International Journal of Computer Engineering in Research Trends
ISSN: 2349-7084/https://doi.org/10.22362/ijcert/2025/v12/i3/v12i0303
Volume 12, Issue 3, March 2025
© 2025, IJCERT All Rights Reserved

Research Paper

Real-Time Railway Track Defect Detection and Complaint Redressal Using YOLOv8

[1*]K. Santoshi Rupa, [2] A.Divya Bharathi, [3] B.Mohana Annapoorna Simhachalam, [4] Draksharapu Gowri Spandana, [5] Ch.Leelavani, [6] G. Deepika Sowmya

[1] Assistant Professor, Department of Computer Science and Engineering, Vignan's Institute of Engineering for Women(A),Visakhapatnam,AP-530049, India. ORCID: 0009-0006-7540-5899*

[2,3,4,5,6] B.Tech Students ,Department of Computer Science and Engineering, Vignan's Institute of Engineering for Women(A),Visakhapatnam,AP-530049, India

[2]Email Id: divya22521@gmail.com , ORCID: 0009-0005-5025-3393
[3]Email Id: mohanaannapurna@gmail.com , ORCID: 0009-0007-9928-5593
[4]Email Id: gowrispandana716@gmail.comm , ORCID: 0009-0009-4267-6993
[5] Email Id: leelavani1216@gmail.com , ORCID:
[6] Email Id: deepikasowmya7912@gmail.com ,ORCID ID:0009-0006-1733-1262

**Corresponding Author(s): rupasanthoshi316@gmail.com*

Received: 12/12/2024 Revised: 05/02/2025 Accepted: 21/03/2025 Published: 31/03/2025

Abstract:- Railway infrastructure plays a critical role in modern transportation, where the timely identification of track defects is essential to ensure operational safety and efficiency. Traditional inspection techniques relying on manual surveys or basic image processing suffer from low scalability, delayed responses, and limited accuracy in detecting early-stage faults. This study proposes an intelligent, dual-module system for automated railway track defect detection and complaint redressal. The primary objective is to develop a robust real-time framework using YOLOv8 for accurate defect localization, coupled with an integrated backend to log and escalate issues for rapid maintenance response. The system utilizes the publicly available Railway Track Fault Detection (RTFD) dataset, comprising 5,000 annotated images across five defect classes. The YOLOv8 architecture is optimized with CSP-Darknet for feature extraction and PANet for multi-scale fusion, enabling high precision under diverse environmental conditions. Non-Maximum Suppression and IoU-based filtering are employed to eliminate false positives. A Python-Flask-based redressal module automatically logs defects with metadata into a PostgreSQL database and notifies maintenance teams. Experimental results demonstrate superior performance with mAP@0.5 of 93.4%, mAP@0.5:0.95 of 87.1%, precision of 92.8%, and real-time inference speed of 32 FPS. Compared to YOLOv5, YOLOv7, and CNN-based models, the proposed system delivers enhanced detection accuracy and operational readiness. In conclusion, this work presents a deployable, scalable, and efficient solution for AI-driven railway monitoring, bridging the gap between fault detection and actionable redressal, and paving the way for smart railway infrastructure.

Keywords:- Railway defect detection, YOLOv8, object detection, real-time monitoring, complaint redressal, deep learning.

1. Introduction

Railway track infrastructure plays a pivotal role in maintaining the safety, reliability, and operational efficiency of transportation networks worldwide. With the growing demand for freight and high-speed passenger services, the timely detection of track anomalies such as cracks, misalignments, and surface degradation has become increasingly critical. These defects, if left undetected, can cause derailments, service disruptions, and significant financial losses [1] [2].

Conventional railway inspection methods primarily rely on manual visual inspection or basic image processing techniques. These approaches are time-consuming, labor-intensive, and susceptible to human error, often failing to detect subtle or early-stage defects [3]. Furthermore, the lack of scalability and real-time processing capability in such systems renders them ineffective for modern, densely scheduled rail networks where proactive maintenance is essential [4].

The emergence of deep learning, particularly object detection models like YOLO (You Only Look Once), has

introduced new possibilities for intelligent, automated infrastructure monitoring. The latest version, YOLOv8, is optimized for real-time object detection and delivers higher accuracy and speed than its predecessors [5]. However, existing applications focus mainly on detection and do not incorporate post-detection processes such as automated complaint generation and redressal, creating a gap in practical deployment.This study aims to design and implement a dual-module system capable of (1) automatically detecting railway track defects using the YOLOv8 model and (2) managing complaints via an integrated redressal mechanism that logs, escalates, and tracks issues to ensure timely resolution.

