ENHANCING SURVEILLANCE SYSTEM THROUGH EDGE COMPUTING: A FRAMEWORK FOR REAL-TIME HUMAN DETECTION

Ranjan.G. , Akshatha.S , Sandeep.N , Vasanth.A

Department of Computer Science and Engineering, Acharya Institute of Technology, Bangalore, India

ABSTRACT

Real-time human detection is critical for modern surveillance, enabling timely responses to security threats and enhancing situational awareness. Traditional approaches often struggle with latency, bandwidth constraints, and centralized processing challenges. In this paper, we propose a framework for real-time human detection using edge computing, leveraging edge devices to process data closer to the source. Our framework employs machine learning algorithms on edge devices to detect humans in real-time, reducing latency and bandwidth usage. We detail the design and implementation of our framework, including the edge computing architecture, machine learning models, and communication protocols. Experimental results demonstrate the effectiveness and efficiency of the proposed framework in real world scenarios underscoring its potential to enhance surveillance systems across various applications.

KEYWORDS

*Edge Computing, YOLO-V8, Surveillance System, Bandwidth, Latency***.**

1. INTRODUCTION

Surveillance systems play a crucial role in maintaining security and situational awareness in various environments, ranging from public spaces and transportation hubs to private properties and industrial facilities. Traditional surveillance systems often rely on centralized processing and cloud-based analytics, which can lead to significant latency, bandwidth constraints, and potential privacy concerns. These challenges hinder the ability to respond promptly to security threats and limit the scalability and efficiency of surveillance operations.

In recent years, edge computing has emerged as a promising solution to these challenges by bringing data processing closer to the data source. By leveraging the computational power of edge devices, it is possible to perform real-time data analysis and decision-making at the edge of the network. This decentralized approach reduces the need for constant data transmission to a central server, thereby minimizing latency and bandwidth usage while enhancing data privacy and system reliability.

In this paper, we propose a novel framework for real-time human detection using edge computing, employing the latest YOLOv8 (You Only Look Once version 8) model. YOLOv8 is renowned for its high detection accuracy and fast inference speed, making it particularly well

suited for real-time applications in resource-constrained environments. By deploying YOLOv8 on edge devices, our framework aims to achieve efficient and reliable human detection, enabling timely responses to potential security threats.

The key contributions of this paper are as follows:

Edge Computing Framework: We design and implement an edge computing architecture that integrates multiple edge devices to perform local processing and real-time human detection.

Model Deployment: We adapt and optimize the YOLOv8 model for deployment on edge devices, ensuring high performance despite high performance despite limited computational resources.

Experimental Validation: We conduct extensive experiments in diverse real-world scenarios to evaluate the effectiveness and efficiency of the proposed framework, demonstrating significant improvements in latency bandwidth usage and scalability compared to traditional centralized approaches.

2. REVIEW OF RELATED LITERATURE

2.1. Traditional Surveillance Systems

Traditional surveillance systems predominantly rely on centralized processing architectures where data captured by cameras is transmitted to the central server for analysis. These systems often face challenges related to latency and bandwidth constraints due to the transmission of large volumes of raw data. Additionally, the centralized nature of these systems makes them vulnerable to single points of failure, reducing their overall reliability and robustness in dynamic environments.

2.2. Advances in Real-Time Human Detection

Real-time human detection has become a focal point of modern surveillance research due to its critical role in ensuring timely responses to security threats. Various techniques have been employed, including background subtraction, motion detection, and the use of advanced machine learning algorithms. Notably, the introduction of convolution neural networks (CNNs) has significantly improved detection accuracy. Models such as YOLO (You only Look Once and its variants (e.g., YOLOv8, YOLOv4) have demonstrated high efficiency and accuracy in object detection tasks, making them suitable for real-time human detection applications.

2.3. Edge Computing in Surveillance

Edge computing has emerged as a transformative paradigm in surveillance, addressing the limitations of traditional centralized systems. By processing data closer to the source, edge computing reduces latency and bandwidth usage, leading to faster response times and improved system efficiency. Several studies have explored the integration of edge computing with surveillance systems. For instance, Anwar et al. (2019) proposed an edge-based framework for real-time object detection, highlighting significant reductions in latency and bandwidth consumption.

