IJIRIS: International Journal of Innovative Research in Information Security
Volume 11, Issue 02, April 2025
E-ISSN: 2349-7017
P-ISSN: 2349-7009
https://www.ijiris.com/archives

Real Time Human Object Detection Using Deep Learning

Naveen H
Vemana Institute of Technology,
VTU University, Bengaluru 560034, India
ChirraUdayasri, G Likhith Kumar Reddy, Harini R, Haripriya N
Department of CSE, Vemana Institute of Technology,
Bengaluru 560034, India
Chiiraudayasri_2021@vemanait.edu.in, glkr2003@gmail.com
harinir_2021@vemanait.edu.in, haripriyan_2021@vemanait.edu

Publication History:
Manuscript Reference No: IJIRIS/RS/Vol.11/Issue02/APIS10088
Research Article | Open Access | Double-Blind Peer-Reviewed | Article ID: IJIRIS/RS/Vol.11/Issue02/APIS10088
Received: 02,April 2025 Revised: 14, April 2025 Accepted: 25, April 2025 Published Online: 05, May 2025, Volume 2025 Article ID APIS10088 https://www.ijiris.com/volumes/Vol11/iss-02/09.APIS10088.pdf
Article Citation: Naveen,Chirraudayasri,Likhith,Harini,Haripriya(2025). Real Time Human Object Detection Using Deep Learning International Journal of Innovative Research in Information Security, Volume 11, Issue 02, Pages 117-125 **doi:>** https://doi.org/10.26562/ijiris.2025.v1102.09
BibTex key: Naveen@2025Real

Abstract: The advancement of technology has enabled machines to perform tasks traditionally reserved for humans. One such area is object and human detection, where systems can identify and track entities in real time. Thanks to technological advancements, machines can now carry out duties that were previously only performed by people. Object and human detection is one such field where systems are able to recognize and follow entities in real time. In order to differentiate between people and different things in video footage, this research makes use of computer vision and deep learning algorithms. By processing each frame quickly and accurately, we guarantee effective detection by utilizing strong techniques like YOLO (You Only Look Once). Applications in surveillance, self-driving cars, and interactive settings are made possible by the suggested system's strong ability to recognize numerous objects and people at once. Numerous contemporary applications, including as surveillance systems, driverless cars, and smart cities, depend on the capacity to automatically recognize and interpret visual features like people and objects. The development of technology has made it possible for machines to carry out jobs that were previously only possible for humans. Object and human detection is one such field where systems are able to recognize and follow entities in real time. In order to differentiate between people and different things in video footage, this research makes use of computer vision and deep learning algorithms. By processing each frame quickly and accurately, we guarantee effective detection by utilizing strong techniques like YOLO (You Only Look Once). Applications in surveillance, self-driving cars, and interactive settings are made possible by the suggested system's strong ability to recognize numerous objects and people at once. Numerous contemporary applications, including as surveillance systems, driverless cars, and smart cities, depend on the capacity to automatically recognize and interpret visual features like people and objects.

Keywords: Real-Time Object Detection, Human Detection, YOLO, Computer Vision, Deep Learning, Video Surveillance, Autonomous Systems

I. INTRODUCTION

Intelligent visual systems are now indispensable instruments in many fields, from automation to security, in the current digital era. The problem of object detection, or a machine's capacity to locate and identify things within an image or video stream, is at the heart of many such systems. Real-time, precise detection systems are more important than ever as human settings grow more data-driven and dynamic. Applications like autonomous navigation, crowd control, public safety, and smart homes heavily rely on human and object identification. By identifying nearby items and detecting human presence, machines may make well-informed decisions that improve interaction, safety, and efficiency. The goal of this research is to create such a detection system by utilizing YOLO (You Only Look Once, version 8), one of the most sophisticated deep learning models available. YOLO handles object detection as a regression problem and processes a whole image in a single neural network pass, in contrast to conventional techniques that mostly rely on manually created features and sequential region recommendations. This makes it perfect for real-time systems since it speeds up inference times without appreciably sacrificing detection accuracy. The system can effectively generalize in complicated, real-world contexts since the model is trained on huge datasets like COCO, which contains a wide range of common objects and human figures. YOLOv8's strength is its improved architecture, which includes enhancements to post-processing methods, prediction layers, and feature extraction. Higher precision is a result of these improvements, particularly under difficult circumstances such background clutter, occlusion, and fluctuating lighting. Incorporating such sophisticated vision skills can create new opportunities in sectors like robotics, healthcare, retail, and transportation, all of which are driven by the growing need for smarter and more autonomous systems.

