Question 2: Real-time Object Detection Pipeline

For this project, I implemented real-time object detection using the YOLOv3-Tiny model, a lightweight version of the original YOLOv3 architecture optimized for speed. I utilized the yolov3-tiny.cfg file for model configuration, yolov3-tiny.weights for pretrained weights, and coco.names for class labels representing real-world objects. The model was successfully able to detect 12 different objects in the real-time video stream. A confidence threshold of 0.3 was applied to filter out low-confidence predictions.

The system was executed using CPU-only inference, which limited the frame rate to under 10 FPS. However, this performance can be significantly improved (15+ FPS) when executed with GPU acceleration (CUDA support).

The script begins by loading the YOLOv3-Tiny model using OpenCV's Deep Neural Network (DNN) module. It reads the configuration and weight files and loads the COCO class names. The video feed is captured from the webcam and resized. Each frame is preprocessed into a blob and passed to the model. The model returns bounding box predictions, which are filtered based on confidence and refined using Non-Maximum Suppression (NMS) to remove overlapping boxes. The remaining boxes are drawn on the frame with class labels and confidence scores. Frame rate (FPS) is also calculated and displayed. The program runs in real-time and exits when the user presses the 'q' key.