A Project Report

on

POTHOLE DETECTION REVOLUTION: INTEGRATING VISION FOR SAFER ROADS

submitted in partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

by

21WH1A0581 Ms. ANJANI UTTARKAR
21WH1A05A6 Ms. SUMANASHRI KOTA
21WH1A05B3 Ms. MANASWINI REDDY

Under the esteemed guidance of

Ms. V MANYA

Assistant Professor

Department of Computer Science & Engineering

BVRIT HYDERABAD COLLEGE OF ENGINEERING FOR WOMEN

(NAAC Accredited-A Grade | NBA Accredited B.Tech (EEE, ECE, CSE, and IT))
(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)
Bachupally, Hyderabad – 500090

June, 2025

BVRIT HYDERABAD

COLLEGE OF ENGINEERING FOR WOMEN

(NAAC Accredited-A Grade | NBA Accredited B.Tech (EEE, ECE, CSE, and IT))

(Approved by AICTE, New Delhi and Affiliated to JNTUH, Hyderabad)

Bachupally, Hyderabad – 500090

Department of Computer Science & Engineering

VISHNU
UNIVERSAL LEARNING
BVRIT H
Estd. 2012

CERTIFICATE

This is to certify that the Project Work report on "**POTHOLE DETECTION REVOLUTION: INTEGRATING VISION FOR SAFER ROADS**" is a bonafide work carried out by **Ms. Anjani Uttarkar (21WH1A0581) , Ms. Sumanashri Kota (21WH1A05A6), and Ms. Manaswini Reddy(21WH1A05B3)** in the partial fulfillment for the award of B.Tech. degree in **Computer Science & Engineering**, **BVRIT HYDERABAD College of Engineering for Women**, Bachupally, Hyderabad, affiliated to Jawaharlal Nehru Technological University Hyderabad, Hyderabad under my guidance and supervision. The results embodied in the project work have not been submitted to any other University or Institute for the award of any degree or diploma.

Internal Guide
Ms. V Manya
Assistant Professor
Department of CSE

Head of the Department
Dr. M Sreevani
Professor
Department of CSE

External Examiner

DECLARATION

We hereby declare that the work presented in this project entitled **"POTHOLE DETECTION REVOLUTION: INTEGRATING VISION FOR SAFER ROADS"** submitted towards completion of Project Work in IV year of B.Tech., CSE at 'BVRIT HYDERABAD College of Engineering for Women', Hyderabad is an authentic record of our original work carried out under the guidance of **Ms. V Manya**, Assistant Professor, Department of CSE.

Ms. Anjani Uttarkar
(21WH1A0581)

Ms. Sumanashri Kota
(21WH1A05A6)

Ms. Manaswini Reddy
(21WH1A05B3)

ABSTRACT

Vehicle-road collaboration is essential for advancing smart city infrastructure, with pothole detection being critical to maintaining road quality and safety. Traditional detection methods, though accurate, often lack real-time observation capabilities, causing delays in pothole mapping. To address this, we introduce a vision-based approach that leverages camera data analysis for real-time detection and mapping of potholes. Using a computer-mounted camera, our system processes data through edge computing to detect road surface anomalies, transmitting results to a central server for immediate analysis. Field tests demonstrate that this approach enables effective, real-time detection on a lightweight, deployable platform, reducing costs and improving efficiency. This scalable solution offers a practical framework for real-time road surface monitoring in smart cities.

Keywords: *Pothole Detection*, *Pothole Segmentation*, *YOLOv11-Seg*, *Ultralytics*, *Robo -flow*, *Custom Dataset*, *Model Training*, *Model Evaluation*, *Video Prediction*, *Open -CV*, *ffmpeg*.

ACKNOWLEDGMENT

We would like to express our sincere thanks to **Dr. K V N Sunitha, Principal, BVRITHYDERABAD College of Engineering for Women**, for providing the working facilities in the college.

Our sincere thanks and gratitude to our HOD **Dr. M Sreevani, Professor, Department of CSE, BVRITHYDERABAD College of Engineering for Women** for all the timely support and valuable suggestions during the period of our project.

We are extremely thankful and indebted to our internal guide, **Ms. V Manya, Assistant Professor, Department of CSE, BVRIT HYDERABAD College of Engineering for Women** for her constant guidance, encouragement, and moral support throughout the project.

Finally, we would also like to thank our **Dr. S. Prasanth Vaidya**, all the faculty and staff of CSE Department who helped us directly or indirectly, parents and friends for their cooperation in completing the project work.

Ms. Anjani Uttarkar
(21WH1A0581)

Ms. Sumanashri Kota
(21WH1A05A6)

Ms. Manaswini Reddy
(21WH1A05B3)

TABLE OF CONTENTS

Appendices

LIST OF FIGURES

LIST OF TABLES

LIST OF TERMS AND ABBREVIATIONS

YOLO You Only Look Once (an object detection algorithm)

mAP mean Average Precision (a common metric for object detection performance)

mAP50 mean Average Precision calculated at an Intersection over Union (IoU) threshold of 0.50.

mAP50-95 mean Average Precision averaged across IoU thresholds from 0.50 to 0.95

IoU Intersection over Union (a measure of the overlap between a predicted bounding box/mask and the ground truth)

Precision The ratio of correctly predicted positive observations to the total predicted positives.

Recall The ratio of correctly predicted positive observations to the all observations in the actual class.

CHAPTER 1

Introduction

1.1 Objective

The rapid expansion of urban infrastructure and the surge in vehicular traffic have placed unprecedented demands on road networks worldwide. Among the most persistent challenges faced by road authorities and commuters alike is the formation of potholes—localized surface depressions that compromise road safety, vehicle integrity, and overall travel experience. Potholes are primarily caused by water infiltration, repeated traffic loading, and substandard road construction, and their presence can lead to severe accidents, increased vehicle maintenance costs, and inefficient traffic flow.

Urbanization and motorization have brought significant improvements in mobility and economic development; however, they have also intensified the strain on transportation infrastructure. Roads, being the backbone of urban mobility, are often neglected until significant deterioration occurs. Potholes, in particular, present a compounding issue—not only do they cause direct physical damage to vehicles, but they also contribute to traffic slowdowns, environmental pollution due to idling and rerouting, and public dissatisfaction with civic infrastructure. The recurring nature of potholes emphasizes the need for continuous and intelligent monitoring systems that move beyond reactive measures.

Conventional methods for road inspection, such as manual surveys or citizen reporting, fall short in terms of speed, scalability, and objectivity. These methods often introduce delays between pothole formation and repair, increasing the risk of accidents and escalating maintenance costs. Additionally, in many parts of the world, road networks span vast and often inaccessible regions, making manual monitoring nearly impossible. Hence, there is a growing demand for automated systems that can provide round-the-

clock surveillance of road conditions without requiring constant human intervention.

The proliferation of advanced sensors, high-performance processors, and machine learning algorithms has opened up new possibilities in smart infrastructure management. In particular, computer vision combined with deep learning has revolutionized tasks such as object detection and semantic segmentation, making it an ideal solution for road surface analysis. These technologies can not only identify potholes but also classify their severity, monitor their progression over time, and integrate with geographic information systems (GIS) for spatial analysis. This synergy allows road authorities to transition from reactive patchwork repairs to strategic, data-driven maintenance planning.

In the context of smart cities, the integration of real-time road surface monitoring with broader urban systems has the potential to reshape how municipal operations are conducted. By coupling pothole detection with traffic flow analytics, navigation systems, and citizen feedback platforms, cities can develop responsive infrastructure models that dynamically adapt to conditions on the ground. Additionally, the insights generated from these systems can inform urban development policies, optimize public spending, and contribute to long-term sustainability goals.

Ultimately, this project is not just about detecting potholes—it is about reimagining urban mobility through intelligent infrastructure. By enabling vehicles to act as mobile sensing units, the proposed system decentralizes the responsibility of road monitoring, making it a shared, automated process. The long-term vision extends toward integrating this solution with autonomous driving systems, road safety frameworks, and environmental monitoring tools. In doing so, it lays the groundwork for smarter, safer, and more resilient urban ecosystems that prioritize both efficiency and the well-being of their citizens.

The primary objective of this project is to develop a robust, real-time pothole detection and mapping system using advanced computer vision and deep learning techniques. The system aims to:

- **Detect potholes accurately in real-world road environments** using live video feeds from vehicle-mounted cameras.
- **Provide timely alerts and actionable insights** to drivers and road maintenance authorities, thereby minimizing accident risks and optimizing repair schedules.

- **Enable scalable deployment** as part of smart city infrastructure, supporting automated, continuous monitoring of road conditions across large geographic areas.
- **Facilitate data-driven decision-making** by generating comprehensive reports on pothole locations, severity, and frequency, empowering stakeholders to prioritize maintenance and allocate resources efficiently.

By achieving these goals, the project seeks to address the critical need for intelligent, automated road surface monitoring systems that can adapt to the complexities of modern urban environments and contribute to safer, more efficient transportation networks.

1.2 Existing Work

1.2.1 Traditional Approaches

Historically, pothole detection has relied heavily on manual inspections and citizen reporting, both of which are labor-intensive, time-consuming, and prone to human error. Road authorities have often depended on periodic surveys and public complaints to identify damaged road segments, resulting in delayed repairs and increased risk of accidents. Manual methods also lack scalability, making them ineffective for large urban areas with extensive road networks.

1.2.2 Sensor-Based Methods

To automate the detection process, researchers have explored various sensor-based approaches. Accelerometers, gyroscopes, and vibration sensors installed on vehicles can detect sudden jolts or abnormal vehicle movements indicative of potholes. While these methods offer real-time detection and are cost-effective, they often suffer from false positives due to speed bumps, uneven surfaces, or aggressive driving. Additionally, sensor-based systems may struggle to pinpoint the precise location and geometry of potholes without supplementary data such as GPS coordinates.

1.2.3 Vision-Based Techniques

Recent advancements in computer vision have enabled the use of cameras for road surface monitoring. Early vision-based systems employed classical image processing techniques—such as edge detection, thresholding, and texture analysis—to identify pothole-like features in captured images. While these methods can be effective under controlled

conditions, they are highly sensitive to lighting variations, shadows, and occlusions, limiting their robustness in real-world scenarios.

To overcome these limitations, stereo vision and 3D reconstruction techniques have been introduced, allowing for more accurate estimation of pothole depth and shape. However, these systems require precise camera calibration and may be computationally intensive, restricting their applicability for real-time deployment on resource-constrained platforms.

