MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Pothole Detection for Autonomous Cars

BACHELOR OF TECHNOLOGY

IN

MECHATRONICS

Submitted by

Shreyas J Rao - 200929194

Aditya Vadduri - 200929080

K Sai Kavya - 200929299

Diya Parekh - 200929176

Isha Harish - 200929098

Under the guidance of

Dr. Umesh Kumar Sahu

Assistant Professor

DEPARTMENT OF MECHATRONICS MANIPAL INSTITUTE OF TECHNOLOGY
(A Constituent of Manipal Academy of Higher Education) MANIPAL - 576104, KARNATAKA, INDIA November 2023

ABSTRACT

Globally, potholes are a major source of concern for drivers. They can harm automobiles, result in collisions, and make roads uncomfortable and dangerous for both pedestrians and cars. The development of automatic pothole detecting systems has garnered increasing attention in the past few years. These systems detect potholes in real time or from stored data using a range of sensors and technology. The development of devices that function consistently in all weather and traffic situations is one of the primary problems in pothole identification. In poor light or inclement weather, potholes might be hard to spot because of traffic or debris. Pothole detection and repair could be completely transformed by deep learning.

The goal of this project is to develop an accurate, dependable, and reasonably priced deep learning-based pothole detection system. An extensive dataset of photos, both of potholes and non-potholes, will be used to train the algorithm. This project uses the YOLO technique to find potholes. You Only Look Once, or YOLO, is an algorithm designed to recognize and classify items in images or video frames. The primary innovation of YOLO is its ability to detect things in a single neural network pass, which allows it to detect objects faster than traditional two-stage detection algorithms.Because of its versatility, the YOLO method has been applied to a wide range of tasks, such as real-time video and object recognition in photos. Various iterations of YOLO have been created, including YOLOv2, YOLOv3, and YOLOv4.

Keywords: Pothole detection, Deep Learning, YOLO

LIST OF FIGURES

Figure-1	15
Figure-2	17
Figure-3	18
Figure-4	21
Figure-5	21
Figure-6	22
Figure-7	22
Figure-8	22
Figure-9	22
Figure-10	23
Figure-11	23
Figure-12	23
Figure-13	24
Figure-14	24
Figure-15	24
Figure-16	24
Figure-17	24
Figure-18	25
Figure-19	26
Figure-20	26
Figure-21	27
Figure-22	27

LIST OF TABLES

Table-1	9
Table-2	28

TABLE OF CONTENTS

Abstract 2

List of Figures 3

List of Tables 4

Chapter 1 Introduction 6

1.1 Introduction 6

1.2 Motivation 7

Chapter 2 Background

2.1 Review of Literature 8

2.2 Summary of Literature Review 8

2.3 Theoretical Background 9

Chapter 3 Definition 11

3.1 Problem Definition 11

3.2 Objectives 11

Chapter 4 Methodology 13

4.1 YOLO 13

4.2 YOLOv5 14

4.3 YOLOv8 16

Chapter 5 Results 20

5.1 Results 20

5.2 Discussion 28

Chapter 6 Conclusion 29

6.1 Conclusion 29

6.2 Future Scope 29

Reference **30**

CHAPTER 1
Introduction

2. Prologue

Pothole detection involves the use of technology to identify and pinpoint the locations of road surface depressions or holes known as potholes. These road imperfections result from a combination of factors, including weathering, traffic, and the natural deterioration of road materials. Potholes present a substantial risk to road users, contributing to accidents, vehicle damage, and heightened maintenance expenses for transportation authorities.

The incorporation of autonomous vehicles Into our transport system presents significant potential to transform how we move, promising increased efficiency, safety, and convenience. Nevertheless, the effective implementation of autonomous cars is contingent on the strength of their sensing and perception mechanisms. Potholes, common road hazards found everywhere, present a distinctive obstacle for these vehicles, impacting safety and the overall durability of the autonomous vehicle fleet.

In recent years, there has been a growing interest in developing automated pothole detection systems. These systems use a variety of sensors and technologies to identify potholes in real time or from recorded data. There are several different automated pothole detection systems that are currently available or under development. These systems can be broadly classified into two categories:

- ***Sensor-based systems**: These systems use sensors such as cameras, accelerometers, and ultrasonic sensors to detect potholes. Sensor-based systems are typically mounted on vehicles, but they can also be installed on fixed roadside structures.*
- ***Computer vision-based systems**: These systems use cameras to capture images of the road surface and then use computer vision techniques to identify potholes in the images. Computer vision-based systems can be deployed on vehicles.*

Deep learning has recently emerged as a powerful tool for pothole detection. Deep learning models can be trained on large datasets of images of potholes and non-potholes to learn to identify potholes with high accuracy. One such method is YOLO which we are adopting in this project. YOLO, an acronym for You Only Look Once, is an algorithm designed for the identification and classification of objects within images or video frames. The key innovation of YOLO lies in its capacity to rapidly detect objects,

surpassing traditional two-stage detection algorithms by executing the task in a single pass through a neural network. YOLO has been widely used in various applications, including autonomous vehicles, surveillance, and robotics, due to its real-time performance and accuracy.

1.2 Motivation

The motivation behind implementing pothole detection in self-driving cars stems from the essential need to improve the safety, efficiency, and durability of autonomous vehicle operations. Various crucial factors underpin this motivation:

1) *Ensuring Passenger Safety: Potholes present a considerable safety hazard for all vehicles, including self-driving cars. The real-time detection of potholes enables autonomous vehicles to navigate and adjust their actions to avoid these road imperfections, ensuring the safety of both passengers and pedestrians.*

2) *Preserving Vehicle Integrity: Potholes have the potential to inflict significant damage on vehicles, leading to heightened maintenance costs and possible breakdowns. Equipping self-driving cars with pothole detection capabilities enables them to proactively steer clear of or minimize the impact of these hazards, thereby contributing to the overall resilience and dependability of the vehicle fleet.*

3) *Enhancing Ride Comfort: Well-maintained, smooth roads contribute to a more enjoyable ride experience. By recognizing and steering clear of potholes, self-driving cars can enhance passenger comfort, ultimately improving the overall appeal and acceptance of autonomous transportation.*

4) *Integration with Smart City Initiatives: The integration of pothole detection in self-driving cars is in harmony with larger smart city endeavors. Through actively contributing to the establishment of a live road condition database, autonomous vehicles play an integral role in the maintenance of urban infrastructure, promoting a cooperative and interconnected transportation ecosystem.*

Chapter 2

Background

2.1 Literature Review

Several pothole detection methods such as Naïve Bayes [1], logistic regression [1], K-Nearest Neighbors [1], YOLO [2], etc. have been proposed over the last few years.

