G.A.I.A.

Ground Assessment & Identification Assistant

Presented By:

Anshul Gada (210303105579)

Aman Jaiswal (210303105259)

Juiee Yadav (2103031030004)

Guided By:

Ankita Gandhi

Asst. Prof.

CSE Department

PIET, Parul University

Content

- Introduction
- Project Team : Roles & Responsibilities
- Objective of the Project
- Problem Statement
- Research Paper Summary (in Tabular form)
- Time Line Chart - Schedule
- UML Diagrams
- Flowchart of the system
- Implementation Details
- Demonstration of the Project
- Testing
- Conclusion
- Future Work
- References

Introduction

G.A.I.A (Ground Assessment and Identification Assistant) project is an innovative solution designed to address the escalating issue of road safety by detecting potholes using advanced technologies. By integrating Artificial Intelligence (AI), Machine Learning (ML), and the Internet of Things (IoT), G.A.I.A provides a proactive approach to infrastructure maintenance, ensuring early detection of potholes that can prevent accidents and improve road quality. This project leverages state-of-the-art techniques to develop a reliable and efficient system that can operate in real-time, offering a comprehensive solution to one of the most common yet critical problems faced by modern transportation systems.

Project Team : Roles & Responsibilities

Sr. No	Team Member	Enrollment Number	Role	Responsibility
1	**Anshul Gada**	210303105579	Project Manager / AI Developer	• Oversee project progress and manage team coordination. • Develop and optimize the AI model for pothole detection.
2	**Aman Jaiswal**	210303105259	IoT / System Engineer	• Design and implement the IoT infrastructure. • Ensure seamless communication between hardware and AI systems.
3	**Juiee Yadav**	210303130004	UI / UX Web Developer	• Design a user-friendly interface for G.A.I.A. • Develop the front-end and back-end of the web application. • Integrates AI and IoT components with the web platform.

Objective

- To develop a reliable and efficient system that detects potholes in real-time, enabling prompt repairs and reducing the risk of accidents caused by road hazards.
- To contribute to safer transportation by providing an advanced technological solution that identifies road defects, thereby minimizing the likelihood of vehicle damage and accidents.
- To provide a proactive approach to infrastructure management by leveraging AI, ML, and IoT, facilitating timely maintenance, and extending the lifespan of roads.
- To collect and analyze data on road conditions, helping authorities prioritize repair work based on real-time information and improving overall road quality.
- To design a system that can be easily scaled and adapted to different environments, ensuring widespread applicability and effectiveness in diverse geographic and climatic conditions.

Problem Statement

Potholes are a significant hazard on roads, causing numerous accidents, vehicle damage, and even fatalities. Traditional methods of detecting and repairing potholes are often reactive, inefficient, and costly, leading to delays in maintenance and increased risks for road users. The challenge lies in developing a system that can autonomously detect potholes in real-time, providing accurate and timely information to authorities for quick repairs. The G.A.I.A project addresses this challenge by utilizing AI/ML models and IoT devices to create an automated pothole detection system that can significantly enhance road safety and maintenance efficiency.

Research Paper Summary

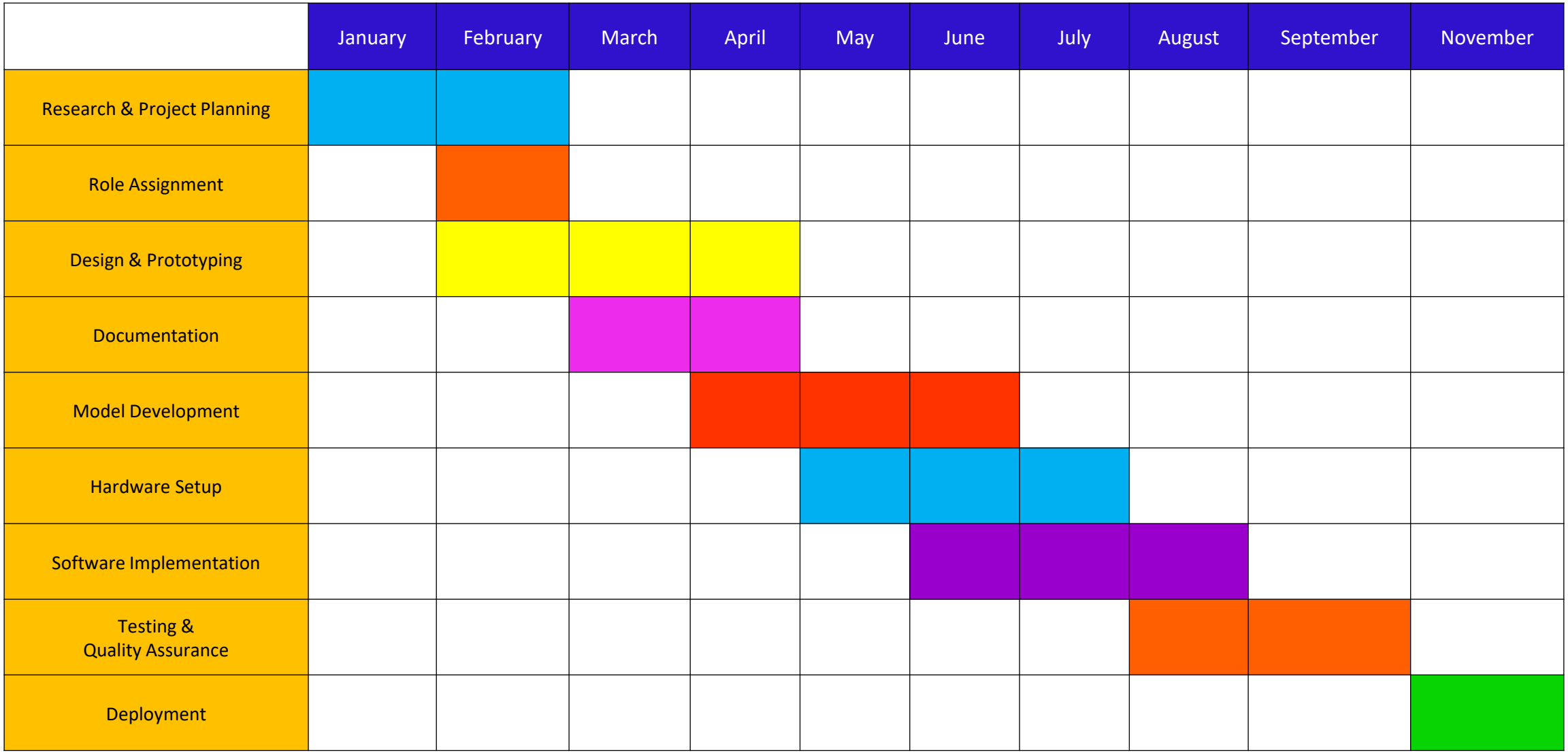
Sr. No	Paper Title	Publisher	Year	Take - away points
1	**Detection and Segmentation of Cement Concrete Pavement Pothole**	Inner Mongolia Agricultural University	**2020**	• Multi-method approach for better detection. Otsu Edge Detection K-Means Watershed used. Small pebbles don't affect results.
2	**AI Enhanced Drones in Pothole Detection**	Xi'an Aeronautical Institute China	**2022**	• Deep learning (SVM CNN RNNs) outperforms edge methods. High memory needed for CNN on high-res images.
3	**Detection of Potholes on Roads using a Drone**	BMS Institute of Technology India	**2021**	• YOLOv3 for real-time pothole detection. 85% accuracy with low false positives/negatives. Real-time updates on open-source map.
4	**Pothole Detection by Image Processing Using UAVs**	Pimpri Chinchwad College of Engineering India	**2020**	• Automatic detection via RGB image segmentation. MATLAB filters and operations used. Five-step process for detection.

