# AI-DRIVEN DETECTION OF ROAD POTHOLES FOR ENHANCED INFRASTRUCTURE SAFETY

CHING KAI XIN

[illegible]

Dr. Nurulaqilla Khamis
Pensyarah Kanan
Fakulti Kejuruteraan Elektrik
Universiti Teknologi Malaysia
81310 Johor Bahru, Johor, Malaysia
Email: nurulaqilla@utm.my

UNIVERSITI TEKNOLOGI MALAYSIA

PSZ 19:16 (Pind. 1/23)

**UNIVERSITI TEKNOLOGI MALAYSIA**
**DECLARATION OF UNDERGRADUATE REPORT**

| | | | | | |
|---|---|---|---|---|---|
| Author's full name | : | CHING KAI XIN | | | |
| Student's Matric No. | : | A21EE0036 | Academic Session | : | 2024/2025-1 |
| Date of Birth | : | 11/12/2002 | UTM Email | : | chingxin@graduate.utm.my |
| Report Title | : | AI-DRIVEN DETECTION OF ROAD POTHOLES FOR ENHANCED INFRASTRUCTURE SAFETY | | | |

I declare that this thesis is classified as:

| | | |
|---|---|---|
| ☒ | **OPEN ACCESS** | I agree that my report to be published as a hard copy or made available through online open access. |
| ☐ | **RESTRICTED** | Contains restricted information as specified by the organization/institution where research was done. *(The library will block access for up to three (3) years)* |
| ☐ | **CONFIDENTIAL** | Contains confidential information as specified in the Official Secret Act 1972) |

*(If none of the options are selected, the first option will be chosen by default)*

I acknowledged the intellectual property in the report belongs to Universiti Teknologi Malaysia, and I agree to allow this to be placed in the library under the following terms :

1. This is the property of Universiti Teknologi Malaysia
2. The Library of Universiti Teknologi Malaysia has the right to make copies for the purpose of only.
3. The Library of Universiti Teknologi Malaysia is allowed to make copies of this report for academic exchange.

Signature of Student:

Signature : [illegible]

CHING KAI XIN
Date : 16/1/2025

Approved by Supervisor(s)

Signature of Supervisor I: [illegible]

Full Name of Supervisor I
NURUL AQILLA BINTI KHAMIS

Date : 23/1/25

NOTES : If the thesis is CONFIDENTIAL or RESTRICTED, please attach with the letter from the organization with period and reasons for confidentiality or restriction

“I hereby declare that I have read this report and in my
opinion this report is sufficient in term of scope and quality for the
award of the degree of Bachelor of Engineering (Electrical-Mechatronics) with
Honours”

| | | |
|---|---|---|
| Signature | : | [illegible] |
| Name of Supervisor I | : | NURULAQILLA BINTI KHAMIS |
| Date | : | 20 JANUARY 2025 |

# AI-DRIVEN DETECTION OF ROAD POTHOLES FOR ENHANCED INFRASTRUCTURE SAFETY

CHING KAI XIN

A report submitted in partial fulfilment of the
requirements for the award of the degree of
Bachelor of Engineering (Electrical-Mechatronics) with Honours

Faculty of Electrical Engineering
Universiti Teknologi Malaysia

JANUARY 2025

## DECLARATION

I declare that this report entitled "AI-DRIVEN DETECTION OF ROAD POTHOLES FOR ENHANCED INFRASTRUCTURE SAFETY" is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.

| | | |
|---|---|---|
| Signature | : | [illegible] |
| Name | : | CHING KAI XIN |
| Date | : | 20 JANUARY 2025 |

## ACKNOWLEDGEMENT

First and foremost, I would like to express my deepest gratitude to my supervisor Dr. Nurulaqilla binti Khamis for her unwavering guidance and constant encouragement throughout the course of Final Year Project Part 1. Her expertise and dedication have been invaluable in shaping this work and ensuring its success.

I extend my heartfelt appreciation to my family and friends for their unwavering support and patience during this journey. Their belief in me has been a source of constant motivation, especially during the most challenging moments.

I am sincerely thankful to my peers and colleagues for their constructive discussions and collaboration. Their support has enriched my academic experience and made this journey truly rewarding.

Finally, I would like to express my gratitude to Universiti Teknologi Malaysia (UTM) for fostering an excellent academic environment and providing state-of-the-art facilities. The opportunities I have received at UTM have been instrumental in my personal and professional growth. I am proud to have been part of this esteemed institution.

## ABSTRACT

This study presents the design and implementation of an artificial intelligence (AI)-driven pothole detection system aimed at enhancing road safety and optimizing infrastructure maintenance. With road potholes posing significant risks to public safety and financial burdens, this research proposes a lightweight AI model deployable on resource-constrained edge devices. By utilizing the YOLO framework, the study compares the performance of YOLOv5n and YOLOv8n in terms of precision, recall, computational efficiency and real-world applicability. The research methodology encompasses data collection, cleaning, labeling, model development and evaluation. A dataset comprising of 959 images was collected and used to train and validate the models. Results indicate that while YOLOv8n offers marginal improvements in recall and detection accuracy, YOLOv5n demonstrates superior computational efficiency, faster inference speed and higher precision. As a result, this makes YOLOv5n as the preferred choice for deployment. Key innovations include model optimization for real-time performance and the integration of the detection system with edge devices. This work underscores the importance of balancing detection accuracy and computational demands, particularly in real-time applications. By deploying this system, the study contributes to scalable and cost-effective road monitoring solutions, paving the way for improved public safety and infrastructure sustainability.

# ABSTRAK

Kajian ini mempersembahkan reka bentuk dan pelaksanaan sistem pengesanan lubang jalan yang didorong oleh kecerdasan buatan (AI) yang bertujuan untuk meningkatkan keselamatan jalan raya dan mengoptimumkan penyelenggaraan infrastruktur. Disebabkan lubang jalan yang menimbulkan risiko besar kepada keselamatan awam dan beban kewangan, kajian ini mencadangkan model AI ringan yang boleh diterapkan pada peranti tepi yang terhad sumber. Menggunakan rangka kerja YOLO, kajian ini membandingkan prestasi YOLOv5n dan YOLOv8n dari segi ketepatan (precision), kebolehcapaian (recall), kecekapan pengkomputeran, dan kebolehgunaan dalam dunia sebenar. Metodologi kajian merangkumi pengumpulan data, pembersihan, pelabelan, pembangunan model, dan penilaian. Dataset yang terdiri daripada 959 imej telah dikumpulkan dan digunakan untuk melatih serta mengesahkan model. Hasil kajian menunjukkan bahawa walaupun YOLOv8n menawarkan sedikit peningkatan dalam kebolehcapaian dan ketepatan pengesanan, YOLOv5n menunjukkan kecekapan pengkomputeran yang lebih unggul, kelajuan inferens yang lebih pantas, dan ketepatan yang lebih tinggi, menjadikannya pilihan utama untuk pelaksanaan. Inovasi utama termasuk pengoptimuman model untuk prestasi masa nyata dan integrasi sistem pengesanan dengan peranti tepi. Kajian ini menekankan kepentingan untuk mengimbangi ketepatan pengesanan dan keperluan pengkomputeran, terutamanya dalam aplikasi masa nyata. Dengan pelaksanaan sistem ini, kajian ini menyumbang kepada penyelesaian pemantauan jalan yang berskala dan kos efektif, membuka jalan untuk meningkatkan keselamatan awam dan kelestarian infrastruktur.

