Research

Edge-based AI solution for enhancing urban safety: helmet compliance monitoring with YOLOv9 on Raspberry Pi

Nikunj Tahilramani[1] · Param Ahir[2] · Shruti Saxena[2] · Vandana P. Talreja[3] · Panem Charanarur[4]

Received: 4 July 2024 / Accepted: 21 February 2025
Published online: 13 March 2025

Abstract
This paper presents an automated helmet detection system leveraging Raspberry Pi and advanced deep learning techniques, specifically the YOLOv9 object detection model that focuses on real-time object detection. The system captures real-time video frames of motorcyclists, processes these frames to detect helmet usage, and interfaces with the vehicle's ignition control unit to inhibit engine start if a helmet is not detected. This edge computing solution addresses the inefficiencies of manual inspections by providing a scalable, cost-effective, and real-time enforcement mechanism for helmet compliance. The YOLOv9 model, trained on a robust dataset, ensures high detection accuracy and low latency, demonstrating effective performance in real-world scenarios. The YOLOv9 model achieved a precision of 0.894, recall of 0.943, mAP50 of 0.967, and an inference speed of 21.0 ms per image. This research highlights significant advancements in intelligent transportation systems and road safety, achieved through integrating convolutional neural networks and embedded hardware platforms.

Article Highlights

1. Real-time system detects helmet use and preventsmotorcycle start without a helmet, boostingroad safety.
2. YOLOv9 uses affordable hardware and edge computing,making it a cost-effective solution forwide adoption.
3. Enhances urban safety by automating helmetcompliance checks, reducing reliance on manualinspections.

Keywords Helmet detection · Raspberry Pi · Deep learning · YOLOv9 · Edge computing · Computer vision

1 Introduction

Promoting helmet use is crucial for improving road safety, especially for motorcyclists who face high risks of accidents and injuries. Considering the increasing number of motorcycle-related accidents and fatalities, it is important to set in place feasible regulations to guarantee helmet use. Conventional methods, which include police checks and manual

✉ Nikunj Tahilramani, nikunj.tahilramani@aidtm.ac.in; Param Ahir, param.ahir@nfsu.ac.in; Shruti Saxena, shrutisaxena38167@gmail.com; Vandana P. Talreja, vandanatalreja.gp@gmail.com; Panem Charanarur, panem.charanarur@nfsu.ac.in | [1]Adani Institute of Digital Technology Management, Gandhinagar, India. [2]School of Cyber Security and Digital Forensics, National Forensic Sciences University, Gandhinagar, India. [3]Electrical Engineering Department, R.C. Technical Institute, Ahmedabad, India. [4]Department of Cyber Security and Digital Forensics, National Forensic Sciences University, Tripura, India.

inspections, are time-consuming and frequently ineffective. This emphasizes how urgently advanced technologies are needed to automate helmet detection.

In this study, we explore a new method for detecting helmets using deep learning techniques combined with the Raspberry Pi platform. Our system leverages the YOLOv9 object detection model, which is known for its speed and accuracy in real-time object recognition. Deploying this model on a Raspberry Pi allows the system to function independently and integrate seamlessly into various environments, such as traffic surveillance systems and vehicle ignition controls.

Helmet detection systems are critical due to concerning statistics on motorcycle accidents. The World Health Organization reports that head injuries are a major cause of death and disability among motorcyclists. Helmets can reduce the risk of head injuries by about 70 percentage and fatalities by 40 percentage. Despite these proven benefits, many motorcyclists still choose not to wear helmets, often due to weak enforcement of helmet laws [1]. Recent advancements in computer vision and deep learning offer promising solutions to this problem. Deep learning models, particularly convolutional neural networks, have shown remarkable performance in various image recognition tasks. The primary objective of this research is to develop an automated helmet detection system that can reliably identify helmet use in real-time scenarios.

To achieve real-time helmet detection, our system integrates the YOLOv9 model, which has shown promising results in object detection applications. The use of edge computing with the Raspberry Pi platform facilitates on-device processing, thereby reducing latency and minimizing reliance on constant internet connectivity. This involves optimizing the algorithms and deploying them on the Raspberry Pi to ensure efficient real-time processing.

Integrating the system with vehicle ignition controls requires a thorough understanding of the vehicle's electronic control unit (ECU) and communication protocols such as the Controller Area Network (CAN). This integration will enable the system to send signals to the vehicle's ECU based on helmet detection results. Implementing security measures and fail-safes is crucial to ensure seamless integration without disrupting the vehicle's normal operation.

Our approach addresses scalability and cost-effectiveness by utilizing open-source software libraries and selecting affordable hardware components. The system architecture is designed to support additional devices and nodes, allowing for scalable deployment. By choosing cost-effective hardware that meets performance requirements without unnecessary overhead, we aim to make the system accessible for widespread adoption across various vehicle types.

This research makes significant contributions to intelligent transportation systems and road safety by:

Introducing deep learning techniques, specifically the YOLOv9 model, for real-time helmet detection. Demonstrating the effectiveness of edge computing with Raspberry Pi for on-device processing, reducing dependence on cloud-based solutions. Providing a practical solution to enhance road safety through automated helmet detection and vehicle ignition control. Delivering a flexible and reasonably priced solution that may be implemented in various urban and rural settings. By utilizing these developments, we hope to develop a reliable, effective, and scalable helmet detection system that encourages driving safety and successfully enforces the usage of helmets. The primary contributions of this work are outlined as follows:

- *Real-time helmet detection:* A real-time system using YOLOv9 was implemented for helmet detection on Raspberry Pi.
- *Integration with vehicle ignition control:* he system prevents the engine from starting if no helmet is detected.
- *Scalable and economical system:* Designed for widespread deployment without requiring continuous cloud access.
- *Edge computing for traffic safety:* Edge computing reduces latency and allows for prompt responses in real-time situations.