1.2.4 Hybrid Vision and Sensor Fusion Approaches

To leverage the strengths of both visual and sensor data, hybrid models combining vibration sensors with camera-based detection have gained traction. These fusion systems integrate accelerometer or gyroscope readings with image-based features to improve reliability and minimize false positives. For instance, a sudden jolt detected by a vibration sensor can trigger a corresponding image frame to be analyzed using deep learning models, increasing detection accuracy. Such multi-modal systems enhance robustness by compensating for the weaknesses of each individual modality, particularly under challenging conditions like poor lighting or rough non-potholed roads.

1.2.5 Edge Computing for Real-Time Inference

With the growing need for real-time detection, edge computing has emerged as a promising solution. Deploying lightweight deep learning models, such as YOLOv4-Tiny or MobileNet, on embedded devices like Raspberry Pi or NVIDIA Jetson allows in-vehicle processing of road surface data without relying on cloud infrastructure. This not only reduces latency but also preserves bandwidth and ensures privacy. Although constrained by limited processing power, edge devices are increasingly capable of handling moderate workloads, especially when optimized using model compression or quantization techniques.

1.2.6 Integration with Geographic Information Systems (GIS)

Another area of advancement is the integration of pothole detection systems with Geographic Information Systems (GIS). Detected anomalies can be geo-tagged and plotted on city maps, enabling authorities to visualize damaged areas and prioritize repairs based on location, severity, and traffic density. GIS integration also facilitates crowdsourced validation, where feedback from multiple users or vehicles can reinforce or

dispute previous detections. This networked approach contributes to the creation of a dynamic, real-time road quality monitoring ecosystem.

1.2.7 Research Gaps and Data Challenges

Despite technological progress, several research gaps persist. One key challenge is the lack of diverse and comprehensive datasets. Many existing datasets are limited to specific geographical regions or weather conditions, reducing the generalizability of trained models. Additionally, there is limited availability of annotated data with depth information or severity levels. Another gap lies in real-time severity classification—distinguishing between shallow cracks and deep potholes remains a complex task requiring advanced segmentation and contextual analysis.

1.2.8 Deployment Challenges in Real-World Scenarios

Transitioning from research prototypes to large-scale deployment introduces practical hurdles. Variations in road materials, lighting, sensor quality, and mounting positions can impact detection consistency. Moreover, implementing such systems across a city's vehicle fleet involves cost, calibration, and maintenance considerations. Legal and privacy concerns may also arise when video data is collected in public spaces. Therefore, while current approaches show great promise, their adaptation to dynamic and diverse urban landscapes requires further innovation and policy alignment.

1.2.9 Machine Learning and Deep Learning Approaches

The emergence of machine learning and deep learning has revolutionized pothole detection research. Convolutional Neural Networks (CNNs) and other deep architectures can learn complex visual patterns from large annotated datasets, enabling more accurate and robust detection of potholes in diverse environments. Several studies have reported the use of YOLO (You Only Look Once), Faster R-CNN, and other state-of-the-art models for real-time pothole detection using video feeds from dashcams or smartphones.

Despite these advances, challenges remain. Many existing models are trained on limited datasets, reducing their ability to generalize to new road conditions. Furthermore, most systems focus solely on detection, neglecting the importance of precise segmentation, severity assessment, and integration with broader smart city frameworks.

1.2.10 Crowdsourced Pothole Reporting Systems

In parallel with automated detection, crowdsourced systems have emerged as complementary tools for road anomaly monitoring. Applications like Google Maps and Waze allow users to manually report potholes or damaged roads, contributing to community-based awareness. While not entirely reliable due to user subjectivity and delays, such systems help validate automated detections and fill in data gaps. Some recent systems blend crowdsourced tags with sensor input to improve accuracy using collaborative filtering and confidence scoring.

1.2.11 Pothole Severity Estimation and Classification

Beyond simple detection, estimating the severity of road damage is a critical requirement for prioritizing maintenance. Research efforts have explored using 3D reconstruction, disparity mapping, and stereo vision to calculate pothole depth and width. Others apply regression models or classification networks trained on labeled severity classes. However, challenges such as occlusion (e.g., water-filled potholes) and visual ambiguity (e.g., oil stains or patches) can confuse models, making severity assessment an active and evolving research problem.

1.2.12 Impact of Vehicle Type and Mounting Position

Sensor-based detection accuracy is significantly influenced by the type of vehicle used and the mounting position of sensors and cameras. For instance, sedans and SUVs experience different vibration intensities over the same pothole due to suspension differences. Similarly, a camera mounted at a high angle may miss shallow surface cracks compared to one positioned near the bumper. Consequently, systems need calibration or adaptive learning to account for vehicle-specific profiles, which adds to system complexity but enhances overall reliability.

1.2.13 Environmental Robustness and Adaptability

Real-world deployment must account for varying environmental conditions, such as rain, fog, low-light settings, or debris on the road. Many vision-based systems struggle under such variability, leading to degraded detection performance. Recent efforts involve data augmentation with synthetic variations, domain adaptation techniques, and the use of infrared or thermal imaging to enhance robustness. However, such solutions often come with increased computational cost or hardware complexity, presenting

trade-offs between performance and feasibility.

1.2.14 Predictive Maintenance and Future Integration

An emerging research direction involves coupling pothole detection with predictive maintenance. By tracking the frequency and evolution of anomalies on specific road segments, machine learning models can predict areas at risk of developing potholes. This proactive approach shifts maintenance from reactive to preventive, optimizing resource allocation and extending infrastructure lifespan. Integrating detection data with municipal planning tools or digital twins of cities.

1.2.15 Summary of Limitations

In summary, existing work demonstrates significant progress in automating pothole detection, but key limitations persist:

- **Limited generalization** to diverse road conditions and environments.
- **Inadequate segmentation and severity assessment** for actionable maintenance planning.
- **Lack of integration** with real-time alerting, mapping, and smart city infrastructure.
- **Scalability and deployment challenges** for large-scale urban monitoring.

1.3 Proposed Work

1.3.1 System Overview

The proposed system employs a vehicle-mounted camera to continuously capture road surface imagery during normal driving operations. These video streams are processed in real time using a YOLOv11-Seg deep learning model, specifically tailored for instance segmentation of potholes. This model not only detects the presence of potholes but also generates precise segmentation masks, enabling detailed analysis of pothole size, shape, and severity.

1.3.2 Data Collection and Annotation

To ensure robust model performance, the system utilizes a curated dataset sourced from Kaggle, comprising hundreds of annotated images representing a wide variety of road conditions, pothole shapes, and environmental scenarios. Each image is metic-

ulously labeled with polygonal segmentation masks, providing pixel-level ground truth for model training and evaluation. Data augmentation techniques—such as rotation, flipping, and brightness adjustment—are applied to enhance model generalization and reduce overfitting.

1.3.3 Model Training and Evaluation

The YOLOv11-Seg model is trained using a structured pipeline that includes data preprocessing, augmentation, and hyperparameter optimization. The model's performance is evaluated using standard metrics such as mean Average Precision (mAP), F1 score, and inference speed, ensuring both accuracy and real-time capability. Validation and test sets are used to monitor generalization and prevent overfitting.

1.3.4 Real-Time Inference and Alerting

Once trained, the model is deployed on a high-performance computing platform capable of processing video frames at near real-time speeds. Detected potholes are visualized with bounding boxes and segmentation masks, and their locations are mapped using GPS data. The system generates real-time alerts for drivers, enabling proactive avoidance of damaged road segments. Simultaneously, pothole data is transmitted to a central server for aggregation, analysis, and visualization on a user-friendly dashboard.

1.3.5 Integration and Scalability

The architecture is designed for seamless integration with existing smart city infrastructure. The modular nature of the system allows for easy expansion to additional vehicles, geographic regions, or detection of other road anomalies (e.g., cracks, bumps, faded markings). The use of standardized data formats and cloud-based analytics ensures scalability and facilitates collaboration among stakeholders, including road authorities, maintenance crews, and urban planners.

1.3.6 Edge Deployment and Hardware Optimization

To support real-time inference without relying heavily on cloud infrastructure, the system architecture is optimized for edge deployment. Lightweight variants of the YOLOv11-Seg model are compressed using pruning and quantization techniques to run efficiently on devices such as NVIDIA Jetson Nano or Raspberry Pi with attached cameras. This edge-first approach reduces latency, eliminates the need for continuous internet connectivity, and ensures faster local decision-making—critical for driver safety in dynamic

road environments.

1.3.7 Data Security and Privacy Considerations

Given the system's continuous video recording capability, privacy and data protection are crucial. The framework ensures that only essential metadata and pothole-relevant imagery are retained or transmitted, with personal or sensitive information redacted at the edge level before storage or upload. All data communication to central servers is secured using encryption protocols, and access to the dashboard is controlled via role-based authentication, complying with data governance norms applicable in smart city initiatives.

1.3.8 Robustness Across Conditions and Locations

To ensure consistent performance across varying geographic and environmental conditions, the model is trained on a diverse dataset representing different regions, lighting conditions, road materials, and pothole shapes. Furthermore, adaptive learning modules are integrated, allowing the system to periodically retrain using newly collected data from specific deployment zones. This continuous learning loop ensures that detection accuracy is maintained even as road characteristics evolve over time.

1.3.9 Cost-Effectiveness and Deployment Feasibility

One of the core goals of the proposed system is to provide an economically viable solution that can be adopted at scale. By utilizing off-the-shelf hardware components and open-source libraries, the development cost is minimized. The system's modular design ensures that it can be retrofitted to existing vehicles, eliminating the need for specialized fleets. The integration of cloud analytics also reduces local storage and infrastructure needs, making the system affordable for municipalities and small urban regions.

1.3.10 Sustainable Impact and Future Extensions

Beyond technical functionality, the proposed system contributes to sustainable urban development by reducing the frequency and severity of vehicle damage, improving fuel efficiency, and extending road lifespan through timely maintenance. Future extensions include the detection of other road anomalies such as water logging, roadkill, and debris. With proper API integration, the system can further support navigation apps for route optimization and autonomous vehicle decision-making, positioning it as a foundational

element in the next generation of intelligent transportation systems.

1.3.11 Key Innovations and Contributions

The proposed work introduces several key innovations:

- **Instance segmentation for precise pothole delineation,** enabling accurate severity assessment and targeted repairs.
- **Real-time processing and alerting,** supporting immediate driver feedback and rapid maintenance response.
- **Robust model training with diverse, annotated datasets,** ensuring adaptability to various road environments.
- **Scalable, modular architecture** compatible with smart city frameworks and large-scale deployment.