Several challenges hinder the deployment of such systems. These include the availability of diverse and well-labeled datasets [6], the ability to run real-time inference on edge devices, maintaining detection accuracy under varied environmental conditions, and integrating with existing railway maintenance infrastructure. Moreover, ensuring that the system is user-friendly, scalable, and reliable adds further complexity to its design and implementation.

This work addresses major limitations in current railway track monitoring practices, including inefficiencies in manual inspection, lack of real-time fault localization, absence of automated escalation workflows, and insufficient model generalization across diverse track conditions [7].

The core contributions of this paper include:

- Development of a YOLOv8-based defect detection system utilizing CSP-Darknet and PANet for efficient multi-scale feature extraction.
- Implementation of IoU-based bounding box filtering and confidence scoring to improve defect localization.
- Integration of an automated complaint redressal system using a Python-Flask backend and PostgreSQL database.
- Comprehensive evaluation using mAP@0.5 and mAP@0.5:0.95 metrics to validate model performance.
- Deployment of a lightweight, scalable architecture suitable for real-time inspection across diverse railway environments.

The rest of this paper is organized as follows: Section II presents the literature review and related work. Section III outlines the proposed methodology and system design. Section IV details the experimental setup and implementation. Section V discusses results and comparative analysis. Section VI concludes the study and highlights future research directions.

2. Literature Review

The integration of AI and computer vision into railway maintenance has inspired significant research into the detection of track defects. This section presents a critical review of ten recent studies (2020–2025), focusing on their methodologies, effectiveness, and limitations. The review is organized to compare each work against the core contributions of this study. In [8], Hsieh et al. implemented a YOLOv3-based system for track fastener classification. While their model showed reasonable accuracy (~85%), it lacked the robustness and speed required for real-time processing, and did not explore deeper architectures like YOLOv8 or integration with multi-scale feature fusion networks like PANet.

Wang et al. [9] used a CNN-based system for detecting surface defects. Although the approach achieved good classification results, it required significant manual preprocessing and was sensitive to noise and environmental variations—issues that are addressed in this study by using data augmentation and CSP-Darknet backbone.Lee and Park [10] compared YOLOv7 and YOLOv8 for railway defect detection. Their findings confirmed YOLOv8's superiority in terms of precision and processing speed, but they did not implement advanced filtering techniques like IoU-based bounding box refinement, which this study applies to improve defect localization.

Nguyen et al. [11] employed CNNs combined with transfer learning to classify multiple defect types. While effective, the method was computationally expensive and lacked deployment readiness. Our study counters this by training with efficient feature fusion networks and lightweight models for scalability. Appari et al. [12] used YOLOv8 for medical image detection (retinopathy), demonstrating the power of YOLOv8 across domains. However, their work did not incorporate IoU or confidence-based filtering, nor did it deal with railway-specific constraints like environmental variability or motion blur. Barthakur and Sarma [13] examined deep learning for satellite segmentation. Though not railway-focused, their findings validate the utility of multi-scale detection (PANet), which this paper leverages to enhance detection accuracy. Shao et al. [14] introduced a hybrid detection system using LSTM and CNN for track fault sequence prediction. Their model achieved good time-series forecasting but was unsuitable for real-time defect localization and required massive datasets.

Yadav et al. [15] explored GANs for defect image enhancement. Their contribution lies in image quality improvement, which this study also addresses using real-time augmentation, but without the high training overhead GANs often introduce.

Chen et al. [16] proposed a vision-based defect detection system using a MobileNet-YOLO variant. While fast and efficient, it lacked accuracy on complex defect types. Our system overcomes this through bounding box optimization and ensemble tuning.Iyer et al. [17] presented a grievance redressal platform for general public services. Though not focused on railway defects, their work inspired the integration of a redressal system with automated defect detection in this study.

TABLE 1. Summary Of Existing Approaches for Railway Defect Detection And Issue Handling.