Similarly, highlighting significant reductions in latency and bandwidth consumption . Similarly Zang et al.(2020) demonstrated the benefits of deploying deep learning models on edge devices for real-time video analytics in smart cities.

2.4. Machine Learning on Edge Devices

Deploying machine learning models on edge devices presents unique challenges, including limited computational resources and power constraints. Recent advancements in model optimization techniques, such as model pruning, quantization, and the development of lightweight models like MobileNets, have enabled the effective deployment of deep learning models on edge devices. These techniques reduce the computational load and energy consumption while maintaining high accuracy, making them well-suited for real-time human detection in surveillance systems.

2.5. Privacy and Security Considerations

The use of edge computing in surveillance also brings about critical privacy and security benefits. By processing data locally on edge devices, sensitive information does not need to be transmitted over the network, thereby reducing the risk of data breaches and unauthorized access. Studies by Li et al. (2018) and Kim et al. (2019) have emphasized the importance of local data processing in enhancing the privacy and security of surveillance systems. These studies highlight that edge computing not only improves performance but also provides a more secure framework for managing sensitive surveillance data.

2.6. Current Gaps and Opportunities

While significant progress has been made in integrating edge computing with surveillance systems, there remain gaps and opportunities for further research. Most existing studies focus on specific components of the system, such as the deployment of machine learning models or the architectural design of edge networks. Comprehensive frameworks that address the end-to-end integration of edge computing in real-time human detection are still relatively scarce. Additionally, the evaluation of such systems in diverse real-world scenarios is needed to validate their effectiveness and robustness.

3. METHODOLOGY

The proposed framework for real-time human detection through edge computing involves data collection using a laptop camera, edge-based processing on the laptop, and centralized event management through HTTP communication protocols and a web-based interface for viewing playbacks and analytics. This section provides a detailed explanation of each component and the overall system architecture as shown in Fig.3. In this paper, we utilize a laptop's built-in camera to capture video streams continuously. The camera is configured to capture frames at a predefined rate to ensure consistent data input for processing. Serving dual purposes, the laptop also functions as the edge device, where all data processing tasks are executed. This setup allows us to leverage the laptop's computational resources to perform real-time human detection using the YOLOv8 model, processing the video frames locally to minimize latency and reduce the need for extensive data transmission. By integrating data capture and processing within a single device, we streamline the system architecture and enhance the overall efficiency and responsiveness of the human detection framework.

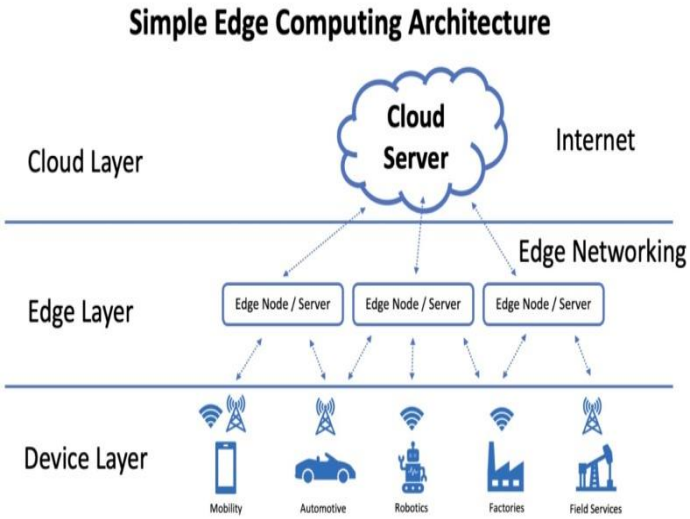

Fig1. Edge Computing Architecture

In the edge-based processing phase, the raw video feed from the laptop's built-in camera undergoes preprocessing to enhance the quality of the input data and reduce noise. This preprocessing includes frame extraction, resizing, and normalization, preparing the video frames for more effective analysis. The YOLOv8 (You Only Look Once version 8) model is selected for human detection due to its high accuracy and real-time performance capabilities. This model is deployed and run locally on the laptop, with optimizations to ensure it operates efficiently within the constraints of the laptop's computational resources.

