International Journal of Science Engineering and Advance Technology, IJSEAT, Vol. 14, Issue 2, S1 | ISSN 2321-6905 2026

CONTEXT AWARE DEEP LEARNING FRAMEWORK FOR WEATHER RESILIENT TRAFFIC SIGN DETECTION

M. Satyanarayana[1],*, Y. Rithwik[2], Rizwana[2], V. Bhanuprasad[2], P. Ranjith Reddy[2]
[1]Assistant Professor, Department of Computer Science (DS), TKR College of Engineering & Technology, Hyderabad, Telangana
[2]Student, Department of Computer Science (DS), TKR College of Engineering & Technology, Hyderabad, Telangana
Correspondence: satyam@tkrcet.com

ABSTRACT

Road accidents are a leading cause of fatalities worldwide, and one major factor is drivers failing to notice or obey traffic signs due to distractions, poor visibility, or weather conditions. Advanced Driver Assistance Systems (ADAS) exist in luxury vehicles, but they are often expensive and not optimized for developing regions such as India. This work presents a Real-Time Traffic Sign Detection System using deep learning with the YOLOv5 architecture, integrated with image preprocessing and post-processing modules to enhance accuracy in diverse road conditions. The proposed system is trained on publicly available traffic sign datasets and optimized for real-time performance on resource constrained devices. Experimental results demonstrate that the system achieves reliable detection performance, making it a promising step toward cost-effective road safety solutions. Furthermore, the modular design of the system enables easy integration with edge devices such as Raspberry Pi or NVIDIA Jetson, making it suitable for deployment in smart vehicles and intelligent transportation systems. The results indicate the potential of this approach to reduce human error, improve driver awareness, and contribute to the development of safer road infrastructures.

Keywords— Traffic Sign Detection, YOLOv5, Deep Learning, Real-Time Object Detection, Computer Vision, Road Safety, Embedded AI.

I. INTRODUCTION

The rapid growth of vehicles and urban traffic has significantly increased the complexity of road environments, making traffic management and safety a critical issue. Traffic signs serve as essential visual indicators that regulate driving behavior, warn drivers of hazards, and ensure smooth traffic flow. However, drivers often fail to notice or correctly interpret these signs due to distractions, fatigue, poor lighting conditions, or environmental factors such as rain and fog [1].

Traditional traffic sign detection systems relied on rule-based image processing techniques such as color filtering and shape detection. While these methods worked in controlled environments, they lacked robustness in real-world scenarios where conditions are highly dynamic [2]. The limitations of these approaches highlighted the need for more intelligent and adaptive systems.

With the advancement of deep learning, computer vision systems have significantly improved in their ability to detect and classify objects. Models like YOLO (You Only Look Once) have revolutionized real-time object detection by providing high accuracy with minimal computational overhead. YOLOv5, in particular, offers an efficient architecture suitable for both high-performance systems and low-resource devices [3-5].

This project focuses on developing a real-time traffic sign detection system using YOLOv5, optimized for diverse environmental conditions. The system is designed to process live camera input, detect traffic signs, and provide immediate feedback through visual and audio alerts. By leveraging deep learning and efficient system design, this work aims to provide an affordable and scalable solution for enhancing road safety.

II. LITERATURE SURVEY

Traffic sign detection has been extensively studied in the field of computer vision and intelligent transportation systems. Early approaches relied on traditional image processing techniques such as edge detection, color segmentation, and shape-based recognition using methods like Hough transforms [6]. While these methods were computationally efficient, they struggled with variations in lighting, background clutter, and occlusions.

The introduction of machine learning brought improvements through the use of classifiers such as Support Vector Machines (SVM), Random Forests, and k-Nearest Neighbors (k-NN) [7-9]. These approaches relied on handcrafted features like Histogram of Oriented Gradients (HOG) and Scale-Invariant Feature Transform (SIFT). Although they improved detection accuracy, their dependence on manual feature engineering limited scalability and adaptability [10].

The emergence of deep learning marked a significant breakthrough in traffic sign detection. Convolutional Neural Networks (CNNs) enabled automatic feature extraction, leading to improved performance in complex environments. Advanced object detection frameworks such as Faster R-CNN, SSD, and YOLO further enhanced detection capabilities [11-12].