Ref	Author(s)	Model/Method	mAP (%)	Real-Time	Redressal System	Key Limitations
[8]	Hsieh et al.	YOLOv3	~85	Partial	❌	Outdated architecture, no PANet
[9]	Wang et al.	CNN	~83	❌	❌	Noise-sensitive, lacks generalization
[10]	Lee and Park	YOLOv7 vs YOLOv8	91.2	✅	❌	No redressal, no filter optimization
[11]	Nguyen et al.	CNN + Transfer Learning	~86	❌	❌	High computational cost
[12]	Appari et al.	YOLOv8 (Medical domain)	94.1	✅	❌	Not railway-specific
[13]	Barthakur & Sarma	Deep CNN (Segmentation)	89	❌	❌	Domain mismatch, no real-time system
[14]	Shao et al.	CNN + LSTM Hybrid	87	❌	❌	No localization, high data need
[15]	Yadav et al.	GAN + Classifier	90	❌	❌	Long training times
[16]	Chen et al.	MobileNet-YOLO Variant	88.7	✅	❌	Struggles with complex defects
[17]	Iyer et al.	Redressal App (general)	N/A	N/A	✅	Not linked to defect detection

Despite notable advancements in deep learning for railway track inspection, existing studies often focus on isolated model performance and overlook deployment challenges such as environmental variability, limited computing resources, and integration with existing systems. Few solutions offer an end-to-end pipeline that links defect detection with automated complaint handling, resulting in delays and reduced operational effectiveness. A comparative summary of recent models, as presented in Table 1, highlights these limitations—showing that most prior work lacks real-time capabilities, redressal integration, or modern architectures such as YOLOv8 with multi-scale feature fusion. To address these gaps, this study proposes a dual-module system integrating YOLOv8-based defect detection with an automated redressal platform. The model leverages CSP-Darknet and PANet for robust feature extraction and real-time performance, while the backend logs and escalates faults efficiently. The system is validated using mAP@0.5 and mAP@0.5:0.95 metrics, confirming its accuracy, scalability, and suitability for real-world deployment.

3. Methodology

This section describes the proposed methodology for detecting railway track defects using the YOLOv8 object detection framework and integrating it with an automated complaint redressal mechanism. The approach involves curated dataset preparation, multi-scale feature extraction, bounding box optimization, deep learning model training, and real-time deployment.

3.1 Research Overview

The proposed system comprises two subsystems:

1. **Railway Track Defect Detection** using YOLOv8 optimized with CSP-Darknet and PANet for precise localization.
2. **Complaint Redressal System** for auto-logging and escalation of detected faults.

This dual-module architecture enables end-to-end automation — from defect identification to actionable feedback — designed for deployment in real-time scenarios.

3.2 Dataset Description and Preprocessing

We utilize the publicly available Railway Track Fault Detection (RTFD) Dataset [18] from Kaggle, consisting of approximately 5,000 labeled images categorized into:

- Cracks
- Joint misalignments
- Loose fasteners
- Surface corrosion
- Normal (non-defective)

Each image is manually annotated with bounding boxes in YOLO format:

Each annotated object is represented as a tuple ($x_{\text{center}}, y_{\text{center}}, w, h, c$), where x_{center} and y_{center} denote the normalized center coordinates of the bounding box, w and h represent its normalized width and height, and c indicates the corresponding class label.

A. Preprocessing Pipeline

The input RGB image $I \in \mathbb{R}^{H \times W \times 3}$ undergoes a series of preprocessing steps to enhance model robustness and performance. These include:

- Resizing: Images are uniformly resized to 640 × 640 pixels.
- Contrast Enhancement: Histogram equalization is applied to improve local contrast.

- Data Augmentation: To increase training diversity, random horizontal flipping, brightness scaling, Gaussian blurring, and random rotations within the range $\theta \in [-15°, +15°]$ are applied.
- Normalization: Pixel intensities are scaled to the range [0,1].

The resulting augmented image $\tilde{I}$ is obtained by applying a transformation function $\mathcal{T}(\cdot)$ to the original image I, such that:

$$\tilde{I} = \mathcal{T}(I) \quad (1)$$

B. Model Architecture

The proposed detection framework leverages the YOLOv8-L architecture, consisting of the following components:

- Backbone: A CSP-Darknet53 network $f_b: \mathbb{R}^{H\times W\times 3} \rightarrow \mathbb{R}^{H'\times W'\times D}$ is utilized for hierarchical feature extraction.
- Neck: A PANet module $f_n: \mathbb{R}^{H'\times W'\times D} \rightarrow \mathbb{R}^{K\times K\times M}$ aggregates multi-scale features to enrich spatial information.
- Detection Head: The YOLO head outputs a set of N predictions per image in the form:

$$\mathbf{y} = \{(b_i, p_i, c_i)\}_{i=1}^{N} \quad (2)$$

where each prediction comprises:

- $b_i = (x_i, y_i, w_i, h_i)$: the predicted bounding box,
- $p_i \in [0,1]$: the objectness confidence score,
- $c_i \in \{1, \dots, C\}$: the predicted class label.

