UAV-based Street Road Inspection for Smart Cities Using Machine Learning

A Project Report

Presented to

The Faculty of the Department of Applied Data Science

San Jose State University

In Partial Fulfillment

Of the Requirements for the Degree

Master of Science in Data Analytics

By

Sweekruthi Balivada, Manisha Lagisetty, Yuting Sha, Damini Prashant Vichare

Dec 2, 2024

APPROVED FOR DEPARTMENT OF APPLIED DATA SCIENCE

Dr. Jerry Gao, Project Advisor

Dr. Lee C. Chang, Department Chair

Abstract

UAV-based Street Road Inspection for Smart Cities Using Machine Learning
By

Sweekruthi Balivada, Manisha Lagisetty, Yuting Sha, Damini Prashant Vichare

Efficient transportation management is vital in modern urban environments, where deteriorating road conditions, increasing traffic, and rising safety concerns pose significant challenges. Traditional road inspection methods are labor-intensive, time-consuming, and prone to inaccuracies, limiting their effectiveness. To overcome these limitations, this project introduces a UAV-based road inspection system utilizing advanced machine learning techniques. The system operates in a modular two-stage framework: in Stage 1, an enhanced YOLOv8 model detects six categories of road obstacles, including cracks, potholes, wildlife debris, garbage dumping, construction activities, and accidents. In Stage 2, category-specific models, including Vision Transformers (ViT) and MobileNetV1-SSD (Single Shot MultiBox Detector), dynamically classify the severity of the detected anomalies, categorizing them into low, medium, or high severity levels. Insights from this system are seamlessly integrated into an interactive dashboard, providing transportation authorities with actionable information for proactive maintenance, improved road safety and efficient resource allocation. This approach combines precision and computational efficiency, offering a scalable solution for smart city infrastructure management and transportation planning.

Keywords: Smart cities, UAV, Road anomaly detection, Machine Learning, Transportation management, Object detection, Severity classification

ACKNOWLEDGMENTS

We would like to express our deepest gratitude to our project advisor, Dr. Jerry Gao, for his invaluable guidance, encouragement, and support throughout the course of this project. His expert insights and constructive feedback have been instrumental in shaping the direction and quality of our work. We are immensely thankful for his patience and mentorship, which have motivated us to excel in our research and overcome challenges with determination. We also extend our heartfelt thanks to the Faculty of the Department of Applied Data Science at San Jose State University for providing us with the knowledge, tools, and resources needed to pursue this project successfully. Their dedication to academic excellence has greatly enriched our learning experience. Our deepest appreciation goes to our families and friends for their unwavering support, understanding, and encouragement during this journey. Their belief in our abilities has been a constant source of strength. Lastly, we are grateful to our peers and colleagues for fostering a collaborative and inspiring environment that has encouraged the exchange of ideas and mutual growth throughout this endeavor. This project would not have been possible without the contributions and support of everyone mentioned above. Thank you.

TABLE OF CONTENTS

1. Introduction

1.1 Project Background and Execute Summary

Project Background, needs and importance

In light of the growing difficulties posed by declining road conditions and rising traffic volumes in contemporary urban areas, there is a pressing requirement for innovative approaches to uphold effective transportation infrastructure. The presence of potholes, cracks, debris obstructions, and collisions not only hinders the smooth flow of traffic but also pose significant safety risks. Traditional methods of manual inspection are frequently insufficient, resulting in delayed maintenance and increased expenses for local governments. The American Society of Civil Engineers (ASCE), has reported that around 20% of major roads in the United States are in unsatisfactory condition, leading to over 14,000 fatalities each year and causing an average annual expenditure of $523 per motorist on vehicle repairs Yan and Zhang (2023). Additionally, debris on roadways accounts for approximately a quarter of all traffic-related deaths annually, underscoring the urgent requirement for effective road inspection and maintenance strategies as indicated by the National Highway Traffic Safety Administration (NHTSA) Figshare (2022). Potholes in developed countries have substantial economic ramifications with costs amounting to billions of dollars every year roboflow (2023).

In recent years, there has been a noticeable increase in urbanization, leading to added pressure on transportation infrastructure. The United Nations' World Urbanization Prospects report predicts that the world's urban population will grow by 2.5 billion by 2050, with significant growth happening in developing regions Anugrahakbarp (2023). This fast-paced urbanization worsens the difficulties of sustaining road systems and guaranteeing public safety. To address these pressing challenges, our project aims to revolutionize transportation management by employing data-driven models and UAV (unmanned aerial vehicles) imagery to detect and classify road anomalies, such as potholes, cracks, debris, dead animals, construction activities, and accidents, in real-time. In smart cities, proactive transportation management is essential, which highlights the significance of our project.

Potholes and fissures not only deteriorate road conditions but also lead to vehicle damage and collisions, causing considerable financial burdens each year. Furthermore, obstructions on roads present notable safety threats and environmental risks, while accidents and construction work disrupt traffic flow and hinder emergency response efforts. As urban areas contend with the difficulties of urbanization and climate change, it is crucial to prioritize investment in advanced technologies such as UAV-based road inspection systems. By addressing road flaws and hazards, this initiative strives to create safer and more adaptable urban settings for both residents and commuters.

UAVs carry out thorough and effective road inspections, swiftly covering extensive areas and offering detailed images for analysis. Utilizing advanced object detection algorithms like YOLO and SSD, the system enables UAVs to independently identify and categorize road issues with great accuracy Kraft et al. (2021). Moreover, sensor fusion methods improve anomaly detection reliability by integrating data from diverse sources including cameras, infrared sensors, and LiDAR technology on WKYC. The utilization UAVs has a benefit in that they may gather high-resolution sensor data in real-time, which is essential for detecting irregularities on the road and can assist in the monitoring of encroachments, traffic jams, and road building activities, offering insightful information for the management of transportation infrastructure on youtube.

Traditional techniques frequently depend on manual visual examinations, which can be time-consuming, labor-intensive, and susceptible to inaccuracies. Additionally, accessing specific locations for inspection like highways or bustling urban areas can pose risks to workers and cause disruptions to traffic flow. These constraints emphasize the necessity for automated and data-driven methods for road inspection, analysis, and enhancing transportation management by harnessing data from UAV and satellite imagery. City officials and transportation authorities will have access to accurate and timely information about road conditions through the advancement of computer vision techniques and machine learning algorithms. This will enable them to make informed decisions in order to enhance the safety and efficiency of transportation networks. Furthermore, a dedicated web portal for smart city road inspection, analysis, and monitoring using drone data will be established to improve the accessibility and usefulness of the insights generated.

After doing thorough market analysis we identified a diverse range of target segments, including government transportation agencies responsible for road networks, smart city initiatives aiming at urban infrastructure enhancement, urban planning departments overseeing development, and private companies specializing in infrastructure maintenance. We also evaluated the strengths and weaknesses of the existing technologies such as technological prowess, their market reputation, etc. From weaknesses point of view, we found out of any outdated technology, limited coverage, and any gaps in their offerings, which could be leveraged for our advantage. Furthermore, we delve deeper into market trends and drivers, while considering urbanization trends, government regulations, technological advancements, and shifting consumer preferences towards smarter, more efficient transportation solutions. By understanding these trends, we can align our project with emerging needs and capitalize on evolving market dynamics. Our main motto is to address customer needs. We also identified pain points such as if there are any delayed maintenance or if there is any inefficient inspection processes. We also determined priorities like enhancing road safety and reducing costs. After carefully assessing the market size and growth potential we will analyze demographic data, infrastructure spending projections, and market research reports. This will help us to estimate the demand for UAV-based road inspection systems and identify opportunities for innovation and expansion. We also conducted a thorough risk analysis which was essential for our project to mainly evaluate regulatory hurdles, technological barriers and develop a strategic plan for success. By conducting a comprehensive market analysis covering these key areas, we aim to position our UAV-based road inspection system strategically in the market, ensuring its viability and effectiveness in addressing the evolving needs of transportation infrastructure management, as mentioned by Johnson, B. et al. (2020).

1.2 Project Requirements

The purpose of implementing this system is to develop an intelligent system using various UAV technologies combined with a bunch of machine learning algorithms which will be beneficial for efficient street road inspection in smart cities, as mentioned by Gao, J. (2023). Our system will ensure that it will play a very crucial role in not only detecting but also classifying various road anomalies like potholes, cracks, debris, dead animals, construction activities, and accidents in real-time. By making the best use of time-efficient methodologies and expertise available we aim to develop a system which further aims to enhance transportation safety as well as infrastructure management. Our project aims to empower users to interact with the system seamlessly and access valuable insights derived from drone data analysis by providing a user-friendly web interface.

Functional Requirements

Functional requirements are nothing but essential features and capabilities that the system must have in case of effectively performing its tasks. These requirements directly influence the functionality of the system and are listed as follows:

- Gather a comprehensive database of images to train machine learning models for road anomaly detection, including potholes, cracks, debris, and traffic hazards like garbage, car accidents, dead animals on road.
- Develop efficient machine learning models that are specifically tailored for not only object detection but also classification, and anomaly recognition to analyze road conditions and identify potential hazards.
- Design and implement an efficient model so that it can be useful for hazard avoidance by enabling drones to navigate safely while avoiding collisions with obstacles and other hazards on the road.
- Implement two user levels: administrator and operator. Administrators will have access to system configuration and management functionalities, while operators will be able to utilize the inspection and analysis features.
- Develop a user-friendly web interface for operators to interact with the system who can further visualize inspection results and also access analytical reports.
- Generate evaluation reports after each inspection session or bi-weekly or monthly which will highlight detected anomalies, road conditions, and potential hazards identified during the process.
- Achieve all functional requirements utilizing only UAV based systems and computer vision methods, ensuring that the system operates efficiently and effectively without relying on additional sensors or hardware

Non-Functional Requirements

Non-Functional Requirements are collectively responsible for ensuring the various factors such as reliability, safety, responsiveness, efficiency, and adaptability. Following are the non-functional requirements for our UAV-based Street Road Inspection system:

- Develop a system that will ensure to operate effectively in various weather conditions, the lighting levels, and the road surface textures. The system should be able to have consistent performance and accuracy in road hazard detection in whatever weather conditions.
- Achieve an accuracy rate of minimum 75-80% for hazard detection, considering factors such as hazard size, distance, and orientation. Implement threshold optimization mechanisms based on detection probabilities to ensure reliable hazard identification.
- Develop a system that should have capability of real-time processing and analysis of streaming data from UAVs to ensure timely detection and classification of road anomalies and hazards.
- Optimization of computational resources is most essential as it will ensure efficient processing of large-scale datasets while minimizing computational costs and energy consumption.
- The system should exhibit high reliability and availability to support continuous road inspection and hazard analysis operations, minimizing downtime and ensuring uninterrupted service delivery.

1.3 Project Deliverables

The project aims to deliver a comprehensive range of tailored solutions aimed at improving transportation management by implementing intelligent systems for road inspection and analysis. Key deliverables include creating and deploying a data-driven machine learning model for real-time detection and classification of road issues such as potholes, cracks, illegal dumping, car accidents, dead animals, and traffic hazards. The model will be trained using cutting-edge technologies like YOLOv5, Faster R-CNN, YOLOv8, ResNet-101, and CNN-SSD-16 on UAV and satellite imagery datasets to ensure accurate identification of road anomalies. Furthermore, the project will evaluate various convolutional neural network architectures to determine the most effective model for this purpose.

The development of a user-friendly web portal dedicated to smart city road inspection, analysis, and monitoring is another important key deliverable. This platform will act as a centralized space for viewing and analyzing data on road conditions. A thorough technical report will be prepared, outlining the methods, algorithms, and findings of the project. This report will be a valuable asset for future research and development in transportation management and road inspection.

1.4 Technology and Solution Survey

Based on the technology survey shown in Table 1, we found out that different technologies and systems have been built for various issues like illegal dumping, garbage detection, potholes and cracks detection as well as construction activities and dead animal detection. This provided us with a comprehensive overview of the current landscape of UAV-based street road inspection for smart cities which highlighted the role of machine learning in enhancing the efficiency, accuracy, and safety of urban infrastructure maintenance. By leveraging advanced UAV platforms and state-of-the art models the systems were able to address road anomalies and hazards, leading to improved road quality, transportation management, and overall urban resilience.

Table 1

Comprehensive overview of Technology Survey

Name	Support Functions						Conn		AI					Deploy Model			Service Model		
Title	Tracking	Monitoring	Misson planning	Simulation	Communication	Drone management	WI-Fi	Satellite	Pothole and cracks	Cracks detection	Illegal dumping	Animal on road	Car Accident	Public	Private	Hybrid	SaaS	PaaS	IaaS
CVEDIA-RT	N	Y	Y	N	Y	N	Y	N	Y	Y	Y	Y	Y	Y	Y	Y	N	Y	N
Improving Trash Monitoring with Drone Imagery, Artificial Intelligence, Mapping	Y	Y	Y	N	Y	Y	Y	N	N	N	Y	N	Y	Y	N	N	Y	N	N
viso.ai: Computer Vision Applications for Smart City	Y	Y	Y	N	Y	Y	Y	N	Y	Y	N	Y	Y	Y	N	N	N	N	N
viAct- Danger Zone Detection	N	N	Y	N	N	N	Y	N	Y	N	N	Y	N	Y	N	N	Y	Y	N
Drone-Based Animal Detection	N	Y	Y	Y	N	N	Y	Y	Y	N	N	N	N	N	N	N	N	N	N
Debris Object	N	N	N	Y	N	N	N	N	Y	N	N	N	N	N	Y	N	N	N	N

Detection Caused by Vehicle Accidents Using UAV and Deep Learning Techniques																			
Automated Crack Detection in Quality Control with AI.SEE	Y	Y	Y	N	N	N	Y	N	N	Y	N	N	N	N	Y	N	N	Y	N

1.5 Literature Survey

Exploration and Analysis of Existing Research

Conducting a comprehensive review of existing literature as shown in Table 2, is crucial for grasping the variety of detection models used in different urban environments. The provided table presents a thorough summary of diverse models, their sources of data, targeted locations, altitudes, and specific measurements undertaken. This examination reveals that deep learning frameworks like YOLO, CNN models, Faster R-CNN, and others are pivotal in handling detection responsibilities such as identifying vehicles on roads and detecting unauthorized waste disposal in urban areas. These adaptable models exhibit potential to improve safety and effectiveness across various fields, as mentioned by Aburasain et al. (2020).

Model comparison is a crucial step for the research as shown in Table 2. Our survey reveals that models such as VGG-19 and U-net DCNNs are used to detect cracks on roads, while

SVM classification and SWM algorithms are applied to identify cracks and litter on highways as well as smaller streets. Additionally, research also covers the detection of road debris, potholes, and vehicle accidents using models like YOLOv5, Tiny YOLO, and Convolutional LSTM, indicating the promising capabilities of AI in enhancing road safety and maintenance efforts, as mentioned by Bezdan A. et al. (2023) and Rabbany A. et al (2022).

Furthermore, the survey explores the use of advanced models such as AAE and ResNet50 with Feature Pyramid Network for identifying illegal activities like illegal dumping and landfills. This review emphasizes how detection models can address a range of challenges in urban environments and suggests potential advancements in detection technology that could improve safety, sustainability, and efficiency as mentioned by Awolusi, I. et al. (2023) and Mittal, P. et al. (2020).

Table 2

Comprehensive overview of Literature Survey

Detection Models	**Data Source**	**Urban Area**	**Altitude**	**Measurements**	**Ref Paper**
YOLO	Custom	Roads and Highways	Various	Vehicle Detection	Yan and Zhang (2023)
CNN models-SSD-500 and YOLO V3	Custom	Street & desserts	Various	Cattle Detection	Figshare (2022)
Faster R-CNN, SSD, YOLOv5, YOLOv7, and	Custom	Highway, forest, deserts	Various	Animal Detection	Roboflow (2023)

YOLOv8					
Faster R-CNN and YOLOv3	Custom	Construction Sites	Various	Construction	Anugrahakbarp’s GitHub repository (2023)
Faster RCNN, Cascade RCNN, R-FCN, YOLO	Custom	Object detection	Various	Animals	Kraft et al. (2021)
CNN models	Custom	Roads, Forests	Various	Wild Animals	iStockphoto
VGG- 19 and U-net DCNNs.	CRACK500	Roads	Various	Cracks Detection	WKYC
Faster-RCNN, YOLOv5s, YOLOv7-tiny, and YOLOv8s	CRACK500	Roads	22.5 m	target detection, cracks detection, road damage evaluation	YouTube video
CNN VGG16	SDNet	Roads	Various	Cracks Detection	YouTube video
SVM classification	Custom	Highways	Various	Cracks Detection	Zhou et al. (2018)
SWM, simple linear iterative clustering (SLIC)	Custom	Small streets and Roads	Various	Litter Detection	Aidouni (2022)

CNN1, CNN2	Custom	Parks, Beaches, Streets	Various	Garbage Detection	Murthy et al. (2021)
SSD, YOLOv3, YOLOv4	UAVVDataset	Roads, Parking lots, sidewalks, parks, etc.	Various	Garbage Detection	Chen et al. (2024)
YOLOv3	Custom	Streets, Roads, sidewalks, parks, etc.	Various	Domestic Garbage Detection	Khare et al. (2023)
HFCN and UNet++	Custom	Roads	Various	Road Garbage Segmentation	American Society of Civil Engineers. (2020)
AAE models	Custom	Streets, Roads, sidewalks, parks, etc.	Various	Illegal dumping Detection	National Highway Traffic Safety Administration. (2019)
ResNet50 and Feature Pyramid Network (FPN)	Custom	Streets, Roads, sidewalks, parks, etc.	Various	Illegal Landfills Detection	Smith, J. et al. (2018)

YOLOv5	Custom Dataset	highway	Various	Potholes and Cracks Detection, Classification	United Nations. (2019)
Tiny Yolo Object detection, CNN	Online Open Source and Custom Dataset	Road	Various ranges of height up to 30m	Pothole Detection	Redmon, J., & Farhadi, A. (2018)
YOLOv3–ResNet101	consecutive RGB frames of custom video	highways	Various	Pothole Detection	Liu. et al. (2016)
Faster R-CNN, and SSD models	Custom Dataset	road	Various	Debris caused by Vehicle Accidents Detection	Liao et al. (2022)
Convolutional LSTM	Custom Dataset	road	Various ranges of height	Crashed Car Detection, Classification	Truong et al. (2021)

Model Comparison:

The comparison table, that is Table 3, presents the unique features and performance characteristics of five prominent models for object detection: YOLOv5, Faster R-CNN, YOLOv8, ResNet-101, and CNN-SSD-16. YOLOv5 and YOLOv8 demonstrate real-time capability and efficiency during training; however, they may face challenges in detecting small objects. Faster R-CNN excels in precise localization but lacks real-time performance. On the other hand, ResNet-101 offers high accuracy at the expense of computational complexity, while CNN-SSD-16 maintains a balance between efficiency and accuracy, as mentioned by Ahmed F., et al. (2022) and Torres R.N. et al (2018). These insights support informed decision-making when selecting a model based on specific application requirements by highlighting the significance of understanding trade-offs in object detection performance.

Table 3

Comprehensive Comparison of Object Detection Models

Characteristics	**YOLOv5**	**Faster R-CNN**	**YOLOv8**	**Resnet-101**	**CNN-SSD-16**
Architecture	Single stage	Two-Stage	Single-Stage	Residual Network	Single Shot MultiBox Detector
Dataset	COCO	Region based CNN	COCO	Various	COCO
Data Type	Image	Image	Image	Image	Image
Real-Time	Yes	No	Yes	No	Yes
Targeted Problems	Real time Object Detection	Object Detection and Localization	Real time Object Detection	Image Classification, Object Detection	Object Detection
Approaches	Deep Learning	Deep Learning	Deep Learning	Deep Learning	Deep Learning
Training Speed	Fast	Moderate	Fastest	Moderate	Fast

Training Time	Few Hours	Longer than YOLOv5	Few Hours	Longer	Moderate
Accuracy	High	High	Most Accurate	Very High	Moderate to High
Preprocessing Required	Minimal Pre-processing	Moderate Pre-processing	Minimal Pre-processing	Moderate Pre-Processing	Minimal Pre-processing
Space Complexity	Moderate	High	Moderate	High	Moderate
Computational Complexity	Moderate	High	Moderate to High	High	Moderate to High
Strengths	Efficient	Accurate Localization	Efficient	Excellent Accuracy, Depth	Efficient
Known issues	May Struggle with small object detection	Slower inference speed compared to single-stage detectors like YOLO	May Struggle with small object detection	Highly Computational cost due to depth	May have slightly lower accuracy compared to two-stage detectors like Faster R-CNN.

2. Data and Project Management Plan

2.1 Data Management Plan

As we aim to develop a UAV-based road inspection system, we focus on collecting UAV-based images as our datasets. There are two approaches we used to collect our datasets: One is downloading existing UAV-based image datasets, the other is converting drone video into frames using OpenCV.

After collecting the raw data, we check the data quality to ensure it is qualified to address our target problem. Additionally, for some custom data without existing labels, manual annotation is required to make the data reliable and suitable for model training.

For data storage methods, we have chosen Amazon S3 cloud server. It offers user-friendly features such as data replication and version updating. Also, it provides robust security controls, including access control policies through AWS Identity and Access Management (IAM).

The following data usage mechanisms would be applied to ensure the effective utilization of our collected data. Firstly, we perform cleaning and preprocessing on both UAV-based images and frames converted from drone videos. Next, we implement data augmentation techniques to increase the number of training samples and enhance the robustness of our model. Finally, the comprehensive dataset, which consists of six different tracks, would be divided at a ratio of 70:20:10 to conduct model training to achieve our project objective, such as cracks and potholes detection. Table 4 lists various datasets collected, detailing the number of classes, the quantity of images per class, the number of annotations, and the intended purpose for each dataset.

Table 4

List of Datasets Used

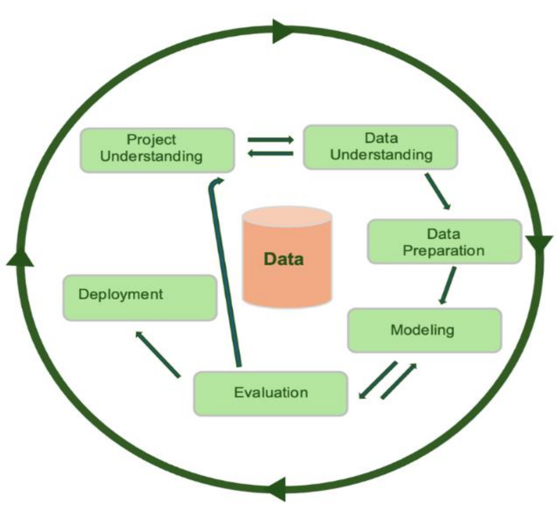
Dataset	Raw Images	Class	Number of Annotation	Data Preparation		
				Train	Validate	Test
UAV Dataset for Garbage Detection	1019	Garbage, litter, illegal dumping	917	713	204	102
UAV Dataset for Cracks	1378	Cracks	1240	965	274	138
UAV Custom dataset for Animal Detection	201	Cattle, Wild-life, Dogs	181	141	40	20
UAV Custom dataset for Construction sites near roads	110	Roads, highways, bridges under construction, under constructed buildings	99	77	22	11
Videos-Frames for crashed car detection	300	Crashed Cars	270	210	60	30
UAV dataset for Pothole Detection	998	Potholes, cracks	898	699	199	100

2.2 Project Development Methodology

Data analytics with intelligent system development life cycle, the goal of our project is to develop a UAV-based road inspection system for smart cities. Figure 1 shows the life cycle designed to perform data analytics and address our target problem. The process is divided into six phases: project understanding, data understanding, data preparation, modeling, evaluation, and deployment. We aim for our system to be capable of identifying road cracks and potholes, as well as detecting traffic hazards such as constructions, debris from car crashes, and dead animals on the roads. After understanding our target problem, we move on to data collection and quality checking to ensure that the datasets meet the project requirements. Once the dataset is finalized, we ensure that the data is well-prepared for modeling by applying data cleaning, normalization, and other preprocessing techniques. Then, we train and tune the model, comparing and evaluating different models. Finally, we select the best-performing model that meets our objectives for deployment. The entire process is designed in a cyclical loop to ensure that our system aligns with the expectations of the project.

Figure 1

Intelligent System Development Life Cycle



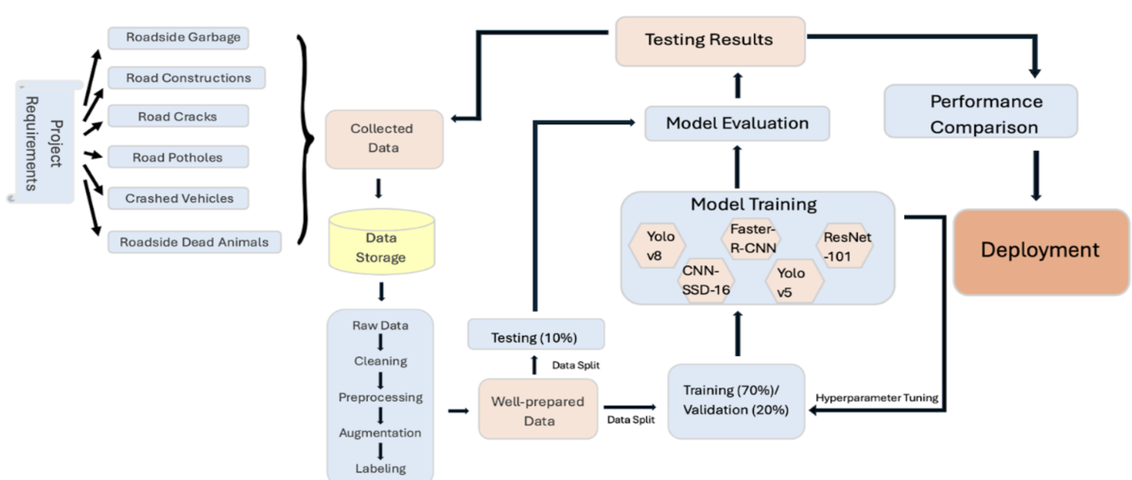
The following flow chart Figure 2 outlines our system development plan. First, we conduct data exploration based on our project requirements. We gather UAV-based images from various sources, including road potholes and cracks, roadside incidents like dead animals, illegal dumping, road constructions, and debris from car crashes affecting traffic. Then we combine all datasets into one comprehensive dataset, performing data essential preprocessing steps and labeling to ensure the data is suitable for modeling.

Next, we split the dataset into training, validation, and testing sets using a ratio of 70:20:10. We plan to employ popular object detection models such as YOLO and Faster R-CNN for data training based on our literature and technology survey. Hyperparameter tuning would be performed on the validation set to validate and optimize model performance.

Once we achieve satisfactory training results, we evaluate the unseen testing dataset. The testing outcomes are stored in our cloud platform, and after thorough performance comparison, the best-performing model will be selected and deployed.

Figure 2

Project Development Processes



2.3 Project Organization Plan

The project organization plan consists of 5 phases which are Project understanding, Data preparation, Modeling, Evaluation and Deployment.

In the first phase, all team members are fully involved in comprehending the project aims, deployment setting, and outlining project objectives. Comprehensive investigation is carried out to examine existing academic literature and technical resources related to the project's range. Following this, a detailed project plan is carefully developed to set up a definite timetable and allocate specific roles to each team member.

During the project's second phase, the team initiated an extensive data gathering initiative. Each team member was assigned to gather different datasets related to specific categories relevant to the project goals. Allocating various categories among team members ensured a wide-ranging and thorough compilation of data. After creating custom datasets, they were carefully merged into a unified comprehensive dataset.

Moving on to the third phase of the project, which is the modeling stage, we have carefully chosen multiple models for our analysis and experimentation. As outlined in Table 2, our selected models consist of YOLOv5, YOLOv8, Faster R-CNN, ResNet-101, and CNN-SSD-16. The decision to choose these models was based on insights gathered from an extensive review of Literature and Technical survey. Our selection process took into consideration the capabilities and characteristics of these models in addressing the objectives of our project effectively. By focusing on these specific models, we aim to optimize the efficiency and effectiveness of our modeling efforts with a view towards successfully achieving our project goals.

In the fourth phase, once satisfactory training outcomes are attained, the unseen test set is assessed to evaluate the performance of the model. We will examine the test results and compare how various models perform and supervise the storage of testing results on a cloud platform for future reference and analysis.

The last stage, Deployment phase, involves selecting the most effective model after conducting a comprehensive performance comparison. Implementation of the chosen model, guaranteeing smooth integration with the cloud platform will be executed and we will verify the system's effectiveness in practical situations post-deployment and offer insights for future enhancements.

2.4 Project Resource Requirements and Plan

Deep Learning models like Yolov8, Yolov5, Faster-R-CNN and ResNet-101 model training can be done on CPU as well as GPU. Training these deep learning models on GPU has a unique advantage, that the time taken to train these models are significantly reduced as compared to CPU as the training can be done in parallel using CUDA for hardware acceleration. The only disadvantage towards usage of GPUs is the cost associated with it which is almost 3x the cost of training on CPUs. To aid in high performance training, we will be using Google Collaboratory Pro which is a cloud service provided by Google Cloud Platform (GCP) that offers GPU and TPU acceleration for running Jupyter notebooks. We plan to store our training images and videos on Google Drive which will be free with SJSU student ID and, we will be using AWS S3 for scalable cloud infrastructure and a wide range of cloud services. Google Drive integrates well with Google Collaboratory, and it becomes easy to mount the directory from the Jupyter Notebook.

We will use Python 3.11 as our primary coding language as we have various python packages like PyTorch 2.2 for model development and training and we will have a web application which will be built using Flask 3.0.2 which is a lightweight framework for building web applications in Python.

Since we are already using resources from Google Cloud Platform like Google Collaboratory Pro, we plan to host this web-based application on Compute Engine which offers virtual machine (VM) instances which can be used for hosting web applications and for persisting data using MongoDB Atlas database. By comparing the available resource providers, we make a list in Table 5, of the required resources as follows:

Table 5

Project resource requirements and plan

Function	**Resource Type**	**Resource**	**Time requirement**	**Cost**
Training and testing model with GPU	Hardware	Google Collaboratory Pro with GPU (NVIDIA® A100)	2/10/2024 - 4/20/2024	$1000 ($1.4 per GPU/hour, we may need up to 5 GPU instances)
Web service hosting	Hardware	GCP Compute Engine instance with 64GB memory	4/1/2024 - 5/13/2024	$800 ($0.56/per instance/hour)
Training Data Storage	Software	Google Drive and AWS S3	2/10/2024 - 4/20/2024	Google Drive - Free with SJSU student ID AWS S3 - $0.023

				per GB - 1st 50 TB
Database management for web service	Software	MongoDB Atlas	4/1/2024 - 5/13/2024	Free
Deep Learning Frameworks	Software	PyTorch 2.2	2/10/2024 - 4/20/2024	Free
Coding	Software	Python 3.11 Flask 3.0.2	2/10/2024 - 5/13/2024	Free

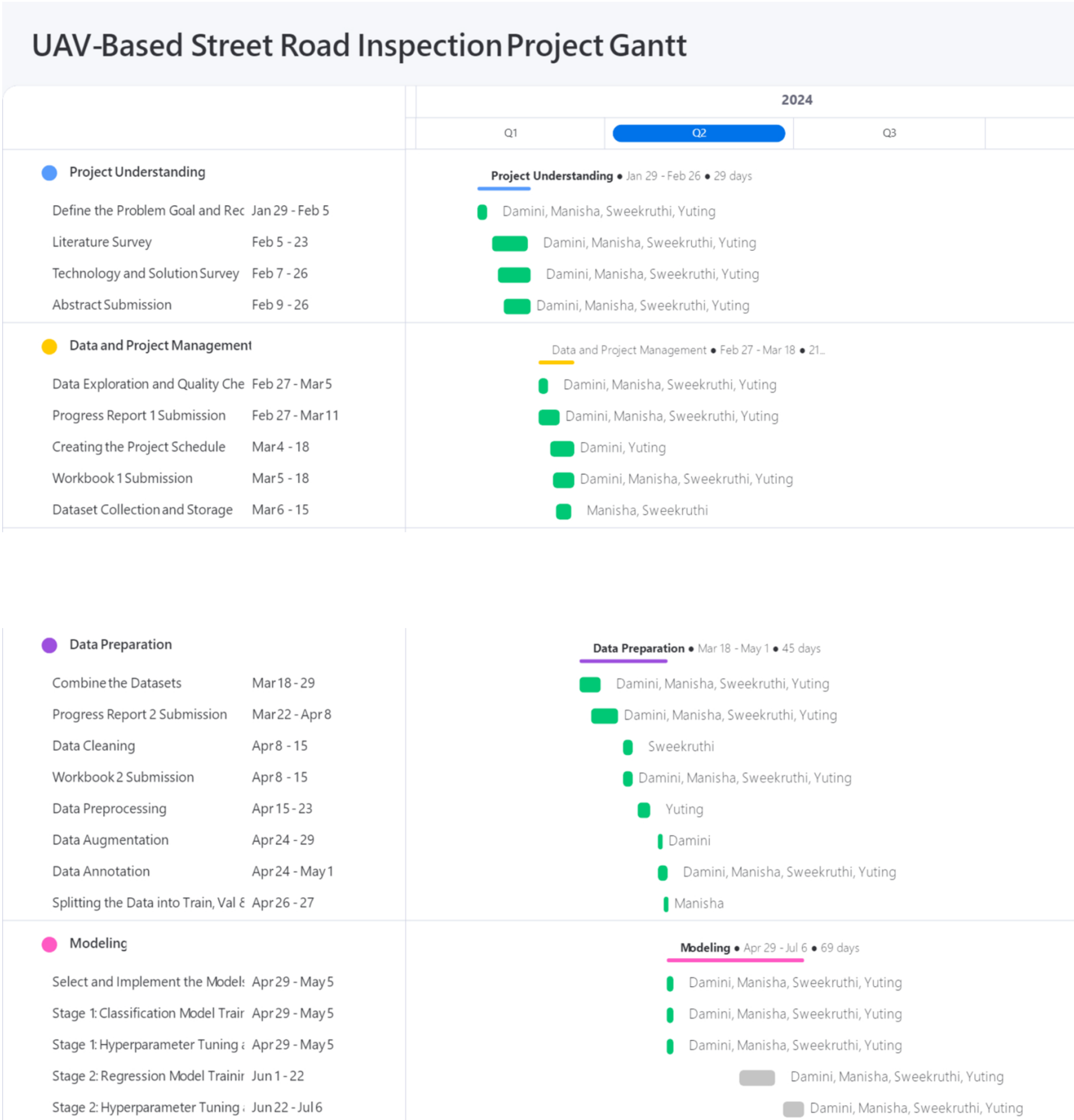
2.5 Project Schedule

We created a Gantt chart, shown in Figure 3, to ensure the project is well organized and each team member is well-informed about the planning and schedule of our projects. On the left side of the Gantt chart, a list of tasks is displayed, each with a clear timeline. On the right side of the chart, the responsible team member for each task is indicated at the end of the corresponding bar. Also, the status of deliverables is color-coded: green represents tasks that are completed, yellow represents tasks that are currently being worked on, and gray represents tasks that have not yet been started.

To better display the dependencies among each task, we have created a PERT chart. Each individual task is clearly defined with start dates, end dates, and duration. Critical paths, which are essential for developing our system, are indicated in red solid lines. Normal steps are represented by black dashed lines. The details of the PERT chart are shown in Figure 4.