Comparison of all classical methods for pothole detection systems has been done in [1]. Logistic regression [1] assumes a linear decision boundary between classes. If the relationship between the features and presence of potholes is highly non-linear, logistic regression struggles to capture the complexities of the data. LR does not generalize well to unseen and diverse data. LR does not consider the spatial dependencies of pixels in an image.

KNN [1], computational cost of KNN increases with size of dataset as it needs to calculate distance between the query point and all data points. In image data feature space is high dimensional, this decreases the effectiveness of the algorithm. KNN doesn't train models, it stores data and makes predictions. This lack of trained model limits its ability to generalize effectively to unseen data, especially in cases where complex patterns need to be learned.

Naïve Bayes [1], is designed for numeric data, pothole detection tasks involve non-numeric data, it requires considering of spatial arrangement of pixels in an image.

Random forest trees [1] cannot be used in real time as it requires a lot of memory which is limited in real time applications. It is often considered as 'black box' model meaning it is challenging sometimes to interpret the reason behind a prediction.

SVM [1], does not perform well in high dimensional spaces, feature extractions. To solve these limitations deep learning approaches are used, particularly CNNs.

YOLOv2's [2,3,4] accuracy decreases when detecting small objects or when the pothole has low contrast with surrounding road surfaces. YOLOv4 [2,3,4] is more complex to train compared to earlier versions. YOLOv5 is implemented in PyTorch, it doesn't support other frameworks.

2.2 Summary

In summary, each method has its strengths and limitations. Logistic regression and KNN face challenges in capturing complex relationships and handling high-dimensional data. Naïve Bayes struggles with non-numeric data, while Random Forest trees are memory-intensive and lack interpretability. SVM faces difficulties in high-dimensional spaces. YOLOv2, YOLOv4, and YOLOv5 exhibit varying accuracies and complexities in

detecting potholes, with the latter having framework restrictions. The limitations of classical methods have led to the adoption of deep learning approaches like CNNs for improved performance in pothole detection tasks.

Method	Accuracy
Logistic Regression	66.67%
Naïve Bayes	77.56%
KNN	78.79%
SVM	82.00%
Random Forest Trees	80.71%
YOLOv2	82.96%
YOLOv4	83.78%

Table 1

2.3 Theoretical Background

Pothole detection involves 3 major techniques: Object Detection, Localization, Object Classification.

1) Object Detection: *Object detection is a task in computer vision that encompasses the identification and localization of objects within an image or a frame of a video.*

2) Localization: *In computer vision localization pertains to the task of ascertaining the spatial coordinates or placement of an object or entity within a specific environment. This process includes determining the position of the object within either a two-dimensional (2D) image or a three-dimensional (3D) space. Localization serves as a critical component in a range of applications, such as object detection, tracking, and navigation systems.*

3) Object Classification: *Object classification is a task in computer vision that involves assigning a specific category or label to an object within an image or a video frame. The primary goal is to identify the class or type to which the object belongs from a predefined set of classes. This process is a fundamental component of various applications, such as image recognition, scene understanding, and content-based image retrieval. Object classification plays a crucial role in enabling machines to interpret and categorize visual information accurately.*

The algorithm which we have adopted is YOLO algorithm which is an object detection algorithm that falls in the category of real-time computer vision tasks. It uses a deep neural network to perform object detection in real time. Specifically, YOLO falls under the category of convolutional neural networks (CNNs), which are a type of neural network architecture particularly effective for tasks involving images.

CNN stands for Convolutional Neural Network, which is a specialized type of deep neural network designed for processing and analyzing visual data. CNNs are particularly powerful in computer vision tasks, such as image recognition and object detection.

Key features of CNNs include:

- *Convolutional Layers:*
  - *CNNs use convolutional layers to apply convolutional operations to input data. These layers are capable of learning hierarchical representations of features within the data.*
- *Pooling Layers:*
  - *Pooling layers are used to down sample the spatial dimensions of the input volume, reducing computation in the network and making the learned features more robust to variations in scale and orientation.*
- *Fully Connected Layers:*
  - *Fully connected layers are typically found at the end of a CNN and are responsible for making predictions based on the learned features. These layers connect every neuron to every neuron in the subsequent layer.*
- *Activation Functions:*
  - *Activation functions, such as ReLU (Rectified Linear Unit), are applied to introduce non-linearity into the network, allowing it to learn complex patterns in the data.*
- *eight Sharing:*
  - *CNNs utilize weight sharing using shared filters in convolutional layers. This helps the network efficiently learn and detect features irrespective of their location in the input.*
- *Feature Hierarchies:*
  - *CNNs naturally form hierarchies of features. Lower layers often capture simple features like edges, while higher layers combine these features to represent more complex patterns and objects.*
- *Transfer Learning:*
  - *CNNs are well-suited for transfer learning, where pre-trained models on large datasets (e.g., ImageNet) can be fine-tuned for specific tasks with*

smaller datasets. This is especially useful when training resources are limited.

- *Applications:*
    - *CNNs are widely used in various applications, including image classification, object detection, facial recognition, and medical image analysis.*

CNNs have significantly advanced the field of computer vision and have been a key enabler for breakthroughs in tasks that involve visual data interpretation. Their ability to automatically learn hierarchical features from data makes them particularly effective in extracting complex patterns and representations.

Chapter 3

Problem Definition

3.1 Problem Definition

Pothole detection for self-driving cars represents a critical challenge in advancing the safety and efficiency of autonomous transportation systems. The problem at hand involves identifying and accurately locating potholes on road surfaces to enable proactive navigation and response mechanisms for self-driving vehicles. Potholes, resulting from factors like weathering, traffic, and road material wear, pose significant risks, including safety hazards, potential vehicle damage, and increased maintenance costs for transportation authorities.

The scope of the problem encompasses developing a robust and real-time pothole detection system capable of integrating seamlessly into the autonomous driving framework. The challenge involves designing algorithms and deploying sensor technologies, such as cameras, lidar, or radar, to detect and classify potholes. The system must not only identify the presence of potholes but also accurately determine their spatial coordinates, allowing for precise navigation adjustments.

Contextual understanding is crucial in considering the dynamic nature of road conditions and the varying characteristics of potholes. The system needs to distinguish between potholes and other road irregularities, ensuring a high level of accuracy in detection. Additionally, the problem involves addressing challenges such as adverse weather conditions, varying lighting scenarios, and the potential presence of multiple potholes in proximity.