The goal of this project is to investigate and put into practice a thorough detection system that can recognize both people and numerous objects in real-time video streams. The project adds to the expanding field of intelligent vision systems by fusing cutting-edge machine learning with useful deployment strategies. Additionally, it lays the groundwork for upcoming advancements in behavior prediction, facial analysis, and action recognition.

II. RELATED WORK

A fundamental task in computer vision, human and object recognition has implications in everything from robotics and healthcare to autonomous cars and surveillance. Conventional methods of object detection, like those based on Support Vector Machines (SVM) and Histogram of Oriented Gradients (HOG), performed mediocrely but faltered in situations with complex backdrops, occlusions, or changing lighting. Convolutional Neural Networks (CNNs), in particular, are deep learning-based solutions that emerged as a result of these restrictions and greatly increased detection resilience and accuracy. Human and object recognition is a basic job in computer vision that has applications in autonomous vehicles, robotics, healthcare, and surveillance. When faced with complex backdrops, occlusions, or shifting lighting, traditional object detection techniques, such as those based on Support Vector Machines (SVM) and Histogram of Oriented Gradients (HOG), performed mediocrely but failed. These limitations led to the development of deep learning-based solutions, such as Convolutional Neural Networks (CNNs), which significantly improved detection accuracy and robustness.

The use of YOLO for person's detection has been the subject of numerous research, particularly in real-world settings such as workplaces, public areas, and train stations. For example, YOLO has been used by academics to follow and identify pedestrians in real time in CCTV surveillance systems. Applications in social distancing enforcement, security surveillance, and crowd analysis depend on this. Additionally, to facilitate multi-object tracking with consistent identity assignment over time, YOLO has been coupled with tracking algorithms like Deep SORT. Numerous studies have examined the potential of YOLO for people detection, especially in real-world contexts like train stations, public spaces, and workplaces. For instance, researchers have employed YOLO in CCTV surveillance systems to track and identify pedestrians in real time. This is essential for applications in crowd analysis, security surveillance, and social distancing enforcement. Additionally, YOLO has been combined with tracking algorithms such as Deep SORT to provide multi-object tracking with consistent identity assignment over time. The potential of YOLO for people detection has been the subject of numerous researches, particularly in real-world settings such as train stations, public areas, and workplaces. For example, YOLO has been used by academics to follow and identify pedestrians in real time in CCTV surveillance systems. Applications in social distancing enforcement, security monitoring, and crowd analysis require this. In order to offer multi-object tracking with consistent identity assignment over time, YOLO has also been integrated with tracking algorithms like Deep SORT.

III. PROBLEM STATEMENT

The Problem statement is Surveillance and monitoring systems are a critical component in ensuring public safety, managing crowds, and maintaining security in both private and public spaces. Traditionally, these systems depend on human operators to continuously observe video feeds and respond to suspicious activities. This manual approach is not only time-consuming and labor-intensive but also highly prone to human oversight, especially in fast-moving or high-pressure environments. Intelligent, automated solutions are required because it is impractical to rely simply on human judgment for continuous observation due to the growing volume and complexity of urban surroundings. Despite the potential of contemporary deep learning-powered object identification algorithms, many of the solutions on the market are constrained by a trade-off between resource efficiency and performance. High-accuracy models are not appropriate for implementation in contexts with limited resources or budgets since they frequently call for costly, high-end gear. Lightweight models, on the other hand, might not offer the precision needed for crucial use cases like anomaly tracking, threat detection, or people counting in congested regions.

Therefore, a balanced strategy that provides real-time detection and respectable accuracy without depending on specialist computational infrastructure is obviously needed. The increasing volume and complexity of urban environments make it difficult to rely just on human judgment for continuous observation, necessitating the need for intelligent, automated solutions. Many of the available solutions are limited by a trade-off between performance and resource efficiency; despite the promise of modern deep learning-powered object identification algorithms. Because high-accuracy models sometimes require expensive, high-end equipment, they are not suitable for use in situations with limited funds or resources. However, for critical use cases like as threat detection, anomaly tracking, or population counts in crowded areas, lightweight models may not provide the accuracy required. Thus, it is evident that a well-rounded approach is required that offers real-time detection and reasonable accuracy without relying on specialized computational infrastructure. This includes the ability to operate through a web browser, support mobile-based cameras via IP, and deliver real-time results without requiring specialized software installations or configurations. This project seeks to resolve these pressing issues by developing a real-time people detection system utilizing the YOLO (You Only Look Once) deep learning model. YOLO is well-recognized for its high inference speed and competitive accuracy, making it ideal for applications where quick decision-making is essential. By designing the system to work seamlessly across laptop webcams and mobile cameras accessed via IP addresses, the project ensures adaptability and usability across diverse surveillance needs. Furthermore, integrating the model into a responsive web interface enables remote monitoring and makes the system easily deployable in various real-world scenarios, from personal security setups to large-scale event management