Figure 3

Gantt Chart


UAV-Based Street Road Inspection Project Gantt
2024
Q1
Q2
Q3
Project Understanding
Define the Problem Goal and Rec
Jan 29 - Feb 5
Literature Survey
Feb 5 - 23
Technology and Solution Survey
Feb 7 - 26
Abstract Submission
Feb 9 - 26
Project Understanding • Jan 29 - Feb 26 • 29 days
Damini, Manisha, Sweekruthi, Yuting
Data and Project Management
Data Exploration and Quality Che
Feb 27 - Mar 5
Progress Report 1 Submission
Feb 27 - Mar 11
Creating the Project Schedule
Mar 4 - 18
Workbook 1 Submission
Mar 5 - 18
Dataset Collection and Storage
Mar 6 - 15
Data and Project Management • Feb 27 - Mar 18 • 21...
Damini, Yuting
Manisha, Sweekruthi
Data Preparation
Combine the Datasets
Mar 18 - 29
Progress Report 2 Submission
Mar 22 - Apr 8
Data Cleaning
Apr 8 - 15
Workbook 2 Submission
Apr 8 - 15
Data Preprocessing
Apr 15 - 23
Data Augmentation
Apr 24 - 29
Data Annotation
Apr 24 - May 1
Splitting the Data into Train, Val &
Apr 26 - 27
Data Preparation • Mar 18 - May 1 • 45 days
Sweekruthi
Yuting
Damini
Manisha
Modeling
Select and Implement the Model:
Apr 29 - May 5
Stage 1: Classification Model Trair
Apr 29 - May 5
Stage 1: Hyperparameter Tuning
Apr 29 - May 5
Stage 2: Regression Model Trainir
Jun 1 - 22
Stage 2: Hyperparameter Tuning
Jun 22 - Jul 6
Modeling • Apr 29 - Jul 6 • 69 days

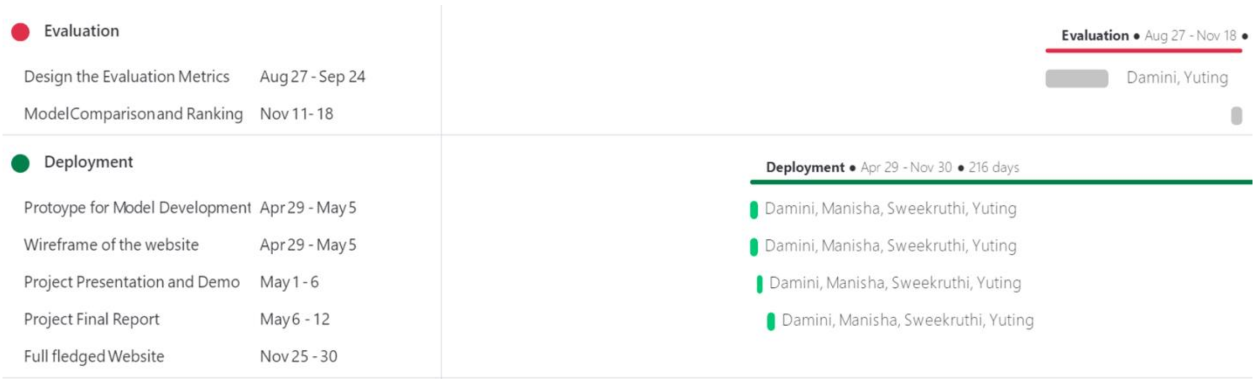
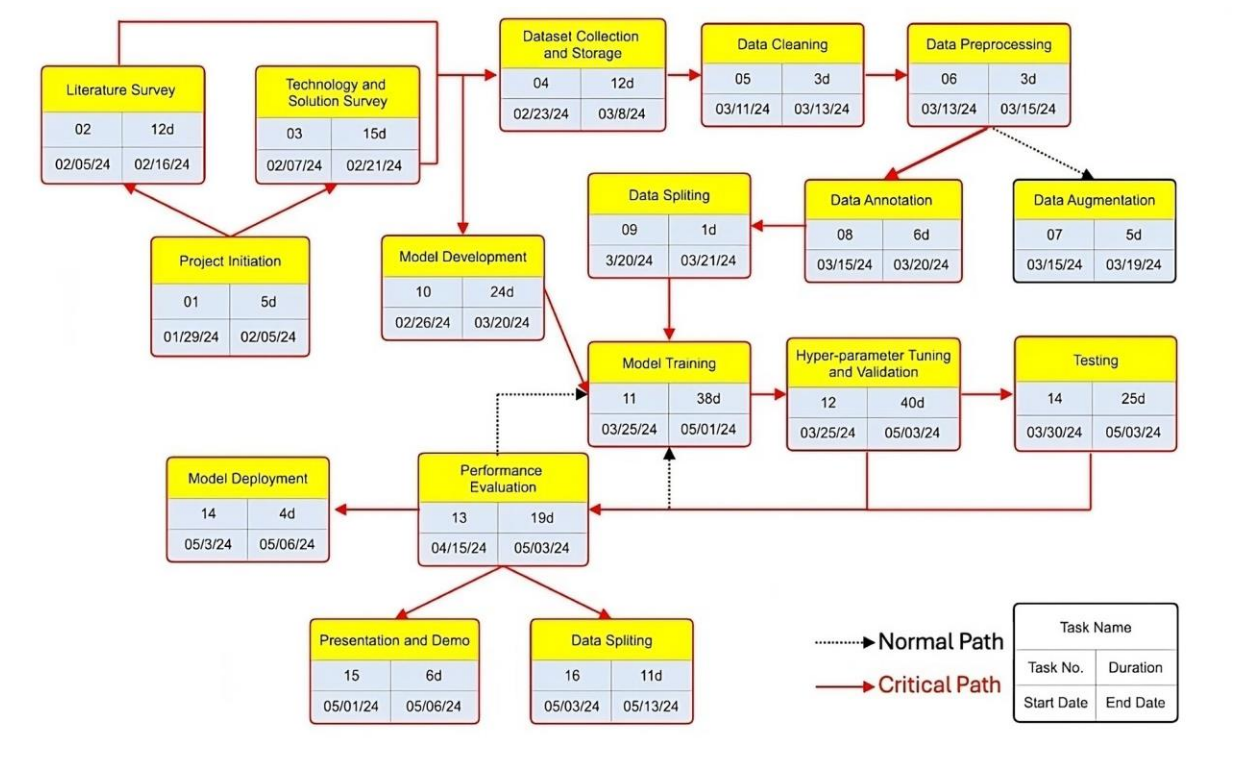


Figure 4

PERT Chart



3. Data Engineering

3.1 Data Process

As our UAV-based system is designed to support transportation by providing various functions, we have collected our datasets from six directions, including road anomalies such as cracks and potholes detection, illegal dumping detection, real time vehicle accident alerting, animal detection and construction activity monitoring. In this section, we provide an overview of the methods we employed to acquire the qualified data.

For road distress detection, there are numerous public datasets available online. To enhance the capability of our system in the detection of pavement anomalies, we have included various crack types in our selection. Figure 5 displays the cracks we collected: longitudinal, transverse, oblique, and alligator cracks. Since potholes pose a more severe problem in terms of road safety, and successfully identifying the sizes, depth and shapes of potholes help in prioritizing road repairment plans. Our collected pothole images exhibit a wide range of diversity. Figure 6 shows potholes with different characteristics. Also, we have considered the impact of different road surfaces on detection effectiveness, so our dataset consists of images captured on highways, urban roads, and county roads.

Figure 5

Different Types of Cracks

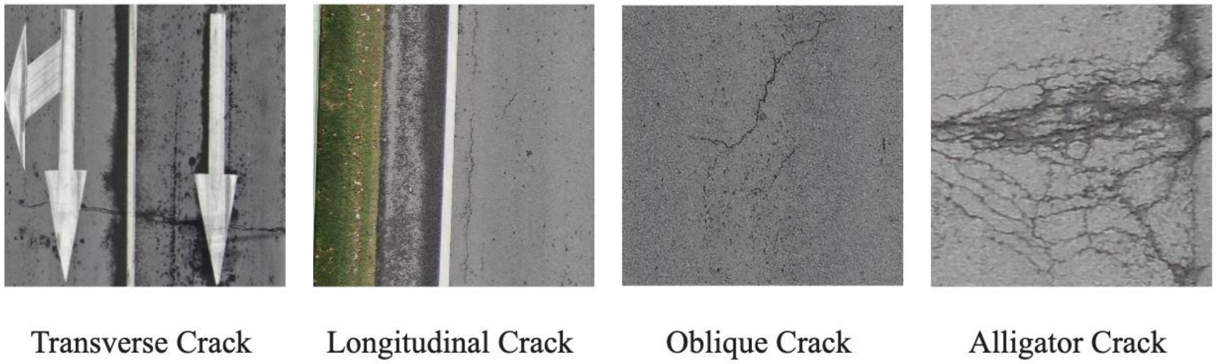
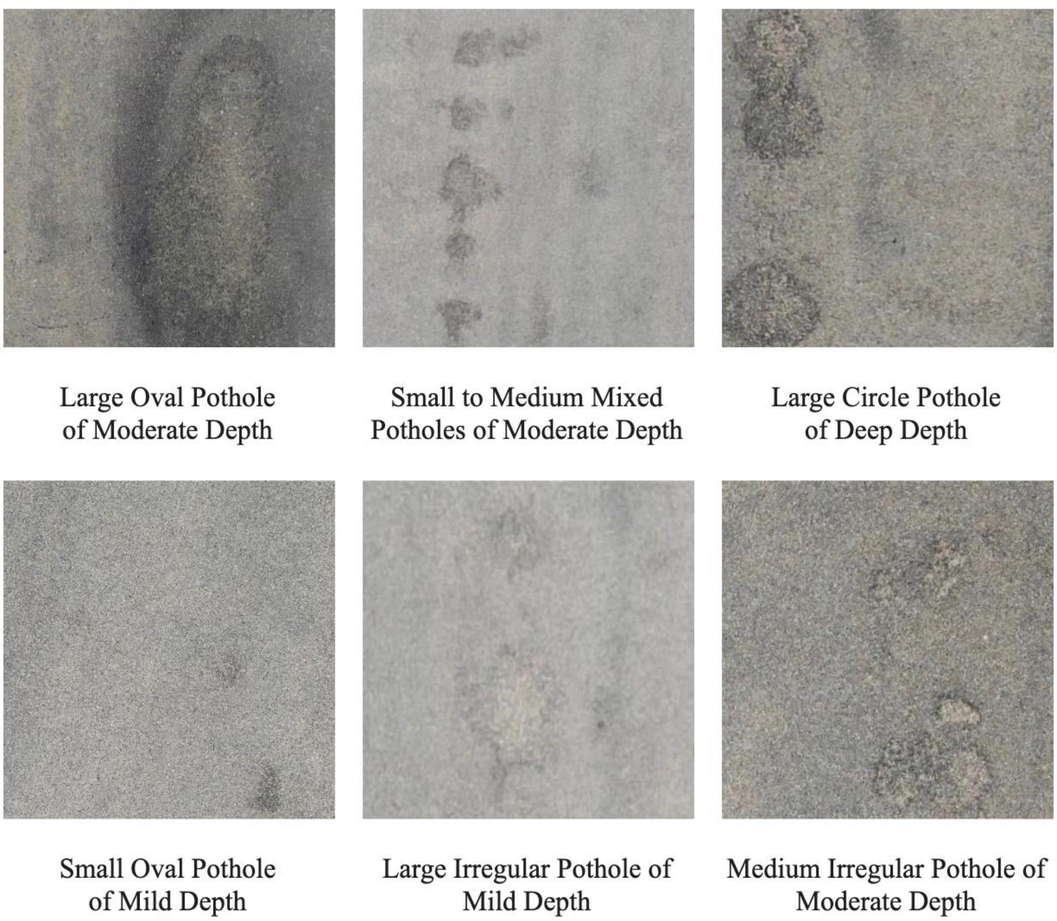


Figure 6

Potholes with Different Categories



To tackle the problem of illegal dumping on roads, we take the range of dumping severity into account as well. We carefully compile a dataset including images of varying degrees of illegal dumping. Figure 7 illustrates the level of the illegal dumping, from left to right. The left image shows only scattered plastic bottles along the trash bin. The middle one displays a medium-sized heap of garbage, while the right one reveals a large area covered with trash.

Figure 7

Different Levels of Illegal Dumping

There are limited open datasets containing categories such as dead animals and construction available in our literature research, so we meticulously gather individual images accessible online one by one. Simulated images are collected in these two categories as well as it is challenging to find sufficient real image samples.

Lastly, to address the real-time vehicle accident detection challenge, we collected drone videos captured and shared by various drone and news companies. However, drones are usually dispatched for severe and fatal accidents. For minor car crashes where no lives are lost, the possibility of acquiring real drone footage is limited. So, we also collected manually created simulated videos that depict the post-accident scene of minor vehicle accidents, filling the gap for such scenarios. In summary, the steps of data collection process can be concluded as:

- Search and verify the availability of publicly accessible real-world image datasets for each category based on our requirements.

- Dig into the sources of datasets used in existing popular papers and message the authors for accessibility when possible.
- Collect drone videos for certain categories such as car accidents and convert the videos into frames.
- Search for simulated data for specific categories which lack sufficient raw data samples such as dead animals and construction activities.
- Check the quality of the data and filter out any poor-quality samples.
- Organize all images into different category folders under one folder to create a comprehensive dataset for this project, stored in our shared team Google Drive.

Training, validation, and testing data are prepared in different ways based on our project and candidate model requirements. For the Yolo-based model, Yolo requires images coming with boxing boxes as labels to accomplish the training part. Part of our datasets including .txt label files, for the others, the bounding boxes are manually created on the raw images using tools such as LabelImg.

On the other hand, models such as Faster R-CNN can extract bounding box coordinates from .xml files, which is typically included in public UAV-based datasets. We separated the .xml file and its corresponding images into training and validation and testing folders respectively. However, for our custom dataset, annotation is necessary as a prerequisite for modeling.

All The images are categorized into five groups, each of which is subdivided into training, validation, and testing sets in a ratio of 70:20:10. The images and their respective labels in three sets are paired and well organized to serve the purpose of training the model, parameter fine tuning and validation or model performance testing and evaluation.

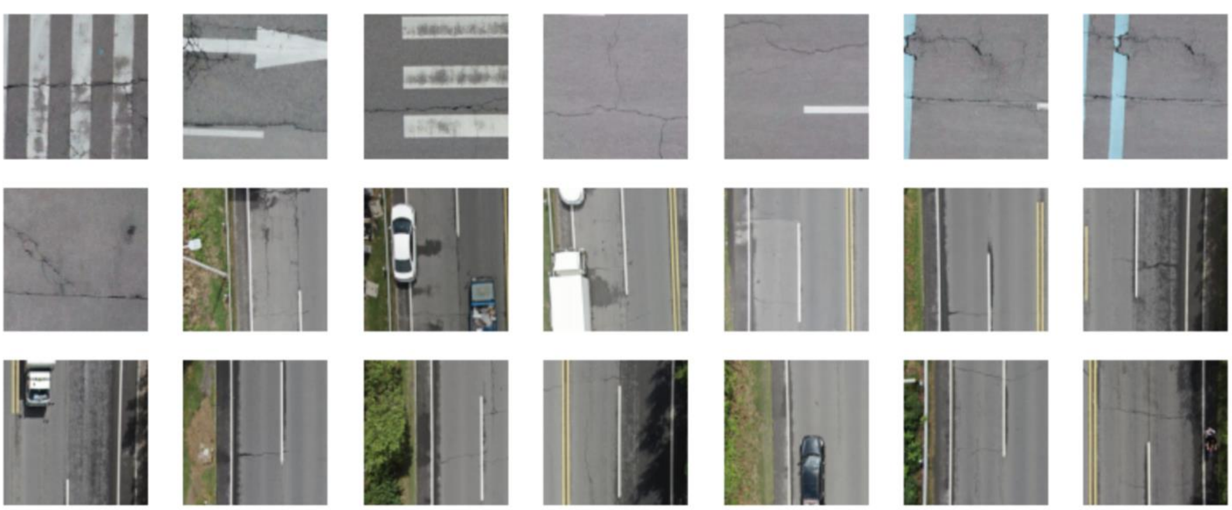
3.2 Data Collection

Our datasets primarily come from three different sources. Firstly, we collect existing public online open-source image datasets. Secondly, we acquire drone videos shared on platforms like YouTube by drone companies or news publishers. Lastly, we compile our custom dataset by gathering data from various websites.

The first public dataset we utilize is UAV-PDD2023 dataset, consisting of five types of pavement distress samples, including longitudinal cracks, transverse cracks, oblique cracks, alligator cracks and potholes. The images are captured by a UAV at an altitude of 30m in China. A total of 11,154 instances in 2,439 images are collected in this dataset. Another dataset we collect for road inspection is a pavement crack image dataset from UAV imagery. A total of 1388 pavement crack images were collected and labeled in this dataset, containing 304 samples classified as LC, 303 samples as TC, 313 samples as OC, 368 samples as AC, and 100 samples as no-crack type. Figure 8 displays samples from the combination of the two datasets mentioned previously.

Figure 8

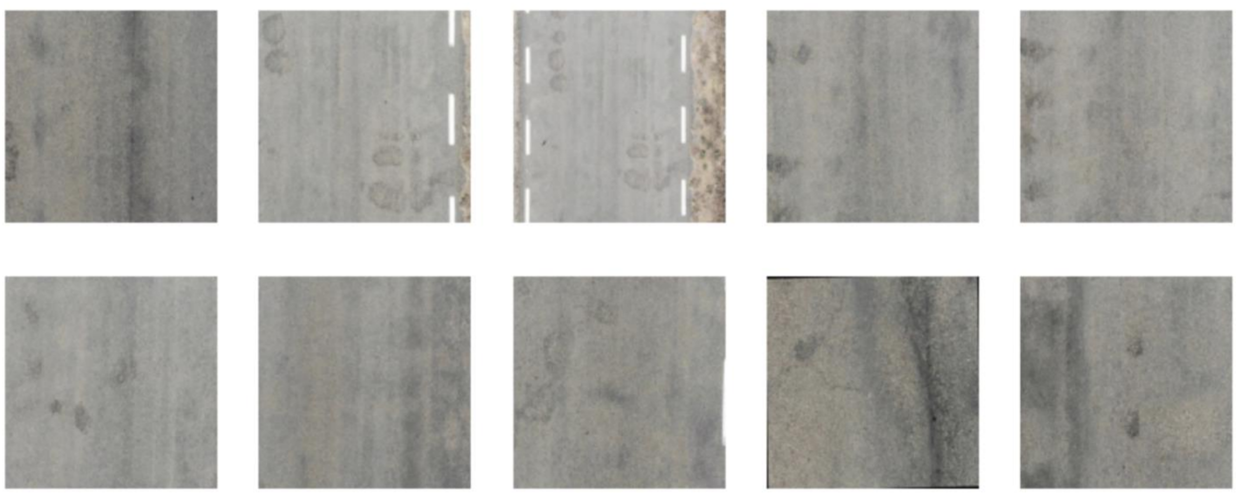
Samples of Cracks Dataset



Furthermore, we got an image dataset on roboflow specifically for pothole detection called PDS-UAV Image Dataset, which includes 1534 images for training, 179 images for validation and 136 images for testing. Another dataset for pothole detection called B21-CAP0388 created by Anugrah Akbar Praramadhan, consisting of 3125 images for training and 843 for testing. Figure 9 shows the positive samples from the combination of all the dataset containing potholes.

Figure 9

Samples of Potholes Dataset



For the illegal dumping problem, we have constructed a mixed dataset containing images from both publicly available datasets and images downloaded through Google searches. The public garbage dataset named UAVVaste is used in paper, consisting of 772 images and 3718 instances. Figure 10 illustrates the samples from our illegal dumping dataset.

Figure 10

Samples of Illegally Dumped Garbage

All the public and custom datasets are rearranged into 6 categories and details are displayed in Table 6.

Table 6

Summary of Datasets

Dataset Source &Type	**Category**	**Class**	**Quantity**		**Annotated**	**Samples**
			Partial	**Total**		
Public Image Dataset	Cracks	Longitudinal Crack	1209	5047	Yes	Figure 4
		Transverse Crack	1187			
		Oblique Crack	1043			
		Alligator Crack	1013			
		No Crack	595			
Public Image Dataset	Potholes	Small Potholes	1063	3458	Yes	Figure 5
		Medium Potholes	947			
		Large Potholes	1098			

		No Potholes	350			
Mixed Image Dataset	Illegal Dumping	Low Level	928	3064	Yes	Figure 6
		Medium Level	1127			
		High Level	1009			
Custom Image Dataset	Road Animals	Dogs	192	2040	No	Figure 7
		Cattle	519			
		Wildlife	1329			
Custom Image Dataset	Construction Activities	Road Construction	756	2224	No	Figure 8
		Highway Construction	802			
		Bridge Construction	321			
		Roadside Building Construction	345			
Drone Videos	Vehicle Accidents	Severe Accident	5 Videos	9 Videos	No	Figure 9
		Minor Accident	4 Videos			

Figure 11 displays samples from custom animal image datasets, showing dead animals like dogs, cattle walking on the road, as well as wild animals such as elephants and kangaroos.

Figure 11

Samples of Wildlife

Figure 12 displays the samples from custom construction activities image datasets, including the constructions on different locations.

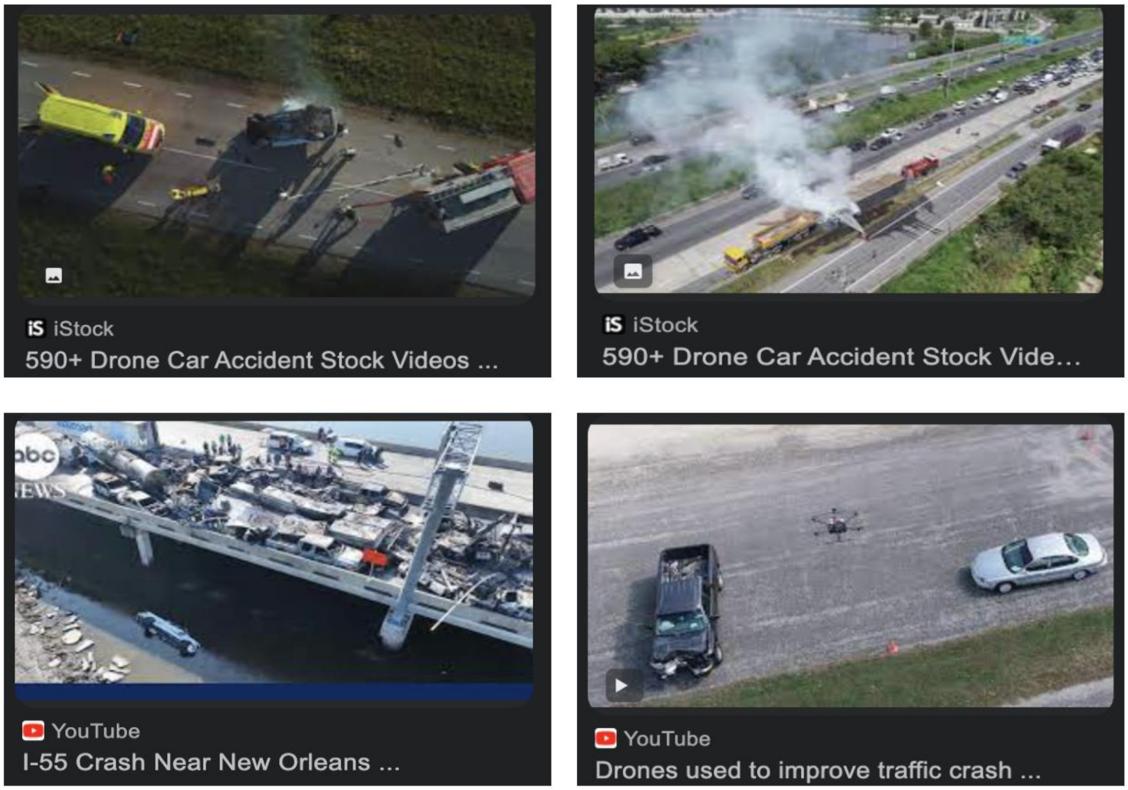
Figure 12

Samples of Construction Activities

Figure 13 showcases the sample videos we found from some platforms.

Figure 13

Video Samples of Accidents



3.3 Data Preprocessing

Since we carefully selected our images during the data collection process, few steps of data preprocessing are required at this stage. We double checked the quality of the images, excluding those images with low resolution or too much noise, ensuring all the images are non grey scale. As our datasets are sourced from various sources, the image sizes vary. Considering we will use Yolo as part of our model, as YOLO requires a minimum image size of 416 x 416, we have resized all images to 512 x 512 pixels to ensure compatibility with our model. Also, we convert the images to the proper tensor format as we plan to utilize TensorFlow or PyTorch to develop our model. Figure 14 shows the code how we implement these preprocessing steps.

Figure 14

Script for the Resizing of Images

```
from torchvision import transforms
from torchvision.datasets import ImageFolder
from torch.utils.data import DataLoader, random_split
data_transform = transforms.Compose([
    transforms.Resize((512,512)),
    transforms.ToTensor()])
dataset = ImageFolder(data_dir,data_transform )

for data in dataset:
    print(data[0].size())
    break

torch.Size([3, 512, 512])
```

For video data, directly converting entire videos could lead to an excessive number of frames and too many repeated images, resulting in computational and storage costs as well. So, we manually clipped segments from the original videos and converted them into frames using OpenCV and stored them in our google drive for further analysis and modeling.

Figure 15 showcases a series of images obtained by converting several drone videos of car accidents into frames.

Figure 15

Series of Images of Car Accidents

Figure 16 displays the code script of how we handle the videos.

Figure 16

Script for Handling the Videos Data

```
import cv2
#Set Keyframes Per Second
KPS = 15
# specify the path of the video
VIDEO_PATH = "/content/drive/MyDrive/Data298A/car crash.mov"
# specify the path of the image frames
IMAGE_PATH = "/content/drive/MyDrive/Data298A/298A datasets/Vehicle Accidents"
cap = cv2.VideoCapture(VIDEO_PATH)
fps = round(cap.get(cv2.CAP_PROP_FPS))
# exit()
hop = round(fps / KPS)
curr_frame = 0
# success is a boolean value, when success is true, meaning frames are successfully read from the videos
success, frame = cap.read()
while success:
    # limit the number of frames
    if curr_frame % hop == 0:
        # imwrite() saves the image with specified name and path, %d would be replaced with count value
        cv2.imwrite(IMAGE_PATH + "frame%d.jpg" % curr_frame, frame)
    success, frame = cap.read()
    curr_frame += 1
```

3.4 Data Transformation

One main technique that we used as part of data transformation is image data augmentation. We have considered the significance of quality and diversity in our training dataset to enhance our model performance and to generalize well. Our aim is to augment the size, variety of the training dataset by transforming our data into various versions and to achieve this, we planned to apply various data augmentation techniques to the original 6 classes of dataset that we have. These augmentation operations that we have chosen to align with our use case. For example, the real time input data may arise from different angles and lighting conditions and to simulate these conditions, we have employed few transformations techniques using the Keras Deep Learning Library's ImageDataGenerator class stated as below:

- Rotation: This operation gives us a view of images to various degrees which can simulate different orientations in which the data might be captured.
- Brightness Adjustment: By specifying a brightness range of [0.2,1], we simulate

different lighting conditions, ensuring the model's robustness to variations in brightness levels.

- Translation: Shifting images vertically and horizontally can mimic changes in perspective or position, contributing to a more varied dataset.
- Scaling: This operation resizes images to different dimensions and can simulate variations in distance from the device, adding further diversity to the dataset.
- Zooming: This operation adjusts the zoom level of images and can simulate varying focal lengths of the camera lens.
- Noise Injection: Adding random noise to images can enhance the model's ability to handle noise in real-world data.

These transformations that are implemented within the ImageDataGenerator class help us enrich the training dataset and improve the model's ability to generalize across different environmental conditions. Below, Figure 17 shows code snippets and Figure 18 shows some samples of augmented images.

Figure 17

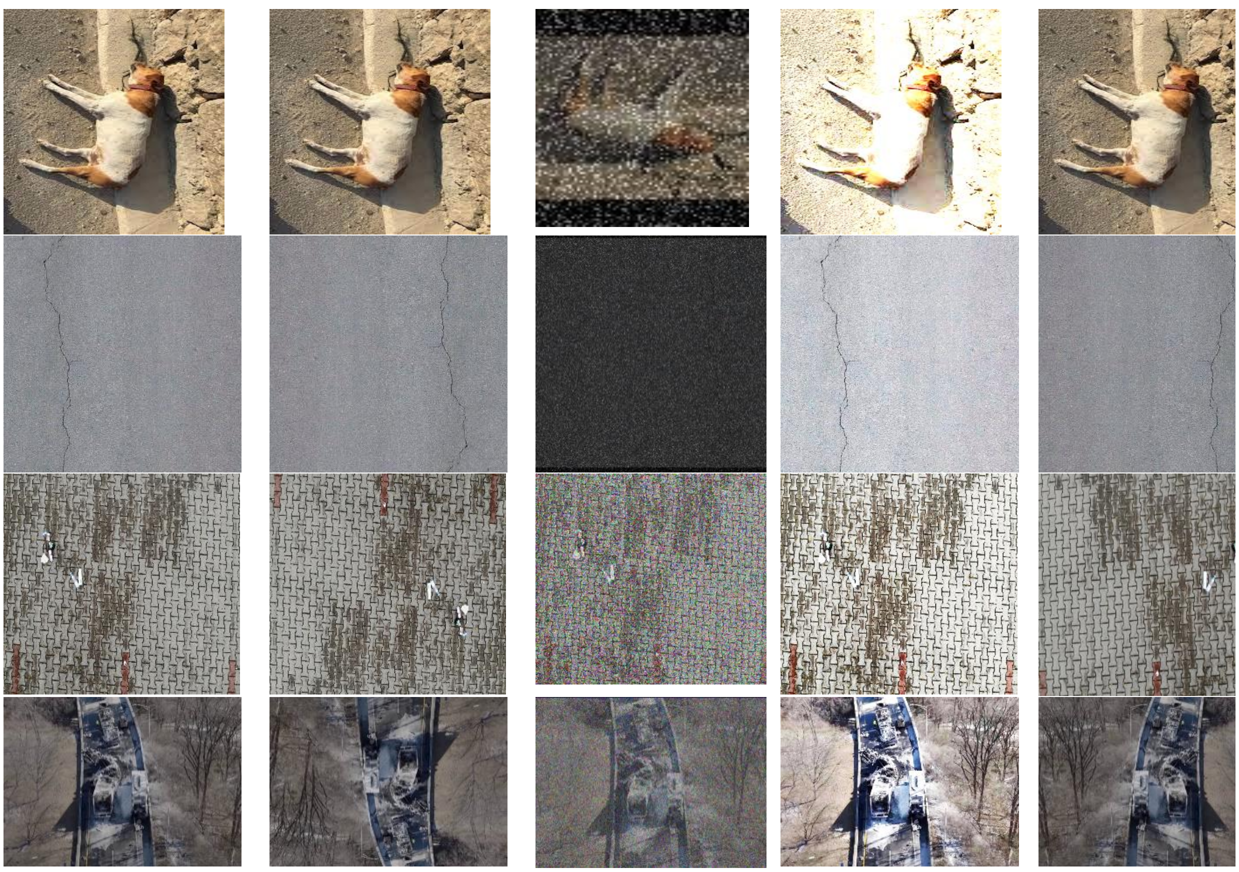
Script for Image Augmentation

```
datagen = ImageDataGenerator(
    rotation_range=180,
    brightness_range=[0.2, 1.0],
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    vertical_flip=True,
    fill_mode='nearest'
)
```

```
train_generator = datagen.flow_from_directory(
    'training_data',
    target_size=(img_height, img_width),
    batch_size=batch_size,
    class_mode='binary'
)
datagen.fit(train_generator)
model.fit_generator(
    train_generator,
    steps_per_epoch=num_train_samples // batch_size,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=num_validation_samples // batch_size
)
```

Figure 18

Some Augmented Images for each Category



3.5 Data Preparation

We gathered all images and videos data through various sources. Videos were further divided into image frames using OpenCV library from Python. We labeled the images and arranged them to further divide the final datasets into three parts: training, validation, and test sets. The training set we used to train the models, and the validation set will help estimate prediction error for model selection. The test set will then help us to evaluate the generalization performance of the chosen model. To split the data, we have used something called the ImageDataGenerator which will help us in organizing folders containing files, such as images, into separate sets. This package makes sure that there is randomness, and this is done by shuffling the input files across the three sets. Our target split is a ratio of 70% for training, 20% for validation, and 10% for testing and will be achieved using this package module. We are also paying careful attention towards annotation if it is not previously annotated. Figure 19 and Figure 20 shows samples of Training, Validation and Testing datasets respectively.

Figure 19

Sample Train & Validation Data for Classification

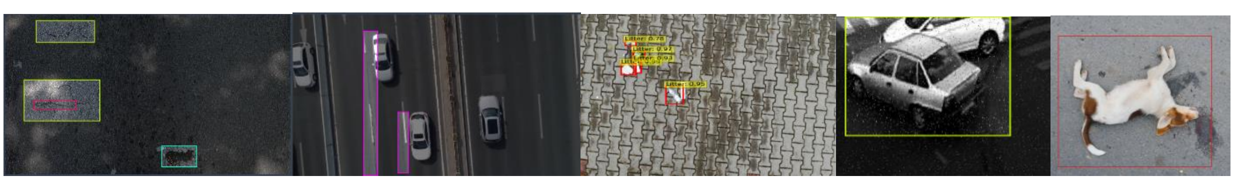


Figure 20

Sample Test Data for Classification

3.6 Data Statistics

Our complete dataset involves gathering raw data from various available open-source datasets. Through these processes, we have around 17000 images. To enhance the quality and diversity of our dataset, we have employed transformation techniques like data augmentation. This step significantly expanded our dataset, resulting in a comprehensive dataset containing 153000 images distributed across 6 classes. The breakdown of image counts for each feature is detailed in Table 7. With our prepared dataset, we proceeded to split it into in a ratio of 70% for training, 20% for validation, and 10% for testing.

Table 7

Datasets Splits

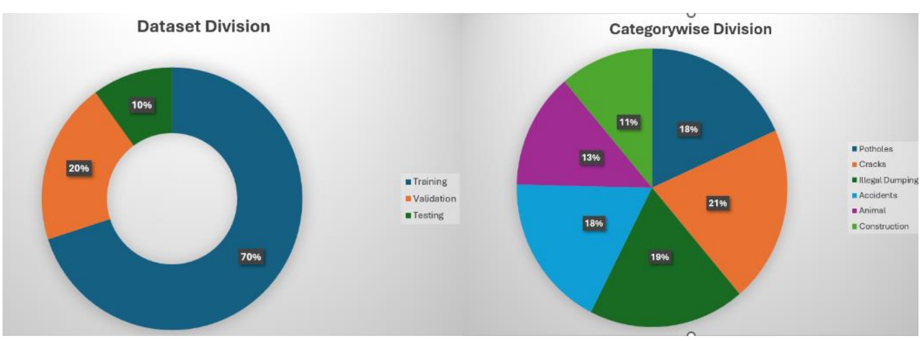
Dataset (UAV/Satellite)	Raw	Transformed	Prepared		
			Train	Validation	Test
Dataset for Cracks	5,047	45,423	31,796	9,085	4,542
Dataset for Potholes	3,458	31,122	21,785	6,224	3,113
Dataset for Illegal Dumping on roads	3,064	27,576	19,303	5,515	1,758

Dataset for animals found on road	2,040	18,360	12,852	3,672	1,836
Dataset for road construction activities	2,224	20,016	14,011	4,003	2,002
Dataset for Car accidents on road	1,023	10,057	7,390	2,111	1,056
Total	**17,006**	**153,054**	**107,137**	**30,610**	**15,307**

According to the above split we have saved all images in the Google Shared Folder as all team members can have access.

Figure 21

Percentage Breakdown for each Category



To get the best performance on our model we have made sure we create a good balanced dataset for each feature. Figure 21 shows the detailed percentage on each dataset.

3.7 Data Analytics Results

Building on the dataset preparation efforts outlined above, the project employed a structured data pipeline to optimize model performance. The augmentation of the initial dataset from 17,000 raw images to 153,000 enhanced images provided sufficient diversity for robust model training. Each class was equally represented, ensuring balanced data distribution, which is critical for unbiased machine learning outcomes. Figure 22-26 shows detailed data statistics for each of the obstacle category.

The dataset was divided into training, validation, and testing sets in a 70:20:10 ratio. This division enabled efficient model development: the training set was used to train and fine-tune the models, the validation set guided hyperparameter tuning and model selection, and the test set served to evaluate the final performance. Figure 27 shows the detailed number for each of the above divisions.

This rigorous approach to dataset preparation and analytics ensured high-quality input data, providing the foundation for the successful deployment and evaluation of the two-stage system. By addressing issues such as data diversity and class imbalance, the project established a reliable framework for achieving precision and efficiency in road anomaly detection and severity classification.

