**VISUAL EYE TECHNOLOGY IN DEEP LEARNING**

**Introduction**

This document explains the project "Visual Eye Technology in Deep Learning," which uses the YOLO (You Only Look Once) object detection model to identify objects in real-time using a webcam. The project provides audio feedback for the detected objects using text-to-speech.

**Code Explanation**

The following sections provide a detailed explanation of the code.

**Import Libraries**

python

Copy code

import cv2

import math

import pyttsx3

from ultralytics import YOLO

* cv2: OpenCV library for computer vision tasks.
* math: Standard Python library for mathematical functions.
* pyttsx3: Text-to-speech conversion library.
* YOLO: YOLO (You Only Look Once) object detection model from the ultralytics package.

**Initialize Webcam**

python

Copy code

cap = cv2.VideoCapture(0)

cap.set(3, 640)

cap.set(4, 480)

* cap = cv2.VideoCapture(0): Initializes the webcam. The parameter 0 refers to the default webcam.
* cap.set(3, 640) and cap.set(4, 480): Set the width and height of the video frames.

**Load YOLO Model**

python

Copy code

model = YOLO("yolov8n.pt")

* Loads the YOLO model from the specified file yolov8n.pt.

**Define Object Classes**

python

Copy code

classNames = ["person", "bicycle", "car", "motorbike", "aeroplane", "bus", "train", "truck", "boat",

"traffic light", "fire hydrant", "stop sign", "parking meter", "bench", "bird", "cat",

"dog", "horse", "sheep", "cow", "elephant", "bear", "zebra", "giraffe", "backpack", "umbrella",

"handbag", "tie", "suitcase", "frisbee", "skis", "snowboard", "sports ball", "kite", "baseball bat",

"baseball glove", "skateboard", "surfboard", "tennis racket", "bottle", "wine glass", "cup",

"fork", "knife", "spoon", "bowl", "banana", "apple", "sandwich", "orange", "broccoli",

"carrot", "hot dog", "pizza", "donut", "cake", "chair", "sofa", "pottedplant", "bed",

"diningtable", "toilet", "tvmonitor", "laptop", "mouse", "remote", "keyboard", "cell phone",

"microwave", "oven", "toaster", "sink", "refrigerator", "book", "clock", "vase", "scissors",

"teddy bear", "hair drier", "toothbrush"]

* Defines a list of class names corresponding to the objects that the YOLO model can detect.

**Initialize Text-to-Speech Engine**

python

Copy code

text\_speech = pyttsx3.init()

* Initializes the text-to-speech engine.

**Main Loop**

python

Copy code

while True:

success, img = cap.read()

results = model(img, stream=True)

* while True: Infinite loop to continuously read frames from the webcam.
* success, img = cap.read(): Reads a frame from the webcam.
* results = model(img, stream=True): Passes the frame to the YOLO model for object detection.

**Process Detection Results**

python

Copy code

for r in results:

boxes = r.boxes

for box in boxes:

x1, y1, x2, y2 = box.xyxy[0]

x1, y1, x2, y2 = int(x1), int(y1), int(x2), int(y2)

cv2.rectangle(img, (x1, y1), (x2, y2), (255, 0, 255), 3)

confidence = math.ceil((box.conf[0] \* 100)) / 100

print("Confidence --->", confidence)

cls = int(box.cls[0])

print("Class name -->", classNames[cls])

org = [x1, y1]

font = cv2.FONT\_HERSHEY\_SIMPLEX

fontScale = 1

color = (255, 0, 0)

thickness = 2

cv2.putText(img, classNames[cls], org, font, fontScale, color, thickness)

text\_speech.say(f"{classNames[cls]} detected with confidence {confidence}")

text\_speech.runAndWait()

* Iterates over the detection results and processes each bounding box.
* x1, y1, x2, y2 = box.xyxy[0]: Gets the coordinates of the bounding box.
* Converts the coordinates to integer values.
* Draws a rectangle around the detected object.
* Calculates the confidence score of the detection.
* Retrieves the class index and prints the class name.
* Puts the class name text on the image.
* Converts the detection details to speech and speaks it out.

**Display the Result and Handle Exit**

python

Copy code

cv2.imshow('Webcam', img)

if cv2.waitKey(1) == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

* Displays the processed frame in a window named 'Webcam'.
* Waits for the 'q' key to be pressed to exit the loop.
* Releases the webcam and destroys all OpenCV windows.