Key Challenges:

1) *Developing systems that can work reliably in all weather and traffic conditions. Potholes can be difficult to identify in low light or bad weather, and they can be obscured by traffic or debris.*
2) *Developing systems that are affordable and easy to deploy. Pothole detection systems need to be accessible to road maintenance crews and other stakeholders.*
3) *Developing systems that are robust to noise and occlusion. Pothole detection systems should be able to accurately identify potholes even in the presence of noise (e.g., shadows, leaves, etc.) and occlusion (e.g., vehicles, pedestrians, etc.).*

3.2 Objectives

The main objective of this project is to develop a pothole detection system using YOLOv5 and YOLOv8, compare results and implement the best model for our application in real time maintaining high accuracy.

Secondary objectives include learning about YOLO algorithm, implementation of the same, difference between the two models.

Investigate and implement advanced computer vision techniques to enhance the pothole detection system's performance. Explore methods such as image preprocessing,

feature engineering, and post-processing algorithms to improve the accuracy and robustness of pothole identification.

Integrate the trained YOLOv8 model into the self-driving car's software stack. Curate and annotate a comprehensive dataset specifically tailored for pothole detection. Include diverse scenarios, road conditions, and pothole types to ensure the model's ability to generalize. Implement data augmentation techniques to enhance the dataset's diversity and improve the model's robustness.

Design and execute rigorous testing procedures for the pothole detection system. Evaluate the model's performance under various scenarios, including challenging lighting conditions, different road surfaces, and the presence of multiple potholes. Identify and address any false positives or false negatives to enhance system reliability.

These objectives highlight the diverse roles and responsibilities of individuals involved in the development of a pothole detection system for self-driving cars using YOLOv5 and YOLOv8. Each team member contributes to the project's success by leveraging their expertise in machine learning, computer vision, software integration, dataset curation, and quality assurance.

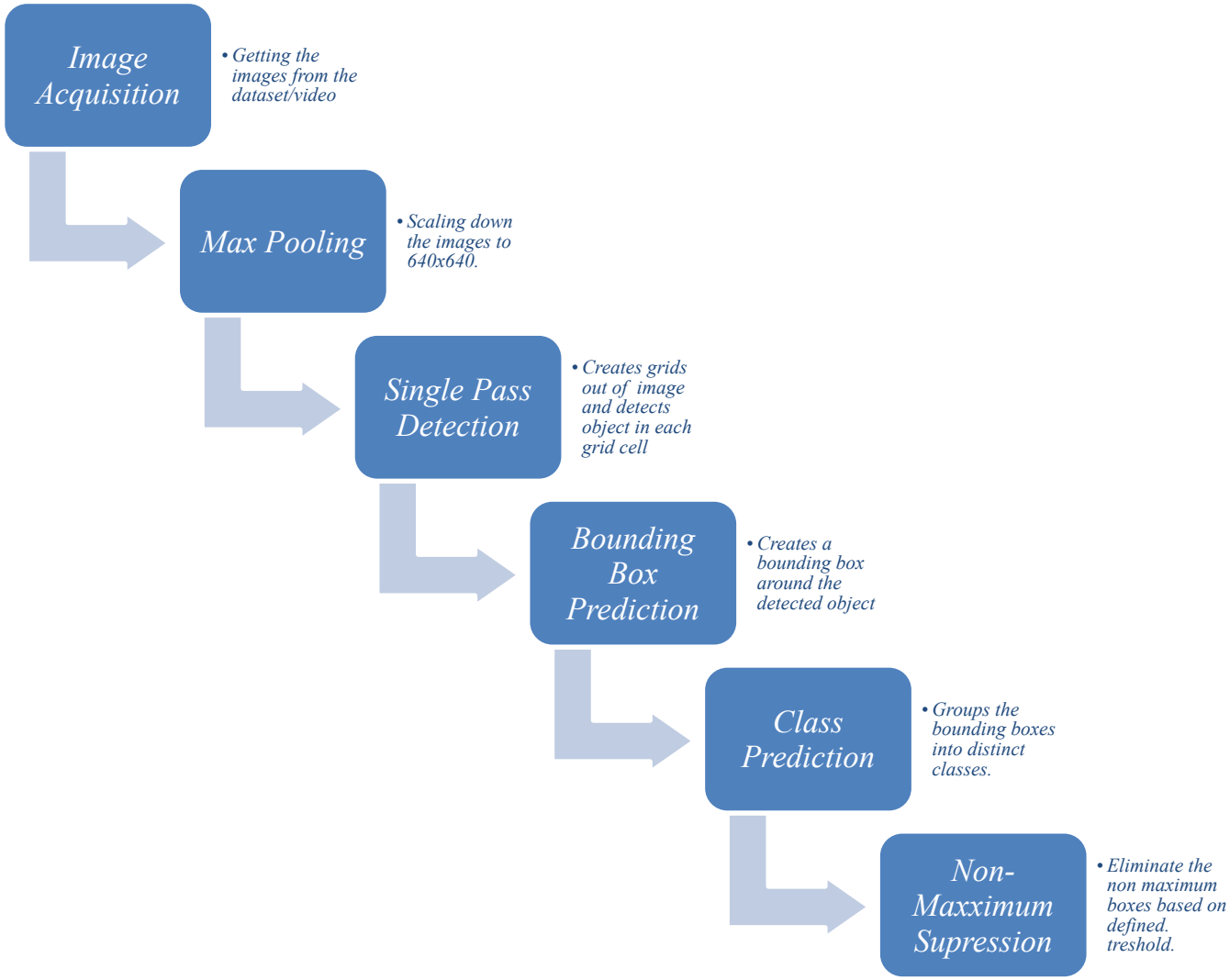
Chapter 4

Methodology

4.1 YOLO Algorithm

YOLO (you only look once) an object detection algorithm that falls in the category of real-time computer vision tasks. It uses a deep neural network to perform object detection in real time. Specifically, YOLO falls under the category of convolutional neural networks (CNNs), which are a type of neural network architecture particularly effective for tasks involving images.

Flow Chart:



The above process repeats for every image.