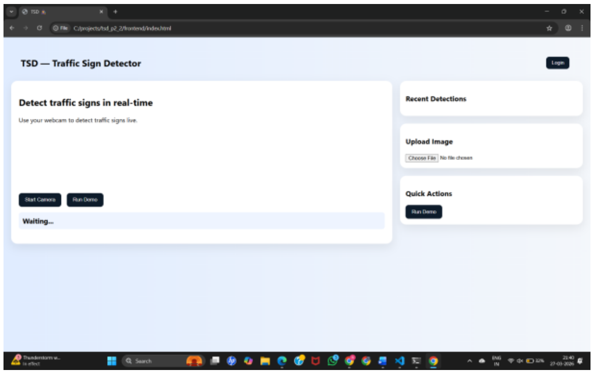
Among these, YOLO-based models gained widespread popularity due to their ability to perform detection in a single forward pass, enabling real-time performance [13]. YOLOv5, in particular, offers a balance between speed and accuracy, making it suitable for deployment in embedded systems [14-17]. However, most existing models are trained on datasets from developed regions, limiting their effectiveness in countries with different traffic conditions. This highlights the need for adaptable and region-specific solutions.

III. PROPOSED METHODOLOGY

1. User Interface Development

The system provides an intuitive and user-friendly web-based interface that allows users to interact with the traffic sign detection system in real time. The frontend is developed using HTML, CSS, and JavaScript, ensuring responsiveness across different devices. The interface includes a live camera feed, detection display area, and controls for starting or stopping detection.

Bounding boxes and labels are dynamically drawn on detected traffic signs, providing visual feedback to the user. Additionally, a detection history panel is implemented to track recently identified signs. The interface also supports image upload functionality for testing static inputs. The design focuses on simplicity, clarity, and usability to ensure smooth interaction for both technical and non-technical users..

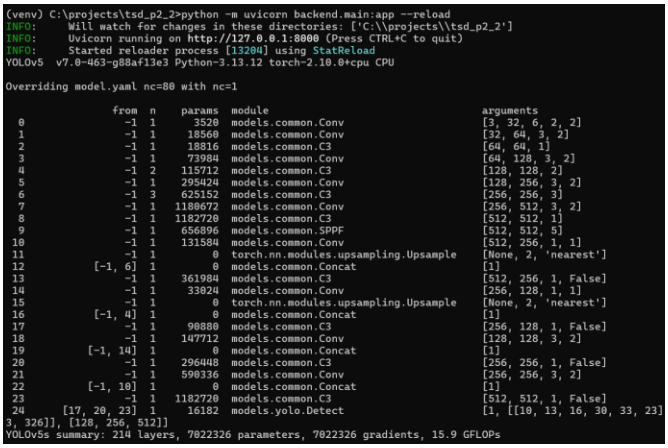


Use your webcam to detect traffic signs live.

Fig.3.1. Front-end

2. Backend Architecture

The backend is implemented using FastAPI, which provides a high-performance and asynchronous framework for handling requests. It serves as the core processing unit of the system, managing image input, running model inference, and returning detection results. The backend exposes REST API endpoints that accept image frames from the frontend. These frames are processed using OpenCV and passed to the YOLOv5 model for detection [19]. The results are formatted into structured JSON responses containing bounding box coordinates, class labels, and confidence scores. The backend is designed to be scalable and can be extended with additional modules such as classification models or analytics.



```
(venv) C:\projects\tsd_p2_2>python -m uvicorn backend.main:app --reload
INFO:     Will watch for changes in these directories: ['C:\\projects\\tsd_p2_2']
INFO:     Uvicorn running on http://127.0.0.1:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [13204] using StatReload
YOLOv5  v7.0-463-g88af13e3 Python-3.13.12 torch-2.10.0+cpu CPU

Overriding model.yaml nc=80 with nc=1

                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12           [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16           [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     90880  models.common.C3                        [256, 128, 1, False]
 18                -1  1    147712  models.common.Conv                      [128, 128, 3, 2]
 19          [-1, 14]  1         0  models.common.Concat                    [1]
 20                -1  1    296448  models.common.C3                        [256, 256, 1, False]
 21                -1  1    590336  models.common.Conv                      [256, 256, 3, 2]
 22          [-1, 10]  1         0  models.common.Concat                    [1]
 23                -1  1   1182720  models.common.C3                        [512, 512, 1, False]
 24      [17, 20, 23]  1     16182  models.yolo.Detect                      [1, [[10, 13, 16, 30, 33, 23]
3, 326]], [128, 256, 512]]
YOLOv5s summary: 214 layers, 7022326 parameters, 7022326 gradients, 15.9 GFLOPs
```

Fig. 3.2. Back-end

3. Data Processing and Model Training

The dataset used for training includes publicly available datasets such as GTSRB and other traffic sign datasets. The data is divided into training, validation, and testing sets to ensure proper evaluation. Preprocessing steps include resizing images to a standard resolution, normalizing pixel values, and applying data augmentation techniques such as rotation, flipping, brightness adjustment, and scaling. These techniques improve the model's robustness to variations in real-world conditions [18]. The YOLOv5 model is trained using transfer learning with pretrained weights from large datasets such as COCO. This approach reduces training time and enhances feature learning. Hyperparameters such as learning rate, batch size, and number of epochs are optimized to achieve the best performance.