Sr. No	Paper Title	Publisher	Year	Take - away points
5	**Detecting Potholes Using Simple Image Processing**	Stellenbosch University South Africa	**2015**	• Single optical camera for pothole detection (2m-20m range). Effective at speeds < 60 km/h.
6	**Drone-based IoT System for Pothole Detection and Volume Calculation**	Liepaja University Latvia	**2019**	• Uses electromagnetic impulses and geo radar. Developed algorithm for object ID and volume calculation.
7	**Terrain Surveillance with Drone and Machine Vision**	Madras Scientific Research Foundation India	**2019**	• Uses SSD Mobilenet Faster RCNN for pothole detection. Integrated with Google Maps for superior navigation.
8	**IoT-based Detection of Bore-Well Unclosed Holes**	Bannari Amman Institute of Technology India	**2021**	• Depth sensing with ultrasonic sensors. Analyzes pothole danger level.

Sr. No	Paper Title	Publisher	Year	Take - away points
9	**Object Detection Using IoT and ML for Road Safety**	D.Y Patil School of Engineering Pune India	**2020**	• Real-time obstacle detection for accident avoidance. Raspberry Pi-based system.
10	**Automated ML Approach for Pothole Detection Using Smartphone Sensor Data**	Zhejiang University China	**2020**	• 90% success in speed breaker detection, 85% in pothole detection. Smartphone sensor-based approach.
11	**Pothole Detection Using Deep Learning: Real-Time and AI-on-the-Edge**	National Centre of Robotics and Automation Pakistan	**2022**	• Tiny-YOLOv4 identified as best for real-time detection. High mAP achieved with YOLOv4 and YOLOv5.
12	**Real-time ML-based Approach for Pothole Detection**	University of South Wales UK	**2021**	• Collected data from multiple Android devices. Pre-processing and training of binary classifier.

Sr. No	Paper Title	Publisher	Year	Take - away points
13	**Review of Automated Pothole Detection Methods**	Defense Agency for Technology and Quality Korea	**2022**	• Vision-based, vibration-based, and 3D reconstruction methods overview.
14	**Smart Pothole Detection Using Deep Learning with Dilated Convolution**	Southern Illinois University USA	**2021**	• Modified VGG16 for high accuracy and mAP. Compared with Faster R-CNN and YOLOv5.
15	**Multi-lane Pothole Detection from Crowdsourced Vehicle Sensor Data**	Carnegie Mellon University USA	**2017**	• Crowdsourced noisy accelerometer data for pothole detection. Road angle data isolated pothole-related accelerations.
16	**Real-Time Pothole Detection Using YOLOv3**	Universiti Teknologi Malaysia Johor	**2020**	• YOLOv3-based system for public transport.

Time Line Chart - Schedule



UML Diagrams

Unified Modeling Language (UML) is a standardized modeling language in software engineering, facilitating visual representations of a system's design and behavior. The following diagrams are commonly used:

1. Flowchart Diagram:

- Illustrates the static structure of a system, showing classes, attributes, methods, and relationships.

2. Class Diagram:

- Illustrates the static structure of a system, showing classes, attributes, methods, and relationships.

3. Sequence Diagram:

- Shows interactions between objects over time, portraying message flow during a specific scenario.

