Autonomous Object Detection Dataset: A Study on Bangladeshi Roadways

by

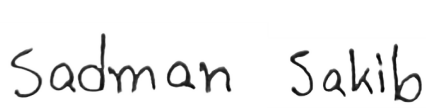
Kazi Sadman Sakib
19101125
Tahsinul Islam Abir
19301166
Shahida Sultana Kareena
20101306
Rupanti Alam
20101090

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
October 2024

Declaration

It is hereby declared that
1.The thesis submitted is the author's original work completed while pursuing a degree at Brac University.
2. The thesis does not include any previously published or written by a third party, unless it is properly cited through full and accurate referencing.
3. The thesis does not include material that has been accepted or submitted. for any other degree or credential from a university or another institution.
4. I/We acknowledged all major sources of assistance.



Kazi Sadman Sakib
19101125

Rupanti Alam
20101090

Tahsinul Islam Abir
19301166

Shahida Sultana Kareena
20101306

Approval

The thesis/project titled "Autonomous Object Detection Dataset: A Study on Bangladeshi Roadways" submitted by

1. Kazi Sadman Sakib (19101125)
2. Rupanti Alam (20101090)
3. Tahsinul Islam Abir (19301166)
4. Shahida Sultana Kareena (20101306)

Examining Committee:

Supervisor:
(Member)

Dr. Md. Ashraful Alam

Associate Professor
Department of Computer Science and Engineering
BRAC University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
BRAC University

Department Head:
(Chair)

Sadia Hamid Kazi

Associate Professor and Chairperson of
Department of Computer Science and Engineering
BRAC University

Ethics Statement

his thesis was conducted in adherence to ethical principles and guidelines to ensure the integrity and ethical responsibility of our research. All efforts were made to respect the rights, dignity, and welfare of all participants involved in the study.

Abstract

This research aims to enhance object detection in autonomous driving by testing the three new advanced YOLO models, namely, YOLOv5, YOLOv7, and YOLOv8, on a private dataset proposed specifically for Bangladesh. It is suggested that the dataset include conventional vehicles such as cars, trucks, and buses, non-conventional modes of transport such as rickshaws and CNGs, and complex scenes with poor visibility or high congestion. The contribution of this research involves the creation of a custom dataset that overcomes the deficiencies in other publicly available datasets in terms of improved representation for unconventional vehicles and managing different traffic and weather conditions. YOLOv5, YOLOv7, and YOLOv8 are trained and tested by comparing performances concerning mean Average Precision (mAP), precision, recall, and inference speed. YOLOv8 was the best performer in different difficult conditions related to traffic and even challenge lighting environments, while YOLOv5 was observed to perform more efficiently in the case of real-time applications where the resources are constrained. YOLOv7 strikes a good balance between speed and accuracy, hence is suitable for moderately complex environments. However, all the models were vulnerable to class imbalance and the detection of smaller or occluded objects. Comparing these models with their predecessors, like Faster R-CNN, testifies that the YOLO models perform the best in real time among other models. In general, but especially for YOLOv8, more improvements should be made on dataset representation, occlusion handling, and edge optimization if it is to find a place in autonomous driving systems in the near future. Hence, the research relies on a custom-built private dataset, which effectively captures the dynamic and unstructured environment of traffic in Bangladesh in much finer detail compared to other, earlier datasets.

Keywords: Autonomous Driving, Object Detection, YOLOv5, YOLOv7, YOLOv8, Bangladeshi traffic, BADODD dataset, Real-Time Detection, Deep Learning, Computer Vision, Class Imbalance, Model benchmarking.

Dedication

This thesis is dedicated to our families and friends, whose constant support, encouragement, and love have made this journey possible. Thank you for believing in us every step of the way.

Acknowledgement

Firstly, all praise to the Great Allah for whom our thesis have been completed without any major interruption.
Secondly, to our supervisor Dr.Ashraful Alam sir for his kind support and advice in our work. He helped us whenever we needed help.
And finally to our parents without their throughout sup-port it may not be possible. With their kind support and prayer we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Nomenclature

The next list describes several symbols & abbreviation that will be later used within the body of the document