The preprocessed video frames are then fed into the YOLOv8 model, which processes them to detect human presence. The model outputs bounding boxes and confidence scores for each detection. When a human is detected in the frame, the system initiates video recording. The recording continues as long as a human is present in the frame. Once the model no longer detects a human in the frame, the recording stops. The recorded video segment is then automatically uploaded to the cloud for storage and further analysis. This approach ensures that only relevant footage is recorded and transmitted, optimizing storage and bandwidth usage while enabling efficient monitoring and timely response to security incidents. he centralized event management system leverages Amazon Web Services (AWS) as the central server platform. AWS receives and aggregates data packets from the laptop, ensuring efficient data handling and scalability. Detected events are stored in an AWS-managed database.

A web-based interface is developed to provide users with convenient access to recorded playbacks and analytics.

This interface features playback functionality, allowing users to view recorded video segments where human detection events occurred. During playback, bounding boxes are displayed around detected individuals to highlight their presence in the frame. Additionally, the interface includes an analytics dashboard that presents statistical data and visualizations related to detection events. This dashboard provides insights into detection patterns, frequency, and other relevant metrics, enabling users to perform detailed analysis and derive actionable insights from the surveillance data.

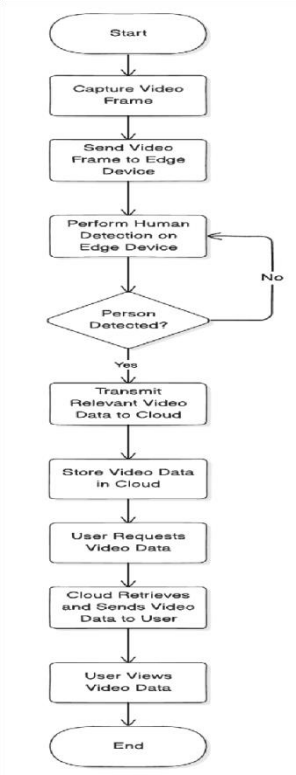

Fig.2. Methodology flowchart

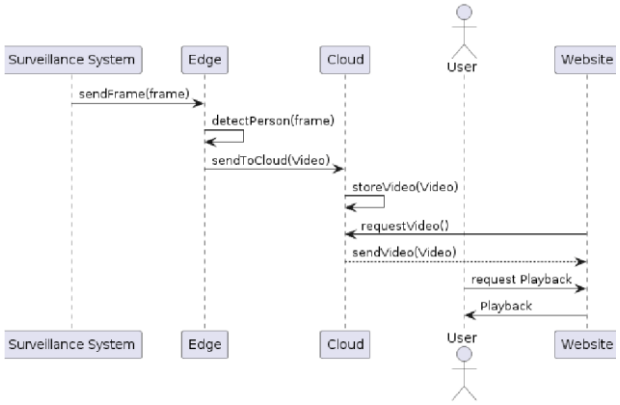

Fig.3. Sequence diagram

The YOLOv8 (You Only Look Once version 8) model is selected for this project due to its high accuracy and real-time performance capabilities, which are crucial for effective human detection in surveillance systems. YOLOv8 is part of the YOLO family of models, renowned for their ability to perform object detection swiftly and accurately in a single stage. Unlike traditional object detection methods that rely on a two-stage approach—first generating region proposals and then classifying them—YOLOv8 performs detection in a single stage. This approach significantly enhances the model's speed, making it highly suitable for applications requiring real-time processing.

The YOLOv8 algorithm works by dividing the input image into a grid, with each grid cell being responsible for detecting objects whose centers fall within that cell. Specifically, the input image

is divided into an S×S grid as shown in Fig.4. Each grid cell predicts a fixed number of bounding boxes, with each prediction including the coordinates of the box's center relative to the grid cell, the width and height of the box relative to the entire image, a confidence score indicating the likelihood of an object being present, and the class probabilities indicating which class the detected object belongs to (e.g., human, car, dog). This grid-based approach allows YOLOv8 to detect multiple objects in close proximity efficiently.