**TABLE OF CONTENTS**

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | | |
|---|---|---|
| AI | - | Artificial Intelligence |
| SSD | - | Single Shot Detector |
| R-CNN | - | Region-Based Convolutional Neural Network |
| YOLO | - | You Only Look Once |
| CNN | - | Convolutional Neural Network |
| FPS | - | Frame Per Second |
| ONNX | - | Open Neural Network Exchange |
| RTMP | - | Real-Time Messaging Protocol |
| IP | - | Internet Protocol |
| GPS | - | Global Positioning System |
| IR | - | Infrared |
| LiDAR | - | Light Detection and Ranging |
| CVAT | - | Computer Vision Annotation Tool |
| mAP | - | mean Average Precision |
| FYP | - | Final Year Project |
| UTM | - | Universiti Teknologi Malaysia |

# CHAPTER 1

# INTRODUCTION

## 1.1 Problem Background

Road conditions play a crucial role in ensuring traffic safety, reducing vehicle maintenance costs and preserving infrastructure longevity. In Malaysia, the formation of road potholes is exacerbrated by the country's tropical climate which is characterised by alternating sunny and rainy conditions. The formation of a road pothole can be referred in Figure 1.1. Cracks in road surface often begin as a minor issue caused by heat and vehicles stress. However, during heavy rainfall, water seeps into these cracks and accumulates beneath the road base. When temperature drops, the water accumulated will freeze and expand, causing pavement to lift and bulge. As temperature rise, the trapped water evaporated and leaves gaps beneath the pavement. Over time, the repeated cycle of water penetration, expansion, evaporation and traffic stress weakens the pavement, causing the cracks to grow larger and eventually form potholes. If road potholes were not addressed promptly, they can significantly affect drivers and riders which pose serious safety and financial risks.

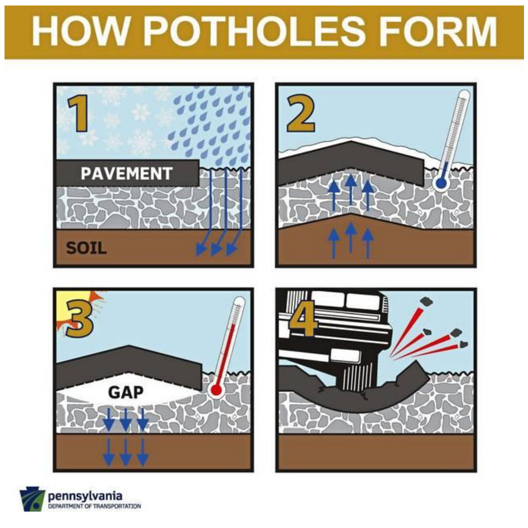


Figure 1.1 Formation of a Road Pothole

Road potholes pose significant risks to drivers and riders which often causing sudden jolts and loss of vehicle control. This is particularly dangerous for motorcyclists and cyclists who are more vulnerable to accidents due to their lack of structural protection. For drivers, the sudden appearance of a pothole can lead to abrupt braking or swerving thus increasing the likelihood of collisions with other vehicles. The safety implications are severe as highlighted by data from the Malaysia Road Accident Management System and Works Ministry. It is reported that potholes caused 181 road accidents, including 23 fatalities between 2022 and July 2024 [1]. Beyond safety concerns, potholes also impose significant financial burdens on road users. Vehicles that hit potholes at high speeds often suffer from tire blowouts, damaged suspension systems and misaligned wheels, leading to costly repairs. Persistent exposure to poor road conditions accelerates wear and tear on vehicles will further escalating maintenance expenses over time. These combined risks underscore the urgent need for improved monitoring and timely road repairs to safeguard both lives and finances while enhancing overall road safety.

Currently, pothole detection and management systems rely heavily on manual reporting processes. Applications like MyJalan [2] in Figure 1.2 and Waze [3] in Figure 1.3, enable users to report potholes by uploading photos and providing the coordinates of the pothole location. Once reported, the relevant authorities will analyze the submissions and dispatch maintenance teams to repair the potholes. The time taken to process these reports exacerbates the risks for road users as unrepaired potholes remain a hazard. This also increases the chances of accidents for drivers and riders who may unexpectedly encounter these road anomalies. Furthermore, manual systems often fail to offer comprehensive and real-time monitoring that may cause many affected areas unreported or unattended. Without timely maintenance, the problem will become worsen which leads to further damage to roads and greater risks for the public.

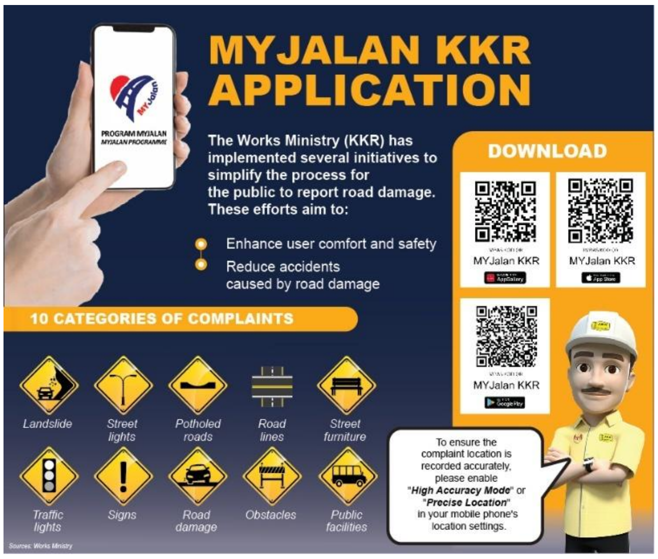

MYJALAN KKR APPLICATION
PROGRAM MYJALAN MYJALAN PROGRAMME
The Works Ministry (KKR) has implemented several initiatives to simplify the process for the public to report road damage. These efforts aim to:
Enhance user comfort and safety
Reduce accidents caused by road damage
DOWNLOAD
MYJalan KKR
MYJalan KKR
MYJalan KKR
10 CATEGORIES OF COMPLAINTS
Landslide
Street lights
Potholed roads
Road lines
Street furniture
Traffic lights
Signs
Road damage
Obstacles
Public facilities
To ensure the complaint location is recorded accurately, please enable "High Accuracy Mode" or "Precise Location" in your mobile phone's location settings.
Sources: Works Ministry

Figure 1.2 MyJalan KKR Application

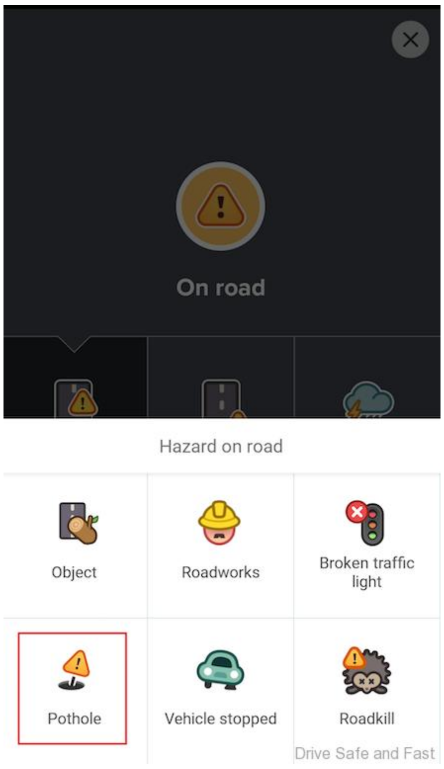


Figure 1.3 Pothole Report using Waze Application

## 1.2 Problem Statement

Despite advancements in technology, current road maintenance systems still face challenges in achieving automation and scalability for pothole detection. Research shows that while AI and computer vision offer promising solutions, many existing models are limited by high computational demands. This makes them unsuitable for deployment on edge devices which are resource-constrained such as Raspberry Pi or Nvidia Jetson. As a result, real-time detection of hazardous road conditions is difficult to implement which leads to delays in corrective measures and reduced road safety.

This project aims to design and implement a lightweight AI model for real-time pothole detection on edge devices. The solution will use advanced computer vision techniques to detect potholes accurately and provide their precise coordinates. Results will be displayed directly on laptop to ensure efficient and scalable monitoring that improves road maintenance practices.