ϵ	Epsilon
υ	Upsilon

Chapter 1

Introduction

1.1 Introduction

The concept of autonomous driving has gained rapid progress and is transforming the way in which vehicles communicate with the immediate environment to better facilitate more efficient, safe, and reliable transportation networks. This evolvement rests on the ability to actually observe and understand the outside environment of one's eyes; hence comes object detection technology. Object detection is called forth in turn to enable the self-driving car to make its way around safely by detecting other moving objects, pedestrians, traffic signs, and barriers in real time. In this respect, deep learning models, and particularly YOLO, have grown to be preeminent because of their swiftness and accuracy in object detection tasks [11]. Unique traffic scenarios in regions like Bangladesh call for new challenges that demand thorough assessment and contextual benchmarking of these models. The rapid growth of motor vehicle, and consequent roadway congestion, has led to safety and traffic accident problems throughout the world [2]. With traditional vehicles like rickshaws, CNGs and vans blended in with conventional cars, trucks and buses, Bangladesh is a particularly tough landscape in which to develop autonomous driving systems. Additionally, fog, precipitation and decreased lighting conditions only make the situation worse with respect to object detection in this area. Existing object identification algorithms, like YOLO are effective in usual cases, but need adjustment and evaluation in specific road environment of Bangladesh to determine their applicability and dependability to real issue. Effective and real time object identifier is of great importance for the progress of autonomous driving systems. Manually engineered features were used in initial models and did not allow models to adapt to different situations and traffic conditions [8]. However, YOLO, a kind of deep learning method, that uses neural networks to extract object features automatically, has overcome these limitations so that it can handle massive datasets and object identification at faster speed with greater accuracy. Since its introduction in 2016, the YOLO model has emerged as a golden standard for fast real time object detection, doing object predictions on complete images in just one pass and thus performs extremely faster than Fast R-CNN or SSD (Single Shot MultiBox Detector) [7]. However, despite these gains, YOLO struggles to achieve its efficacy, mainly in the identification of small, closely located objects common in crowded urban environments such as those in Bangladesh [19]. The goals of this study are to solve these difficulties by testing 2 iterations of YOLO model, YOLOv5 and YOLOv8 on a custom dataset that specifically captures

the unique traffic details of Bangladesh or private dataset. This dataset represents the incomparably complicated Bangladesh roadways, from varying classes of vehicles and traffic conditions. traditionally represented vehicles like cars and buses, and less frequently shown in global datasets used for training autonomous systems, BADODD includes local transportation vectors such as rickshaws and CNGs. We assess the generalization capabilities of YOLO models using a peculiar dataset to traffic in a environment where non standard vehicles are present and the environment is undesirable for driving, e.g. low visibility and poor lighting. In this analysis we will be able to describe the flexibility and performance of YOLOv5 and YOLOv8 toward object detection in real world scenarios.

Object detection models, particularly for autonomous driving, must satisfy two essential criteria: with high precision and low latency. Given their robustness in providing quick detections, YOLO models are very appropriate for real-time applications in autonomous cars, because response times need to be short [25]. However, accuracy remains a challenge, especially in detecting small objects, or with occlusions such as occurs in the crowded, unstructured traffic conditions typical of Bangladesh. Due to their small size and erratic movement patterns, rickshaws and pedestrians are a big hurdle for object detection algorithms, even when they dwell with motorized vehicles [18]. On the other hand, Bangladesh's roadways experience a substantial volume of traffic during peak hours, making this further challenging, and models need to precisely differentiate and label objects amongst traffic that is frequently close together. And YOLOv8 and YOLOv5 both have the value to be used in object detection application in autonomous driving. Real time detection tasks in resource constrained situations have been widely used YOLOv5, that has its lightweight architecture and efficiency. If we're talking about embedded devices (e.g. low power devices like that in an autonomous car), YOLOv5 is a nice solution, as it has low latency and it's quick. On the contrary, YOLOv8, the latest improvement on the YOLO series, is a version against which numerous improvements made to enhance detection accuracy for poor weather, fog, rain or even low light are incorporated. All of these changes added to the robustness of YOLOv8 to the environmental fluctuations common to the Bangladeshi roadway. We aim to compare the performance of YOLOv5 and YOLOv8 on the custom dataset to see if model solves the problem of a balance between speed and accuracy concerning the object detection in Bangladesh traffic situation. Essential performance parameters, including precision, recall, and mean average precision (mAP) will be evaluated under various traffic circumstances including daytime, midnight and foggy conditions [22]. In this research will also study how each model detects non standard vehicles, such as rickshaws and CNGs, that are prevalent in Bangladesh yet completely absent from all worldwide object identification datasets. The aim of this research is to reconcile the heavily processable generalized object detection models with the unique needs of autonomous driving in Bangladesh. In this work, we will evaluate YOLOv5 and YOLOv8 on a region based data set for improvements in the models based on real world application of these models for autonomous driving specially in the region where traffic is un structured. It is expected that the results of this research contribute to the development of more accurate and more effective object identification models connected to self driving cars, not only in Bangladesh but countries struggling with the same traffic problem.

