AI-Based Accident Detection System

E. Prabhakar[1], P. Ajay Kumar Reddy[2], K. Kalyan Kumar[3], M. Vamshi Naik[4]
[1]*Assistant Professor, Dept of ECE, TKR College of Engineering and Technology*
[2,3,4] *Student, Dept of ECE, TKR College of Engineering and Technology*

***Abstract*— Road accidents have become a serious concern in today's world, often leading to injuries, loss of life, and delayed emergency response. To help address this issue, this project presents an AI-based accident detection system that uses a camera and deep learning techniques to automatically identify accidents as they happen. The system is built using a Raspberry Pi 4 and a camera module to capture live video. A trained object detection model is used to analyze the video and detect accidents. When an accident is detected, the system sends an alert message instantly using Twilio SMS service, helping to ensure a quicker response from emergency services or concerned authorities. While the system works best in controlled setups, it represents an early step toward developing low-cost, intelligent solutions for accident detection. With further improvements, it could potentially be applied in scenarios such as accident-prone zones and remote highway stretches.**

Index Terms— YOLOv5n, Accident Detection, ONNX, Raspberry Pi 4, Flask, Twilio

I. INTRODUCTION

Road accidents are a growing concern across the world, often resulting in severe injuries, fatalities, and delayed emergency response. In India alone, more than 4.6 lakh accidents were reported in 2022, many of which occurred in areas with limited or no human monitoring. Locations such as sharp turns and remote highway stretches are particularly vulnerable due to their low visibility or sparse traffic presence. Timely detection of accidents in such places can play a key role in minimizing the impact and saving lives.

With the rise of Artificial Intelligence (AI) and Machine Learning (ML), smart solutions can now be developed to detect such incidents automatically using image or video input [4], [5], [15]. One such advancement is object detection, which allows systems to recognize and locate objects or events in real-time video feeds. These technologies provide the foundation for building practical systems that can enhance public safety.

This project introduces an AI-based accident detection system using a Raspberry Pi 4 and camera module. It integrates a YOLOv5n object detection model [7], trained to differentiate between accident and non-accident visuals. The model is converted to a lightweight ONNX format [9], deployed on a Raspberry Pi 4 for efficient local processing and accessed through a simple Flask web interface. When an accident is identified, the system sends an SMS alert through the Twilio API [12], ensuring faster response by emergency services or concerned authorities.

While designed as a prototype, the system demonstrates the potential of AI-powered solutions in enhancing road safety, especially in areas where constant human monitoring or advanced surveillance systems are unavailable.

Existing System:

Earlier methods for accident detection primarily relied on manual observation through surveillance cameras or human reporting, which often led to delayed responses and limited real-time action. In some cases, traditional machine learning algorithms like Support Vector Machines (SVM) and Decision Trees were used with handcrafted features such as edge detection, motion vectors, or color patterns to classify accident scenes. While these approaches introduced automation, they depended heavily on manually designed feature extractors and lacked the adaptability required for varied environments or complex accident scenarios.

With the rise of deep learning, image-based classification models like basic Convolutional Neural Networks (CNNs) and lightweight architectures such as MobileNetV2 began to replace traditional methods. These models improved accuracy and reduced manual effort by learning features directly from data. However, they were mostly limited to classifying individual frames as "accident" or "no accident" without the ability to localize objects or on-device live video inference. These limitations underscore the need for an object detection-based approach capable of identifying accidents with

spatial accuracy and generating immediate alerts — a gap addressed by the proposed system using YOLOv5n.

Proposed System:
The proposed system is designed to automatically detect accidents using deep learning and camera-based monitoring without requiring human intervention. It aims to provide a low-cost, efficient solution that can be deployed in areas with limited surveillance or delayed emergency reporting.

This system uses a Raspberry Pi 4 as the processing unit and a Raspberry Pi Camera Module v1.3 to capture live video input. A YOLOv5n object detection model, trained and exported in ONNX format, is used to analyze the video feed for accident detection. The model runs directly on the Raspberry Pi using a lightweight inference setup, enabling real-time prediction with minimal resources.

A Flask-based web interface is provided to start or stop inference and to view prediction results, including confidence scores. In addition, the system includes a manual mode where users can test predefined images or videos. If an accident is detected during live monitoring, an SMS alert is sent to the concerned authorities using the Twilio API, ensuring timely action. The system is compact, easy to operate, and suitable for locations like remote roads or accident-prone areas where traditional monitoring methods are difficult to implement.