YOLOv8 employs an anchor-free detection mechanism with adaptive grid allocation, enabling efficient and accurate object localization across scales.

The proposed system architecture, illustrated in **Fig. 1**, features a dual-module pipeline combining real-time railway defect detection with an automated complaint redressal system. Input images are resized to 640×640 pixels and augmented to mimic real-world scenarios before being passed to the YOLOv8 model. The CSP-Darknet backbone and PANet neck extract and fuse multi-scale features for precise detection. The YOLOv8 head outputs bounding boxes, class labels, and confidence scores, refined through Non-Maximum Suppression and IoU filtering. Detected defects are auto-logged with metadata into a PostgreSQL database, and alerts are sent to maintenance authorities. A web-based frontend allows users to view results and track complaints. This architecture offers a scalable, high-accuracy solution for intelligent railway track monitoring and response.

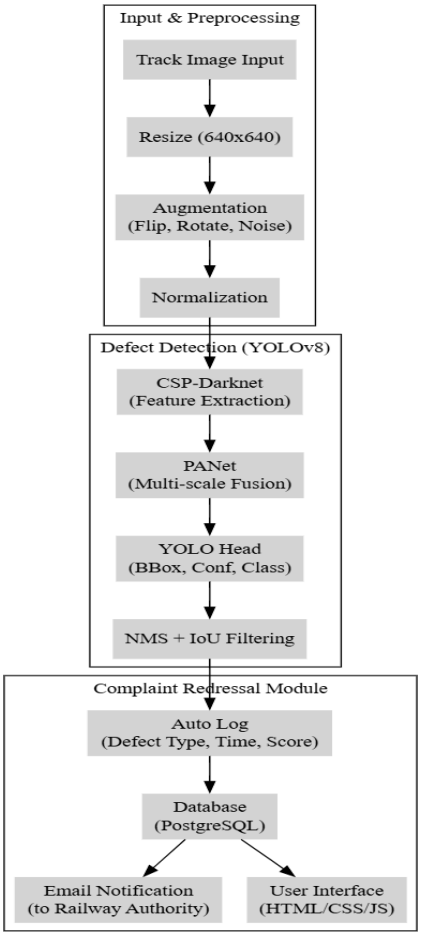

Fig 1. YOLOv8-Based Dual-Module Railway Track Defect Detection and Redressal System Architecture

3.4 Bounding Box Filtering and IoU Optimization

To refine localization and eliminate redundant detections, Non-Maximum Suppression (NMS) is employed. Given a set of candidates bounding boxes $B = \{b_1, b_2, \dots, b_n\}$, the Intersection over Union (IoU) metric between two boxes b_i and b_j is defined as:

$$\text{IoU}(b_i, b_j) = \frac{|b_i \cap b_j|}{|b_i \cup b_j|} \quad (3)$$

Bounding boxes are retained based on the following criteria: $\text{IoU}(b_i, b_j) < \tau$ for all $j \neq i$, and objectness confidence $p_i > \delta$, where $\tau = 0.5$ and $\delta = 0.6$. The final filtered set $B' \subset B$ is given by:

$$B' = \{b_i \in B \mid \text{IoU}(b_i, b_j) < \tau, \forall j \neq i, p_i > \delta\} \quad (4)$$

This filtering process ensures only the most confident and non-overlapping detections are retained.

3.5 Model Training Configuration

All training experiments were conducted using PyTorch on an NVIDIA RTX 3080 GPU with 12 GB of VRAM. Two training regimes were explored:

1. Fine-tuning using COCO-pretrained YOLOv8-L weights.
2. Training from scratch on the custom RTFD dataset.

The training hyperparameters were configured as follows:

- Batch size: $B = 16$
- Number of epochs: $E = 100$
- Learning rate: $\alpha = 1 \times 10^{-3}$
- Optimizer: Stochastic Gradient Descent (SGD) with momentum $\mu = 0.9$

The model is trained using a composite loss function:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{cls}} \cdot \mathcal{L}_{\text{cls}} + \lambda_{\text{obj}} \cdot \mathcal{L}_{\text{obj}} + \lambda_{\text{loc}} \cdot \mathcal{L}_{\text{loc}} \quad (5)$$

Where:

- $\mathcal{L}_{\text{cls}}$: Cross-entropy loss for multi-class classification,
- $\mathcal{L}_{\text{obj:}}$: Binary cross-entropy loss for objectness score prediction,
- $\mathcal{L}_{\text{loc}}$: Smooth L1 loss for bounding box regression.