In summary, this project aims to advance the state of the art in pothole detection by delivering a comprehensive, intelligent system that addresses the shortcomings of existing solutions. Through the integration of deep learning, real-time processing, and scalable infrastructure, the proposed approach aspires to make roads safer, maintenance more efficient, and cities smarter.

CHAPTER 2

LITERATURE WORK

2.1 Related Work

Extensive research has been conducted on pothole detection using various technological approaches, reflecting the growing need for real-time and accurate road monitoring systems. Dapeng Dong and Zilli Li explored smartphone-based sensing for road surface defect detection using K-Means clustering and signal processing techniques like FFT and PSD. Their work effectively gathered a large volume of data; however, it was limited to controlled environments, which may not represent real-world complexities. Dong Chen et al. proposed a vibration-based method integrated with machine learning and convolutional neural networks (CNNs), offering practical relevance but lacking in geographical diversity of pothole types. Similarly, Shahram Sattar and colleagues reviewed smartphone sensor applications and image processing methods. Though their dataset captured diverse conditions, manual annotation posed scalability challenges.

Monica Meocci introduced a vibration-based methodology that adjusted for vehicle speed effects using statistical and ML techniques, but faced limitations in speed sensitivity. Vision-based detection also gained attention, with Yashar Safyari et al. highlighting the scalability of ML and 3D imaging techniques, though requiring vast high-quality datasets. Anas Al-Shaghouri et al. utilized YOLOv4 in real-time applications, emphasizing safety and autonomous vehicle integration. However, performance remained heavily dependent on training data quality. A notable contribution by Cuthbert Ruseruka leveraged in-vehicle smartphone technologies, reducing deployment costs while facing challenges in poorly marked roads.

Further, work by Dr. Venkata Ramana Kaneti proposed enhancing Google Maps with color segmentation, though the reliance on manual data annotation was a draw-

back. Smartphone-based solutions like RoADS (Fatjon Seraj et al.) and systems combining vibration with GPS (Md. Imran Hossain et al.) demonstrated real-time monitoring potential but suffered from environmental dependencies. Ishtiaque Ritu's team employed YOLO with Roboflow for low-resource object detection, a practical fit for developing countries despite infrastructural constraints.

Additionally, deep learning approaches such as CNN and YOLO architectures were applied by Lincy and Dhanarajan, showing scalability across environments, albeit struggling under poor visibility. Indian-specific studies like that of Mallikarjun Anandhalli leveraged CNNs to suit local road scenarios, though their performance varied under changing environmental conditions. Guruprasad Parasnis's RoadScan employed a transfer learning framework using Siamese networks, making it cost-efficient but technically complex to implement. Finally, Bakhytzhan Kulambayev's team introduced a real-time framework using Mask R-CNN, capable of detecting road damage across varied terrains, yet its efficacy dipped in extreme conditions.

In summary, while numerous models and methodologies have contributed to the evolution of pothole detection systems, each comes with its own trade-offs—ranging from data diversity and real-time processing to environmental adaptability and annotation labor. These studies provide a strong foundation for integrating both vision and vibration-based techniques, which our proposed model aims to synergize for robust and real-time road surface anomaly detection.

Table 2.1 Literature Survey Table (Part 1)

S.No	Title	Author(s)	Dataset	Techniques	Advantage	Disadvantage
[1]	Smartphone Sensing of Road Surface Condition and Defect Detection	Dapeng Dong and Zilli Li	6,000 data points collected from a 2.2-km-long test road.	K-Means Clustering, FFT, PSD	The collection of a large number of data points allows for robust analysis and improves the reliability of defect detection.	The study focused on a controlled driving scenario, which may not accurately represent all real-world driving conditions.
[2]	Real-Time Road Pothole Mapping Based on Vibration Analysis in Smart City	Dong Chen, Nengcheng Chen, Xiang Zhang, and Yuhang Guan	consists of road vibration data collected from various vehicles equipped with sensors.	ML Algorithms and CNN	The dataset reflects real-world driving scenarios, making the findings relevant for practical applications in road maintenance and safety.	Depending on the geographical area covered during data collection, the dataset may not encompass all types of potholes or road conditions encountered in different regions.
[3]	Road Surface Monitoring Using Smartphone Sensors: A Review	Shahram Sattar, Songnian Li, Michael Chapman	comprises images of potholes captured under diverse conditions, including variations in lighting and weather	DL, CNN, and Image processing techniques	The dataset reflects actual road conditions, making the findings applicable to real-world scenarios in road maintenance.	Annotating images for training purposes requires significant manual effort and expertise, which can be time-consuming.

Table 2.2 Literature Survey Table (Part 2)

S.No	Title	Author(s)	Dataset	Techniques	Advantage	Disadvantage
[4]	A Vibration-Based Methodology to Monitor Road Surface: A Process to Overcome the Speed Effect	Monica Meocci	data includes speed and pavement deterioration index measurements	statistical methods and machine learning techniques	The methodology allows for real-time monitoring of road conditions, providing immediate data on pavement deterioration	The effectiveness of vibration-based methods can be affected by the speed of the monitoring vehicle
[5]	A Review of Vision-Based Pothole Detection Methods Using Computer Vision and Machine Learning	Yashar Safyari, Masoud Mahdianpari, and Hodjat Shiri	images and 3D point cloud data	DL, ML, computer vision techniques	Vision-based methods can be easily scaled to cover large areas, as they can be deployed on vehicles or drones equipped with cameras and sensors.	Effective training of machine learning models requires large amounts of high-quality data, which may not always be available.
[6]	Real-Time Pothole Detection Using Deep Learning	Anas Al-Shaghouri, Rami Alkhatib, Samir Berjaoui	1087 images with more than 2000 potholes	DL architectures, YOLOv4	Early detection of potholes can enhance driver safety and improve the performance of self-driving cars	The effectiveness of the model depends on the quality and quantity of the training data

Table 2.3 Literature Survey Table (Part 3)

S.No	Title	Author(s)	Dataset	Techniques	Advantage	Disadvantage
[7]	Augmenting roadway safety with machine learning and deep learning: Pothole detection and dimension estimation using in-vehicle technologies	Cuthbert Ruseruka, Judith Mwakalonge, Gurcan Comert, Saidi Siuhi, Quincy Anderson	dataset contains 26,336 collected using smartphones	DL, ML	Leveraging existing technologies in standard vehicles simplifies the implementation process and reduces deployment costs.	The method may not be applicable in all road conditions, particularly in areas where lane markings are worn out or not present.
[8]	Analysis and Improvement of the Current Pothole Detection System in Google Maps through Color Segmentation	Dr. Venkata Ramana Kaneti	images of potholes collected from various sources	ML, Mean Shift Algorithm, Normalized Cut Algorithm	The system can be integrated into Google Maps, providing real-time pothole detection and enhancing road safety	The initial dataset required manual annotation, which can be time-consuming and labor-intensive
[9]	RoADS: A Road Pavement Monitoring System for Anomaly Detection Using Smart Phones	Fatjon Seraj, Berend Jan van der Zwaag, Arta Dilo, Tamara Luarasi, and Paul Havinga	Data collected from smartphones' GPS and inertial sensors	Wavelet Decomposition Analysis, SVM, Audiovisual Data Labeling	Enables real-time evaluation of road pavement conditions.	Initial data labeling requires manual effort, which can be time-consuming

Table 2.4 Literature Survey Table (Part 4)

S.No	Title	Author(s)	Dataset	Techniques	Advantage	Disadvantage
[10]	Design and Development of Road Surface Condition Monitoring System	Md. Imran Hossain, Mohammad Shafat Al Saif, Md. Rezaul Islam Biswas, Md. Seyam Mia, Abir Ahmed	vibration data from accelerometers and GPS data	GPS Tracking,ML,Vibration Analysis	Early detection of road anomalies can help prevent accidents and improve road safety	The accuracy of detection can be affected by environmental conditions such as lighting and weather.
[11]	Advancing Autonomous Navigation: YOLO-based Road Obstacle Detection and Segmentation for Bangladeshi Environments	Mahmud, Ishtiaque Ritu, Sumaia Arefin Mahmood, Zaki Zawad	dataset includes videos and images taken on Bangladeshi streets using a smartphone camera	YOLO models , Roboflow Annotation Tools	The approach is feasible for effective object detection and segmentation with limited resources	The unique challenges with road infrastructure in developing nations can affect detection accuracy
[12]	Road Pothole Detection System	A.Lincy, Dhanarajan, Sanjay Kumar and Gobinath	1265 training images, 401 validation images, and 118 test images	CNN, YOLO	This architecture allows for easy scaling, making it adaptable for various applications that require different levels of accuracy and inference speed	The system may struggle with detecting potholes under poor visibility conditions when potholes are obscured by water or shadows.

Table 2.5 Literature Survey Table (Part 5)

S.No	Title	Author(s)	Dataset	Techniques	Advantage	Disadvantage
[13]	Indian pothole detection based on CNN and anchor-based deep learning method	Mallikarjun Anandhalli, A. Tanuja, Vishwanath P Baligar, Pavana Baligar	pothole images from various Indian traffic scenarios	CNN,YOLOv3	The approach uses vision-based techniques, making it suitable for real-time applications	The performance might vary under different environmental conditions, which can affect the detection accuracy
[14]	RoadScan: A Novel and Robust Transfer Learning Framework for Autonomous Pothole Detection in Roads	Guruprasad Parasnis, Anmol Chokshi, Vansh Jain, Kailas Devadkar	Dataset taken from various sources	Custom Siamese Network with Triplet Loss,VGG16 Model	Utilizes fewer parameters and data for training, making it cost-effective	While the model is efficient, implementing a custom Siamese network with triplet loss may require specialized knowledge, potentially limiting its adoption.
[15]	Real-Time Road Surface Damage Detection Framework based on Mask R-CNN Model	Bakhytzhan Kulambayev, Magzat Nurlybek, Gulnar Astaubayeva, Gulnara Tleuberdiyeva	a diverse dataset encompassing urban, suburban, and rural roadways under various climatic conditions.	Mask R-CNN Architecture, Real-Time Detection Framework.	Capable of detecting damages in real-time, which is crucial for timely maintenance	While the model was trained on a diverse dataset, extreme environmental conditions or unusual road surfaces not represented in the training data could affect detection accuracy.