1.2 Problem Statement

Deep learning, a subset of Machine Learning (ML) and Artificial Intelligence (AI), has markedly progressed the domain of computer vision by facilitating the automatic extraction of characteristics and patterns from unprocessed data, such as photographs [4]. Convolutional Neural Networks (CNNs) have emerged as the benchmark for object identification and recognition, particularly in autonomous vehicles that necessitate an extensive comprehension of their surroundings. Autonomous vehicles must identify diverse items on the roadway to navigate securely, including other vehicles, pedestrians, cyclists, and barriers. Although picture data offers extensive information conducive to object recognition, the projection of a 3D scene onto a 2D imaging sensor leads to the forfeiture of depth information, complicating the precise estimation of the positions of identified objects. Single-stage and two-stage detectors are prevalent architectures in 2D object identification; nonetheless, they encounter constraints in intricate real-world scenarios [9].
Numerous perceptual complexities necessitate additional investigation to improve object recognition efficacy in autonomous driving [41]:

• Partial View and Angled View: Objects may present distinct appearances from different perspectives, influencing detection precision.
• Obscured Objects: Objects may be partially or entirely concealed by other objects, impeding identification efforts [4(p-6)]. In autonomous driving scenarios, objects may be obscured by stationary structures such as lampposts or by other moving cars [3].
• High-density traffic scenarios present multiple items concurrently, hence elevating computational requirements [41].
• Similar Objects at Varying Distances: Objects belonging to the same category may exhibit different sizes based on their distance, confounding recognition algorithms [25].
• Diverse Lighting Conditions: Illumination profoundly influences object detection. Cameras frequently provide substandard images in low-light environments, diminishing detection precision [15].
• Inclement Weather Conditions: Rain, fog, and snow impair visual signals, thus impacting object identification efficacy [16].
• Invariance Challenges: Transformations include translation, rotation, and scaling might hinder object recognition, referred to as the "invariance problem" [14].
• Detection of Small Objects: Small objects pose challenges for detection due to the difficulty in differentiating them from the backdrop and analogous objects. Research on the detection of small objects is rather scarce relative to larger things [12].

Although deep learning models like YOLO (You Only Look Once) demonstrate exceptional accuracy and low error rates on datasets such as Pascal VOC and ImageNet, attaining real-time performance with minimal mistake rates in autonomous driving continues to be challenging [33]. The velocity of these architectures is vital for real-time applications where rapid decision-making is needed [41]. In areas such as Bangladesh, these issues are intensified by atypical vehicles (e.g., rickshaws, CNGs, vans) and disorganized traffic patterns, complicating object detection tasks. Consequently, there is an urgent necessity to improve object detection models to successfully tackle these issues. The emphasis must be on enhancing precision in identifying occluded objects, managing fluctuating lighting circumstances, recogniz-

ing small objects, and functioning under severe weather conditions, all while meeting real-time processing demands. Evaluating sophisticated models such as YOLOv5 and YOLOv8 on region-specific datasets, like the Bangladeshi Autonomous Driving Object Detection Dataset (BADODD), is crucial for creating resilient autonomous vision systems adapted to the distinct conditions of Bangladesh's roadways.

1.3 Aims and Objective

The major aim of the research is to improve object detection in autonomous vehicles by checking the efficiency of the YOLOv5 and YOLOv8 models on a Bangladeshi Autonomous Driving Object Detection Dataset BADODD and a private dataset designed specifically for the research. The present study explores the challenges imposed by the unique and complex road environment of Bangladesh, a country that presents a very specific profile of uncontrolled traffic, a high proportion of non-motorized vehicles like rickshaws, vans, and CNGs, to name a few, with various environmental conditions such as fog, rain, low visibility, and congested roads. Custom data is done in a way that the limitations, especially class imbalance and rare vehicle-type information and BADODD, are kept in mind to make it more representative of diverse traffic and weather conditions of Bangladesh. It was designed to improve object detection in real-time, giving emphasis to scenarios that most of the global databases often disregard. In this respect, the approach pursued will seek verification in these scenarios regarding the application of deep learning techniques for object detection and, in particular, using the detectors YOLOv5 and YOLOv8 for identification in such a setting with small and fast-moving objects from a distance, considering different weather conditions and levels of illumination. The secondary aim is to study these models in comparison with each other on the BADODD and custom datasets, in light of their capability to handle abnormal vehicles and the unique challenge of streets in Bangladesh. This will be achieved by checking the performance of detection, speed, and robustness under different driving conditions. It enhances the models' adaptability to adverse weather conditions and complex situations, hence robustness and accuracy in real-life driving applications. Accordingly, practical recommendations based on such assessment will be given to tune the models for better performance with respect to Bangladesh-specific needs for driving. The research work will definitely enrich the autonomous vehicle technology and object detection system in places where unconventional traffic patterns with abrupt environmental challenges prevail, such as Bangladesh.

1.4 Research Motivation

This research is motivated by the increasing demand for dependable and precise object detection systems in autonomous cars, especially in areas with atypical traffic patterns and difficult environmental circumstances, such as Bangladesh. Although advanced models such as YOLOv5 and YOLOv8 demonstrate significant potential in worldwide applications, they frequently neglect the intricacies of road conditions in developing nations, where unconventional vehicles like rickshaws, CNGs, and vans prevail. Moreover, elements such as inadequate road infrastructure, recurrent in-

clement weather, and disorganized traffic provide considerable obstacles for current object detection methods. This project aims to connect generalized object detection methods with the specific requirements of Bangladesh's road ecology. This study seeks to enhance real-time detection accuracy and robustness in complicated circumstances by testing these models on the custom dataset, thereby leading to safer and more efficient autonomous driving systems. The primary objective is to create solutions that improve the reliability of autonomous vehicles, rendering them feasible in various worldwide contexts, particularly in areas with atypical traffic and environmental conditions.