Figure 22

Detailed statistics of Cracks category

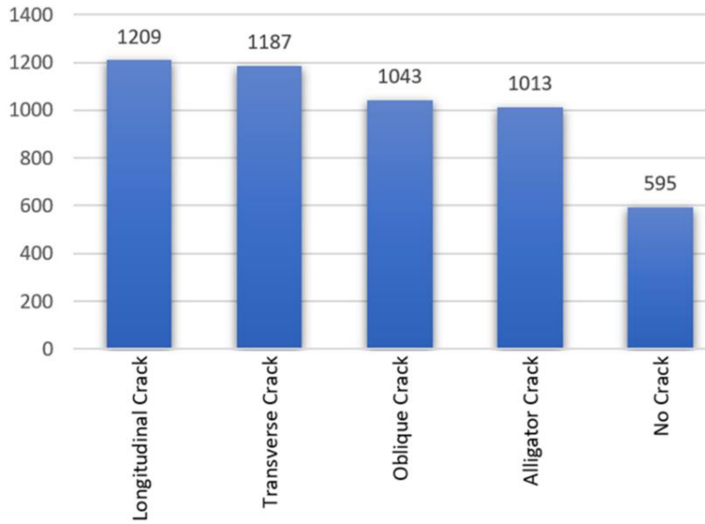


Figure 23

Detailed statistics of Potholes category

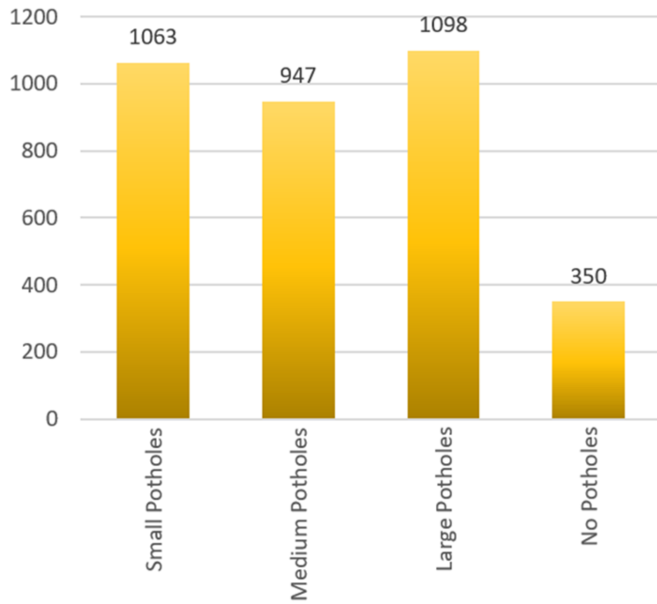


Figure 24

Detailed statistics of Wildlife category

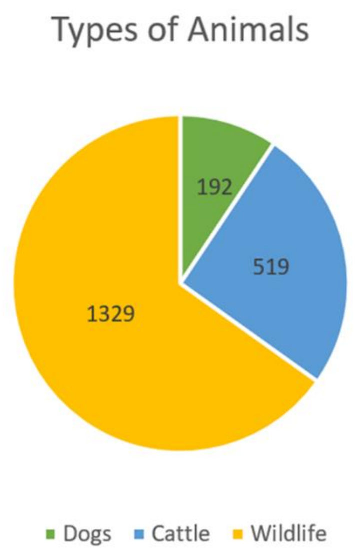


Figure 25

Detailed statistics of Construction category

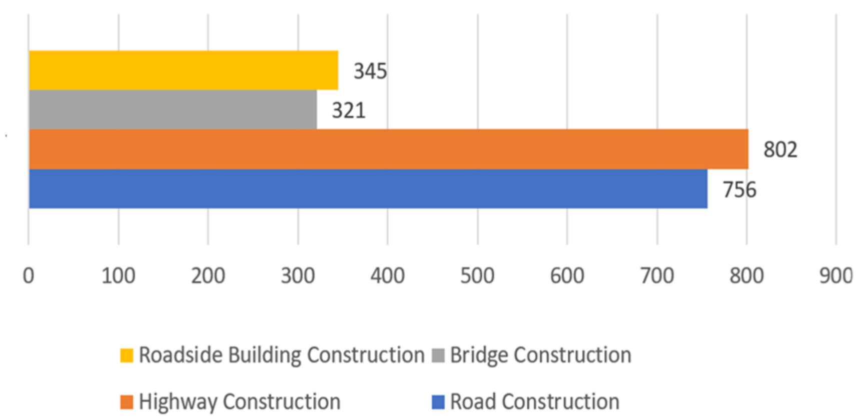


Figure 26

Detailed statistics of Illegal dumping category

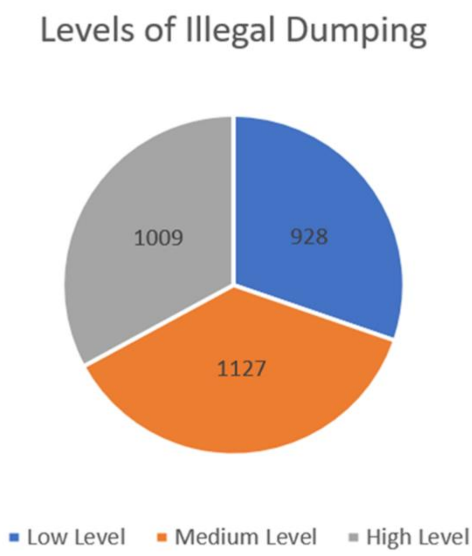
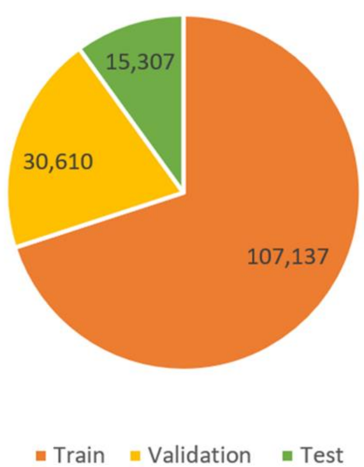


Figure 27

Data count breakdown across Training, testing and validation datasets



4. Model Development

4.1 Model Proposals

Improved YOLOv8

YOLOv8 (You only look once) represents variations of the YOLO algorithm, which are optimized for real-time object detection applications. In our project, these variants are valuable for identifying a range of objects and irregularities on roadways, including potholes, fractures, debris, and unauthorized waste disposal using UAV-derived imagery.

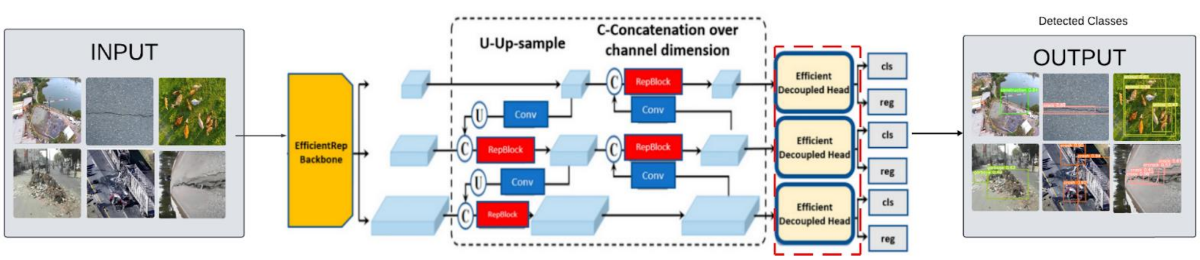
Improvisations in YOLOv8

Our improved YOLOv8 model is derived from the pretrained YOLOv8, originally trained on the COCO dataset with 80 classes and 2.5 million images. We have modified the head of the pretrained YOLOv8 to specifically fine-tune it for our object detection task, reducing the classes to six. The head now focuses on these classes, optimizing detection for our targeted objects.

The model retains the powerful backbone and uses a PANet (Path Aggregation Network) layer, enhancing multi-scale feature extraction for better object localization. This modification improves the model's performance, particularly for smaller objects, ensuring high accuracy in detecting our six specific classes. Figure 28 shows the improved architecture of improved Yolov8

Figure 28

Architecture of improved YOLOv8

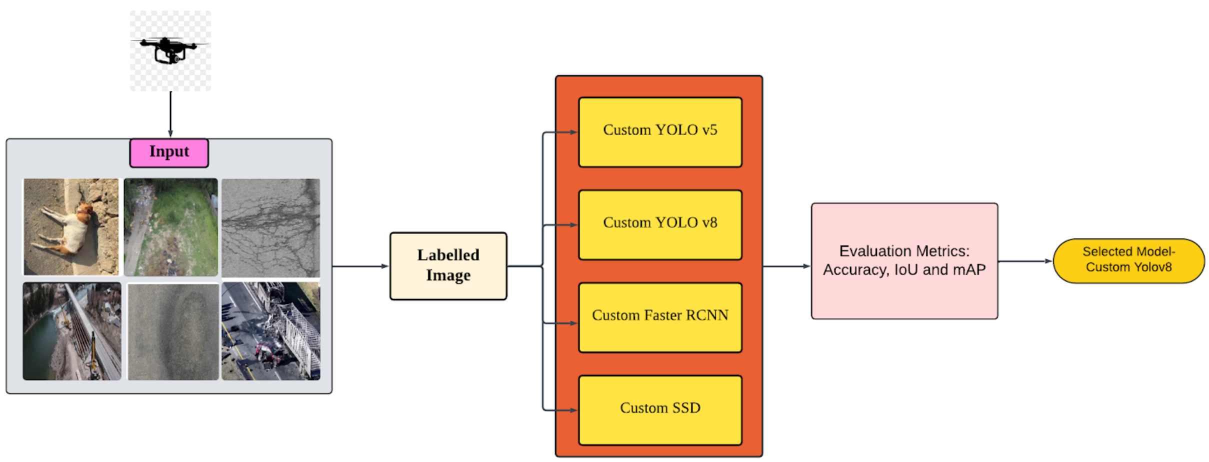


In our project, YOLOv8 is employed during the object detection phase to recognize and classify sources of congestion such as road irregularities, unauthorized waste disposal, incidents, and construction operations. By utilizing their live object detection features, we can effectively analyze image data captured by UAVs and make well-informed choices regarding transportation management.

Additionally, we have experimented with models like Faster RCNN, Yolov5 and MB1-SSD for object detection. Based on the comparison of evaluation results of these models we have selected Improved Yolov8 to be our object detection model. Figure 29 shows the model selection technique we have used in our project to select our final object detection model which performed extensively well on detecting all the categories with highest accuracy and mAP score.

Figure 29

Object detection model selection technique



Improved MB1-SSD

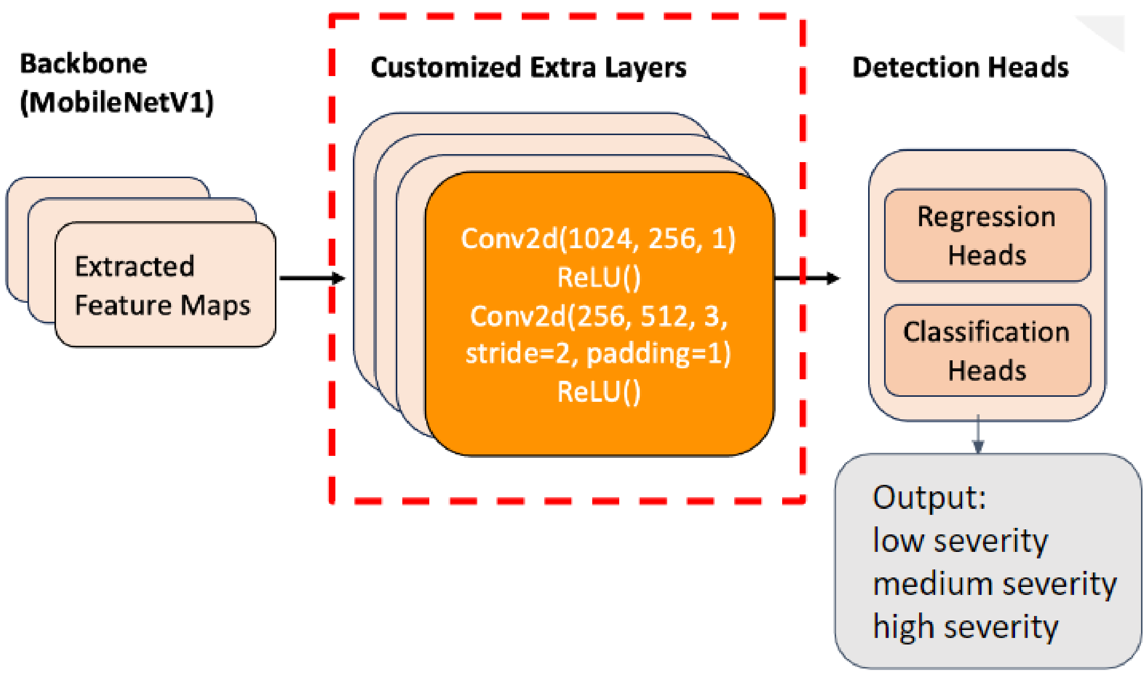
MB1-SSD a variant of the SSD. SSD is a deep learning model designed for efficient object detection in images. Unlike traditional methods that involve multiple stages, SSD performs both object localization and classification in a single forward pass of the network. This characteristic makes it well-suited for identifying road irregularities such as cracks, potholes, and other real-time applications where speed is crucial.

Due to MB1-SSD's lightweight design architecture, it's able to conduct rapid processing of images even on resource-constrained devices such as our local laptops. By leveraging the combination of MobileNet's feature extraction efficiency and SSD's single-shot detection mechanism, MB1-SSD achieves fast and precise object detection, as referred to in paper Liu, C et al. (2024). The model achieved 80.4% average precision with a detection speed of 34.01 frames per second (fps). demonstrating its effectiveness in identifying road irregularities. This capability is crucial for transportation management, as it enables timely detection and intervention to ensure road safety and maintenance. Figure 30 shows the architecture of MB1-SSD.

Improvisations in MB1-SSD

Figure 30

Improved Architecture of MB1-SSD



Improved Vision Transformer (ViT)

The Vision Transformer (ViT) model is a deep learning model designed specifically for image classification tasks. Unlike traditional convolutional neural networks (CNNs), ViT uses a transformer-based architecture. In pursuit of building a more efficient and innovative system, alternative models such as the Vision Transformer (ViT) have been explored. This exploration not only introduces a distinct approach to the problem but also allows for an investigation into the potential advantages of transformer-based architectures in image classification tasks. This direction aligns with the goal of enhancing the research beyond the common methodologies covered in the literature survey, particularly in leveraging transformers for complex visual tasks. Figure 31 shows the architecture of our Improved ViT model.

Improvisations in ViT Architecture

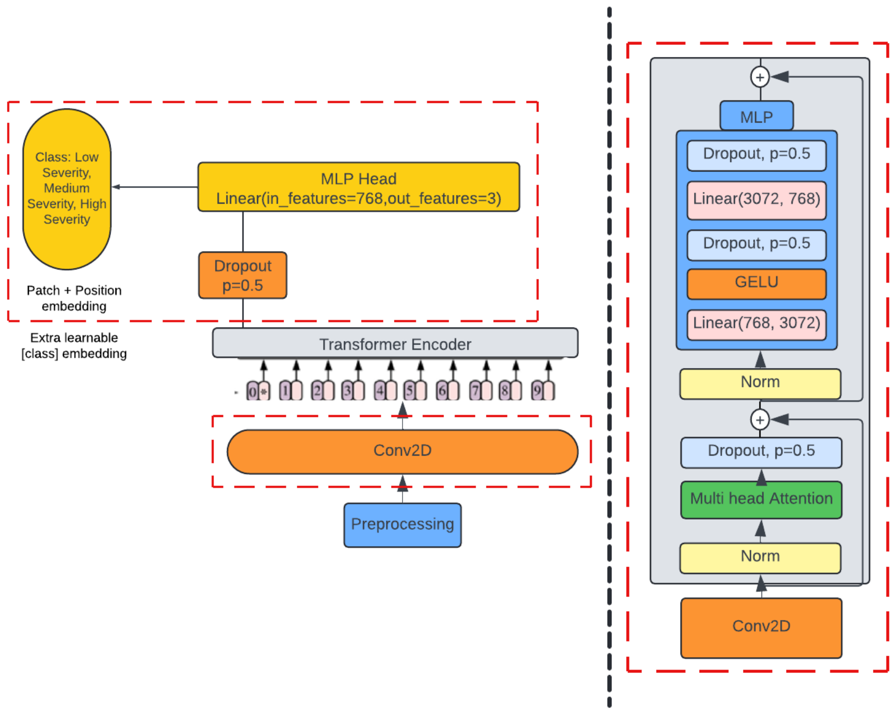
ViT model architecture shown in Figure 31 incorporates some key improvements which enhanced the model performance. One of the major modifications is the introduction of a Conv2D layer before the transformer encoder, which is responsible for pre-processing and extracting preliminary feature representations. This Conv2D layer allows the model to capture more localized spatial information from the input image, which is particularly useful for our severity classification task, where finer details could determine the class (low, medium, or high severity). Additionally, extra learnable embeddings for class identification are integrated along with the patch and position embeddings, providing the model with more contextual information to distinguish between severity levels.

Usage of multiple dropout layers throughout the architecture, aims at improving the generalization by preventing the overfitting. The MLP head at the final stage is designed to classify images into three severity classes and includes a linear layer that maps the 768-

dimensional features to the output space, with dropout to further ensure robustness. Inside the Transformer encoder, a multi-head attention mechanism is applied, followed by normalization and GELU activation for more efficient learning. Together, these changes result in a more robust and adaptable ViT model, specifically tailored for severity classification tasks.

Figure 31

Improved Architecture of Vision Transformer



Category-Specific Severity Classification Models

We have 6 obstacle-categories in our data which are Cracks, Garbage, Potholes, Animals, Construction and Vehicle Crashes. These are the major obstacle categories that can cause delays on roads. To effectively manage and mitigate these disruptions each of these obstacle types requires precise detection and a customized severity classification approach.

Our hybrid model architecture integrates an enhanced YOLOv8 in Stage 1 for object detection, where various obstacle categories in the input image are identified. If multiple categories are detected within a single image, the model prioritizes the category associated with the largest bounding box, ensuring the most significant obstacle is addressed first.

Once the object detection is completed in Stage 1, the system passes the input to a corresponding Stage 2 model, specifically tailored to the detected category. Each model in Stage 2 is meticulously customized and trained on data relevant to its specific category (e.g., cracks, potholes, vehicle crashes). This approach allows for fine-tuned severity classification, ensuring that each category is accurately assessed for its potential impact, whether it's low, medium, or high severity. The division into category-specific models enables a more precise and specialized analysis of each type of obstacle, optimizing both detection and severity evaluation. Figure 32 shows the sample pipeline along with model architecture for one of the categories. It explains how the data flows into the stages of our model architecture. Figure 32 shows the detailed architecture of our proposed model.

Figure 32

Sample model and data flow pipeline of the system shown for one of the categories

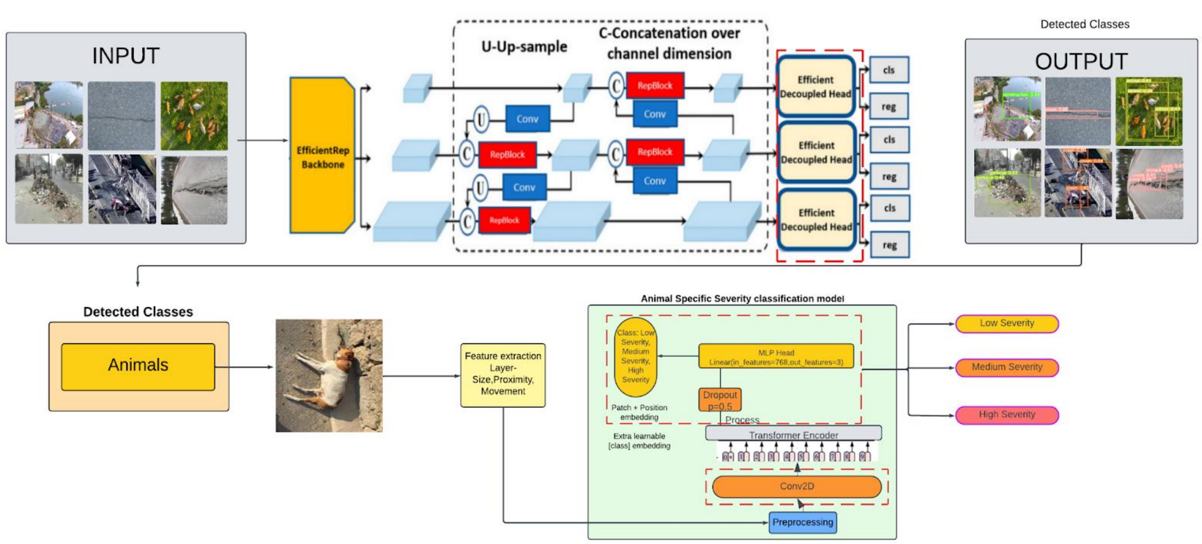
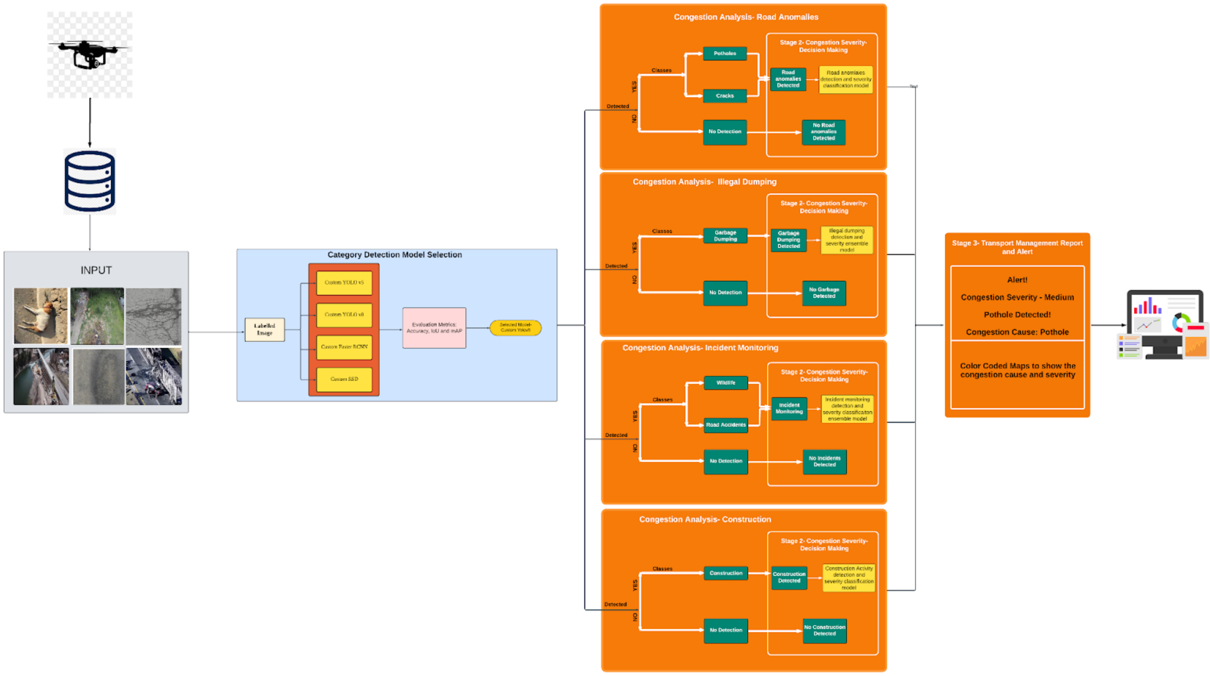


Figure 33

Detailed Architecture of the proposed Model

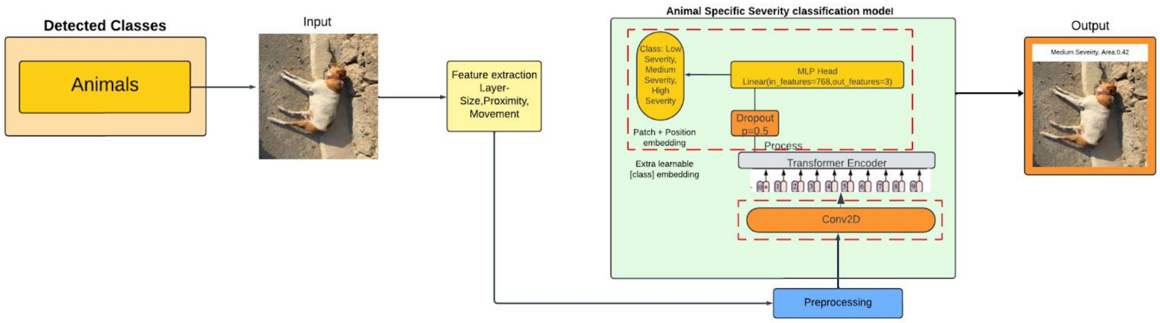


Animal Detection and Severity Classification Model

Our Animal Detection and Severity classification model is designed specifically for detecting and classifying the severity of animal related hazards including wildlife on or around the roads. This ensemble model uses a Yolov8 which is trained on UAV images of wildlife on roads, near highways and in forests, detects the category in first stage. In the second stage it utilizes Vision Transformer (ViT) architecture, which is specifically trained on Animal data only and customized for fine-tuning on animal-specific dataset. The stage 2 animal specific severity classification model captures the severity based on proximity, size and movement of the animal, into low, medium and high classes. For instance, Larger or closer animals, especially those that could cause accidents, are classified as higher risk. This model captures fine-grained features such as proximity to the road, movement, or size of the animal. Figure 34 shows the Architecture of Animal Specific Severity classification model.

Figure 34

Architecture of Animal Specific Severity Classification Model

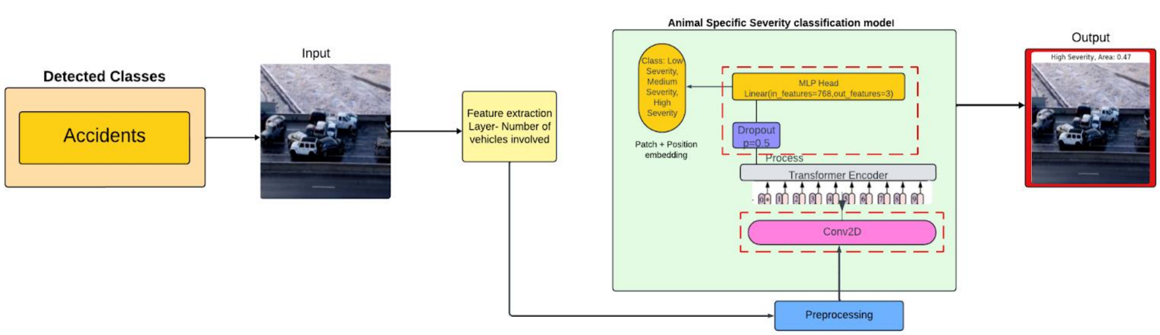


Vehicle Crash Detection and Severity Classification

The ensemble model named Vehicle Crash Detection and Severity classification model is tailored for vehicle crash detection and estimation of severity. Accidents can highly impact the traffic flow and cause delays. The debris which lies on the road after a crash will also cause a significant delay. To address this obstacle, we have trained a hybrid model which precisely detects the Crash category and focuses on detecting the severity of the damage levels. The severity is based on the overall accident impact including number of vehicles involved, extent of damage and whether the crash is blocking the road. More severe crashes with higher damage or multiple vehicles are classified with greater urgency. Figure 35 shows the Architecture of vehicle crash specific Severity classification model.

Figure 35

Architecture of Vehicle Crash Detection And Severity Classification Model

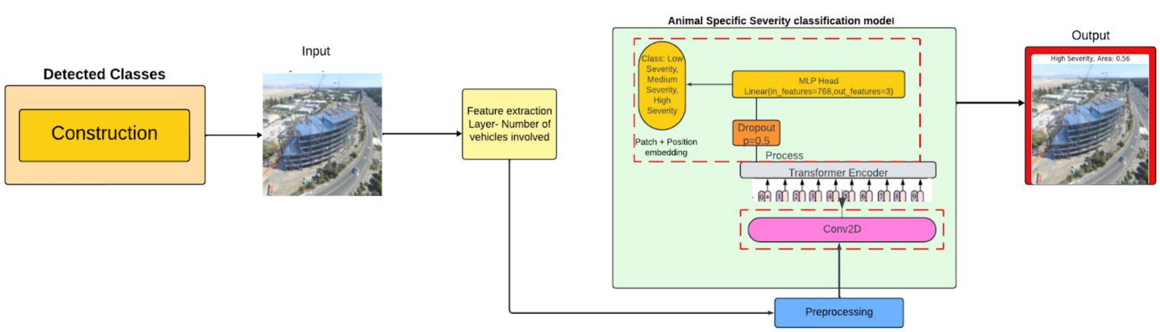


Construction Activity Detection and Severity Classification

The Construction activity detection and severity classification model is specifically designed to identify ongoing construction activities on roadways, which may present significant hazards to travelers. This model aims to enhance safety by accurately detecting construction zones and assessing the level of risk they pose to ensure informed decision-making for all road users. When the stage 1 model, yolov8 detects ongoing construction on the road, this model is triggered to assess the severity level considering the factors such as road blockage, presence of heavy machinery and proximity to active traffic lanes. The model focusses on the construction sites regions, the conv2D layer captures the larger structural changes, such as equipment, signs or barricades. The fine-tuned Transformer encoder understands the spatial features of constructional activities and the MLP head classifies the severity based on the impact of the construction into low, medium and high severity classes. Figure 36 shows the architecture of the Construction activity detection and severity classification model.

Figure 36

Architecture of Construction Activity Detection and Severity Classification Model

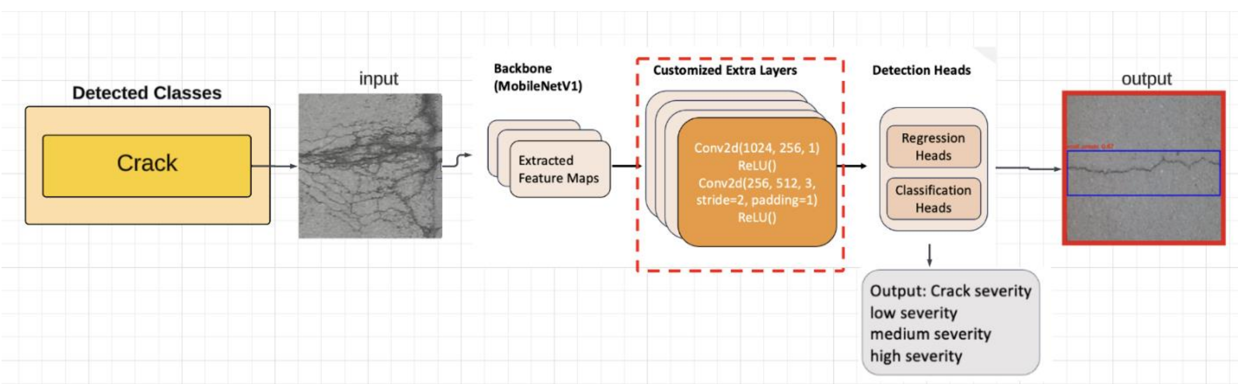


Crack detection and Severity Classification

This model is designed to identify and assess road cracks, enhancing safety for vehicles and pedestrians. Utilizing SSD based on the category information detected by YOLOv8 model, the model extracts feature maps and includes custom extra layers that improve crack detection by capturing critical structural details. Extra layers consist of multiple sequential blocks, each beginning with a 1x1 convolution followed by a ReLU activation, followed by a 3x3 convolution with a stride of 2 and padding of 1. This architecture has two output heads: the regression head, which predicts bounding boxes around detected cracks, and the classification head, which categorizes severity into low, medium, and high levels. The custom layers enhance feature extraction, enabling the model to differentiate between minor and significant cracks effectively. Figure 37 illustrates the architecture of the Road Crack Detection and Severity Classification Model, showcasing its ability to combine feature extraction, customized extra layers, and bounding box regression, and severity classification for improved road crack detection.

Figure 37

Architecture of Crack Detection and Severity Classification Model

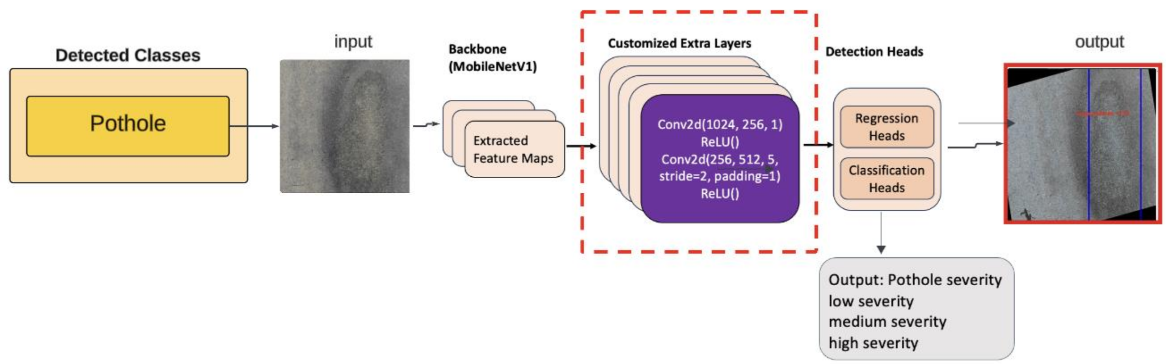


Pothole Detection and Severity Classification

The Pothole Detection Model is designed to accurately identify and assess potholes on roadways, enhancing safety for vehicles and pedestrians. This model shares a similar architecture to the Road Crack Detection Model. Figure 38 shows the detailed architecture of the pothole model. However, it incorporates an additional sequential block in its custom extra layers, where the convolution layers utilize a 5×5 kernel instead of 3x3. This adjustment allows the model to capture larger features associated with potholes while maintaining important spatial information. By increasing the kernel size and adding an extra block, the model enhances its feature extraction capabilities, improving the accuracy of pothole detection and classification based on severity, thereby contributing to safer driving conditions.

Figure 38

Architecture of Pothole Model and Severity Classification Model

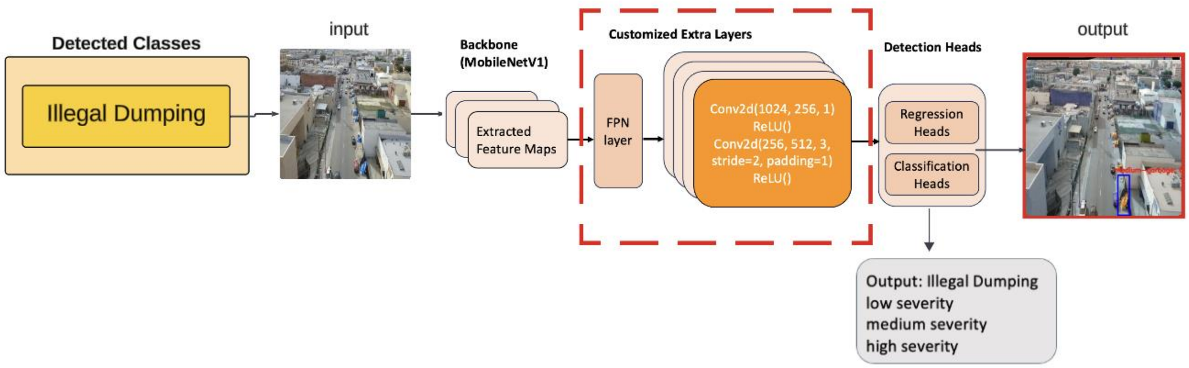


Illegal Dumping Model

The Illegal Dumping Model is utilized to spot and evaluate cases of illegal waste disposal in urban areas, aiming to keep our environment safe and raise community awareness. This model is modified based on the SSD model and adds a special layer called a Feature Pyramid Network to make it even better at finding trash in different sizes. Figure 39 shows how FPN is integrated with the SSD based model. The FPN works by taking features captured at various levels of detail from the backbone network, which processes the images. It first up-samples the features from the highest level, which contain broader, more abstract information about the image, and then combines these features with those from lower levels that capture finer details. This combination involves using a simple method, like adding or averaging the features together, after adjusting their sizes to match, ensuring that it can identify and locate various types of waste. By leveraging such a FPN layer, the model becomes more accurate at detecting illegal dumping, helping us better address waste issues and maintain cleaner neighborhoods.

Figure 39

Architecture of Illegal Dumping Model and Severity Classification Models



Model Result Aggregation

The six models are developed to identify various objects on the road, including potholes, cracks, dead animals, damaged vehicles involved in collisions, construction materials, and illegally dumped items. Each object is classified by its severity level. An aggregation function is then applied to count the total number of detected objects and organize them by severity levels. This aggregated data is transmitted to the front-end system, where it is visually represented on a map. The map uses different colors to indicate the severity levels for specific areas or blocks. This provides actionable insights to relevant departments and personnel, enabling them to prioritize and address road maintenance and safety issues effectively.

4.2 Model Supports

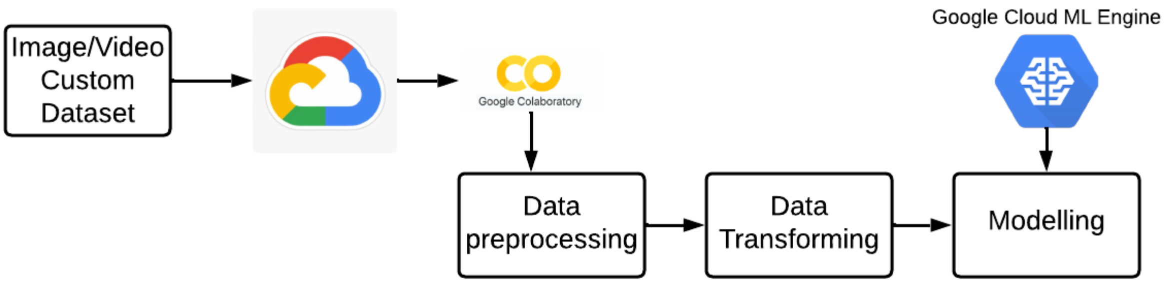
To advance our model development, we utilize cloud-based services like Google Cloud for storing data, conducting analysis, and deploying models. The scalable and dependable infrastructure provided by Google Cloud can effectively handle extensive amounts of data and intricate machine learning processes. Utilizing the storage services offered by Google Cloud enables us to securely store our data and access it easily for analysis and model training.

Google Colab, a cloud-based platform compatible with Jupyter notebooks, is employed for collaborative data preparation and model development. This tool facilitates real-time sharing and collaboration on notebooks among team members, supporting seamless cooperation and version control. These capabilities enhance the efficiency of data preprocessing, transformation, and personalized model development to meet specific project needs. Moreover, Google Cloud offers an array of machine learning tools and services for activities such as model training, evaluation, and deployment. For instance, the Google Cloud AI Platform enables scaling up machine learning model training by utilizing distributed computing resources to achieve faster training times and enhanced model performance. Additionally used is Google Cloud AutoML which automates much of the process involved in building and deploying machine learning models without extensive manual intervention needed.

Moreover, Google Cloud offers comprehensive solutions for deploying and monitoring models, ensuring that our models are deployed in production environments and can be easily monitored for performance and reliability. With Google Cloud's complete machine learning platform, we can streamline the entire model development process from data preparation to deployment, driving innovation in transportation management. It may also be valuable to include a diagram showcasing the architecture of our machine learning pipeline on Google Cloud. This diagram could illustrate various components such as data storage, preprocessing, model training, evaluation, deployment along with integration points with tools like Google Colab. Furthermore, a visual representation illustrating in Figure 40, the flow of data through our pipeline from collection to decision-making would enhance understanding of how our model functions.