4. Activity Diagram:

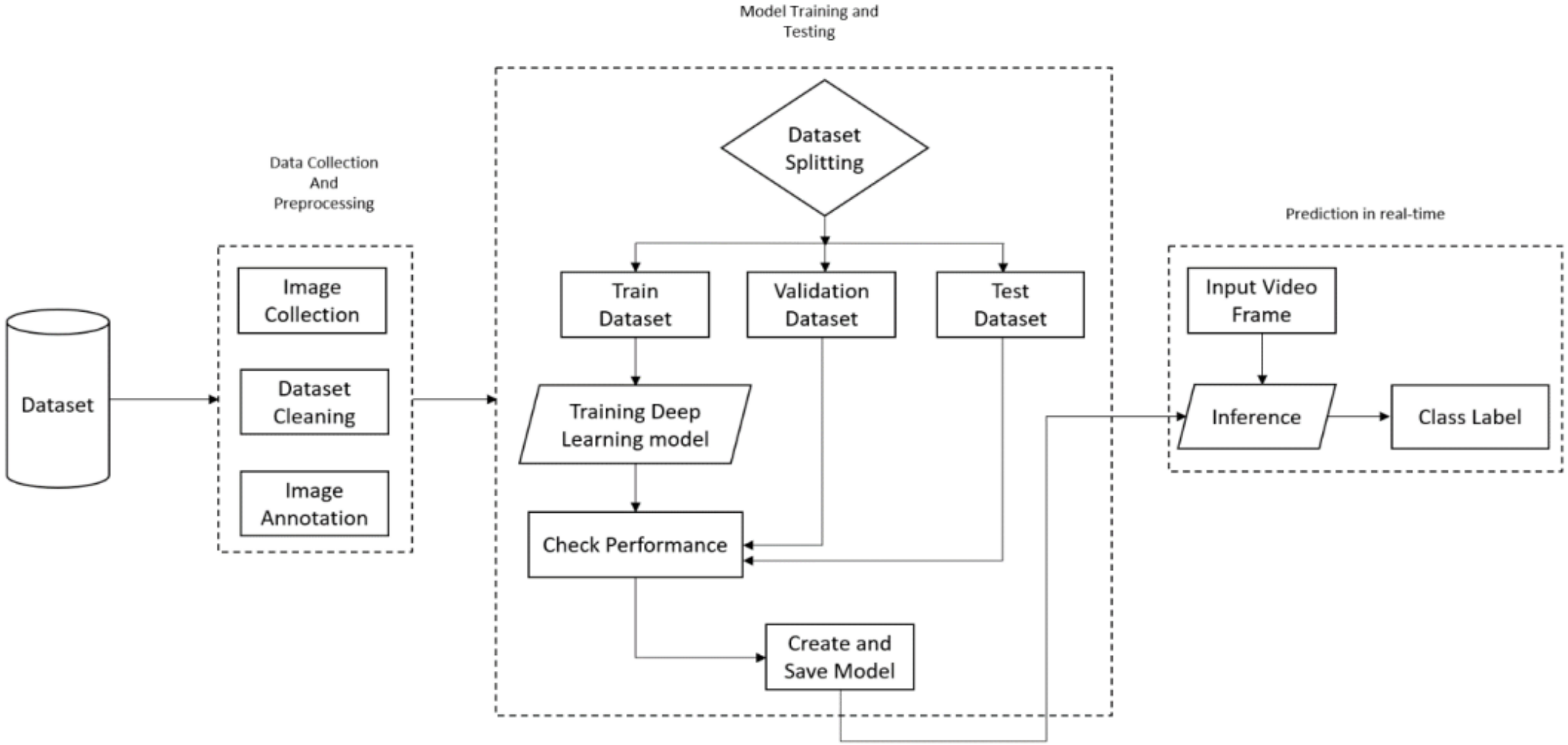
- Represents the flow of activities within a system, depicting actions, decisions, and concurrent activities.

5. Deployment Diagram:

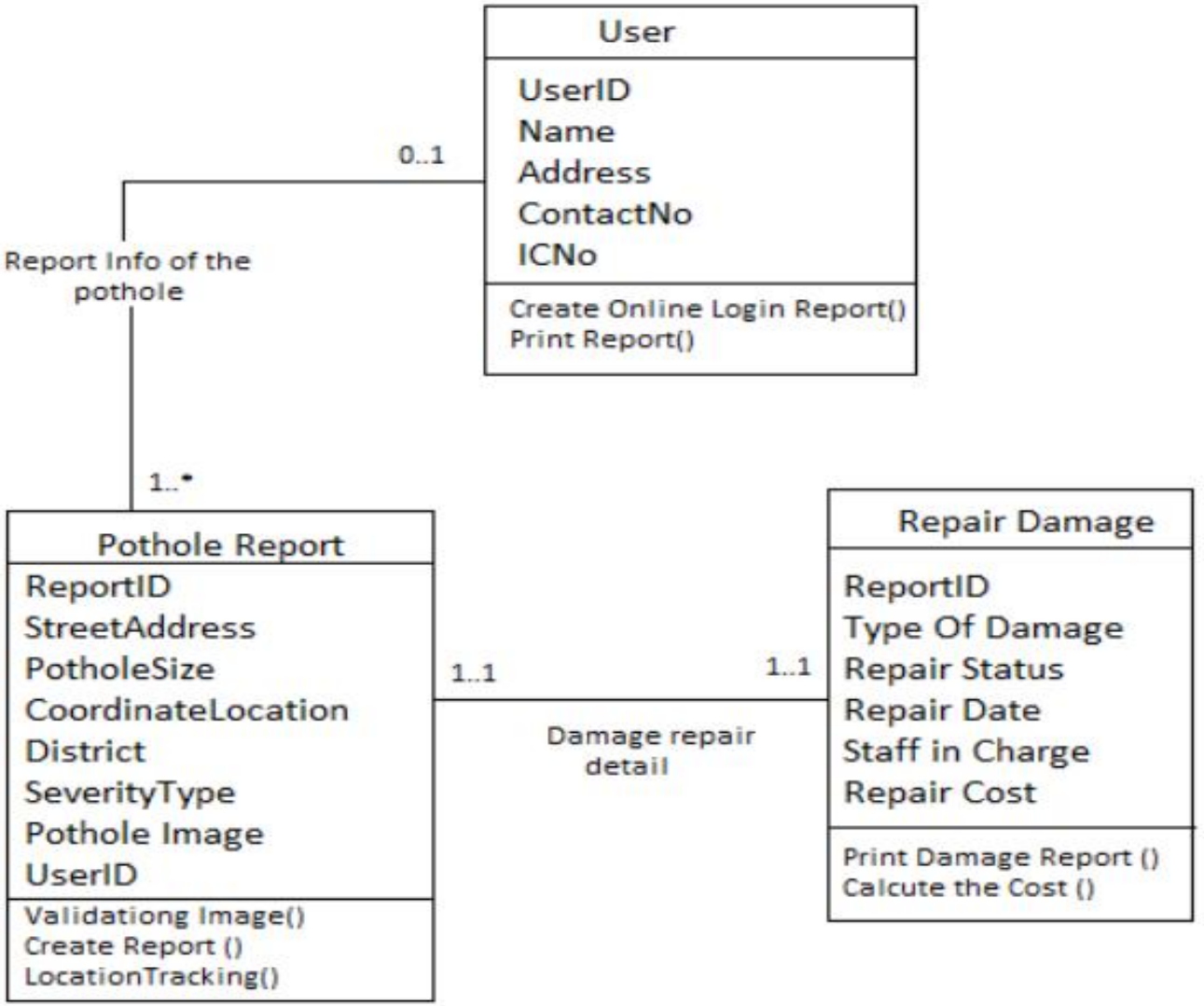
- Illustrates the physical deployment of software components and hardware nodes in a distributed system.