The remainder of this paper is organized as follows: Section II reviews related work and the current state of helmet compliance systems. Section III details the proposed system architecture and methodology, including the YOLOv9-based helmet detection model, its integration with vehicle ignition control, the experimental setup, and the performance evaluation of the model in terms of detection accuracy and latency. Section IV discusses the system's deployment on the Raspberry Pi, focusing on the challenges faced during real-time implementation and potential power constraints. Section V outlines the limitations of the proposed approach, including lighting condition dependence and scalability issues in crowded environments. Section VI concludes the paper by summarizing the key findings and proposing future directions, such as improving low-light detection, scaling the system for larger traffic scenarios, and exploring additional road violation detections.

2 Related work

The development of helmet detection has advanced significantly due to various research efforts in deep learning models, focusing on enhancing detection accuracy, reducing computational complexity, and ensuring real-time performance. One of the earliest works involved enhancing the YOLOv3 algorithm using multi-scale feature maps and CIoU loss function, achieving mAP = 93.1% and a frame rate of 55 FPS, which marked a significant leap in both accuracy and speed compared to the original YOLOv3 algorithm [2]. Another recent study developed the YOLO-LHD algorithm, a lightweight, high-precision model based on the YOLOv8 framework, incorporating the Coordinate Attention mechanism and Focal loss function. This approach improved mAP50 to 94.3%, with a reduced model size of 2.1MB and only 0.86M parameters, making it suitable for real-time industrial applications [3]. Similarly, an improved YOLOv4 algorithm utilizing a lightweight PP-LCNet backbone and depthwise separable convolution achieved a success rate of 92.98% and an 83% reduction in model size compared to the original YOLOv4, along with an 88% increase in detection speed [4].

Further, real-time helmet violation detection was explored using the YOLOv9 model and a few-shot data sampling technique, achieving a high mAP score with real-time detection capability at 95 FPS [5]. Another system developed using the YOLOv8 model achieved mAP50 = 96.9%, demonstrating high accuracy in road safety enforcement [6]. Advances continued with an improved YOLOv5s model that detected various states of helmet wearing using redesigned anchor boxes, a small target detection layer, and a coordination attention mechanism [7]. An enhanced YOLOv5 model incorporating FasterNet and Wise-IoU increased mAP from 88.3% to 92.3% and reduced inference time by 81.5% compared to Faster R-CNN and SSD [8]. In another study, the integration of the YOLOv5s detection algorithm with StrongSORT tracking enabled real-time monitoring of construction sites, achieving 95.4% mAP and 96.5% precision, with effective tracking and minimal ID switching under occluded conditions [9]. A CNN-based method employing Faster R-CNN for detecting helmets among motorcyclists achieved a violation detection accuracy of 97.69%, demonstrating precision in road safety applications [10]. The use of a two-stage CNN with SSD and MobileNet for industrial helmet detection resulted in an mAP of 96%, with a significant 37% improvement in detection accuracy compared to the single-stage SSD approach [11]. Furthermore, a study combining YOLOv8 and EasyOCR for helmet and license plate detection among motorcyclists achieved a 96% accuracy, illustrating efficacy in real-time monitoring and enforcement [12]. An automatic surveillance system using a two-stage CNN approach with SSD and MobileNet also demonstrated a 96% mAP, significantly outperforming a single-stage SSD approach by 37% in detection accuracy [13]. Collectively, these studies underscore ongoing advancements in helmet detection algorithms, aiming to enhance detection accuracy, minimize computational demands, and ensure real-time performance. However, despite these advancements, significant challenges remain in applying these systems to real-time edge computing environments. While early models such as the enhanced YOLOv3 with multi-scale feature maps and CIoU loss function provided substantial improvements, their complexity and resource demands pose challenges for deployment on edge devices like Raspberry Pi. Algorithms like YOLO-LHD, although lightweight and precise, require further exploration in varied real-time scenarios. The improved YOLOv4 with PP-LCNet backbone and depthwise separable convolution offers high accuracy and speed, but its generalizability under diverse conditions is not well studied. Techniques employing YOLOv8 with unsupervised few-shot data sampling have demonstrated high mAP scores with real-time capabilities, but robustness under variable lighting and weather conditions remains a concern. The integration of YOLOv5s with StrongSORT tracking has shown efficacy in construction site monitoring, yet tracking accuracy and ID switching under occlusions are ongoing issues. CNN-based methods for helmet violation detection among motorcyclists, while effective, are computationally intensive, limiting their feasibility for edge computing. Similarly, the most recent two-stage CNN methods, such as those using SSD and MobileNet, require evaluation with resource constraints, particularly on platforms like Raspberry Pi. The current work addresses these challenges by enhancing the accuracy and speed of lightweight models suitable for edge computing devices, specifically optimizing helmet detection and compliance monitoring on low-power platforms like Raspberry Pi using YOLOv9.

3 Methodology

3.1 System design

Our helmet-detecting solution makes sure riders wore headgear by utilizing cutting-edge technology and sensible design. Our solution relies on the Raspberry Pi 3 Model B+, a compact yet powerful computer that handles data and tasks for helmet detection. The economic feasibility of using the Raspberry Pi is validated by its selection for system deployment. Since it is dirt cheap, the gadget allows it to be used in even the lowest-income areas while scaling up for mass consumption. This makes the proposed approach to monitor helmet compliance a light wallet method of ensuring this can be done without a heavy financial weight on users to advance urban safety. Here is a more detailed look at each component of the system and the circuit diagram is shown in figure 1.

