Object Detection

Syeda Fatima, G.Bhargavi, V.Mokshitha, L.Srihitha, Ch.Tejaswi
[1]Assistant Professor, Department Of Cse (AI&Ml), Bhoj Reddy Engineering College For Women, India.
[2,3,4,5]B. Tech Students, Department Of Cse (AI&Ml), Bhoj Reddy Engineering College For Women, India.

ABSTRACT

Object detection is a foundational task in the field of computer vision that focuses on identifying and localizing objects within digital images or videos. This mini project presents a real-time object detection system built using YOLOv4 (You Only Look Once Version 4), a state-of-the-art deep learning model known for its high detection accuracy and fast processing speed. The main objective of this project is to develop a lightweight, efficient, and adaptable object detection model suitable for real-time applications across diverse domains. The system has been implemented using Python and deep learning libraries such as TensorFlow and OpenCV, with Streamlit providing a user-friendly web interface. The model is trained on custom datasets to detect specific object classes, and it supports both image and video input. Through the interface, users can upload media or use a webcam to perform object detection and view results instantly with bounding boxes and labels drawn around identified objects.

1. INTRODUCTION

Object detection is a crucial task in computer vision that involves identifying and localizing objects within images or videos. This project presents a real-time object detection system utilizing the YOLOv4 (You Only Look Once version 4) deep learning model, known for its high speed and accuracy. The primary goal of this system is to detect and classify objects efficiently while maintaining low computational requirements, making it suitable for edge devices and real-time applications. The system is designed to be flexible and customizable, allowing users to train the model on specific object classes according to their application needs. Key application areas include security surveillance, industrial automation, autonomous vehicles, traffic monitoring, and healthcare. The YOLOv4-based approach offers significant advantages over traditional models like Faster R-CNN by delivering faster inference, reduced computational load, and simpler training processes.

Existing System

Traditional object detection systems primarily rely on models such as R-CNN, Fast R-CNN, and Faster R-CNN. These models use a two-stage detection pipeline, where the first stage generates region proposals and the second stage classifies these regions and refines their boundaries. While these models provide high accuracy, they are computationally expensive and have slower inference speeds, making them less suitable for real-time applications

Proposed System

To overcome the limitations of traditional object detection models, this project proposes a real-time object detection system based on YOLOv4 (You Only Look Once version 4). YOLOv4 is a single-stage object detection model that performs both object localization and classification in one unified process, resulting in significantly faster inference times.

2. REQUIREMENT ANALYSIS

Functional Requirements

In this project, we have designed the following modules:

Admin Module

- Manage user images and videos
- Monitor training progress and accuracy
- Manage users and access control
- View object detection analytics

User Module:

- Upload images or videos for object detection
- Perform real-time object detection
- View and download detection results

Non-Functional Requirements

Sofware Requirements

Operating System: Windows 10/11 or Ubuntu 20.04

Programming Language: Python 3.8

Libraries/Frameworks: OpenCV, TensorFlow, Streamlit.

Hardware Requirements

Processor: Intel i5 or higher

RAM: Minimum 8GB (16GB recommended)

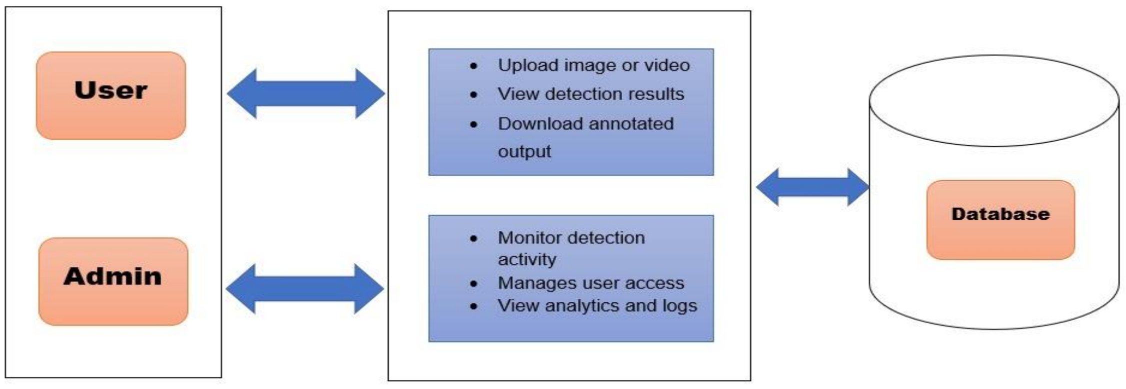
Storage: Minimum 50GB free space for dataset and model training.