Chapter 2

Related Work

2.1 Background Study

Autonomous vehicles are set to transform transportation, providing advantages such as improved safety, less traffic congestion, and heightened travel efficiency. Prominent corporations such as Tesla, Waymo, and Uber are leading the advancement of this technology, with forecasts indicating that completely autonomous vehicles may materialize within the next 15 to 20 years [59]. Nonetheless, considerable obstacles persist, especially in guaranteeing that these vehicles can precisely detect their environment in real-time, an essential prerequisite for safe and dependable autonomous driving. A primary function of perception is object detection, which entails recognizing and categorizing many items on the roadway, including automobiles, pedestrians, and barriers. Attaining elevated precision and rapidity in object identification is crucial for the efficient functioning of autonomous vehicles. Recent breakthroughs in deep learning, especially Convolutional Neural Networks (CNNs), have propelled substantial progress in object detection. CNN-based models have emerged as the favored method for object detection, exceeding older techniques in accuracy, adaptability, and efficiency [22]. These deep learning models have demonstrated efficacy in object detection across many driving circumstances, rendering them essential for autonomous driving systems. A significant issue remains in identifying occluded items—those partially concealed by other objects or environmental factors. This represents a considerable constraint for existing models, which frequently have difficulties in sustaining high detection accuracy in these circumstances [48].
There are two principal categories of deep learning-based object identification architectures: single-stage and two-stage detectors. Single-stage detectors, like YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector), are recognized for their rapidity and efficacy. These models forecast bounding boxes for object identification in a single traversal of the network, rendering them optimal for real-time applications such as autonomous driving, where swift decision-making is essential. [30] Conversely, double-stage detectors, such as Faster-RCNN, function in two phases: initially producing region proposals and subsequently categorizing and refining bounding boxes. Although double-stage detectors generally exhibit more accuracy, they are slower and less appropriate for real-time applications. Notwithstanding these advancements, contemporary object detection models encounter considerable obstacles in complex, real-world settings, especially in areas such as Bangladesh, where atypical traffic patterns, non-standard vehicles (e.g., rickshaws, CNGs), and

adverse weather conditions hinder detection efforts. This study aims to fill these gaps by examining the efficacy of YOLO models, particularly YOLOv5 and YOLOv8, utilizing the Bangladeshi Autonomous Driving Object Detection Dataset (BADODD). This dataset is designed to address the specific issues of Bangladesh's road ecosystem, facilitating a targeted investigation of the optimization of deep learning models for optimal performance in these conditions.

2.2 Literature Review

Object detection is essential in autonomous driving, as cars must accurately sense their environment in real-time to ensure safe navigation. Conventional vehicle recognition algorithms utilized manual feature extraction methods, including the background update method, frame difference method, and optical flow method, to identify moving vehicles [39][[]Nonetheless, these methodologies frequently encountered obstacles such as elevated processing expenses, susceptibility to external variables like illumination, and insufficient resilience in practical applications [52]. Recently, deep learning methodologies, particularly Convolutional Neural Networks (CNNs), have markedly enhanced the precision and velocity of object identification in autonomous cars. Significant breakthroughs in CNN-based object detection include region-based methodologies such as R-CNN, which pioneered the generation of candidate areas for object recognition, subsequently followed by classification [5]. The two-stage method was further refined by Fast R-CNN, which increased computational efficiency, and Faster R-CNN, which substituted the conventional selective search algorithm with a Region Proposal Network (RPN) [6]. Despite their great accuracy, these two-stage algorithms are computationally demanding and therefore inappropriate for real-time applications such as autonomous driving. Single-stage detectors such as YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector) were developed to overcome the speed constraints of two-stage detectors. YOLO analyzes the complete image in one iteration, concurrently anticipating object categories and bounding boxes, therefore markedly improving detection speed [10]. Nonetheless, initial iterations of YOLO encountered difficulties in identifying small objects and achieving accurate localization. Since its inception in 2015, the YOLO algorithm has undergone several enhancements, including the creation of YOLOv4, YOLOv5, and YOLOv8, which include sophisticated methodologies such as Feature Pyramid Networks (FPN) and optimized loss functions to enhance precision and manage intricate situations.In the realm of autonomous driving, object detection systems must function in extremely dynamic and unexpected surroundings. Research indicates that YOLO-based algorithms rank among the most effective for real-time applications, with iterations such as YOLOv5 and YOLOv8 attaining exceptional accuracy and speed on datasets like MS COCO [45]. Although processing economy and detection precision are balanced in YOLOv5, this is a feature that makes it perfect for embedded systems in autonomous cars. Although these models can work in well controlled conditions, it largely fails in locations where there are unorthodox traffic system like Bangladesh because they lack features for that. However, obstacles are posed by objects like rickshaws and CNGs which comprise such non standard vehicles along with poor road infrastructure, to object detection models [22]. As a result, these models need to be evaluated using datasets specifically geared towards such contexts but to then fully understand their performances

