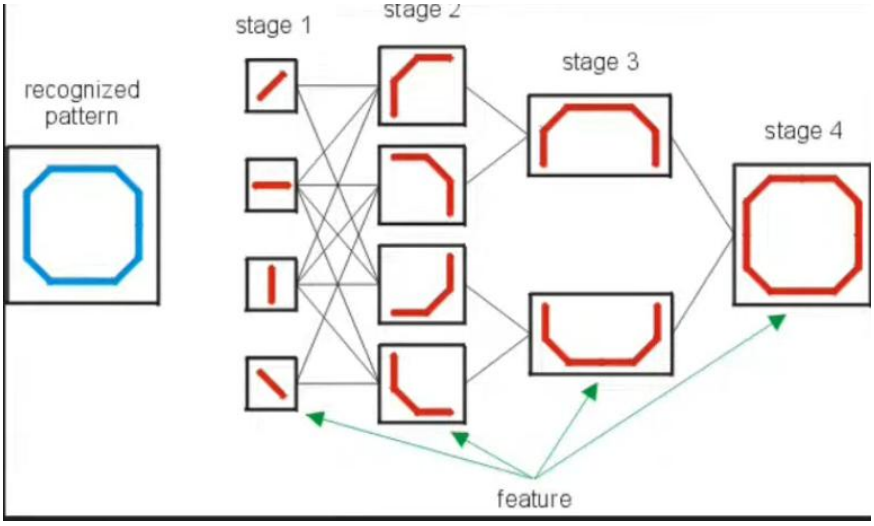
Have a **bigger receptive field**

- Help with movement and pattern recognition

Why This Is Important for CNN

Their experiments inspired CNN concepts:

- **Local receptive fields**
- **Hierarchical feature learning**
- **Edge → shape → object progression**



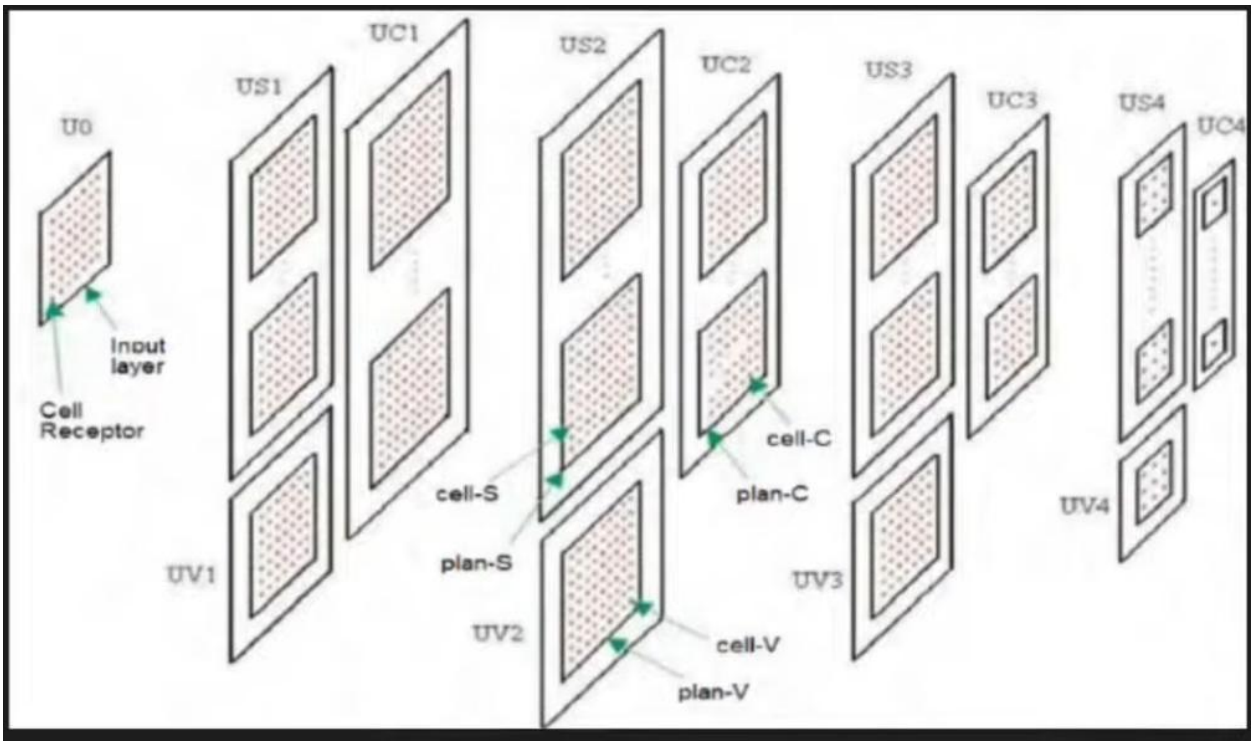
👉 CNNs work the same way:

- Early layers → edges
- Middle layers → shapes
- Deep layers → objects

Neocognitron – Fukushima (1980)

What is Neocognitron?

- **Neocognitron** is an early neural network model
- Proposed by **Kunihiko Fukushima**
- It is considered the **foundation of modern CNNs**



Key Idea Behind Neocognitron

Neocognitron was inspired by:

- **Hubel & Wiesel's experiments**
- **Simple cells and complex cells** in the visual cortex

Structure Shown in the Image (Simple Explanation)

The diagram shows **multiple layers**, where each layer learns more complex patterns.

1. Input Layer

- Receives the raw image
- Similar to image pixels

---

2. S-layers (Simple Cells)

- Detect **basic features**
  - Edges
  - Lines
  - Orientations
- Have **small receptive fields**

👉 Similar to **convolution layers** in CNN

---

3. C-layers (Complex Cells)

- Combine outputs of S-layers
- Provide **position tolerance**
- Have **larger receptive fields**

👉 Similar to **pooling layers** in CNN

---

4. Hierarchical Learning

- Early layers → simple features
- Middle layers → shapes
- Deeper layers → object parts

📌 This hierarchy is exactly how **CNNs work today**

Biology → Neocognitron → CNN

Biology	**Neocognitron**	**CNN**
Simple cells	S-layers	Convolution

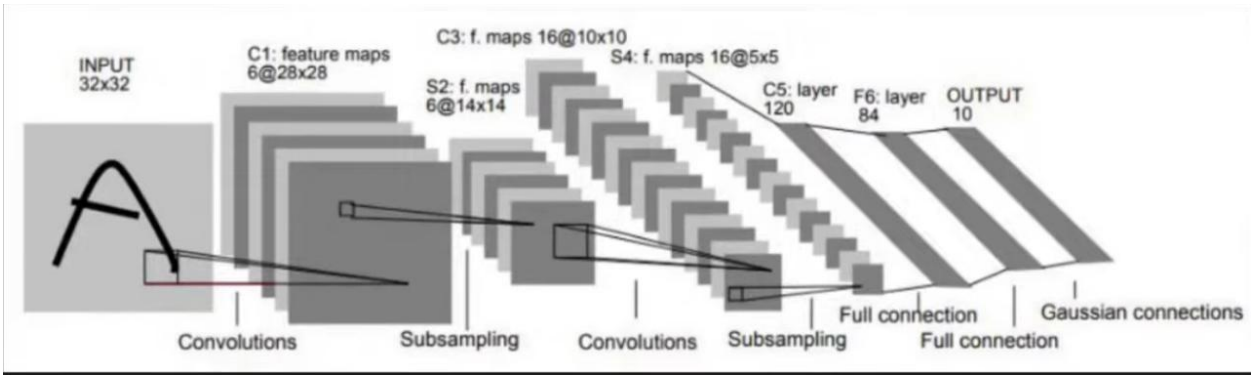
Biology	Neocognitron	CNN
Complex cells	C-layers	Pooling
Receptive fields	Layer hierarchy	Feature maps

Yann LeCun and Convolutional Neural Networks (CNN)

Yann LeCun introduced convolutional neural networks trained using backpropagation, leading to the LeNet architecture for handwritten digit recognition.