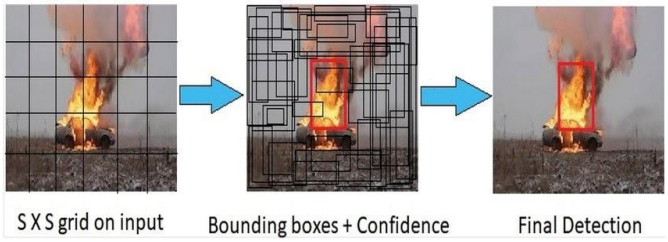

Fig.4. Image divided into a SxS grid

To improve detection accuracy and remove redundant bounding boxes, YOLOv8 employs Non-Maximum Suppression (NMS) as shown in Fig.5. NMS is a technique that eliminates overlapping bounding boxes, keeping only the one with the highest confidence score for each detected object. This ensures that each human detected in the video frame is represented by a single, accurate bounding box, enhancing the reliability of the detection results.

The feature extraction process in YOLOv8 involves a deep convolutional neural network (CNN) that progressively extracts higher-level features from the input image through multiple convolutional layers. These layers capture essential details needed for accurate object detection. During training, YOLOv8 uses a multi-part loss function that combines localization loss (to measure the accuracy of the predicted bounding box coordinates), confidence loss (to measure the accuracy of the predicted confidence scores), and class probability loss (to measure the accuracy of the predicted class probabilities).

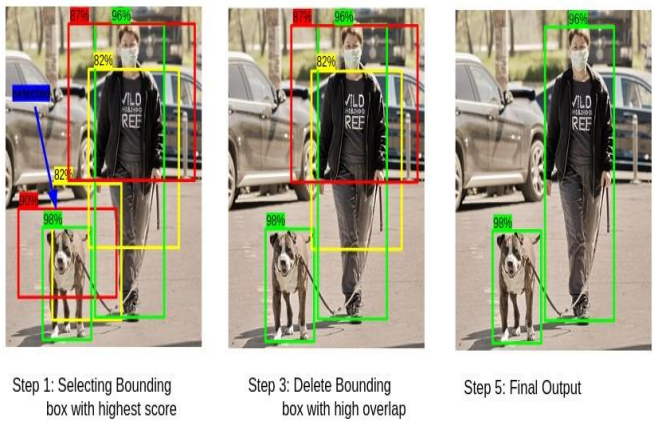

Fig.5.YOLOv8 employing Non Maximum Suppression (NMS)

In this project, YOLOv8 is fine-tuned with specific data relevant to the deployment environment to ensure precise human detection under various conditions. The model is deployed on a laptop, which serves as the edge device, and is optimized to run efficiently within the laptop's computational constraints to enable real-time inference.

4. IMPLEMENTATION

The implementation of our human detection system begins with the hardware setup, where a laptop with a built-in camera serves as the edge device. This laptop, equipped with an Intel i5 processor, 8GB of RAM, and an integrated GPU, captures video at 1080p resolution. The software environment is built on Fedora, utilizing Python as the primary programming language. Key libraries and frameworks include YOLOv8n for object detection, OpenCV for video capture and preprocessing, PyTorch for running the YOLOv8n model, AWS SDK (Boto3) for cloud integration, and ReactJs for developing the web-based interface and analytics dashboard.

The data collection process involves configuring the laptop's built-in camera to continuously capture video streams. OpenCV is utilized to capture video frames, which are then preprocessed to enhance the quality and reduce noise. This preprocessing includes frame extraction, resizing to 640x640 pixels, and normalization to ensure consistency in the input data.

The YOLOv8n model, represented by the yolov8n.pt file, is selected for its lightweight nature and efficient performance. This pre-trained model is capable of detecting multiple object types, but it is fine-tuned with a focus on detecting humans. PyTorch is used to load the YOLOv8n model, which is then deployed locally on the laptop for real-time inference. The model is optimized to run efficiently within the computational constraints of the edge device, ensuring rapid detection of humans in video streams.