Detailed Explanation of the Algorithm:

1) *Single Pass Detection: YOLO creates a grid out of the input image and uses a single forward pass of the neural network to detect objects in each grid cell. YOLO can attain real-time performance as a result.*

2) *Bounding Box Prediction: YOLO forecasts bounding boxes that might contain objects for every grid cell. Coordinates for the center, width, and height of each bounding box are used to represent it.*

3) *Objectness Score: YOLO predicts an "objectness" score for each bounding box, indicating the likelihood that an object is present in that box.*

4) *Class Prediction: For every bounding box, Yolo additionally forecasts the probability distribution over predefined classes. This indicates that in addition to detecting objects, the algorithm also groups them into distinct categories.*

5) *Multiple Scales: Yolo can handle objects of various sizes in the input image because it predicts bounding boxes and class probabilities at multiple scales.*

6) *Non-Maximum Suppression (NMS): Following the prediction phase, a post-processing technique known as NMS is utilized to remove redundant and uncertain bounding box predictions, retaining solely the most definite ones.*

4.2 YOLOv5

YOLOv5 is an improved version of its predecessors which are YOLOv4, v3, v2 and YOLO. YOLOv5 further improved the performance of the model, features such as hyperparameter optimization, integrated experiment tracking and automatic export to popular export formats are the reason for its better performance.

Architecture of YOLOv5

It is made of 3 major networks, they are called ad Head, Neck and Backbone of the model.

1) ***Backbone**: The backbone is the core of the network, responsible for extracting high-level features from the input image. YOLOv5 employs a modified version of the Darknet53 architecture, known as CSPDarknet53, which incorporates Cross-Stage Partial connections (CSP) to improve efficiency and reduce computational cost.*

2) ***Neck:*** *The neck acts as a bridge between the backbone and the head, responsible for combining and refining the feature maps extracted from the backbone. YOLOv5 utilizes a combination of SPPF (Spatial Pyramid Pooling) and PANet (Path Aggregation Network) to enhance feature fusion and multi-scale feature extraction.*
3) ***Head:*** *The head is responsible for generating the final object predictions. It receives the feature maps from the neck and applies a series of convolutional layers and detection layers to predict the presence of objects, their classes, and their bounding boxes.*

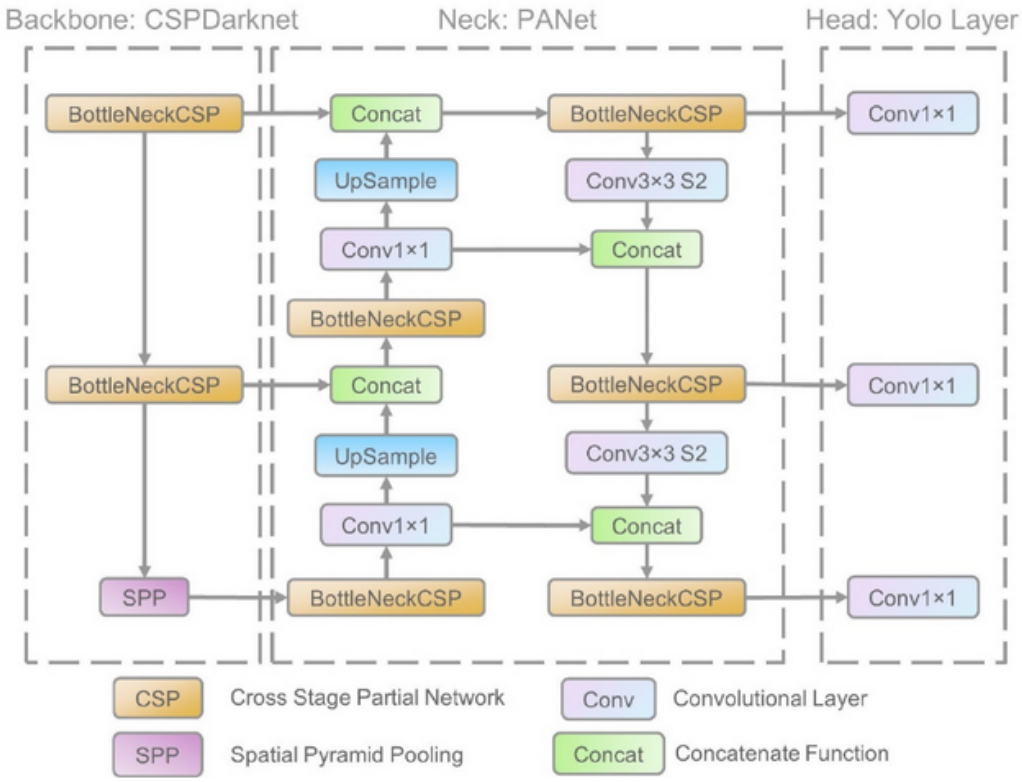


Figure 1 Architecture of YOLOv5 (from Ultralytics)

YOLOv5's architecture is carefully designed to achieve a balance between accuracy and speed, making it suitable for real-time object detection applications. Its efficient use of convolutional layers and feature fusion techniques contributes to its impressive performance.

4.3 YOLOv8

YOLOv8 is the latest version of YOLO by Ultralytics. As a cutting-edge, state-of-the-art (SOTA) model, YOLOv8 builds on the success of previous versions, introducing new features and improvements for enhanced performance, flexibility, and efficiency. YOLOv8 supports a full range of vision AI tasks, including detection, segmentation, pose estimation, tracking, and classification. This versatility allows users to leverage YOLOv8's capabilities across diverse applications and domains.

Architecture of YOLOv8

YOLOv8 builds upon previous versions of YOLO algorithms, it utilizes a convolutional neural network that is divided into two parts- the backbone and the head. A modified version of the CSPDarknet53 architecture forms the backbone of YOLOv8. This consists of 53 convolutional layers and employs cross stage partial connections to improve information flow between the different layers.

Multiple convolutional layers make up the head of YOLOv8, which is followed by several fully connected layers. For each object found in an image, these layers forecast bounding boxes, objectness scores, and class probabilities. The use of a self-attention mechanism in the network's head is one of YOLOv8's primary features. Through this mechanism, the model can focus on different areas of the image and modify the weighting of various features according to how relevant they are to the task at hand. The capability of YOLOv8 to carry out multi-scale object detection is another significant feature.

The model detects objects in an image with varying sizes and scales by using a feature pyramid network. This feature pyramid network consists of multiple layers that detect large and small objects within an image.

Key Architectural Improvements:

YOLOv8 introduces several architectural improvements over its predecessors:

Path Aggregation Network (PAN): The PAN module effectively aggregates features from different backbone stages, enhancing feature fusion and improving detection performance.

CyoBN (Cross-Stage Partial Connections with Batch Normalization): This modification to the CSP architecture further enhances feature flow and reduces computational cost.

Mish Activation Function: The Mish activation function replaces the commonly used ReLU activation, providing a smooth and non-monotonic non-linearity that improves model performance.