UML diagrams serve as powerful communication tools, aiding in design, documentation, and collaboration throughout the software development process.

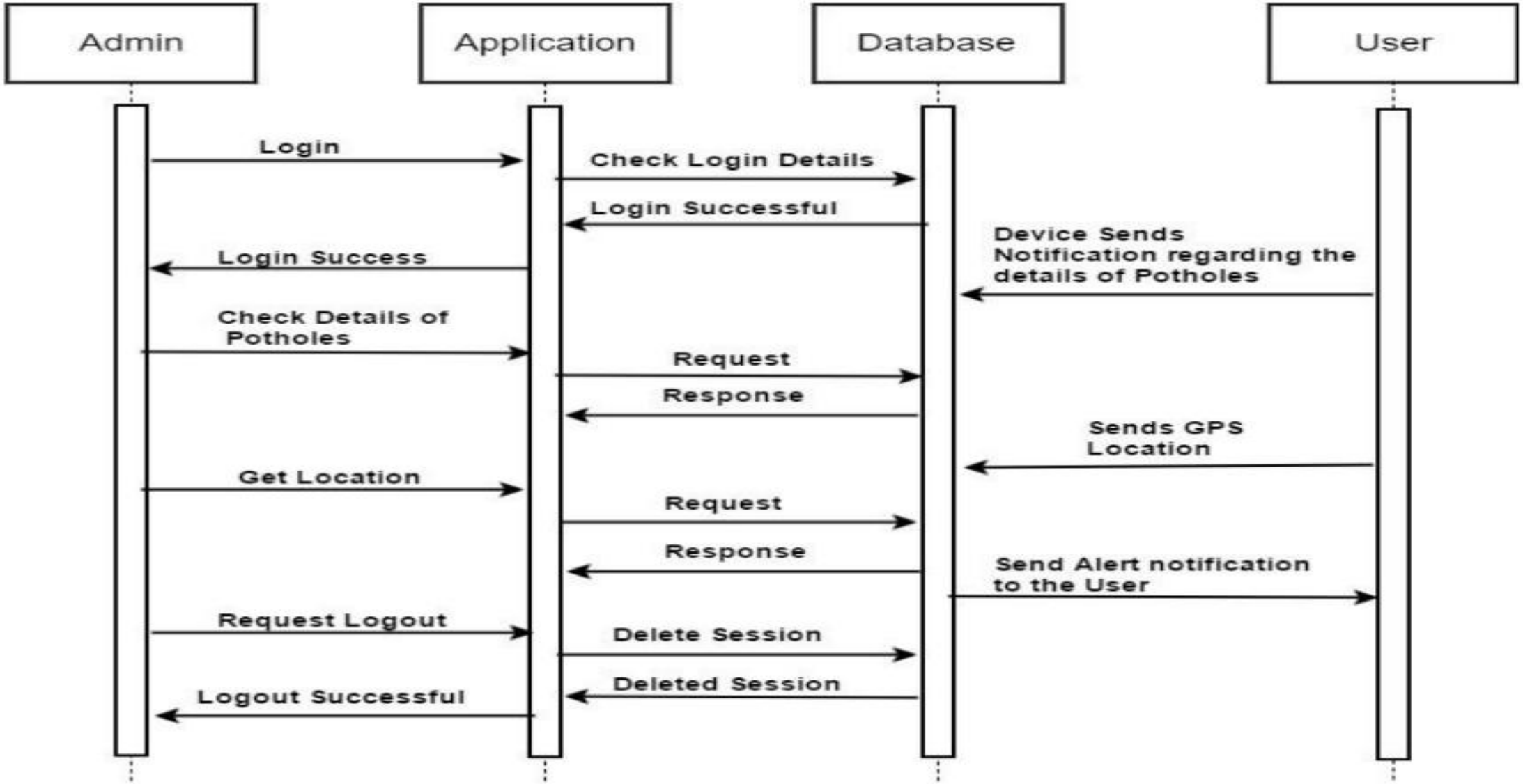
System Flowchart



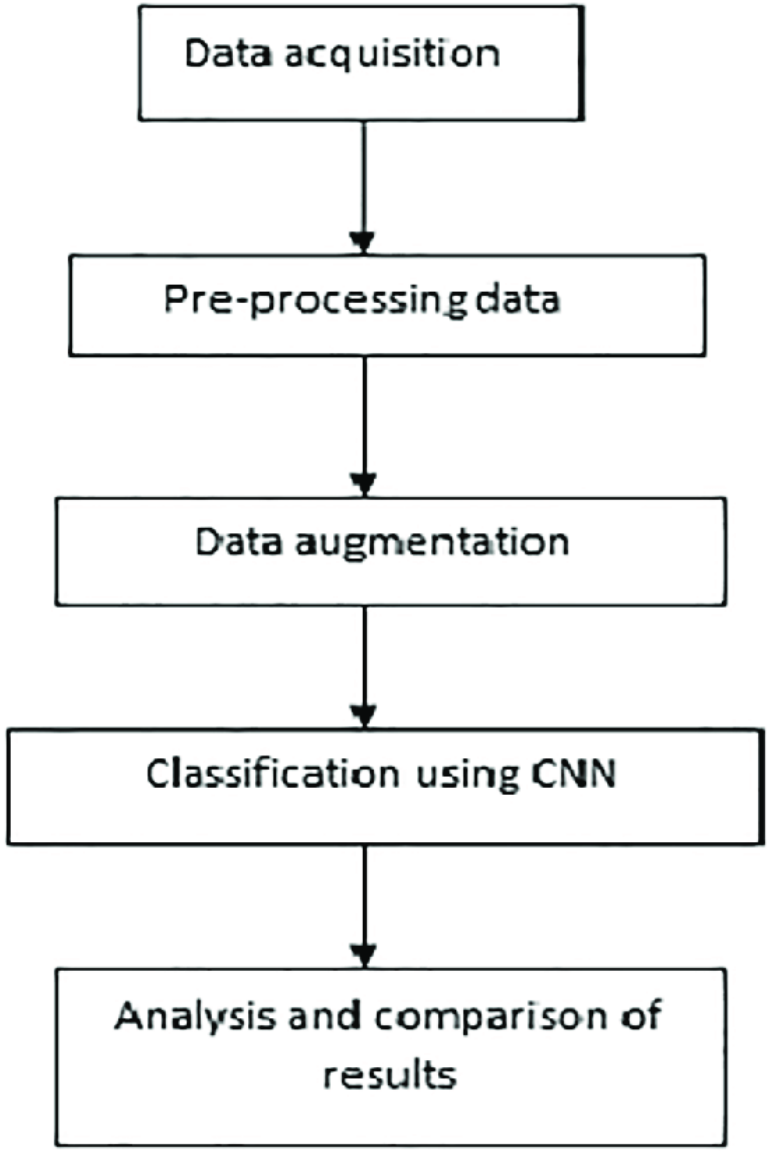
Class Diagram



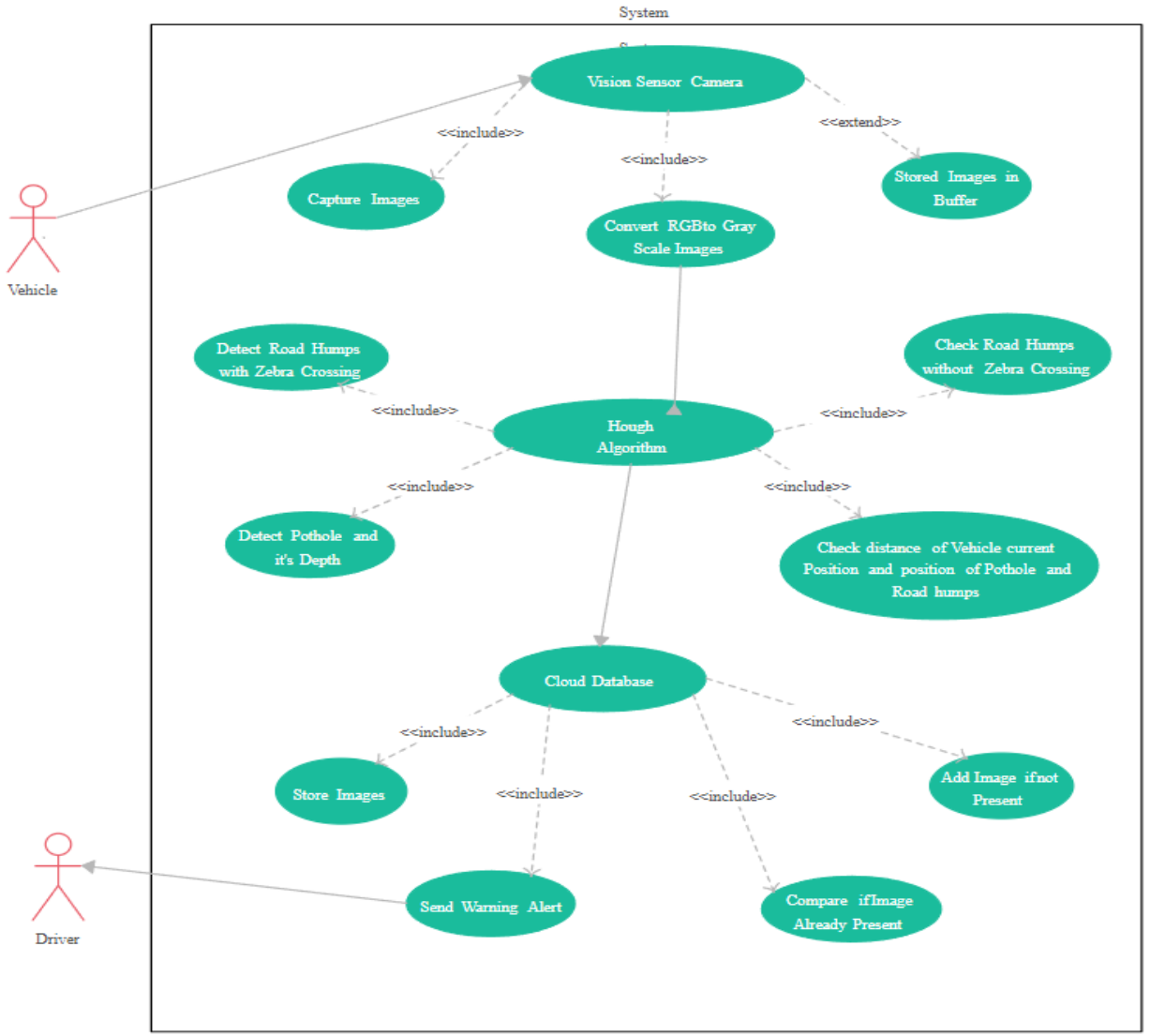
Sequence Diagram



Activity Diagram



Deployment Diagram



Implementation Details

The implementation of G.A.I.A involves several cutting-edge technologies and methodologies. The system is built using YOLOv10, a state-of-the-art object detection model from Ultralytics, trained on a dataset of over 50,000 images of roads with varying conditions. The model is deployed on an NVIDIA Jetson Orin Nano, an IoT device capable of running AI/ML models efficiently in real-time. The entire system operates on port 8000, where the inferencing is performed. The model is capable of image, video, and live inferencing, with detection accuracies ranging from 0.68 to 0.93. The minimum detection accuracy observed is 0.68, while the highest accuracy achieved is 0.93. The training of the model was conducted on Google Colab, leveraging its powerful GPU capabilities. Additionally, an inferencing website has been developed, where users can perform image and URL inferencing, similar to the functionality provided by Roboflow's example web app.