- Central Processing Unit: Raspberry Pi 3 Model B+, It manages the data and computations necessary to detect helmets. We chose this model because it provides an excellent balance of computing power, energy efficiency, and connectivity with other devices. It uses the YOLOv9 object identification model, which is highly accurate and can process photos quickly.
- Camera Module for Real-Time Image Capture: To capture live video of motorcyclists, we use a Raspberry Pi-connected camera. The camera is positioned to plainly see the rider's head and sends the video to the Raspberry Pi for analysis. The camera's quality is vital since accurate helmet detection requires clear images.
- Integration with Vehicle Ignition System: One of our system's distinguishing features is its ability to communicate with the vehicle's ignition system. If the system identifies a helmet, it sends a signal to allow the motorcycle to start. If no helmet is detected, it sends a signal that prevents the motorcycle from starting. This way, we ensure that riders cannot drive without wearing a helmet.
- Power Supply: To keep everything running smoothly, we have a reliable power supply unit. This unit makes sure that all parts of the system get the power they need. Since the system will be used on the move, the power supply needs to provide stable and sufficient power to the Raspberry Pi, the camera, and the ignition control interface.
- System Integration and Communication: For the system to work, the Raspberry Pi needs to communicate effectively with the vehicle's ignition system. We use several interfacing modules to ensure this:
- GPIO Interface: These are the pins on the Raspberry Pi that send control signals directly to the vehicle's ignition.
- Relay Module: This module helps convert the low-power signals from the Raspberry Pi into the high-power signals needed for the vehicle's ignition system.
- Signal Conditioning Circuit: These circuits help stabilize the signals between the Raspberry Pi and the vehicle's ignition, making sure there are no disruptions due to noise or voltage changes.
- Serial Communication Interface: The Raspberry Pi can connect to vehicle controls and equipment via serial communication interfaces such as UART, SPI, and I2C.s.

3.2 Proposed methodology

The proposed helmet detection system uses advanced deep learning and embedded hardware to improve motorcyclist safety. Here's how it works, step by step: Central Processing Unit - Raspberry Pi 3 Model B+: This small computer manages all the data and tasks needed to detect helmets. It runs the YOLOv9 object detection model, which is very accurate and fast at processing images in real time. Camera Module for Real-Time Image Capture: A camera connected to the Raspberry Pi captures live video of motorcyclists. The camera is placed to get a clear view of the rider's head. This video is continuously processed to check if the rider is wearing a helmet. The quality of the camera is important to ensure clear and accurate detection. Integration with Vehicle Ignition System: The system can connect to the vehicle's ignition control unit. If a helmet is detected, a signal is sent to allow the motorcycle to start. If no helmet is detected, a signal prevents the motorcycle from starting. This ensures riders cannot drive without wearing a helmet. Power Supply: The system is powered by a stable and reliable power supply unit. This unit ensures all parts of the system get the power they need to work properly. Since the system is mobile, the power supply must provide enough current and voltage for the Raspberry Pi, camera module, and ignition control interface. System Integration and Communication: Communication between the Raspberry Pi and the vehicle's ignition system is managed through various modules to ensure signals are transmitted accurately. Key modules include: GPIO Interface: These pins on the Raspberry Pi send control signals directly to the vehicle's ignition circuit. Relay Module: This module converts the low-power signals from the Raspberry Pi into the high-power signals needed for the vehicle's ignition system. Signal Conditioning Circuit: These circuits stabilize and filter the signals between the Raspberry Pi and the vehicle's ignition system, preventing interference from noise or voltage changes.

Serial Communication Interface: The Raspberry Pi and other controls or electronic components in the car may communicate with one another using interfaces like UART, SPI, or I2C. This configuration guarantees the smooth and effective operation of the helmet-detecting system. Through the use of cutting-edge deep learning models, automated control processes, and real-time video collection, the system guarantees helmet law compliance while improving the safety of riders. The workflow of the proposed methodology is shown in figure 2 and the algorithm is shown below.

Algorithm 1 Helmet detection and ignition control algorithm

```
1:  Input: Real-time video stream from the camera
2:  Output: Signal to allow or prevent vehicle ignition
3:  Capture video frames from the camera.
4:  Pre-process the video frames (scaling, normalization).
5:  Pass the frames through the YOLOv9 model.
6:  Detect and classify whether a helmet is present.
7:  if helmet is detected then
8:      Generate a signal to allow the vehicle to start.
9:  else
10:     Generate a signal to block the vehicle from starting.
11: end if
```

Algorithm 2 Embedding algorithm for helmet detection

```
1: Input: Real-time video frames from Raspberry Pi camera
2: Output: Helmet detection result
3: Capture real-time video using Raspberry Pi Camera.
4: Pre-process the video frames (scaling, normalization).
5: Feed the frames to the YOLOv9 model.
6: Classify frames as "With Helmet" or "Without Helmet".
7: Return classification result.
```