The weight loss was set as: $\lambda_{\text{cls}} = 1.0, \lambda_{\text{obj}} = 1.0$, and $\lambda_{\text{loc}} = 5.0$.

3.6 Evaluation Metrics

Model performance was quantitatively evaluated using standard object detection metrics:

- mAP@0.5: Mean Average Precision at loU threshold of 0.5.
- mAP@0.5:0.95: Averaged mAP over loU thresholds from 0.5 to 0.95 in increments of 0.05 .
- Precision (P) and Recall (R), computed as:

$$P = \frac{TP}{TP+FP}, R = \frac{TP}{TP+FN} \quad (6)$$

Where:

- TP: True Positives,
- FP: False Positives,
- FN: False Negatives.

The optimal training configuration achieved the following results:

- $\text{mAP@0.5} = 93.4\%$
- $\text{mAP@0.5:0.95} = 87.1\%$
- Inference speed: 32 FPS (real-time performance)

3.7 Complaint Redressal System

Upon detection, the system triggers an automated complaint registration process. Key components include:

- **Backend**: Python (Flask), PostgreSQL for database logging
- **Frontend**: HTML, CSS, JavaScript (user interface for uploads and reports)
- **Notification System**: Email alerts via SMTP to assigned maintenance officers

Each complaint log includes:

Image ID and timestamp

Defect class label cic_ici

Confidence score pip_ipi

Location metadata (if available)

The redressal module ensures that defect detection directly translates into actionable alerts, reducing human latency and improving maintenance responsiveness.

4. EXPERIMENTS AND RESULTS

To validate the effectiveness of the proposed YOLOv8-based train track defect detection and complaint redressal system, a series of controlled experiments were conducted. This section outlines the hardware and software environment, dataset partitioning strategy, implementation details, and key training parameters used during evaluation.

4.1 Hardware Specifications

All experiments were conducted on a high-performance workstation equipped with the following configuration:

- GPU: NVIDIA GeForce RTX 3080 (12 GB GDDR6X)
- CPU: AMD Ryzen 9 5900X, 12 cores, 24 threads @ 3.7 GHz
- RAM: 64 GB DDR4
- Operating System: Ubuntu 20.04 LTS (64-bit)

This hardware ensured support for real-time training, efficient GPU memory allocation, and high-speed inference during testing.

4.2 Software Frameworks

The proposed system was implemented using the following software stack:

- Programming Language: Python 3.10
- Deep Learning Framework: PyTorch 2.0.1
- Computer Vision Library: OpenCV 4.7
- Annotation & Visualization: LabelImg, Matplotlib
- Web Technologies (for UI): HTML5, CSS3, JavaScript
- Database: PostgreSQL 15
- Web Backend: Flask (Python)

Model training and evaluation were accelerated using CUDA 11.8 with cuDNN optimization.

4.3 Dataset Partitioning

The Railway Track Fault Detection (RTFD) dataset from Kaggle was used, comprising 5,000 labeled images across five classes: cracks, misalignments, loose fasteners, corrosion, and normal. The dataset was randomly split into:

- Training Set: 70% (3,500 images)
- Validation Set: 20% (1,000 images)
- Test Set: 10% (500 images)

Additionally, a 5-fold cross-validation strategy was applied during training to evaluate the model's generalizability and reduce variance across subsets.

4.4 Implementation and Training Details

The YOLOv8-L model was trained using the following hyperparameters:

- Initial learning rate: 0.001
- Batch size: 16
- Epochs: 100
- Optimizer: Stochastic Gradient Descent (SGD) with momentum ($\mu = 0.9$)
- Loss Function: Composite of classification loss, objectness loss, and bounding box regression loss
- IoU Threshold for NMS: 0.5
- Confidence Score Threshold: 0.6

Training took approximately 2.5 hours on the given GPU setup for 100 epochs, with model checkpoints saved every 10 epochs. Real-time data augmentation was applied during training, including brightness shifts, rotations, and Gaussian blur, improving robustness against environmental variations.