## 1.3 Research Objectives

The ultimate aim of this project is to design and implement a lightweight AI model for real-time pothole detection on edge devices. The objectives in this project are:

(a) To collect images or videos showing different types and sizes of potholes.

(b) To train a YOLO model which recognise pothole accurately by using the labelled data.

(c) To deploy trained model on both camera and edge device for real-time potholes detection.

## 1.4 Scope of Research

The scope of this research focuses on the development of an artificial intelligence system for real-time pothole detection. In the initial phase of this study (FYP Part 1), a detailed comparison between two of the most widely used object detection models (YOLOv5 and YOLOv8) will be conducted. This comparison will evaluate their performance in terms of detection accuracy, inference speed, computational efficiency and overall suitability for real-world applications. By analyzing the strengths and weaknesses of both models, the study aims to identify the most appropriate framework for use in constrained environments such as edge devices.

# CHAPTER 2

# LITERATURE REVIEW

## 2.1 Introduction

The topic of object detection has garnered significant scholarly attention in recent years across various disciplines. Object detection is a computer technology that integrates computer vision and image processing to localize and classify objects in images and videos. This technology has been widely applied in areas such as intelligent healthcare monitoring, autonomous driving, smart video surveillance, anomaly detection and robot vision. The proposed project is mainly focused on pothole detection in real-time which is critical for road safety and infrastructure maintainnace. To address these challenges, selecting an appropriate framework to train and deploy a reliable model is essential. Many frameworks are available for training, including SSD, Fast R-CNN and YOLO. Each framework offers unique advantages based on its application. The theoretical framework employed in this literature review is rooted in CNN which provides a foundation for understanding object detection within computer vision. This literature review explores the evolution of CNN-based frameworks while comparing their performance and highlighting their suitability for real-time pothole detection by addressing existing gaps in the field.

## 2.2 Convolutional Neural Network (CNN)

A CNN is a deep learning architecture widely used in computer vision tasks including image classification, object detection and image segmentation. As shown in Figure 2.1 a typical CNN consists of three main components: convolutional layers, fully connected layers and an output layer. The convolutional layers which may consist of multiple stages are responsible for extracting features from input images or video. These layers enable neural network to detect simple patterns like edges and textures in the early stages and more complex features in deeper layers. After that, the fully connected layers at the end of the network integrate with these features and make predictions based on the learned patterns. Finally, the output layer generates the final results of the network such as classifying objects or predicting bounding boxes in object detection tasks.

CNN architectures have evolved significantly with innovations such as ResNet's residual connections which help address vanishing gradient problems and separable convolutions in MobileNet which improve computational efficiency. These advancements have led to the development of frameworks like YOLO, Fast R-CNN and SSD. Each of these frameworks addresses key challenges in object detection including real-time performance, accuracy and computational efficiency.

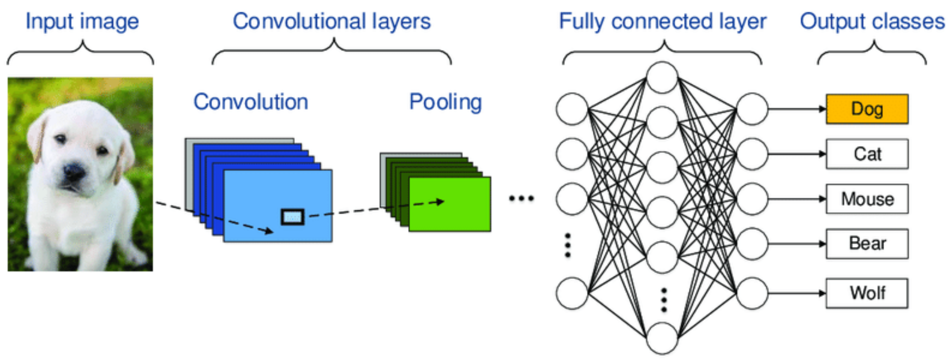


Figure 2.1 A Convolutional Neural Network (CNN) Architecture

## 2.3 Review on YOLO Framework

After having an extensive review of multiple research journals, it is evident that YOLO has become the most preferred framework among researchers for training AI models in pothole detection. One key reason for its popularity is its fast-processing speed [8]. YOLO is a one-stage object detection algorithm that performs detection in a single neural network pass that divides the image into a grid and directly predicting bounding boxes and class probabilities [4]. This streamlined design reduces computational overhead which makes YOLO ideal for real-time applications like pothole detection [5]. In contrast, the R-CNN family, including Fast R-CNN [6] and Mask R-CNN [7], employs a two-stage object detection approach. Initially, they create region proposals, followed by the classification of every suggested region [4]. This method is effective but requires more computational resources, hence resulting in longer processing times. The above-mentioned theories can be proven by Table 2.1 extracted from [5], showing that YOLO has the highest 45 FPS outpass Fast R-CNN with 0.5 FPS. Higher FPS is needed in pothole detection to capture, process and analyze frames at a fast rate to provide real-time feedback of the pothole detected without noticeable delay.

Table 2.1 Real-Time Systems on PASCAL VOC 2007 [5]

| Detector | FPS |
|---|---|
| YOLO | 45 |
| Fast R-CNN | 0.5 |

Moreover, YOLO has the capability of detecting potholes with varying shapes and sizes through a technique called multi-scale prediction [8]. This technique analyse objects at multiple scales, which allows it to detect both small and large potholes. Besides, YOLO also uses an anchor boxes mechanism to train the model with the varying transformation of the image, including scaling, rotating or flipping. This improves its ability to recognize potholes in diverse conditions. These two methods were important in handling potholes of varying shapes, especially in terms of size, shape and orientation. Additionally, after optimizing the YOLO AI model using Tensorflow Lite or Onnx file format, it becomes a relatively smaller model size which facilitates deployment on edge devices such as drones [9-10] and embedded systems (Raspberry Pi & Nvidia Jetson) [11] to ensure portability without compromising performance.

Advanced versions of YOLO, such as YOLOv5 and YOLOv8, introduce powerful features like anchor-free detection [12] and mosaic augmentation [13-14] that significantly enhance detection speed and accuracy [15]. Mosaic augmentation helps improve training by combining several images into one larger image, as shown in Figure 2.2. This technique makes the training data more varied and diverse, which helps the model learn better and generalize more effectively. On the other hand, Anchor-free detection, as referred to in Figure 2.3, eliminates the need for predefined anchor boxes, and instead, the model detects objects based on key points or pixels, leading to greater flexibility and accuracy. Such developments make YOLO not only efficient in pothole detection but versatile in recognizing other road hazards, such as cracks and debris, to ensure road safety and infrastructure maintenance.

In summary, this review emphasizes YOLO as the preferred framework for pothole detection due to its efficient one-stage object detection which offers fast processing speeds ideal for real-time applications. YOLO's multi-scale prediction and anchor box mechanisms allow it to detect potholes of various shapes and sizes. On the other hand, YOLO models that are optimized with TensorRT [16] or ONNX [17] are compact and suitable for deployment on edge devices. Advanced versions like YOLOv5 and YOLOv8 enhance speed and accuracy with features like mosaic augmentation and anchor-free detection, making YOLO versatile for detecting road hazards.