Figure 40

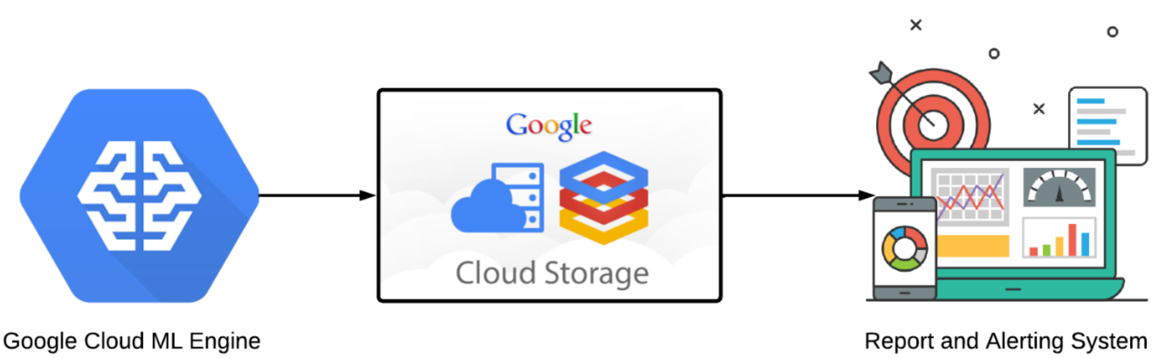
Dataflow/Pipeline



The deployment architecture diagram demonstrates the way our cloud-based machine learning models are implemented and supervised using various services on Google Cloud Platform. It highlights the utilization of Google Cloud ML Engine for model serving, and Google Cloud storage for data storage. Furthermore, there is an established system for reporting and alerting to monitor model performance. This architectural framework, shown Figure 41 guarantees scalability, dependability, and immediate insights crucial for transportation management.

Figure 41

Deployment Architecture



4.3 Model Comparison and Justification

Our model selection process involved extensive literature research to identify models that align with the criteria such as interpretability, scalability, accuracy, false negative minimization, and to ensure both effectiveness and efficiency.

Our proposed models include Faster R-CNN, MB1-SSD, Vision Transformer, YOLOV5 and YOLOV8. Each was selected for its distinct strengths and contributions to our objectives.

MB1-SSD

This model is a combination of MobileNetV1 and the SSD framework, is perfect for road inspection duties due to its efficiency, speed, and precision. With a relatively low number of parameters, it strikes a balance between model complexity and performance. Its lightweight design enables immediate object detection under different scenarios, which is crucial for our road inspection problem.

In terms of the workflow of MB1-SSD, firstly, the input to MB1-SSD consists of images captured by cameras mounted on UAV devices. Then a data transformation layer was added to resize the image to a 300x300 resolution. By processing images at a 300x300 resolution, MobileNetV1 achieves an optimal point between fast processing and reliable accuracy. Then the model pulls out important image features. The extracted features are then passed through additional convolutional layers in the SSD framework to predict object bounding boxes and class probabilities. To maintain precision and prevent outliers, MB1-SSD employs various strategies, including training data augmenting, applying regularization techniques like dropout to prevent overfitting, and implementing non-maximum suppression during inference phase to eliminate redundant detections and enhance the accuracy of the final output.

To justify the selection and performance of the MB1-SSD model, we will carry out the subsequent experiments:

- **Comparative Analysis:** Individual vs. Combined Category Performance: Initially training and testing models on individual categories: cracks, potholes, car accident, animals, garbage, and constructions, allows for assessing performance per class, revealing accuracy and classification strengths and weaknesses for each category. Then we will train and test on the full dataset that provides a comprehensive evaluation of the model's overall capability to handle diverse objects.
- **Sensitivity Analysis for Hyperparameter Tuning:** The MB1-SSD model will be trained using various combinations of hyperparameters such as learning rate, batch size, and network depth. Subsequently, the segmentation performance on a validation dataset will be assessed. A comparison of the results will help identify the most optimal hyperparameter settings to enhance segmentation accuracy and efficiency.

Considering the limitations, the lightweight design of MB1-SSD may result in lower accuracy compared to more intricate models, especially in detecting small or distant objects due to its fixed 300x300 resolution. Also, just like most the deep learning models, it requires sufficient and diverse training data to achieve effective generalization across diverse road conditions and environments.

Faster RCNN

Faster R-CNN, a pioneering framework in object detection, transformed the field with its efficient Region Proposal Network (RPN) and two-stage architecture. The RPN produces region proposals, which are subsequently improved and categorized by a Region-based Convolutional Neural Network (R-CNN).

This framework utilizes a backbone convolutional neural network (CNN) for feature extraction, typically pre-trained on extensive datasets to capture detailed characteristics from the input image enabling precise detection of various road anomalies such as potholes, cracks, debris, and dead animals. The RPN complements this by generating candidate bounding boxes, while the detection network then refines these proposals and classifies objects into categories.

The detection network contains a SoftMax layer with many units, enabling classification into different categories of road defects and hazards. This robust architecture facilitates accurate and efficient object detection, making Faster R-CNN suitable for a wide range of computer vision tasks.

The performance of Faster R-CNN in detecting cracks using UAV imagery was demonstrated in [13]. Their research highlights its superior accuracy compared to other deep learning algorithms, particularly in identifying cracks of different shapes and sizes. However, their focus was primarily on evaluating the effectiveness of Faster R-CNN alone and did not explore the integration of Faster R-CNN with single-stage detectors like YOLO.

Our model's strength lies in the customization of Faster R-CNN through fine-tuning with annotated datasets tailored for road inspection tasks, rather than relying solely on pre-trained models. Furthermore, we incorporate Faster R-CNN with YOLO, to benefit from the strengths of both approaches. This integration addresses a gap in existing research by exploring the synergies of combining multiple detection frameworks, improving the overall efficiency and effectiveness.

YOLOv5

YOLOv5 serves as a pivotal tool for real-time detection and classification of road anomalies using UAV imagery. It effectively extracts and analyzes image features to identify various types of road defects, debris, and obstructions encountered in smart city environments. It employs its single-stage object detection architecture to extract features and detect anomalies.

YOLOv5's efficient architecture and minimal resource requirements make it highly adaptable to various UAV platforms and computational setups. It is interpretability features, such as bounding box visualization and confidence scores, enhance transparency and comprehension. Throughout training, the model iteratively alters its parameters to reduce the difference between predicted bounding boxes and ground truth annotations, optimizing its ability to accurately localize and classify anomalies. To justify our model, we propose a comprehensive evaluation approach by:

- **Performance Comparison:** We will compare the detection performance of YOLOv5 across various road anomaly classes and evaluate accuracy, precision, and recall metrics to validate its effectiveness in identifying and classifying various types of road defects and obstructions.
- **Severity Prediction:** Our method goes beyond other typical detection models, which only concentrate on spotting anomalies, by also forecasting the seriousness of these issues. YOLOv5 is great at identifying anomalies such as cracks, our model goes a step further by integrating the outputs of the detection model into severity classification models which are SSD and Vision transformer, to estimate the severity of each anomaly. This predictive feature provides a more detailed insight into road conditions.
- **Through adaptive learning and fine-tuning:** we aim to enhance the model's performance by iteratively refining the model on annotated datasets of road anomalies, we optimize its ability to distinguish between various defect types and prioritize maintenance activities accordingly.

YOLOv8

YOLOv8, a development of the You Only Look Once series of object detection models, emerges as a strong option for detecting road anomalies in real-time UAV imagery. In contrast to conventional object detection techniques that utilize sliding window methods, YOLOv8 utilizes a single neural network architecture to predict bounding boxes and class probabilities for numerous objects in an image simultaneously.

The primary objective of YOLOv8 in our project is to precisely identify and categorize different road irregularities such as potholes, fractures, debris, deceased animals, construction work, and accidents in high-resolution UAV images. YOLOv8 utilizes its multi-scale feature extraction abilities to effectively capture detailed information and contextual information crucial for precise anomaly localization. This corresponds with the findings in Khare et al. (2023) where researchers emphasized the effectiveness of YOLOv8 in extracting multi-scale features for diverse object detection tasks.

YOLOv8 employs a deep convolutional neural network (CNN) architecture to analyze input images and generate predictions for object bounding boxes and class probabilities. The model uses a sequence of convolutional and pooling layers to extract features, along with fully connected layers for classification. YOLOv8's distinctive design enables it to achieve real-time inference speeds.

Despite its strengths, YOLOV8 also has some limitations. It may have limited sensitivity to small or subtle anomalies, such as minor cracks or debris, which could affect the model's ability to fully assess road conditions, especially in densely populated areas with complex infrastructure. Another limitation is the quality and diversity of the training data. In scenarios where UAV imagery is captured under challenging conditions, such as low light or adverse weather, the model's detection accuracy might decrease, resulting to potential false positives or missed anomalies.

One of the key strengths to justify this model is the ability of it to achieve real-time performance without compromising accuracy and to validate its suitability for our road anomaly detection system, we propose these experiments.

- **Comparative Analysis of Detection Performance and severity prediction:** We will evaluate how well YOLOv8 performs in detecting road anomalies compared to other object detection models like Faster R-CNN. This analysis aims to demonstrate its effectiveness in accurately identifying and localizing anomalies in UAV imagery. And we will predict the severity of identified issues by feeding the output into a severity classification model. Additionally, we will examine the real-time processing speed of YOLOv8 on various hardware platforms, such as GPUs and CPUs.
- **Hyperparameter Sensitivity Analysis:** We plan to optimize the model's performance for our road anomaly detection task by performing a sensitivity analysis on key hyperparameters, such as anchor box sizes, input image resolution, and confidence threshold. By systematically tuning these parameters and evaluating their impact on detection accuracy and speed, we aim to enhance the overall effectiveness of our model.
- **Quality Control for Training Data:** We will perform rigorous quality control measures for training data collection and annotation which can mitigate the impact of data quality and diversity on YOLOv8's performance. Ensuring that training data accurately depicts various road anomaly scenarios and environmental conditions can enhance the model's generalization ability and detection accuracy.

Table 8 aims to compare various models used in our road inspection problem. Each row in the table represents a specific characteristic of the selected models, the strengths and weakness of each model are also summarized in this comparison table.

Table 8

Selected Road Inspection Model Comparison Table

Characteristic	**YOLOv5&v8**	**Faster R-CNN**	**SSD**	**Ensemble Model**
Architecture	Custom CNN	CNN + RPN	MobileNet + SSD	Combine the output of four detection model
Preprocessing	Resize, normalization, augmentation	Resize, normalization, augmentation	Resize to 300X300	Same as individual models
Training Speed	Moderate	Slow	Fast	Depends on individual models
Accuracy	High	High	Moderate to High	High
Computational Complexity	Moderate	High	Low to Moderate	High
Strengths	High accuracy, real-time detection, single-	High accuracy, precise object localization, good	Lightweight, single-shot detection,	Combines strengths of individual

	shot detection	at identifying cracks of different shapes and sizes	efficient in resource limited environments	models, higher accuracy
Weaknesses	Sensitivity to small or subtle anomalies, such as minor cracks or debris, potentially slower inference	Slower training and inference speed, higher computational complexity	Sacrifice some accuracy for speed, lower accuracy compared to more complex models	Complexity of managing multiple models

Stage 2: Severity Classification

MB1-SSD

The Single Shot MultiBox Detector (SSD) is leveraged in our project for its real-time classification capabilities, specifically in categorizing the severity of road anomalies such as cracks, potholes, and debris. Known for its speed and efficiency, SSD predicts object bounding boxes and severity levels in a single forward pass, making it ideal for UAV-based road inspections.

SSD processes the detected road anomalies from Stage 1 and classifies them into low, medium, or high severity. Its ability to handle varying object scales and resolutions ensures precise classification across different severity levels, offering a fast and reliable solution for real-time road condition analysis.

Vision Transformer

The Vision Transformer (ViT) model, known for its ability to capture global dependencies between image regions, was utilized to classify the severity of road anomalies such as cracks, potholes, and debris. ViT's self-attention mechanism allows it to process large-scale UAV images efficiently, enabling more detailed classification of these anomalies. It provides a unique advantage by focusing on essential image features at various scales, which is critical for accurately determining severity levels in road conditions.

ViT processes UAV images through a sequence of patches, embedding each one to analyze spatial information across the entire image. This detailed attention to both local and global features ensure a thorough analysis of road anomalies. Despite achieving an average accuracy of 60% in classifying severity across categories, ViT provides substantial insights into the nature of road defects by leveraging its ability to learn deep feature representations.

To justify the model, we propose a detailed analysis by:

- ViT stands out in its ability to perform nuanced severity classification due to its advanced attention mechanism.
- ViT provides more detailed insights into the severity of detected road anomalies, making it an ideal complement for post-detection analysis.
- ViT contributes significantly to severity classification, offering a multi-stage approach for thorough road condition analysis.

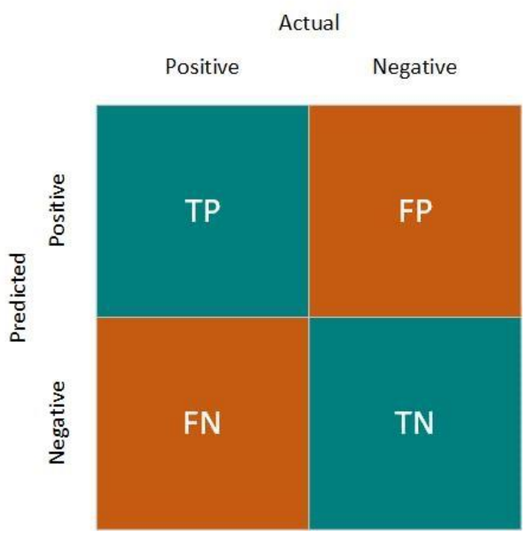
4.4 Model Evaluation Methods

Model-oriented Evaluation Metrics- Confusion Matrix

A confusion matrix presents the model's classification performance for various classes in a tabular format. It displays the counts of true positive, false positive, true negative, and false negative predictions for each class. The matrix allows for the analysis of model errors and the identification of areas for improvement. Figure 42 describes its components.

Figure 42

Confusion Matrix



In a binary classifier, the class representing the positive outcome is typically labeled as 1, while the class representing the negative outcome is labeled as 0.

- True Positives (TP) is when the classifier correctly identifies the positive class (the classifier predicts 1).
- False Positives (FP) happens when the classifier incorrectly identifies the positive class (the classifier predicts 1 when it should predict 0).
- True Negatives (TN) is when the classifier correctly identifies the negative class (the classifier predicts 0).
- False Negatives (FN) happen when the classifier incorrectly identifies the negative class (the classifier predicts 0 when it should predict 1).

Accuracy

It measures the proportion of correctly classified instances among all instances. Indicate how often the model correctly identifies road anomalies. Calculated as the ratio of the number of correct predictions to the total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Here, *TP = True Positives, TN = True Negatives, FP = False Positives and FN = False Negatives.*

Precision

It measures the proportion of correctly identified instances among all instances classified as positive by the model. It is calculated by dividing true positives (correctly identified instances of road anomalies) by the sum of true positives and false positives (instances incorrectly classified as road anomalies). Precision Indicates how effectively the model ensures that identified positive instances correspond to actual occurrences of specific road anomalies.

Precision = $\frac{True\ Positive}{True\ Positive + False\ Positive}$

Recall

It is also known as sensitivity, measures the ability of the model to correctly identify all instances of a particular class. It is computed as the ratio of true positives to the sum of true positives and false negatives. A high recall score would indicate that the model effectively captures the presence of all anomalies, minimizing the risk of missing critical hazards that could impact transportation infrastructure.

Recall = $\frac{True\ Positive}{True\ Positive + True\ Negative}$

F1 Score

F1 Score is a thorough measure that combines both precision and recall, providing a balanced assessment of the model's performance in identifying road anomalies. The F1 score represents how well the model balances the needs of accurately detecting anomalies on the road (precision) and making sure that all anomalies are identified (recall). A high F1 score indicates that the model effectively identifies road irregularities while minimizing false positives and false negatives.

It is calculated as the harmonic mean of precision and recall, giving equal weight to both metrics

F1 Score = $2 * \frac{Precision * Recall}{Precision + Recall}$

Mean Average Precision(mAP)

Mean Average Precision is a commonly used measure to evaluate the performance of object detection models, particularly in scenarios where multiple classes are involved. It provides a comprehensive assessment of the model's ability to accurately detect and classify multiple classes of road anomalies. mAP is calculated by first computing the Average Precision (AP) for each class separately and then taking the mean of these AP scores across all classes. AP measures the precision-recall trade-off for a specific class, considering the model's ability to correctly identify instances of that class at different confidence thresholds. A high mAP score indicates that the model performs well across all classes of road anomalies, achieving high precision and recall consistently. This means that the model accurately detects and classifies various road irregularities while minimizing false positives and false negatives.

mAP = $\frac{\sum_{i=1}^{n} AP_i}{n}$

here, n is the total number of classes and AP_i is the average precision for class i.

Intersection over Union (IoU)

Intersection over Union is a fundamental measure used primarily in tasks involving object detection and segmentation. It quantifies the spatial overlap between the predicted regions and the ground truth annotations, providing a measure of the accuracy of object localization. IoU is calculated as the ratio of the intersection area between the predicted region and the ground truth region to the union area of both regions. It is expressed as:

$$\text{IoU} = \frac{Area\ of\ Intersection}{Area\ of\ Union}$$

IoU values range from 0 to 1, where a value of 1 indicates perfect overlap between the predicted and ground truth regions, while a value of 0 indicates no overlap. When the model's predictions closely match the ground truth annotations, it is said to have higher localization accuracy (higher IoU).

Loss

Loss measures the difference between the model's predictions and the actual labels during training. Keeping track of loss values offers valuable information about training advancement and model optimization. Lower loss values signify stronger convergence and enhanced performance.

4.4.2 Targeted problem-oriented evaluation metrics

Our goal is to create a smart city road inspection system. It's crucial to ensure that our model is not only accurate but also efficient and user-friendly, capable of quickly identifying road anomalies while being accessible and intuitive for road maintenance crews and transportation authorities.

Speed and Efficiency

To ensure our road inspection system works efficiently for smart cities, we measure its processing speed using metrics like frames per second (fps) or inference time per image. This helps us ensure quick analysis of road images, enabling timely detection of hazards and enhancing road safety.

User Feedback and Usability

To ensure our road inspection system is user-friendly. We track the utility and usability of the system or a specific model across different scenarios and timeframes. This involves documenting how well the system performs and how easy it is to use in practice and the probability for a model to be chosen. By logging this information, we can monitor changes, identify trends, and make informed decisions about improvements or adjustments to the system.

4.5 Model Results

4.5.1 Model Evaluation

In this section, we will present the outcomes of our chosen model, assessing their performance across diverse metrics to ensure their reliability for further deployment in the real-world.

YOLOv5

In Figure 43, the confusion matrix for YOLOv5 reveals notable accuracies for the detection of animal, construction, and pothole instances, each achieving approximately 92%. While the detection accuracy for garbage instances is the lowest, standing at a mere 30%. Figures 44 and 45 present the F1-confidence and Precision-Recall Curve, respectively.

Figure 43

Confusion Matrix for YOLOv5

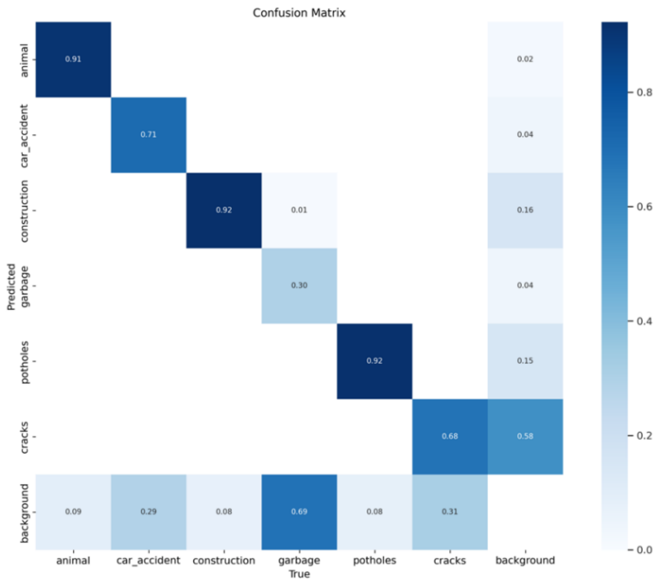


Figure 44

F1-confidence Curve for YOLOv5

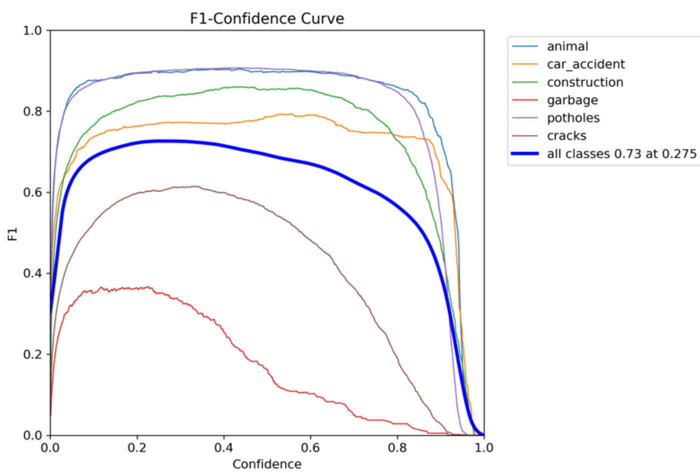
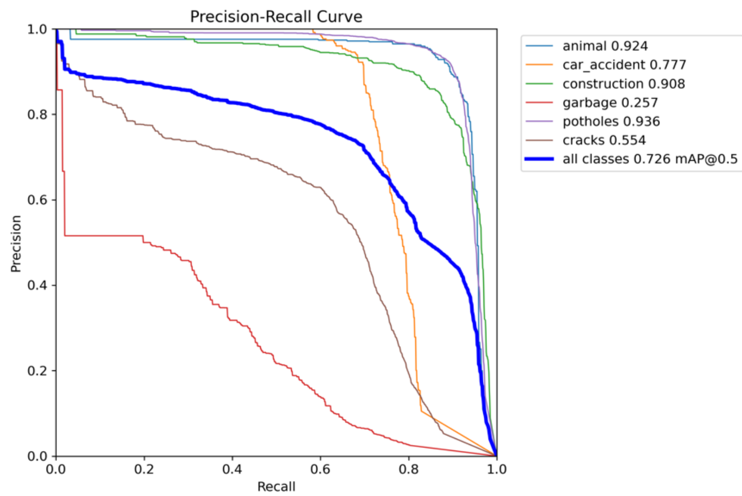


Figure 45

Precision-Recall Curve for YOLOv5



YOLOv8

Figure 46 shows the confusion matrix from YOLOv8. It follows a similar pattern to YOLOv5, with animal, construction, and pothole detection being the most accurate categories. However, these top categories are a bit less accurate compared to YOLOv5. On the bright side, the accuracy for identifying garbage has improved by 12% compared to YOLOv5. Figure 47 displays the F1-confidence Curve and the plot in Figure 48 illustrates the trends of various loss and evaluation metrics, including recall, precision, and mAP50.

Figure 46

Confusion Matrix for YOLOv8

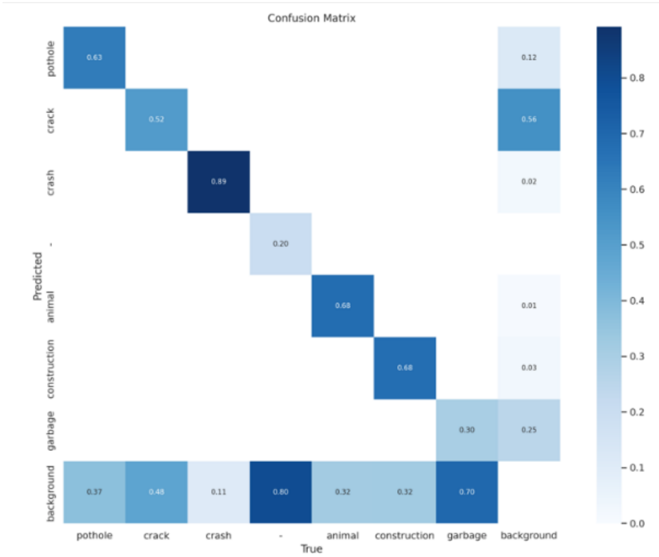


Figure 47

F1-confidence for YOLOv8

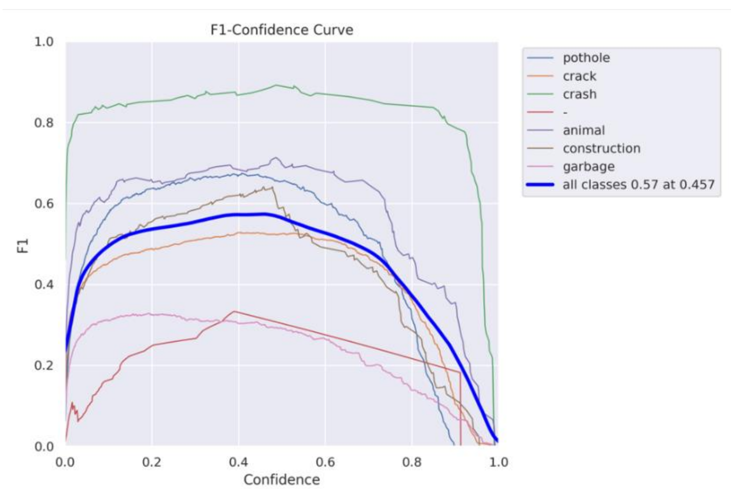


Figure 48

Line Plot of Various Loss and Evaluation metric for YOLOv8

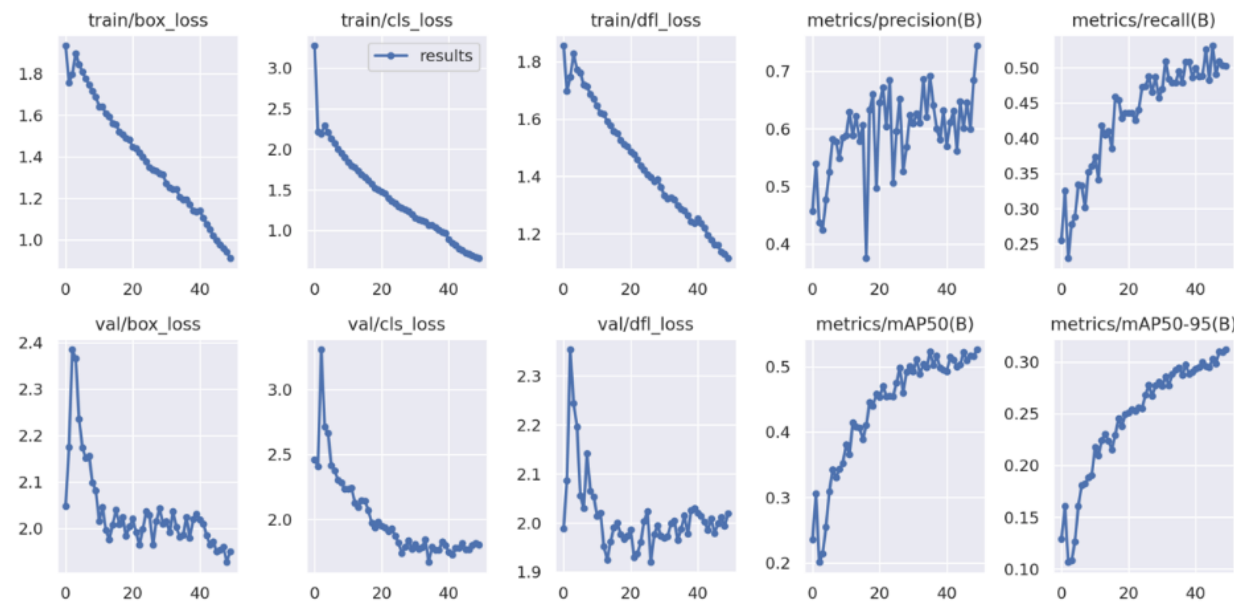


Figure 49 illustrates the Mean Average Precision at 50 confidence and at 50-95 confidence for individual categories as well as the aggregated values. The overall mAP50 for all classes are reported as 0.559, respectively.

Figure 49

Mean Average Precision Analysis for YOLOv8

Class	Images	Instances	Box(P	R	mAP50	m
all	737	1715	0.744	0.503	0.526	0.311
pothole	737	249	0.826	0.562	0.637	0.305
crack	737	738	0.609	0.468	0.446	0.265
crash	737	74	0.911	0.838	0.922	0.803
-	737	10	1	0.192	0.217	0.108
animal	737	88	0.74	0.636	0.676	0.388
construction	737	87	0.655	0.598	0.551	0.208
garbage	737	469	0.469	0.226	0.233	0.0994

Faster R-CNN

In terms of evaluation on Faster R-CNN model, after 50 epochs, for animal detection, the Average Intersection over Union (IoU) stands at 0.384, with an Average Precision of 0.397 and an Average Recall of 0.350, resulting in an Average F1-Score of 0.365. Moving to cracks detection, the Average IoU improves to 0.463, accompanied by an Average Precision of 0.430

and an Average Recall of 0.389, yielding an Average F1-Score of 0.400. However, for potholes detection, the performance drops significantly, with an Average IoU of 0.152, Average Precision of 0.026, Average Recall of 0.005, and an Average F1-Score of 0.009.

MB1-SSD

Figure 50 illustrates that while MB1-SSD has a lower overall mAP value of 0.59 compared to YOLO, it outperforms in garbage detection accuracy, reaching an impressive 87.7%, twice as much as the YOLO model. This underscores MB1-SSD's proficiency in accurately identifying and categorizing garbage instances.

Figure 50

Mean Average Precision Analysis for MB1-SSD

```
2024-05-12 17:07:12 - Average Precision Per-class:
2024-05-12 17:07:12 -     animal: 0.7809195545902645
2024-05-12 17:07:12 -     car_accident: 0.6926531896676031
2024-05-12 17:07:12 -     construction: 0.3977226474566815
2024-05-12 17:07:12 -     garbage: 0.8770475956262063
2024-05-12 17:07:12 -     potholes: 0.41419417266870967
2024-05-12 17:07:12 -     cracks: 0.40722698277295644
2024-05-12 17:07:12 - Mean Average Precision (mAP):  0.5949606904637369
```

Category Specific Model Evaluations

Animal Category Specific Severity Classification Model

Figure 51 shows the mAP score for this particular model, In here the base model is a transformer which was trained on Animal custom UAV dataset. On Testing data the model performed well with an accuracy of 77% and precision of 78% where it correctly identifies the severity involved in the image into 3 distinct classes - Low severity, medium severity and high severity. This model is great at identifying all kind of severity levels with an average mean precision of 90%.

Figure 51

mAP of Animal category specific severity classification model

```
Average Precision for Class 0: 0.8660
Average Precision for Class 1: 0.8725
Average Precision for Class 2: 0.9822
Mean Average Precision (mAP): 0.9069
```

Accident Category Specific Severity Classification Model

Figure 52 shows the mAP for the Accident category specific severity classification Model. the model performs well with a precision score of 81% and a test accuracy of 90% where it correctly classifies High severity accidents, medium severity accidents and low severity accidents.

Figure 52

mAP score for accident category specific severity classification model

```
Average Precision for Class 0: 0.8596
Average Precision for Class 1: 0.5354
Average Precision for Class 2: 0.9557
Mean Average Precision (mAP): 0.7836
```

Cracks Category Specific Severity Classification Model

Figure 53 shows the crack model achieves an average precision of 0.88 overall, with 98% for small cracks, 77% for medium cracks, and 89% for large cracks, showing it's a bit better at identifying smaller cracks.

Figure 53

mAP score for crack category specific severity classification model

```
2024-11-22 19:39:26 -     small_crack: 0.9772727272727275
2024-11-22 19:39:26 -     medium_crack: 0.7711154094132818
2024-11-22 19:39:26 -     large_crack: 0.8863636363636365
2024-11-22 19:39:26 - Mean Average Precision (mAP):  0.8782505910165487
```

Construction Category Specific Severity Classification Model

Figure 54 shows the mAP score for the Construction category specific severity classification Model where the model is trained only on construction specific UAV data and the accuracy on test data is 81%.

Figure 54

mAP score for construction category specific severity classification model

```
Average Precision for Class 0: 0.8596
Average Precision for Class 1: 0.5354
Average Precision for Class 2: 0.9557
Mean Average Precision (mAP): 0.7836
```

Pothole Category Specific Severity Classification Model

Figure 55 shows the model performs well, with an overall precision of 85%. It correctly identifies small potholes with a mAP of 74%. For medium potholes, the model performs better, with mAP around 87%. It's especially good at detecting large potholes, with a mAP of 95%. The result shows the model is reliable at detecting potholes in general, especially for the large ones.

Figure 55

mAP score for pothole category specific severity classification model

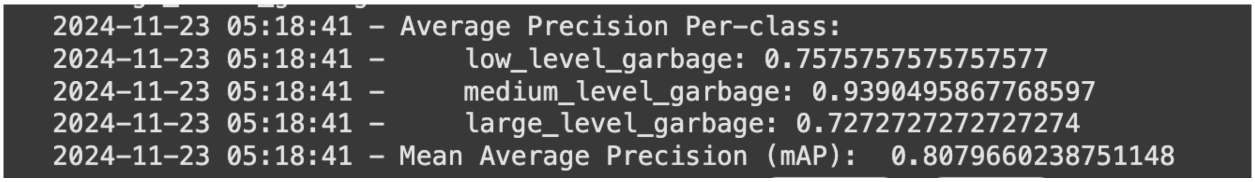
```
2024-11-22 21:30:31 - Average Precision Per-class:
2024-11-22 21:30:31 -     small_pothole: 0.7386081426550998
2024-11-22 21:30:31 -     medium_pothole: 0.8650484059948536
2024-11-22 21:30:31 -     large_pothole: 0.9490909090909093
2024-11-22 21:30:31 - Mean Average Precision (mAP):  0.8509158192469543
```

Garbage Category Specific Severity Classification Model

Figure 56 shows that the garbage detection model achieves an average precision of 81% overall. However, detecting low-level and high-level garbage remains relatively challenging, with precision scores of approximately 75% and 72%, respectively.

Figure 56

mAP score for garbage category specific severity classification model



```
2024-11-23 05:18:41 - Average Precision Per-class:
2024-11-23 05:18:41 -     low_level_garbage: 0.7575757575757577
2024-11-23 05:18:41 -     medium_level_garbage: 0.9390495867768597
2024-11-23 05:18:41 -     large_level_garbage: 0.7272727272727274
2024-11-23 05:18:41 - Mean Average Precision (mAP):  0.8079660238751148
```

4.5.2 Model Comparison

By comparing all 4 models, we observed that YOLOv5 demonstrates strong performance in detecting animals, construction, and potholes, with an accuracy of around 92% in these categories. However, it struggles significantly with garbage detection, where the accuracy drops to just 30%. YOLOv8 follows a similar pattern but shows a 12% improvement in garbage detection compared to YOLOv5, although it slightly underperforms in the top categories overall. Faster R-CNN underperforms in detecting cracks and potholes, making it impractical for real-time detection across all categories. As a result, we chose to exclude this model from detecting the remaining categories. Meanwhile, MB1-SSD stands out for its impressive performance in garbage detection, achieving an accuracy of 87.7%. However, its overall accuracy is lower than that of the YOLO family models, making it less versatile for comprehensive detection tasks.

After careful consideration, we prioritized mean Average Precision (mAP) as our main evaluation criterion, as it is widely recognized as the industry standard for object detection and is commonly used in literature. This metric offered a balanced and reliable basis for determining the most suitable model for the subsequent severity determination tasks. Table 9 presents a comparison table showcasing the performance of these four models across the different detection categories.

Table 9

Model Comparison on mAP-50

Model	Cracks	Pothole	Animal	Construction	Garbage	Car Accident	All Class
Yolov5	0.554	0.936	0.924	0.908	0.257	0.777	0.726
Yolov8	0.645	0.921	0.906	0.896	0.362	0.749	0.747
Faster-RCNN	0.39	0.025	0.43	N/A	N/A	N/A	N/A
MB1-SSD	0.4	0.414	0.780	0.3977	0.877	0.692	0.594

Yolov8 outperformed the other models in detecting all the categories. So, we have chosen our improvised Yolov8 model for the stage 1 layer that is Object detection.

In the second stage of our architecture, we introduced category-specific severity classification models to enhance the efficiency and performance of the system. For each detected category in Stage 1, a dedicated model was developed and trained exclusively on the corresponding dataset. These models leveraged two base architectures, MB1-SSD and Vision Transformer, with custom improvisations tailored to the respective datasets.

The performance of these category-specific models was evaluated using metrics such as accuracy, mAP scores, and ROC curves. Each model classified severity levels into three categories: Low, Medium, and High. The use of category-specific models significantly improved the system's overall precision and reliability.