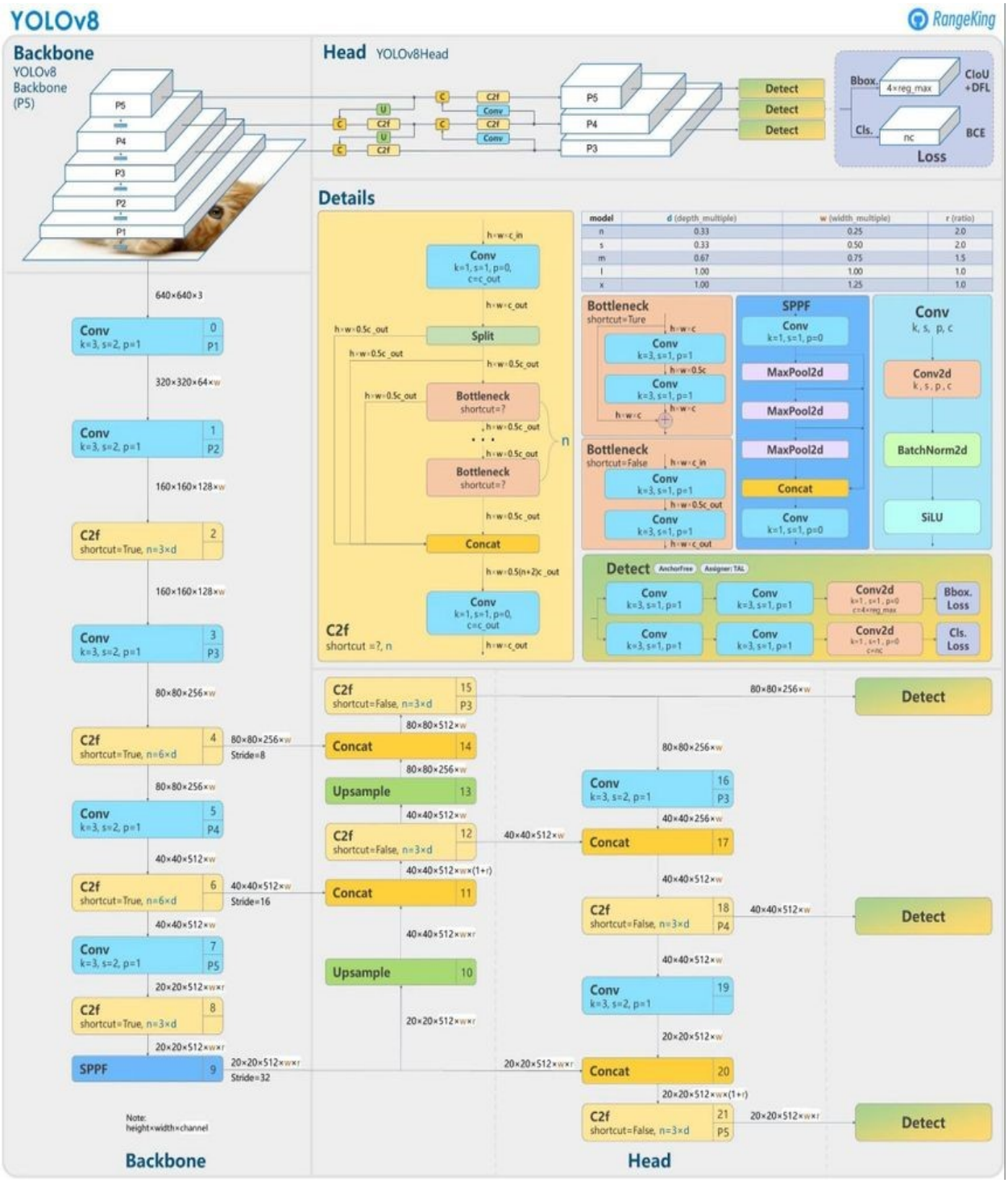


Figure 2Architecture of YOLOv8 (from ultralytics)

Dataset Preparation

Dataset for the project was prepared on roboflow. Roboflow is a platform where labeling of images is made easy. Around thousand images were taken from uditri.
We labeled the dataset and exported in yolo format. It creates folders of images and labels – each image will have .txt files which contains bounding box co-ord and a yaml file which contains information about class.

The images were divided into 3 parts test, train, valid.
700 images in train, 200 in valid, 100 in test.

Figure 3-Dataset Prep

Parameters that define the model:

The parameters defined are crucial for configuring the behavior of the YOLOv5,v8 model during the inference process. Let's go through each parameter and discuss its significance:

- *conf (NMS Confidence Threshold):*
  - *Significance: This parameter sets the confidence threshold for non-maximum suppression (NMS). During object detection, the model predicts bounding boxes along with confidence scores. NMS is applied to filter out redundant bounding boxes based on their confidence scores. Setting conf determines the minimum confidence score required for a prediction to be considered valid. We have taken 0.5.r*
- *iou (NMS IoU Threshold):*
  - *Significance: IoU (Intersection over Union) is a metric that measures the overlap between predicted bounding boxes. NMS uses the IoU threshold to decide which boxes to keep and which to discard. A higher iou threshold results in more overlapping boxes being suppressed, potentially leaving only the most confident and accurate predictions. We have kept it as 0.5.*
- *agnostic_nms (NMS Class-Agnostic):*
  - *Significance: When agnostic_nms is set to True, NMS is applied independently of the predicted class labels. In other words, if multiple bounding boxes of different classes overlap, they can still be suppressed based on their confidence scores and IoU, even if the classes are different. Setting it to False considers class information during NMS. Since we only have one class, we have set it as false.*
- *max_det (Maximum Detections):*
  - *Significance: This parameter limits the maximum number of detections that the model can output for a single image. If there are a large number of predictions, setting max_det can help control the number of results, ensuring that you only keep the top N predictions with the highest confidence scores. This is useful for managing the output in scenarios where you want to focus on a certain number of the most confident predictions. We have kept it as 7.*

These parameters allow you to fine-tune the behavior of the YOLOv8, v5 models based on your specific requirements and the characteristics of your dataset. Adjusting these parameters can impact the precision, recall, and overall performance of the object detection system. It's often necessary to experiment with different values to find the optimal configuration for your use case.

Chapter 5

Results

5.1 Results

In the context of YOLO (You Only Look Once), "MAP" typically refers to "Mean Average Precision." Average Precision (AP) is a common metric used to evaluate the performance of object detection algorithms and Mean Average Precision (mAP) is the average AP across multiple object classes. Other than that confusion matrix is also used to evaluate the model. To calculate mAP we need precision, recall.

Brief explanation of terms used to evaluate the model.