in the real world one needs to evaluate them on datasets such as the Bangladeshi Autonomous Driving Object Detection Dataset (BADODD).
Bangladeshi Autonomous Driving Object Detection Dataset, a dataset collected from 9 districts of Bangladesh to fill up these deficiencies in existing autonomous driving datasets. Using data from smartphone cameras only, realistic real world road conditions and scenarios, such as varying lighting, traffic, and so forth, are taken into consideration. Based on characteristics rather than local vehicle names, the vehicle classification system is scalable for better object detection in Bangladeshi roads peculiar traffic scenario.
Over the last few years, research has demonstrated that occluded object identification is essential to autonomous driving, where an object is blocked from view by other cars or even barriers. However, conventional non maximal suppression (NMS) techniques often fail to correctly pinpoint such items and thus produce wrong detections and missed detections. To address this challenge, ideas such as softNMS and multi scale feature extraction are proposed to improve detection effectiveness, particularly for small and poorly observed objects [51]. Although two stage detectors like Faster R-CNN are still advantageous when more accuracy is required, its high cost in terms of speed makes it unsuitable in context where speed is particularly important such as mobile phone applications. Faster R-CNN uses a Region Proposal Network (RPN) to generate candidate regions followed by localization refinement for classification [6]. While slower than YOLO, this approach produces better results in terms of identifying occluded and small objects due to the two step refinement in it. However, it is infeasible due to the processing cost for real time applications such as autonomous driving.
Real time detection has been made more practical by amalgamating the most sophisticated object detection models with hardware accelerators such as NVIDIA's Jetson AGX Xavier. NVIDIA's TensorRT and DeepStream SDK are instruments to realize efficient inference for deep learning models, reducing latency and enhancing performance [1]. Integration of hardware and software optimizations is critical for autonomous driving system that can cope up with harsh challenging conditions in real word, and deep learning based object recognition models such as YOLO and Faster R-CNN are now quite popular but are little researched on for solving the object recognition problems in tricky, real world conditions like a highway in Bangladesh. Developed by the authors is a Bangladeshi Autonomous Driving Object Detection Dataset (BADODD) for assessing these models in challenging traffic situations such as unconventional vehicles and erratic weather situations. The main contribution of this research is to compare the performances of YOLOv5, YOLOv7 and YOLOv8 on BADODD and benchmark YOLOv5, YOLOv7 and YOLOv8 to understand their weaknesses and strengths for optimisation of these models in the real time autonomy driving applications.
Object detection is an essential element of autonomous driving, allowing vehicles to see their environment in real-time and make precise decisions regarding navigation, safety, and traffic management. Conventional techniques for vehicle detection, including the background update approach, frame difference method, and optical flow method, depended on human feature extraction to recognize and categorize moving vehicles in video sequences [39]. These initial techniques faced considerable obstacles because to elevated processing expenses, susceptibility to external variables such as illumination and vehicular velocity, and their inadequacy in managing intricate

real-world situations with high precision and resilience [52]. Therefore, conventional vehicle detection algorithms suffered from their practical limits, particularly in dynamic and unpredictable high ways or urban traffic systems.

