International Journal of Advanced Multidisciplinary Research and Educational Development
Volume 2, Issue 2 | March – April 2026 | www.ijamred.com
ISSN: **3107-6513**

Railway Track Crack Detection Using Deep Learning

Swedha G[1], Velumani T[2]

[1,2]*Department of Computer Science, Rathinam College of Arts and Science (Autonomous), Coimbatore, Tamil Nadu, India.*

[2]*Corresponding Author : swedheesan@gmail.com*

Abstract: Railway safety is a critical concern in modern transportation systems, as undetected structural failures can lead to catastrophic accidents. Traditional manual inspection methods are time-consuming, labor-intensive, and prone to human error. This paper presents an intelligent and automated system for railway track crack detection using deep learning and computer vision techniques. The proposed system utilizes **Convolutional Neural Networks (CNN)** and **OpenCV** to identify defects in rail surfaces. By leveraging libraries such as **TensorFlow**, **Keras**, **Pillow**, **NumPy**, and **Matplotlib**, the system processes track images to distinguish between healthy and cracked rails. The final model is deployed via a **Streamlit** web application, providing a real-time, user-friendly interface for monitoring. Experimental results indicate that this deep learning approach enhances decision-making and provides a cost-effective solution for railway maintenance.

Keywords – Deep Learning, CNN, OpenCV, Railway Safety, Crack Detection, TensorFlow, Streamlit, Computer Vision1,

1. Introduction

The railway sector is a fundamental transportation pillar, supporting both global economies and public mobility. Maintaining the integrity of railway tracks is paramount, as factors like environmental stress and heavy loads lead to surface cracks. If left undetected, these cracks can compromise the structural stability of the rails.

Traditional methods of track analysis involve visual inspections by experts or expensive specialized machinery. While these methods are established, they are often inefficient for large-scale networks and lack the speed required for real-time safety updates. Consequently, there is a growing need for automated systems that can quickly analyze visual data and provide reliable results.

This paper proposes a crack detection system utilizing **CNN** and **OpenCV**. The system takes track images as input and classifies them into "Healthy" or "Cracked" categories. By reducing the dependency on manual labor, this framework helps rail authorities make informed maintenance decisions, improving overall safety and reducing operational risks

2. Related Works

Surface defect detection has been extensively studied using both traditional image processing and modern machine learning. Conventional systems categorized defects based on physical properties using manual feature extraction. However, these methods often struggled with varying lighting conditions and complex background noise.

With advancements in deep learning, researchers have explored **Convolutional Neural Networks (CNN)** to automate feature extraction. CNNs are capable of identifying complex patterns in images, such as minute fractures, with higher accuracy than manual methods. Recent studies have shown that deep learning models

can achieve accuracy levels above 90% in classification tasks. Despite these improvements, many systems lack an accessible interface for real-time field use. Our proposed system addresses this by integrating a robust CNN model with a **Streamlit**-based web platform.

3. System Design

The system design focuses on an efficient, accurate, and user-friendly platform for real-time rail analysis. It integrates image preprocessing, deep learning classification, and a web interface into a single framework.

3.1 Architectural Overview

The proposed system is divided into five distinct operational layers:

- **Data Acquisition Layer:** Collects high-resolution images of railway tracks using specialized cameras.
- **Image Processing Layer:** Uses **OpenCV** to clean and prepare images for the neural network.
- **Feature Extraction & Learning Layer:** The core **CNN** engine built with **TensorFlow** and **Keras**.
- **Prediction Layer:** Classifies the track as "Healthy" or "Cracked" in real-time.
- **Deployment Layer:** A **Streamlit** web interface that provides the end-user with immediate visual feedback.

3.2 Data Preprocessing Module (OpenCV & Pillow)

Raw images cannot be fed directly into a CNN due to noise and varying sizes. This module performs the following:

- **Resizing:** All images are standardized to a fixed resolution (e.g., 224×224 pixels) to match the CNN input layer.
- **Normalization:** Pixel values are scaled to a range of $[0, 1]$ using **NumPy** to speed up the model's convergence during training.
- **Noise Reduction:** OpenCV applies filters (such as Gaussian Blur) to remove environmental artifacts like dust or shadows on the track.

3.3 The CNN Model Architecture (TensorFlow & Keras)

The machine learning module is the most critical component. It utilizes a Deep Convolutional Neural Network (CNN) consisting of:

- **Convolutional Layers:** These layers use filters to detect edges, shapes, and textures characteristic of cracks.
- **Pooling Layers:** Reduces the spatial dimensions of the data, focusing only on the most significant features and preventing overfitting.
- **Fully Connected (Dense) Layers:** Interprets the features extracted by the previous layers to make the final classification.
- **Activation Functions:** ReLu (Rectified Linear Unit) is used in hidden layers, while Sigmoid or Softmax is used in the output layer for binary classification.