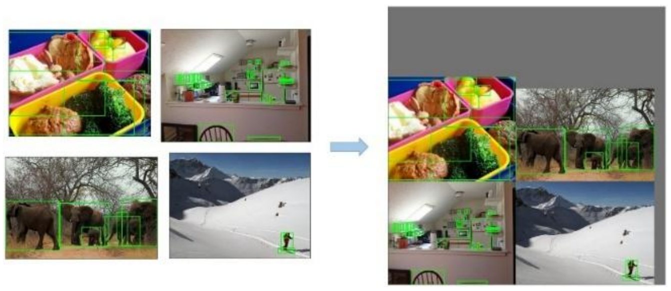


Figure 2.2 Mosaic Augmentation

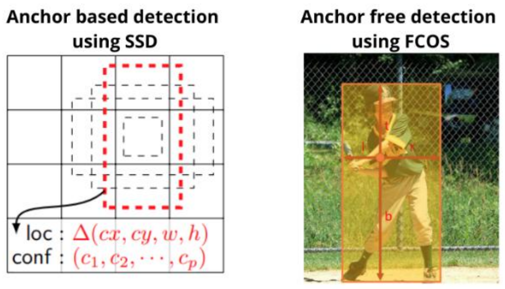

Anchor based detection using SSD
loc : $\Delta(cx, cy, w, h)$
conf : $(c_1, c_2, \cdots, c_p)$
Anchor free detection using FCOS

Figure 2.3 Anchor-free Detection Technique

## 2.4 Existing Methods and Applications for Pothole Detection

There were three existing methods used for pothole detection deployments: vehicles, drones and web or app-based applications. Each of the deployment's method will be explained in detail.

First method is deployment on vehicles [18]. The system operates using Intel Realsense D435i camera and Nvidia Jetson Xavier NX mounted on the vehicle. During operation, the camera captures images and depth information which used to calculate how far a detected pothole is. The model can accurately detect potholes at 1.5 meter and beyond. The captured images are processed by the YOLOv5 model running on the Nvidia Jetson Xavier NX. The model will analyse the image and draw bounding boxes around detected potholes with pothole class labelled. Besides, the distance of the detected pothole from the vehicle will also be displayed beside the output class.

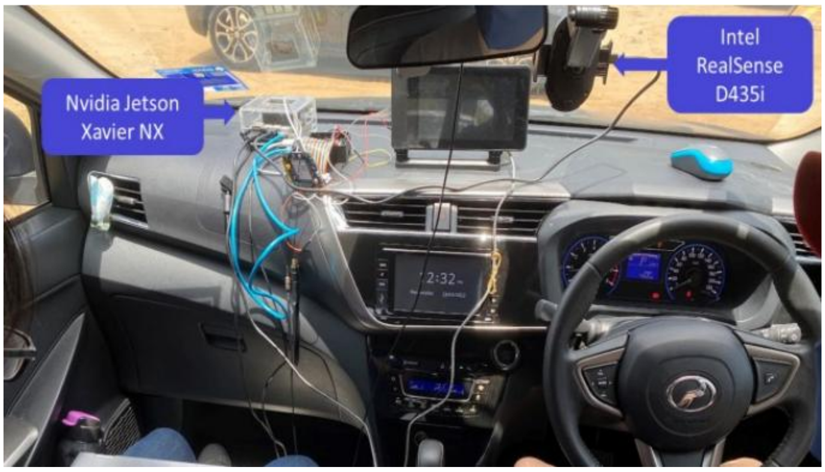


Figure 2.4 Pothole Detection System Mounted onto Vehicle

The second method is deployment on drone and edge AI hardware [19]. The whole process can be referred to block diagram as shown in Figure 2.5. In the research journal, a DJI-Mavic Mini 2 drone is being employed to capture ral-time images and videos of the road surface. After that, the capture video feed from the drone is transmitted to the edge device using the RTMP. This setup requires an IP address configuration to establish connectivity between the drone and edge device. The transmitted video was the same as last method process by YOLO model deployed on the edge device. The result of the detection is display on a hardware interface (LED monitor) connected to the edge device. The system shows the detected piotholes in video stream, allowing for immediate visual feedback on the detection process.

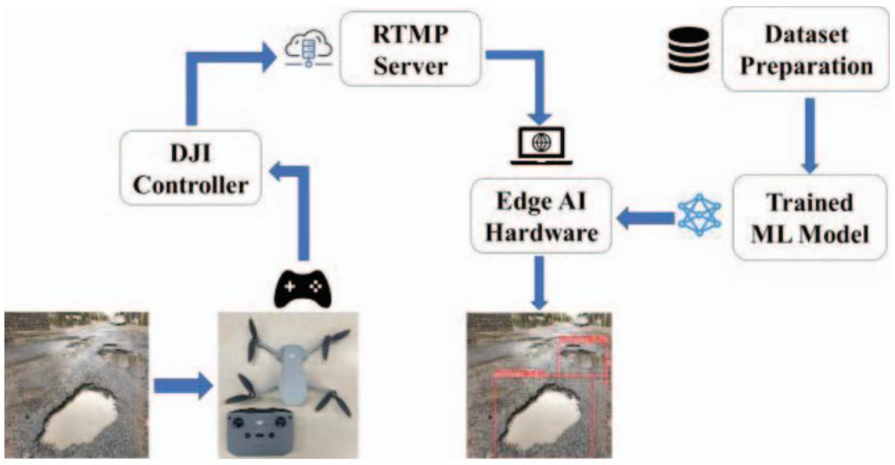


Figure 2.5 Block Diagram Showing Pothole Detection System using Drone and Edge AI Hardware

The third method involves deployment on mobile applications [20]. This application enables users to report potholes through two modes: Hands-Free Detection and Capture Image Mode. In Hands-Free Detection, the smartphone's accelerometer detects potholes automatically by measuring changes in acceleration as the vehicle passes through a pothole. In Capture Image Mode, users manually capture and upload images of potholes along with GPS location data. These images are processed using the YOLOv3 model, which generates alerts for nearby users if a pothole is detected. All reported potholes are geotagged and displayed on a map within the app. The government authorities, such as the Malaysian Public Works Department, able to use this platform to manage reports and track repair progress.

## 2.5 Existing Challenges and Solutions for Pothole Detection

Pothole detection is a critical aspect of road maintenance as it directly impacts vehicular safety and infrastructure durability. However, detecting potholes accurately and efficiently remains a significant challenge [21]. The dynamic nature of road environments can raise these challenges, variability in lighting conditions and the complexity of real-time implementation. Recent computer vision and machine learning advances have introduced promising solutions to these issues. This section examines the existing challenges in pothole detection and the solutions proposed by researchers in the field.

One of the primary challenges in pothole detection is the variability in road surface conditions. Road abnormalities such as cracks, debris and manhole will be classified as potholes because they share the same visual characteristics in terms of size, shape and color. Traditional methods that rely on edge detection or morphological operations often fail to distinguish between potholes and other types of road surface defects. Modern solutions involve leveraging CNNs for robust feature extraction and multi-class classification, enabling the detection of various road anomalies alongside potholes. Models like YOLO have demonstrated significant potential for this application by incorporating additional output classes [22] for cracks, patches and manholes, thereby broadening the scope of their functionality.

Another challenge that must be addressed is the impact of varying lighting conditions. This includes shadows, low visibility at night and reflections caused by wet surfaces which degrades the performance of the detection system. Researchers have proposed solutions such as data augmentation techniques [23-24] to train models under diverse lighting conditions. Besides, they also used IR or thermal imaging for enhanced visibility in low-light scenarios [25]. By incorporating diverse data during training, models become more robust and better equipped to generalize across varying environments. As a result, this can improve their reliability in real-world applications. Additionally, fusion techniques combining camera and LiDAR data [26] have shown promise in mitigating the influence of adverse lighting conditions.

Real-time implementation of pothole detection systems remains a daunting challenge, especially in resource-constrained environments. Edge devices often lack the computational capacity to run deep learning models at high FPS, leading to latency issues. To address this, researchers have developed lightweight neural networks optimized for edge devices [27] and explored techniques like quantization and pruning [28] to reduce model size without sacrificing accuracy. Optimization frameworks such as ONNX [29] and TensorRT [30] have also been utilized to accelerate inference on edge devices that enable faster processing while maintaining accuracy. These innovations collectively balance efficiency and effectiveness, allowing real-time deployment in vehicular systems and smart city infrastructure.