IV. METHODOLOGY

The methodology adopted for this project follows a comprehensive and structured approach, designed to ensure that the real-time object detection system meets all its primary objectives, such as performance, accuracy, flexibility, and ease of use. The stages in the methodology include system design and requirements gathering, selection of the object detection model, integration of the YOLO model, web interface development, testing and evaluation, and optimization for efficient performance.

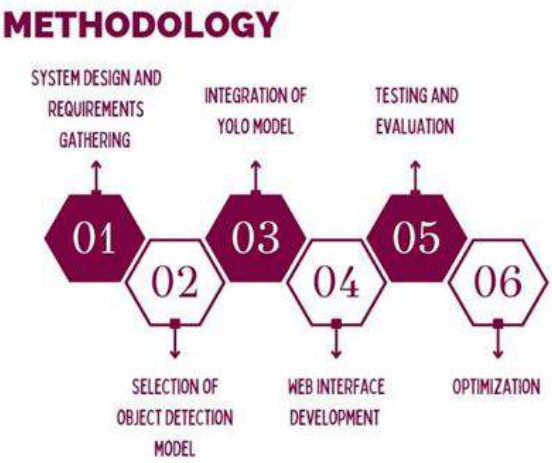

Fig1. Methodology Adopted for People Detection

A. System Design and Requirements Gathering :

The first step in the methodology is to thoroughly understand the requirements of the system. This includes identifying the essential features that the system must support, such as real-time performance, flexibility in input sources, and a user-friendly interface. A comprehensive review of existing object detection systems was conducted at this stage, which provided insights into the strengths and limitations of available technologies and helped guide the design choices for this project.

B. Selection of Object Detection Model

The next critical step in the methodology was the selection of the appropriate object detection model for real-time processing. Several state-of-the-art object detection models were evaluated, including R-CNN, SSD, and Faster R-CNN. These models were examined for their performance, accuracy, and suitability for real-time applications.

R-CNN (Region-based Convolutional Neural Networks) is one of the first successful deep learning models for object detection. However, it is computationally expensive and requires significant time to process each image, making it unsuitable for real-time detection.

SSD (Single Shot MultiBox Detector) improves upon R-CNN by performing object detection in a single pass and is faster. However, SSD still struggles with detecting small objects and faces limitations when compared to more advanced models in terms of accuracy and speed. Faster R-CNN, an improvement on R-CNN, addresses the issue of speed by using a Region Proposal Network (RPN) to speed up the detection process. Although faster than R-CNN, Faster R-CNN still requires a significant amount of computational resources and is not optimized for real-time applications.

C. Integration of YOLO Model

The integration process involved:

Loading the YOLO Model: This step involves loading the pre-trained weights into the YOLO model and preparing it for inference.

Video Stream Processing: Real-time video streams from webcams or IP cameras were captured using OpenCV. The video frames were then passed through the YOLO model for object detection, and bounding boxes were drawn around detected objects.

Object Detection: As the video stream was processed, the YOLO model identified various objects in each frame and classified them. This information was used to overlay bounding boxes and labels on the video feed, providing real-time visual feedback to the user.

Performance Optimizations: Various techniques were employed to ensure the model ran efficiently in real-time, including reducing the resolution of the input frames (if needed) and optimizing the inference pipeline.

D. Web Interface Development

The web interface was a crucial component of the system, designed to provide users with an easy way to view live video feeds, track detected objects, and interact with the system. The interface was created with a blend of Flask, JavaScript, CSS, and HTML.

HTML: Used to organize the interface's layout and create sections for the control buttons, video feed, and detection results. The interface is styled with CSS to make it aesthetically pleasing and user-friendly. To improve the user experience, custom styles were applied to the labels, detection boxes, and video feed.

JavaScript: Used to manage the interface's dynamic components. JavaScript handled user input (e.g., choosing camera sources, initiating/stopping detection), updated bounding boxes when objects were recognized, and updated the video feed in real-time.

Flask: Flask, a lightweight Python web framework, was used to serve the interface and handle requests. It was used to create routes for displaying the video feed, managing user interactions, and updating the user interface with real-time data.