5. Data Analytics System

5.1 System Requirement Analysis

5.1.1 System boundaries and use cases:

Our primary focus is on enhancing transportation management within smart cities by utilizing a UAV-based road transportation system. This system identifies and classifies various road anomalies such as potholes, cracks, debris, dead animals, construction activities, and accidents. The project integrates machine learning models that detect these anomalies from UAV-captured images, estimate their severity, and feed this information into a dashboard for visualization and decision-making. The system's effectiveness lies in its ability to improve the efficiency of road maintenance and transportation planning, ultimately leading to better infrastructure management. The system boundaries are established by several factors:

- **Physical Boundaries:** The system is intended to operate within urban environments, primarily focusing on roads, highways, and other transportation infrastructure in smart cities. UAVs capture high-resolution images over predefined routes, which are then processed by machine learning algorithms to detect anomalies. The physical scope includes both central urban areas and surrounding roads where frequent maintenance is critical for managing transportation networks.
- **Data Boundaries:** The system relies on visual data collected from UAVs in the form of high-resolution images, supplemented with video data for specific cases like traffic accidents or large-scale road damage. The data collected is limited to visual information, and no additional sensors such as LiDAR or infrared are used. All data is processed and stored in cloud environments such as Google Cloud or Amazon S3, ensuring that the system is scalable and capable of handling large volumes of image

data. The images undergo preprocessing to improve model accuracy in detecting anomalies.

- **Functional Boundaries:** The system is designed to process UAV images, detect road anomalies, and provide severity estimations based on the identified objects. Its core functionalities include anomaly detection, classification, severity scoring, and reporting. These processes culminate in a real-time decision-support system where the detected anomalies are displayed on a dashboard. Alerts are sent to city officials or maintenance crews, enabling rapid response to issues affecting road safety. The system focuses primarily on analyzing road conditions and ensuring that transportation networks remain functional and efficient. The following are the use cases:
- **Road Anomaly Detection:** This use case focuses on maintaining road quality and safety by detecting common issues such as potholes, cracks, and debris. UAVs capture high-resolution images over urban roads and highways, which are then analyzed by machine learning models. Detected anomalies are classified, and their severity levels are estimated, helping city officials and road maintenance teams prioritize road repairs. Early detection enables proactive road management, ensuring smoother traffic flow, minimizing accident risks, and reducing long-term repair costs.
- **Incident Detection:** One of the critical components of transportation management is responding to unexpected incidents such as traffic accidents, road blockages, or sudden obstructions. The system analyzes UAV-captured video footage to detect incidents in real time and estimate their severity. This information is sent to

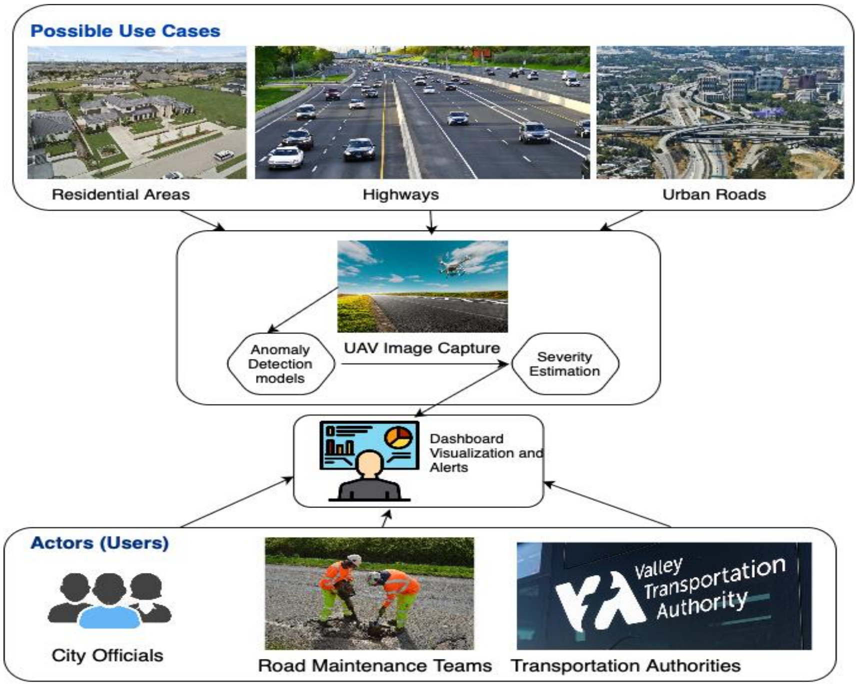
transportation authorities and emergency responders to facilitate quick decision-making. By identifying these incidents early, the system ensures that traffic can be rerouted, emergency vehicles can arrive on the scene faster, and congestion is minimized. The use of this technology helps manage traffic disruptions more effectively and supports smooth traffic flow even in the face of unexpected incidents.

- **Maintenance Planning:** Effective transportation management is not limited to reacting to problems but also involves planning for ongoing maintenance. The UAV-based system continuously monitors road conditions and identifies areas requiring repair. By analyzing the severity of detected anomalies, such as large potholes or severe road degradation, the system helps city planners allocate maintenance resources more efficiently. This proactive approach ensures that critical areas are repaired before they cause significant disruption, improving road reliability and reducing long-term repair costs. The insights provided by the system enable city officials to maintain a high standard of road safety and infrastructure quality.
- **Environmental Impact Monitoring:** This use case extends the system's functionality to environmental monitoring. UAVs capture images that are analyzed to detect illegal dumping and other environmental hazards along roads. These hazards are flagged for city authorities or environmental agencies to enable swift cleanup. By integrating environmental monitoring, the system ensures safer, cleaner roads and promotes sustainability in urban environments, supporting smart city development goals.

The interactions between the use cases and actors (users) are visually represented in Figure 57 below. This flowchart illustrates how UAVs capture images across various environments, which are then processed through anomaly detection and severity estimation models. Its highlights how actors such as city officials, road maintenance teams and transportation authorities engage with the system using the dashboard for real-time decision-making and prioritizing road repairs based on detected anomalies.

Figure 57

Overview of UAV-Based Road Inspection System Use Cases, Data Flow, and Actors



5.1.2 High-level data analytics and machine learning functions and capabilities

This system leverages advanced data analytics and machine learning (ML) capabilities to detect road anomalies, estimate their severity, and provide actionable insights to city officials through a dashboard. These high-level functions ensure that the system can support smart city transportation management efficiently and effectively, using state-of-the-art data processing techniques to deliver precise and timely results. Below are the key functions and capabilities of the system:

- **Data Collection and Preprocessing:** The system begins with the collection of high-resolution imagery from UAVs that survey urban road networks. These images are captured during UAV flights and cover a range of road conditions and environments, including highways, urban streets, and peripheral areas. Before feeding the images into machine learning models, preprocessing steps are applied, such as resizing, noise reduction, and normalization, to enhance the quality of the data. This step ensures the machine learning models receive clean, standardized data for accurate anomaly detection.

- **Anomaly Detection:** The core functionality of the system lies in detecting road anomalies such as potholes, cracks, debris, dead animals, and construction activities. Using advanced object detection models like SSD, YOLOv5, YOLOv8, Faster R-CNN and Vision Transformer, the system analyzes the UAV-captured images and identifies these anomalies with high accuracy. These machine learning models utilize deep convolutional neural networks (CNNs) trained on large datasets to recognize various types of road hazards in real-time. Once detected, the anomalies are classified into predefined categories, ensuring that all types of road damage are accounted for.
- **Severity Estimation:** Beyond simply detecting road anomalies, the system provides an additional layer of intelligence by estimating the severity of the detected anomalies. Severity classification models including SSD and Vision Transformer were used to analyze the size and impact of the anomalies. Based on the dimensions of the detected objects and their context, the system categorizes them into low, medium, or high severity levels. This classification helps city officials prioritize road repairs and allocate resources effectively, ensuring that the most critical issues are addressed promptly to avoid further deterioration.

- **Dashboard Visualization and Reporting:** The system integrates a comprehensive dashboard for visualizing the results of anomaly detection and severity estimation. Detected road anomalies are displayed on an interactive map, where users can see the locations of hazards along with their severity levels. The dashboard also provides detailed reports that summarize the condition of the roads, highlighting areas that need urgent attention. These reports can be generated on-demand or at scheduled intervals, providing city officials with the necessary insights to make informed decisions about road maintenance and infrastructure planning.
- **Machine Learning Model Integration and Scalability:** The system is designed with scalability in mind, ensuring that as more data is collected over time, the machine learning models can continue to adapt and improve. By utilizing google drive for data storage and model deployment, the system can easily handle large volumes of data and increase processing capacity as needed. This scalability ensures that the system remains effective as the scope of its application expands, making it suitable for both small-scale urban areas and larger cities with more complex road networks.
- **Continuous Learning and Model Optimization:** A vital component of the system is its ability to continuously improve its performance through ongoing learning. As new data is collected from UAVs, it is used to fine-tune and retrain the machine learning models. This iterative learning process helps the system become more accurate over time, improving its ability to detect road anomalies under different conditions. Regular evaluation and optimization of the models also ensure that the system stays up to date with the latest advancements in machine learning techniques, maintaining high performance in real-world applications.

- **Proactive Transportation Management:** The integration of data analytics and machine learning functions in this system facilitates proactive transportation management. By identifying road anomalies early and estimating their severity, the system enables transportation authorities to take preemptive measures to repair damaged roads, thereby avoiding traffic disruptions and minimizing long-term maintenance costs. Additionally, the system provides valuable data-driven insights that help in optimizing road maintenance schedules and allocating resources efficiently, supporting the broader goal of smarter and safer transportation infrastructure in urban areas.

5.2 System Design

5.2.1 System Architecture and Infrastructure

Our system has an architecture which utilizes a surveillance drone that is capable of patrolling along routes and capturing video footage via its onboard camera, which is transmitted back to the system over either a cellular network or Wi-Fi, depending on the deployment environment. Figure 58 shows a high-level diagram in which video data, along with control commands, is processed through a backend pipeline, which consists of several modules, including a roadway hazard detection module which is a custom model, a severity assessment module which detects if hazard is low, high or medium, and a frontend interface for display and alerts.

Once images are received, we process it through the roadway hazard detection module which can identify a wide range of hazards, such as potholes, cracks, illegal dumping, car accidents, and the presence of animals. If no hazards are detected, the system continues monitoring other images and analyzing them. When a hazard is detected, the system immediately sends the information to the Severity Assessment Module. Here, we have enabled our system to compute a severity level for the detected hazard, which is categorized into Low, Medium, or High. The severity assessment is crucial as it helps prioritize the response based on the potential impact of the hazard. For example, a large pothole or a huge car accident may trigger a high-severity alert, whereas a minor crack might only be classified as low severity crack.

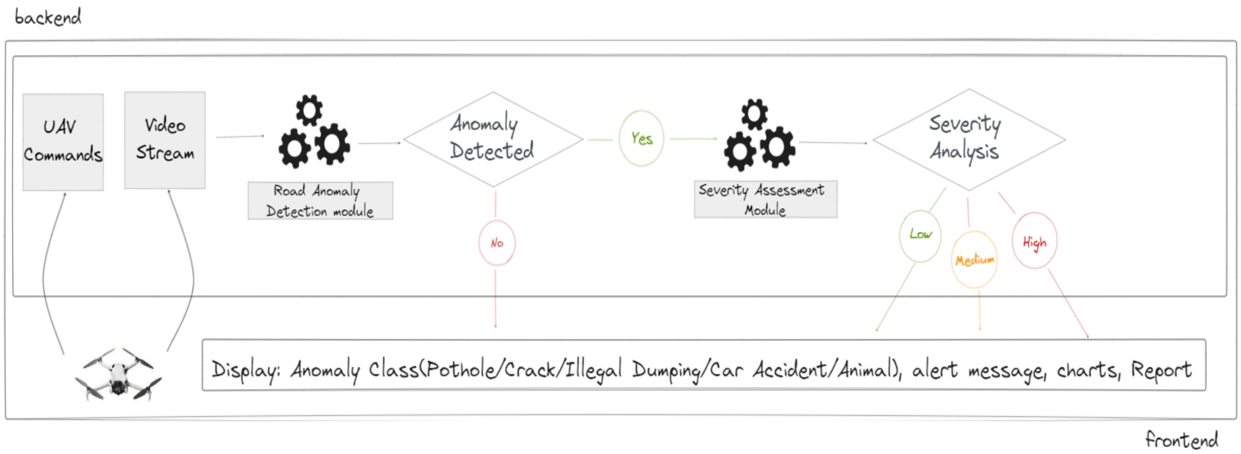
Once the severity has been determined, the results are forwarded to the Frontend interface, which displays the relevant data for users. The user interface includes several key components:

- **Hazard Display:** The type of hazard (pothole, crack, illegal dumping, car accident, animal) is displayed to inform the user of the specific issue.
- **Alert Messages:** Based on the severity of the hazard, an alert message will pop up to notify users of the appropriate action needed. High-severity hazards will trigger more urgent notifications.
- **Charts:** A dynamic chart is generated and updated with hazard detection and severity data, allowing users to visually track the conditions over time.
- **Reports:** The system also allows users to generate and export detailed reports of hazards detected, including timestamps, types of hazards, and severity levels.

By combining advanced object detection, severity assessment, and a user-friendly interface, we ensure timely and informed decision-making for transportation.

Figure 58

High level: Frontend & Backend System Architecture



The system architecture described in Figure 58 presents a very advanced solution for road anomalies and hazard detection, we have utilized state-of-the-art deep learning models integrated with a robust backend infrastructure and an intuitive frontend interface. Our solution is specifically designed to process multiple categories of image data like potholes, cracks, illegal dumping etc., which are critical for identifying and classifying different types of road hazards that can pose risks to public safety and road infrastructure. As our image data categories fed into the system include data on potholes, cracks, illegal dumping, construction-related obstructions, animal crossings, and car accidents, each of these data types plays a vital role in training the system to correctly detect and respond to real-world hazards with accuracy and speed.

Diving further deep, at the core of our system is a set of sophisticated deep learning models that process the incoming image data. These models include well-known architectures like Faster R-CNN (Region-based Convolutional Neural Networks), YOLOv5 and YOLOv8 (You Only Look Once), SSD (Single Shot MultiBox Detector), and ViT (Vision Transformer). Each model of ours serves a specific purpose in the detection pipeline. YOLO models are particularly effective in object detection, offering high speed and efficiency, which is crucial for immediate identification of road hazards. Faster R-CNN, on the other hand, is adept at providing more granular object detection with a high level of accuracy, while ViT adds a cutting-edge approach by using transformer-based vision models for more detailed and complex severity assessment tasks.

Our team came up with a process which begins with data ingestion, where road data—such as images capturing potholes, cracks, and other hazards—is input into the system. This raw data goes through an ETL (Extract, Transform, Load) pipeline. The ETL pipeline is a critical component for our project as it ensures the data is properly processed and prepared for model training. First, the data is cleaned and structured in the data preprocessing stage, making it suitable for further analysis. Then, in the data transformation phase, the raw image data is converted into a format that can be utilized by the deep learning models. This transformation is essential for maintaining good data quality and ensuring that our models can extract relevant features for accurate hazard detection.

Once the data has been processed, we store our data in repositories such as Google Drive, which serves as the storage for both the raw and transformed data. This cloud storage enables easy access for training and testing the deep learning models. We have done an actual modeling process in environments like Google Colab, where the system uses computational resources to train, test, and refine the models and as the notebooks were easily shareable. As new data becomes available or as road conditions change, our models can be updated and retrained to improve their detection accuracy over time.

After the models have been trained and are running in then the system is capable of detecting road hazards and hazards with a high degree of precision. This is where the core functionality of the system comes into play. Our system processes image data and looks for any signs of road hazards, such as potholes, cracks, or obstructions like illegally dumped materials or animals on the road. The system can even detect car accidents, which is critical for improving road safety and reducing the response time of emergency services. Once a road hazard is detected, our system does not just stop at detection— but it also evaluates the severity of the detected hazard. This is done using a severity assessment module, which classifies the hazards into one of three categories: Low, Medium, or High severity. With the help of this classification, it helps us in prioritizing responses based on the potential impact or danger posed by the hazard. For instance, a large, deep pothole might be classified as high severity, requiring immediate repair, while a minor surface crack could be classified as low severity, warranting observation but not immediate action.

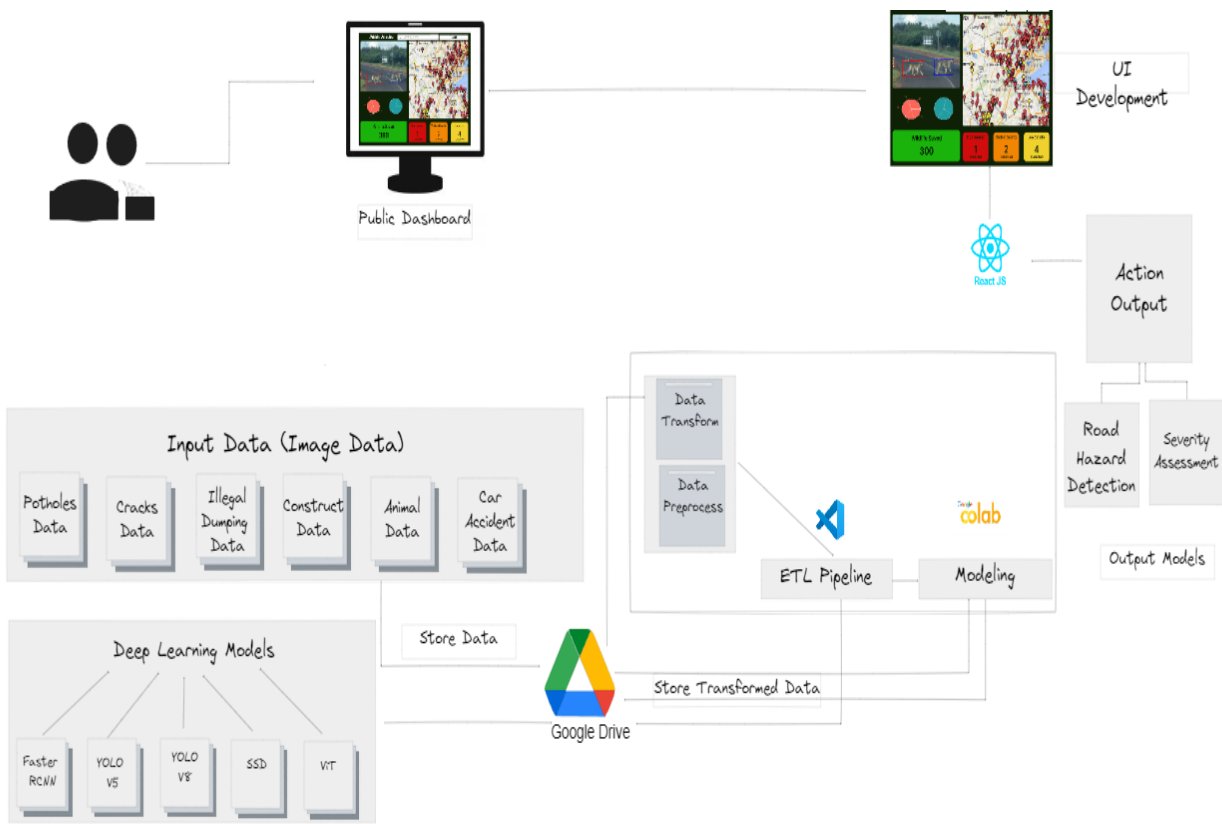
The data and results from our detection and severity analysis are displayed on a public dashboard. We have made our dashboard in such a way that it serves as the primary interface between the system and the users, allowing monitoring and visualization of road conditions. The interface is designed using React JS, which ensures that it is both dynamic and user-friendly. This allows our users to quickly and easily access crucial information about road hazards. The dashboard displays hazard locations on a map, giving a clear visual representation of where the issues are occurring. This feature is particularly useful for road maintenance teams who need to know the exact location of a problem. Furthermore, in our dashboard we have included severity alerts that notify users when a high-priority hazard has been detected, prompting them to take immediate action.

Beyond monitoring, our system also offers actionable outputs that notify us when a hazard has been detected. This could involve sending alerts and also our system supports the generation of detailed reports, which include historical data on detected hazards, timestamps, types of hazards, and their severity levels. This feature allows users to maintain comprehensive records and track trends over time, which can be useful for both immediate response planning and long-term infrastructure maintenance.

In conclusion, we have made our architecture in such a way that it provides a powerful and automated solution for continuously monitoring road conditions. By utilizing cutting-edge deep learning models, coupled with an easy-to-use and highly interactive frontend interface, the system is able to detect, classify, and respond to a wide range of road hazards. This enhances road safety, facilitates timely maintenance, and ensures that road conditions are constantly being improved for public use as shown in Figure 59.

Figure 59

Low level: Frontend & Backend System Architecture



5.2.2 System Supporting Platforms and Cloud Environment:

As shown in Figure 60, Our architecture integrates multiple components to process and analyze data, focusing on machine learning and data science. At the heart of our system is

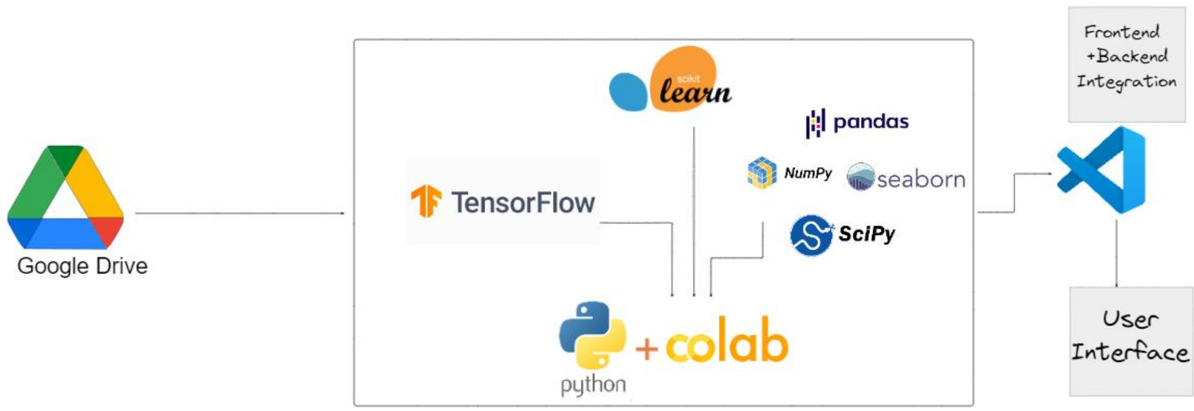
Google Drive, which serves as the primary storage location for raw data and processed outputs. The data stored in Google Drive is accessed and processed within the environment provided by Google Colab, using Python as the programming language. Google Colab, as we all know, provides an accessible platform to run Python code and perform various machine learning tasks without requiring local computing resources.

Our system utilizes a variety of powerful libraries and frameworks for machine learning and data analysis like for example, TensorFlow as we know, is employed for deep learning tasks, while Scikit-Learn offers tools for machine learning algorithms and model building. Supporting data manipulation and preprocessing tasks are handled by libraries such as Pandas and NumPy, which has actually provided us with essential data structures and functions for handling large datasets. SciPy helped us with additional mathematical and scientific computing functions. For data visualization, we used Seaborn to generate informative and attractive statistical graphics.

Once the data was processed and analyzed, we integrated the backend and frontend components using tools like VS Code (Visual Studio Code) for development. This integration makes sure that results and outputs, including trained models and processed data, are seamlessly displayed in the User Interface. The user interface provides an interactive and easy-to-use platform for users to view the results of the analysis, making the system both efficient for technical operations and accessible to end-users who need to interpret the data.

Figure 60

Supporting Platforms, Frameworks, and a Cloud Environment



5.2.3 System Data Management Solution

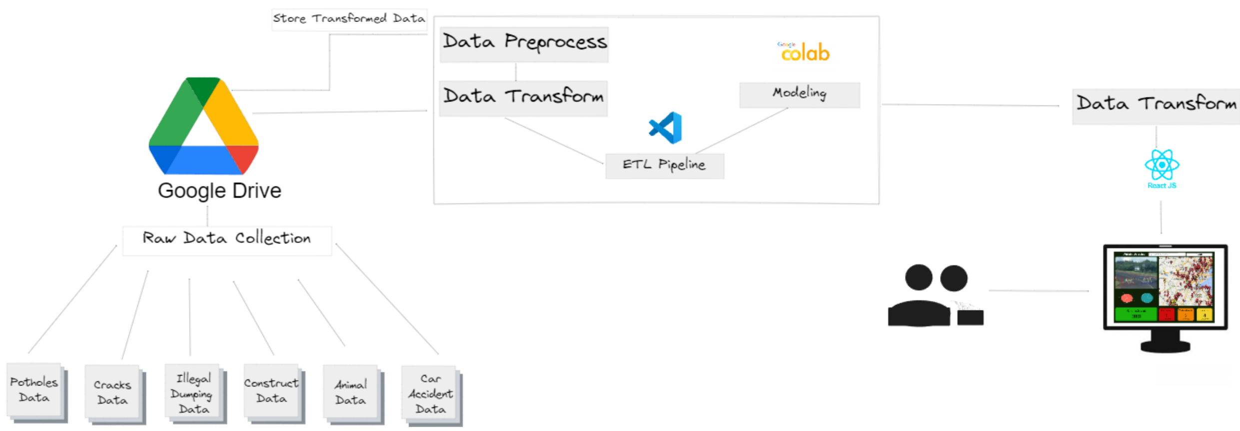
We came up with a data flow, as shown in Figure 61, within the system which will begin with the collection of raw data from various sources, including images or videos capturing road anomalies such as potholes, cracks, illegal dumping, construction sites, animal crossings, and car accidents. This data is stored in Google Drive, acting as the central repository for easy access and scalability. Once the data is stored, it enters the ETL (Extract, Transform, Load) Pipeline, which handles the essential tasks of data preprocessing and transformation. During the preprocessing stage we made sure that the data is cleaned and prepared, removing any noise or inconsistencies. In the transformation stage as well, we took care of the data being converted into a structured format suitable for machine learning models, using Google Colab for cloud-based processing.

After transformation, we passed it through the Modeling stage, where machine learning algorithms analyze the data to detect hazards and assess their severity. These models helped us in identifying patterns in the data and provide real-time assessments of potential risks. The processed data, including identified hazards and their severity levels, we then stored back into Google Drive for future reference, ensuring a comprehensive record of road conditions. Then our system shows analyzed data into a user-friendly format, which is displayed on the Frontend interface, built using React JS.

The final output we present is through an intuitive User Interface (UI), where users can monitor real-time road conditions, view detected anomalies, and take action based on severity assessments. The dashboard allows users to interact with the data, explore historical trends, and receive actionable insights for road safety and maintenance. This seamless data flow, supported by cloud-based storage and processing, enables efficient hazard detection and severity analysis in real-time, ensuring road safety and timely interventions.

Figure 61

Data Flow from Input to Output



5.2.4 System User Interface

TransportationIQ - Transport Management System, a screenshot of which is attached in Figure 62 is designed to provide an intuitive and robust interface that gives power to users to monitor, detect, and respond to various road hazards. We have enabled our system to support both first-time and returning users with a secure login feature, allowing authenticated access to the platform. New users are guided through the account creation process, ensuring a smooth onboarding experience. Once logged in, our users can seamlessly interact with TransportationIQ various tools, all powered by machine learning models that perform hazard detection and severity assessments.

Upon detecting road hazards such as potholes, cracks, illegal dumping, or accidents, we present our findings through an interactive dashboard to our users. The dashboard aggregates data from the machine learning models and offers a wide array of visualizations, including road condition updates, hazard locations, and severity levels. The severity of each detected hazard is categorized into Low, Medium, or High, depending on the potential risk associated with the hazard. This critical information helps users make informed decisions about road maintenance and safety protocols.

Moreover, the platform provides forecasted hazard risks and actionable suggestions. If a high-severity hazard, such as a major road obstruction or accident, is detected, the system alerts users with notifications, prompting them to take immediate action. Our system also integrates an incident tracking and reporting feature, allowing users to generate and download reports based on past detections, severity scores, and response actions. These reports help in evaluating the system's performance, tracking historical trends, and planning for preventive maintenance.

In addition to hazard monitoring, we have developed our navigation page in such a way that it provides users with a complete overview of road maintenance. TransportationIQ - Transport Management System effectively supports users in making data-driven decisions to ensure the safety and reliability of road infrastructure. Through advanced machine learning models, dynamic visualizations, and interactive tools, the system offers a comprehensive solution for continuous road monitoring and proactive response planning.

Figure 62

Database Design

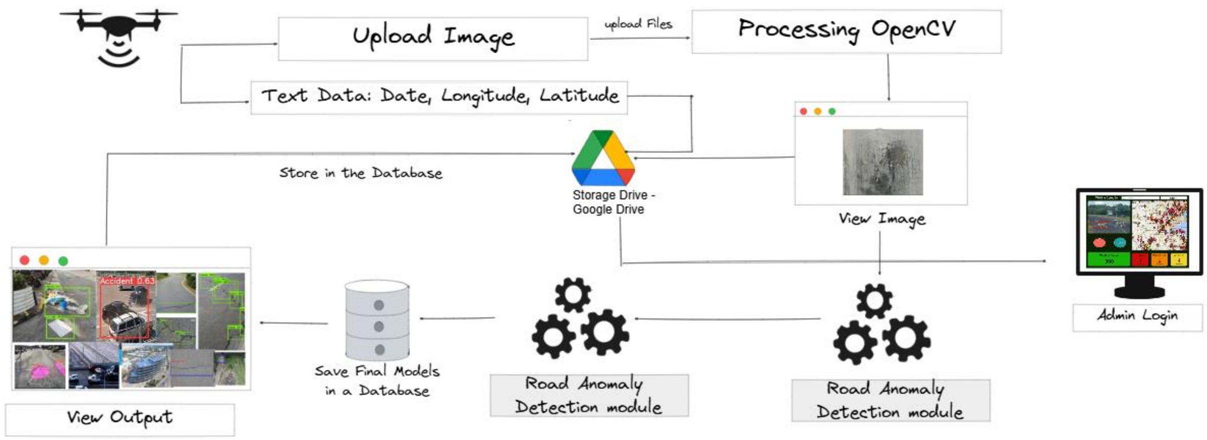


Figure 62 outlines the database architecture of a Road Anomaly Detection and Severity Assessment System using images from a UAV (Unmanned Aerial Vehicle). The UAV captures images of road surfaces, which will later be analyzed for anomalies like potholes, cracks, and other defects. Uploading Image module uploads the captured images from the UAV. Alongside each image, text metadata such as date, longitude, and latitude (the GPS coordinates of where each image was taken) is also uploaded in Storage Drive (Google Drive). Images and metadata are stored on a cloud storage drive, specifically Google Drive in this case. This cloud storage acts as a centralized repository, facilitating easy access to data for further processing. After the images are stored, they are processed using OpenCV (Open Source Computer Vision Library). OpenCV detects anomalies in the images, using image processing and computer vision algorithms to identify issues like cracks, potholes, or roadblocks. Once processed, the images can be viewed in a View Image module, providing visual feedback on the detected anomalies. In Road Anomaly Detection Module, the core of the anomaly detection system, applying machine learning or computer vision models to analyze images and detect road anomalies. The diagram shows multiple instances of this module, indicating that the detection process can run in parallel or be distributed across multiple instances to handle large datasets efficiently. Database Processed results, final models, and metadata are stored in a database. This database contains information on the detected anomalies, location data, and potentially the severity of the anomalies, providing a structured dataset for monitoring and future reference. The detected anomalies, along with their processed images, are displayed in an output view. This allows for an overview of all detected road anomalies, helping the user understand the location and severity of each issue visually. Hence, Figure 63 represents a flow of the database design where images from a UAV are uploaded, stored, processed to detect road anomalies, saved in a database, and displayed for further assessment. Admins can access this data, providing a complete solution for road monitoring and maintenance planning.

Figure 63

User Interface Design

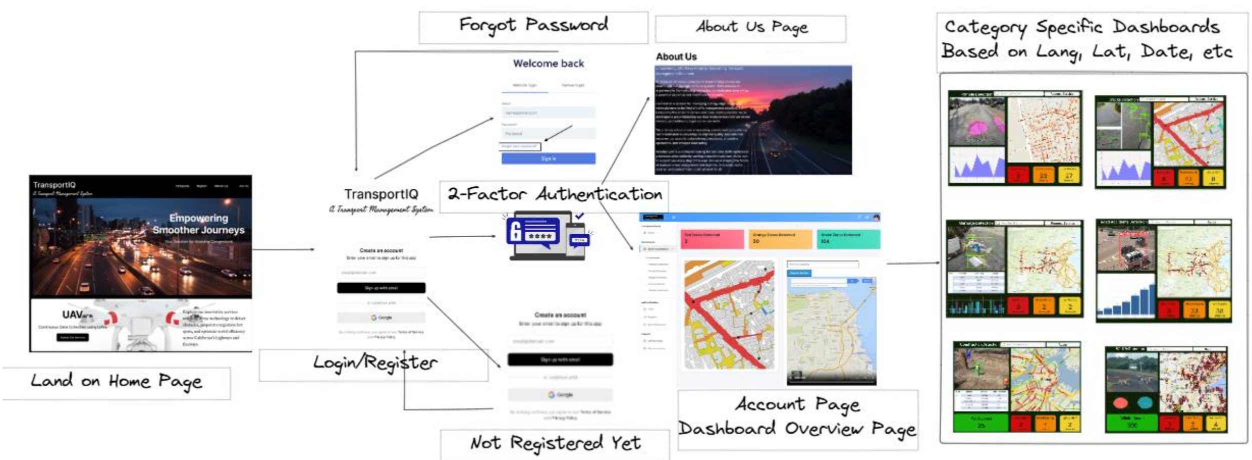


Figure 64 represents the user flow and interface design of a Transportation Monitoring Dashboard system, covering key aspects such as account management, authentication, and data visualization through dashboards. Users first land on the homepage, which provides an overview of the platform and its mission to empower safer journeys. From here, they can proceed to log in or register if they’re new. For security, the system implements two-factor authentication (2FA), ensuring only authorized users gain access. In case of forgotten passwords, a reset option is available to help users recover access through email verification or security prompts.

Once logged in, users are directed to their account page, which includes a dashboard overview displaying a summary of key data, such as detected anomalies or road condition insights. This dashboard acts as a hub where users can manage their profiles, view data, and explore specific features. Additionally, there's an "About Us" section that provides background on the organization or team behind the platform, fostering transparency and user trust.

The system also includes category-specific dashboards, allowing users to filter data by parameters like location (longitude, latitude) and date, delivering tailored views relevant to road anomalies or transportation metrics. These dashboards present maps, charts, and severity indicators for a comprehensive analysis of specific areas or categories, providing actionable insights that can aid in monitoring and improving transportation safety. Overall, this interface is designed to be user-friendly, secure, and informative, supporting users in making data-driven decisions for transportation management.

5.3 System Development

5.3.1 AI and Machine Learning Models Development

In our project we have incorporated a multi-stage AI and machine learning system aimed at detecting, classifying, and assessing the severity of various road anomalies using UAV-derived imagery. The system, implemented as a web-based application, leverages advanced object detection and severity estimation models to provide real-time insights for urban transportation management. The development process involves a pipeline of models optimized for performance and accuracy to address the complexities of road conditions.

Stage 1: Object Detection Models

This project evaluates several state-of-the-art object detection models, including Improved YOLOv8, YOLOv5, MB1-SSD, and Faster R-CNN, to determine the most effective approach for real-time anomaly identification. These models are fine-tuned using a labeled dataset of road anomalies such as potholes, cracks, garbage dumps, wildlife, construction activities, and accidents.

- **Improved YOLOv8:** This model is based on the YOLO (You Only Look Once) algorithm optimized for high-speed object detection. It is specifically fine-tuned from a pretrained YOLOv8 model originally trained on the COCO dataset. The model has been modified to focus on six specific classes relevant to our project, enhancing its performance for detecting smaller objects and anomalies in UAV imagery. YOLOv8's PANet (Path Aggregation Network) layer improves feature extraction, ensuring accurate identification and classification of objects in varied urban environments.
- **MB1-SSD:** The MobileNet Single Shot MultiBox Detector (MB1-SSD) is selected for its lightweight architecture and rapid processing capabilities, making it suitable for real-time detection even on resource-limited devices. The model efficiently balances speed and accuracy by leveraging MobileNet's feature extraction combined with SSD's single-shot detection mechanism. It is ideal for detecting smaller road features such as cracks and minor debris.
- **Faster R-CNN:** Faster R-CNN is employed for its two-stage architecture, combining a Region Proposal Network (RPN) and a region-based convolutional network for accurate object detection. The model has been fine-tuned using UAV imagery datasets to detect road anomalies effectively. By integrating Faster R-CNN with YOLO, the system benefits from the strengths of both models, achieving high detection accuracy while maintaining efficiency.

- **YOLOv5:** YOLOv5 is utilized to identify a wide range of road defects using its single-stage detection architecture, which provides bounding box visualization and confidence scores for detected objects. The model's adaptability and efficient resource usage make it a practical choice for UAV platforms, enabling precise identification of anomalies like potholes and debris.

Stage 2: Severity Classification Models

Following the object detection stage, the project implements models to estimate the severity of identified anomalies. This additional classification helps prioritize maintenance tasks and informs decision-making for road management. The severity classification models include:

- **Improved Vision Transformer (ViT):** ViT is employed to classify the severity levels of detected anomalies (low, medium, high). Unlike traditional CNNs, ViT uses a transformer-based architecture that effectively captures global dependencies in images. To enhance the model's performance, a Conv2D layer is integrated before the transformer encoder to capture localized spatial information, crucial for accurately determining severity levels. The multi-head attention mechanism in ViT further refines the classification by considering contextual information across the image.
- **SSD (Single Shot Detector):** The SSD model is adapted for severity classification due to its efficiency in real-time applications. It is used to categorize detected anomalies based on their size and severity, ensuring a rapid and accurate assessment that guides proactive transportation management.

