Introduction to Deep Learning

Computer Vision

Important Definitions

- **Computer Vision**: A Field focused on enabling machines to interpret and understand visual information.
- **Image Processing**: Focuses on transforming images (e.g., filtering, enhancement) without necessarily understanding their content.
- **Recognition Tasks**:
  - **Classification**: Assign a label to the whole image.
  - **Detection**: Identify object locations.
  - **Segmentation**: Pixel-level labeling.
  - **Pose Estimation**: Infer the position/orientation of objects/people.
  - **Activity Recognition**: Understand actions from videos.
- **Bag of Visual Words**: A technique adapted from NLP to represent images using a vocabulary of local features.
- **Geons**: Basic 3D shapes used in the "recognition by components" theory.

What are the applications of Computer Vision?

High-Level Summary

Computer Vision enables machines to **see, interpret, and make decisions based on visual data**, with applications across healthcare, transportation, security, entertainment, manufacturing, and more.

Detailed Explanation

Computer Vision (CV) allows systems to extract meaningful information from images or videos and act upon it. These applications range from **simple classification** (e.g., identifying objects) to **complex decision-making systems** (e.g., autonomous driving). Here's a categorized overview:

🏥 Healthcare

- **Medical Imaging**: Detecting tumors in X-rays, CT, or MRI scans.
- **Histopathology**: Analyzing microscopic images of tissues.
- **Retinal Analysis**: Diagnosing diabetic retinopathy or glaucoma.

🚗 Autonomous Vehicles

- **Object Detection**: Identifying pedestrians, traffic lights, and other vehicles.
- **Lane Detection**: Recognizing road boundaries and lane markings.
- **Driver Monitoring**: Detecting drowsiness or distractions.

🏭 Manufacturing

- **Quality Inspection**: Detecting defects or anomalies in assembly lines.
- **Robotic Vision**: Enabling industrial robots to grasp and manipulate objects.

📱 Consumer Applications

- **Face Recognition**: Face unlock on smartphones, tagging in social media.
- **Augmented Reality (AR)**: Snapchat filters, AR games like Pokémon Go.
- **Photo Organization**: Google Photos recognizes people, scenes, and places.

🎮 Entertainment & Media

- **Motion Capture**: Capturing actor movements for animation and games.
- **Image/Video Enhancement**: Super-resolution, colorization, and style transfer.
- **Deepfake Detection/Creation**: Synthetic media generation and detection.

🔍 Security & Surveillance

- **Facial Recognition**: Identifying individuals in real time.

- **Activity Recognition**: Detecting suspicious behavior or intrusions.
- **License Plate Recognition**: Reading vehicle registration numbers.

🛒 Retail & E-Commerce

- **Visual Search**: Finding products from photos.
- **Shelf Monitoring**: Tracking inventory in physical stores.
- **Virtual Try-Ons**: Trying clothes or glasses using your camera.

🌾 Agriculture

- **Crop Monitoring**: Detecting plant diseases or pest infestations.
- **Yield Estimation**: Using drone images to assess crop health.

What are the vision problems that can best be solved in a generalized manner using ML/DL techniques?

Concept: Vision Problems Solved Using Machine Learning / Deep Learning

---

High-Level Summary

Machine Learning and Deep Learning allow us to **automatically learn patterns in visual data**, making them ideal for solving a broad range of vision tasks like classification, detection, pose estimation, and more—all through generalized, trainable models.

Detailed Explanation

The power of ML/DL comes from their ability to **learn complex features from data**, making them suitable for generalizing across multiple computer vision tasks. Below is a breakdown of key tasks and how ML/DL methods are used to solve them:

🧩 1. Classification

- **What it is**: Predict the class of the entire image (e.g., cat, dog, airplane).

- **How DL solves it**: CNNs extract features and pass them to fully connected layers for prediction.
- **Typical models**: ResNet, VGG, DenseNet.