To address the constraints of two-stage detectors in real-time applications, single-stage detectors such as YOLO (You Only Look Once) and SSD (Single Shot Multi-Box Detector) were created. Single-stage detectors forecast both the object's categorization and bounding box in a singular traversal of the neural network, hence obviating the necessity for distinct region suggestions and classification phases [45]. YOLO, announced in 2015, represented a substantial advancement in object identification speed, capable of processing images at 45 frames per second (fps) in real time, rendering it suitable for autonomous driving applications that require swift response times. YOLO's principal innovation was its capacity to approach object detection as a regression problem, directly forecasting bounding boxes and object classes from complete photos. Nonetheless, initial iterations of YOLO encountered difficulties in identifying small objects and precisely localizing items, especially in congested or intricate environments [10]. Subsequent iterations of YOLO, such as YOLOv4, YOLOv5, and YOLOv8, implemented numerous enhancements to rectify these deficiencies. YOLOv4 used sophisticated methodologies such as Feature Pyramid Networks (FPN) for multi-scale detection and introduced a novel loss function termed CIoU_Loss, enhancing the precision of bounding box predictions. Although not formally considered a successor by the creators of YOLOv5, the approach has gained a lot of attention due to its efficient architecture as well as the fact that it performs better on multiple benchmarks. With YOLOv8, most recent iteration, detection precision and resilience were improved significantly especially in challenging condition like inclement weather, poor illumination, and high population density cities [22]. YOL models have become one of the most used for real-time object recognition in autonomous driving because the enhancements. Despite these advances; contemporary object detection techniques, in particular YOLO, can be problematic when they are deployed in environments that differ from typical traffic systems, for example in Bangladesh. Due to a prevalence of non-standard vehicle types such as rickshaws, CNGs, and vans and the disorganized traffic patterns, appearance of road infrastructure in Bangladesh is quite different compared to developed countries. Ultimately, these models trained on global datasets are seldom evaluated in region specific settings, and these situations are underrepresented within global datasets used for training object detection models [22]. To study the optimization of deep learning models for these two specific problems, we provide a dataset called the Bangladeshi Autonomous Driving Object Detection Dataset (BADODD) that addresses these problems specifically. Occluded objects—objects partially or totally occluded by other objects or ambient conditions—are a major challenge in autonomous driving. Ensuring safety and dependability of autonomous cars is particularly important for their point of partial occlusion in urban scenarios when vehicles or pedestrians can be obstructed by other vehicles, structures, or nature. Many current object detection models struggle with occluded objects using conventional non maximal suppression (NMS) methods utilized in these models, with the result being often not seeing the object or seeing the object but registering an erroneous positive. Luckily, to solve this problem, various methods were proposed to improve the accuracy of occluded object recognition for small and contiguous objects [51].

A second significant area of research is the development of lightweight object detec-

tion models that operate well on edge devices. In applications where inference speed is essential, like autonomous driving, vehicles must make instant judgments about objects that they identify. We show that YOLO based models achieve promising performance on embedded hardware like NVIDIA's JetsonAGX Xavier, designed for edge computing scenarios. The detection accuracy of these models is achievable in real world implementation with optimizations such as NVIDIA's TensorRT and DeepStream SDK offering accelerated inference times [1]. Recent studies have investigated the amalgamation of CNN-based object detectors with tracking algorithms, like DeepSORT and NvDCF, to enhance the identification and tracking of dynamic objects in autonomous driving contexts. These trackers utilize the bounding boxes supplied by object detectors to preserve object identity over video frames, even amid partial occlusion or alterations in object appearance. DeepSORT improves conventional tracking algorithms by integrating appearance information, allowing it to monitor objects throughout extended occlusion durations with reduced identity switches [30]. NvDCF, developed by NVIDIA, utilizes Discriminative Correlation Filters for tracking and has demonstrated encouraging outcomes in urban driving contexts, however it has not been comprehensively evaluated against other leading trackers [13]. Although considerable advancements have been achieved in object identification for autonomous driving, numerous problems persist, especially in areas such as Bangladesh, where distinctive traffic conditions and environmental elements hinder the detection process. The Bangladeshi Autonomous Driving Object Detection Dataset (BADODD) offers a significant chance to assess the efficacy of advanced models like as YOLOv5, YOLOv7, and YOLOv8 in real-world scenarios. This research seeks to evaluate these models using BADODD to ascertain their strengths and weaknesses, providing insights for further optimization in autonomous driving systems designed for complex and unstructured situations. In order to surmount these constraints, we have seen the success of deep learning methods in particular, Convolutional Neural Networks (CNNs), for object detection. Since Convolutional Neural Networks (CNNs) have vastly improved the accuracy and efficiency of vehicle identification models based on images, removing the salient characteristics from images is important. Region based Convolutional Neural Networks (R-CNN) was a significant milestone in CNN based object detection for the reason that it created the two step detection methodology. This method includes the generation of candidate zones of interest, categorization and refinement to further increase the object identification accuracy [5]. This approach was a significant step forward in using a CNN to object ID and initiated a new approach of more robust and scalable detection models.

Hence, after CNN based object detection, there emerged subsequent work to improve the R-CNN architecture, such as Fast R-CNN and Faster R-CNN. As a result, Fast R-CNN utilized a shared convolutional feature map for region recommendations, improving feature extraction efficiency and decreasing computation time and speed [6]. In Faster R-CNN, a new technique based on a network was used to generate region proposals instead of the conventional selective search method, which is known as Region Proposal Network (RPN). Bounding box predictions with Faster R-CNN got improved precision thanks to the use of the anchor notion, which made it powerful in recognizing objects of various sizes and scale. Both of these two stage approaches reached quite high accuracy, but due to the significant computational complexity they were not suitable for real time applications like autonomous driv-