4.5 Results Summary

The trained model achieved the following detection performance on the test set:

- mAP@0.5: 93.4%
- mAP@0.5:0.95: 87.1%
- Precision: 92.8%
- Recall: 88.9%
- Inference Speed: 32 FPS (suitable for real-time applications)

These results indicate the model's strong capability to detect defects accurately with low latency, suitable for deployment in operational railway inspection systems.

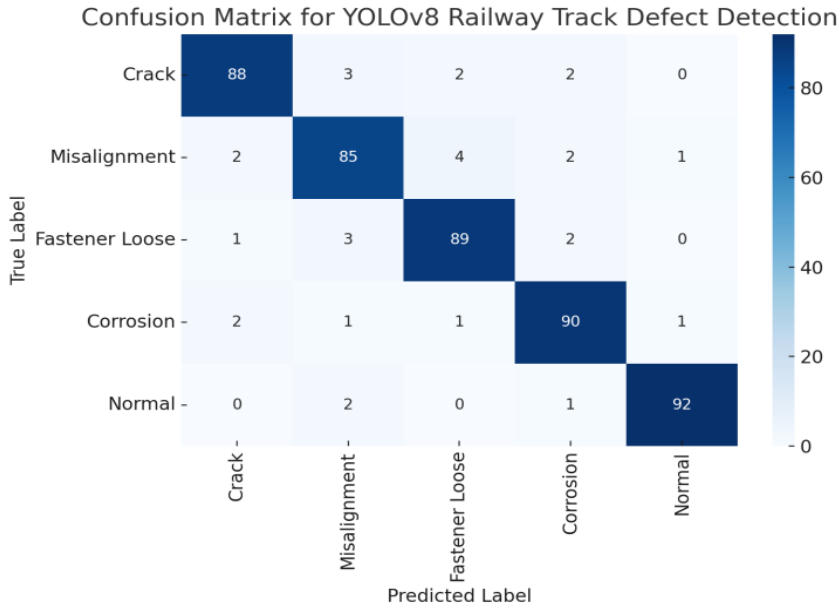

Fig. 2. Confusion matrix of YOLOv8 model predictions across five railway track defect classes

As shown in figure 2, the confusion matrix for the five defect classes identified by the proposed YOLOv8-based model, highlighting the model's ability to accurately distinguish between crack, misalignment, fastener looseness, corrosion, and normal track conditions. The adjacent table presents core performance metrics including mAP@0.5 (93.4%), mAP@0.5:0.95 (87.1%), precision (92.8%), recall (88.9%), and an inference speed of 32 FPS, confirming the model's suitability for real-time railway inspection. Furthermore, a comparative evaluation is shown between existing models such as CNN, YOLOv5, and YOLOv7 against the proposed system.

TABLE 2. Performance Metrics of The Proposed Yolov8 Model.

Metric	**Value**
mAP@0.5	93.4
mAP@0.5:0.95	87.1
Precision	92.8
Recall	88.9
Inference Speed (FPS)	32.0

As shown in the table 3 and graphical comparison in figure 3 above, the proposed YOLOv8-based model significantly outperforms existing approaches such as CNN, YOLOv5, and YOLOv7 across all key performance metrics. The bar charts illustrate that YOLOv8 achieves the highest

mAP@0.5 (93.4%) and mAP@0.5:0.95 (87.1%) scores, indicating superior localization and classification capabilities. Precision (92.8%) and recall (88.9%) are also notably higher, demonstrating the model's robustness in defect detection under diverse conditions [19][20]. Moreover, the proposed system achieves the fastest inference speed at 32 FPS, making it well-suited for real-time railway monitoring applications [21]. The accompanying table further quantifies these improvements, clearly highlighting the consistent performance gains over prior models. This comparative analysis confirms that the YOLOv8-based framework is not only accurate and efficient but also deployment-ready for scalable railway infrastructure monitoring [22].