**How to Use**

**Prerequisites**

* Python 3.x
* Required Python libraries: opencv-python, pyttsx3, ultralytics

**Installation**

1. Install Python from [python.org](https://www.python.org/).
2. Install the required libraries using pip:

bash

Copy code

pip install opencv-python pyttsx3 ultralytics

**Running the Project**

1. Clone the repository:

bash

Copy code

git clone https://github.com/yourusername/visual-eye-technology.git

cd visual-eye-technology

1. Run the script:

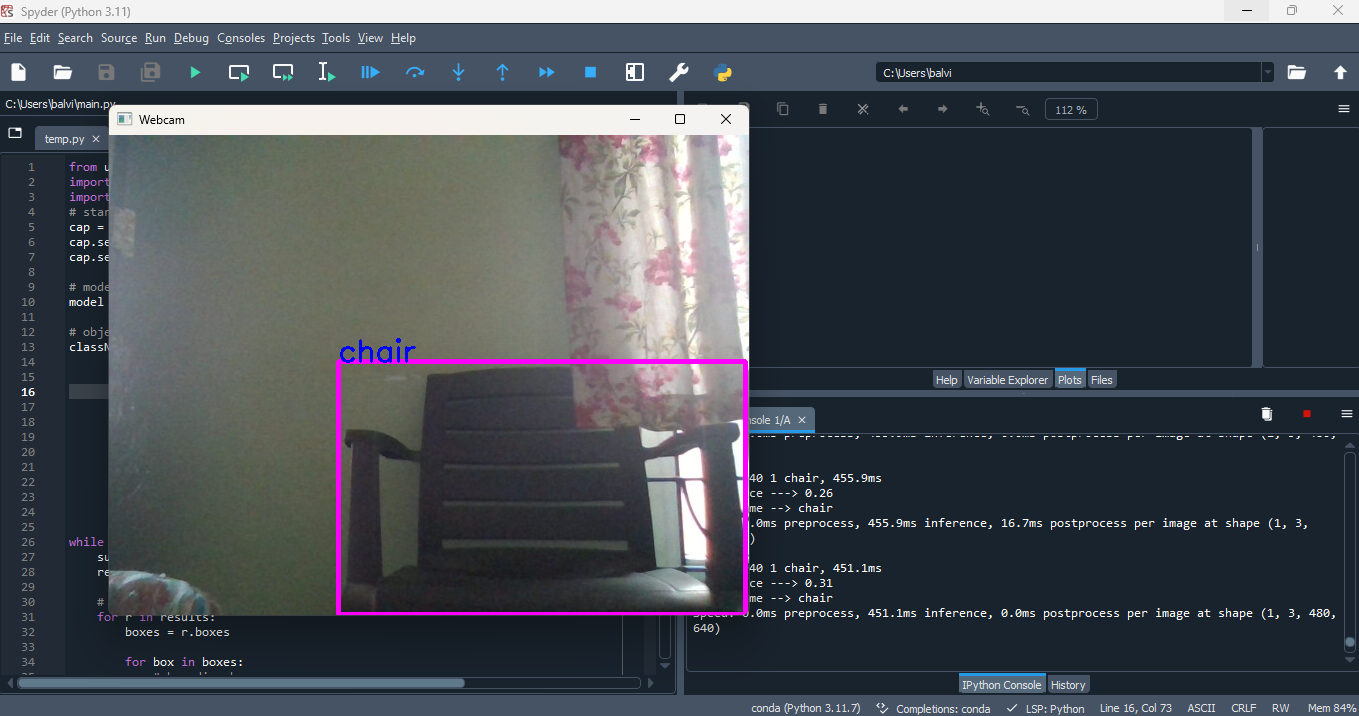
bash

Copy code

python your\_script\_name.py

1. The webcam will start, and the YOLO model will begin detecting objects in real-time. Detected objects will be highlighted with bounding boxes, and their names and confidence levels will be announced.

**Sample output:**



**Conclusion**

This project demonstrates the integration of computer vision and text-to-speech technologies to create an interactive system that detects and announces objects in real-time. It can be extended and modified for various applications, such as assistive technologies for visually impaired individuals.