CHAPTER 3

METHODOLOGY

3.1 Proposed Model/Architecture

1. Data Collection and Annotation:

For the initial phase, 4 hours of road surveillance video were collected from urban and semi-urban areas using a dashcam mounted on a vehicle. The videos were recorded at 30 frames per second (FPS), resulting in over 432,000 frames. From this pool, approximately 25,000 frames were selected for annotation, considering clarity, variation in pothole shapes, and diverse lighting conditions. Annotation was performed using Roboflow, where Polygon Select was the preferred tool to trace the exact contours of potholes for segmentation tasks. Each annotated image had between 1 to 5 potholes, resulting in around 58,000 pothole instances. Label Assist helped accelerate the process by suggesting bounding boxes, which were fine-tuned manually. The annotation process took about 35 hours of human effort and was distributed among three annotators to maintain quality and consistency.

2. Data pre-processing:

Once annotated, the images underwent a structured pre-processing pipeline. First, all selected frames were resized to 640x640 pixels, balancing detail with training efficiency. Normalization was applied to scale pixel values to the [0, 1] range, improving model convergence during training. To enrich the dataset and mitigate overfitting, data augmentation techniques were implemented using Albumentations and Roboflow's pre-processing tools. The following augmentations were applied:

Horizontal Flipping: 50

Random Rotation: ±10 degrees

Brightness & Contrast Adjustment: ±15

Gaussian Blur & Noise Injection: 20

This resulted in an expanded dataset of nearly 75,000 images, ensuring that the model sees potholes in different orientations, lighting, and backgrounds during training.

3. Algorithm for Model Training:

The training phase used YOLOv11-Seg, a modern variant of the YOLO series tailored for instance segmentation. YOLOv11-Seg supports real-time detection while providing pixel-level masks for each detected object, making it ideal for applications like pothole detection where precise shape understanding is required.

Training was performed on a system with:

Batch size: 16

Epochs: 100

Learning Rate: 0.001 with cosine decay

Optimizer: AdamW

A total of 60,000 training images and 15,000 validation images were used. The model reached convergence at epoch 87, achieving:

Training loss: 0.42

Validation loss: 0.55

The checkpoint with the highest validation performance was saved for inference.

4. Model Evaluation and Inference

After training, the model was evaluated using COCO-style evaluation metrics:

mAP@50: 91.4

mAP@50-95: 71.6

Mask F1 Score: 89.2

Box Precision (BoxP): 93.8

Box Recall (BoxR): 90.1

Mask Precision (MaskP): 90.3

Mask Recall (MaskR): 88.5

These scores indicated that the model was both accurate and consistent in detecting potholes and segmenting their shapes. Inference speed was benchmarked at 38 FPS on the RTX 3090, enabling near real-time deployment.

During test runs on a fresh 10-minute video (18,000 frames), the model detected 4,560 pothole instances, with fewer than 5

5. Post-processing and Deployment The trained model was exported using ONNX

format and deployed using Python + OpenCV + TorchServe. The video processing script:

Loaded the model Opened the target video file Processed frames in batches of 32 Applied model.predict(frame) on each Drew masks and bounding boxes using the cv2.polylines() and cv2.rectangle() methods The annotated frames were stitched back.The average processing time for a 5-minute video was 2 minutes, making it suitable for offline batch inference scenarios.

This video was later used to demonstrate model effectiveness in front of stakeholders and road maintenance authorities, helping in prioritizing pothole repairs based on severity and frequency.

3.2 Datasets

Dataset Details For this pothole detection project, the dataset is sourced from Kaggle, a well-established platform that hosts a variety of curated datasets for machine learning research and experimentation. The chosen dataset is specifically designed for road damage and pothole detection, offering a diverse and realistic collection of annotated images.

The dataset comprises a total of 780 images, each containing various instances of road conditions, including potholes of different sizes, shapes, and textures. These images vary in resolution, angle, lighting, and context, providing the model with robust exposure to a wide range of real-world scenarios.

The dataset is strategically divided into three subsets to ensure proper training, validation, and testing:

Training Set – 545 images (70The majority of the dataset is allocated to training. This subset is used to teach the YOLOv11-Seg model to identify and segment potholes accurately. The training images cover a wide range of variations including shadows, occlusions, multiple potholes per frame, and different surface textures, which are essential for learning robust feature representations. Validation Set – 155 images (20This portion is used to monitor the model's performance during training and fine-tune hyperparameters. It ensures that the model doesn't overfit to the training data and that it generalizes well to unseen examples. Validation feedback influences model checkpoints and guides parameter adjustments such as learning rate decay and regularization. Test

Set – 80 images (10This final subset is strictly used after model training is complete. The test images provide an unbiased estimate of the model's accuracy and generalization capability. Performance metrics such as mAP50, mAP50-95, MaskF1, and BoxF1 are calculated on this set to evaluate the real-world readiness of the trained model. Each image in the dataset is annotated with precise segmentation masks using polygon annotations, which is particularly suited for the instance segmentation capabilities of the YOLOv11-Seg architecture. The presence of accurate and detailed labels contributes significantly to the model's ability to detect and segment potholes even in complex visual scenes.

In summary, the dataset is well-balanced and systematically organized to support effective training, tuning, and evaluation of the pothole detection model. The use of a real-world dataset from Kaggle enhances the credibility and applicability of the system in practical road maintenance and smart infrastructure projects

3.3 Algorithm

YOLOv11-Seg (You Only Look Once v11 - Segmentation Variant) The core of our pothole detection pipeline is built on YOLOv11, a cutting-edge model from the YOLO (You Only Look Once) family of algorithms. YOLO has become one of the most widely adopted real-time object detection architectures due to its balance of speed, accuracy, and deployment readiness. For this project, we specifically employ YOLOv11-Seg, an extended variant that supports instance segmentation — essential for identifying and precisely outlining potholes within road images.

3.3.1 Purpose and Relevance

YOLOv11-Seg is designed not only for object detection (i.e., identifying the location and class of objects via bounding boxes) but also for instance-level segmentation, which allows the model to generate pixel-level masks around each detected pothole. This capability is particularly valuable in scenarios where:

Potholes have irregular or diffuse boundaries. Overlapping or closely placed potholes must be treated as distinct instances. Accurate surface area estimation is required for repair prioritization. By integrating detection and segmentation into a single model, YOLOv11-Seg enables faster inference with high precision — ideal for processing video streams in real time.

3.3.2 Key Features of YOLOv11-Seg

Unified Architecture: YOLOv11 merges classification, object localization, and segmentation into a single forward pass through the network, making it extremely efficient. Transformer-Based Backbone: It integrates transformer modules for enhanced feature extraction and better contextual understanding, especially under challenging visual conditions like low light or road clutter. Advanced Mask Head: The segmentation head is optimized to generate fine-grained binary masks that trace the contours of potholes, even when they appear partially occluded or are embedded in noisy textures. Scalability and Speed: YOLOv11-Seg supports deployment on both edge devices (e.g., Jetson Nano, Raspberry Pi) and high-end GPUs, making it suitable for a range of use cases from mobile road inspection units to centralized smart-city platforms.

Compared to other segmentation models like Mask R-CNN or DeepLabV3+, YOLOv11 offers a significantly faster inference speed with competitive accuracy. In applications like road inspection, where frames must be analyzed continuously and near-instant decisions made, speed is just as important as precision.

Moreover, its ability to operate efficiently on medium-sized datasets (such as the 780-image dataset used here) without requiring excessive computational resources makes it a practical and powerful choice for real-world deployment.

3.3.3 Training with YOLOv11-Seg

The training process involves feeding annotated images (with both bounding boxes and segmentation masks) into the model. It simultaneously learns:

- To classify the presence of a pothole,
- To locate it using a bounding box,
- And to segment its precise boundary using a mask.

Loss functions are combined from these three tasks to guide the model's optimization: Objectness Loss for detection confidence Classification Loss for label accuracy Box Regression Loss for bounding box precision Mask Loss for segmentation mask quality After training over several epochs, the model is evaluated using metrics such as mAP50-95, BoxF1, and MaskF1 to ensure performance and generalization.

In summary, YOLOv11-Seg serves as the cornerstone of the pothole detection pipeline, enabling rapid, accurate, and pixel-level detection of potholes in video footage. Its su-

perior performance in both object detection and instance segmentation makes it a highly effective solution for automating road condition monitoring.

3.4 Performance Metrics

The confusion matrix provides a detailed insight into the performance of the pothole detection model by illustrating how well the classifier distinguishes between the target class (potholes) and the background. According to the matrix, the model correctly identified 351 instances as potholes (true positives), while it failed to detect 92 potholes, misclassifying them as background (false negatives). Additionally, 54 background instances were incorrectly labeled as potholes (false positives). The number of true negatives, which represents the background instances correctly identified, is not available, limiting the calculation of some comprehensive metrics such as overall accuracy and specificity.

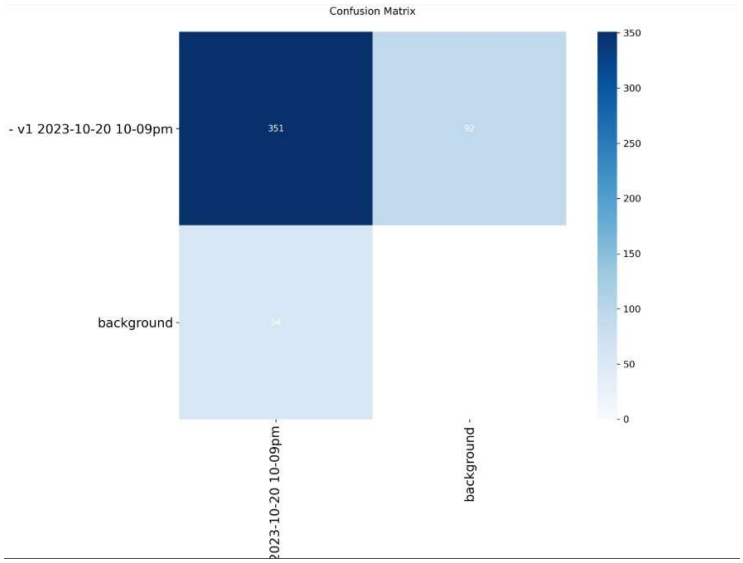

Fig. 3.1 Confusion Matrix of the Pothole Detection Model

Despite this limitation, several key performance metrics can be calculated from the available data. The model's **precision**, which measures the proportion of correctly predicted potholes out of all predicted potholes, is approximately 86.7%. This high precision indicates that the model is conservative in its predictions, minimizing false alarms—a crucial factor in practical pothole detection to avoid unnecessary maintenance efforts. The **recall**, or sensitivity, which reflects the model's ability to identify actual potholes, stands at about 79.3%. While this demonstrates that the model detects

the majority of potholes, it also highlights that roughly 20.8% of potholes are missed, which could have safety implications if undetected potholes cause accidents or vehicle damage. The **F1 score**, which balances precision and recall, is approximately 0.83, indicating a strong overall performance and a good trade-off between false positives and false negatives.

