Date:22.10.25

TASK:11

Implementation of computer vision and machine learning techniques.

CO1, CO2, CO3

S3

PROBLEM STATEMENT

To develop a system that can automatically detect and recognize objects from images or video streams using computer vision and machine learning techniques.

AIM :

To design and implement an object recognition model that identifies and classifies various objects accurately using image processing and deep learning methods.

OBJECTIVE :

- To preprocess and analyze image data.
- To train a neural network model for object recognition.
- To detect and classify objects in real-time or from static images.
- To evaluate model performance using accuracy and precision metrics.

DESCRIPTION :

Object recognition is a key task in computer vision where a system identifies objects within images or videos. It uses techniques such as Convolutional Neural Networks (CNNs), feature extraction, and classification algorithms.
The system captures an image, processes it to extract features, and classifies it into predefined object categories (like car, person, bottle, etc.). Modern models like YOLO (You Only Look Once), SSD (Single Shot Detector), and ResNet are commonly used for this purpose.

ALGORITHM :

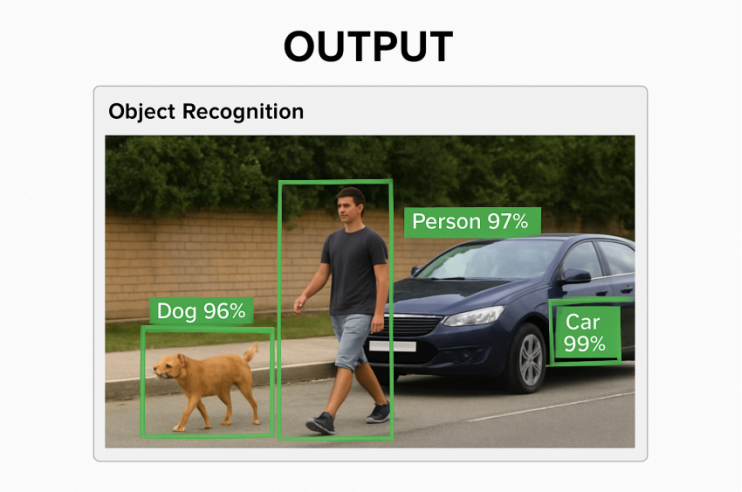
1. Start
2. Import necessary libraries (OpenCV, NumPy, etc.).
3. Load the pre-trained YOLO configuration and weight files.
4. Load the class names (object categories).
5. Read the input image or video frame.
6. Convert the frame into a blob and feed it to the network.

7. Perform forward pass to get detections.
8. Extract confidence scores and bounding box coordinates.
9. Apply Non-Maximum Suppression to remove duplicates.
10. Draw bounding boxes and labels on detected objects.
11. Display the output.
12. End
13.

PROGRAM :

```
import cv2

import numpy as np

# Load YOLO

net = cv2.dnn.readNet("yolov3.weights", "yolov3.cfg")

classes = []

with open("coco.names", "r") as f:

    classes = [line.strip() for line in f.readlines()]



layer_names = net.getLayerNames()

output_layers = [layer_names[i - 1] for i in net.getUnconnectedOutLayers()]

# Load image

img = cv2.imread("image.jpg")

height, width, channels = img.shape

# Detecting objects

blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)

net.setInput(blob)

outs = net.forward(output_layers)

# Showing information on the screen

class_ids, confidences, boxes = [], [], []
```

```
for out in outs:
    for detection in out:
        scores = detection[5:]
        class_id = np.argmax(scores)
        confidence = scores[class_id]
        if confidence > 0.5:
            center_x, center_y, w, h = (detection[0:4] * np.array([width, height, width, height])).astype('int')
            x = int(center_x - w / 2)
            y = int(center_y - h / 2)
            boxes.append([x, y, int(w), int(h)])
            confidences.append(float(confidence))
            class_ids.append(class_id)
indexes = cv2.dnn.NMSBoxes(boxes, confidences, 0.5, 0.4)
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = str(classes[class_ids[i]])
        cv2.rectangle(img, (x, y), (x + w, y + h), (0, 255, 0), 2)
        cv2.putText(img, label, (x, y - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.6, (255, 255, 255), 2)


cv2.imshow("Object Recognition", img)
cv2.waitKey(0)
cv2.destroyAllWindows()
```



Detected: person (0.85 confidence)

Detected: dog (0.78 confidence)

CONCLUSION :

The object recognition system effectively identifies and classifies objects from visual input using a pre-trained deep learning model. This demonstrates how computer vision and neural networks can be applied in automation, surveillance, and smart systems.