1. ***Confusion Matrix**:*
   - 1.1. *True Positives (TP): The model correctly predicts the presence of an object.*
   - 1.2. *False Positives (FP): The model predicts the presence of an object when there is none.*
   - 1.3. *True Negatives (TN): The model correctly predicts the absence of an object.*
   - 1.4. *False Negatives (FN): The model predicts the absence of an object when there is one.*
2. ***Precision:** Precision is the ratio of true positives to the total number of positive predictions. It measures the accuracy of positive predictions.*

$$P = \frac{TP}{TP + FP}$$

3. ***Recall** (Sensitivity or True Positive Rate): Recall is the ratio of true positives to the total number of actual positives. It measures the ability of the model to capture all the relevant instances.*

$$R = \frac{TP}{TP + FN}$$

4. ***Intersection over Union (IoU)**: IoU is a measure of the overlap between the predicted bounding box and the ground truth bounding box. It is often used to determine whether a detection is considered correct.*

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

5. ***Average Precision (AP):*** *AP is a metric used to measure the accuracy of object detectors. It considers both the precision and the recall of a model.*

$$AP = \frac{Sum\ of\ max(P) values}{Number\ of\ R\ levels}$$

6. ***Mean Average Precision (mAP)****: mAP is the mean of the AP values calculated for each class. It provides an overall performance measure across all object classes.*

$$mAP = \frac{\sum_{i=1}^{N} APi}{N}$$

N is number of object class.

7. ***F1 Score****: F1 Score is the harmonic mean of precision and recall. It provides a balance between precision and recall.*

$$F1 = 2\frac{PXR}{P+R}$$

Sample Images

Figure-4

Figure – 5

YOLOv5:

F1 Confidence Curve: This curve can provide insights into how the F1 score changes as you adjust the confidence threshold. It helps you understand the trade-off between precision and recall at different confidence levels.