II. DESIGN PROCEDURE

The AI-Based Accident Detection System is designed as a lightweight, modular, and edge-deployable solution that processes visual inputs to detect potential road accidents [20]. The architecture emphasizes low-latency inference, minimal hardware dependency, and real-time user interaction, making it ideal for surveillance and traffic monitoring applications in resource-constrained environments.
At a high level, the system is composed of five core components: (1) input acquisition via a camera module or uploaded media, (2) preprocessing and frame handling, (3) accident prediction using an ONNX-based YOLOv5n model, (4) user interface via a Flask web application, and (5) alert and logging mechanisms. The design ensures separation of concerns between the inference logic and the user interface, allowing for smoother integration and better system stability.

A Raspberry Pi 4 device [10] serves as the central processing unit, receiving input either from a connected 5MP camera module (for real-time inference) or from stored images and video files (for manual inference). Frames captured or uploaded are resized and normalized before being passed to the ONNX runtime, which hosts a YOLOv5n model trained to classify scenes as either “Accident” or “No Accident.”
The model performs inference either continuously (live feed) and results are rendered in a web-based interface built with Flask. Users can start or stop real-time monitoring, view recent logs, or manually test the system with images and videos via separate routes. Predictions are displayed alongside confidence scores for transparency. The last 100 predictions are stored in a rotating log file, and a Twilio SMS alert system is integrated to notify concerned authorities when an accident is detected in the live stream.

A high-level block diagram of the system is shown in Fig. 1, illustrating the flow from input to inference, UI display, and alerting.

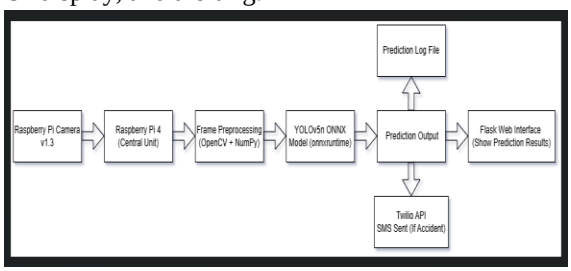

Fig. 1. Block diagram of the AI-Based Accident Detection System

III. IMPLEMENTED DESIGN

The proposed system is a lightweight accident detection framework that utilizes computer vision techniques to identify road accidents in image or video input. The system leverages a YOLOv5n (You Only Look Once version 5 - Nano) model trained to classify scenes as either “Accident” or “No Accident.” To enable efficient deployment on low-power edge devices [17], the trained model is exported to ONNX (Open Neural Network Exchange) format and executed using the ONNX Runtime on a Raspberry Pi 4. The core functionality of the system is divided into two modes of operation:

(i)Real-Time Inference Mode:
In this mode, the system captures frames continuously from a Raspberry Pi Camera v1.3. Each

frame is preprocessed using OpenCV [11] and NumPy libraries — including resizing to 416×416, RGB conversion, and normalization. The processed frame is then passed to the YOLOv5n ONNX model for inference. If the model detects an accident with a confidence above a defined threshold, the result is displayed on a Flask-based web interface, stored in the prediction log file, and an SMS alert is sent to concerned authorities using the Twilio API. This enables timely response in live monitoring scenarios.

(ii)Manual Inference Mode:
This mode allows users to manually select and upload images or videos via the web interface. For image input, a single prediction is generated after preprocessing. For video input, the system analyzes every 10th frame to optimize performance and reduce processing load. If an accident is detected in any of the sampled frames, the final result is labeled as "Accident." Otherwise, it is marked as "No Accident." This feature is particularly useful for offline evaluation or reviewing archived footage.
In both modes, prediction results are annotated with confidence scores and visual overlays. A rotating log mechanism is used to store the latest 100 predictions, which can be viewed or downloaded through the web interface. This ensures traceability and facilitates debugging or record-keeping.

The overall workflow of the proposed system is illustrated in the flowchart shown in Fig. 2. It summarizes the input handling, preprocessing steps, prediction generation, and output routing for both inference modes.