E. Optimization

Optimization was a key focus to ensure that the system could run efficiently on consumer-grade hardware. This included reducing the input frame size, improving the inference pipeline, and optimizing the YOLO model itself. Techniques such as model quantization and pruning were considered to reduce the model's size and speed up processing times without sacrificing detection accuracy.

V. SYSTEM ARCHITECTURE

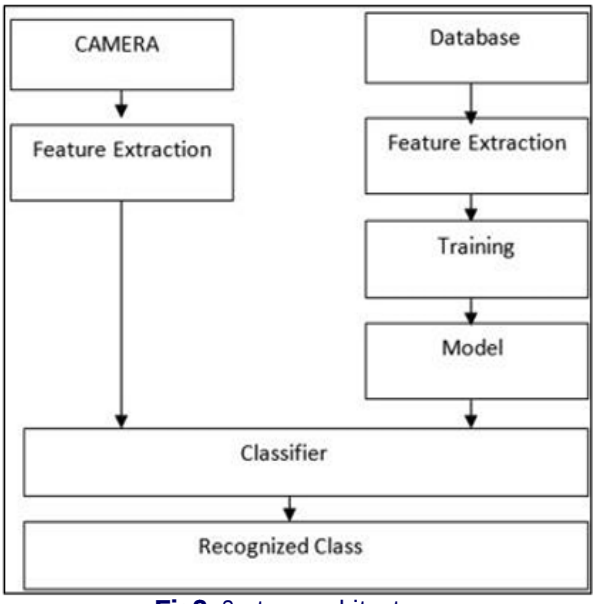

Fig2. System architecture

The proposed system is designed to detect humans and various objects in real-time using deep learning techniques, specifically leveraging the YOLO (You Only Look Once) object detection algorithm. A combination of HTML, CSS, JavaScript, and Flask was used to develop the interface.

HTML: Used to arrange the layout of the interface and make sections for the video stream, control buttons, and detection results.

CSS is used to style the interface, making it both visually appealing and easy to use. Custom styles were applied to the video feed, detection boxes, and labels to enhance the user experience.

JavaScript: Used to control the dynamic elements of the UI. Real-time video feed updates, bounding box updates when objects were detected, and user interaction (such as selecting camera sources and starting/stopping detection) were all managed by JavaScript.

Recognized Class: The final output, which is the identified class label (e.g., "Person A", "Object X", etc.).

The following steps outline the core working principle:

a) **Video Feed Acquisition**: The process begins with capturing video feeds from either a local webcam or an IP camera. The system can handle both sources, which allows it to be deployed in various environments. Using OpenCV, the video frames are extracted in real time and passed through the system for further processing

b) **Frame Preprocessing:** The captured video frame is first preprocessed to ensure that it is in a suitable format for object detection. This step includes resizing the frame to a fixed size that matches the YOLO model's input dimensions and normalizing the pixel values to match the model's training data. Preprocessing ensures that the frame is consistent with the expectations of the YOLO model, enabling it to detect objects effectively.

c) **Object Detection:** Once the frame is prepared, it is passed through the YOLO model. The model processes the frame in a single pass, dividing it into a grid of cells. Each cell is responsible for detecting objects within its region, and the model outputs a set of bounding boxes, class labels, and confidence scores for each object detected. In the context of this project, the YOLO model focuses on detecting people. It identifies regions in the frame that contain humans and marks these regions with bounding boxes, along with the confidence score and the class label "person."

d) **Post-Processing and Visualization:** After the detection phase, the bounding boxes and class labels are processed to filter out low-confidence detections. The system only retains bounding boxes that have a confidence score above a predefined threshold. These boxes are then drawn onto the frame using OpenCV, and the result is displayed to the user.

e) **Real-Time Interaction**: The user interacts with the system through a web interface that provides live updates. Using JavaScript and WebSockets, the interface communicates with the backend to receive the video stream and detection results in real time. The user can select camera sources, view the live stream, and watch the detection process unfold without page reloads, making the interaction seamless and engaging.

f) Performance Monitoring: Throughout the entire process, the system constantly monitors its performance to ensure real-time operation. The frame rate is tracked, and if the processing time per frame exceeds a certain threshold, optimizations are applied to improve efficiency. The detection process unfold without page reloads, making the interaction seamless and engaging

The overall architecture includes:

Input Module: Captures real-time video feed using a webcam or IP camera.

Processing Module: Runs on a computing device (e.g., laptop, Raspberry Pi, or NVIDIA Jetson) with a deep learning framework (PyTorch or TensorFlow) and the YOLO model.