ing for which fast processing speed is important [52]. In order to overcome the deficiencies of two stage detectors on real-time applications, single stage detectors like YOLO (You only look once) and SSD (Single shot Multibox detector) were developed. As a result, single stage detectors yield predictions for both of the object's categorization and bounding box in a single traversal of the neural network, making region proposals and classification phases unnecessary [45]. Announced in 2015, YOLO was a big jump in object identification speed, processing images at 45 frames per second (fps) in real time for applications using fast response time such as autonomous driving. Its principal innovation was the ability to treat object detection as a regression problem—i.e. run object bounding box and class predictions directly off full image photos. However, YOLO's first runs had challenges in distinguishing small objects, and precisely localizing objects, particularly in crowded or intricately diverse environments [10]. YOLO, as well as subsequent iterations of YOLO – YOLOv4, YOLOv5, and so forth – had addressed these errors to some extent. Sophisticated methodologies like a Feature Pyramid Networks (FPN) for multi scale detection, and a new loss function CIoU_Loss for bounding box precision was introduced in YOLOv4. While YOLOv5 is not formally referred to as a successor by the original creators, its popularity is due to its efficient architecture and their superior performance against multiple benchmarks. In particular, the most recent ... iteration [22] significantly improved detection precision and resilience, particularly in challenging environments, such as inclement weather, low illumination, and crowded urban environments. Under these enhancements, YOLO models are among the most common ones for real time object recognition in autonomous driving. These advancements notwithstanding, existing approaches of object detection, particularly YOLO are difficult to deploy in places where traffic systems are not commonplace, like Bangladesh. Road infrastructure in Bangladesh has its own peculiarities: these are ambulances and rickshaws, CNGs and other ambulatory, as well as unorganized traffic and poor roads. In these situations, these models are often trained on global datasets, but they are underrepresented in global datasets used for object detection model training, thus the evaluation of these models in region specific settings [22]. To address these distinct issues and to enable a focused investigation of deep learning models adapted to these conditions, we present the Bangladeshi Autonomous Driving Object Detection Dataset (BADODD). The identification of occluded objects, those that are either fully or partially occluded by other objects or ambient elements, is one of the major obstacles to making autonomous driving feasible. Then the identification of occluded objects is necessary for the safety and reliability of autonomous cars, both in urban environments, where vehicles or pedestrians can be partly occluded by other vehicles/structures or obstructions. Many of the object detection models use conventional non maximal suppression (NMS) methods which find difficulty in occluded objects and often miss detections or produce erroneous positives. To address this challenge there have been proposals such as soft-NMS and other methods to increase accuracy on occluded object detection, particularly on small and nearby objects. There is another important direction in developing lightweight object detectors that can run on edge devices well. For example, real time applications for autonomous driving require fast inference speed since objects have to be detected instantaneously and the vehicle has to make immediate decisions. Additionally, the YOLO based models were demonstrated to well perform on constrained resource hardware platforms (Edge

oriented NVIDIA Jetson AGX Xavier). Thanks to optimizations such as NVIDIA's TensorRT and DeepStream SDK fast inference time at no cost to detection accuracy makes the models deployable in the real world [1]. Thus, some recent works have tried to combine CNN-based object detector as tracking algorithm, such as DeepSORT and NvDCF, to improve the recognition and tracking dynamic objects in human's intelligent driving environment. These trackers follow object identity by tracking bounding boxes detected by object detectors across successive video frames where object appearance changes or partial occlusion arises.Traditional tracking algorithms are extended with appearance information to extend the length of occlusion periods and identity switches to as little as two [49].ects. Another important direction is developing lightweight object detectors that can run effectively on edge devices. For example, real-time autonomous-driving applications require fast inference speeds since the vehicle must make instantaneous decisions based on the objects detected. Moreover, the YOLO-based models were also shown to yield promising results when deployed on resource-constrained embedded hardware platforms, such as the edge-oriented NVIDIA Jetson AGX Xavier. Optimizations like NVIDIA's TensorRT and DeepStream SDK provide fast inference times without compromising on detection accuracy, hence making the models deployable in the real world [1]. Given this, some recent works have focused on the integration of CNN-based object detectors with tracking algorithms like DeepSORT and NvDCF for better recognition and tracking of dynamic objects in an autonomous driving environment. These trackers follow the object identity using the bounding boxes provided by object detectors along successive video frames, even when partial occlusion or object appearance changes occur. DeepSORT extends traditional tracking algorithms by incorporating appearance information, thus allowing o monitor objects over longer occlusion periods with fewer identity switches [49]. NVIDIA's proposed NvDCF was based on Discriminative Correlation Filters giving accurate results in urban driving with their never tested against the other top trackers [13].