Yolov10 : Key features & Work Flow

Key Features of YOLOv10:

- **Real-Time Performance**: YOLOv10 maintains the YOLO tradition of real-time object detection with high frame rates and low latency, making it suitable for applications requiring fast processing.
- **Improved Accuracy**: Enhancements in architecture and training techniques have improved detection accuracy, especially in challenging conditions such as small or occluded objects.
- **Advanced Backbone**: YOLOv10 utilizes an advanced backbone network for feature extraction, improving the model's ability to detect and classify objects with greater precision.
- **Enhanced Detection Heads**: The model incorporates improved detection heads that better handle objects at various scales and aspect ratios.
- **Efficient Training**: YOLOv10 benefits from optimized training processes, which enhance its learning capability and generalization to new datasets.
- **Smarter Anchor Boxes**: The model uses smarter anchor boxes to better predict bounding boxes around objects, which helps in reducing false positives and improving localization.
- **Scalability**: YOLOv10 is designed to be scalable, with variants that balance between speed and accuracy, allowing you to choose the best model for your specific use case.

Typical Workflow with YOLOv10:

- **Data Collection**: Gather a diverse set of images or videos of potholes from various environments, lighting conditions, and road types to ensure dataset diversity, and organize the data with relevant metadata for easy access and management.
- **Annotation**: Use annotation tools to draw precise bounding boxes around potholes in the images, and perform a review or validation of annotations to ensure accuracy and consistency.
- **Training**: Configure hyperparameters like learning rate, batch size, and number of epochs based on our dataset.
- **Validation**: Adjust model hyperparameters based on performance metrics from the validation set to optimize model performance.
- **Testing**: Assess the model's performance on a test set that was not seen during training to measure its generalization ability, and identify strengths and weaknesses by analyzing metrics such as precision, recall, and F1-score.
- **Deployment**: Deploy the trained model into the G.A.I.A system, ensuring it operates effectively in real-time scenarios, and conduct field tests to verify the system's performance and reliability.
- **Monitoring**: Continuously track the model's accuracy and performance in the deployed system to detect any issues.

Algorithm (Yolov10)

Overview Of Yolov10

- YOLOv10, an advanced version in the YOLO (You Only Look Once) series, is a state-of-the-art object detection algorithm known for its high performance and efficiency.

- YOLOv10 is an extension of the YOLO object detection family, building upon the innovations introduced in previous versions such as YOLOv3, YOLOv4, and YOLOv5. YOLOv10 focuses on enhancing the model's accuracy, speed, and robustness in detecting objects, including small and occluded items, making it well-suited for real-time applications.

Technical Details of Yolov10

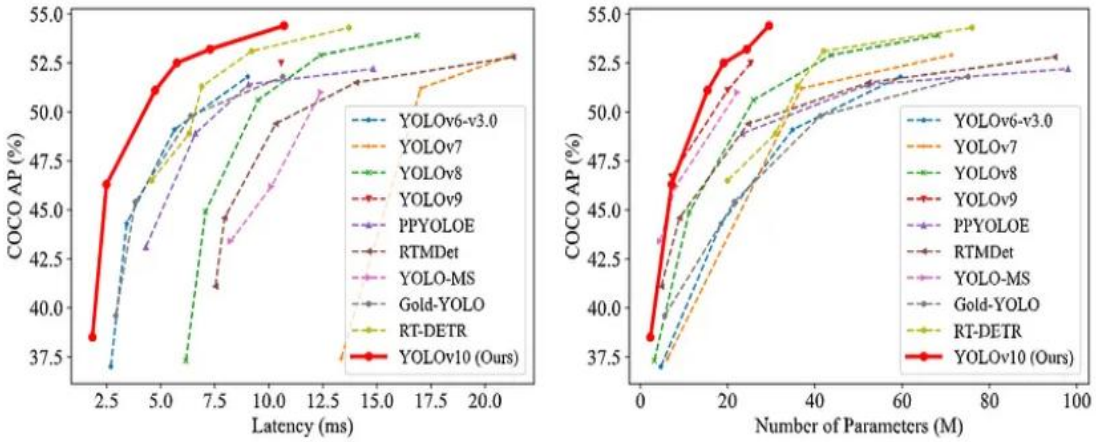
- **Architecture**: YOLOv10 introduces architectural improvements over previous versions, including better backbone and neck structures that enhance feature extraction and object detection.

- **Training Process**: The training process involves using a diverse dataset with labeled objects. YOLOv10 uses advanced augmentation techniques and optimized loss functions to improve model performance.

- **Inference**: During inference, YOLOv10 processes input images through its network to produce predictions. It uses a combination of feature maps and detection heads to identify objects and their locations within the image.

- **Applications**: YOLOv10 is used in a wide range of applications, including autonomous vehicles, surveillance systems, robotics, and any scenario where real-time object detection is critical.

Yolov10 Charts, Comparison, Compatibility and Models



YOLO v10

Model	Input Size	AP^{val}	FLOPs (G)	Latency (ms)
YOLOv10-N	640	38.5	**6.7**	**1.84**
YOLOv10-S	640	46.3	21.6	2.49
YOLOv10-M	640	51.1	59.1	4.74
YOLOv10-B	640	52.5	92.0	5.74
YOLOv10-L	640	53.2	120.3	7.28
YOLOv10-X	640	**54.4**	160.4	10.70

Export Format	Export Support	Exported Model Inference	Notes
TorchScript	✅	✅	Standard PyTorch model format.
ONNX	✅	✅	Widely supported for deployment.
OpenVINO	✅	✅	Optimized for Intel hardware.
TensorRT	✅	✅	Optimized for NVIDIA GPUs.
CoreML	✅	✅	Limited to Apple devices.
TF SavedModel	✅	✅	TensorFlow's standard model format.
TF GraphDef	✅	✅	Legacy TensorFlow format.
TF Lite	✅	✅	Optimized for mobile and embedded.
TF Edge TPU	✅	✅	Specific to Google's Edge TPU devices.
TF.js	✅	✅	JavaScript environment for browser use.
PaddlePaddle	❌	❌	Popular in China; less global support.
NCNN	✅	❌	Layer `torch.topk` not exists or registered

Nvidia Jetson (Edge Inference Device)

NVIDIA Jetson is a series of embedded computing platforms designed for AI and machine learning applications in robotics, autonomous machines, and edge computing.