3.4 The Prediction and Output Pipeline

Once the model is trained and validated, it is integrated into a **Streamlit** dashboard. When a user uploads a railway image:

1. The system converts the image into a **NumPy** array.
2. The **Keras** model performs a forward pass to calculate the probability of a crack.
3. **Matplotlib** generates a visual report, often including a bounding box around the detected

defect or a heat map (Grad-CAM) showing where the model "looked" to find the crack.

4. The final result—**"CRACK DETECTED"** or **"TRACK SAFE"**—is displayed with a high confidence score.

3.5 Hardware and Software Integration

To ensure the system is scalable for real-world agricultural or industrial use, it is built on a modular stack:

- **Programming Language:** Python 3.x.
- **Deep Learning:** TensorFlow 2.0+ and Keras.
- **Data Handling:** Pandas and NumPy.
- **Interface:** Streamlit for cloud-based or local web deployment.

4. Outcomes and Future Work

4.1 Outcomes

The proposed system successfully provides an automated solution for railway monitoring.

- **Accuracy:** The CNN model achieves high reliability, outperforming standard image filters.
- **Efficiency:** Drastic reduction in time compared to manual visual inspections.
- **Accessibility:** The **Streamlit** interface allows for easy interaction without technical expertise.

4.2 Future Work

In the future, the system can be enhanced by:

- **IoT Integration:** Attaching high-speed cameras to train engines to collect data in real-time.
- **Deep Learning Optimization:** Implementing advanced architectures like YOLO for faster object localization.
- **Mobile Deployment:** Developing a mobile application for maintenance workers in remote areas.

---

5. Conclusion

The development and implementation of the "Railway Track Crack Detection System" mark a significant advancement in the integration of Deep Learning with public safety infrastructure. This research successfully demonstrated that a synergy between **Convolutional Neural Networks (CNN)** and **OpenCV** can provide a reliable, high-speed alternative to traditional manual track inspections, which are often limited by human fatigue and subjectivity.

5.1 Technical Summary

The core of this system, built using **TensorFlow** and **Keras**, proves that automated feature extraction is superior to manual heuristic-based methods for identifying surface fractures. By utilizing **OpenCV** for specialized image preprocessing—such as noise reduction and grayscale normalization—we ensured that the CNN model received high-quality, standardized data. This directly resulted in a more robust classification accuracy. The use of **NumPy** for numerical processing and **Matplotlib** for performance visualization allowed for a rigorous evaluation of the model's training efficiency, ensuring that the final deployment was both stable and precise.

5.2 Strategic Impact

The deployment of this framework through a **Streamlit** web application bridges the gap between complex

machine learning research and practical field application. It allows railway maintenance teams to upload track imagery and receive instantaneous results without requiring an in-depth understanding of neural network architecture. This accessibility is vital for scaling the technology across vast railway networks, where rapid decision-making can prevent derailments and save lives.

5.3 Final Remarks

In conclusion, the proposed system offers a cost-effective, scalable, and highly accurate solution for railway monitoring. While traditional methods will always have a place in the industry, the transition toward AI-driven diagnostics is inevitable for modernizing transportation safety. This project serves as a foundational blueprint for future intelligent transportation systems, proving that deep learning is not just a theoretical tool but a practical necessity for 21st-century infrastructure management.

References

1. **R. Umamageshwari et al.**, "Train Track Crack Classification using Convolutional Neural Network," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 2, no. 1, pp. 946–951, 2022.
2. **Y. Zhan, W. Li, H. Chen, and M. He**, "An Improved Method for Detecting Cracks in the CRTSII Type Ballastless Track Slab Using Enhanced YOLOv3," *IEEE Access*, vol. 9, pp. 20340–20352, 2021.
3. **A. Mohan and S. Poobal**, "Crack detection using image processing: A critical review and analysis," *Alexandria Engineering Journal*, vol. 57, no. 2, pp. 787–798, 2017.
4. **K. Akhila et al.**, "Deep Learning-Based Approach for Rail Surface Crack Detection using CNN," *Journal of Emerging Technologies and Innovative Research (JETIR)*, vol. 10, no. 3, pp. 450–458, 2023.
5. **Sharin N. and Anitha K.**, "Automated Railway Track Defect Detection System using YOLO Deep Learning Algorithm," *International Journal of Engineering Research & Technology (IJERT)*, vol. 14, no. 12, 2025.
6. **I. Goodfellow, Y. Bengio, and A. Courville**, *Deep Learning*, MIT Press, 2016. [Textbook for CNN and TensorFlow fundamentals].
7. **G. Bradski**, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000. [Primary reference for OpenCV].
8. **M. Abadi et al.**, "TensorFlow: A System for Large-Scale Machine Learning," *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, pp. 265–283, 2016.
9. **F. Chollet**, "Keras: Deep Learning for Humans," GitHub repository, 2015. [Reference for the Keras library].
10. **A. Grus**, *Data Science from Scratch: First Principles with Python*, O'Reilly Media, 2nd ed., 2019. [Reference for NumPy and Matplotlib integration].