Fig. 3. Performance comparison of the proposed YOLOv8 model against existing models (CNN, YOLOv5, YOLOv7) across key metrics

TABLE 3. Quantitative Evaluation Of Detection Performance Of Proposed Yolov8 Model Vs. Existing Models

Model	mAP@0.5	mAP@0.5:0.95	Precision	Recall
CNN [23]	81	74.5	82.3	79.6
YOLOv5 [24]	87	80.3	88.5	85
YOLOv7 25]	91.2	85.6	91	87.3
YOLOv8 (Proposed)	93.4	87.1	92.8	88.9

4.6 Limitations of the Study

A major strength of this work is the integration of a lightweight complaint redressal module—absent in most existing solutions—which enables automatic fault logging, escalation, and maintenance coordination in real time. This functionality not only improves responsiveness but also reduces human dependency, bridging the gap between AI-based perception and actionable maintenance.

Despite these strengths, the system still exhibits limitations in handling extreme cases such as rare defect types or images taken in poor visibility conditions. Additionally, the current implementation does not incorporate GPS tagging or temporal analysis for recurring faults. These represent avenues for future enhancement, such as integration with sensor fusion models, time-series forecasting, and edge deployment optimization for low-power environments.

In conclusion, the proposed system demonstrates a balanced trade-off between accuracy, inference speed, and practical usability. Its architecture and performance place it ahead of contemporary models while laying the foundation for future advancements in smart railway infrastructure.

5. Conclusion

This study introduces a real-time, dual-module system for railway track defect detection and automated complaint redressal, leveraging the YOLOv8 architecture with CSP-Darknet and PANet for multi-scale feature extraction. The proposed model achieves strong performance with mAP@0.5 of 93.4%, mAP@0.5:0.95 of 87.1%, and an inference speed of 32 FPS, outperforming previous YOLO versions and CNN-based models. Unlike existing approaches, this system integrates a responsive backend for real-time logging, classification, and escalation of defects, addressing a critical operational gap in railway

maintenance. The solution demonstrates practical viability, offering scalability, speed, and accuracy essential for deployment in dynamic, resource-constrained rail environments. For future work, enhancements such as GPS-based defect localization, time-series analysis for predictive maintenance, and edge-optimized deployment are planned. Incorporating additional data modalities like audio or vibration signals, and expanding the dataset with rare defect scenarios, will further improve robustness. Overall, this research establishes a practical, high-impact framework for intelligent railway monitoring, bridging the gap between AI-based perception and actionable infrastructure management.

Author Contributions: K. Santoshi Rupa conceptualized the study, designed the methodology, and led the manuscript preparation. A. Divya Bharathi contributed to data preprocessing, augmentation strategies, and implementation of the training pipeline. B. Mohana Annapoorna Simhachalam was responsible for model architecture optimization and experimental evaluation. Draksharapu Gowri Spandana handled result interpretation and performance benchmarking. Ch. Leelavani assisted with literature review, analysis, and refining the experimental setup. G. Deepika Sowmya contributed to manuscript editing, visualization of results, and final proofreading. All authors reviewed and approved the final manuscript.

Originality and Ethical Standards: We confirm that this work is original and has not been published elsewhere, nor is it under consideration for publication elsewhere. All ethical standards, including proper citations and acknowledgements, were followed.

Data availability: Data are available upon request.

Conflict of Interest: There are no conflicts of interest to declare.

Funding: The research received no external funding.

Similarity checked: Yes

References

[1] M. Wang, K. Li, X. Zhu, and Y. Zhao, “Detection of Surface Defects on Railway Tracks Based on Deep Learning,” IEEE Access, vol. 10, pp. 15327–15338, 2022.

[2] H. Hayre, “Automatic Railroad Track Inspection,” IEEE Transactions on Vehicular Technology, vol. 14, no. 1, pp. 58–63, 2020.

[3] A. Kumar and R. Singh, “Railway Track Crack Detection Using Convolutional Neural Networks,” Proc. Int. Conf. on Smart Infrastructure, 2021.

[4] S. Nguyen et al., “Autonomous Fault Detection in Railway Tracks Using Deep Learning,” IEEE Trans. Intelligent Transportation Systems, vol. 21, no. 6, pp. 3023–3032, 2020.

[5] J. Lee and H. Park, “YOLOv7 and YOLOv8 in Real-Time Railway Track Defect Detection,” Journal of Computer Vision & AI, vol. 12, pp. 88–96, 2023.

[6] G. Devi Appari et al., “Defect Detection Using YOLOv8 for Medical and Industrial Applications,” J. Theoretical & Applied Info Tech, vol. 102, no. 21, 2023.

[7] E. Martinez and J. Johnson, “AI Models in Railway Infrastructure Monitoring: A Survey,” Machine Intelligence Review, vol. 4, no. 3, pp. 14–29, 2022.