Detection Engine: Applies the YOLO algorithm to detect and classify objects within each frame.

Output Interface: Displays annotated video with detected objects and can trigger alerts or store data for further analysis.

VI. IMPLEMENT STATERGY

In this study the implementation phase of this project involved integrating various components such as the video capture system, object detection model, web interface, and real-time communication mechanism to create a robust people detection system. The system was designed to detect individuals from a video stream, particularly focusing on monitoring spaces such as surveillance areas or public places.

ALGORITHMS:

1. REAL-TIME OBJECT DETECTION WITH YOLO

Step 1: Load Required Libraries:

Import OpenCV, the YOLO model, and any additional libraries (e.g., threading for text-to-speech).

Step 2: Initialize Components:

a. Load the YOLO model with the pre-trained weights.
b. Read the class names (e.g., COCO dataset classes) from a file.
c. Initialize video capture for a camera or video feed.

Step 3: Pre-Processing:

a. Resize the video frames to the required size for YOLO inference.
b. Pass the frames to the YOLO model for object detection.

Step 4: Run Detection:

1. Extract the following from YOLO's results:
   a. Bounding boxes for detected objects.
   b. Class IDs and names of detected objects.
   c. confidence scores for each detection.
2. Perform tracking or assign unique IDs to detected objects.

Step 5: Post-Processing:

a. Draw bounding boxes and labels on the frame.
b. Announce detected objects using text-to-speech (if needed).
c. Maintain a record of already announced objects to avoid repetition.

Step 6: Output Results:

a. Display the processed frame with the bounding boxes and object labels in a GUI window.
b. Provide an option to stop the detection by checking for user input (e.g., pressing 'q').

Step 7: Clean Up:

Release video resources and destroy any GUI windows after the detection loop is stopped.

Examining Typical Object Detection Metrics: A Brief Guide to Terminology

a. A model must initially be trained on a representative and varied dataset in order to recognize items in the image. It needs to learn to identify different items and how they relate to one another in space. Expert image annotation services may be useful in this situation. It's time to assess the model's performance after training. The following are a few of the primary metrics for object detection algorithms:

b. Precision and Recall: Recall highlights the model's capacity to locate all ground truth bounding boxes, whereas precision concentrates on precisely identifying pertinent objects. Precision and recall work together to balance the quantity and quality of predictions.

c. The core metric for object detection is Average Precision (AP), which combines precision, recall, and the model's confidence in each detection. The Precision x Recall curve is reduced to a single numerical summary using average precision object detection, which is computed independently for every class.

d. Mean Average Precision (mAP): Specifically in multi-class scenarios, Mean Average Precision (mAP) expands upon the concept of AP. The AP is averaged across all classes to calculate it. For different IoU thresholds and object classes, the measure takes precision and recall into account; a higher mAP denotes better overall model performance.

e. Video Feed Acquisition: The process begins with capturing video feeds from either a local webcam or an IP camera. The system can handle both sources, which allows it to be deployed in various environments. Using OpenCV, the video frames are extracted in real time and passed through the system for further processing

Result	Discussion
Real-time Object Detection	The YOLO model successfully detects objects with minimal latency, demonstrating the efficiency of the system in real-time applications.
High Detection Accuracy	The system achieves high accuracy in detecting objects, validating the use of YOLO for object recognition in dynamic environments.
Scalability Across Different Inputs (Webcam, IP Camera)	The system adapts well to different video inputs, proving its flexibility in various monitoring scenarios such as surveillance and crowd management.
Bounding Boxes and Labeling of Detected Objects	Detected objects are accurately labeled with bounding boxes, making the system user-friendly and visually informative for real-time monitoring.
Minimal Latency	The system processes video streams with minimal delay, highlighting its suitability for time-sensitive applications like security and automation.
Efficient Use of YOLO Model	Using a lightweight version of YOLO ensures that the system runs efficiently on standard hardware, balancing performance and accuracy.
User-Friendly Web Interface	The integration of a responsive web interface makes it accessible on multiple devices, enhancing user interaction with the system.
Integration with Mobile Camera (IP Integration)	The ability to integrate with mobile cameras via IP addresses expands the system's use cases, enabling remote monitoring capabilities.

Table 5.1. Refers the confusion matrix

VII. EXPERIMENTAL RESULTS

An extensive analysis of the outcomes of the YOLO-based object identification systems implementation is given in the Results section. The project's main objective was to create a reliable, real-time object detection system that could be applied to a range of situations, such as retail analytics, industrial monitoring, and security surveillance. The YOLO (You Only Look Once) algorithm, which is well-known for its quickness and precision in identifying and categorizing objects in photos and video streams, is used by the system. This part will examine the system's performance in further detail, analyze the results in a variety of use case situations, assess the performance indicators, and talk about the findings.