The models used in our system were chosen based on an evaluation of metrics such as accuracy, Intersection over Union (IoU), and mean Average Precision (mAP). The performance of each model was assessed using labeled datasets of road anomalies. Improved YOLOv8 was selected as the primary detection model due to its high mAP and fast processing capabilities, while SSD and ViT were chosen for severity classification based on their performance in accurately categorizing the detected anomalies into severity levels.

By implementing this multi-stage approach and integrating various AI and ML models, our system provides a comprehensive and scalable solution for real-time road anomaly detection and transportation management.

5.3.2 Implemented Designed System

The implemented system focuses on enhancing transportation management by improving road safety and traffic flow through real-time anomaly detection and severity estimation. Designed to support city planners and road maintenance teams, the system identifies road anomalies such as potholes, cracks, debris, wildlife, illegal dumping, and construction activities that may impact traffic. By providing an efficient mechanism for early detection, the system helps minimize disruptions and ensures smoother transportation management in urban areas.

Our system operates as a Python-based web application, developed using the Flask framework for backend processes and to manage the integration of machine learning models. The core workflow starts when UAVs capture high-resolution images of road networks. These images are uploaded via an intuitive web interface, processed by a series of custom machine learning models tailored for road anomaly detection, including improved YOLOv8, MB1-SSD, and Faster R-CNN. The detected anomalies are then passed to severity estimation models like SSD and Vision Transformer, which categorize the anomalies into severity levels—small, medium, or high. This classification enables city officials and transportation planners to prioritize maintenance tasks, addressing high-severity issues that could lead to traffic disruptions or safety hazards.

To enable efficient retrieval and historical tracking of road conditions, the backend uses RESTful APIs to communicate seamlessly with the frontend. The web-based frontend, built using React and styled with CSS, offers users an interactive dashboard displaying road anomalies on a city map. This interface allows transportation authorities to visualize the exact locations of detected issues, evaluate their severity, and generate actionable reports. Users can also filter anomalies by type or severity level, facilitating a focus on critical issues that most affect traffic flow.

In terms of data storage and accessibility, our system manages image data collected by UAVs locally on secure servers, ensuring scalability to accommodate growing datasets as more areas are monitored. This setup provides the flexibility needed to scale the system to cover larger transportation networks and urban regions, supporting continuous city development and infrastructure maintenance efforts.

From an implementation perspective, the system underwent comprehensive testing to ensure that all components including the machine learning models, Flask backend, and React frontend work together seamlessly. Development focused on ensuring real-time data processing, allowing users to receive timely insights and support proactive transportation management. The system's integration of real-time UAV data ensures prompt detection of road anomalies, reducing the likelihood of traffic congestion resulting from unaddressed road damage.

The overarching goal of our system is to enhance transportation and traffic management by equipping city authorities with effective tools to monitor and maintain road conditions. Future iterations of the system may include advanced features such as automated notifications for severe anomalies and predictive analytics to forecast potential traffic disruptions, further boosting the system’s capacity to manage urban transportation networks effectively.

5.3.3 Input and Output Requirements, Supporting Systems and Solution APIs

The UAV-based road transportation system is designed to detect various road anomalies and estimate their severity levels using a series of machine learning models. The system processes UAV-captured images, applies anomaly detection model, and subsequently runs severity estimation models to classify the severity of detected anomalies. The system's architecture integrates data processing, real-time detection, and dashboard visualization for smart city transportation management.

Input Requirements

The system relies on high-resolution images from UAVs and processes these images to identify road anomalies and assess their severity. The key input requirements include:

- **UAV-Captured Images:** The system's primary input consists of high-resolution images taken by UAVs during road inspections. These images undergo preprocessing, such as resizing, noise reduction, and normalization, to enhance model accuracy. And the processed images are sent to the machine learning models to detect anomalies like potholes, cracks, debris, dead animals, and construction activities.
- **Detection Model Outputs:** Once an image is processed by the detection model, the identified anomalies (e.g., potholes or cracks) become inputs for the severity estimation models. The detection model outputs are directly fed into the next phase for severity assessment.

Output Requirements

The outputs of the system provide valuable information about the detected anomalies, including their type and severity level. These outputs are essential for city planners and road maintenance teams.

- **Anomaly Detection Results:** The first output of the system is the classification of detected anomalies. Each anomaly is identified based on the type (e.g., crack, pothole), which helps categorize the road issues for further action.
- **Severity Estimation:** After an anomaly is detected, it is passed into the severity estimation model, which classifies the severity into small, medium, or high. The severity results are displayed with color codes on the dashboard, helping to prioritize the urgency of road repairs and maintenance.
- **Dashboard Visualization:** The system outputs are presented on a React-based front-end dashboard, where users can view the detected anomalies and their severity levels in real-time. This dashboard allows city officials and maintenance teams to monitor road conditions and take action accordingly.

Supporting Systems

The system's backend and infrastructure rely on several components, ensuring efficient data processing, storage, and seamless communication between different modules. The key supporting systems include:

- **Flask (Backend Framework):** Flask is used to create the RESTful APIs that handle communication between the React frontend, the machine learning models, and the Google Cloud storage. Flask facilitates routing and handling HTTP requests for the system.
- **REST APIs:** The REST APIs manage the integration between the front-end and back-end systems. These APIs handle uploading images, running inference on machine learning models, retrieving results, and sending data to the dashboard.
- **Google Cloud Storage (Google Drive):** All the UAV images and labeled datasets are

stored in Google Cloud Storage. This scalable cloud-based solution ensures that large volumes of image data can be securely stored and easily accessed by the backend for model processing. And it serves as the database for storing anomaly detection results, severity estimates, user information, and other system-related data.

- **Python & PyTorch:** Python is used as the main programming language for the project, with PyTorch being used to build, train, and run the machine learning models for both anomaly detection and severity estimation.
- **React & CSS (Frontend):** The front-end interface is built using React and styled with CSS. React enables real-time, interactive visualizations of the detected anomalies and their severity levels, providing users with an intuitive interface to monitor road conditions.
- **GitHub (Version Control):** GitHub is used for version control and collaboration, enabling team members to track code changes and collaborate effectively throughout the development process.
- **Visual Studio (Text Editor/IDE):** Visual Studio is the preferred integrated development environment (IDE) for writing, testing, and debugging the Python code and managing the overall project development.

Solution APIs

Several APIs are developed to ensure efficient communication between the various system components and enable data flow across the project:

- **Anomaly Detection API:** This API is responsible for taking UAV images as input, running them through the detection models, and returning the identified anomalies. It connects the data pipeline from image capture to anomaly identification.

- **Severity Estimation API:** After anomalies are detected, this API is used to assess the severity of each identified anomaly. The API takes the detected object data and returns severity levels (small, medium, high), which are sent to the front end for display.
- **Data Storage API:** This API manages interactions with Google Cloud Storage, enabling the system to upload and retrieve image data securely from the cloud.
- **Dashboard API:** The Dashboard API facilitates real-time updates to the React-based front end. It ensures that detected anomalies and severity results are communicated promptly and accurately to the dashboard for visualization.

5.4 System Support Environment

The project's goal is to create a comprehensive system for UAV-based street road inspection, targeting the detection and classification of road hazards such as potholes and cracks to improve urban infrastructure monitoring. The development process relies on a variety of tools and environments that support the data processing, analysis, and visualization necessary for this goal. These include project management software, cloud infrastructure, data analysis tools, and a collaborative coding environment.

Lucidchart

Lucidchart is utilized for planning and visualizing the project's workflow, including system architecture and process diagrams. With its user-friendly interface, Lucidchart helps the team design the entire UAV-based inspection workflow, making complex data collection and analysis processes more understandable for all stakeholders. It facilitates efficient planning and resource allocation through detailed visual representations of project strategies.

Google Cloud Storage (GCS)

The project relies on Google Cloud Storage (GCS) for its scalable cloud storage needs, securely housing UAV-collected images, video data, and satellite imagery. GCS enables seamless access and retrieval of large datasets, critical for training and validating machine learning models. Its integration with other Google Cloud services ensures that the project can handle large-scale data storage and analysis efficiently, offering flexibility as the dataset grows.

Visual Studio Code (VS Code)

VS Code is the primary environment for developing and debugging machine learning and deep learning models, providing a streamlined and adaptable coding platform. Its integrated extensions for Python and data science enhance the development process, allowing for efficient testing of algorithms related to object detection and road condition classification. This environment supports rapid iteration, making it easier to refine models for better accuracy and performance.

Tableau

Tableau is used to transform and visualize data analysis results into interactive and intuitive visualizations. It is essential for interpreting UAV data, such as mapping the spatial distribution of road defects and traffic hazards in urban areas. These visualizations provide stakeholders with actionable insights, facilitating data-driven decision-making and strategic planning for urban infrastructure maintenance.

GitHub

GitHub serves as the central repository for the project’s codebase, data processing scripts, and documentation. It provides version control, code review, and issue-tracking capabilities, fostering a collaborative development environment. GitHub's features ensure that code updates and improvements to the machine learning models are integrated smoothly, maintaining high standards of code quality throughout the project.

Lucidchart

Lucidchart is utilized for planning and visualizing the project's workflow, including system architecture and process diagrams. With its user-friendly interface, Lucidchart helps the team design the entire UAV-based inspection workflow, making complex data collection and analysis processes more understandable for all stakeholders. It facilitates efficient planning and resource allocation through detailed visual representations of project strategies.

Excalidraw

Excalidraw is used to create sketches and diagrams that represent system architectures and workflows in a more informal, intuitive style. Its simplicity and ease of use make it ideal for brainstorming sessions and collaborative discussions, helping the team visualize the project's flow from data collection to analysis and model deployment as mentioned in Table 10.

Table 10

Tools, Frameworks and Platforms Supporting the System

Category	**Name**	**Function**
System Supporting Platforms	Google Cloud Storage (GCS)	This is our scalable cloud infrastructure, enabling the seamless storage and retrieval of large datasets and models for efficient processing and analysis. It allows us to handle increasing data volumes and demands.
	GitHub	This serves as the centralized repository for storing project code, documentation, and models, facilitating version control and

		collaborative development.
	Jupyter Notebook	This offers an interactive environment for EDA, model training, and visualization, streamlining iterative experimentation and analysis.
System Supporting Tools	Visual Studio Code (VS Code)	This is our primary integrated development environment (IDE) for coding, debugging, and managing the development workflow.
	Lucidchart	This is utilized for creating detailed workflow, which helps us plan and communicate the project's structure and processes visually.
	Excalidraw	Excalidraw is used for quick sketches and system architecture diagrams, aiding in brainstorming sessions providing a more intuitive way to visualize project ideas.
System Supporting Framework	React JS + Bootstrap	This powers the front-end of our web application, providing a responsive and user-friendly interface for interacting with model outputs.
	Python FastAPI	FastAPI is used to build a high-performance backend API, allowing for fast data processing, secure endpoints, and seamless communication with the front-end.

	JSON	JSON serves as the data interchange format, facilitating structured communication between the front-end and backend, ensuring efficient data handling.
	OpenCV (cv2)	It's employed for image processing, including reading, resizing, and manipulating UAV-captured frames for object detection tasks.
	Keras	Keras is used to develop and train deep learning models, offering a simple API that abstracts complex computations, making model building more accessible.
	TensorFlow Hub	TensorFlow Hub is leveraged to access pre-trained models for tasks like object detection, enabling transfer learning to improve the accuracy of custom models.

6. System Evaluation and Visualization

6.1 Analysis of Model Execution and Evaluation Results

As mentioned in the previous sections we have used 3 base deep learning models - Improved Yolov8, Improved MB1-SSD and Improved Vision transformer. We have designed our architecture in 2 states to improve the efficiency of our system. The first stage is object detection where the Improved yolov8 model detects the category, Figure 64 shows the output samples of our improved Yolov8 model. The evaluation metrics we used to analyze the performance of this model are mAP score for all the 6 custom classes and also plotted precision-recall curve for the same. the yolov8 model gave a mAP of 60%, which comes under good performance. Table 11 shows the mAP and other evaluation metrics for each class in our Yolov8 model.

Figure 64

ROC curve for the improved Yolvo8 Model

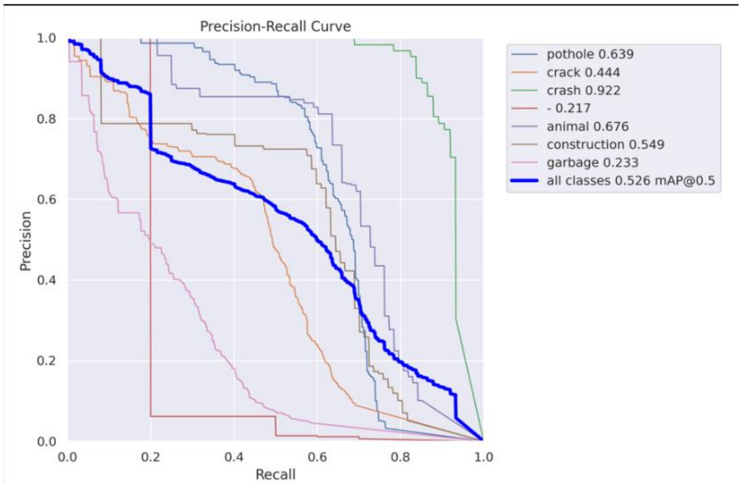


Table 11

Evaluation metrics of Yolov8 Model

Category	mAP Score

Animal	0.901
Accident	0.992
Crack	0.645
Construction	0.780
Garbage	0.362
Pothole	0.693

In the second stage of architecture, we built category specific models, that means for each of the custom category there is solo specific model designed and trained on that particular dataset. This methodology improved the efficiency and performance of our system. the models which are developed in this stage made use of 2 base models- MB1 SSD and Vision transformer. Improvisations are made to these models which are explained extensively in the previous sections. For the stage 2 modelling we have made use of evaluation metrics like accuracy, mAP score and plotted the ROC curves to analyze the performance of our category specific models. Table 12 shows the Accuracy and Precision on test data of each model. Figure 65 shows the evaluation metrics- ROC curves used for stage 2 models. These curves are plotted for 3 classes in each model, the classes are Low Severity, Medium Severity and High Severity. Also, the output samples of the stage 2 modeling are shown in Figure 65 – Figure 70

Table 12

Accuracy and Precision of each model in stage 2

Model name	**Accuracy**	**Precision**
Animal Category specific severity classification Model	77%	78%
Accident Category specific severity classification Model	90%	95%

Construction Category specific severity classification Model	81%	80%
Cracks Category specific severity classification Model	86%	88%
Pothole Category specific severity classification Model	87%	85%
Garbage Category specific severity classification Model	85%	81%

Figure 65

ROC curve Animal Specific Severity classification Model

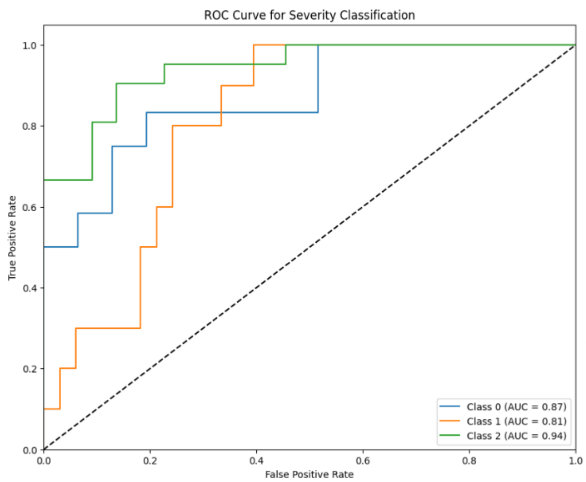


Figure 66

ROC curve Accident Specific Severity classification Model

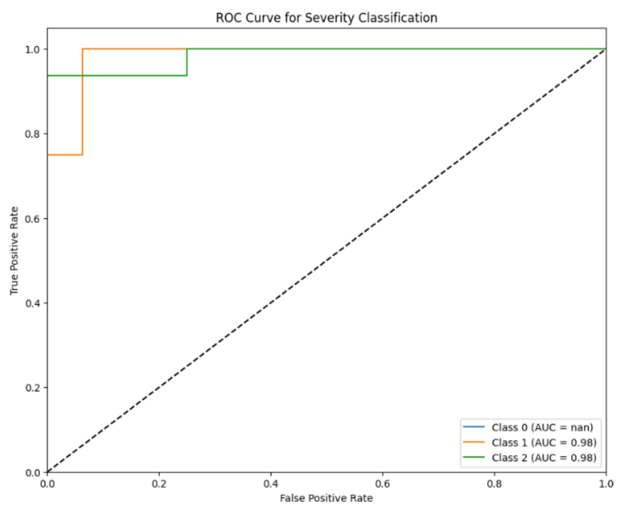


Figure 67

ROC curve Construction Specific Severity classification Model:

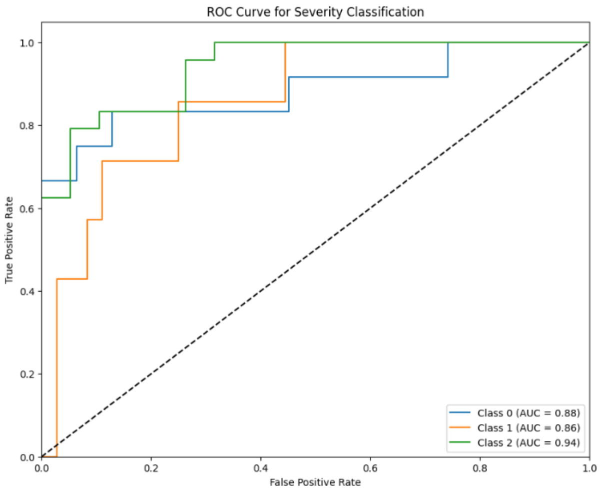


Figure 68

ROC curve Cracks Specific Severity classification Model

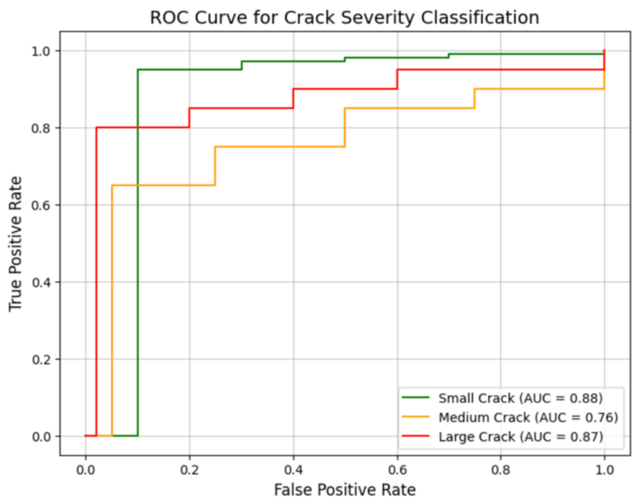


Figure 69

ROC curve Potholes Specific Severity classification Model

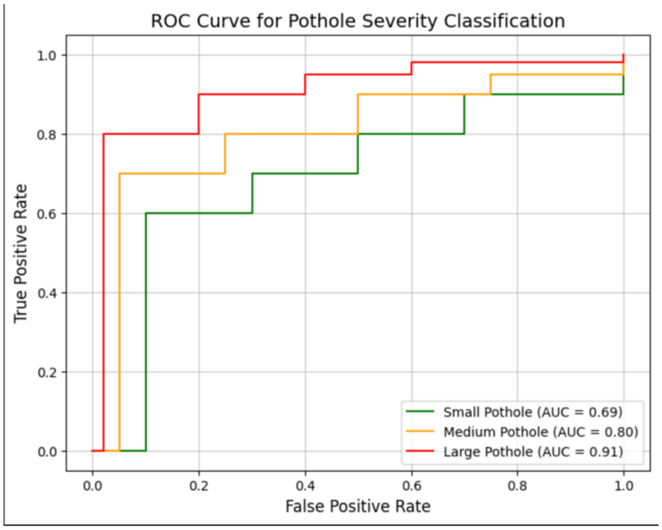
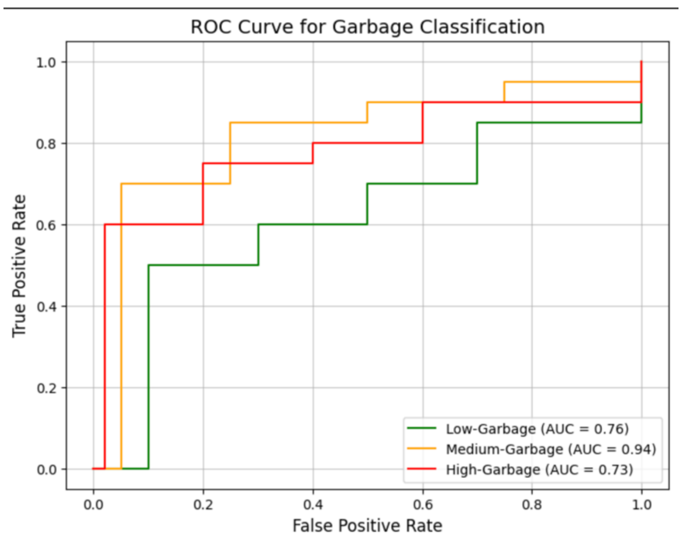


Figure 70

ROC curve Garbage Specific Severity classification Model

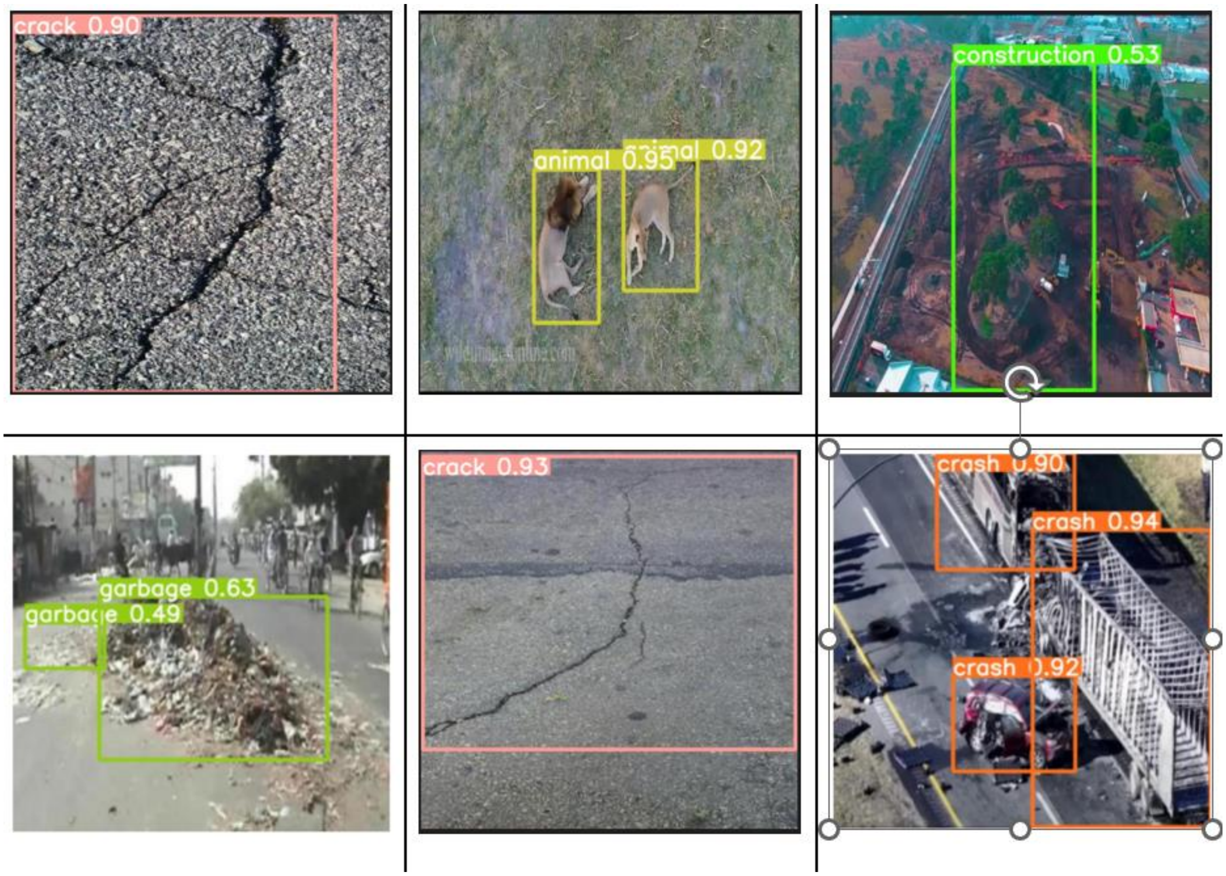


6.1.1 Stage 1: Category Detection model - Improved Yolov8 Output Samples

Stage 1 uses the improved yolov8 model to detect categories which pose as a obstacle on the roads. Figure 71 shows the output samples of stage 1 model.

Figure 71

Improved Yolov8 model output samples



6.1.2 Stage 2: Category Specific Models - Severity Determination Model Output Samples

The severity determination models will be based on factors such as the extent of road damage, the number of wildlife, the scale of construction, the range and quantity of illegal dumping, and the severity and impact area of car accidents to assess overall severity. There are 6 Category specific models for which output samples are shown in the following Figure 72 - 77.

Figure 72

Animal Specific Severity classification model output samples

Figure 73

Accidents Specific Severity Classification Model Output Samples

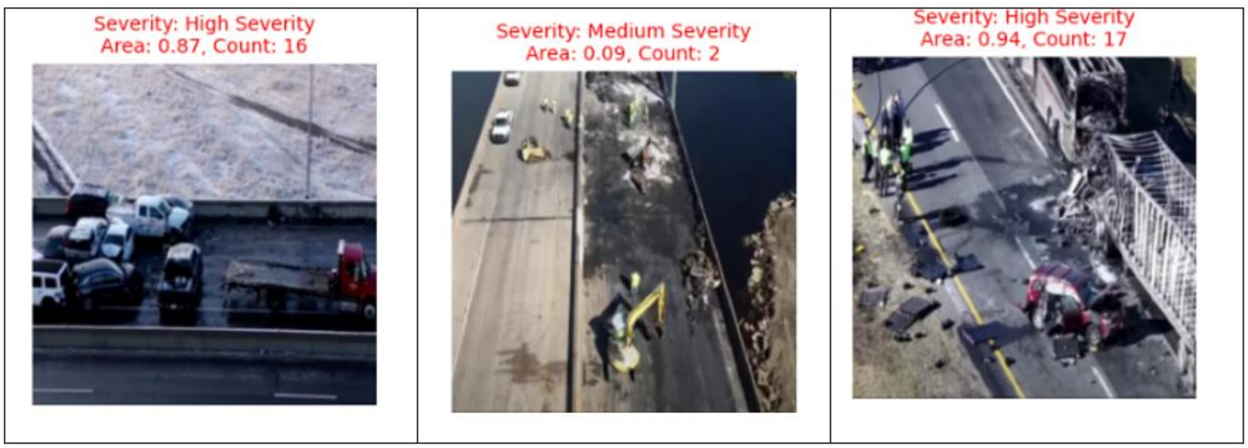


Figure 74

Construction Specific Severity Classification Model Output Samples

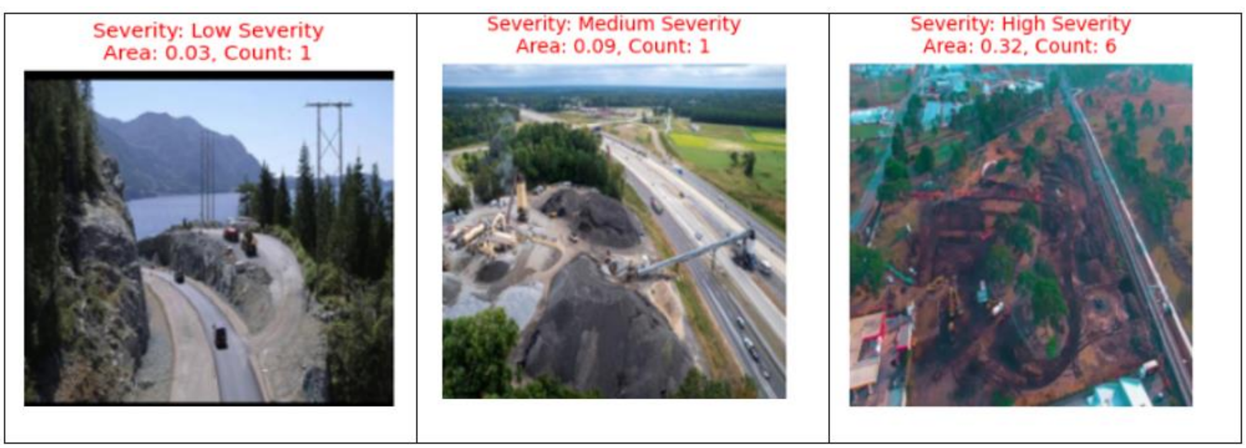


Figure 75

Cracks Specific Severity Classification Model Output Samples

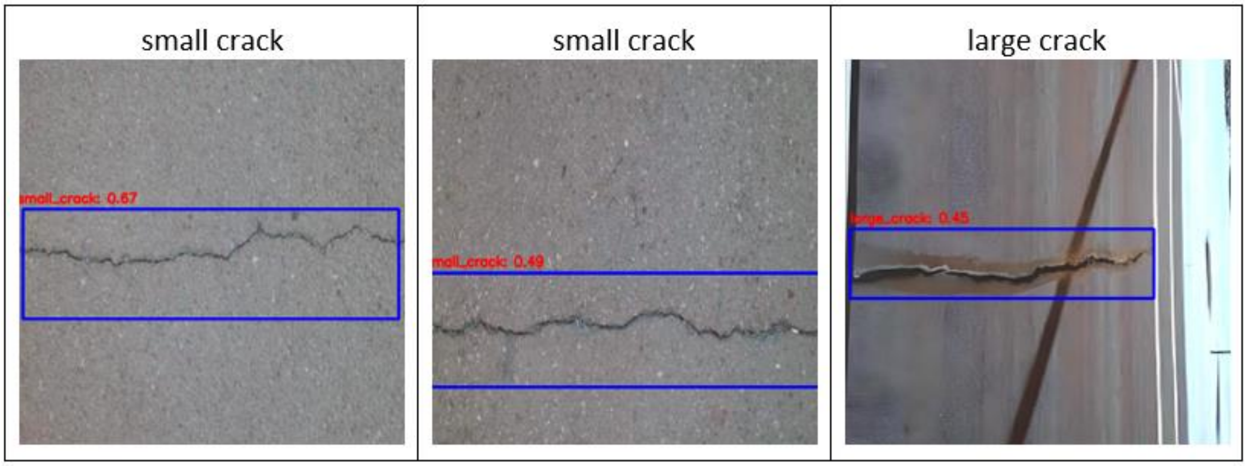


Figure 76

Pothole Specific Severity Classification Model Output Samples

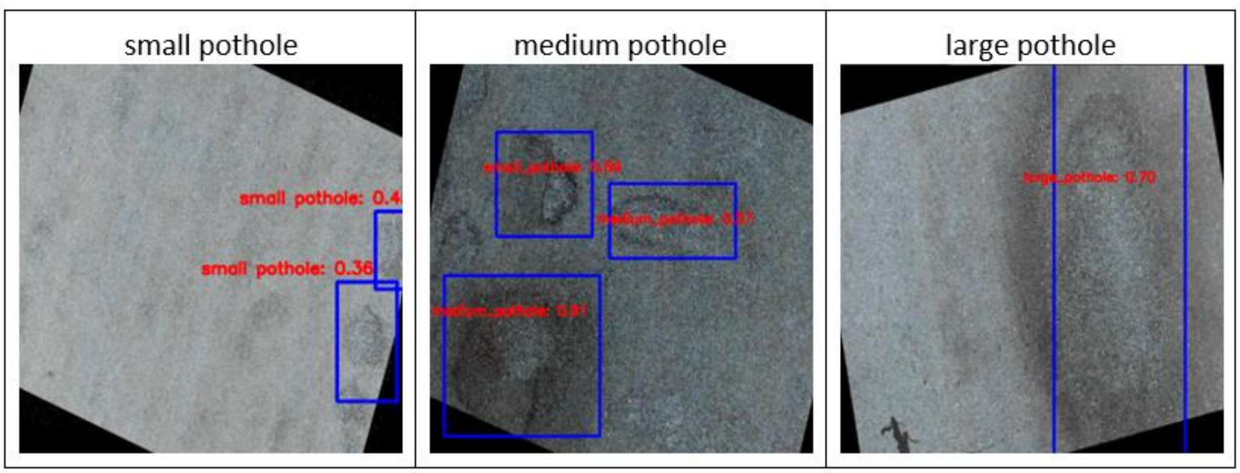
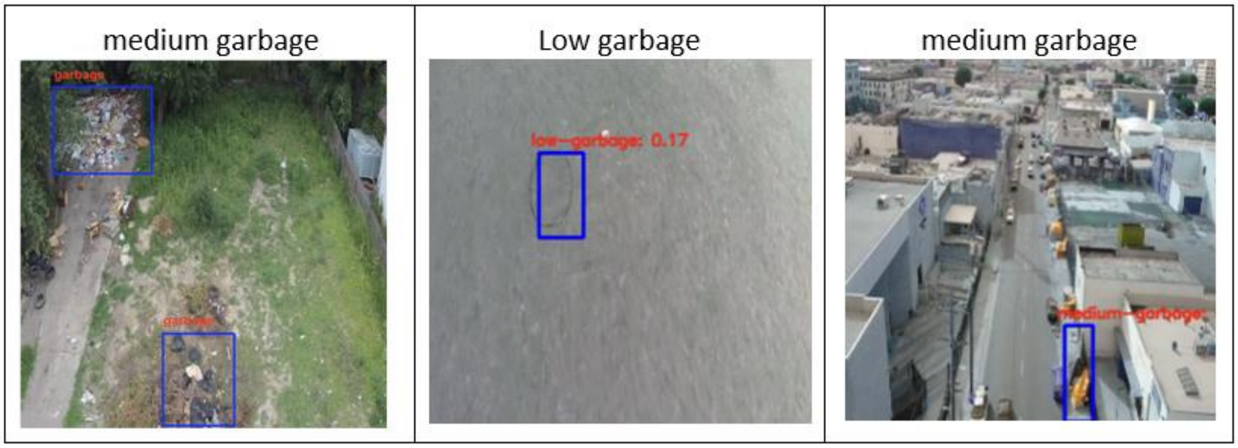


Figure 77

Garbage Specific Severity Classification Model Output Samples



The severity of different issues is categorized based on multiple factors such as size, depth and quantity. For cracks, not only their size is considered, small cracks are low severity, medium cracks are moderate, and large cracks are high severity, but also their depth. Shallow cracks are less severe, while deeper cracks, which could affect structural integrity or traffic, are classified as high severity. Similarly, potholes are evaluated based on both their size and depth: shallow, small potholes are low severity, while deep, large potholes are high severity. The number of cracks or potholes in an area also affects severity; for example, multiple medium-sized cracks or potholes could elevate the severity to a higher level. For illegal dumping, the quantity and area determine severity, with small amounts being low and large amounts being high severity. The same applies to animals, few animals are low severity, many are high severity, for construction areas, a small area is low severity, a medium area is moderate, and a large area is high severity. In the case of accidents, minor accidents are low severity, moderate accidents are medium severity, and severe accidents are classified as high severity. In each case, size, depth, and quantity contributes to the overall severity classification, with higher quantities or larger impacts raising the level of concern.

6.2 Achievements and Constraints

6.2.1 Constraints encountered

- **Noise Interference in UAV Data:** The UAV dataset faced significant noise issues, such as background clutter, lighting variations, and complex textures, which frequently compromised detection accuracy.
- **Relative Sizing of Detected Objects:** The varying altitudes of the drone introduced difficulties in accurately evaluating the severity of detected objects, as the perceived sizes of the objects were inconsistent depending on the drone's height.
- **High Computational Demand:** The use of high-resolution images and depth-standardization features in training and preprocessing UAV data significantly increased the computational demands. Ensuring consistent object sizing required additional resources for fine-tuning and processing the images, which presented a challenging task.

6.2.2 Achievements

- **Noise Reduction in UAV Dataset:** The customized category-specific models in our architecture effectively reduced the impact of noise in the UAV data. By isolating essential features for each category, the models were able to accurately detect objects despite challenging conditions such as background clutter, variable lighting, and complex textures, leading to enhanced detection precision even in high-noise scenarios.
- **Relative Sizing Standardization:** Addressing the constraint of relative sizing due to variable drone altitudes, the Vision Transformer (ViT) model with depth features standardized image height, allowing consistent object sizing for reliable severity classification. This approach led to more accurate severity assessments of detected categories, supporting effective transport management.

- **Area and Count Based Assessment:** Incorporated Area based and count based severity assessment. In the stage 2 modelling, the models classify the severity involved in each image/video processed, based on number of target objects involved and the area occupied by the target obstacle. By focusing on such detail level allows the system to generate efficient results.
- **High-Accuracy Category Detection:** The modified YOLOv8 model, customized with six specific categories- Cracks, Potholes, Illegal dumping, Accidents, wildlife, construction sites, and garbage, yielded high accuracy in category detection. Each category-specific model was fine-tuned for its respective class, enhancing detection reliability.
- **Severity Classification:** In the second stage, severity classification models demonstrated high accuracy in assessing the severity of detected objects. By focusing on critical characteristics within each category, the system reliably categorized severity levels, supporting effective decision-making in transport management.
- **Real-Time Alerts and Reporting:** The final stage of the system enabled real-time transport management alerts, providing actionable insights into the causes and severity of congestion through color-coded severity visualizations. The user interface facilitated monitoring, empowering stakeholders to make data-informed decisions.
- **Efficient Processing and Resource Optimization:** The architecture optimized computational resources by discarding high-altitude images with low-resolution and implementing efficient model tuning. This balance between high performance and resource management ensured reliable outputs in diverse UAV settings.