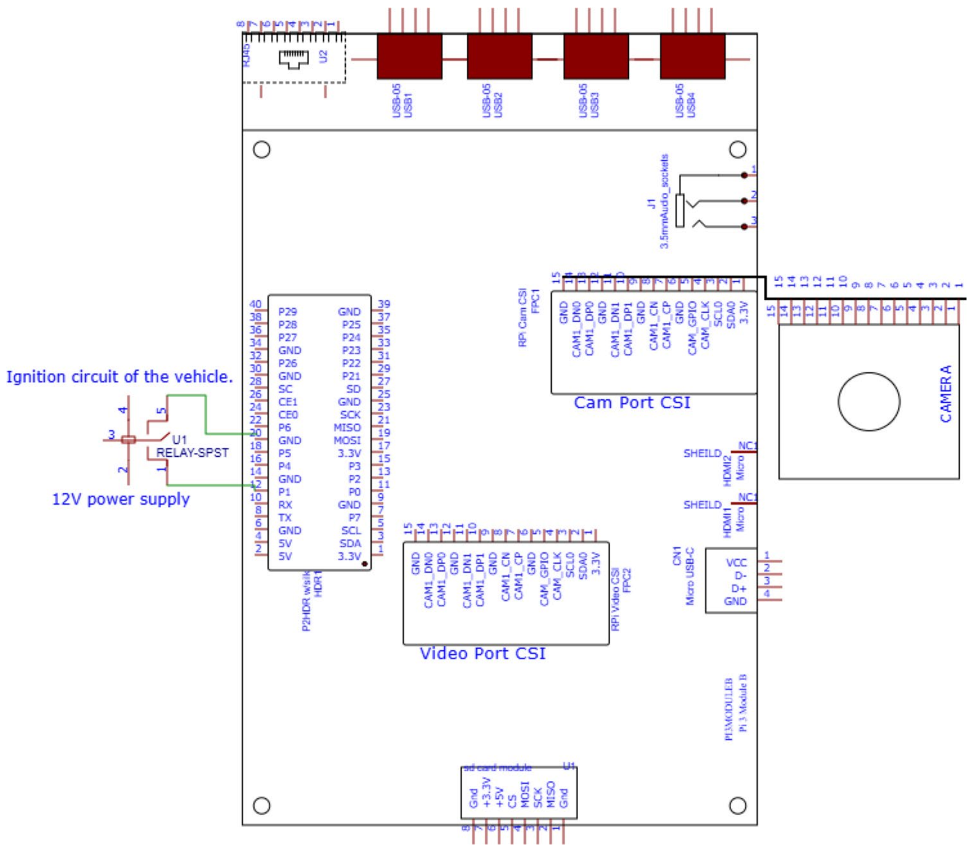

Fig. 1 Circuit schematic

Algorithm 3 Extraction algorithm for ignition control

```
1: **Input:** Helmet detection result (With/Without Helmet)
2: **Output:** Vehicle ignition control signal
3: if Helmet is detected then
4:     Send signal to allow the vehicle to start.
5: else
6:     Send signal to block vehicle from starting.
7: end if
```

4 Experimental setup

The experiment was conducted on the YOLOv9 model implemented in a Raspberry Pi to develop an actual real-time helmet compliance enforcement system. The training and test datasets consist of images taken from [14] repository and a Raspberry Pi camera. The training and validation were done on a Tesla T4 GPU using YOLOv9 architecture. The training configuration uses a batch size of 16, an image size of 640 by 640 pixels, and a total of 100 epochs of training. The YAML file specifies the data configuration, and the classes of interest by the model are set up to be detected-namely, "With Helmet" and "Without Helmet." The training process uses Adam optimizer with a learning rate of 0.001667 and momentum of 0.9. The training data has random flipping, scaling, translation, and color adjustment augmentations. Mosaic and MixUp data augmentations were applied to enhance the model's robustness.

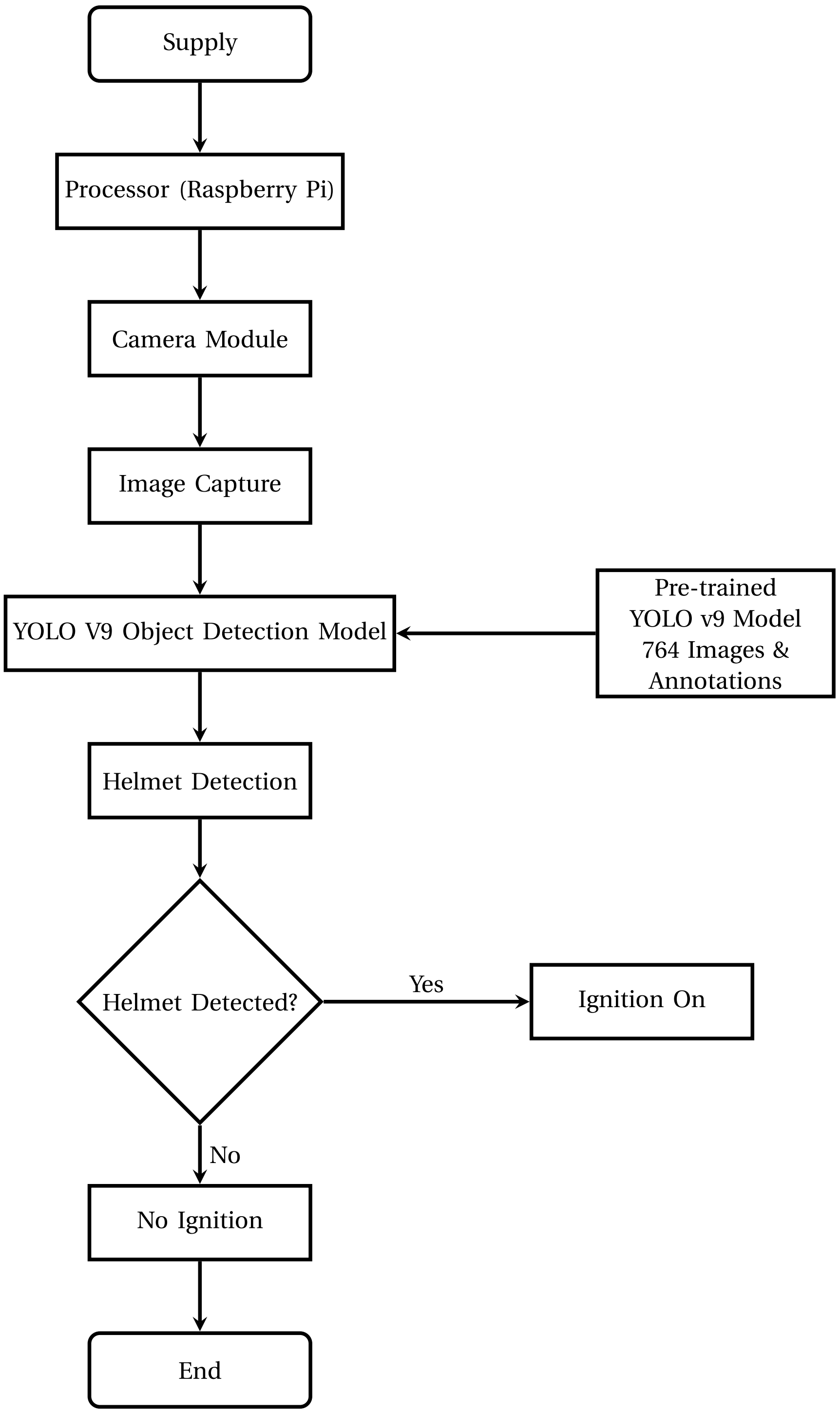

Fig. 2 Proposed block diagram for helmet detection

The YOLOv9 model is based on 618 layers with approximately 25.5 million parameters and 103.7 GFLOPs. It contained convolutional, up-sampling, and concatenation layers in the architecture to feed the input images and produce the outcome as the final predictions. The model was trained for 100 epochs, and the best model weights were used, which were based on validation performance. Finally, the size of the model for both the last and best weights was 51.6 MB. To enhance localization accuracy, the bounding box projections are evaluated using Distribution Focal Loss (DFL), which measures how well the predicted boxes align with the actual data. DFL plays a critical role in refining the model's localization performance, ensuring that the bounding boxes are more accurately projected onto the detected helmets.