- Performance: Real-time detection with minimal latency; YOLOv4 optimized for low-power devices; best accuracy after training.
- Scalability: Supports more object classes, multiple users, and larger datasets without performance loss.
- Security: Only authenticated users can access model training; data is securely stored; strong password policies enforced.
- Usability: User-friendly UI, clear detection results, configurable detection settings.
- Reliability & Maintainability: System should be fault-tolerant, ensuring continuous operation even in case of failures.

3. DESIGN

- Design represents the number of components we are using as a part of the project and the flow of request processing i.e., what components in processing the request and in which order.
- An architecture description is a formal description and representation of a system organized in a way that supports reasoning about the structure of the system.

System Architecture



System Architecture of Object Detection

Software Process Model

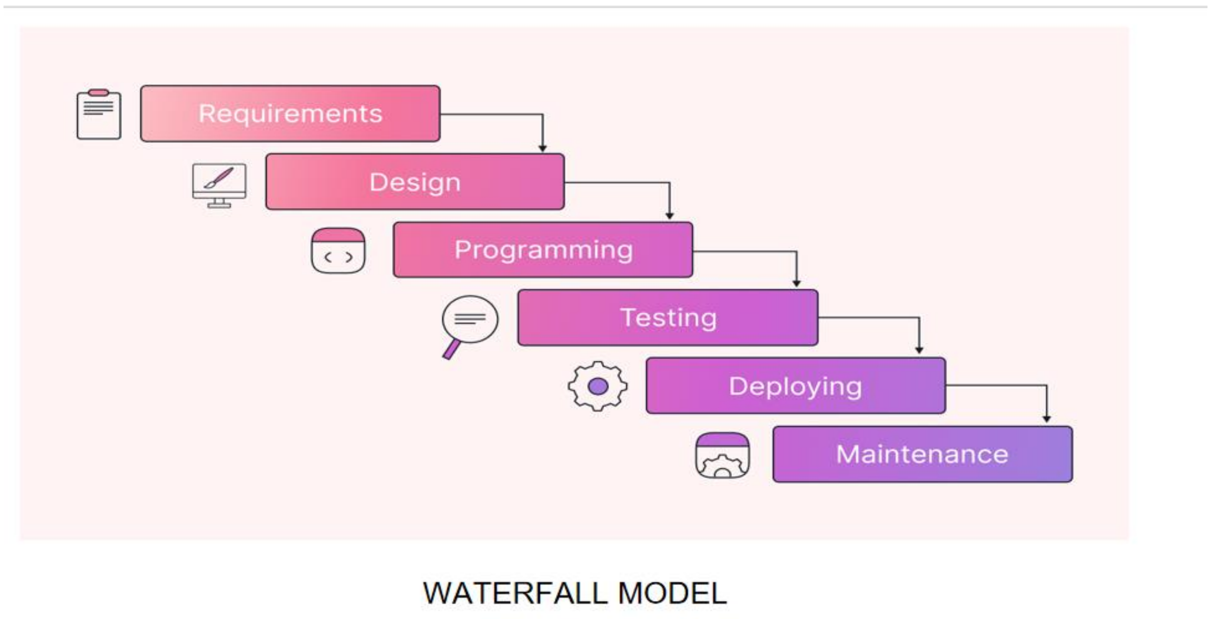


Fig 3.2 Software process model

4. IMPLEMENTATION

1. Environment Setup

Technology Used: Python, OpenCV, TensorFlow, Streamlit (optional)

Set up a Python development environment with required packages for computer vision, deep learning, and model deployment.

2. Data Input (Video/Images)

Technology Used: OpenCV

Reads video input or image sequences from the local directory. Enables frame-by-frame extraction and processing.

3. Frame Preprocessing

Technology Used: NumPy, OpenCV

Operations performed:

- Frame resizing
- Color conversion (BGR to RGB)
- Normalization for model compatibility

4. Object Detection

Technology Used: YOLOv4 (Darknet or TensorFlow version)

Loads pre-trained YOLOv4 weights and configuration. Detects objects in real-time with high accuracy and draws bounding boxes.

Implements detection through the Darknet-based .weights and .cfg files or converted .pb TensorFlow models

5. Output Annotation

Technology Used: OpenCV

Overlays detected object labels and bounding boxes on each frame. Adds class name and confidence score dynamically.

6. Result Saving

Technology Used: OpenCV

Uses VideoWriter to generate and save the annotated output video (e.g., output.avi) to the specified directory (data/video).