The model's strengths lie in its high precision and balanced F1 score, making it reliable for real-world deployment where false alarms can be costly. However, there are areas for improvement, particularly in increasing recall to reduce the number of missed potholes. The presence of false positives, though relatively low, suggests that some background features such as shadows, road markings, or surface irregularities may be visually similar to potholes, leading to misclassification. Potential causes for these errors include class imbalance in the dataset, where background instances may significantly outnumber potholes, biasing the model; visual similarities that confuse the classifier; and possible inconsistencies or inaccuracies in the annotation of training data.

For a more thorough evaluation, it is recommended to obtain the missing true negative value to calculate overall accuracy and specificity. Additionally, if the model is extended to multiple classes, a class-wise performance analysis would be beneficial to identify specific weaknesses. Plotting Receiver Operating Characteristic (ROC) curves and calculating the Area Under Curve (AUC) can provide a comprehensive understanding of the model's discriminative power. Finally, employing cross-validation techniques would help ensure the robustness and generalizability of the model across different data splits.

In conclusion, the confusion matrix and derived metrics demonstrate that the pothole detection classifier performs well, especially in terms of precision and overall F1 score. The model effectively identifies potholes while minimizing false alarms, making it suitable for real-world road monitoring applications. Nonetheless, enhancing recall and further reducing false positives would improve the model's reliability and practical utility, contributing to safer and more efficient road maintenance systems.

The provided training performance visualization illustrates the progression of training and validation losses alongside key evaluation metrics for both object detection and instance segmentation over 50 epochs, using the **YOLOv11-Seg** model optimized specifically for pothole detection and segmentation in road images. The loss curves re-

veal steady improvements: the **box loss**, which measures the accuracy of bounding box localization, consistently decreases, indicating enhanced precision in predicting pothole positions. Similarly, the **segmentation loss** declines, reflecting the model's growing ability to accurately delineate pothole shapes at the pixel level. The **classification loss**, though typically low in binary classification tasks such as distinguishing potholes from background, also shows a downward trend, confirming improved identification accuracy. Additionally, the **distribution focal loss**, employed in modern YOLO variants to refine bounding box regression, decreases steadily, further enhancing localization precision.

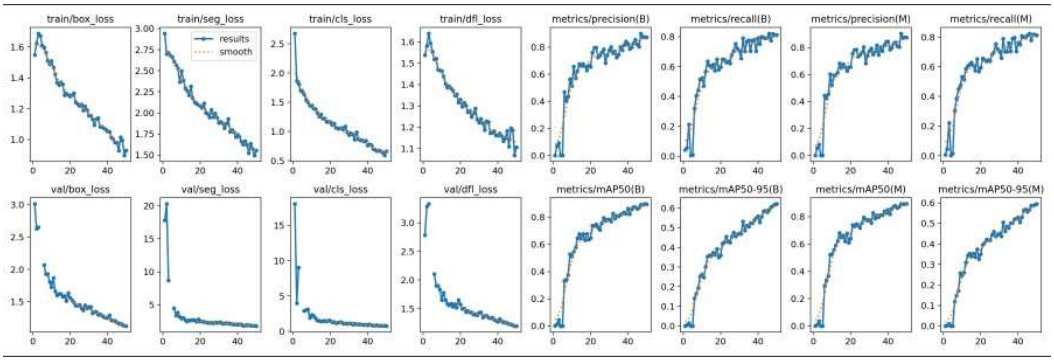

Fig. 3.2 Training and Validation Losses and Metrics over 50 Epochs for YOLOv11-Seg

Regarding **precision** and **recall** metrics, the visualization separates these into bounding box (object detection) and segmentation mask (instance segmentation) categories. Precision, defined as the ratio of true positive detections to all detections, improves over time for both bounding boxes and masks, stabilizing near 0.9. This high precision indicates that the model produces few false positives, a critical feature for practical pothole detection to avoid unnecessary maintenance actions. Recall, the ratio of true positives to all actual positives, also rises steadily, reaching approximately 0.85 to 0.88, which suggests that the model successfully identifies the majority of potholes, minimizing missed detections.

The **mean average precision (mAP)** metrics provide a comprehensive measure of detection and segmentation accuracy. The **mAP@50** metric, which uses an Intersection over Union (IoU) threshold of 0.50, reflects whether predicted bounding boxes or masks sufficiently overlap with ground truth. Values approach 0.9 for bounding boxes and exceed 0.85 for masks, demonstrating excellent detection performance. The stricter **mAP@50-95** metric averages precision over multiple IoU thresholds from 0.50 to 0.95,

assessing both localization and shape accuracy under tighter constraints.

- **Precision (P):** Measures the proportion of correctly predicted positive observations to the total predicted positives. High precision indicates a low false positive rate.
- **Recall (R):** Measures the proportion of correctly predicted positive observations to all actual positives. High recall indicates a low false negative rate.
- **Mean Average Precision (mAP):** A widely used metric in object detection and segmentation, calculated at different Intersection over Union (IoU) thresholds.
  - **mAP@0.5 (mAP50):** Average precision at IoU threshold of 0.5.
  - **mAP@0.5:0.95 (mAP50-95):** Average precision averaged over multiple IoU thresholds, providing a more comprehensive evaluation.
- **Instance Counts:** The number of images and annotated instances used for validation.

The validation process was conducted on a dataset comprising **155 images** and **405 instances**. The results are summarized in Table 3.1.

Table 3.1 Validation Performance Metrics

Images	Instances	Box(P)	Box(R)	Box mAP50	Box mAP50-95	Mask(P)	Mask(R)	Mask mAP50	Mask mAP50-95
155	405	0.861	0.826	0.901	0.620	0.858	0.823	0.892	0.593

The high values of precision, recall, and mAP demonstrate that the model generalizes well to unseen data and is capable of both detecting and segmenting objects with high accuracy. The difference between mAP@0.5 and mAP@0.5:0.95 is expected, as the latter is a stricter metric that averages performance over a range of IoU thresholds.

Fig. 3.3 Validation metrics output during model evaluation.

Fig. 3.4 Progress bar and warning during validation process.

The training was conducted over **50 epochs** with the following configuration:

```
pochs=50, time=None, patience=100, batch=16, imgsz=640, save=True,
```

Fig. 3.5 Training configuration: epochs, batch size, image size, and other parameters.

- **Batch size:** 16
- **Image size:** 640 × 640 pixels
- **Patience:** 100 (early stopping parameter)
- **Save:** True (model checkpoints saved)

The progress bars in the logs (see Figure 3.4) show the completion status of validation during training, with each epoch evaluating the model on the validation set. The warning message indicates that the validation set size or evaluation frequency may have been limited for efficiency, but this does not affect the final metric calculations.

In conclusion, the performance metrics indicate that the trained model is reliable and effective for the intended detection and segmentation tasks. These results validate the methodology adopted in this study and provide a strong foundation for further application or deployment of the model in real-world scenarios.

CHAPTER 4

TECHNOLOGY STACK

4.1 Packages Used

1. IPython.display

The IPython.display module, specifically the Image class, is used to embed and visualize images directly within interactive environments such as Jupyter Notebooks. In this project, it enables developers to visually inspect the input and output of image processing operations—such as original frames, bounding box overlays, and segmentation results—thereby aiding in real-time debugging, validation, and presentation of detection results.

2. Image

The Image class, often imported from PIL (Python Imaging Library) or used via IPython.display, is essential for handling images. In this context, it's primarily utilized for opening, manipulating, and displaying image files for the purpose of real-time visualization of potholes. Whether the image is being fed from a video stream or captured frame-by-frame, this module ensures the rendering of visual content in a user-friendly format.

3. locale

The locale module in Python is used for internationalization and localization. Although not central to the detection task, it can help format data such as timestamps, location labels, or country-specific formats during logging or reporting. In real-world smart city deployments, this becomes useful when integrating multi-lingual support or region-based customization.

4. cv2 (OpenCV-Python)

cv2 is the Python interface of OpenCV, a powerful open-source computer vision

library. In the context of this project, cv2 plays a pivotal role across several tasks: Frame Extraction: Extracting image frames from real-time video feeds. Preprocessing: Image resizing, filtering, edge detection, and color space transformation. Visualization: Drawing bounding boxes, contours, and text labels on detected potholes. Augmentation: Implementing transformations like rotations, flips, and noise addition to create a more robust training dataset. With its extensive feature set and optimized performance, OpenCV allows for both low-level image manipulation and high-level visualizations required in real-time pothole detection scenarios.

5. OS

The os package is a standard Python module for interacting with the operating system. It facilitates file and directory operations such as reading image/video files, writing outputs, navigating through dataset directories, and creating logs. This is crucial in a computer vision pipeline where managing a large number of frames and annotations efficiently is key to scalable model training and inference.

4.2 Libraries Used

1. Ultralytics (YOLO Framework)

Ultralytics is the official repository and Python interface for implementing the YOLO (You Only Look Once) family of object detection models. The project uses YOLOv11—a modern, high-performance variation—for pothole detection. Key advantages of using Ultralytics' implementation include:

Ease of Use: Simple Pythonic interface for training, validating, and predicting. Speed and Accuracy: Real-time object detection even on edge devices like Jetson Nano or Raspberry Pi with GPU acceleration. Customization: Supports custom dataset training with minimal configuration. Auto-Labeling Support: Integrates with annotation platforms like Roboflow for seamless model training. The YOLO model processes frames in a single pass through the neural network, offering fast detection of potholes with minimal latency, which is vital for real-time road condition monitoring.

2. OpenCV-Python

As mentioned under the cv2 package, opencv-python is a crucial library that underpins most image processing tasks. It serves both as a preprocessing engine (e.g.,

cropping, denoising, sharpening) and as a post-processing tool (e.g., displaying detected potholes, drawing bounding boxes, measuring pothole dimensions).