📦 2. Detection

- **What it is**: Identify and locate objects in the image (bounding boxes + labels).
- **How DL solves it**: Combines feature extraction with localization.
- **Popular methods**:
  - **Two-stage**: R-CNN, Fast R-CNN, Faster R-CNN.
  - **One-stage**: YOLO (You Only Look Once), SSD (Single Shot Detector).
- **Output**: List of bounding boxes with object classes and confidence scores.

🕺 3. Pose Estimation

- **What it is**: Estimate the positions of keypoints (joints) of the human body.
- **How DL solves it**: Uses CNNs or Heatmap regression to predict keypoint coordinates.
- **Typical models**: OpenPose, HRNet.
- **Applications**: Sports analysis, animation, healthcare posture assessment.

🎥 4. Activity Recognition

- **What it is**: Recognize actions from video clips (e.g., jumping, running, cooking).
- **How DL solves it**:
  - Uses 3D CNNs or RNNs to model **temporal dynamics**.
  - Transformer-based models are also emerging (e.g., VideoBERT).
- **Input**: Sequence of frames or short video clips.
- **Datasets**: UCF101, ActivityNet, Kinetics.

🧠 5. Object Recognition

- **What it is**: General term combining **detection + classification + segmentation**.
- **DL solution**: Uses modular or end-to-end CNN models.
- **Context**: Core task in CV, and deep learning has significantly improved performance over traditional techniques.

🧍‍♀️ 6. Segmentation

- **What it is**: Pixel-wise classification.
  - **Semantic Segmentation**: Classifying each pixel (e.g., all cars = 1 class).
  - **Instance Segmentation**: Differentiating each instance (e.g., car #1, car #2).
- **DL solution**:
  - **U-Net**, **DeepLab**, **Mask R-CNN**.
- **Applications**: Medical imaging, autonomous driving, satellite imagery.

📝 7. Describing Images with Language (Image Captioning)

- **What it is**: Generating natural language captions for an image.
- **How DL solves it**:
  - **Encoder-Decoder architecture**:
    - **CNN**: Encodes image features.
    - **RNN/LSTM or Transformer**: Decodes into text.
- **Popular model**: Show and Tell, Show-Attend-and-Tell, CLIP, BLIP.
- **Emerging trend**: Vision-language models like **GPT-4V**, **Flamingo**, **GIT**.

Analogy

Think of deep learning as a **"universal translator"** for vision—it sees pixels and learns to **map them to meaningful labels, shapes, movements, or words** just like how humans recognize and describe the world.

Mathematical Foundation

All these tasks rely on:

- **Feature extraction using CNNs**: Learning hierarchical visual patterns.
- **Loss functions**:
  - Classification: Cross-entropy
  - Detection: Classification + localization loss (e.g., IoU)
  - Segmentation: Dice loss, pixel-wise cross-entropy
  - Captioning: Sequence loss (e.g., negative log-likelihood, BLEU)

Example (cross-entropy loss):

$$\mathcal{L} = -\sum_{i} y_i \log(\hat{y}_i)$$

Use Case

In a self-driving car:

- **Classification**: Detect traffic sign type.
- **Detection**: Locate pedestrians.
- **Pose estimation**: Track the body posture of pedestrians.
- **Activity recognition**: Predict if someone is about to cross.
- **Segmentation**: Understand road vs. sidewalk.
- **Captioning**: Generate a verbal description for navigation systems.

Why is recognition a difficult problem?

Concept: Why is Recognition a Difficult Problem in Computer Vision?

High-Level Summary

Recognition is hard because **visual inputs are highly variable** due to changes in viewpoint, lighting, scale, occlusion, and background — yet humans can generalize well, and teaching machines to do the same is challenging.

Detailed Explanation

Visual recognition refers to identifying **what** is in an image or scene (e.g., a car, person, cat). This task is deceptively complex because the **same object can look dramatically different** under different circumstances. Here are the key challenges:

🔁 1. Viewpoint Variation

- An object may look completely different from another angle (top, side, back, etc.).
- Example: A car viewed from the front vs. the side.

💡 2. Illumination Changes

- Lighting conditions (daylight, shadows, artificial light) drastically affect image appearance.
- Example: A face in sunlight vs. the same face in a dimly lit room.