Simple Flow :

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



2012 – AlexNet – ImageNet Breakthrough

What Happened in 2012?

- In **2012**, **AlexNet** was introduced by **Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton**
- AlexNet won the **ImageNet Large Scale Visual Recognition Challenge (ILSVRC)**

Why AlexNet Was Important

- Achieved a **huge improvement in image classification accuracy**
- Reduced error rate dramatically compared to previous methods
- Marked the **deep learning revolution in computer vision**

Simple Timeline (Easy to Remember)

- **1980** → Neocognitron (Fukushima)
- **1998** → LeNet (Yann LeCun)
- **2012 → AlexNet + ImageNet = CNN breakthrough**

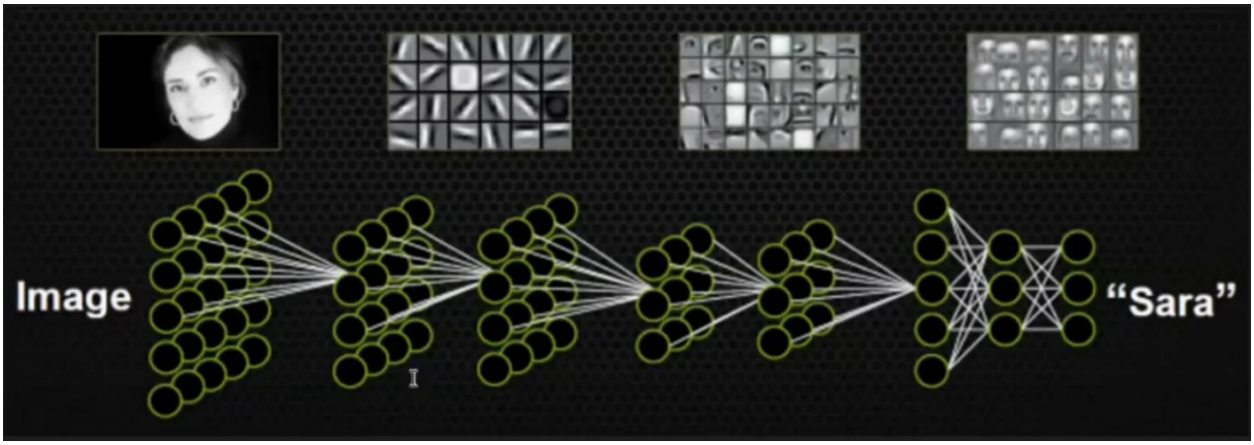
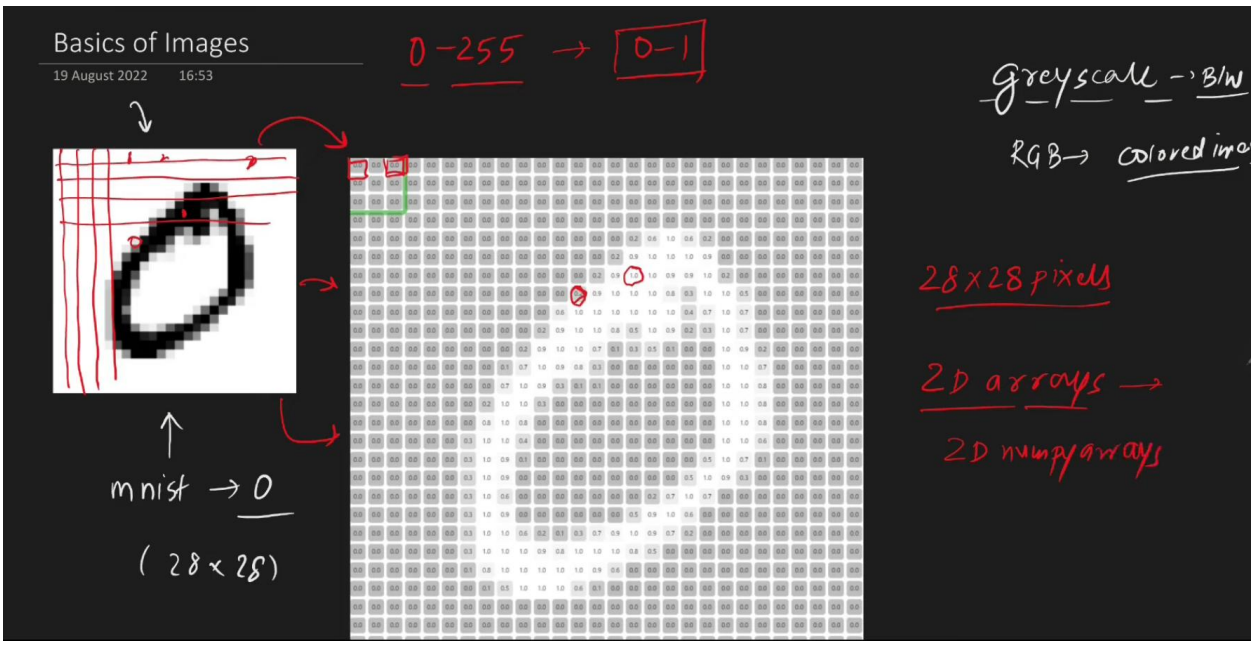