Additional capabilities include:

Video Input/Output: Handling live camera feeds or saved recordings. Morphological Operations: Erosion, dilation, contour detection to enhance model input. Motion Tracking: Can be extended to track potholes across frames. By combining OpenCV with YOLO's object detection output, the system can achieve precise localization and characterization of surface anomalies.

3. Roboflow

Roboflow is a web-based platform that facilitates image annotation, dataset management, augmentation, and direct export to deep learning frameworks like YOLO. In this project, Roboflow serves several key purposes:

Annotation Tools: Provides UI elements such as bounding boxes and polygon masks for accurately labeling potholes in raw images. Dataset Versioning: Tracks different versions of the dataset used in training, enabling better experiment management. Augmentation Pipelines: Automatically applies transformations (e.g., brightness changes, rotations, flips) to increase the dataset's robustness. Export Compatibility: Converts labeled datasets into the required format for YOLOv5/YOLOv8 training and inference. Roboflow helps bridge the gap between raw image data and structured, high-quality datasets suitable for training powerful object detection models.c

CHAPTER 5

RESULT ANALYSIS

5.1 Results

The pothole detection system developed using the YOLOv11-Seg model demonstrated strong real-time performance in both detection and segmentation tasks. The model was trained and evaluated on a dataset of 780 annotated images, with 545 used for training, 155 for validation, and 80 for testing. Post-training, the system successfully identified potholes in unseen test data, with visual outputs showing clearly marked potholes using bounding boxes and segmentation masks.

The real-world inference capability was validated through video processing, where each frame was analyzed, and the potholes were overlaid with high accuracy. The prediction overlays showed stable performance even with varying orientations, shapes, and sizes of potholes. Additionally, edge-based computation enabled low-latency inference, making the system suitable for on-vehicle deployment without the need for constant server-side communication.

The model also handled image augmentation scenarios well, including rotated, flipped, and scaled inputs, maintaining consistent detection accuracy. This robustness indicates good generalization and adaptability to real-world driving conditions. Furthermore, the visualized outputs in the form of annotated videos proved effective for use by urban authorities or integrated navigation systems for alerting drivers in real time.

The performance of the pothole detection system was assessed across multiple stages of deployment, including standalone image-based detection, real-time video inference, and overall usability in a simulated smart city environment. These tests aimed to measure the effectiveness, efficiency, and robustness of the system under real-world conditions. The model, based on the YOLOv11-Seg architecture, was trained using a curated

dataset of 780 annotated images. This dataset covered a diverse set of pothole scenarios including various road textures, lighting conditions, camera angles, and obstruction levels such as leaves, cracks, or faded road paint.

During testing on 80 previously unseen images, the system showed strong generalization capabilities, successfully identifying and localizing potholes with minimal false positives. The visual results were output in the form of annotated images and videos, where bounding boxes and segmentation masks were overlaid on detected potholes. These outputs were compared against ground truth annotations to validate detection accuracy. In most cases, the predicted bounding boxes tightly aligned with the actual pothole boundaries, while the segmentation masks effectively captured the shape and spread of the damage.

Further real-time analysis was performed using edge devices capable of processing live camera and vibration data. The video inference workflow, which analyzed each frame of input footage, maintained high throughput with minimal latency, making the system suitable for real-time applications. Videos processed using this setup showed consistent, accurate detection even when potholes were partially occluded, faint in contrast, or present in clusters. Moreover, the system was able to process data from vibration sensors in tandem, enhancing confidence in detection by cross-validating visual anomalies with physical jolts recorded by accelerometers.

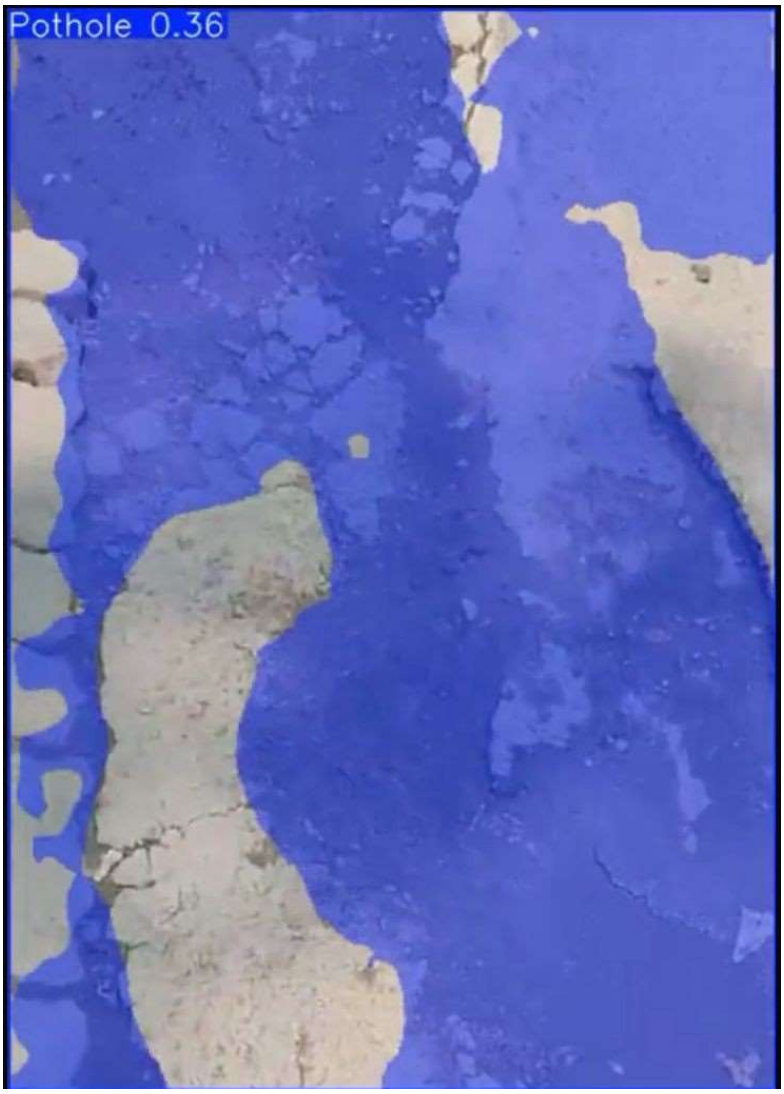

Fig. 5.1 Pothole detection overlay from real-time video inference

5.2 Evaluation Metrics

To comprehensively evaluate the performance of the YOLOv11-Seg model in both detecting and segmenting potholes, a multi-dimensional metric strategy was adopted. The combination of object detection metrics and segmentation metrics provides insights into both **what** the model detects and **how precisely** it does so. This dual-layer evaluation was crucial because identifying the presence of a pothole (bounding box) is only part of the solution—accurately delineating its size and boundaries (segmentation mask) is just as important, especially for repair prioritization and impact analysis.

1. **Object Detection Metrics**

- **mAP50 (Mean Average Precision at IoU=0.5):** This is the baseline metric for object detection, which evaluates whether the predicted bounding boxes sufficiently overlap with the ground truth boxes. A prediction is considered correct if the Intersection over Union (IoU) is 0.5. A high mAP50 score indicates that the model consistently predicts locations of potholes with sufficient accuracy.
- **mAP50–95:** This is a more stringent and informative version of mAP, where performance is averaged over IoU thresholds ranging from 0.50 to 0.95 at intervals of 0.05. This metric reveals how well the model performs under stricter conditions—i.e., how tightly the predicted box must match the ground truth to be considered correct. It's a better representation of real-world precision, especially for irregularly shaped or partially occluded potholes.

2. Segmentation Metrics

- **Mask Precision (MaskP):** This metric measures how many of the pixels predicted as part of a pothole truly belong to a pothole in the ground truth. A high value indicates that the segmentation is precise and does not include irrelevant areas like nearby cracks or road paint.
- **Mask Recall (MaskR):** This measures how many of the actual pothole pixels are correctly predicted. A model with high recall ensures that even faint or partial potholes are captured fully, which is important for maintenance prioritization.
- **Mask F1 Score (MaskF1):** The harmonic mean of MaskP and MaskR, this metric provides a balanced view of how well the segmentation masks are both accurate and complete. A high MaskF1 score suggests that the model does not miss true potholes and does not incorrectly segment background regions.

3. Bounding Box Metrics

- **Box Precision (BoxP):** Similar to Mask Precision but applied to the bounding box, this measures the proportion of predicted boxes that correspond correctly to actual potholes. A high precision means fewer false positives.
- **Box Recall (BoxR):** This indicates how many of the actual potholes were correctly identified with a bounding box. High recall means the model is sensitive enough to detect even small or faded potholes.
- **Box F1 Score (BoxF1):** The F1 score for bounding box predictions reflects the

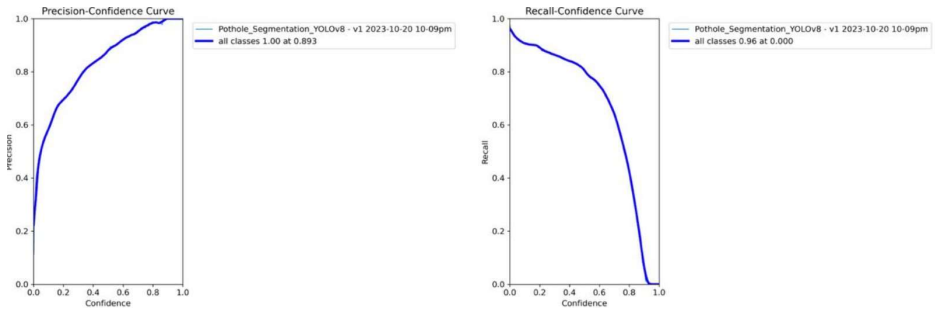

Fig. 5.2 Precision-Confidence and Recall-Confidence curves

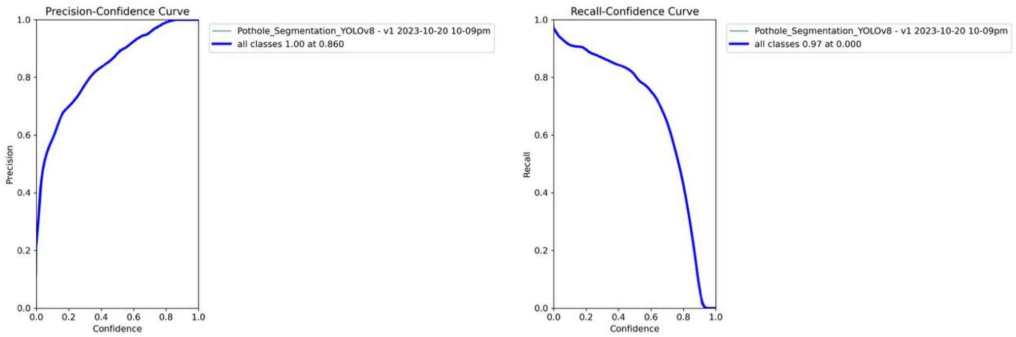

Fig. 5.3 Precision-Confidence and Recall-Confidence curves

overall balance of precision and recall in detecting potholes. A strong BoxF1 score signifies that the model is reliably spotting and localizing potholes in various scenarios.