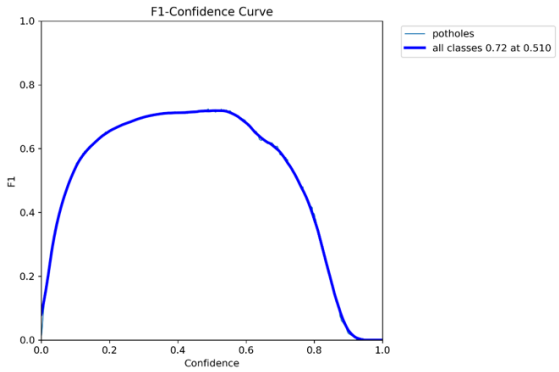


Figure-6 F1 Confidence curve

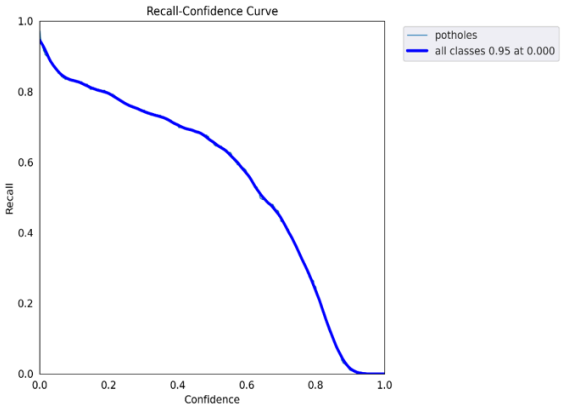


Figure -7 Recall confidence curve

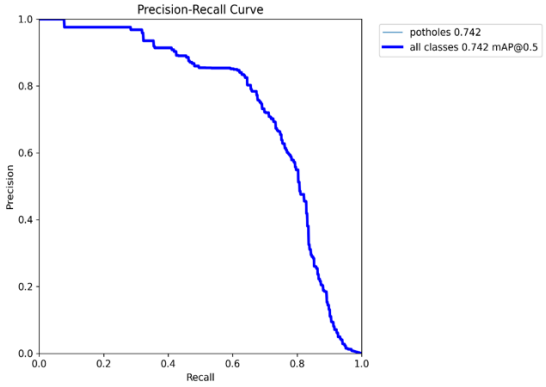


Figure 8 PR_Curve

Figure -9 Validation batch

Predicted Results

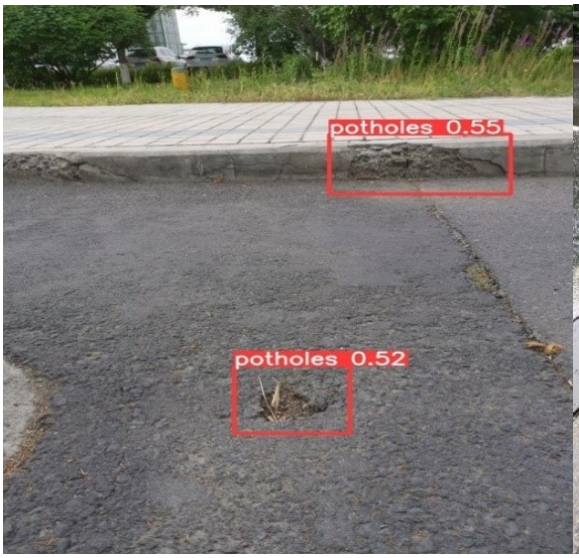


Figure -10

Figure-11

Confusion Matrix

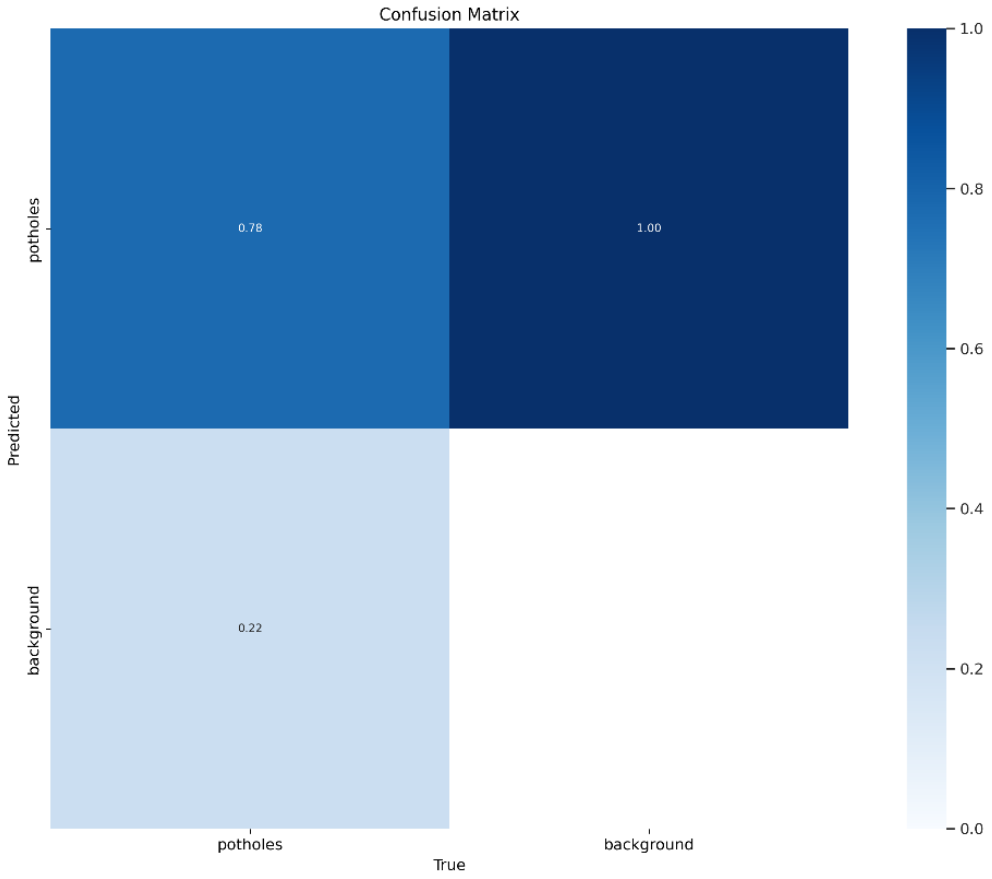


Figure-12

YOLOv8

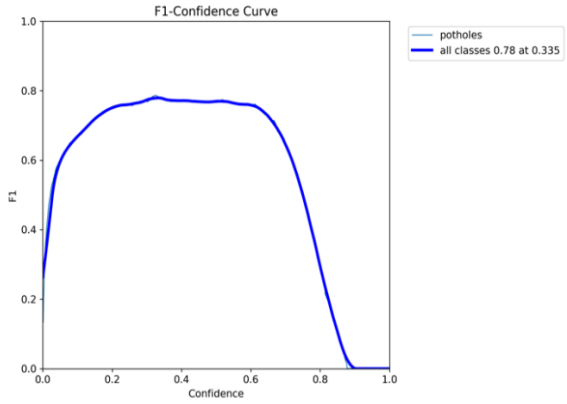


Figure -13 F1 Confidence Curve

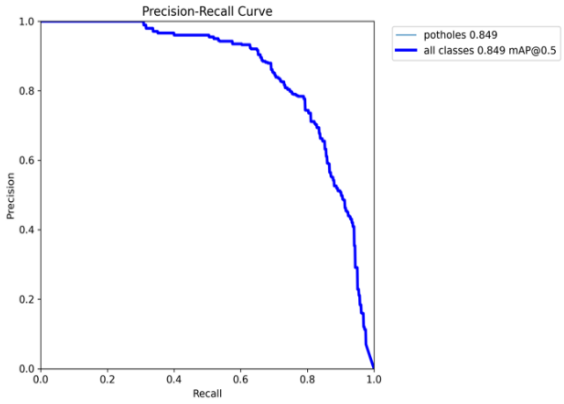


Figure-14 PR Curve

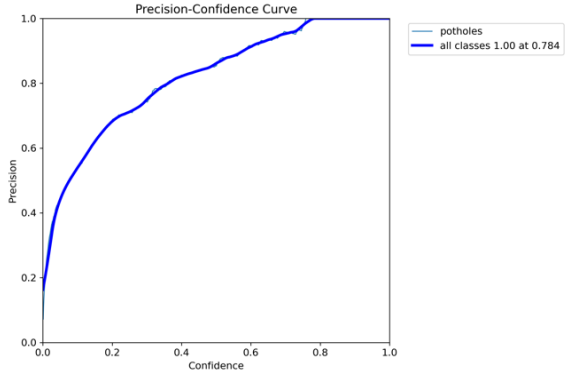


Figure-15 P_Curve

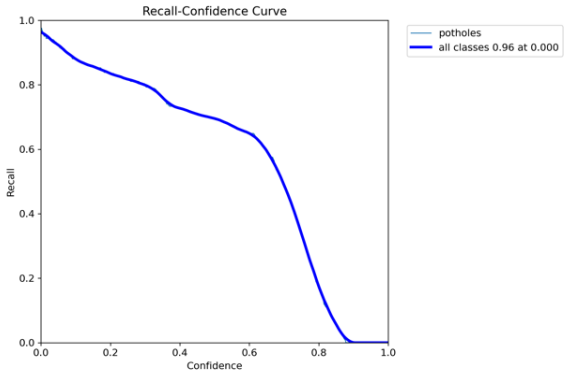


Figure-16 R_Curve

Confusion Matrix

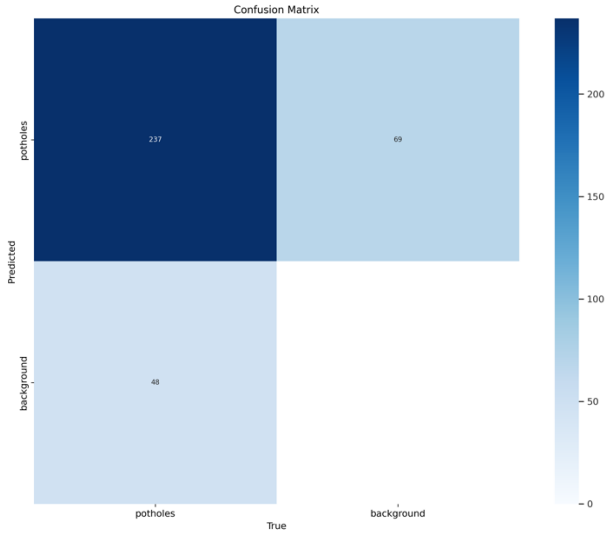


Figure-17

Normalized Confusion Matrix

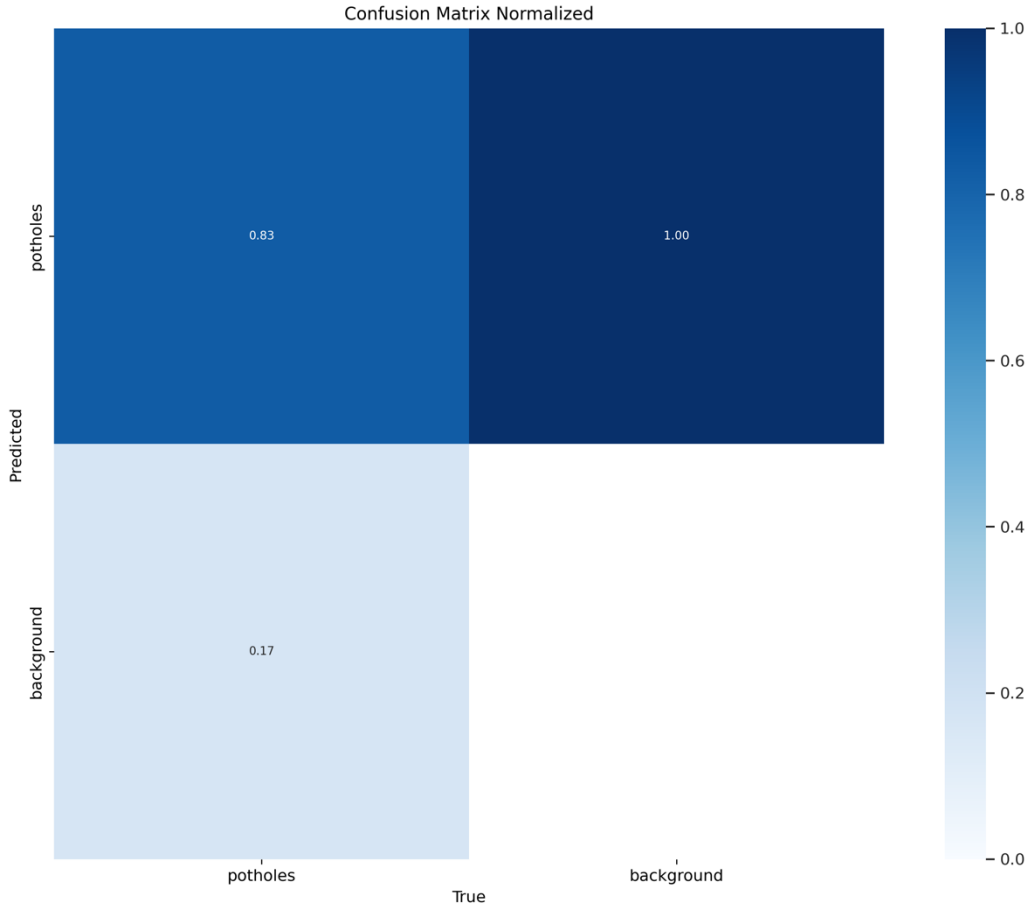


Figure-18

Validation Results

Figure-19

Predicted Results

Figure-20

Figure-21

Figure-22

Model	YOLOv5	YOLOv8

Precision Score	0.812	0.859
Recall Score	0.811	0.821
mAP50 Score	0.862	0.899
mAP(50-95) Score	0.475	0.489
Speed	15ms per image	7ms per image
Size	56.7mb	22.8mb

Table-2

5.2 Discussion

The formulas mentioned in section 5.1 are used to plot the graphs for both models. YOLOv5 model gives normalized confusion matrix as one of the outputs during training (), v8 model gives both confusion () and normalized confusion matrix (). Looking at the matrix we can say that v8 model is better.

Figure-10, which is one of the predicted outputs of v5, we can see that model predicts an irregularity in footpath as pothole whereas in Figure-21, which is a predicted output from v8 model, we can see that even though there are irregularities in the road it predicts pothole accurately. From this we can conclude that yolov8 has greater accuracy.

Table-2 gives evaluation parameter values for each model. For real time application models mAP50 score should be as close to 1 as possible and mAP(50-95) score must be close to 0.5, meaning closer the value is to 1 greater is the accuracy of the model. Looking at the table we can say that YOLOv8 is faster smaller and more accurate.

Chapter 6

Conclusion

6.1 Conclusion

In conclusion, the utilization of YOLOv5 and YOLOv8 for pothole detection in self-driving cars represents a significant stride towards enhancing the safety, efficiency, and overall performance of autonomous transportation systems. The deployment of these advanced object detection algorithms addresses the critical challenge of identifying and localizing potholes in real-time, contributing to a safer and more comfortable driving experience.