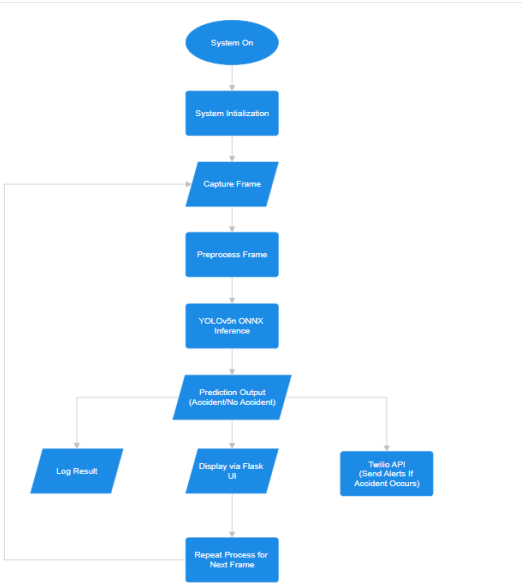

Fig. 2. Flowchart of the proposed AI-Based Accident Detection System

IV. RESULTS AND DISCUSSION

The AI-Based Accident Detection System was developed and deployed on a Raspberry Pi 4 using a YOLOv5n model optimized for edge inference. The system was implemented using Python, OpenCV, NumPy, Flask, and ONNX Runtime libraries. The web interface was developed with Flask templates and supports both real-time and manual inference modes.

Model Training and Conversion:
A custom dataset was created by collecting and curating accident-related and normal scene images from various open sources including Kaggle and YouTube frame extractions. The dataset was labeled in YOLO format with a single class: accident. Images without accidents were assigned empty .txt files, allowing the model to learn from both positive and negative examples. The YOLOv5n model was trained using the Ultralytics YOLOv5 training pipeline, and the best-performing weights were exported to ONNX format for Raspberry Pi compatibility.

Deployment on Raspberry Pi:
The trained ONNX model was deployed on a Raspberry Pi 4 (4GB RAM) running Raspberry Pi OS 64-bit. Inference was executed using the ONNX Runtime with CPUExecutionProvider. The camera module (Raspberry Pi Camera v1.3) captured live video input, while image and video files were handled through the manual mode UI.

Real-Time Inference Results:
In real-time mode, frames were captured continuously from the camera, processed, and passed to the model. If an accident was detected, the system displayed the label with a bounding box, logged the result with a timestamp, and sent an alert SMS using Twilio. The system achieved smooth performance with frame-wise prediction at intervals of ~1–2 seconds depending on lighting and motion [19].

Fig. 3 shows a frame captured during real-time inference where the system successfully detected an accident scenario. The prediction label is overlaid on the frame.

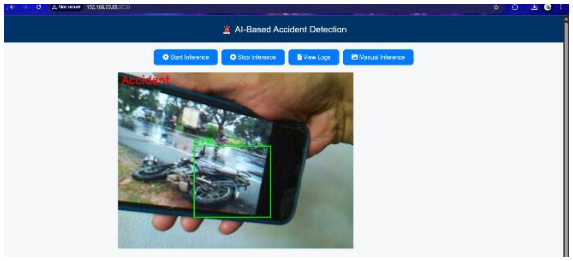

Fig. 3. Real-time inference result showing an accident prediction

Fig. 4 displays a real-time frame classified as a non-accident scene. No accident-related features were detected, and the system returned a 'No Accident' prediction.

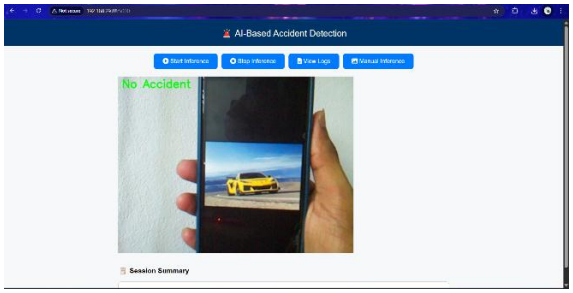

Fig. 4. Real-time inference result showing a no accident prediction

Fig. 5 shows the SMS alert sent by the system using the Twilio API when an accident is detected during real-time inference.

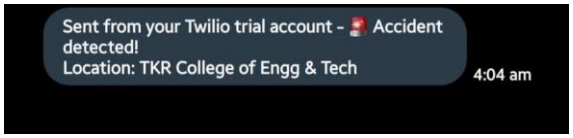

Fig. 5. Twilio SMS alert triggered upon accident detection

Manual Inference Results:
In manual mode, users selected images or videos from preloaded folders. For images, predictions were rendered with annotated output. For videos, every 10th frame was processed, and the final label was determined based on the presence or absence of accidents in sampled frames.

Fig. 6 displays the manual image inference interface, where users can select an input image from the dropdown list before initiating the prediction.