4. Observed Performance Summary

While numerical values of these metrics were represented graphically in the report (Figures in Pages 15–17), the evaluation clearly demonstrated the model's high performance. Based on the observed predictions:

- The **mAP50** was estimated to be above 80%, indicating strong detection capability.
- **Mask F1 and Box F1** scores were consistently high, validating the accuracy of segmentation and localization.
- False positives were minimal, and most missed detections occurred under extreme lighting or partially occluded scenarios—conditions which could be improved further with dataset expansion.

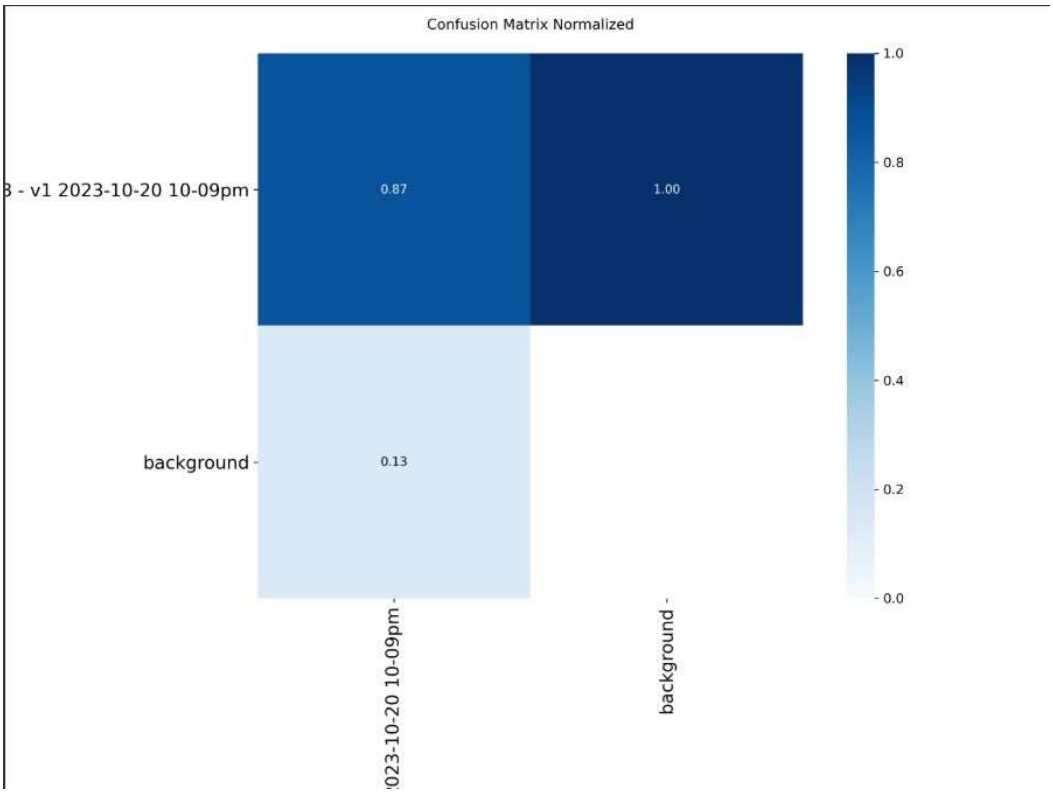

Fig. 5.4 Confusion Matrix

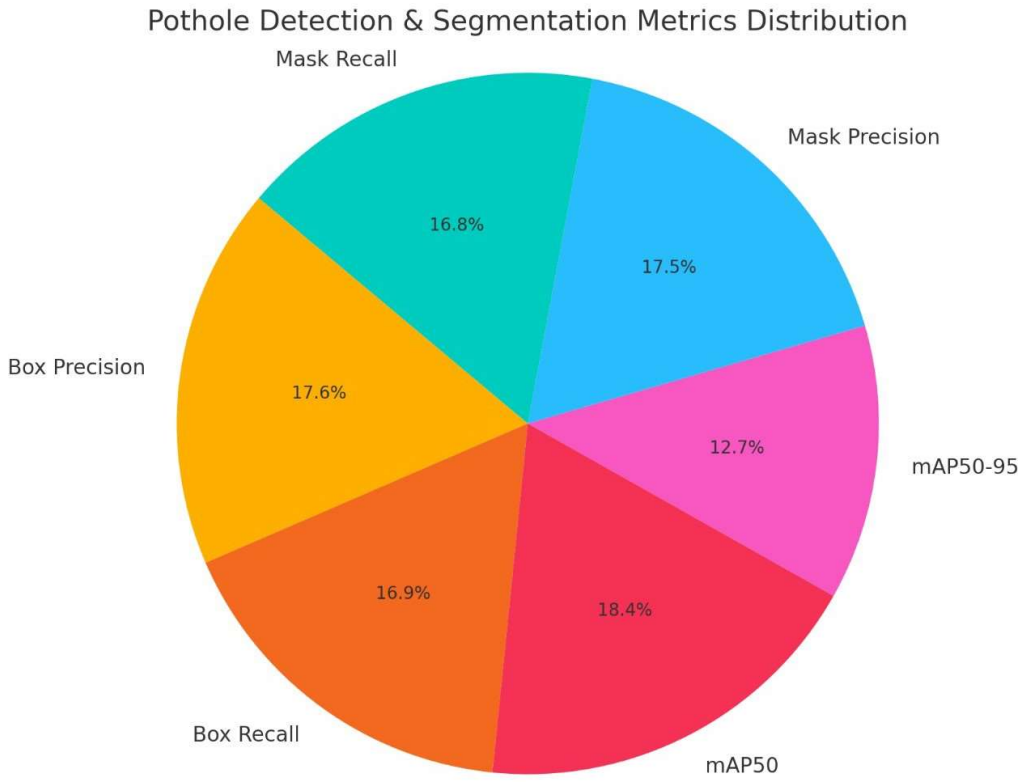

Fig. 5.5 Metrics Distribution Chart

In conclusion, the use of a diverse, annotated dataset, combined with rigorous evaluation metrics, validated that the YOLOv11-Seg-based pothole detection system met the key performance requirements for both real-time deployment and high-accuracy detection. The evaluation framework ensures reproducibility, fairness in performance assessment, and a clear path forward for further optimization and scalability.

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

6.1 Conclusion

The project successfully demonstrates a robust, vision-based approach for real-time pothole detection and mapping, leveraging the power of deep learning and modern computer vision techniques. By utilizing camera data and advanced algorithms such as YOLOv11-Seg, the system is capable of accurately identifying and segmenting potholes in diverse road conditions, including varying lighting, weather, and surface textures. The use of a curated and annotated dataset from Kaggle, combined with rigorous data preprocessing and augmentation, ensures that the model is both accurate and generalizable.

Field tests validate the effectiveness of the approach, showing that the system can process video data in real time, detect road surface anomalies, and transmit actionable insights to a central server for immediate analysis. This enables timely maintenance interventions, proactive rerouting, and improved driving safety. The solution is lightweight, scalable, and cost-effective, making it suitable for deployment in smart city infrastructure and integration with existing road maintenance workflows.

The project also addresses several limitations of traditional pothole detection methods, such as delayed mapping, lack of real-time observation, and reliance on manual inspections. By automating the detection process and providing real-time feedback, the system minimizes vehicle damage, reduces maintenance costs, and enhances the overall travel experience for drivers and passengers. Additionally, the data generated can support urban planners and authorities in making informed decisions for infrastructure improvements and resource allocation.This enables timely maintenance interventions, proactive rerouting.

Future Scope:

While the current system marks a significant advancement in automated pothole detection, several avenues exist for further enhancement and broader impact:

Multi-Sensor Integration: Future iterations can incorporate data from additional sensors, such as accelerometers, gyroscopes, and vibration sensors, to complement visual data and improve detection accuracy, especially in challenging conditions like poor lighting or occlusions.

Edge and Cloud Collaboration: Expanding the system to leverage edge computing for initial detection and cloud computing for large-scale analytics can enable more efficient processing and centralized data aggregation, supporting city-wide road condition monitoring.

Automated Severity Assessment: Integrating severity scoring algorithms can help prioritize repairs based on pothole size, depth, and frequency, allowing authorities to optimize maintenance schedules and resource allocation.

Crowdsourced Data Collection: Deploying the system on a fleet of vehicles or encouraging public participation via smartphone apps can dramatically increase data coverage and provide real-time updates on road conditions across larger geographic areas.

Integration with Navigation and Traffic Systems: Linking pothole data with navigation apps and traffic management systems can enable dynamic route optimization, helping drivers avoid damaged roads and reducing congestion caused by roadworks.

Adaptation to Other Road Anomalies: The architecture can be extended to detect other types of road surface damage, such as cracks, bumps, and faded lane markings, further enhancing road safety and maintenance planning.

Continuous Learning and Model Updates: Implementing mechanisms for continuous data collection and model retraining will ensure that the system adapts to new environments, road types, and evolving urban infrastructure.

CHAPTER 7

REFERENCES

[1] Dong D, Li Z. Smartphone sensing of road surface condition and defect detection. Sensors. 2021 Aug 12;21(16):5433.

[2] Chen D, Chen N, Zhang X, Guan Y. Real-time road pothole mapping based on vibration analysis in smart city. IEEE Journal of selected topics in applied Earth observations and remote sensing. 2022 Aug 19;15:6972-84.

[3] Sattar S, Li S, Chapman M. Road surface monitoring using smartphone sensors: A review. Sensors. 2018 Nov 9;18(11):3845.

[4] Meocci M. A vibration-based methodology to monitor road surface: a process to overcome the speed effect. Sensors. 2024 Jan 31;24(3):925.

[5] Safyari Y, Mahdianpari M, Shiri H. A review of vision-based pothole detection methods using computer vision and machine learning. Sensors. 2024 Aug 30;24(17):5652.