Nvidia Jetson Nano

- **Processor**: Quad-core ARM Cortex-A57 CPU
- **GPU**: 128-core Maxwell GPU
- **Memory**: 4 GB LPDDR4
- **Storage**: microSD card slot (supports up to 256 GB)
- **Networking**: Gigabit Ethernet
- **I/O Ports**:1x USB 3.0
    - 1x USB 2.0
    - 1x HDMI 2.0
    - 1x MIPI CSI-2 (up to 2x cameras)
    - 1x GPIO header (40-pin)
- **Power Supply**: 5V/4A (via barrel jack or USB-C)
- **Dimensions**: 100 mm x 80 mm
- **Operating System**: Ubuntu-based Linux (JetPack SDK)

Nvidia Jetson Orin Nano

- **Processor**: 6-core ARM Cortex-A78AE CPU (quad-core + dual-core)
- **GPU**: 1024-core NVIDIA Ampere architecture GPU
- **Memory**: 8 GB LPDDR5
- **Storage**: 16 GB eMMC (embedded MultiMediaCard)
- **Networking**: Gigabit Ethernet
- **I/O Ports**:2x USB 3.2 Gen 1
    - 1x USB 2.0
    - 1x HDMI 2.1
    - 1x MIPI CSI-2 (up to 4x cameras)
    - 1x PCIe x4 (Gen 3)
    - 1x GPIO header (40-pin)
- **Power Supply**: 5V/4A (via barrel jack)
- **Dimensions**: 100 mm x 87 mm
- **Operating System**: Ubuntu-based Linux (JetPack SDK)