# CHAPTER 3

# RESEARCH METHODOLOGY

## 3.1 Project Workflow

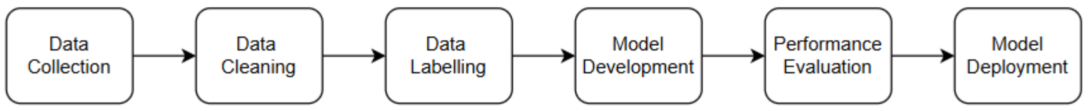


Figure 3.1 General AI Workflow

The workflow of General AI as depicted in Figure 3.1 can be divided into six stages: Data Collection, Data Cleaning, Data Labelling, Model Development, Performance Evaluation and Model Deployment.

During the first stage of data collection, data can be gathered in the form of images or videos. For the pothole detection project, data were collected from two sources: personal and online datasets. Personal data were captured using a smartphone camera for images and a car dashcam for videos. On the other hand, online data were obtained by searching datasets available on platforms such as Roboflow and Kaggle. After collected all the images and videos, it is prepared to enter the next stage which is data cleaning.

While capturing images and videos or sourcing them online, it can be observed that certain data obtained has low resolution or poor contrast. This issue can adversely impact the accuracy of AI when these data were fed into the framework for model training. To address such concerns, data cleaning technique were being employed. For instance, image enhancement technique such as histogram equalization and denoising were applied to improve the quality and contrast of images. Furthermore, duplicate or irrelevant images were removed to ensure that only high-quality and meaningful data proceeded to the next stage of data labelling.

Object detection as a supervised learning task requires that all data be properly labeled to ensure effective training. Accurate labeling provides the foundation for learning by offering clear examples of both positive cases such as images with potholes and negative cases where potholes are absent. After the dataset is prepared in earlier stages, the labeling process begins by defining the class labels for the objects of interest such as "Pothole." Each object is enclosed in a bounding box carefully drawn to specify its exact location within the image and assigned the appropriate class label for systematic categorization. Tools such as CVAT or Roboflow can be used to facilitate this process. To ensure balanced learning, the dataset is designed with a mix of positive and negative samples enabling the model to effectively differentiate between images with and without potholes. Proper labeling practices are vital as they form the groundwork for a precise and reliable object detection system and support the subsequent model development process.

The model development process is regarded as the core of the General AI Workflow where the YOLO framework is selected and labeled data is utilized to train and fine-tune the object detection model. The process starts with the training environment being set up and the labeled dataset being divided into training and validation sets, typically in an 8:2 ratio. The labeled data is then converted into the YOLO format where each image is associated with a text file containing class labels and bounding box coordinates. A pre-trained YOLO model is employed as a base to leverage transfer learning which allow existing features to be utilized and adapted efficiently to the new dataset. This structured approach ensures a solid foundation is established for training and optimizing the model. Once training is completed, key metrics such as Precision (P), Recall (R), mean Average Precision (mAP), and mAP50-90 are computed to evaluate performance. Precision measures the accuracy of identifying true positives, while Recall assesses the ability to detect all relevant instances. The mAP metric evaluates overall performance by combining Precision and Recall, with mAP50-90 offering a detailed view across multiple IoU thresholds. These results are displayed after training and analyzed as the basis for performance evaluation.

The performance evaluation phase focuses on assessing the trained model's effectiveness in detecting potholes accurately and consistently. Metrics such as Precision, Recall, and mean Average Precision (mAP) are analyzed in detail. If low Precision is observed, it may indicate a high number of false positives. This problem can be addressed by refining the dataset to include more negative samples or by adjusting confidence thresholds to filter out detections with low confidence. On the other hand, if low Recall is observed, it suggests that the model is missing actual potholes. This can be improved by increasing the diversity of positive samples in the training data or experimenting with different augmentation techniques. To enhance mAP, additional training data can be incorporated or hyperparameters such as learning rate, batch size and anchor box sizes can be adjusted. This iterative evaluation process ensures the model meets performance standards and is robust for deployment.

Once the model achieved desired performance during evaluation, it is prepared for deployment in the target environment. The deployment process begins by optimizing the model for inference, often converting it to lightweight formats like ONNX or TensorRT to enhance speed and reduce resource requirements. The optimized model is then integrated into the application pipeline, such as a real-time road inspection system, and deployed on platforms like edge devices or cloud servers based on the use case. Performance testing is conducted in the deployment environment to verify that the model operates effectively under real-world conditions, including variations in lighting, weather and camera perspectives. If issues arise, feedback loops are implemented to continuously monitor and address any performance degradation or new challenges post-deployment. This step ensures the model functions reliably and meets practical requirements in real-world applications.

## 3.2 Proposed Method

For the current stage of the Final Year Project (FYP) Part 1, a total of 959 images were collected from both personal sources and online platforms. Out of these, 826 images were allocated for training while the remaining 133 images were set aside for validation. Following data cleaning and labeling, the next step is to select a version of the YOLO framework to train the AI model in the model development process. Among the available options, YOLOv5 and YOLOv8 are the most widely used versions. To determine the most suitable version for deployment in the system, several experiment will be conducted to compare YOLOv5 and YOLOv8. This comparison will focus on key factors such as model accuracy, inference speed, ease of implementation and resource efficiency. The findings from this experiment will guide the final decision on which version to adopt into the system.

The first experiment focused on the development and evaluation of the AI model using the YOLO framework. A total of 826 images were used for training, while 133 images were reserved for validation. Both YOLOv5n and YOLOv8n, lightweight versions of their respective frameworks were trained under the same conditions to ensure fairness. Training was conducted for 30 epochs with identical hyperparameters, including learning rate, batch size and image size. After training, the models were evaluated based on key performance metrics: Precision (P), Recall (R), mean Average Precision at IoU 0.5 (mAP50) and mean Average Precision across IoU thresholds (mAP@50-95). These metrics provide insights into the accuracy, sensitivity and detection quality of the AI models. Additionally, the experiment also analysed the size of the model's pt file generated during training. This file is crucial for deployment and helps evaluate storage efficiency. To further study the models' real-world performance, both YOLOv5n and YOLOv8n trained .pt file were tested on 100 randomly selected images. The test assessed their inference speed, memory usage and number of parameters. These evaluations will help determine the most suitable YOLO framework for deployment by balancing accuracy, efficiency and computational requirements.

The second experiment focuses on evaluating the performance of the models during deployment. Randomly selected images are analyzed using the trained .pt files from YOLOv5n and YOLOv8n. This stage aims to assess the models' ability to accurately detect objects in unseen data and provide reliable predictions. Key aspects analyzed include the quality of bounding boxes and the confidence levels assigned to each detected object. The bounding box quality reflects how well the model identifies and localizes objects within the image which is critical for applications where precise object detection is required. Accurate bounding boxes ensure minimal false positives and false negatives thus improving overall system reliability. The confidence level provides an indication of how certain the model is about its predictions. A higher confidence level suggests more reliable detections which is essential for real-world applications to minimize errors and maintain consistency. Analyzing random images is important because it provides insight into the model's ability to generalize to new and diverse scenarios. This step helps identify potential weaknesses such as overfitting to the training data or difficulty in detecting objects under challenging conditions like varying lighting, occlusion or complex backgrounds. It also highlights the robustness of the model across different environments, ensuring it meets the requirements of practical deployment.

## 3.3 Tools and Platforms

For this project, data was collected using a combination of a smartphone camera for capturing images and a car dashcam for recording videos. These sources provided a diverse set of real-world data essential for developing the pothole detection system. To supplement this, additional datasets were sourced from the Roboflow platform that offered a wide variety of images suitable for training the AI model. The images collected were prepared for labelling using CVAT, a powerful tool for image annotation. This allowed precise labelling of potholes to ensure high-quality training data. The YOLOv5n and YOLOv8n models were trained and evaluated using Jupyter Notebook. This platform provided a flexible environment for data analysis, model training and performance evaluation. The use of Jupyter Notebook facilitated iterative experimentation with hyperparameters and performance metrics, enabling a thorough evaluation of model suitability. Visual Studio Code was employed for further development and fine-tuning of the models. This development environment ensured efficient management of code, streamlined debugging processes and seamless integration of model optimization techniques. This comprehensive approach to data collection, preparation and model development ensured that the tools and platforms used supported the project objectives effectively, aligning with the requirements for deploying a robust and scalable pothole detection system.