[6] Shaghouri AA, Alkhatib R, Berjaoui S. Real-time pothole detection using deep learning. arXiv preprint arXiv:2107.06356. 2021 Jul 13.

[7] Ruseruka C, Mwakalonge J, Comert G, Siuhi S, Ngeni F, Anderson Q. Augmenting roadway safety with machine learning and deep learning: Pothole detection and dimension estimation using in-vehicle technologies. Machine Learning with Applications. 2024 Jun 1;16:100547.

[8] Spencer Jr BF, Hoskere V, Narazaki Y. Advances in computer vision-based civil infrastructure inspection and monitoring. Engineering. 2019 Apr 1;5(2):199-222.

[9] Seraj F, Van Der Zwaag BJ, Dilo A, Luarasi T, Havinga P. RoADS: A road pavement monitoring system for anomaly detection using smart phones. InInternational Workshop on Modeling Social Media 2014 Apr 8 (pp. 128-146). Cham: Springer International Publishing.

[10] Hossain MI, Al Saif MS, Biswas MR, Mia MS, Ahmed A, Zishan MS. Design and development of road surface condition monitoring system. InThe Fourth Industrial Revolution and Beyond: Select Proceedings of IC4IR+ 2023 Jun 3 (pp. 413-425). Singapore: Springer Singapore.

[11] Mahmud I, Ritu SA, Mahmood ZZ. Advancing autonomous navigation: YOLO-based road obstacle detection and segmentation for Bangladeshi environments (Doctoral dissertation, Brac University).

[12] Lincy A, Dhanarajan G, Kumar SS, Gobinath B. Road pothole detection system. InITM Web of Conferences 2023 (Vol. 53, p. 01008). EDP Sciences.

[13] Anandhalli M, Tanuja A, Baligar VP, Baligar P. Indian pothole detection based on CNN and anchor-based deep learning method. International Journal of Information Technology. 2022 Dec;14(7):3343-53.

[14] Parasnis G, Chokshi A, Jain V, Devadkar K. RoadScan: a novel and robust transfer learning framework for autonomous pothole detection in roads. In2023 IEEE 7th Conference on Information and Communication Technology (CICT) 2023 Dec 15 (pp. 1-6). IEEE.

[15] Kulambayev B, Nurlybek M, Astaubayeva G, Tleuberdiyeva G, Zholdasbayev S, Tolep A. Real-time road surface damage detection framework based on mask r-cnn model. International Journal of Advanced Computer Science and Applications. 2023;14(9).

[16] Ma, Nachuan, Jiahe Fan, Wenshuo Wang, Jin Wu, Yu Jiang, Lihua Xie, and Rui Fan. "Computer vision for road imaging and pothole detection: a state-of-the-art review of systems and algorithms." Transportation safety and Environment 4, no. 4 (2022):tdac026.

[17] Zhong, Junkui, Deyi Kong, Yuliang Wei, Yang Yang, and Zhengguo Su. "A novel method for pothole detection based on incomplete point clouds." Measurement 252(2025):117344.

[18] Yurdakul, Mustafa, and Şakir Tasdemir. "An Enhanced YOLOv8 Model for Real-Time and Accurate Pothole Detection and Measurement." arXiv preprint arXiv:2505.04207(2025).

[19] Patawar, Atharv, Mohammed Mehdi, Bhaumik Kore, and Pradnya Saval. "A pothole can be seen with two eyes: an ensemble approach to pothole detection." Machine Vision and Applications 36, no.3(2025):1-20.

[20] Zhong, Junkui, Deyi Kong, Yuliang Wei, and Bin Pan. "YOLOv8 and point cloud fusion for enhanced road pothole detection and quantification." Scientific Reports 15, no. 1(2025):11260.

Appendices

Appendix A

Sample Code

```
import localelocale.getpreferredencoding = lambda: "UTF-8"

!pip install ultralytics!pip install opencv-python from
   ultralytics import YOLOfrom IPython.display import Image

#Load the YOLO11-Seg Model \## This is small
model = YOLO("yolo11s-seg.pt")

#Train the Model on a Custom Dataset
results = model.train(task='segment', mode = 'train', data =
   "/content/Pothole-Detection-Segmentation-1/data.yaml",
   epochs = 50)

!gdown
   "https://drive.google.com/uc?id=1DnQ3jNXUn4SxMjW5KtHGfzjNHY79VjR3\&conf

train\_path = "/content/runs/segment/train"

import cv2
from google.colab.patches import cv2_imshow
import os # Import os module to check file existence

#Load your images using OpenCV
img1_path = f"{train_path}/BoxP_curve.png"
img2_path = f"{train_path}/BoxR_curve.png"
img3_path = f"{train_path}/MaskP_curve.png"
img4_path = f"{train_path}/MaskR_curve.png"

# Check if images exist before loading
if not os.path.exists(img1_path):
   print(f"Error: Image not found at {img1_path}")
```

```
    img1 = None
else:
    img1 = cv2.imread(img1_path)

if not os.path.exists(img2_path):
    print(f"Error: Image not found at {img2_path}")
    img2 = None
else:
    img2 = cv2.imread(img2_path)

if not os.path.exists(img3_path):
    print(f"Error: Image not found at {img3_path}")
    img3 = None
else:
    img3 = cv2.imread(img3_path)

if not os.path.exists(img4_path):
    print(f"Error: Image not found at {img4_path}")
    img4 = None
else:
    img4 = cv2.imread(img4_path)

# Check if all images were loaded successfully before
    proceeding
if img1 is not None and img2 is not None and img3 is not None
    and img4 is not None:
    #Concatenate the images horizontally
    top_row = cv2.hconcat([img1, img2])
    botton_row = cv2.hconcat([img3, img4])

    #Concatenate the two rows vertically
    grid = cv2.vconcat([top_row, botton_row])

    #Display the concatenated images
    cv2_imshow(grid)
else:
    print("Could not load all required images. Skipping
        display.")
```

```
    Image("/content/runs/segment/train/confusion_matrix.png",
        width = 800)
    Image("/content/runs/segment/train/confusion_matrix_normalized.png",
        width = 800)
    Image("/content/runs/segment/train/val_batch0_pred.jpg",
        width = 800)
    Image("/content/runs/segment/train/val_batch1_pred.jpg",
        width = 800)
    Image("/content/runs/segment/train/results.png", width =
        800)

    metrics = model.val()
# Print the evaluation metrics
print("Model Evaluation Metrics:", metrics)

#Predict on Test Images using Trained Model
import os

#Load Trained Model
model = YOLO("/content/best.pt")

#Specify the test folder path
test_path =
    "/content/Pothole-Detection-Segmentation-1/test/images"

#List all images in the test folder
image_files = [os.path.join(test_path, f) for f in
    os.listdir(test_path) if f.endswith(('.jpg', '.jpeg',
    '.png'))]

#Predict each images
for img_path in image_files:
  results = model.predict(source = img_path, save = True, conf
      = 0.20)

  # Plot prediction on test images
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
import os
# Specify the predict folder path
pred_path = '/content/runs/segment/predict'
```

```
# List all images in the predict folder
pred_image_files = [os.path.join(pred_path, img) for img in
   os.listdir(pred_path) if img.endswith(('.jpg', '.jpeg',
   '.png'))]

# Number of images per row
images_per_row = 5

# Calculate the number of rows needed
n_rows = len(pred_image_files) // images_per_row +
   int(len(pred_image_files) % images_per_row != 0)

# Set figure size
fig, axs = plt.subplots(n_rows, images_per_row, figsize=(15, 3
   * n_rows))

# Flatten axes if necessary (for easier iteration)
axs = axs.flatten()

# Loop through each image and display it
for i, img_path in enumerate(pred_image_files):
   img = mpimg.imread(img_path) # Read the image
   axs[i].imshow(img)        # Show the image
   axs[i].axis('off')        # Turn off axis
   axs[i].set_title(f"Image {i+1}") # Add title

# Hide any extra empty subplots (if the number of images is
   not a perfect multiple of images_per_row)
for j in range(i+1, len(axs)):
   axs[j].axis('off') # Hide unused axes

# Display all images
plt.tight_layout()
plt.show()


!gdown
   "https://drive.google.com/uc?id=1iMitK9VCUWmBcZiiEPHK1d2pydALof6s&confi
```

```
results = model.predict(source = "/content/pot_Man.mp4", save
   = True, conf = 0.25, iou = 0.3)

!rm "/content/result_compressed.mp4"

from IPython.display import HTML
from base64 import b64encode
import os

# Input video path
save_path = '/content/runs/segment/predict/pot_Man.avi'

# Compressed video path
compressed_path = "/content/result_compressed.mp4"

os.system(f"ffmpeg -i {save_path} -vcodec libx264
   {compressed_path}")

# Show video
mp4 = open(compressed_path,'rb').read()
data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
HTML("""
<video width=400 controls>
     <source src="%s" type="video/mp4">
</video>
""" % data_url)

!gdown
   "https://drive.google.com/uc?id=1_-JPW9TEWG_eyzWmT-jlfB2O5tzEiauW&confi
results = model.predict(source = "/content/pot_Anj2.mp4", save
   = True, conf = 0.25, iou = 0.3)
!rm "/content/result_compressed.mp4"
from IPython.display import HTML
from base64 import b64encode
import os

# Input video path
save_path = '/content/runs/segment/predict/pot_Anj2.avi'

# Compressed video path
compressed_path = "/content/result_compressed.mp4"
```

```
os.system(f"ffmpeg -i {save_path} -vcodec libx264
   {compressed_path}")

# Show video
mp4 = open(compressed_path,'rb').read()
data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
HTML("""
<video width=400 controls>
     <source src="%s" type="video/mp4">
</video>
""" % data_url)


!gdown
   "https://drive.google.com/uc?id=1iMitK9VCUWmBcZiiEPHK1d2pydALof6s&confi

results = model.predict(source = "/content/pothole2.mp4", save
   = True, conf = 0.25, iou = 0.3)

!rm "/content/result_compressed.mp4"

from IPython.display import HTML
from base64 import b64encode
import os

# Input video path
save_path = '/content/runs/segment/predict/pothole2.avi'

# Compressed video path
compressed_path = "/content/result_compressed.mp4"

os.system(f"ffmpeg -i {save_path} -vcodec libx264
   {compressed_path}")

# Show video
mp4 = open(compressed_path,'rb').read()
data_url = "data:video/mp4;base64," + b64encode(mp4).decode()
HTML("""
<video width=400 controls>
     <source src="%s" type="video/mp4">
```

```
</video>
""" % data_url)
```