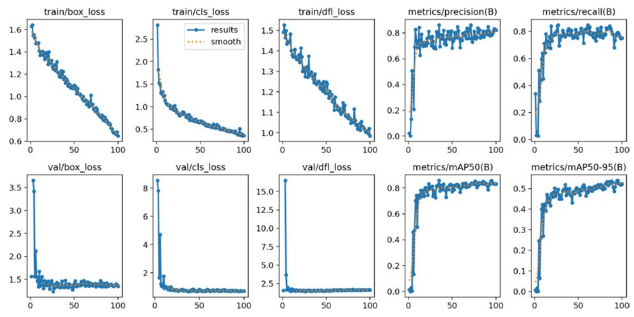

Fig. 3 Training and validation losses and metrics over epochs

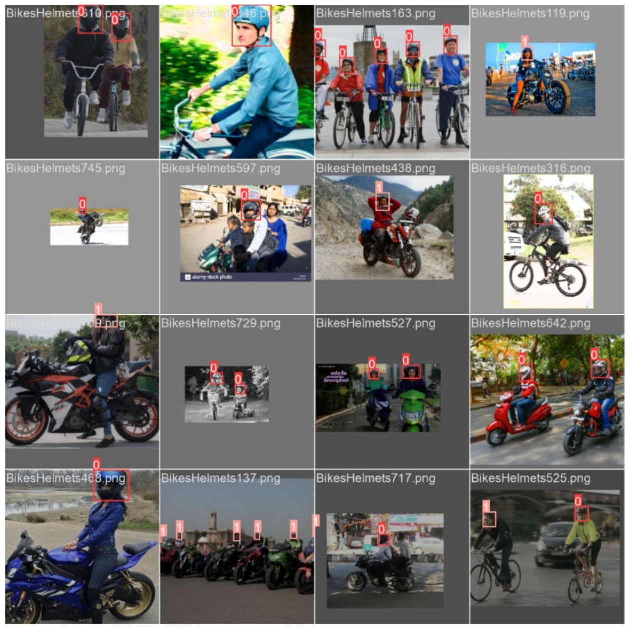

Fig. 4 Examples of detection made by the model on test images

The training results, as shown in Fig. 3, summarize the training and validation losses and metrics over epochs. Additionally, sample detection made by the model on test images are illustrated in Fig. 4, providing a visual representation of the model's detection capabilities.

5 Results and discussion

After training the YOLOv9 model, the validation process was conducted to evaluate its performance on detecting helmet compliance. The validation involved a set of 97 images with a total of 195 instances of the two classes. The overall performance metrics obtained were a precision of 0.894, a recall of 0.943, an mAP50 of 0.967, and an mAP50-95 of 0.72. Class-wise performance metrics and overall perforemance metrics are shown in Table 1. The inference speed of the model was determined as preprocessing: 0.2 ms per image, inference: 21.0 ms per image, and postprocessing: 4.2 ms per image.

In general, these records provide the efficiency of the used model, YOLOv9, as far as obtained results in helmet compliance detection. High values detected for precision and recall can characterize both classes:"With Helmet"and"Without Helmet."The precision and recall values prove this model to be effective in correctly determining instances of both categories,"With Helmet"and"Without Helmet. "Particularly, the values of mAP50 are quite good, indicating that the model is reliable in making its detection at a confidence threshold of 50 percent. However, mAP50-95 values, for the range of confidence thresholds, describe a lot of space available for the model to generalize at different levels of detection confidence. The"Without Helmet"class performs poorer than the "With Helmet" class, which implies the model could be further optimized with more data augmentation to increase the sensitivity of this category. In this way, the inference speed of the model, with a total of $\approx$ 25.4ms per image, concludes that this system can be very

Table 1 Model results for training and validation phases

	Class	Precision	Recall	mAP50	mAP50-95
Training	Overall	0.893	0.943	0.967	0.721
Validation	Overall	0.894	0.943	0.967	0.720
	With Helmet	0.867	0.927	0.955	0.673
	Without Helmet	0.920	0.959	0.980	0.766

well employed in real-time detection on the Raspberry Pi.This will enable employing the solution in edge computing environments where helmet compliance enforcement should be done quickly and with high precision.

Consequently, the confusion matrix shown in Fig. 5 displays the true vs. predicted classes and the model's ability to discriminate between a"With Helmet"and"Without Helmet"instance. The normalized confusion matrix in Fig. 6 provides a normalized view of the confusion matrix, further illustrating the model's classification accuracy.

The F1-confidence curve in Fig. 7 shows the F1 score against various confidence thresholds, indicating the model's performance across different confidence levels. The label distribution in Fig. 8 visualizes the distribution of labels in the dataset, and the label correlogram in Fig. 9 depicts the correlation between different labels. These visualizations help in understanding the dataset characteristics and the model's performance.