YOLOv5, with its real-time capabilities and improved accuracy, offers a robust solution for pothole detection. The model's ability to process entire images in a single pass allows for rapid decision-making, crucial for autonomous vehicles navigating dynamic road conditions. Additionally, YOLOv5's PyTorch implementation facilitates ease of use and integration within the existing software infrastructure of self-driving cars.

The introduction of YOLOv8 builds upon the strengths of its predecessors, incorporating enhancements in accuracy, speed, and model architecture. YOLOv8, with its focus on achieving superior performance, offers a compelling option for pothole detection systems. Its adaptability and efficiency make it well-suited for the complex demands of self-driving car applications, providing a robust foundation for addressing the challenges posed by varying road conditions and potential hazards.

6.2 Future Scope

As we continue to advance in the realm of autonomous transportation, the integration of cutting-edge technologies, exemplified by YOLOv5 and YOLOv8, showcases the potential for transformative solutions. The intersection of computer vision, machine learning, and self-driving technology opens new avenues for innovation, ensuring that self-driving cars can navigate roads with heightened awareness, adaptability, and responsiveness, ultimately fostering a safer and more reliable autonomous driving experience.

As YOLOv8 is also capable of segmentation we can further enhance the visual effects of the model as representing a pothole in different color along its boundaries gives better idea for the driver as to pothole's position and size than a just a bounding box.

In essence, the future scope for pothole detection in self-driving cars involves a multidisciplinary approach, incorporating advancements in sensor technology, machine learning, connectivity, and collaborative urban planning. These developments aim to

further enhance the safety, efficiency, and overall effectiveness of autonomous transportation systems on a global scale.

Reference:

[1] Oche Alexander Egaji, Gareth Evans, Mark Graham Griffiths, Gregory Islas, Real-time machine learning-based approach for pothole detection, Expert Systems with Applications, Volume 184, 2021,115562, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2021.115562

[2] K. P. Rani, M. Arshad and A. Sangeetha, "Pothole Detection Using YOLO (You Only Look Once) Algorithm," 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC), Bhubaneswar, India, 2022, pp. 1-6, Doi: 10.1109/ASSIC55218.2022.10088357
z

[3] Anas Al Shaghouri, Rami Alkhatib, Samir Berjaoui, “Real-Time Pothole Detection Using Deep Learning”, https://doi.org/10.48550/arXiv.2107.0635

[4] Sung-Sik Park, Van-Than Tran, Dong-Eun Lee, "Application of Various YOLO Models for Computer Vision-Based Real-Time Pothole Detection"