⬛ 3. Occlusion

- Objects may be **partially hidden** by other objects.
- Example: A person behind a tree—only part of the body is visible.

🔍 4. Scale Variation

- The same object may appear larger or smaller depending on the distance.
- Models must learn to **recognize the same object at multiple sizes**.

🌀 5. Deformation

- Objects, especially living beings, **change shape** (pose, expression, body movement).
- Example: A dog sitting vs. jumping.

🧱 6. Background Clutter

- Objects often appear in **complex, noisy backgrounds**, which can confuse the model.
- Example: A white cat on a white couch.

🔄 7. Intra-Class Variation

- Different instances of the same object class may look very different.
- Example: Sports cars and hatchbacks are both “cars” but differ in shape, color, and size.

❓ 8. Local Ambiguity

- Small patches of an image may resemble many different things.
- Example: A close-up of fur could be from a cat, dog, or stuffed toy.

Analogy

Imagine trying to recognize your friend in:

- A dark room,
- Wearing different clothes,
- With a new haircut,
- From a side angle,
- Partially blocked by a crowd.

That’s what computer vision models face all the time — **limited and noisy visual cues, but still expected to make accurate decisions.**

Mathematical Foundation

Formally, recognition involves **learning a mapping**:

$f : \mathbb{R}^{H \times W \times C} \rightarrow \mathcal{Y}$

Where:

- $\mathbb{R}^{H \times W \times C}$: Input image (height, width, channels),
- $\mathcal{Y}$: Output label space (e.g., cat, dog, car).

This function must be **robust to transformations**:

- Translation, rotation, scale
- Illumination functions

- Partial occlusion functions

CNNs learn **invariant features** to make this possible, but perfect invariance is difficult.

Use Case

In **facial recognition systems**, these challenges can cause errors:

- People wearing masks (occlusion)
- Nighttime or low light (illumination)
- Profile view vs. front view (viewpoint)

  This makes high-accuracy recognition difficult, especially in unconstrained environments.

Main Idea of Deep Learning

High-Level Summary

The main idea of deep learning is to **automatically learn hierarchical representations** (features) from raw data using multiple layers of neural networks — eliminating the need for manual feature engineering.

Detailed Explanation

Traditionally, machine learning relied on **handcrafted features** (edges, shapes, colors), which required deep domain expertise and did not generalize well across tasks.

Deep Learning changes this by:

- Using **neural networks with many layers** ("deep" networks).
- Each layer **learns a transformation** of the input, building from **simple to complex** features.
- **Early layers** detect simple patterns (e.g., edges).
- **Middle layers** combine them into shapes (e.g., corners, textures).
- **Deeper layers** recognize complex objects (e.g., faces, animals, vehicles).

Instead of telling the model what features to look for, we **feed it raw data (images, audio, text)** and let it **learn the best features** automatically by minimizing a loss function.

This makes deep learning **scalable**, **data-driven**, and capable of solving **a wide variety of tasks** across domains.

Analogy

Think of deep learning like teaching a child to recognize animals:

- You **don’t give rules** like "cats have pointy ears."
- Instead, you show **many examples** of cats and dogs.
- Over time, the child **figures out** what makes a cat a cat — this is **representation learning** in deep learning.

Mathematical Foundation

Deep learning models are typically deep neural networks composed of **multiple layers of neurons**.

A basic forward pass through one layer looks like:

$$z = Wx + b$$
$$a = \sigma(z)$$

Where:

- x: Input vector
- W: Weights matrix
- b: Bias
- σ: Activation function (e.g., ReLU, Sigmoid)
- a: Output/activation

The **depth** (number of layers) enables learning **high-level abstract features**.

Training is done using **backpropagation** and **gradient descent**, adjusting weights to minimize the loss.

Use Case

In image classification:

- Input: Raw pixels from an image
- Output: Predicted class (e.g., “dog”)
- The network learns filters to detect edges, shapes, textures, and entire objects — **no manual rules needed**.