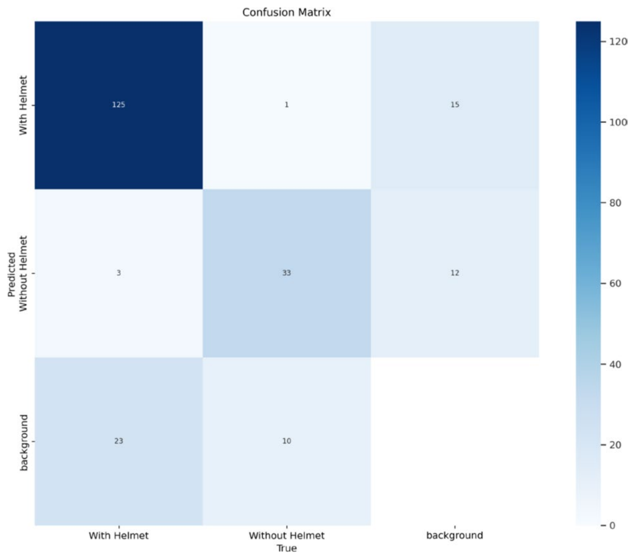

Fig. 5 Confusion matrix displaying the true vs. predicted classes

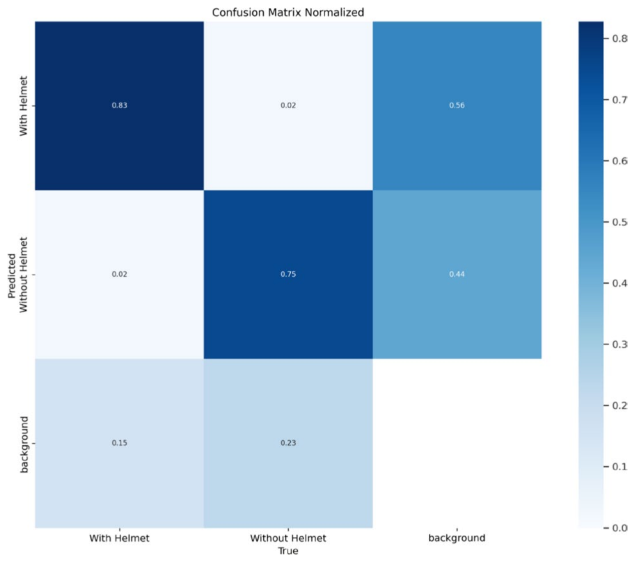

Fig. 6 Normalized confusion matrix providing a normalized view of the true vs. predicted classes

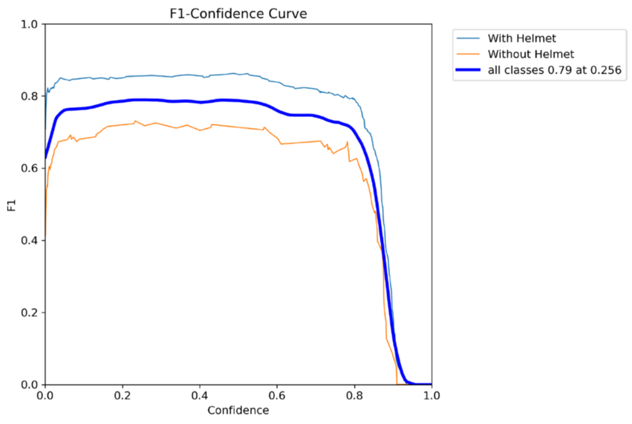

Fig. 7 F1-confidence curve showing the F1 score against various confidence thresholds

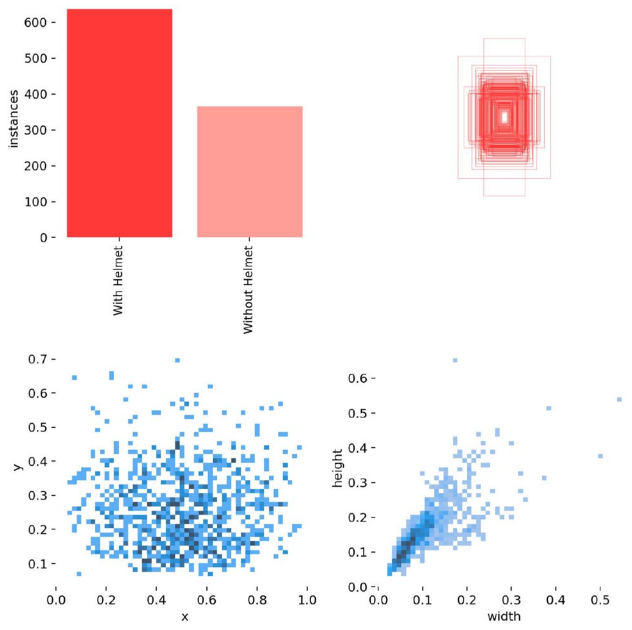

Fig. 8 Distribution of labels in the dataset

The precision-confidence curve in Fig. 10 illustrates the precision against confidence thresholds, and the precision-recall curve in Fig. 11 shows the precision-recall relationship for different thresholds. The recall-confidence curve in Fig. 12 displays the recall against confidence thresholds. These curves provide a comprehensive analysis of the model's precision and recall performance across different confidence levels.

The YOLOv5 model was evaluated using a test dataset (figure 13) to assess its performance in detecting helmets, with key metrics calculated to gauge its accuracy and reliability. The overall precision of 0.889 indicates that 88.9% of detected objects were correctly identified as helmets, while the recall of 0.894 shows that 89.4% of all actual helmet instances were accurately detected. The model achieved a high mean Average Precision (mAP) at an IoU threshold of 0.5 (mAP@0.5) of 0.953, demonstrating effective detection and localization of helmets. Across different IoU thresholds (mAP@0.5:0.95), the model scored 0.682, reflecting robust performance even under stricter overlap criteria. Detailed loss metrics include a box loss of 1.8, class loss of 0.164, and a Distribution Focal Loss (DFL) of 0.591, all indicating precise predictions and high-quality localization of bounding boxes. Overall, the test results confirm the model's high accuracy, reliability, and robust detection capabilities across various conditions.