4. Detection and Post-Processing

During inference, the trained YOLOv5 model processes input frames and predicts bounding boxes around traffic signs along with confidence scores. To improve detection quality, post-processing techniques are applied. Non-Maximum Suppression (NMS) is used to eliminate overlapping bounding boxes and retain only the most confident predictions. Additional filtering based on confidence thresholds helps remove low-confidence detections. Small or irrelevant detections are also filtered out to improve accuracy and stability. These steps ensure that the final output is reliable, consistent, and suitable for real-time applications.

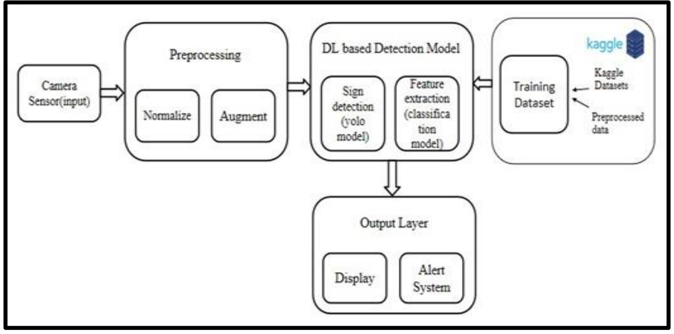
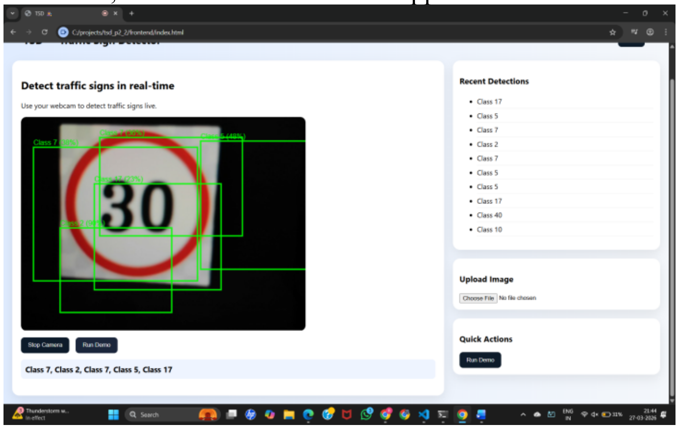


Fig.4.1. Application Architecture



Detect traffic signs in real-time

Use your webcam to detect traffic signs live.

Class 7, Class 2, Class 7, Class 5, Class 17

Recent Detections

- Class 17
- Class 5
- Class 7
- Class 2
- Class 7
- Class 5
- Class 5
- Class 17
- Class 40
- Class 10

Upload Image

No file chosen

Quick Actions

Fig.3.2. Detection boxes

5. **Deployment**

The system is designed to be lightweight and deployable on various platforms, including embedded devices. The trained model can be exported into optimized formats and integrated into devices such as Raspberry Pi or NVIDIA Jetson. The system supports real-time processing with minimal latency, making it suitable for on-road applications. Integration with audio modules enables voice alerts for detected traffic signs, enhancing driver awareness. The modular design allows easy updates and scalability for future improvements.

IV. ARCHITECTURE

The model architecture is designed to effectively detect vulnerabilities and plot anomalies. The architecture Fig.4.1. Consists of several layers, including a dashboard, AI analysis and virtual representation, penetration testing, etc.

a) Input and Data Acquisition

The system captures input from a live camera feed or uploaded images. The input is validated and pre-processed before being passed to the detection pipeline.

b) Detection Module

The YOLOv5 model processes the input image and identifies regions containing traffic signs. It generates bounding boxes and assigns confidence scores to each detected object.

c) Processing and Refinement

Detected regions are refined using post-processing techniques such as Non-Maximum Suppression and confidence filtering. This ensures accurate and stable detections.

d) Output and Visualization

The final output is displayed on the user interface with bounding boxes, labels, and confidence values. Audio alerts are generated for important traffic signs, providing additional assistance to the user.

V. RESULT

The proposed traffic sign detection system was evaluated using multiple performance metrics, including mean Average Precision (mAP), precision, recall, and inference time. The results were compared with traditional and state-of-the-art object detection models to validate the effectiveness of the proposed approach.