Image
"Sara"


CNN → NN special
convolution
Pooling
FC

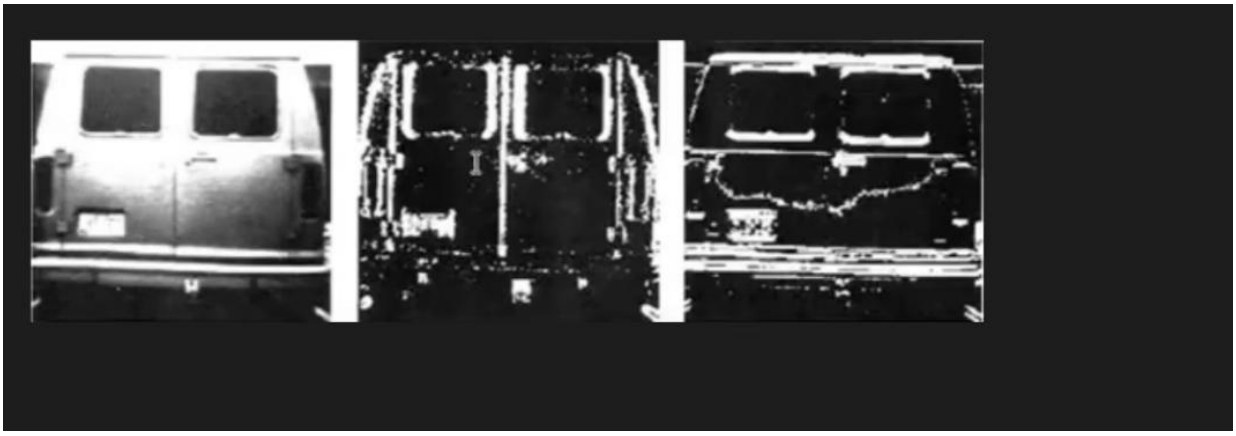

Basics of Images
19 August 2022 16:53
0-255 → 0-1
greyscale → B/W
RGB → colored ima
28x28 pixels
2D arrays →
2D numpy arrays
mnist → 0
(28x28)

Edge Detection (Using Convolution Operation)

What is Edge Detection?

Edge detection is a technique used to:

- Find **boundaries** in an image
- Detect places where pixel values **change sharply**
- Highlight **shapes and object outlines**



Convolution Output Size Formula

$$\text{Output size} = \left\lfloor \frac{n + 2p - f}{s} \right\rfloor + 1$$

Symbol Meaning

n	Input size (height or width)
p	Padding
f	Filter / kernel size
s	Stride
⌊ ⌋	Floor (round down)
+1	Accounts for the starting position

Why Are Strides Required in CNN?

1. High-Level Feature Learning

- Strides help the network **look at larger regions** of the image
- This increases the **receptive field**
- Useful for learning **high-level features** like:
    - Shapes
    - Object parts

- Patterns