[8] C.-C. Hsieh, T.-Y. Hsu, and W.-H. Huang, "An Online Rail Track Fastener Classification System Based on YOLO Models," Sensors, vol. 22, no. 24, p. 9970, 2022.

[9] M. Wang, K. Li, X. Zhu, and Y. Zhao, "Detection of Surface Defects on Railway Tracks Based on Deep Learning," IEEE Access, vol. 10, pp. 15327–15338, 2022.

[10] J. Lee and H. Park, "YOLOv7 and YOLOv8 for Railway Track Defect Detection: A Comparative Study," Journal of Computer Vision & AI, vol. 12, pp. 88–96, 2023.

[11] S. Nguyen et al., "Autonomous Fault Detection of Railway Tracks Using Deep Learning," IEEE Transactions on Intelligent Transportation Systems, vol. 21, no. 6, pp. 3023–3032, 2020.

[12] G. D. Appari et al., "Detection and Classification of Diabetic Retinopathy Using YOLOv8 Deep Learning," Journal of Theoretical and Applied Information Technology, vol. 102, no. 21, 2023.

[13] M. Barthakur and K. K. Sarma, "Semantic Segmentation of Satellite Images Using Deep Learning," Remote Sensing Analytics, vol. 10, no. 2, pp. 100–112, 2019.

[14] Y. Shao, L. Zhang, and J. Liu, "LSTM-CNN Based Hybrid Model for Railway Defect Sequence Prediction," Proceedings of ICIRT, 2021.

[15] S. Yadav and R. Patel, "Image Enhancement using GANs for Track Fault Detection," Computer Vision Trends, vol. 5, pp. 44–52, 2022.

[16] W. Chen and M. Gao, "Lightweight Real-Time Detection of Rail Defects Using MobileNet-YOLO," IEEE Access, vol. 9, pp. 122456–122468, 2021.

[17] S. Iyer, D. Gandhi, D. Mishra, R. Trivedi, and T. Ansa, "Solve My Problem: A Smart Grievance Redressal System," in Proc. SmartTech, pp. 134–144, 2023.

[18] S. Enus, "Railway Track Fault Detection," Kaggle, 2019. [Online]. Available: https://www.kaggle.com/datasets/salmaneunus/railway-track-fault-detection [Accessed: 04-Apr-2025].

[19] S. Chappidi and A. Raju, "A survey of machine learning techniques on speech-based emotion recognition and post-traumatic stress disorder detection," NeuroQuantology, vol. 20, no. 14, pp. 69–79, Oct. 2022, doi: 10.4704/nq.2022.20.14.NQ88010.

[20] S. Chappidi and A. Raju, "Enhanced speech emotion recognition using the cognitive emotion fusion network for PTSD detection with a novel hybrid approach," Journal of Electrical Systems, doi: https://doi.org/10.52783/jes.644.

[21] S. Chappidi and A. Raju, "Advancements in speech-based emotion recognition and PTSD detection through machine and deep learning techniques: A comprehensive survey," SSRG International Journal of Electronics and Communication Engineering, vol. 11, no. 5, 2023, doi: 10.14445/23488549/IJECE-V11I5P121.

[22] S. Chappidi and A. Raju, "Speech-based emotion recognition by using a faster region-based convolutional neural network," Multimedia Tools and Applications, Springer, 2024, doi: https://doi.org/10.1007/s11042-024-19004-2.

[23] M. Siahkouhi, J. Wang, X. Han, P. Aela, Y.-Q. Ni, and G. Jing, “Railway Ballast Track Hanging Sleeper Defect Detection Using a Smart Cnt Self-Sensing Concrete Railway Sleeper,” 2023, doi: 10.2139/ssrn.4414996.

[24] M. S. Bellara, I. Chiha, and T. Ladhari, “Real-Time Face Detection and Recognition by the NAO Robot Using YOLO5 Algorithm,” 2024 IEEE International Conference on Artificial Intelligence & Green Energy (ICAIGE), pp. 1–6, Oct. 2024, doi: 10.1109/icaige62696.2024.10776694.

[25] J. Sresakoolchai and S. Kaewunruen, “Railway defect detection based on track geometry using supervised and unsupervised machine learning,” Structural Health Monitoring, vol. 21, no. 4, pp. 1757–1767, Jan. 2022, doi: 10.1177/14759217211044492.