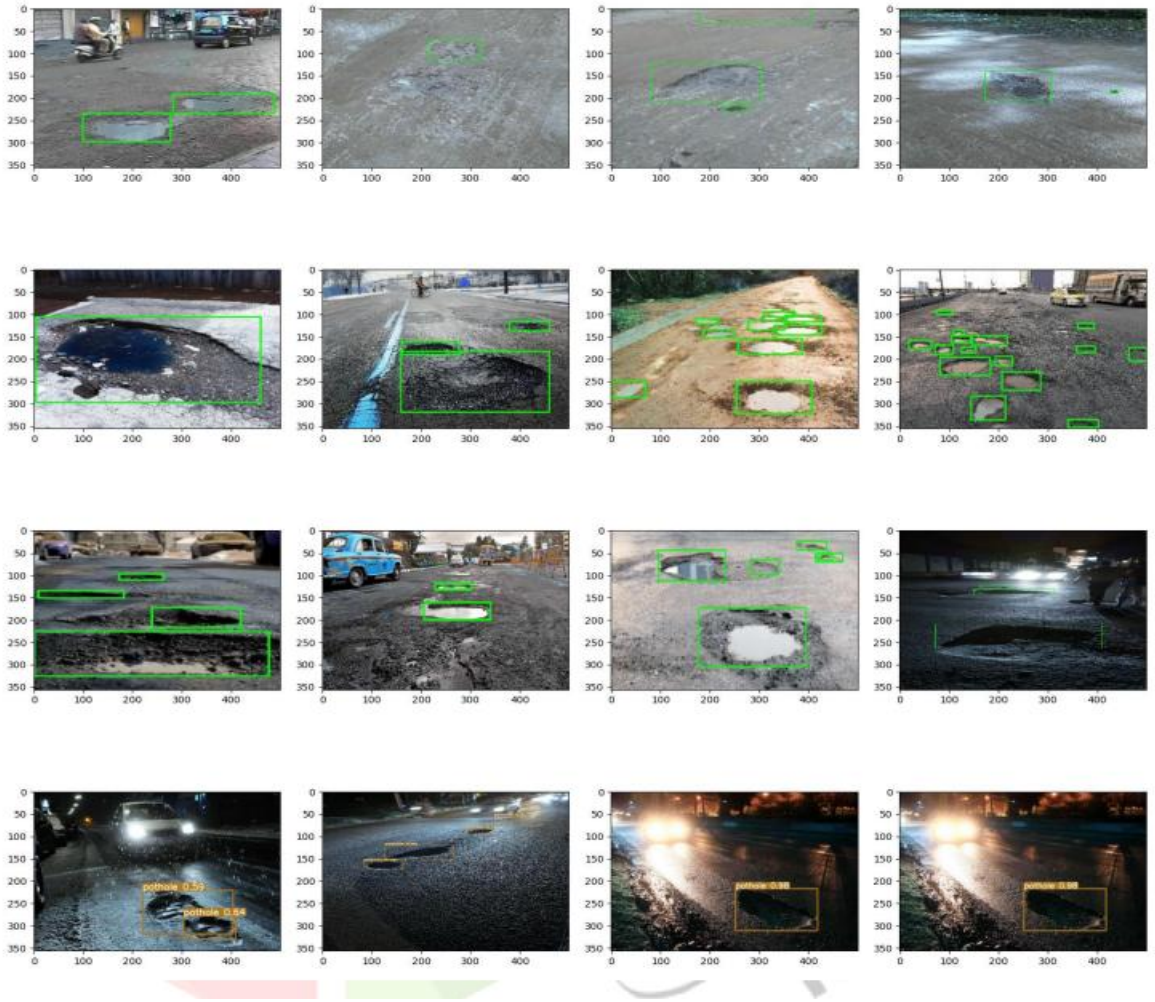
Nvidia Jetson Nano

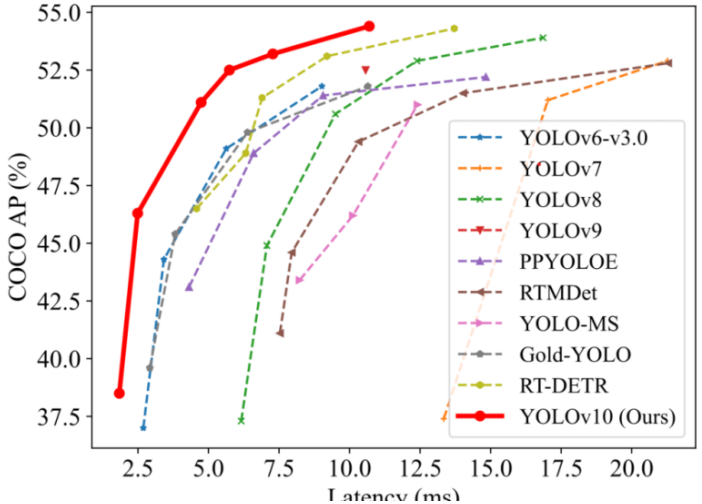
- **Low Processing Power**: The Jetson Nano's limited processing power resulted in slower execution of AI models and increased inference times. This affected the real-time performance of your pothole detection system, leading to delays in processing and reduced responsiveness.
- **Higher Preprocessing Times**: With lower computational capacity, preprocessing tasks such as image resizing, normalization, and data augmentation took longer to complete. This delay contributed to slower overall system performance and hindered the efficiency of the detection pipeline.
- **RAM Overflow Issues**: The Jetson Nano's 4 GB of LPDDR4 memory was insufficient for handling the memory demands of the AI models and data processing tasks. This led to RAM overflow issues, causing crashes or significant slowdowns during model training and inference.

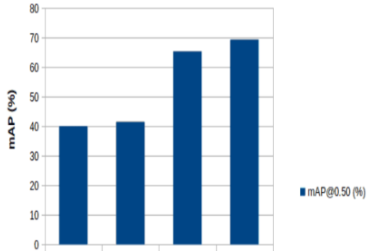
Nvidia Jetson Orin Nano

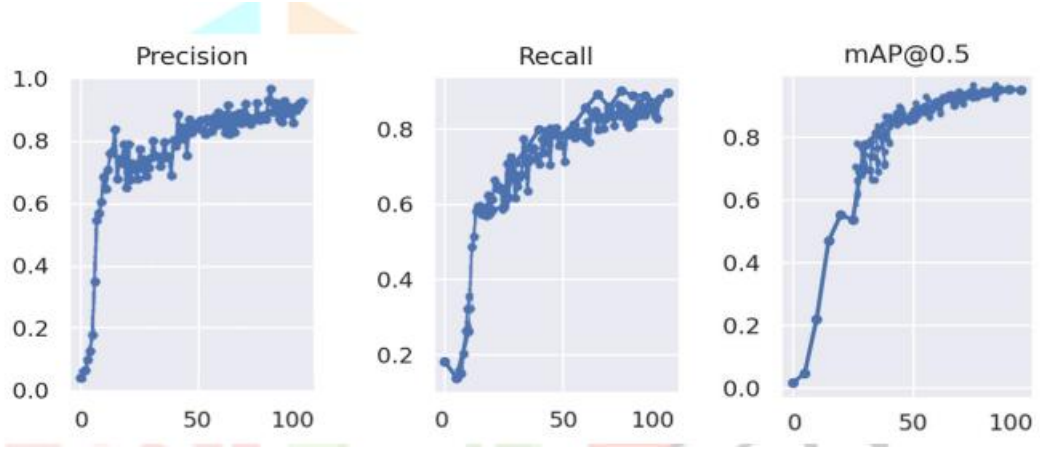
- **Enhanced Processing Power**: The Jetson Orin Nano, with its 6-core ARM Cortex-A78AE CPU and 1024-core Ampere GPU, provided a substantial boost in processing power. This improvement enabled faster execution of AI models and reduced inference times, enhancing the real-time performance of your pothole detection system.
- **Reduced Preprocessing Times**: The increased computational capabilities of the Orin Nano allowed for more efficient preprocessing. Tasks like image resizing and data augmentation were completed more quickly, leading to a smoother and faster overall system performance.
- **Increased RAM**: With 8 GB of LPDDR5 memory, the Orin Nano addressed the RAM overflow issues encountered with the Jetson Nano. This increased memory capacity allowed for better handling of large datasets and more complex models, reducing crashes and improving stability during both training and inference.

Examples of implemented work









Testing

- The testing phase of G.A.I.A involved rigorous evaluation under various conditions to ensure the system's reliability and accuracy. Testing was conducted on different road types, including highways, city streets, and rural roads, under varying weather conditions such as bright sunlight, rain, and low-light scenarios. The model was tested for its ability to detect potholes of different sizes and shapes, ensuring that even small or irregularly shaped potholes could be identified. The testing also included stress tests, where the system was fed large volumes of data to assess its processing speed and efficiency.
- The test cases for G.A.I.A were designed to cover a wide range of scenarios to validate the system's performance. Each test case included specific inputs, expected outputs, and actual results. For instance, one test case involved detecting a large pothole on a rural road in bright sunlight, where the expected output was a bounding box around the pothole with an accuracy label. Another test case focused on detecting multiple small potholes on a city street at night, where the system had to accurately identify each pothole despite low visibility. The results of these test cases showed that the system consistently performed well, with accuracy rates within the expected range.

Future Work

Future work on the G.A.I.A project will focus on expanding its capabilities and improving its performance. Planned developments include integrating live and video inferencing into the inferencing website, allowing users to monitor road conditions in real-time remotely. Additionally, there will be efforts to enhance the model's accuracy by expanding the dataset to include more diverse road conditions and using more advanced training techniques. The team also aims to explore the possibility of integrating the system with autonomous vehicles, enabling on-the-fly detection and reporting of potholes as vehicles navigate the roads.

References

- Object Detection with TensorFlow Lite Model Maker
  - (https://www.tensorflow.org/lite/models/modify/model_maker/object_detection)
- YOLOv10 Object detection Model
  - (https://github.com/ultralytics/ultralytics/blob/main/docs/en/models/yolov10.md)
- Evaluate & Monitor ML Models
  - (https://github.com/evidentlyai/evidently/)
- Drone Flight Path Mapping
  - (https://robots.net/tech/how-to-program-drone-flight-path/)

Thank You