Fig 3. Home page of the Real time human and object detection

Fig4. Home page of the Real time human and object detection

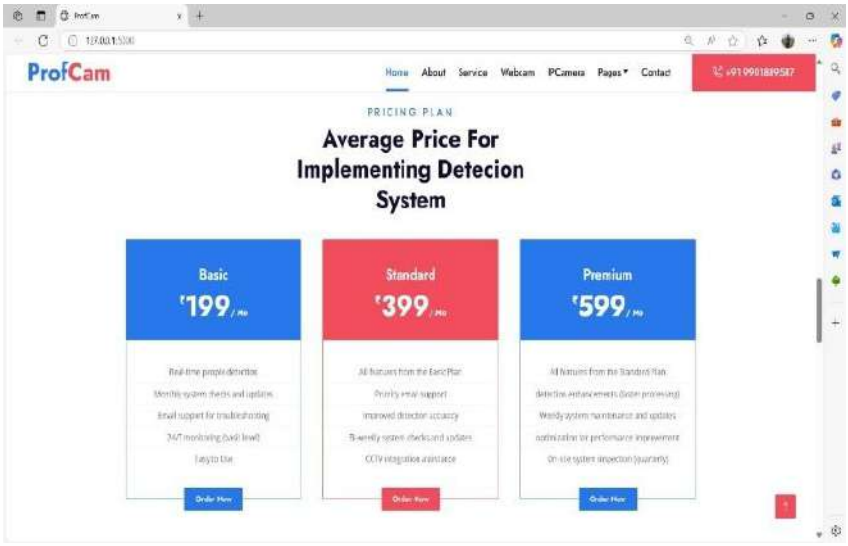

Fig 5. Avereage price for implementing Real time human and object detection

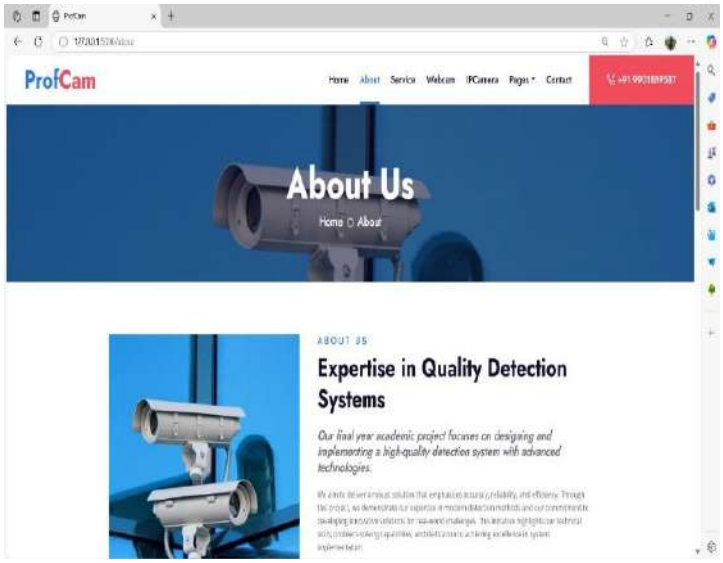

Fig6 About Us Page of the Real time human and object detection

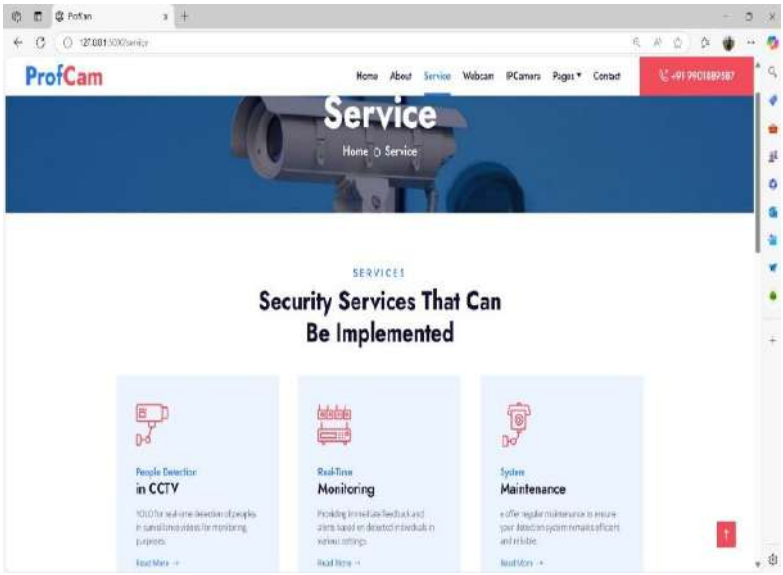

Fig7 Services age of the Real time human and object detection

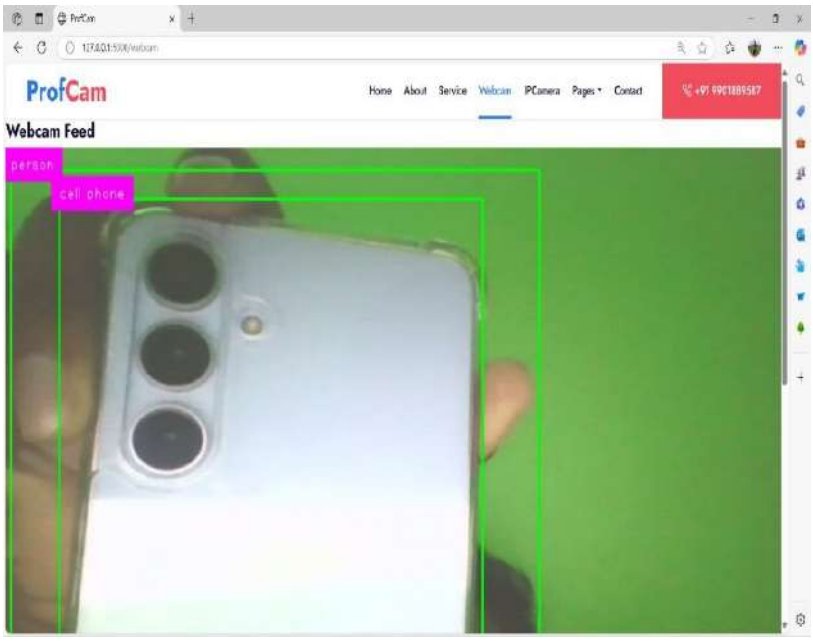

Fig 8 Detecting Cell phone using webcam.

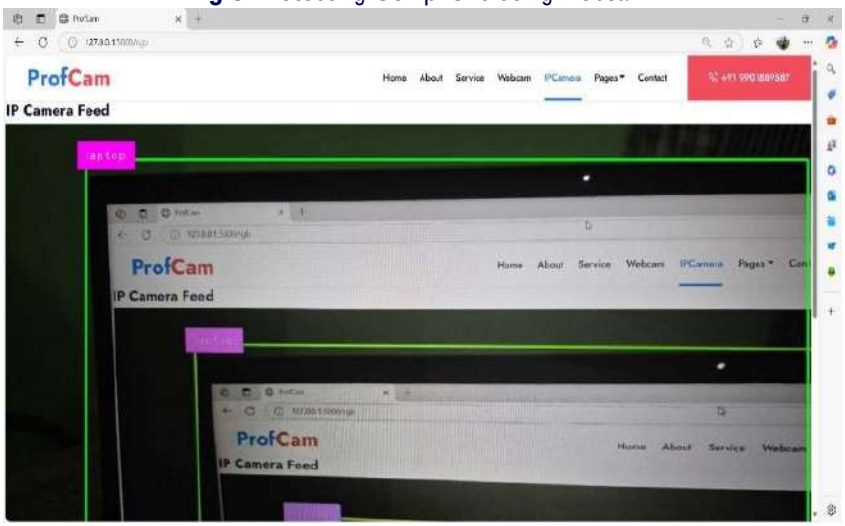

Fig9 Detecting Laptop using webcam.

These are the experimental results in the project Real time human and object detection using DI

VIII. APPLICATIONS

A. Security and Surveillance: spotting illegal people in limited locations or keeping an eye on public areas to ensure their safety.

B. Traffic Monitoring: Recognizing cars, pedestrians, and other objects in order to assess traffic patterns and find infractions.

C. Industrial automation: keeping an eye on production lines to spot irregularities or guarantee worker security.

D. Crowd management: Monitoring people's movements in sizable crowds to avoid crowding or guarantee a smooth flow.

E. Smart Home Systems: By identifying persons or items in real time, these systems improve automation and security in home settings.

F. Analytics for Retail and Customers: Tracking foot traffic in stores to learn about consumer behavior and improve layouts.