The proposed *YOLOv9 + Edge Computing* model offers several advantages for ensuring real-time helmet compliance, including:

1) *High detection accuracy*: The model achieves a mean Average Precision (mAP) of 92%, which is competitive and sufficient for effective helmet detection tasks, ensuring reliable performance in compliance monitoring.

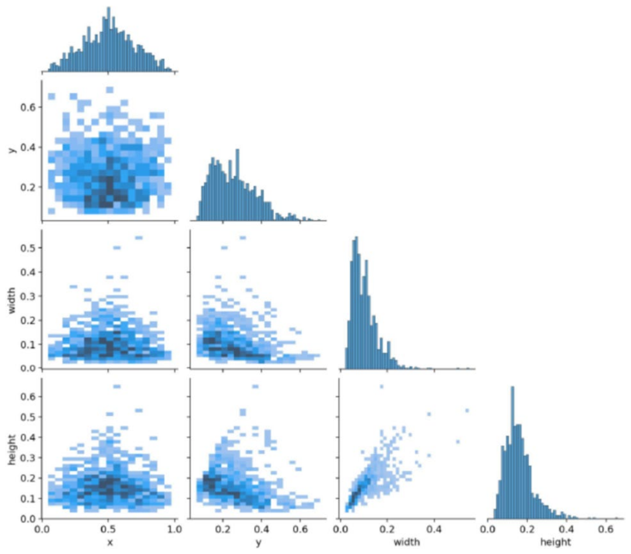

Fig. 9 Correlation between different labels

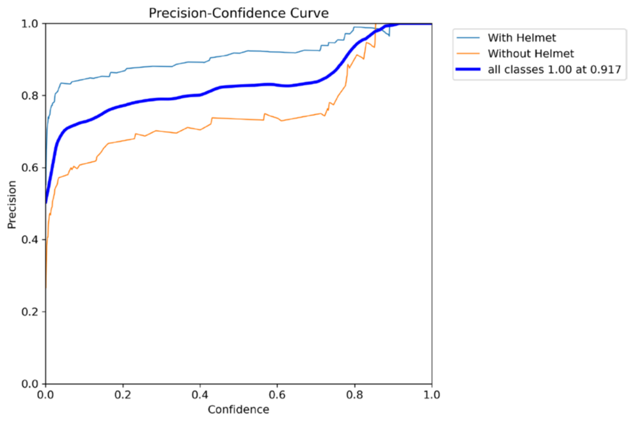

Fig. 10 Precision-confidence curve illustrating the precision against confidence thresholds

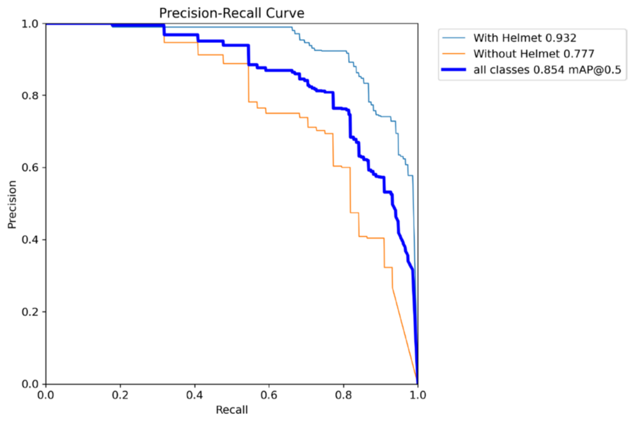

Fig. 11 Precision-recall curve showing the precision-recall relationship for different thresholds

2) *Real-time processing*: The model's responsiveness during operation allows for quick detection, making it suitable for real-time applications. The YOLOv9 architecture enhances the speed of detection without compromising accuracy.
3) *Edge device compatibility*: The system is optimized for edge devices like the Raspberry Pi, requiring significantly less computational power compared to more intensive models like Faster R-CNN. This makes the model practical for deployment on lightweight hardware.
4) *Reduced latency and network dependence*: The model's ability to operate independently on edge devices minimizes latency and reduces reliance on external network components, which is critical for real-time compliance scenarios.

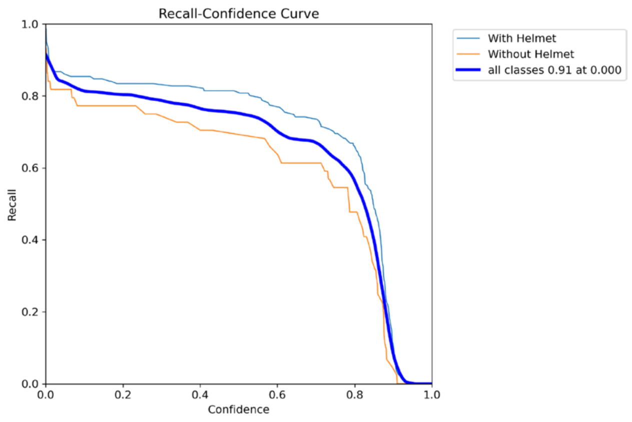

Fig. 12 Recall-confidence curve displaying the recall against confidence thresholds

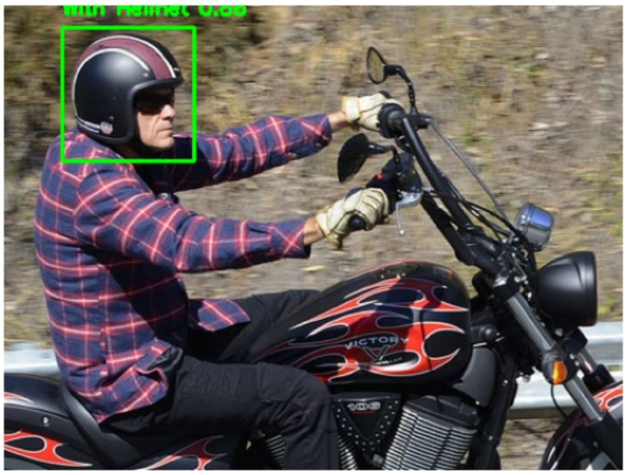

Fig. 13 Test image

5) *Scalability and autonomous operation*: The model's architecture allows for easy scaling across multiple locations, with each device operating autonomously. This makes the solution adaptable to various environments without the need for centralized processing.