5-SCREENSHOTS

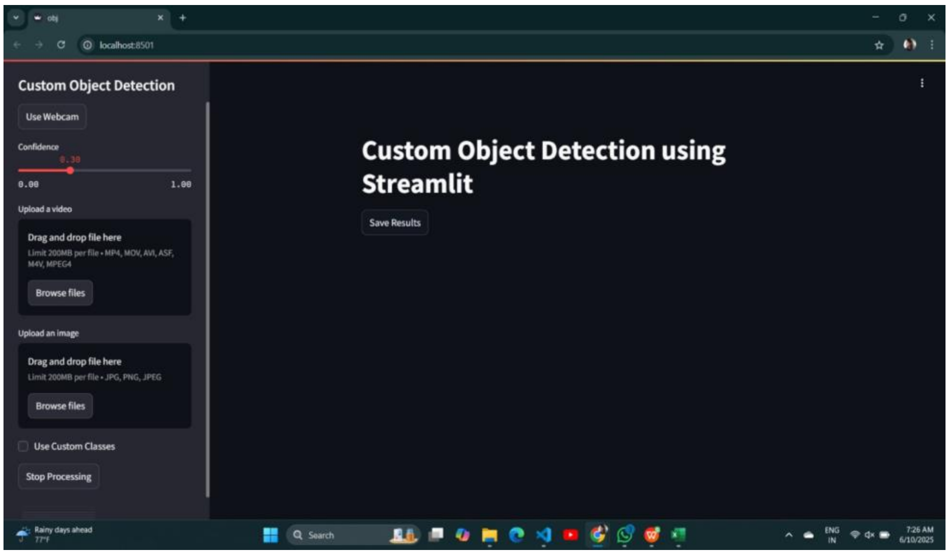


Fig 6.1 Output-1

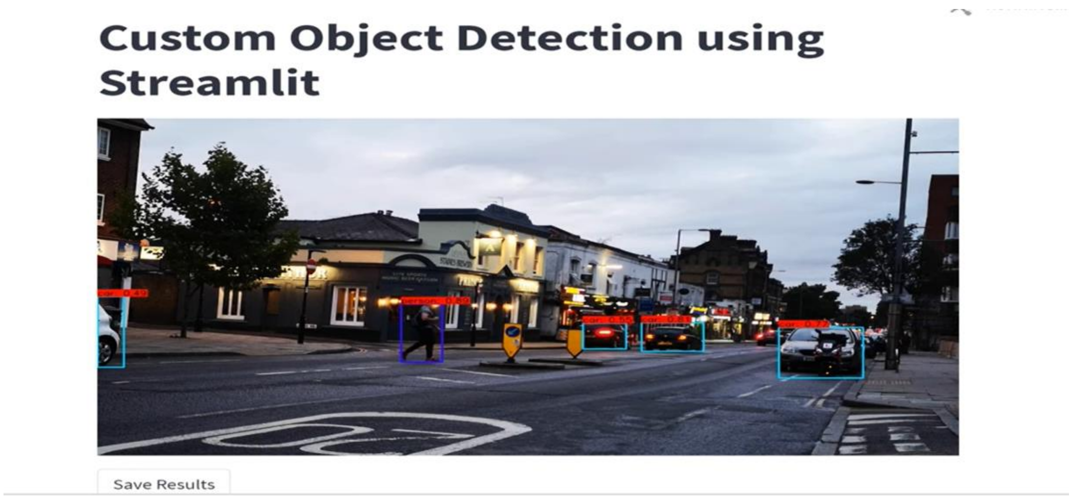


Fig 6.2 Output-2

6-CONCLUSION

In this project, an intelligent object detection system was developed using deep learning techniques and real-time video processing with YOLOv4. The system efficiently processes input video frames, detects objects using a pre-trained YOLOv4 model, and annotates them with bounding boxes and class labels. The integration of OpenCV for frame handling and detection visualization enables smooth and accurate object identification across various scenes.

This approach overcomes the limitations of traditional object detection models by providing high-speed inference, reduced computational requirements, and the ability to operate on custom-trained classes. The system is adaptable, scalable, and suitable for real-time applications such as surveillance, industrial automation, and smart monitoring. The experimental results demonstrate that the model performs reliably and efficiently, making it a practical solution for real-world object detection needs.

Future Scope

Integration of Advanced Object Detection Models

Incorporate state-of-the-art models like YOLOv8, EfficientDet, or transformer-based detectors (e.g., DETR) to enhance detection accuracy, speed, and generalization across varied environments.

Analytics and Alert System

Add a backend system that logs detections, generates statistical reports, and sends alerts or notifications when specific objects are detected (e.g., weapons, intruders, etc.).

REFERENCES

1.Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time bject Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, 779-788.

2.Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, 580-587.

3.Ren, S., He, K., & Sun, J. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Advances in Neural Information Processing Systems (NeurIPS), 2015.