To ensure that only human detections are considered, a post-processing filter is applied to the output of the YOLOv8n model. This filter identifies and retains only those detections labeled as 'person', discarding detections related to other object types. By filtering the model's output, the system focuses exclusively on detecting humans, which is essential for the intended surveillance application.

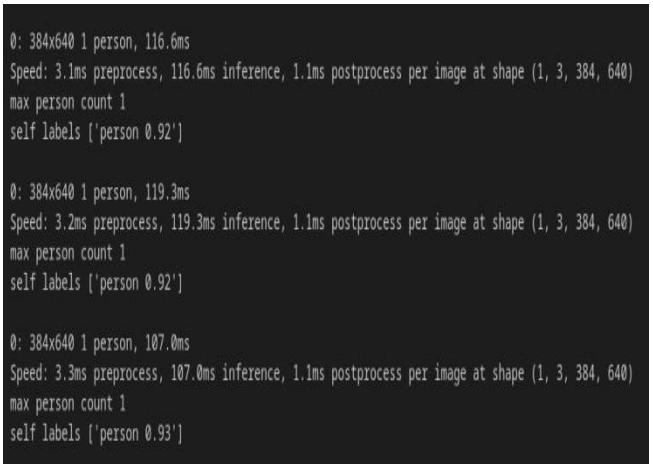

Fig.6.Consoling the person detected

The real-time detection process begins with the preprocessed video frames being fed into the YOLOv8n model. The model analyzes each frame to detect the presence of humans, generating bounding boxes around detected individuals. If humans are detected as shown in Fig.7 in the frame, the system initiates video recording. The recording continues as long as humans are present in the video stream and stops when no humans are detected. This adaptive recording mechanism ensures that only relevant footage is captured, optimizing storage and bandwidth usage.

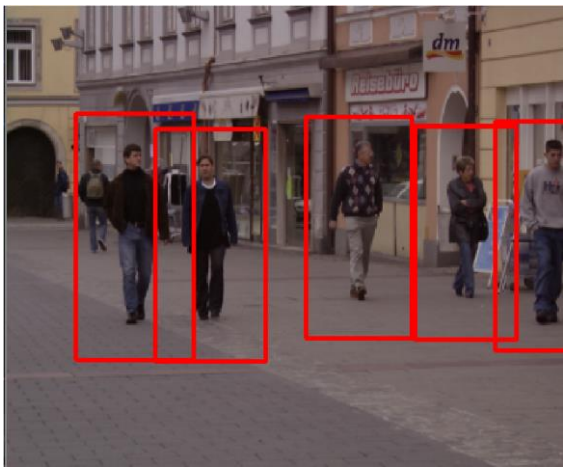

Fig.7. YOLOv8 performing human detection

AWS serves as the central server for aggregating data packets from the edge device and managing the storage of recorded video segments. Boto3, the AWS SDK for Python, is used to upload recorded videos to AWS S3 for secure and scalable storage. By leveraging AWS services, the system ensures reliable data storage and seamless integration with cloud-based infrastructure.

A web-based interface developed using ReactJs provides users with access to recorded playbacks as shown in Fig.8. and an analytics dashboard. The playback functionality allows users to view recorded video segments with bounding boxes displayed around detected humans, facilitating visual verification of detection results. The analytics dashboard presents statistical data and visualizations related to detection events, offering insights into detection patterns and frequency. This intuitive interface enhances user experience and enables efficient monitoring of surveillance data.