## 3.4 Cost of Project

Table 3.1 Cost of Project

| Tools/Software/Equipment | Price (RM) |
|---|---|
| Ultralytics YOLO | 0.00 |
| Jupyter Notebook | 0.00 |
| VS Code (Free Version) | 0.00 |
| CVAT (Trial Version) | 0.00 |
| Roboflow | 0.00 |
| NVIDIA Jetson Orin Nano Super Dev Kits (8GB) | 1299.00 |
| 512GB NVMe SSD Preloaded JetPack | 400.00 |
| ESP32 | 28.00 |
| GY-NEO6MV2 GPS Module | 22.90 |
| 7-Inch Capacitive Touch LCD Screen | 289.00 |
| XIAOMI YI Smart Dash Cam | 200.00 |

# CHAPTER 4

# RESULTS AND DISCUSSION

## 4.1 The Big Picture

Both experiments as mentioned in section 3.2, were conducted with the aim of selecting the most suitable YOLO version to be deployed for this project. Through these experiments, I obtained valuable results and findings about the performance and computational loads of each YOLOv5 and YOLOv8 model. These two aspects are critically important for applying AI in real-world scenario because they directly impact the practicability and the effectiveness of the solution.

In real-world applications, a high-performance AI model is essential to ensure accurate and reliable detection of road potholes. High accuracy enables the consistent identification of potholes, reducing the likelihood of false positives (incorrectly identifying non-pothole areas as potholes) and false negatives (failing to detect actual potholes). This reliability is crucial for the effectiveness of pothole detection systems, ensuring that potholes are accurately identified and timely notifications are sent to drivers. With this system, it can enhance road safety and reduce the risk of accidents. Beyond safety, accurate detection also plays a significant role in optimizing resource allocation. By correctly identifying the locations and severity of potholes, maintenance teams can prioritize repairs and focusing their efforts on actual issues instead of being misled by false alarms. This targeted approach minimizes unnecessary resource usage and effort which enabling maintenance operations to be more efficient. Over time, precise detection contributes to significant cost savings for municipalities and organizations responsible for road maintenance. By streamlining the repair process and avoiding wasteful allocation of resources, these systems make it easier to manage infrastructure effectively. Moreover, the improved efficiency and reliability of such systems can encourage public trust and support for broader adoption of AI in smart city initiatives.

At the same time, computational efficiency is a key factor in deploying AI models for real-world applications. Models with high computational demands may be impractical for deployment on edge devices like Raspberry Pi or Nvidia Jetson, which are commonly used in real-time, low-cost and portable AI systems. Larger AI models, which often have more parameters and layers, require significant processing power and memory. This can lead to increased computational load, resulting in slower inference times and reduced FPS. For instance, a model with a larger size may excel in accuracy but may struggle to process data quickly enough for real-time applications, where delays of even a fraction of a second can compromise the system's effectiveness. In scenarios like continuous pothole detection, a low FPS could result in missed detections, especially in fast-moving vehicles or rapidly changing environments. On the other hand, lightweight models which are optimized for efficiency, able to process data at higher FPS rates which makes them more suitable for edge devices with limited hardware resources.

Thus, the focus of these experiments was to strike a balance between detection accuracy and computational efficiency to ensure that the selected YOLO model is not only robust in detecting potholes but also adaptable to the constraints of real-world deployment environments.

## 4.2 Analytical Evaluation between YOLOv5n & YOLOv8n

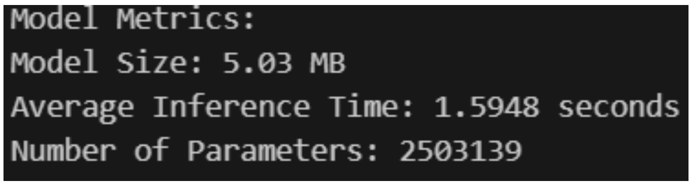
### 4.2.1 YOLOv5n vs YOLOv8n: Performance Comparison

Table 4.1 YOLOv5n and YOLOv8n Performance Matrices

| Version | P | R | mAP50 | mAP50-95 |
|---|---|---|---|---|
| YOLOv5n | 0.776 | 0.709 | 0.776 | 0.479 |
| YOLOv8n | 0.751 | 0.727 | 0.786 | 0.504 |

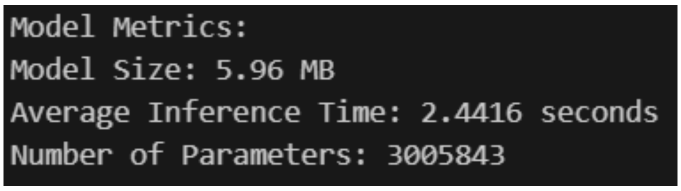
The results of Experiment 1 as outlined in Section 3.2 are presented in Table 4.1. The experiment has demonstrated that YOLOv8n generally outperforms YOLOv5n in terms of recall (0.727 vs. 0.709), mAP50 (0.786 vs. 0.776) and mAP50-95 (0.504 vs. 0.479). However, YOLOv5n shows slightly higher precision (0.776 vs. 0.751).

### 4.2.2 YOLOv5n vs YOLOv8n: Computational Resource Comparison



```
Model Metrics:
Model Size: 5.03 MB
Average Inference Time: 1.5948 seconds
Number of Parameters: 2503139
```

Figure 4.1 YOLOv5n Model Size, Average Inference Time and Number of Parameters

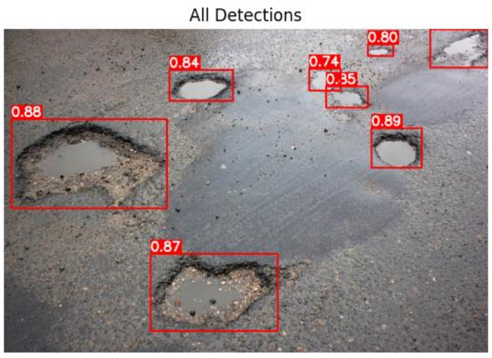


```
Model Metrics:
Model Size: 5.96 MB
Average Inference Time: 2.4416 seconds
Number of Parameters: 3005843
```

Figure 4.2 YOLOv8n Model Size, Average Inference Time and Number of Parameters

Experiment 1 in Section 2.3 involved use trained .pt files from YOLOv5n and YOLOv8n to analyze 100 random images. The model metrics of computational load for both YOLOv5n and YOLOv8n models are presented in Figures 4.1 and 4.2. Based on these results, YOLOv5n has a smaller model size than YOLOv8n (5.03 MB vs. 5.96 MB). The average inference time for YOLOv5n is also lower than YOLOv8n (1.5948 seconds vs. 2.4416 seconds) which makes it faster in processing. Lastly, YOLOv5n has fewer parameters compared to YOLOv8n (2,503,139 vs. 3,005,843) which further highlights its computational efficiency.