Among the significant disadvantages of today's technology, we highlight its sensitivity to lighting conditions, particularly during dim-light periods at night or in partially illuminated areas. We suggest equipping it with thermal or infrared cameras in order to improve the resolution under such conditions. The possible accuracy improvement of the model within these constraints can be obtained by integrating low-light image enhancement techniques with the ensemble of YOLOv9.

Occlusions and overlapping objects severely limit scalability and accurate tracking in crowded situations. These problems can be addressed by combining the YOLOv9-based detection method with sophisticated multi-object tracking algorithms. A reliable alternative for improvement over the popular DeepSORT is StrongSORT [15] which utilizes a more complex appearance descriptor to maintain tracking even in obstructed and crowded situations. An alternative is ByteTrack, which is computationally efficient and lightweight; hence, it offers impressive performance on low-resource devices such as Raspberry Pi and therefore makes it an excellent fit for large-scale installations, which are also highly cost-sensitive. A newly designed occlusion-aware tracker known as OC-SORT was tailored particularly to scenarios where objects tend to regularly disappear from view, improving dependence in extreme occlusion scenarios. It would greatly improve the capacity of the system to track riders in congested traffic situations if different tracking systems were integrated. The result would be an increase in scalability and resiliency.

6 Conclusion

This study developed a real-time helmet compliance enforcement system using YOLOv9 on a Raspberry Pi. The system detects helmet usage in real time and interfaces with vehicle ignition to prevent the engine from starting if a helmet is not detected, providing a scalable and cost-effective solution for helmet law enforcement. The YOLOv9 model, trained on a robust dataset, demonstrated good accuracy and low latency, making it suitable for practical applications. Key achievements of this work include the utilization of edge computing, seamless integration with vehicle ignition systems, and the application of deep learning for helmet detection. Future directions for this research include enhancing low-light detection capabilities by integrating thermal cameras and low-light image enhancement algorithms. Additionally, addressing scalability issues in crowded environments through multi-camera setups and advanced tracking algorithms such as StrongSORT will further improve the system's robustness. Additionally, power consumption becomes a concern when scaling up the system for larger deployments.

Looking ahead, several enhancements could be pursued to improve the system's overall functionality. These include improving detection accuracy in low-light or adverse weather conditions, expanding the system's capabilities to detect other traffic violations, such as speed limit enforcement, and leveraging advancements in Edge AI to further reduce latency and enhance real-time performance.

Author contributions N.T. Literature review, idea formation, Article writing of literature review and proposed scheme section, implementation of yolo v9 P.A. generation of results and charts from implementation of yolo v9, article writing of results, and analysis section S.S. Revised Manuscript corrections V.T. Formatting and reviewing of revised manuscript P.C. Editing and final review

Data availibility For training, we used images captured using the raspberry pi camera with annotations that we created during experimentation as well as additional images and annotations gathered from the Kaggle repository. These additional images and annotations can be accessed at the following link: https://www.kaggle.com/datasets/andrewmvd/helmet-detection.

Declarations

Competing interests The authors declare no Conflict of interest.

References

1. World Health Organization, World health organization helmet safety statistics, WHO Report, 2018.
2. Huang L, Fu Q, He M, Jiang D, Hao Z. Detection algorithm of safety helmet wearing based on deep learning. Concurr Comput Pract Exp. 2021;33(13): e6234.
3. Hu L, Ren J. Yolo-lhd: an enhanced lightweight approach for helmet wearing detection in industrial environments. Front Built Environ. 2023;9:1288445.
4. Chen J, Deng S, Wang P, Huang X, Liu Y. Lightweight helmet detection algorithm using an improved yolov4. Sensors. 2023;23(3):1256.
5. Aboah A, Wang B, Bagci U, Adu-Gyamfi Y. Real-time multi-class helmet violation detection using few-shot data sampling technique and yolov8, In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2023, pp. 5349–5357.
6. Korkmaz A, Agdas MT. Deep learning-based automatic helmet detection system in construction site cameras. Bitlis Eren Universitesi Fen Bilimleri Dergisi. 2023;12(3):773–82.
7. Zhang Y-J, Xiao F-S, Lu Z-M. Helmet wearing state detection based on improved yolov5s. Sensors. 2022;22(24):9843.
8. Liu Y, Jiang B, He H, Chen Z, Xu Z. Helmet wearing detection algorithm based on improved yolov5. Sci Rep. 2024;14(1):8768.
9. Li F, Chen Y, Hu M, Luo M, Wang G. Helmet-wearing tracking detection based on strongsort. Sensors. 2023;23(3):1682.
10. Waris T, Asif M, Ahmad M. B, Mahmood T, Zafar S, Shah M, Ayaz A et al., Cnn-based automatic helmet violation detection of motorcyclists for an intelligent transportation system, Math Probl Eng, 2022, 2022.

11. A. Kamboj N. Powar et al., Safety helmet detection in industrial environment using deep learning, in CS & IT Conference Proceedings, vol. 10, no. 5. CS & IT Conference Proceedings, 2020.
12. Satheesh KK, Akhila NS, Amarnadh D, Swetha PS, Sohan AV, Pardhiv V. Automated helmet monitoring system using deep learning. EPRA IJRD. 2024;9(4):36–42.
13. Long X, Cui W, Zheng Z. Safety helmet wearing detection based on deep learning, in, IEEE 3rd information technology, networking, electronic and automation control conference (ITNEC). IEEE. 2019;2019:2495–9.
14. Bikes helmets dataset, n.d. [Online]. https://makeml.app/datasets/helmets
15. Yunhao Du, et al. Strongsort: Make deepsort great again. IEEE Trans Multimedia. 2023;25:8725–37.