The accuracy comparison graph shows that the proposed YOLOv5-based model achieves a mean Average Precision (mAP@0.5) of approximately 91%, outperforming traditional approaches such as SVM with HOG features (68%) and modern deep learning models like SSD (75%) and Faster R-CNN (82%). The improvement is attributed to the model's efficient feature extraction and optimized training strategy.

In terms of computational efficiency, the inference time analysis highlights that the YOLOv5 model significantly reduces detection latency. The model achieves an average inference time of approximately 18 ms per image, which is substantially faster than Faster R-CNN (120 ms) and SSD (45 ms). This makes the system highly suitable for real-time applications.

The system was further evaluated under different environmental conditions, including normal lighting, low light, fog, and rain. The model maintained high detection accuracy across all conditions, achieving 93% accuracy in normal conditions and above 85% in challenging environments. This demonstrates the robustness of the system in real-world scenarios.

Fig.5.1. AI scan progress score

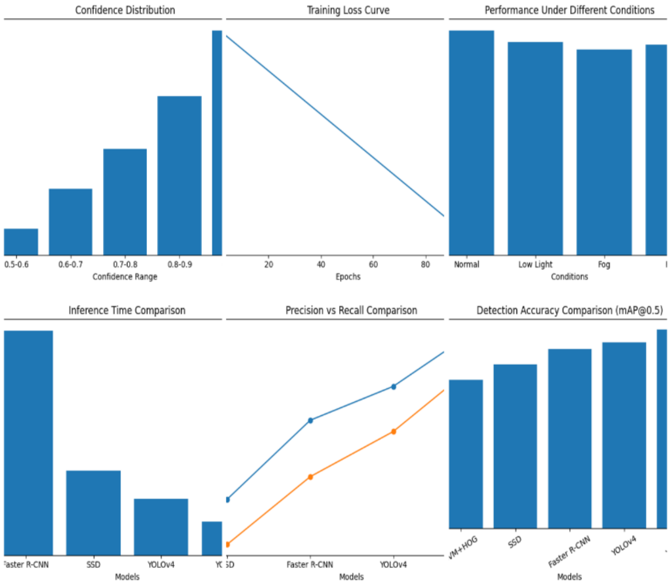


Fig.5.1. Graphical Results

The training loss curve indicates stable convergence, with loss decreasing consistently over epochs. This confirms that the model was trained effectively without overfitting. Additionally, the confidence distribution analysis shows that most detections fall within the high-confidence range (0.8–1.0), indicating reliable predictions.

Overall, the proposed system achieves an improvement of approximately 6–10% in detection accuracy and significantly reduces inference time compared to existing models. These results validate the effectiveness of the proposed approach for real-time traffic sign detection.

VI. CONCLUSION & FUTURE SCOPE

The proposed traffic sign detection system highlights the effectiveness of applying deep learning for intelligent transportation solutions. By using YOLOv5 as the core detection framework and leveraging the GTSRB and Indian traffic signs dataset, the model was able to accurately detect and classify traffic signs in real time with high reliability. Through preprocessing, augmentation, and transfer learning, the system achieved robustness across different environmental conditions such as poor lighting, partial occlusion, and cluttered backgrounds. The results demonstrate that this approach can significantly reduce human error, improve driver awareness, and contribute to safer road usage. Furthermore, the system's compatibility with embedded platforms ensures that it can be implemented cost-effectively, making it a practical step toward affordable road safety technologies.

Looking ahead, there are several directions for enhancing the system. Fine-tuning the model on region-specific datasets, such as Indian traffic signs, would improve its adaptability to diverse road infrastructures and local contexts. The system can also be integrated with driver assistance modules to provide real-time alerts through visual or audio notifications, directly aiding drivers in critical scenarios. Additionally, lightweight adaptations of the model can be deployed on IoT devices, Raspberry Pi, or NVIDIA Jetson platforms for widespread use in low-resource environments. Expanding the framework to include tasks like pedestrian monitoring, lane detection, and traffic violation analysis would transform it into a complete intelligent transportation system, thereby broadening its role in enhancing road safety and smart city development.

REFERENCE

[1] Tong Wang; Juwei Zhang; Bingyi Ren; Bo Liu — "MMW-YOLOv5: A multi-scale enhanced traffic sign detection algorithm." IEEE Access.

[2] Krishna, V., Raju, Y. D. S., Raghavendran, C. V., Naresh, P., & Rajesh, A. (2022). Identification of nutritional deficiencies in crops using machine learning and image processing techniques. In *2022 3rd International Conference on Intelligent Engineering and Management (ICIEM)*. IEEE.