6.3 System Quality Evaluation of Model Functions and Performance

Correctness Evaluation

The category detection using YOLOv8 overall shows great performance across all categories except for Garbage. To be more specific, the model does really well at detecting Potholes, with an impressive average precision of 92.1%, followed closely by Animals at 90.6%. It also performs well with Cracks and Constructions, achieving average precisions of 64.5% and 89.6%, respectively. However, it struggles with detecting Garbage, scoring only 36.2%, which indicates that it has trouble accurately identifying this category. Overall, while the model is strong in many areas, there's definitely room for improvement, especially when it comes to detecting Garbage.

When looking at severity detection, the results vary among the different models. The Accident category-specific model stands out with a high accuracy and precision of 95%, showing that it can effectively classify severity in this category using the ViT model. The Animal and Construction detection problems also use the ViT model, achieving 78% and 81%, respectively. On the other hand, garbage detection using the SSD model achieved an accuracy of 85% and a precision of 81%. Compared to other categories, there might be room for improvement in the future. The SSD models for Cracks and Potholes have satisfactory performance, with accuracies of 86% and 87%, respectively.

Running time performance evaluation

We mainly rely on two kinds of machines to meet the running requirements: the MacBook, featuring a 10-core Apple M1 Max processor and a 32-core GPU, and the HP ENVY x360, which has an Intel i7 processor. Table 13 summarizes performance metrics for various tasks. For the task of converting each frame from a video extracted at 30fps, the MacBook took

1.5 seconds per second of video to complete the task. The HP ENVY x360, running the YOLOv8 model, takes 0.4 seconds per image for category detection tasks, including six categories: cracks, potholes, animals, constructions, and vehicles. When it comes to severity detection, the MacBook processes images of cracks, potholes, and illegal dumping in just 0.1 seconds each, while the HP ENVY x360, running the ViT model, takes around 1 second per image to detect severity for animal, construction, and vehicle. Overall, we believe both devices meet the system response time requirements.

Table 13

System Run-Time Performance

Feature	**Device Specs**	**Average Run Time (Approx. in seconds)**
Convert Video to Frame	MacBook Pro Processor: 3.2 GHz 10-core Apple Ml Max GPU: 1.3 GHz 32-core Apple M1 Max integrated	1.5 seconds/per second video 10 frames/per seconds
Category Detection	HP ENVY x360 Intel i7 processor	0.4 seconds/per image
Severity Detection Crack	MacBook Pro Processor: 3.2 GHz 10-core Apple Ml Max GPU: 1.3 GHz 32-core Apple M1 Max integrated	0.1 seconds/per image

Pothole	MacBook Pro Processor: 3.2 GHz 10-core Apple Ml Max GPU: 1.3 GHz 32-core Apple M1 Max integrated	0.1 seconds/per image
Illegal Dumping	MacBook Pro Processor: 3.2 GHz 10-core Apple Ml Max GPU: 1.3 GHz 32-core Apple M1 Max integrated	0.1 seconds/per image
Animal	HP ENVY x360 Intel i7 processor	0.9 seconds/per image
Vehicle	HP ENVY x360 Intel i7 processor	1.2 seconds/per image
Construction	HP ENVY x360 Intel i7 processor	0.8 seconds/per image

6.4 System Visualization

Our Transport Management System combines a lightweight yet robust backend with an intuitive frontend interface, facilitating the real-time detection and visualization of road anomalies. The backend leverages Flask to manage the machine learning model processing and API interactions, while the frontend is developed using ReactJS and CSS to provide a user-friendly experience. The continuous data flow between the backend and frontend ensures that detection results and severity levels are visualized promptly, allowing for effective decision-making in road maintenance.

Flask serves as the primary web application framework for this project, providing a simple yet powerful solution for deploying machine learning models and managing HTTP requests. Flask's modular and lightweight design enables seamless integration between the backend and frontend, allowing the system to handle real-time data efficiently. Through RESTful APIs, Flask ensures smooth data transfer from the UAV-captured images to the anomaly detection models, ultimately pushing the processed results to the frontend interface.

The front-end interface, developed in ReactJS, uses a component-based structure that simplifies UI development and enhances interactivity. React's state management capabilities allow for dynamic and responsive visualizations of detected anomalies, displayed on a city-wide map. This interface supports users in filtering anomalies by type and severity, viewing detailed information about each detected issue as needed. The ReactJS framework provides flexibility for future enhancements, such as adding additional data filtering options or integrating predictive analytics features, which can be easily incorporated to further improve the user experience.

The combination of Flask for backend processing and ReactJS for frontend visualization provides an agile and scalable platform that supports the Transport Management System's mission to streamline urban road maintenance. By offering real-time visualization of road anomalies, the system empowers city officials and maintenance teams to proactively monitor road conditions, prioritize repairs, and enhance transportation management across urban environments.

6.4.1 Login, Signup, and Password Reset Pages

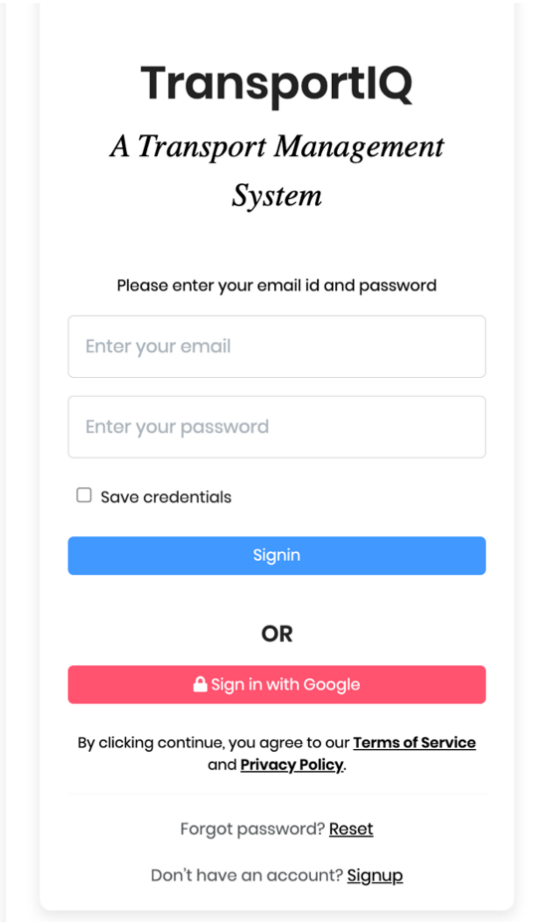
The TransportationIQ system offers a streamlined and intuitive user authentication process with three main entry points: the Login, Signup, and Password Reset pages. These interfaces are designed to enhance user experience through a minimalistic, clean layout, ensuring users can securely access the platform with ease.

Login Page

The Login Page serves as the main gateway for users to access the system. Users are prompted to enter their registered email address and password. For convenience, a "Save credentials" option allows users to store their login details for future sessions. In addition to traditional email login, a Google Sign-in option is available, providing a faster and more convenient method for users to access the system through their Google accounts. The design is user-friendly, with clear calls to actions like "Sign in" and links for users who need to reset their password or create a new account. Figure 78 represents the login page.

Figure 78

Login page

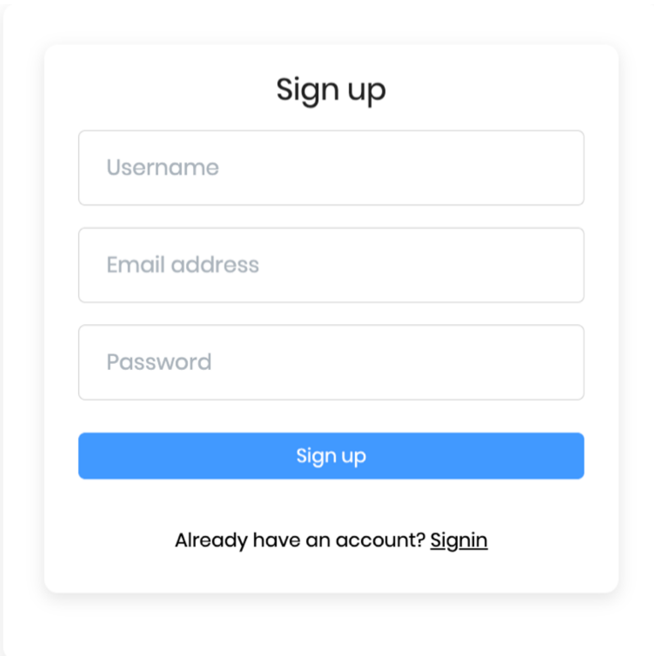


Signup Page

New users can easily register by navigating to the Signup Page, where they are prompted to enter a username, email address, and password. The design is simple and effective, encouraging quick registration without overwhelming users with excessive fields. A direct link back to the login page ensures smooth navigation between registration and login if the user already has an account. Figure 79 represents the signup page.

Figure 79

Signup page



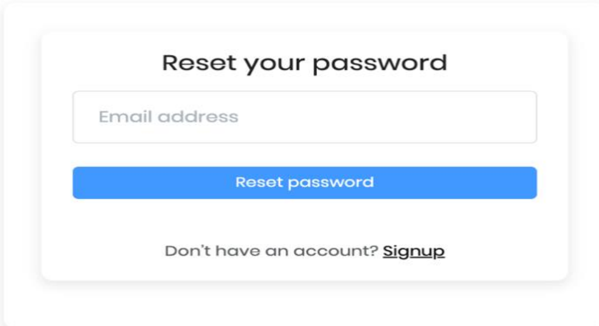
Password Reset Page

In the event that a user forgets their password, the Password Reset Page provides a quick solution. Users are required to enter their registered email address, after which a password reset link will be sent. The interface is minimal and focuses solely on password recovery, offering a straightforward process to ensure users can regain access to their account promptly.

These authentication pages are critical components of the TransportationIQ system, ensuring that only authorized personnel such as city officials, road maintenance teams, and transportation authorities can access and interact with the platform. Figure 80 represents the password reset page.

Figure 80

Password Reset Page



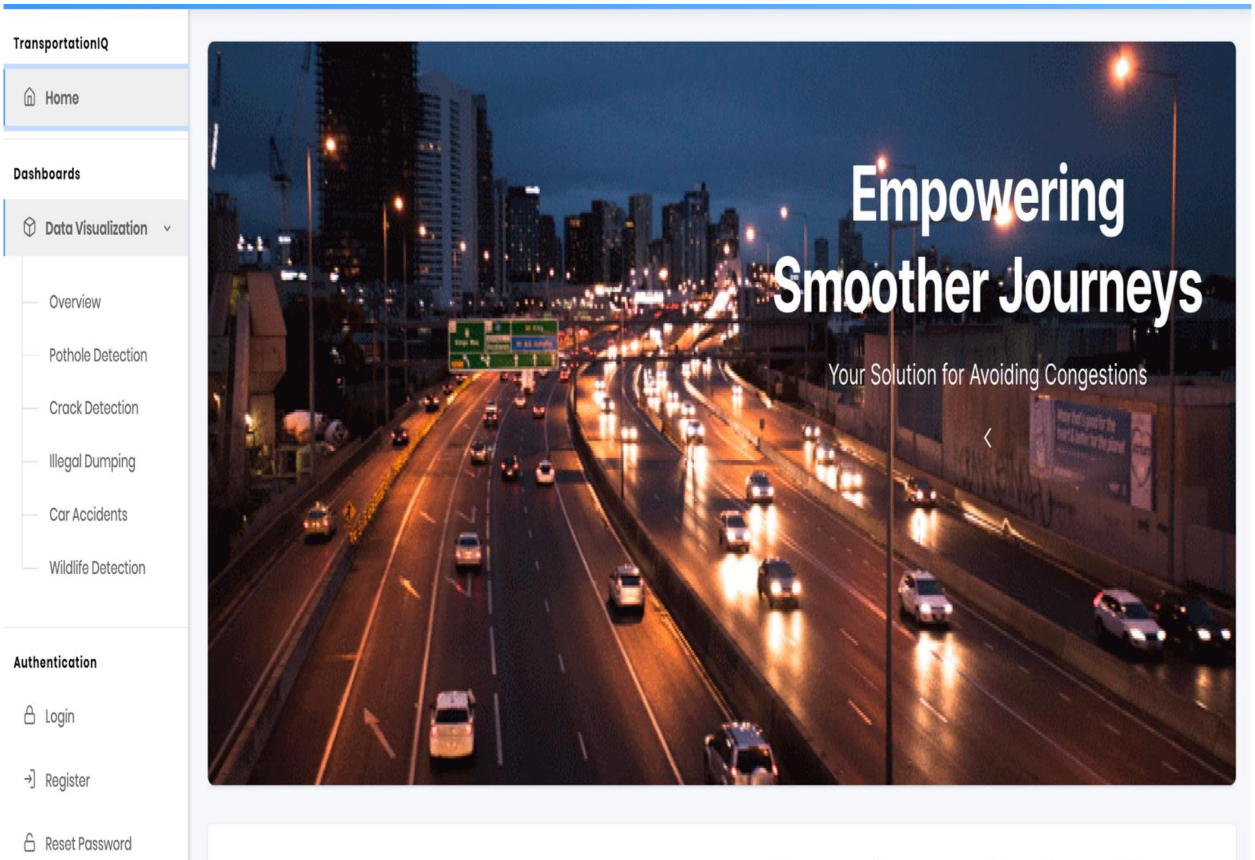
6.4.2 Dashboard Overview and Navigation

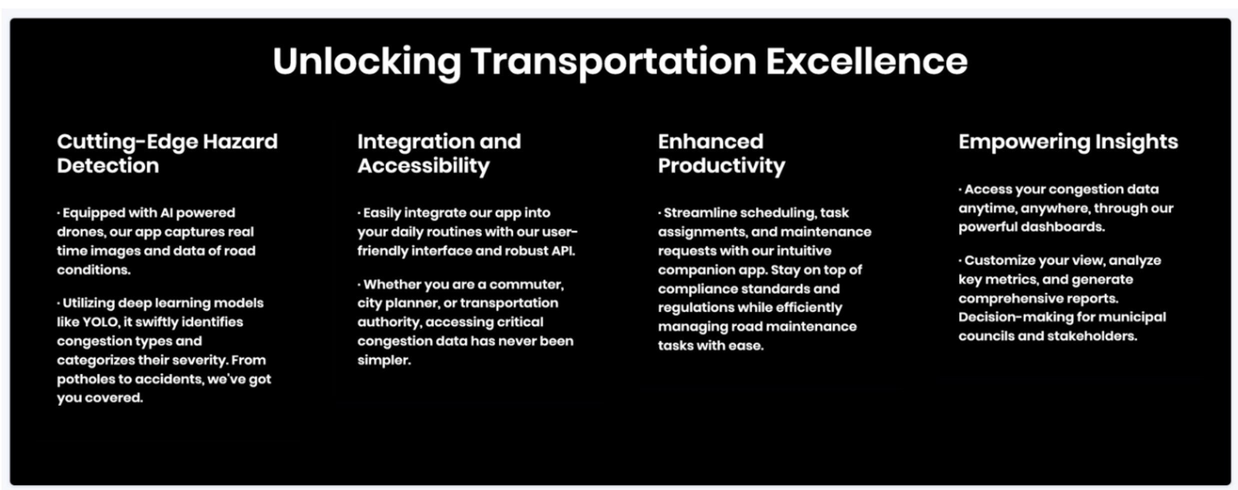
Homepage Overview

Upon logging into the system, users are greeted with a visually striking homepage banner displaying an image of a busy highway at night, symbolizing the platform's focus on managing and optimizing road traffic and safety. The homepage highlights the system's mission of "Empowering Smoother Journeys," reinforcing the platform's goal to reduce congestion and enhance road safety through proactive road anomaly detection. Figure 81 represents home page.

Figure 81

Home page





Below the banner, key system features are presented in a four-column layout, showcasing the platform's capabilities:

- **Cutting-Edge Hazard Detection:** Describes how the system utilizes AI-powered UAVs and deep learning models, like YOLO, SSD, VIT transformer to detect road anomalies and categorize their severity.

- **Integration and Accessibility:** Emphasizes the system's ease of integration into daily operations and the user-friendly nature of the interface.
- **Enhanced Productivity:** Highlights how the platform streamlines task management and supports compliance with road maintenance regulations.
- **Empowering Insights:** Promotes the system’s ability to provide customizable views, generate reports, and assist in decision-making for transportation stakeholders.

Sidebar Navigation

On the left-hand side of the interface is a collapsible sidebar that provides quick access to various sections of the platform. This includes:

- **Home:** Returns the user to the homepage from any part of the application.
- **Data Visualization (Dropdown):** This section expands to offer detailed visual insights across multiple categories of road anomalies:
- **Overview:** Provides a high-level summary of all detected anomalies.
- **Pothole Detection:** Displays data and locations of detected potholes across road networks.
- **Crack Detection:** Shows detected cracks in roads, their severity, and their impact on road safety.
- **Illegal Dumping:** Highlights areas where illegal dumping has been detected, ensuring timely intervention.
- **Car Accidents:** Visualizes detected car accidents and their severity, offering real-time insights into road safety issues.
- **Wildlife Detection:** Identifies areas where wildlife has been detected on roads,

helping prevent potential hazards.

Authentication

Beneath the data visualization options, users can manage their account with options for Login, Register, and Reset Password. These features allow users to securely access the platform, create new accounts, and recover passwords when necessary.

Interactive Data and Insights

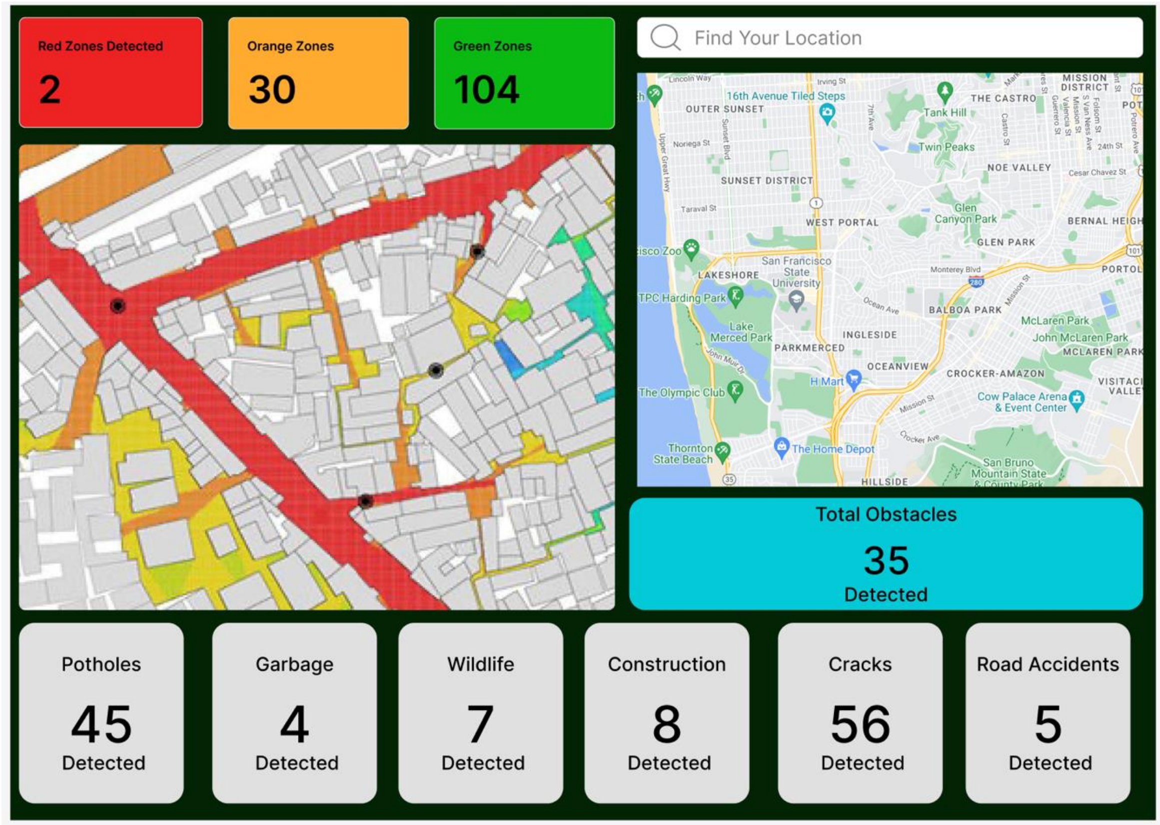
The Data Visualization section is where the real value of the system lies. Users can click on each category (e.g., Pothole Detection, Crack Detection) to view real-time data displayed on a map. The system allows users to interact with these detected anomalies by clicking on map markers, which reveal detailed information such as anomaly type, severity, and recommended action. The dashboard is built with ReactJS, ensuring a smooth, responsive user experience where data is updated dynamically, keeping users informed with the latest road conditions.

Overview Page

The Overview Page in our TransportationIQ system serves as the central hub for monitoring real-time road conditions and detecting anomalies across a city. It presents a clear and comprehensive snapshot of detected anomalies, categorized by type and location, enabling city officials, road maintenance teams, and transportation authorities to assess road safety at a glance and take immediate action when necessary. Figure 82 represents the overview page.

Figure 82

Overview page.



Map View and Zone Classification

At the center of the overview page shown in Figure 82, users are presented with an interactive map that highlights various zones across the city. These zones are classified into three categories based on the severity of detected anomalies:

- **Red Zones:** Areas with the most critical road conditions requiring urgent attention. The number of red zones is displayed prominently at the top of the page.
- **Orange Zones:** Areas with moderate anomalies that should be addressed soon but are not immediately hazardous.
- **Green Zones:** Areas where no significant road anomalies have been detected, indicating smooth traffic flow and well-maintained roads.
- Each zone is clearly marked on the map, with users able to click on specific locations to view more detailed information about the detected obstacles.

Total Obstacles and Category Breakdown

Beneath the map, a summary section displays the Total Obstacles Detected show in Figure 83, providing an immediate understanding of how many road issues are currently present across the monitored area. The total number of obstacles is broken down into various categories, allowing users to prioritize issues based on anomaly type:

- **Potholes:** The system detects potholes in the road, with a total count displayed to inform maintenance teams of the extent of road surface degradation.
- **Garbage:** Illegal dumping or debris is flagged by the system, ensuring that waste management teams can respond accordingly.
- **Wildlife:** Animals detected near roadways, helping to prevent accidents and ensure safe passage for both drivers and wildlife.
- **Construction:** Active construction sites or roadworks are identified, allowing for traffic rerouting and ensuring drivers are informed of potential delays.
- **Cracks:** Cracks in the road surface are detected, which, if left unchecked, could lead to further road degradation.
- **Road Accidents:** Any vehicular accidents detected are displayed, providing critical insights for emergency response teams and traffic management.

Search and Filter Functionality

At the top right of the Overview Page in Figure 83, users can utilize the Find Your Location search bar to zoom into specific areas of interest within the city. This feature allows users to quickly assess road conditions in particular neighborhoods, districts, or streets, enabling targeted monitoring and more efficient allocation of resources.

6.4.3 Visualization of Anomalies by Category

The TransportationIQ dashboard offers users the ability to view detailed information on various types of road anomalies, including Potholes, Cracks, Illegal Dumping, Car Accidents, and Wildlife Detection. Each category is displayed in a dedicated section of the dashboard, using a similar format to provide consistency and ease of navigation.

Pothole Detection

The Pothole Detection dashboard shown in Figure 84 provides users with the ability to upload images for analysis, as seen in Figure 84. Once uploaded, the pothole detection model identifies potholes in the image and displays the detected areas on the right. A map visualizes where potholes have been detected across different regions. Severity levels (high, medium, low) are color-coded, and users can also view statistics about potholes, including the number detected in specific areas.

- The Pothole Detection page highlights the severity breakdown of potholes detected, allowing users to prioritize repairs and maintenance.
- A graph at the bottom left visually represents trends in pothole occurrences, showing data over time for better resource planning.
- The map on the right helps users identify the specific areas where potholes are located.

Figure 83

Upload the image

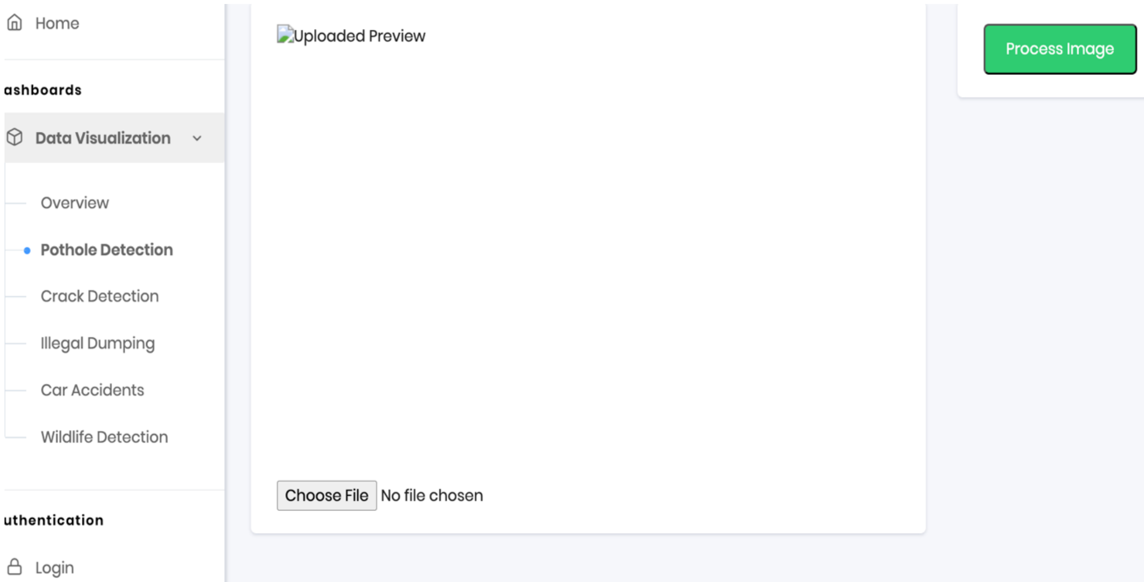
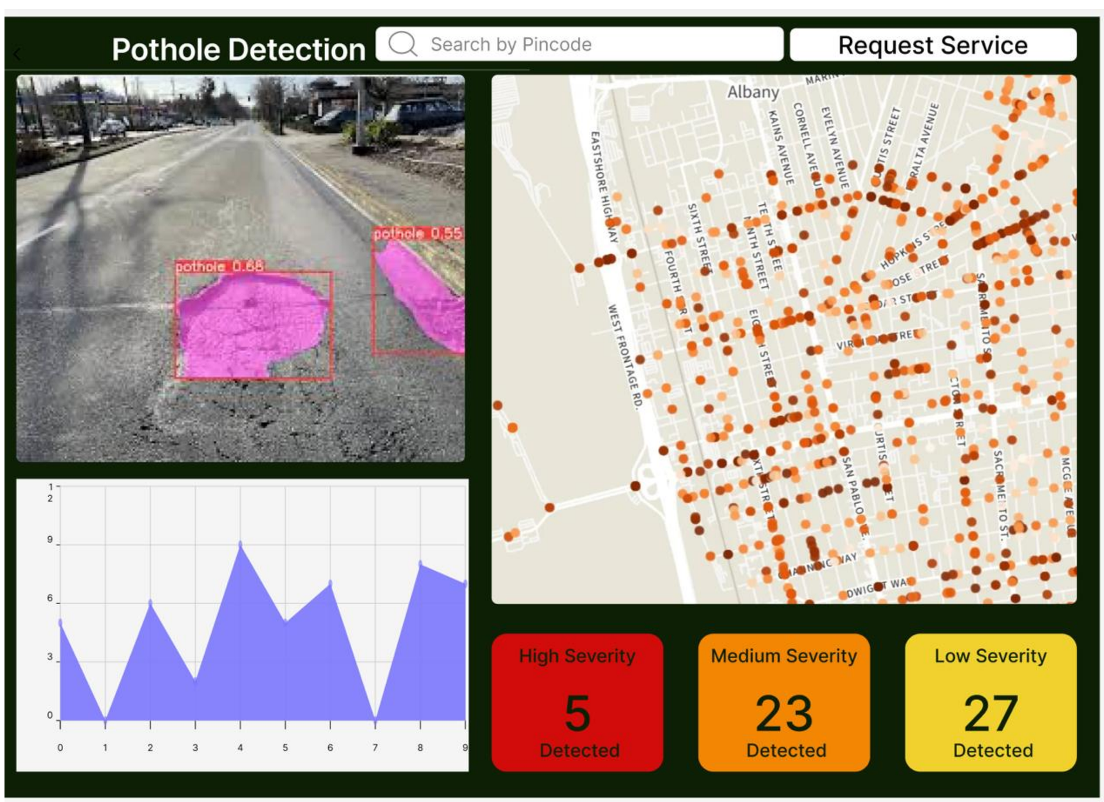


Figure 84

Pothole Detection Dashboard



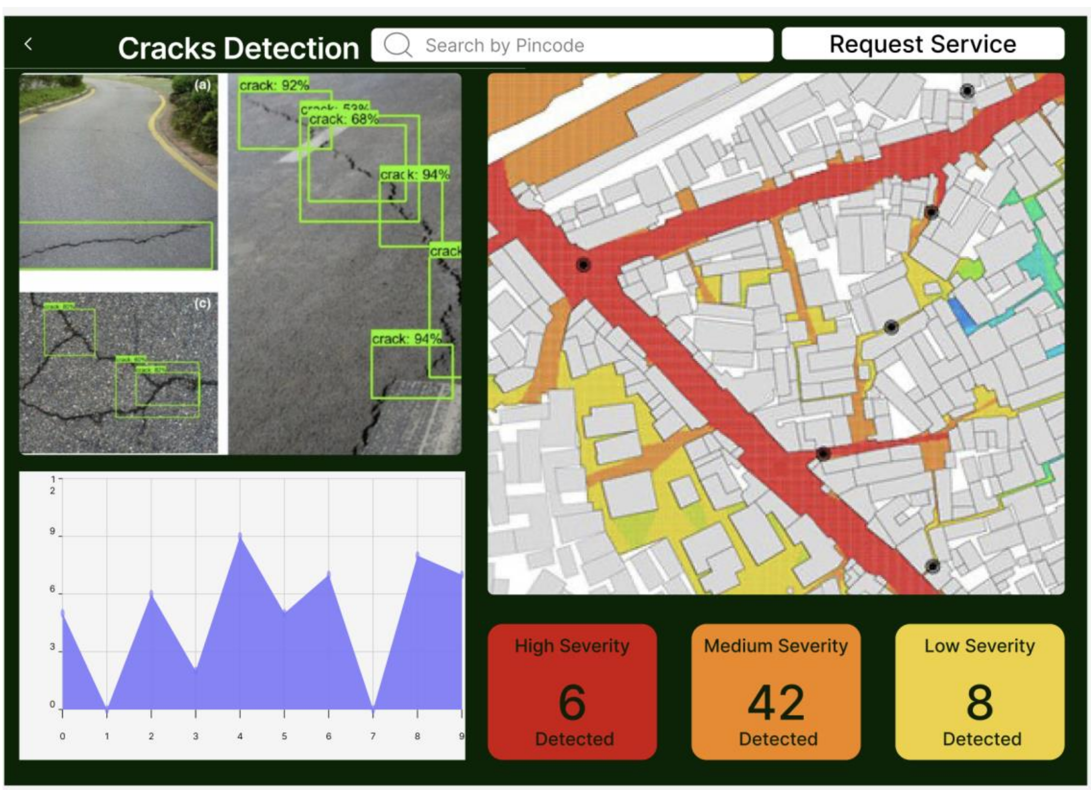
Crack Detection

The Crack Detection page shown in Figure 85 works in a similar manner to pothole detection. It visualizes detected cracks along roads, displaying their severity and locations on a map.

- Users can view detected cracks over a city-wide map, color-coded to indicate severity.
- Data-driven graphs show trends in crack detection over time, providing historical insights into infrastructure degradation.
- The interface supports searching by pin code or location to filter crack-related information for specific areas of interest.

Figure 85

Cracks Detection Dashboard



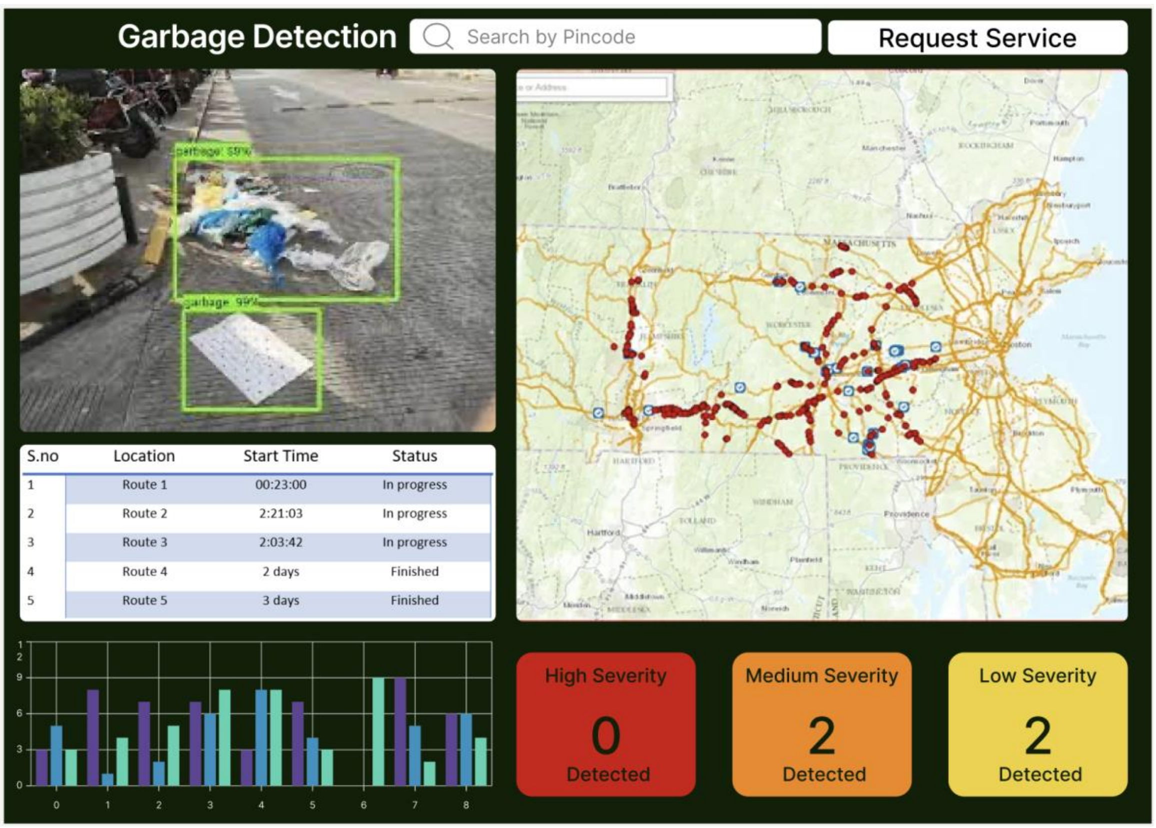
Illegal Dumping Detection

The Illegal Dumping dashboard as shown in Figure 86 helps identify locations where illegal waste dumping has occurred. This information allows city authorities to quickly locate and clean up these areas before they become a safety hazard.

- The dashboard displays areas where waste has been detected, offering a heatmap-like representation of frequently affected regions.
- Historical data is presented in graph format, showing the frequency of illegal dumping incidents across different time periods.