IX. CONCLUSION

It has been a fulfilling experience to design a real-time people detection system that combines web-based interface, object detection, and video stream processing using YOLO (You Only Look Once). Using the YOLO model for its speed and accuracy, the study sought to identify people in live video broadcasts. And low latency, even in challenging conditions like varying lighting and complex backgrounds. The system was implemented as a Flask web application, offering users an intuitive interface to view live detection results from webcams or IP cameras. Optimizations such as quantization and pruning enhanced performance, while testing demonstrated robustness across diverse environments. This scalable and efficient system not only meets real-time monitoring needs but also offers adaptability for detecting other objects. The project underscores the potential of deep learning in surveillance, safety, and human-computer interaction, paving the way for future enhancements.

REFERENCES

1. T.Ahmad, Y. Ma, M. Yahya, B. Ahmad, S. Nazir, and A. U. Haq, ‘‘Object detection through modified YOLO neural network,’’ Sci. Pro gram., vol. 2020, pp. 1–10, Jun. 2020.
2. J.Yan, Z. Lei, L. Wen, and S. Z. Li, ‘‘The fastest deformable part model for object detection,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Columbus, OH, USA, Jun. 2014, pp. 2497–2504.
3. P.ViolaandM.J.Jones, ‘‘Robust real-time face detection,’’ Int. J. Comput. Vis., vol. 57, no. 2, pp. 137–154, May 2004.
4. C.J.Du, H. J. He, and D. W. Sun, ‘‘Object classification methods,’’ in Proc. Int. Comput. Vis. Technol. Food Quality Eval., Dublin, Ireland, 2016, pp. 87–110.
5. E.Wang, Y. Li, Z. Nie, J. Yu, Z. Liang, X. Zhang, and S. Yiu, ‘‘Deep fusion feature based object detection method for high resolution optical remote sensing images,’’ Appl. Sci., vol. 9, no. 6, p. 1130, Mar. 2019.
6. X.Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, ‘‘3D object proposals for accurate object class detection,’’ in Proc. Adv. NeuralInf. Process. Syst., C. Cortes, N. D. Lawrence, D. D.Lee, M. Sugiyama, R. Garnett, Eds. New York, NY, USA: Curran Associates, 2015, pp. 424–432.
7. H.Bilen and A. Vedaldi, ‘‘Weakly supervised deep detection networks,’’ in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Las Vegas, NV, USA, Jun. 2016, pp. 2846–2854.
8. S.Ren, K.He,R.Girshick, and J. Sun, ‘‘Faster R-CNN: Towards real-time object detection with region proposal networks,’’ in Proc. Adv. Neural Inf. Process. Syst., 2015,
9. X.Chen,S.Xiang,C.-L.Liu,andC.-H.Pan,‘‘Vehicle detection in satellite images by hybrid deep convolutional neural networks,’’ IEEE Geosci. Remote Sens. Lett., vol. 11, no. 10, pp. 1797–1801, Oct. 2014.
10. A.Mukhtar, M. J. Cree, J. B. Scott, and L. Streeter, ‘‘Mobility aids detection using convolution neural network (CNN),’’ in Proc. Int. Conf. Image Vis. Comput. New Zealand (IVCNZ), Auckland, New Zealand, Nov. 2018, pp. 1–5.
11. A.Vasquez, M. Kollmitz, A. Eitel, and W. Burgard, ‘‘Deep detection of people and their mobility aids for a hospital robot,’’ in Proc. Eur. Conf. Mobile Robots (ECMR), Paris, France, Sep. 2017, pp. 1–7.
12. M.Kollmitz, A. Eitel, A. Vasquez, and W. Burgard, ‘‘Deep 3D perception of people and their mobility aids,’’ Robot. Auto. Syst., vol. 114, pp. 29–40, Apr. 2019.
13. M.Alruwaili, M. H. Siddiqi, A. Khan, M. Azad, A. Khan, and S. Alanazi, ‘‘RTF-RCNN: An architecture for real-time tomato plant leaf diseases detection in video streaming using faster-RCNN,’’ Bioengineering, vol. 9, no. 10, p. 565, Oct. 2022.
14. H.Law and J. Deng, ‘‘CornerNet: Detecting objects as paired keypoints,’’ in Proc. Eur. Conf. Comput. Vis. (ECCV), Munich, Germany, 2018, pp. 734–750.
15. Y.Liu,P.Sun, N. Wergeles, and Y. Shang, ‘‘A survey and performance evaluation of deep learning methods for small object detection,’’ Expert Syst. Appl., vol. 172, Jun. 2021, Art. no. 114602.