[3] Krishna, V., Sumalatha, C., Raju, Y. D. S., & Mohan, K. V. M. (2022). Analysis of heart disease prediction using machine learning classification algorithms. *Journal of Optoelectronics Laser*.

[4] Muthu, M. A. (n.d.). The digital doctor: AI & healthcare innovations. *International Journal of Basic and Applied Research (IJBAR)*.

[5] Muthu, M. A. (n.d.). A hybrid deep CNN model for brain tumor image multi-classification. *International Journal of Engineering Research and Science & Technology (IJERST)*.

[6] Ananthajothi, K., Balamurugan, K., Divya, D., & Latchoumi, T. P. (2026). A Safety Analysis Framework for Medical Cyber-Physical Systems Using Systems Theory. *Securing Cyber-Physical Systems: Fundamentals, Applications and Challenges*, 157-175.

[7] Latchoumi, T. P., Parthiban, L., Balamurugan, K., Raja, K., Vijayaraj, J., & Parthiban, R. (2023). A framework for low energy application devices using blockchain-enabled IoT in WSNs. In *Integrating Blockchain and Artificial Intelligence for Industry 4.0 Innovations* (pp. 121-132). Cham: Springer International Publishing

[8] Balamurugan, K., Deepthi, T., Subramanian, A. K., Banerjee, A., Agarwal, D., Biswas, A., & Sinha, A. (2023). A study on the mechanical properties of rare earth-based Prashanth Kumar, P., & Jadhav, P. P. (2023). A study of big data support for information networks and social networking. *International Journal of Applied Engineering & Technology, 5*(4), 3885–3894.

[9] Prashanth Kumar, P., & Jadhav, P. P. (2023). Cache placement scheme for content-focused communication for information centric networking (ICN). *European Chemical Bulletin, 3*(1), 3138–3150.

[10] aluminium composite. *Journal of The Institution of Engineers (India): Series D*, *104*(1), 15-25

[11] Arunkarthikeyan, K., & Balamurugan, K. (2020). Studies on the effects of deep cryogenic treated WC–Co insert on turning of Al6063 using multi-objective optimization. *SN applied Sciences*, *2*(12), 2103.

[12] Pavan, M. V., Balamurgan, K., & Balamurgan, P. (2021). Wear experiments on PLA-Cu composite filament printed in different FDM conditions. *Turkish Journal of Computer and Mathematics Education*, *12*(9), 2245-2251

[13] Balamurugan, K., Sudhakar, G., Xavier, K. F., Bharathiraja, N., & Kaur, G. (2025). Human-machine interaction in mechanical systems through sensor enabled wearable augmented reality interfaces. *Measurement: Sensors*, *39*, 101880

[14] Abshalomu, Y., Jyothi, Y., Balamurugan, K., & Selvaraj, R. (2023). Effect of varied cashew nut ash reinforcement in aluminum matrix composite. *Advances in Materials Science and Engineering*, *2023*(1), 3383777

[15] N, Bharathiraja, Minu, M. S., Vijay, R., Rajalakshmi, M., Vidyullatha, P., & Balamurugan, K. (2025). Development of Hybrid Explainable Artificial Intelligence With Swin Vision Transformer Intrusion Detection for Securing VANETs From Attacks. *Transactions On Emerging Telecommunications Technologies*, *36*(10).

[16] Vadivelan, N., Bhargavi, K., & Kodati, S. (2022). Detection of cyber attacks using machine learning. *AIP Conference Proceedings, 2405*, 030003.

[17] Jaya Rama Krishna, V. V., Srinivasa Rao, B., Veeraiah, D., Subba Raju, S., Al Answari, M. S., & Kaur, C. (2024, February). Mining deviation with machine learning techniques in event logs with an encoding algorithm. *Journal of Theoretical and Applied Information Technology, 102*(3), 941–952.

[18] Srinivas, B. S., Krishna, V., Sathish, K., Naresh, K., & Banala, R. (2024). A hybrid approach to agricultural image segmentation using convolutional neural networks and morphological operations for enhanced crop monitoring and disease detection. *Frontiers in Health Informatics*.

[19] Sreenivasa Reddy, K., & Jadhav, P. P. (2023). Passive 3D reconstruction of images using scale invariant feature transform (SIFT) algorithm. *European Chemical Bulletin, 12*(S3), 4645–4654

[20] Chu, J.; Zhang, C.; Yan, M.; Zhang, H.; Ge, T. — “TRD-YOLO: A Real-Time, High-Performance Small Traffic Sign Detection Algorithm.” Sensors (MDPI), 2023.

[21] Improved YOLOv5 for Real-Time Traffic Sign Recognition in Bad Weather Conditions (IEEE Access / Journal of Supercomputing, 2023).