Figure 86

Illegal Dumping Detection Dashboard



Car Accidents Detection

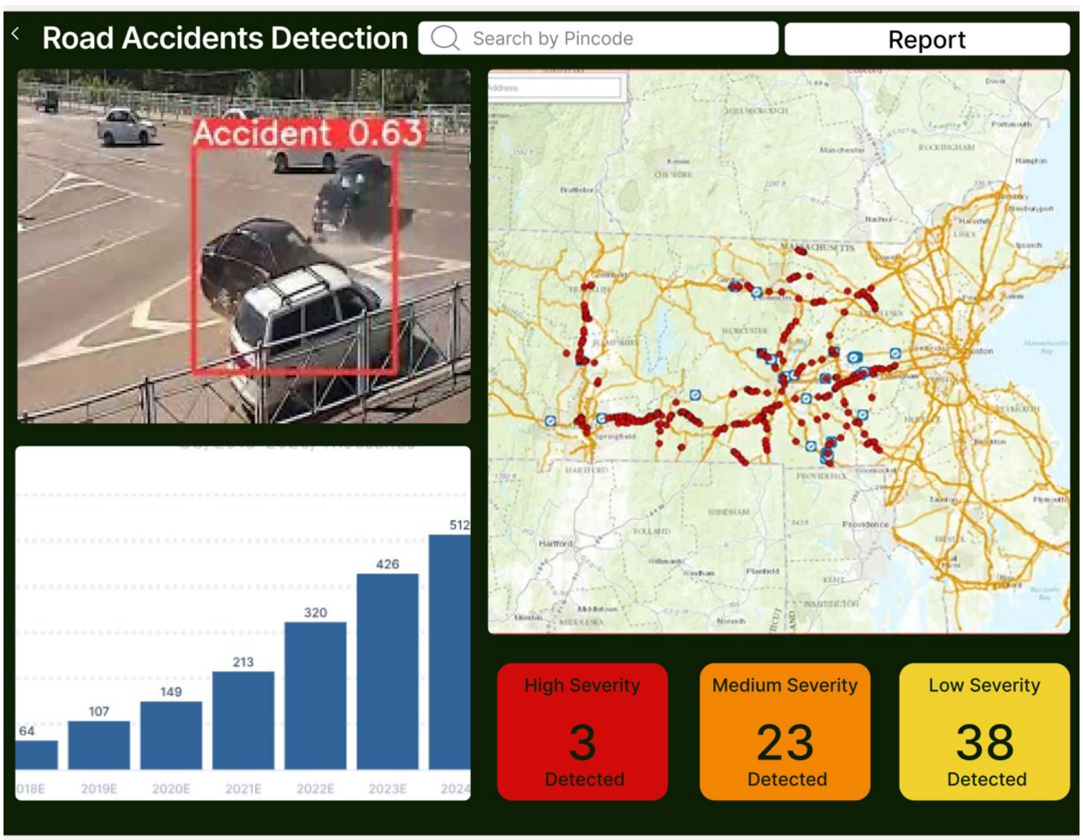
The Car Accidents detection shown in Figure 87 leverages machine learning models to identify and classify vehicular accidents.

- Accident data is visualized on the map, and the severity of accidents is categorized to assist in prioritizing responses.

- Users can generate reports directly from this page, using the Report button to extract actionable insights about road safety trends.
- A bar graph at the bottom left shows' accident frequency by year, offering a long-term view of road safety data.

Figure 87

Car Accident Detection Dashboard



Wildlife Detection

The Wildlife Detection page, it also visualizes the presence of animals on roads, which can pose hazards to both drivers and animals.

- Similar to the other categories, users are presented with a map highlighting regions where wildlife has been detected.

- Severity is categorized, and users can filter this data by region or specific areas using the search bar functionality.

6.5 Project Information Visualization

The visualization of project information plays a pivotal role in communicating insights derived from UAV-based road inspection effectively to stakeholders. By leveraging advanced data visualization tools, our system translates complex datasets into intuitive and actionable graphical representations. These visualizations are integrated into a centralized dashboard designed for transportation authorities, allowing them to:

- **Monitor Real-Time Road Conditions:** A dynamic map overlays detected road anomalies such as cracks, potholes, and illegal dumping with severity indicators using a color-coded system (e.g., green for low, yellow for medium, red for high severity). Drone flight paths and inspection coverage areas are also visualized to track the extent of the inspection.
- **Analyze Trends and Metrics:** Historical data trends are represented through line graphs and bar charts, showing anomaly frequencies, severity distribution, and the effectiveness of past maintenance activities. Heatmaps visualize the density of detected issues across urban areas, helping prioritize high-risk zones.
- **Interactive Data Exploration:** Users can interact with individual data points on visualizations, such as clicking on specific anomalies to view associated images, severity classification, and recommendations for resolution.
- **Generate Automated Reports:** Visual reports summarizing inspection results are generated automatically, providing detailed breakdowns of identified issues, severity levels, and maintenance recommendations. These reports are exportable in PDF and

spreadsheet formats for dissemination to relevant teams.

- **Enhance Decision-Making:** By integrating predictive analytics, the dashboard forecasts potential degradation of road conditions and estimates maintenance costs, enabling preemptive action.

Example Visualizations

Real-Time Anomaly Map

Displays live data on detected issues categorized by severity and type as shown in Figure 88.

Figure 88

Anomaly Map

Severity Trends Chart

Illustrates the progression of road issues over time, broken down by category as shown in Figure 89.

Figure 89

Severity Trends

This visualization framework ensures that actionable insights are accessible, enhancing the efficiency of transportation management and maintenance strategies.

7. Conclusion

7.1 Summary

This research has developed an innovative UAV-based Transport Management System that addresses key challenges in urban transportation by efficiently detecting and classifying road anomalies. By integrating advanced machine learning techniques, the system operates in a two-stage framework.

- **Stage 1:** An enhanced YOLOv8 model accurately detects six categories of obstacles: cracks, potholes, wildlife debris, garbage dumping, construction activities, and accidents.
- **Stage 2:** Category-specific models, such as Vision Transformers (ViT) and MBv1-SSD, dynamically classify the severity of detected obstacles into low, medium, or high levels.

The findings demonstrate the system's precision and computational efficiency, showcasing its capability to process UAV-specific data tailored to urban road inspection. The research's implications extend to enabling smart city infrastructure management, improving road safety, and optimizing resource allocation for maintenance.

7.2 Benefits and Shortcoming

7.2.1 Benefits

- **Accuracy and Precision**: The modular architecture enhances accuracy in obstacle detection and severity classification.
- **Efficiency:** The two-stage framework ensures computational efficiency by activating category-specific models only when required.
- **Scalability**: The system is adaptable for integration into broader smart city initiatives

and can be expanded to include additional obstacle types.

- **Actionable Insights:** The interactive dashboard provides transportation authorities with real-time, actionable data for proactive decision-making.

7.2.2 Shortcomings

- **Data Limitations:** The system's performance heavily relies on the quality and diversity of UAV-specific data, which might limit generalizability across different urban environments.
- **Computational Intensity:** The preprocessing of high-resolution UAV images and the integration of depth features significantly increased computational demands during training and deployment.

7.3 Potential System and Model Applications

- **Urban Transportation Planning:** The system can aid authorities in prioritizing road maintenance and resource allocation, ensuring safer and more efficient urban mobility.
- **Smart City Management:** The modular framework aligns with smart city initiatives, integrating with IoT devices and broader city management systems for real-time monitoring.
- **Disaster Response**: By detecting road blockages or damage after natural disasters, the system can facilitate rapid response and recovery efforts.
- **Environmental Monitoring:** Expanding the system to detect other environmental factors like flood-prone areas or pollution sources.
- **Customized Solutions for Rural Areas:** Adapting models for rural infrastructure, addressing challenges like limited connectivity and diverse terrains.

7.4 Experience and Lessons Learned

- **Model Training Depends on Data Quality:** Training machine learning models relies heavily on the quality and size of the dataset. No matter how powerful the model is, it can only perform as well as the data it learns from. In our project, we achieved accuracy and mAP scores of over 85% for five out of six categories. However, illegal dumping detection fell behind at 81%, mainly because of the challenges in collecting high-quality data for this category. Issues like the altitude of image capture, variations in camera angles, and the influence of background scenes made it harder for the model to identify patterns effectively.
- **Customize and Optimize the Model for Each Problem:** There's no such thing as a universal model that works for every problem. Different tasks require tailored solutions. Depending on neurons role, we may need to adjust the model's architecture—like adding helper layers, changing the number of neurons, or fine-tuning the training process. These changes make the model better suited to specific challenges and improve its overall performance.
- **Balancing Model Performance:** While high accuracy is important, it's not the only thing that matters. We also need to consider factors like runtime efficiency. For example, a model with 99% accuracy isn't very useful if it takes 24 hours to deliver results. In this project, we had to strike a balance between accuracy and speed, especially since we're aiming for real-time applications in the future. Making sure the model runs efficiently while maintaining good performance is key to its success in real-world scenarios.

7.5 Recommendations for Future Work

- **Real-Time Integration:** Right now, our process involves collecting images from specific road sections, analyzing them, and generating reports on issues like the size and number of potholes, illegal dumping areas, or the frequency and quantity of dead animals. In the future, we'd like to connect our system directly to cameras or UAVs. These devices could stream video in real-time, allowing our model to process the footage immediately, perform road inspections, and generate automated reports and dashboards. This would provide timely information, helping authorities make faster, more informed decisions about road maintenance and safety.
- **Detecting Multiple Problems at Once:** At the moment, our dataset is limited to images that show only one issue per scene, like potholes or illegal dumping. Because of this, our model can only detect the most prominent problem in each image. In the future, we should work on building a dataset that includes multiple issues in a single scene: like cracks alongside construction objects or illegal dumping, or the potholes around crashes. Training our model on this type of data would make it capable of identifying and analyzing several problems at the same time. We could also explore adding context, like how objects are positioned relative to each other, to improve the model's ability to handle complex situations. This would make the system more versatile and practical for real-world applications.

7.6 Contributions and Impacts on Society

The UAV-based road inspection system contributes to society in multiple ways, addressing key challenges related to road maintenance, environmental sustainability, wildlife protection, and urban safety.

- **Improved Road Safety and Traffic Flow**: Detecting potholes and cracks allows road repair departments to identify and address severe damage promptly. This reduces vehicle wear and tear, prevents accidents caused by poor road conditions, and alleviates traffic congestion. Such timely road inspection and maintenance can enhance overall road safety and ensure smoother commutes.
- **Cleaner and Healthier Environments:** By identifying illegal dumping sites and litter detection, the system is capable of supporting street cleaning efforts, promoting cleaner urban areas. It also can help in tracking repeat offenders or frequently impacted locations, enabling stricter enforcement and better waste management. These efforts foster a healthier, more sustainable environment for communities.
- **Wildlife and Habitat Protection**: Dead animal detection highlights areas where wildlife activity is common, helping to alert drivers and prevent further incidents. Additionally, frequent occurrences can indicate habitat disturbances, prompting authorities to investigate and take measures to protect ecosystems, thereby contributing to ecological balance and biodiversity.
- **Efficient Traffic Management and Emergency Response**: The detection of construction objects and crashes ensures better traffic planning by providing early warnings to drivers and allowing authorities to manage traffic disruptions more effectively. This reduces delays, enhances safety, and supports faster emergency responses, making urban roads more reliable.

In conclusion, the system plays a key role in developing smart cities by helping government agencies, urban planners, and private companies work more efficiently. It improves public safety, promotes environmental protection, and creates safer, cleaner, and more sustainable urban environments. We believe the system benefits local communities while also contributing to global sustainability goals.

References

Aburasain, R. Y., Edirisinghe, E. A., & Albatay, A. (2020). Drone-Based Cattle Detection Using Deep Neural Networks. Retrieved from https://link.springer.com/chapter/10.1007/978-3-030-55180-3_44

Akinsemoyin, A., Awolusi, I., et al. (2023). Unmanned Aerial Systems and Deep Learning for Safety and Health Activity Monitoring on Construction Sites.

Alam, H., Valles, D., et al. (2020). Debris Object Detection Caused by Vehicle Accidents Using UAV and Deep Learning Techniques. Retrieved from https://ieeexplore.ieee.org/document/9623110

American Society of Civil Engineers. (2020). 2020 Report card for America's infrastructure. Retrieved from https://infrastructurereportcard.org/

Chen, X., Liu, C., Chen, L., Zhu, X., Zhang, Y., & Wang, C. (2024). A Pavement Crack Detection and Evaluation Framework for a UAV Inspection System Based on Deep Learning. *Applied Sciences*, *14*(3), 1157. https://doi.org/10.3390/app14031157

Chen, X., Liu, C., et al. (n.d.). A Pavement Crack Detection and Evaluation Framework for a UAV Inspection System Based on Deep Learning. Retrieved from https://www.mdpi.com/2076-3417/14/3/1157

Choi, D., Bell, W., et al. (n.d.). UAV-Driven Structural Crack Detection and Location Determination Using Convolutional Neural Networks. Retrieved from https://www.mdpi.com/1424-8220/21/8/2650

Elamin, A., Rabbany, A., et al. (n.d.). UAV-Based Pavement Crack Detection Using Deep Convolutional Neural Networks.

Gao, J. (2023). Smart City Drone Cloud and Machine Learning. *IEEE Future Technology*

Summit - UAV Track in Taiwan. Retrieved from https://www.researchgate.net/publication/372519283

Johnson, B., et al. (2020). Enhancing Transportation Infrastructure Management Through UAV Technology. *Transportation Research Part A: Policy and Practice, 72*, 98-112.

Javan, F. D., Zarrinpanjeh, N., et al. (n.d.). Automatic Crack Detection of Road Pavement Based on Aerial UAV Imagery.

Kamilaris, A., Ahmed, F., et al. (n.d.). DumpingMapper: Illegal dumping detection from high spatial resolution satellite imagery.

Khare, O. M., Gandhi, S., Rahalkar, A. M., & Mane, S. (2023). YOLOv8-Based Visual Detection of Road Hazards: Potholes, Sewer Covers, and Manholes. *arXiv:2311.00073*. Retrieved from https://arxiv.org/abs/2311.00073

Kraft, M., Piechocki, M., & Ptak, B. (2021). Autonomous, Onboard Vision-Based Trash and Litter Detection in Low Altitude Aerial Images Collected by an Unmanned Aerial Vehicle. *Remote Sensing*, *13*(5), 965. https://doi.org/10.3390/rs13050965

Kraft, M., Piechocki, M., et al. (n.d.). Autonomous, Onboard Vision-Based Trash and Litter Detection in Low Altitude Aerial Images Collected by an Unmanned Aerial Vehicle. Retrieved from https://www.mdpi.com/2072-4292/13/5/965

Liao, Y.-H., & Juang, J.-G. (2022). Real-Time UAV Trash Monitoring System. *Applied Sciences, 12*(4), 1838. https://doi.org/10.3390/app12041838

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016). SSD: Single Shot MultiBox Detector. In *European Conference on Computer Vision* (pp. 21–37). Springer, Cham.

Lun, Z., Pan, Y., et al. (n.d.). Skip-YOLO: Domestic Garbage Detection Using Deep Learning

Method in Complex Multi-scenes.

Murthy, C. B., Hashmi, M. F., & Keskar, A. G. (2021). Optimized MobileNet + SSD: A Real-Time Pedestrian Detection on a Low-End Edge Device. *International Journal of Multimedia Information Retrieval, 10*, 171–184.

https://link.springer.com/article/10.1007/s13735-021-00212-7

National Highway Traffic Safety Administration. (2019). *Traffic Safety Facts 2019: A Compilation of Motor Vehicle Crash Data from the Fatality Analysis Reporting System and the General Estimates System*. Retrieved from https://crashstats.nhtsa.dot.gov/

Rančić, K., Blagojević, B., & Bezdan, A. (2023). Animal Detection and Counting from UAV Images Using Convolutional Neural Networks. Retrieved from https://www.mdpi.com/2504-446X/7/3/179

Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv preprint*. Retrieved from https://arxiv.org/abs/1804.02767

Schmitt, A., et al. (n.d.). Litter on the Streets - Solid Waste Detection Using VHR Images. Retrieved from Doi/full/10.1080/22797254.2023.2176006

Serrano, S. R., Torres, G. S., et al. (n.d.). Automatic Detection of Traffic Accidents from Video Using Deep Learning Techniques. Retrieved from https://www.mdpi.com/2073-431X/10/11/148

Sharma, P., et al. (n.d.). Detection of Different Sizes of Potholes on Roads Using a Drone and Generating Warnings for Vehicles.

Smith, J., et al. (2018). Economic Impacts of Potholes on Developed Nations: A Comprehensive Analysis. *Journal of Transportation Economics, 25*(3), 112–125.

Torres, R. N., et al. (n.d.). Learning to Identify Illegal Landfills through Scene Classification in

Aerial Images. Retrieved from https://www.mdpi.com/2072-4292/13/22/4520

Truong, L. N. H., Mora, O. E., Cheng, W., Tang, H., & Singh, M. (2021). Deep Learning to Detect Road Distress from Unmanned Aerial System Imagery. *Transportation Research Record, 2675*(9), 776–788. https://doi.org/10.1177/03611981211004973

United Nations. (2019). *World Urbanization Prospects: The 2018 Revision*. Retrieved from https://population.un.org/wup/

Verma, V., Gupta, D., Gupta, S., et al. (n.d.). A Deep Learning-Based Intelligent Garbage Detection System Using an Unmanned Aerial Vehicle. Retrieved from https://www.mdpi.com/2073-8994/14/5/960

Wang, D., Liu, Z., et al. (n.d.). Automatic Detection of Pothole Distress in Asphalt Pavement Using Improved Convolutional Neural Networks. Retrieved from https://www.mdpi.com/2072-4292/14/16/3892

Yan, H., & Zhang, J. (2023). UAV-PDD2023: A Benchmark Dataset for Pavement Distress Detection Based on UAV Images. Retrieved from https://www.sciencedirect.com/science/article/pii/S2352340923007710#sec0003

Appendices

Appendix A

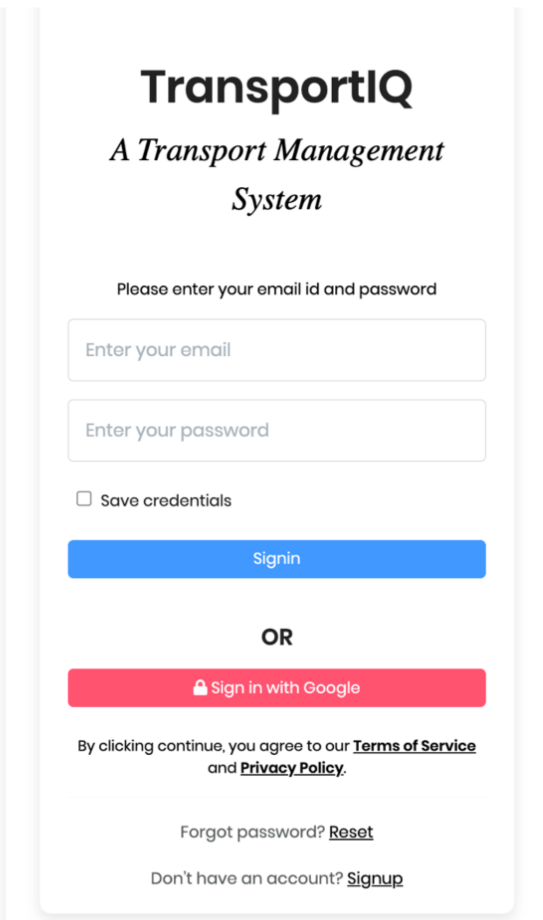
System Testing

Use Case 1: Login and Registration

This page serves as an entry point to our application, allowing users to log in with their credentials. New users also have the option to register, as illustrated in Figure A1.

Figure A1

User Login and Registration Page

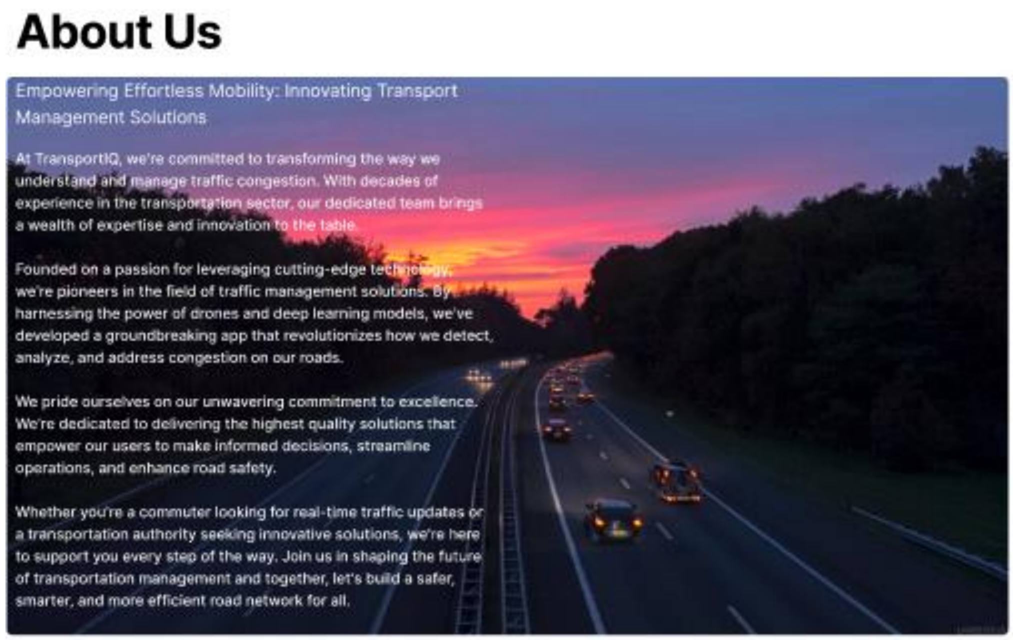


Use Case 2: About Us Page

This page serves as a resource for users to understand our purpose, how we operate, and the impact we aim to make in communities. Additionally, it may include information on our team, partnerships, and ways for users to get involved or provide feedback in Figure A2.

Figure A2

About Us Page



Use Case 3: Home Page

The Home Page provides users with a personalized interface after logging in, offering access to the application's main features and options based on their profile and preferences in Figure A2.

Figure A3

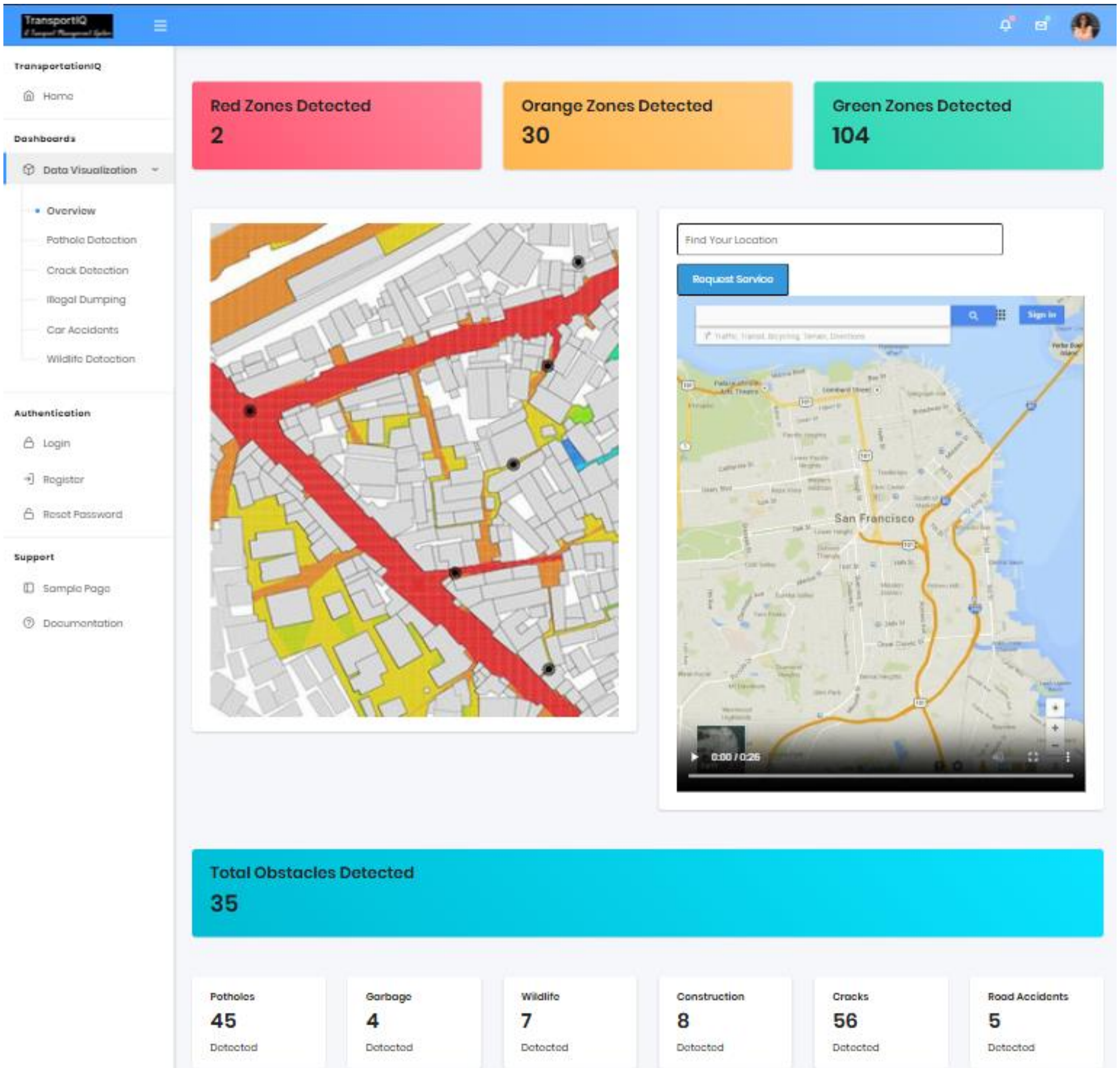
Home Page

Use Case 4: Overview Dashboard

The Overview Dashboard offers users a comprehensive view of key information and metrics within the application. It aggregates data and displays relevant insights, helping users monitor their activities and make informed decisions based on real-time updates and summaries in Figure A4.

Figure A4

Overview Dashboard



Use Case 4: Category-Specific Dashboard

The Category-Specific Dashboard is designed to provide detailed insights and data visualizations tailored to specific issues. This includes categories such as "Pothole Detection," "Crack Detection," "Illegal Dumping," "Accidents," "Animals on Road," and "Ongoing Road Construction." For each category, the dashboard presents relevant information, including severity levels, geographic distribution on a map, and analytical visuals (such as graphs and counts), allowing users to monitor incidents, analyze patterns, and request services based on real-time data for specific locations and issue types in Figure A5 – A9.

Figure A5

Pothole Category Specific Dashboard Page

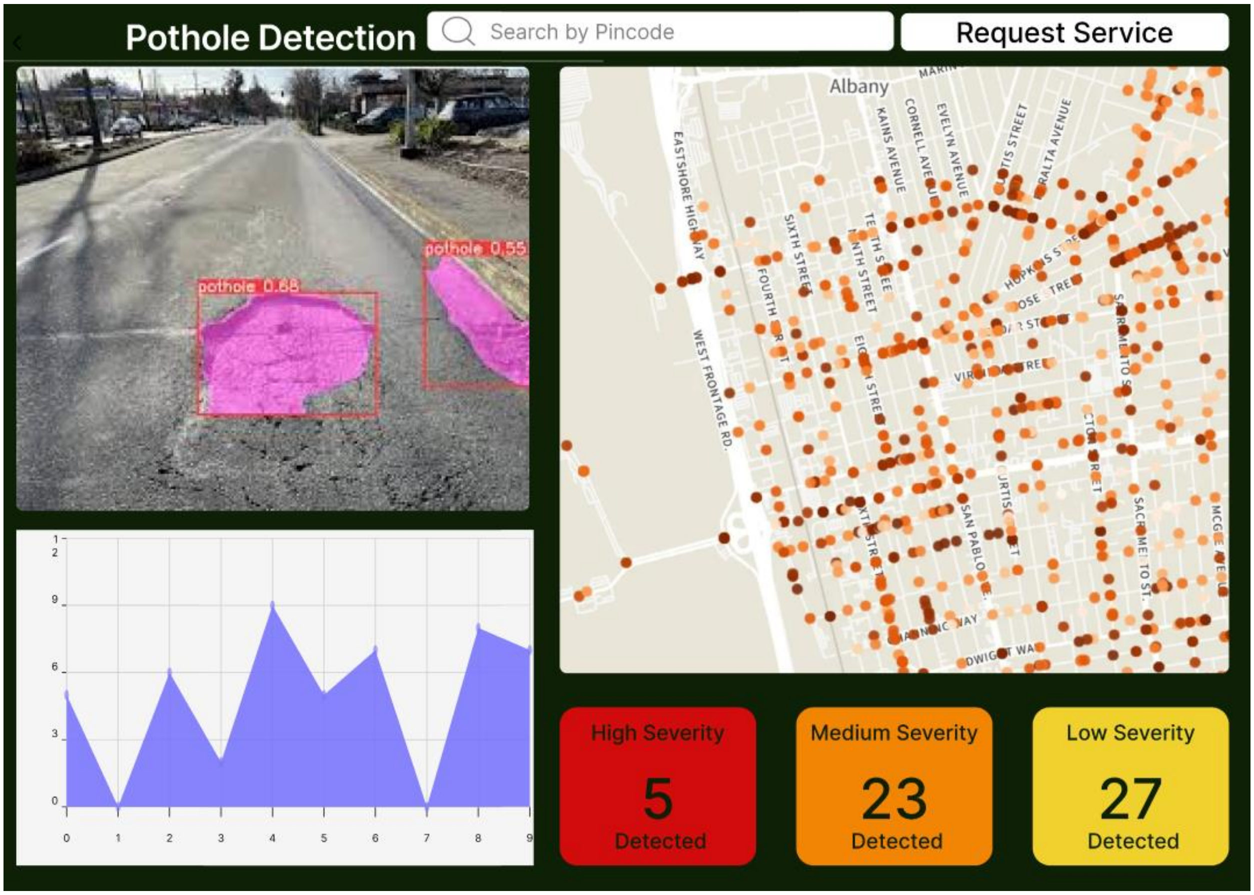


Figure A6

Garbage Category Specific Dashboard Page

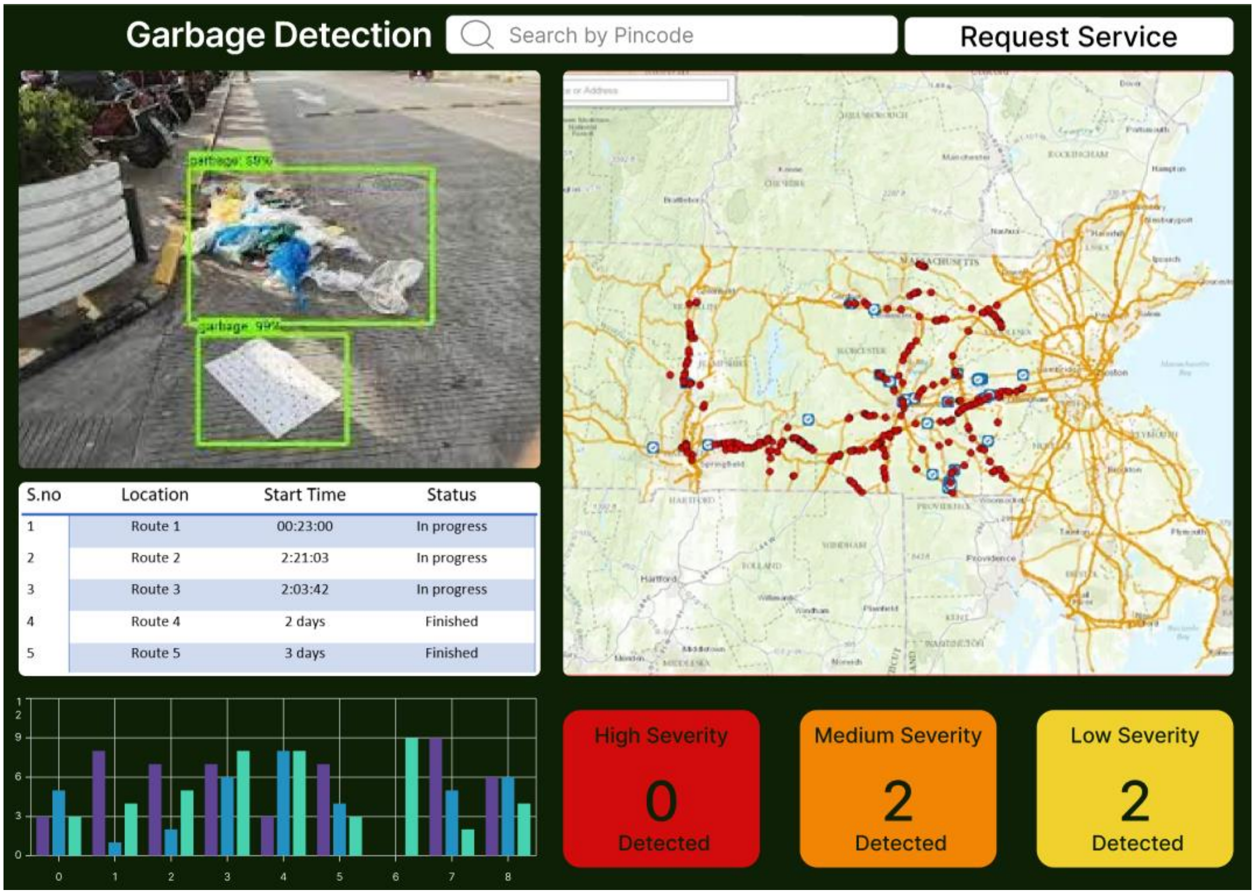


Figure A7

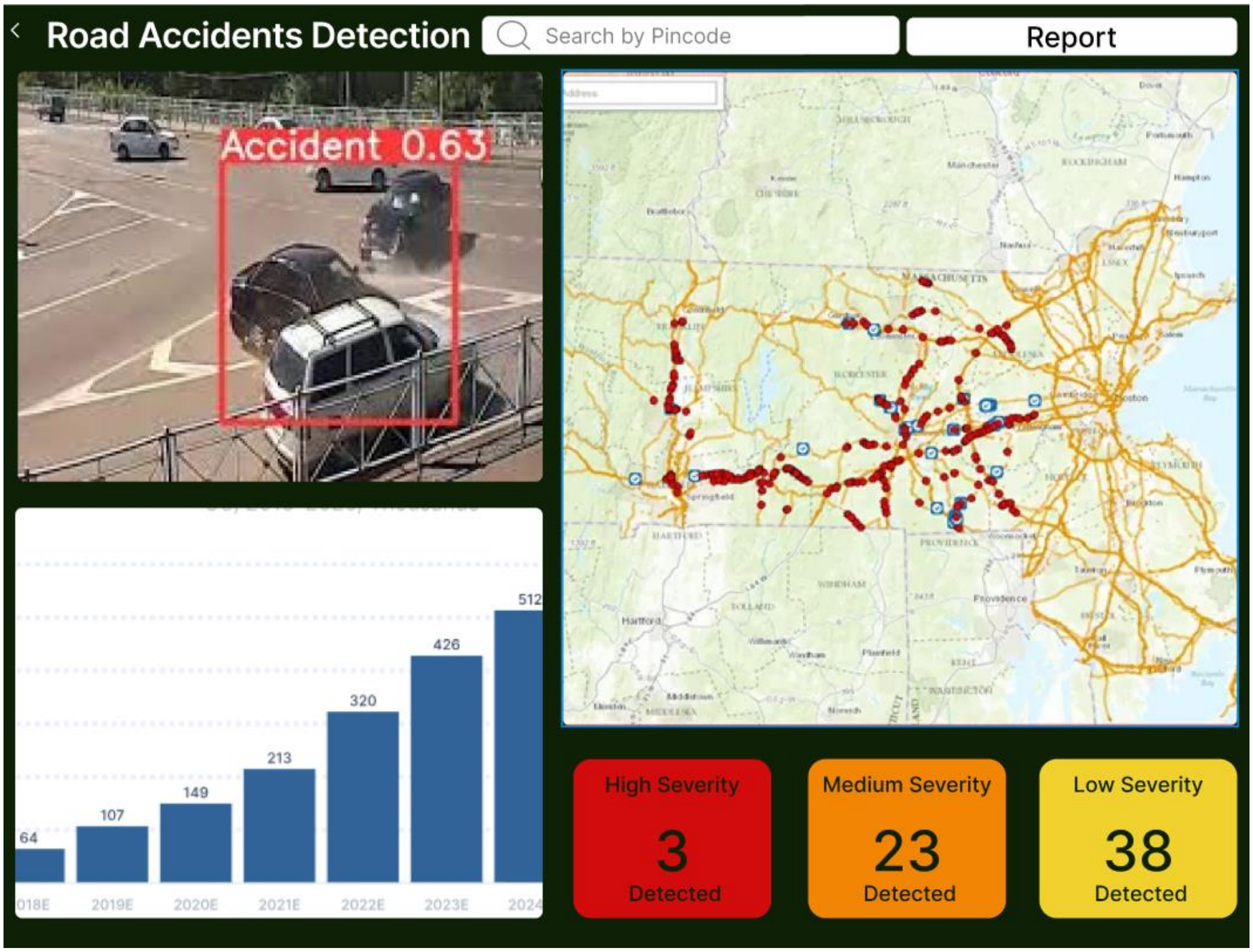
Wildlife Category Specific Dashboard Page

Figure A8

Construction Category Specific Dashboard Page

Figure A9

Road Accident Specific Dashboard Page


Road Accidents Detection
Search by Pincode
Report
Accident 0.63
64
107
149
213
320
426
512
2019E
2020E
2021E
2022E
2023E
High Severity
3
Detected
Medium Severity
23
Detected
Low Severity
38
Detected

Appendix B

Project Data Source and Management Store

The comprehensive data source utilized for all modules in this project is detailed in the following drive link. The data includes training, validation, and test datasets, gathered from multiple sources, along with their corresponding annotations.

Drive Link: https://drive.google.com/drive/u/1/folders/163H5G1Zhbcn__7meGNKW6G5tI-h8kpvQ

Appendix C

Project Program Source Library, Presentation, and Demonstration

This part includes all the essential materials for the road inspection system. It contains the complete source code, model architecture, website and interface, which users to access various dashboards and perform real-time inspections easily. This section also includes detailed documentation and configuration files, as well as presentation slides that highlight the project's goals and methods. And a demo showcasing how the system works. All these materials will be organized and stored in Google Drive for easy access and collaboration.

Google drive link: https://drive.google.com/drive/folders/0AMTYUb8PxY3xUk9PVA