### 4.2.3 YOLOv5n vs YOLOv8n: Image Analysis Performance



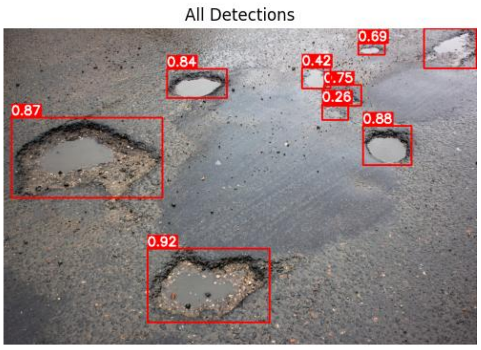
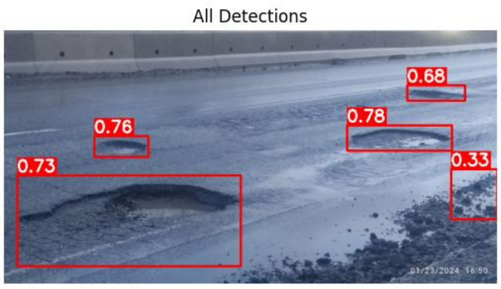


Figure 4.3 & Figure 4.4 Comparison of Pothole Detection of a Random Image using YOLOv5n and YOLOv8n Models



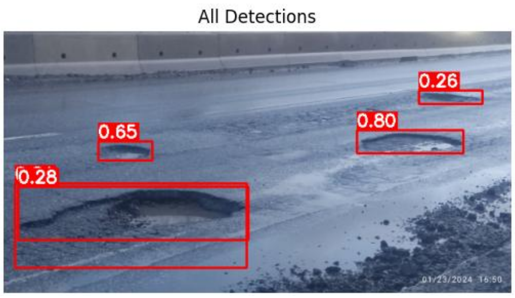


Figure 4.5 & Figure 4.6 Comparison of Pothole Detection of a Random Image using YOLOv5n and YOLOv8n Models

Experiment 2 in Section 2.3 utilized the trained .pt files from YOLOv5n and YOLOv8n to analyze random images. This involve focusing on the bounding boxes and confidence scores displayed on the images. As illustrated in Figures 4.4 and 4.6, YOLOv8n tends to draw multiple bounding boxes for two closely positioned potholes rather than generalizing them into a single bounding box. Further, the same figures indicate that YOLOv8n struggles with objects that have low contrast or visibility which result in lower confidence scores during detection.

## 4.3 Result and Discussion

### 4.3.1 YOLOv5n vs YOLOv8n: Performance Comparison

The results in Table 4.1 demonstrate that YOLOv8n slightly outperforms YOLOv5n in terms of recall (0.727 vs. 0.709), mAP50 (0.786 vs. 0.776) and mAP50-95 (0.504 vs. 0.479). But on the other hand, YOLOv5n achieves a higher precision (0.776 vs. 0.751). This indicates that YOLOv5n is more conservative and reliable in its predictions due to fewer false positives compared to YOLOv8n. For applications where precision is critical, such as safety-critical systems or where false alarms can have significant consequences, YOLOv5n is the preferred choice. While YOLOv8n offers slight improvements in recall, its increased tendency to generate false positives could limit its applicability in precision-focused scenarios.

### 4.3.2 YOLOv5n vs YOLOv8n: Computational Resource Comparison

Figures 4.1 and Figure 4.2 highlight YOLOv5n's superiority in computational efficiency which make it a more practical choice for deployment in real-world applications. With a smaller model size (5.03 MB vs. 5.96 MB), YOLOv5n requires less storage and is better suited for resource-constrained environments like embedded or edge devices. Moreover, its faster inference time (1.5948 seconds vs. 2.4416 seconds) ensures quicker processing which is crucial for real-time object detection applications. In addition, YOLOv5n has significantly fewer parameters (2,503,139 vs. 3,005,843) thus reducing memory usage and computational load. These characteristics make YOLOv5n an optimal choice for scenarios where computational resources are limited without compromising detection reliability.

### 4.3.3 YOLOv5n vs YOLOv8n: Image Analysis Comparison

From Figures 4.3 to Figure 4.6, it is revealed that YOLOv8n's ability to segment closely positioned objects into multiple bounding boxes may lead to over-segmentation. This can introduce unnecessary complexity in certain applications. In contrast, YOLOv5n offers more generalized bounding boxes which are often more practical in situations requiring simplified object localization. Moreover, YOLOv5n demonstrates greater robustness under challenging conditions such as low contrast or poor visibility and avoiding the lower confidence scores that YOLOv8n tends to produce in such scenarios. This highlights YOLOv5n's reliability and suitability for real-world applications where lighting and environmental conditions may vary. Its generalized approach to bounding box generation also reduces the likelihood of over-detection, providing more practical results for downstream processing.

## 4.4 Chapter Summary

These findings strongly position YOLOv5n as a more balanced and practical solution for many object detection tasks. While YOLOv8n achieves marginal improvements in detection metrics, its higher computational requirements, slower inference speed and susceptibility to over-segmentation with low-visibility challenges make it less favorable for real-world, resource-constrained or precision-critical applications. YOLOv5n's higher precision, computational efficiency and robust performance under diverse conditions make it the preferred choice for practical deployment, particularly in edge environments or scenarios requiring reliable and efficient object detection.

# CHAPTER 5

# CONCLUSION AND RECOMMENDATIONS

## 5.1 Conclusions

In this research, a detailed comparison between YOLOv5n and YOLOv8n was conducted to evaluate their suitability for real-time pothole detection. The results demonstrate that while YOLOv8n offers slight improvements in terms of recall and detection accuracy (mAP50 and mAP50-95), YOLOv5n proves to be the more practical choice for deployment in real-world applications. YOLOv5n outperforms YOLOv8n in precision which ensuring fewer false positives. YOLOv5n also excels in computational efficiency with a smaller model size, faster inference time and fewer parameters. These characteristics make YOLOv5n highly suitable for resource-constrained environments like edge devices. Furthermore, YOLOv5n's robustness in handling challenging visual conditions makes it a reliable solution for real-time pothole detection.

## 5.2 Contributions to Knowledge

This research contributes to the field of AI-driven road monitoring by providing a detailed performance and computational efficiency comparison between YOLOv5n and YOLOv8n. It highlights YOLOv5n as a practical solution for real-time pothole detection in resource-constrained environments. By developing a robust and diverse dataset, this study ensures effective training and evaluation of the models under varied real-world conditions. The findings emphasize the importance of lightweight models for deployment on edge devices and addressing challenges like limited computational power and real-time performance requirements. This research not only advances the understanding of model trade-offs for pothole detection but also offers practical insights for designing scalable and cost-effective systems that improve road safety and infrastructure maintenance.

## 5.3 Future Works

Building on the findings from FYP Part 1, the second phase of the research (FYP Part 2) will focus on the deployment of the selected AI model onto edge device NVIDIA Jetson Orin Nano Super (8GB). This phase tackles the challenges of creating a portable AI application capable of real-time performance while minimizing resource usage. The research also investigates integrating this system with XIAOMI YI Smart Dash Cam mounted on vehicle to enable automated pothole detection and monitoring. On the other hand, ESP32 also equipped with GY-NEO6MV2 GPS Module to record real-time coordinate if pothole detected. These approaches introduce additional challenges which includes stable data processing during driving, managing limited hardware battery life, addressing computational constraints and overcoming environmental factors like lighting variability and camera vibrations. Model compression, optimization and hardware-software integration will be prioritized to ensure efficient operation under these conditions. The study aims to deliver a robust, real-time and scalable solution using edge computing and camera technology to improve road infrastructure monitoring sustainably and cost-effectively.

# REFERENCES

[1] S. D. Reporter, “Potholes plaguing Malaysian roads: New solution found? | Sinar Daily,” *Sinar Daily*, Sep. 13, 2024. [Online]. Available: https://www.sinardaily.my/article/221417/focus/national/potholes-plaguing-malaysian-roads-new-solution-found/

[2] Z. M. Yusof, “Report damaged roads on MyJalan app,” *Free Malaysia Today | FMT*, Aug. 23, 2024. [Online]. Available: https://www.freemalaysiatoday.com/category/nation/2024/08/23/report-damaged-roads-on-myjalan-app/

[3] L. Madfa, “Smart Road Maintenance, report using Waze,” *Selangor Journal*, Oct. 31, 2016. [Online]. Available: https://selangorjournal.my/2016/10/smart-road-maintenance-report-using-waze/

[4] M. Hussain, “Deep Learning based One-shot vs Two-shot Object Detection,” *Medium*, Nov. 29, 2024. [Online]. Available: https://mazhar-hussain.medium.com/deep-learning-based-one-shot-vs-two-shot-object-detection-21e9634307bd/

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 779–788. doi: 10.1109/CVPR.2016.91.