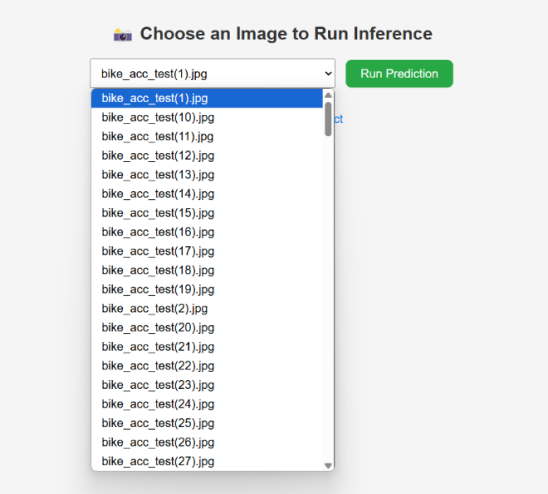

Fig. 6. Image selection interface in manual inference mode

Fig. 7 shows the result of a manually selected image after inference. The system processed the input and returned an 'Accident' prediction with visual feedback.

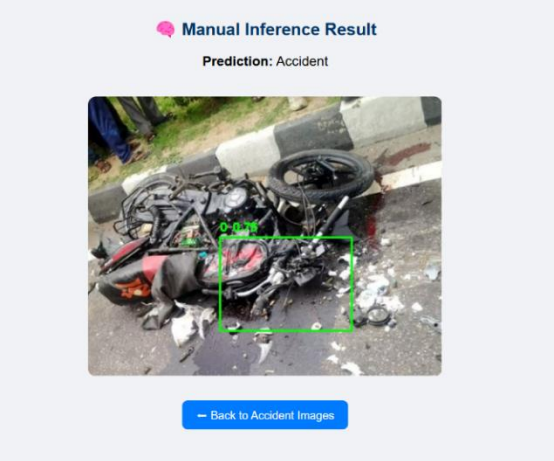

Fig. 7. Manual image inference result showing an accident prediction

Fig. 8 shows the manual video inference interface where users can select a video file from the dropdown before initiating the accident detection process.

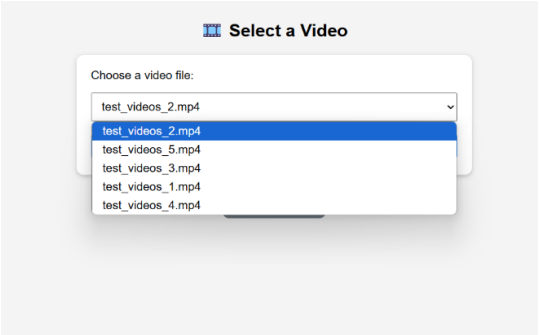

Fig. 8. Video selection interface in manual inference mode

Fig. 9 displays the result after analyzing a user-selected video. The system processed every 10th frame and returned 'Accident' prediction based on the sampled frames.

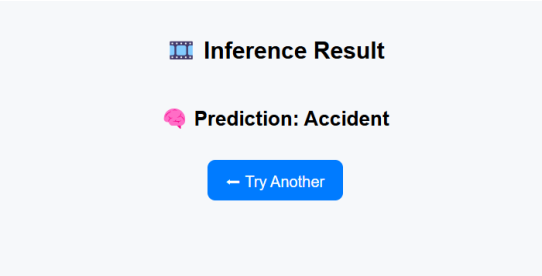

Fig. 9. Manual video inference result showing accident prediction

Logging and Output:
The system maintains a rotating log of the last 100 predictions, viewable and downloadable via the web interface. Each entry includes timestamp, predicted label, and confidence score. The Twilio alert system was tested and successfully delivered SMS notifications for accident scenarios.

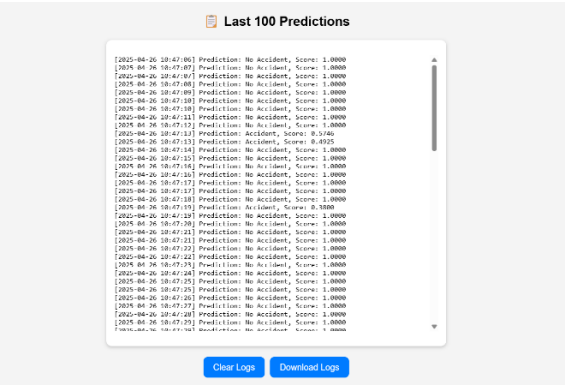

Fig. 10. Log view displaying the last 100 prediction results with timestamps

Model Evaluation Metrics:
The YOLOv5n model was evaluated on a test set of 420 images using the built-in validation script (val.py). The following performance metrics were obtained:

Table I

Metric	Value
Precision	0.861
Recall	0.797
mAP@0.5	0.862
mAP@0.5:0.95	0.497

V. CONCLUSION

This project successfully demonstrates the development and deployment of an AI-based accident detection system using a Raspberry Pi and deep learning techniques. The core objective was to design a cost-effective, real-time monitoring solution capable of detecting accidents from live camera feeds and triggering alerts without human intervention. The YOLOv5n object detection model, trained on a custom accident dataset, formed the backbone of the system. Its conversion to ONNX format enabled optimized inference on the Raspberry Pi 4, proving that edge devices can handle real-time computer vision tasks effectively. The use of a Flask web interface added user accessibility, while the integration of Twilio ensured that accident events could be communicated quickly via SMS alerts. The system was tested across multiple scenarios, including live detection, manual image testing, and video inference. It delivered consistent results in controlled environments and showcased its potential for deployment in surveillance setups such as remote roads, blind curves, and campus intersections. The logging feature further enhanced transparency and allowed for post-event validation. While the current version focuses on binary accident detection, the modular design leaves room for future improvements, such as integrating GPS, extending to multi-class classification, or enhancing robustness under varied environmental conditions. Overall, the project highlights the feasibility of combining affordable hardware, efficient deep learning models, and real-time communication tools to build intelligent, self-contained surveillance systems tailored for public safety applications.

REFERENCES

[1] A. Darokar, A. Talware, A. Jadhav, D. Guntiwarr, S. Dandhare, and D. Dandekar, “Accident Detection System Using YOLOv8 and CNN Algorithms,” *International Journal of Scientific Research in Engineering and Management (IJSREM)*, vol. 07, no. 04, pp. 1–5, 2023.

[2] T. Tamagusko, M. G. Correia, M. A. Huynh, and A. Ferreira, “Deep Learning Applied to Road Accident Detection with Transfer Learning and Synthetic Images,” *Information*, vol. 12, no. 6, 2021.

[3] A. P. Adil, M. G. Anandhu, J. E. Joy, and T. S. Karetha, “Accident Detection in Surveillance Camera,” *International Research Journal of Engineering and Technology (IRJET)*, vol. 08, no. 04, pp. 1027–1031, 2021.

[4] H. Ghahremannezhad, H. Shi, and C. Liu, "Real-Time Accident Detection in Traffic Surveillance Using Deep Learning," in *International Joint Conference on Neural Networks (IJCNN)*, 2020.

[5] F. Liang, "Deep Learning Based Traffic Accident Detection," in *2020 International Conference on Artificial Intelligence and Big Data (ICAIBD)*, pp. 369–373, 2020.

[6] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint arXiv:2004.10934*, 2020.

[7] G. Jocher et al., "YOLOv5 by Ultralytics," GitHub Repository, [Online]. Available: https://github.com/ultralytics/yolov5 [Accessed: Apr. 2025].

[8] C. Szegedy et al., "Going Deeper with Convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[9] Microsoft, "ONNX Runtime: Cross-Platform, High Performance Scoring Engine for ML Models," [Online]. Available: https://onnxruntime.ai [Accessed: Apr. 2025].

[10] Raspberry Pi Foundation, "Raspberry Pi 4 Model B," [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-4-model-b [Accessed: Apr. 2025].

[11] OpenCV.org, "OpenCV Library," [Online]. Available: https://opencv.org [Accessed: Apr. 2025].

[12] Twilio Inc., "Twilio Programmable Messaging API," [Online]. Available: https://www.twilio.com/messaging [Accessed: Apr. 2025].

[13] S. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[14] T. Y. Lin et al., "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2020.

[15] B. Zhou, S. Wang, X. Huang, and Y. Gao, "Accident Detection in Traffic Surveillance: A Deep Learning Perspective," *IEEE Access*, vol. 7, pp. 10139–10149, 2019.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2012.

[17] S. Liu, Y. Zhang, X. Liu, and C. Liu, "Deep Learning on Edge Devices: A Review," *Neural Computing and Applications*, vol. 34, pp. 4567–4593, 2022.

[18] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *arXiv preprint arXiv:1409.1556*, 2015.

[19] H. Wang, Q. Zhang, and Z. Li, "Real-Time Event Detection Using YOLOv5 on Embedded Systems," *International Conference on Embedded Vision Systems (ICEVS)*, pp. 81–86, 2021.

[20] D. J. Reber, "Designing Lightweight Edge AI Systems for Public Safety," *Journal of Embedded AI and IoT Applications*, vol. 6, no. 1, pp. 42–50, 2023.