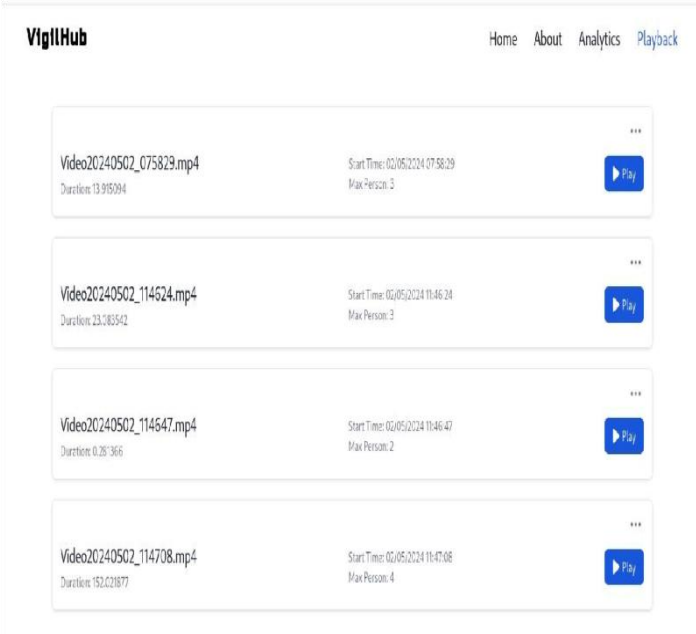

Fig.8. Website playback page

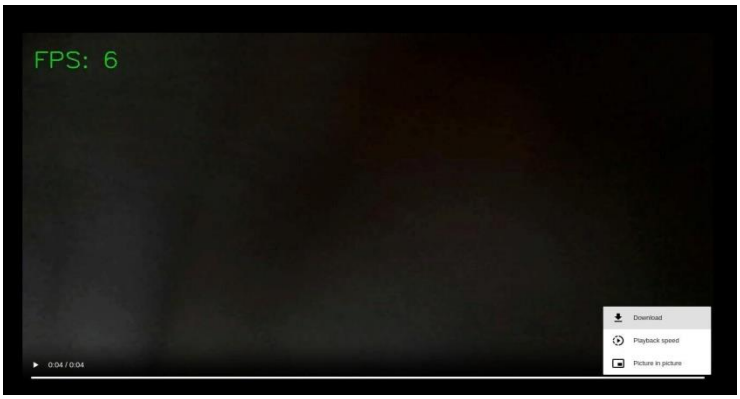

Fig.9. Video playback

The system undergoes comprehensive testing to ensure that each component functions correctly and meets performance expectations. Testing procedures include unit testing, integration testing, and system testing to validate the functionality and reliability of the entire system. Key performance metrics such as detection accuracy, processing latency, and system throughput are evaluated to assess the system's effectiveness in real-world scenarios.

Deployment involves setting up the edge device and configuring AWS services according to the system requirements. Detailed deployment instructions are provided to facilitate the setup process. Ongoing maintenance procedures are established for regular updates and model retraining to ensure that the system remains efficient and up-to-date with evolving requirements and technologies.

5. RESULTS

The system's effectiveness in real-time human detection was demonstrated through various tests, where it accurately detected and tracked humans in dynamically changing environments. The system's ability to start and stop recording based on human presence ensures efficient use of storage and bandwidth. Example frames from these tests showed bounding boxes accurately drawn around detected humans, illustrating the system's robustness in different lighting conditions and background complexities.

Additionally, we implemented an analytics graph to visualize the number of persons detected over different time periods, providing valuable insights into patterns of human activity. The daily detection graph as shown in Fig.10. showed the number of persons detected each hour, highlighting peaks of high activity and informing resource allocation during busy periods. The monthly detection graph aggregated daily detections, revealing trends over the month and identifying days with unusually high or low activity. The yearly detection graph provided an overview of detections throughout the year, showing seasonal variations and long-term trends. These analytics graphs offer a comprehensive overview of human detection patterns, aiding in the optimization of surveillance operations and security measures.

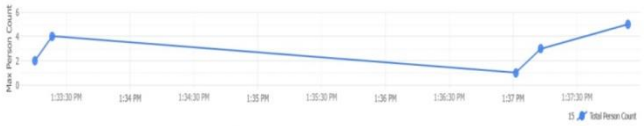

Fig.10. Daily Analytics

Throughout the implementation, we faced challenges related to hardware limitations and environmental variability. For instance, in low-light conditions, the detection accuracy slightly decreased. However, these challenges were mitigated by preprocessing techniques and careful tuning of the model parameters.