[6] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017, doi: 10.1109/TPAMI.2016.2577031.

[7] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct. 2017, pp. 2980–2988. doi: 10.1109/ICCV.2017.322.

[8] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” 2020, *arXiv*. doi: 10.48550/ARXIV.2004.10934.

[9] M. Issame and A. Benyounes, “Real Time Object Detection With Drone Using Deep Learning Algorithm,” *2022 International Conference of Advanced Technology in Electronic and Electrical Engineering (ICATEEE)*, Nov. 2022, pp. 1–6. doi: 10.1109/ICATEEE57445.2022.10093104.

[10] H. Kulhandjian, J. M. Torres, N. Amely, C. Nieves, C. Reeves, and M. Kulhandjian, “AI-based Road Inspection Framework Using Drones with GPS-less Navigation,” *2024 International Conference on Computing, Networking and Communications (ICNC)*, Feb. 2024, pp. 301–305. doi: 10.1109/ICNC59896.2024.10556004.

[11] C. Surianarayanan, P. Raj, and S. K. Niranjan, “The Significance of Edge AI towards Real-time and Intelligent Enterprises,” *2023 International Conference on Intelligent and Innovative Technologies in Computing, Electrical and Electronics (IITCEE)*, Jan. 2023, pp. 1–6. doi: 10.1109/IITCEE57236.2023.10090926.

[12] Z. Li, W. Liu, Z. Xie, X. Kang, P. Duan, and S. Li, “FAA-Det: Feature Augmentation and Alignment for Anchor-Free Oriented Object Detection,” *IEEE Transactions on Geoscience and Remote Sensing*, vol. 62, pp. 1–11, 2024, doi: 10.1109/TGRS.2024.3504598.

[13] F. Dadboud, V. Patel, V. Mehta, M. Bolic, and I. Mantegh, “Single-Stage UAV Detection and Classification with YOLOV5: Mosaic Data Augmentation and PANet,” *2021 17th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, Nov. 2021, pp. 1–8. doi: 10.1109/AVSS52988.2021.9663841.

[14] Y.-Y. Chen, T.-L. Lin, and H.-Y. Chang, “The object detection in remote sensing imagery based on mosaic improvement,” *2023 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, Jul. 2023, pp. 501–502. doi: 10.1109/ICCE-Taiwan58799.2023.10227021.

[15] G. Jocher, “Yolov5,” *PyTorch*, Nov. 22, 2022. [Online]. Available: https://pytorch.org/hub/ultralytics_yolov5/

[16] Y.-Y. Liu, H.-S. Zheng, Y. Fang Hu, C.-F. Hsu, and T. T. Yeh, “TinyTS: Memory-Efficient TinyML Model Compiler Framework on Microcontrollers,” *2024 IEEE International Symposium on High-Performance Computer Architecture (HPCA)*, Mar. 2024, pp. 848–860. doi: 10.1109/HPCA57654.2024.00070.

[17] H. Ham *et al.*, “ONNXim: A Fast, Cycle-Level Multi-Core NPU Simulator,” *IEEE Computer Architecture Letters*, vol. 23, no. 2, pp. 219–222, Jul. 2024, doi: 10.1109/LCA.2024.3484648.

[18] A. Yang Her, W. Kean Yew, P. Jia Yew, and M. Chong Jia Ying, “Real-time pothole detection system on vehicle using improved YOLOv5 in Malaysia,” *IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society*, Oct. 2022, pp. 1–5. doi: 10.1109/IECON49645.2022.9968423.

[19] A. Parmar, R. Gaiiar, and N. Gajjar, “Drone based Potholes detection using Machine Learning on various Edge AI devices in Real-Time,” *2023 IEEE International Symposium on Smart Electronic Systems (iSES)*, Dec. 2023, pp. 22–26. doi: 10.1109/iSES58672.2023.00016.

[20] S. R. Kuthyar *et al.*, “An Intelligent Pothole Detection and Alerting System using Mobile Sensors and Deep Learning,” *2021 IEEE 18th India Council International Conference (INDICON)*, Dec. 2021, pp. 1–6. doi: 10.1109/INDICON52576.2021.9691661.

[21] B. Bučko, E. Lieskovská, K. Zábovská, and M. Zábovský, “Computer Vision Based Pothole Detection under Challenging Conditions,” *Sensors*, vol. 22, no. 22, p. 8878, Nov. 2022, doi: 10.3390/s22228878.

[22] O. M. Khare, S. Gandhi, A. M. Rahalkar, and S. Mane, “YOLOv8-Based Visual Detection of Road Hazards: Potholes, Sewer Covers, and Manholes,” *arXiv*, 2023, doi: 10.48550/ARXIV.2311.00073.

[23] J. Hong, Y. Jung, and K. S. Woo, “Image Data Augmentation and Detection Study for Pothole Detection Algorithm,” *2023 IEEE International Conference on Big Data (BigData)*, Dec. 2023, pp. 6162–6164. doi: 10.1109/BigData59044.2023.10386900.

[24] L. Li, J. Liu, J. Xing, Z. Liu, K. Lin, and B. Du, “Road Pothole Detection Based on Crowdsourced Data and Extended Mask R-CNN,” *IEEE Trans. Intell. Transport. Syst.*, vol. 25, no. 9, pp. 12504–12516, Sep. 2024, doi: 10.1109/TITS.2024.3360725.

[25] Y. Zanevych, V. Yovbak, O. Basystiuk, N. Shakhovska, S. Fedushko, and S. Argyroudis, “Evaluation of Pothole Detection Performance Using Deep Learning Models Under Low-Light Conditions,” *Sustainability*, vol. 16, no. 24, p. 10964, Dec. 2024, doi: 10.3390/su162410964.

[26] A. Karukayil, C. Quail, and F. Auat Cheein, “Deep Learning Enhanced Feature Extraction of Potholes Using Vision and LiDAR Data for Road Maintenance,” *IEEE Access*, vol. 12, pp. 184541–184549, 2024, doi: 10.1109/ACCESS.2024.3512783.

[27] Y. Nimmagadda, “Model Optimization Techniques for Edge Devices,” *Model Optimization Methods for Efficient and Edge AI*, 1st ed., P. R. Chelliah, A. M. Rahmani, R. Colby, G. Nagasubramanian, and S. Ranganath, Eds., Wiley, 2025, pp. 57–85. doi: 10.1002/9781394219230.ch4.

[28] G. Victor Daniel, M. Trupthi, G. Sridhar Reddy, A. Mallikarjuna Reddy, and K. Hemanth Sai, “AI Model Optimization Techniques,” *Model Optimization Methods for Efficient and Edge AI*, 1st ed., P. R. Chelliah, A. M. Rahmani, R. Colby, G. Nagasubramanian, and S. Ranganath, Eds., Wiley, 2025, pp. 87–108. doi: 10.1002/9781394219230.ch5.

[29] F. Xu, “Faster Scalable ML Model Deployment Using ONNX and Open Source Tools,” *2020 IEEE Infrastructure Conference*, Oct. 2020. doi: 10.1109/IEEECONF47748.2020.9377615.

[30] Y. Zhou and K. Yang, “Exploring TensorRT to Improve Real-Time Inference for Deep Learning,” *2022 IEEE 24th Int Conf on High Performance Computing & Communications; 8th Int Conf on Data Science & Systems; 20th Int Conf on Smart City; 8th Int Conf on Dependability in Sensor, Cloud & Big Data Systems & Application (HPCC/DSS/SmartCity/DependSys)*, Hainan, China: IEEE, Dec. 2022, pp. 2011–2018. doi: 10.1109/HPCC-DSS-SmartCity-DependSys57074.2022.00299.