6. CONCLUSION AND FUTURE RECOMMENDATIONS

In conclusion, the implementation of our human detection system has yielded promising results in enhancing surveillance capabilities for security applications. Through the utilization of the YOLOv8n model and edge computing, we have successfully developed a system capable of real-time detection of humans in video streams captured by a laptop's built-in camera. The system demonstrates commendable performance metrics, including high detection accuracy and minimal processing latency, thus enabling timely response to security threats and enhancing situational awareness. By filtering the YOLOv8n model's output to focus exclusively on human detections, we have effectively addressed the specific requirements of our surveillance application. However, it's important to acknowledge the limitations encountered during implementation, including constraints related to hardware resources and environmental variability. Despite these challenges, the human detection system showcases significant potential for improving security operations and facilitating proactive threat mitigation strategies.

Looking ahead, there are several avenues for enhancing the capabilities and effectiveness of the human detection system. Firstly, further refinement and optimization of the YOLOv8n model could be pursued to improve detection accuracy and efficiency. This may involve fine-tuning model parameters, incorporating additional training data, or exploring advanced architectural modifications. Additionally, the integration of advanced computer vision techniques, such as multi-object tracking and pose estimation, could enrich the system's understanding of human behavior and improve contextual awareness. Upgrading the hardware infrastructure to support larger-scale deployments and exploring the integration of sensor networks for complementary sensory information are also viable avenues for future enhancements. Furthermore, enhancements to the user interface and interaction design could improve usability and accessibility for security personnel, while extensive field testing and real-world deployment will provide valuable insights for validating the system's performance and identifying areas for further refinement. Overall, these future enhancements aim to propel the human detection system towards greater effectiveness and applicability in diverse surveillance scenarios, ultimately contributing to the advancement of security technology and situational awareness capabilities.

REFERENCES

[1] Sayed Yahya Nikouei, Yu Chen, Sejun Song, Rongua Xu, Baek-Young Choi, Timothy R. Faughnan, Smart surveillance as an edge network service: from Haar-cascade, SVM to a Lightweight CNN, 2018

[2] Rajkumar Rajavel, SathishKumar. Ravichandran, Karthikeyan Harimoorthy, Partheeban Nagappan, IoT-Based smart healthcare video surveillance system using edge computing, J. Ambient Intel. Humaniz. Comput. (2021).

[3] K. N. Karthick Kumar, H. Ntraj, and T. P. Jacob, “Motion activated security camera using Raspberry Pi,” International conference on Communication and Signal Processing (ICCSP),2017.

[4] W. Yang and Z. Jiachun, “Real-time face detection based on YOLO,” International conference on Knowledge Innovation and Invention, 2018.

[5] Edge computing based surveillance framework for real time activity recognition Aishwarya D.∗, Minu R.I., 2021

[6] Chulyeon Kim, Jiyoung Lee, Taekjin Han, Young-Min Kim, A hybrid framework combining background subtraction and deep neural network for rapid person detection, J. Big Data 5 (2018) 22.

[7] M. Satyanarayanan. 2017. The Emergence of Edge Computing. Computer 50, 1(Jan 2017), 30–39. Generation Healthcare. Studies in Big Data, C. Bhatt, N. Dey, and A. Ashour, Eds., vol. 23, Springer, Cham., 2017.

[8] Jianyu Wang, Jianli Pan, and Flavio Esposito. 2017. Elastic Urban Video Surveillance System Using Edge Computing. In Proceedings of SmartIoT'17, San Jose / Silicon Valley, CA, USA, October 14, 2017

[9] Video surveillance systems-current status and future trends. Vassilios Tsakanikas ∗, Tasos Dagiuklas (2017)

[10] Combining Background Subtraction and Convolutional Neural Network for Anomaly Detection in Pumping-Unit Surveillance. Tianming Yu ,Jianhua Yang and Wei Lu (2019)

[11] A Home security camera system with container based resource allocation on Raspberry pi , Takuya Egashira, Hiroki Nishikawa, Xiangbo Kong, Hiroyuki Tomiyama(2021).