Chapter 3

Data

3.1 Data

3.1.1 Overview

It enhances object recognition in an autonomous driving vehicle through the utilization of a customized dataset privately developed to address the complexities of Bangladesh traffic. The proposed dataset, drawn from the contributions of the authors themselves, has taken inspiration from the publicly available BADODD-Bangladeshi Autonomous Driving Object Detection Dataset; yet, it actually overcomes some limitations of the latter with regard to atypical vehicle representation and erratic traffic patterns in Bangladesh. While BADODD doesn't include non-conventional vehicles like rickshaws and CNGs, along with changing lighting and weather conditions, this dataset allows for a more in-depth and complex presentation of Bangladesh dynamic road environments. The dataset contains various types of vehicles, different traffic scenarios, and a number of environmental factors when the visibility is low, congestion is high, and the weather is adverse. With it, it brings all the complexities of the dataset and challenges any autonomous driving system on Bangladeshi roads. A private dataset curated with great care to ensure its applicability in training object detection models for highly dynamic and unstructured environments. It forms a great base in testing and fine-tuning deep learning models like YOLOv5, YOLOv7, and YOLOv8 for ascertaining excellent performance on the road under different conditions.

3.1.2 Data Collection

In order to ensure diverse road landscapes and traffic patterns, this research employs a bespoke dataset comprising of 19,227 photographs from two Bangladesh areas. Mobile devices that were selected for their accessibility, user friendliness and adaptability were used in data collection. This strategy allowed the researchers to easily adapt to a wide number of locations and situations, taking a great variety of photos in many differing contexts. Mobile devices provided a pragmatic, cost-effective means for both gathering and using real world data that allow incorporated of different traffic conditions that are commonly witnessed in Bangladesh. Photos were taken from different environmental situations like daylight, nighttime, foggy weather etc. The heterogeneity is critical to match the real world problems en-

countered by autonomous driving systems. Due to this, the dataset is more specific representation of the varying visibility and traffic density on Bangladeshi highways. Diverse settings ensure the robustness of the dataset, emproving the trained object identification models to work well in unanticipated situations, for example automobile recognition in bad visibility or adverse weather conditions. The dataset includes different traffic types such as urban congestion, rural roads, freeway and so on. The range of this dataset ensures that the dataset is a complete representation of the complexities of traffic in Bangladesh where both unconventional vehicles such as rickshaws, CNGs, vans, and also conventional vehicles like automobiles, buses, and trucks exist. The Roboflow tool was used for pre processing the photos for training dataset preparation. Consisting in uniform downscaling to 416x416 pixels for the all photos to allow the application of the YOLO models, the photos were Photos. The annotation process became even easier on Roboflow as it allowed to precisely label many objects in each image which is what you need to train and assess object detection models.

3.1.3 Data Description

Using a bespoke dataset of 10,331 photos of 13 distinct object types that have been carefully selected for the unique and unexpected traffic patterns on Bangladeshi roads, this research is carried out. The purpose of creating this dataset is to represent real world scenarios, which has standard and atypical vehicles present together, as they normally appear in the Bangladesh. The aim is to improve object detection model efficiency (i.e. YOLOv5, YOLOv7 and YOLOv8) ability in identifying objects in various and difficult scenarios. The dataset comprises 13 object classes, which are as follows:

• Automobiles: Conventional automobiles are utilized on both urban and rural thoroughfares in Bangladesh.

• Buses: Substantial passenger vehicles frequently utilized in public transit, prevalent on highways and urban thoroughfares.

• Trucks: Robust vehicles employed for the conveyance of commodities, commonly observed on highways and rural thoroughfares.

• Rickshaws: Human-powered three-wheeled vehicles, distinctive to South Asia, commonly found on urban and semi-urban thoroughfares.

• CNGs (Compressed Natural Gas vehicles): Auto-rickshaws serving as a principal means of public transit, particularly in urban areas.

• Vans: Compact, frequently casual, transport vehicles utilized in both urban and rural environments.

• Motorcycles/Bikes: Two-wheeled vehicles frequently utilized for personal transportation, prevalent on all types of roads.

• Bicycles: non-motorized two-wheeled vehicles predominantly located in urban and semi-urban regions, utilized for short-distance travel.

• Pedestrians: Human movement, crucial to identify in both metropolitan intersections and rural roadways.

• Road Signs: Standardized indicators delivering road directions, essential for autonomous driving systems to comprehend traffic regulations.

• Road markers: Lane and directional markers are essential for comprehending road configurations and facilitating vehicle navigation.

• Wheelchairs: Signifying infrequent yet essential items that necessitate particular focus for safety.
• Construction Vehicles: Substantial, sluggish vehicles utilized in construction zones, exhibiting varied movement patterns and associated dangers.

3.2 Class Representation and Data Imbalance

Due to the varied traffic conditions in Bangladesh, certain classes occur more frequently than others. For instance, rickshaws and pedestrians are prominently shown owing to their prevalent use in urban and semi-urban locales, but wheelchairs and construction vehicles are hardly encountered. The natural class imbalance was rectified during dataset creation. No formal data augmentation approaches, such as oversampling, were utilized; rather, the dataset's authentic representation guarantees that models may generalize effectively across diverse traffic scenarios. The incorporation of common objects such as rickshaws, CNGs, and pedestrians enables models to be trained for the detection of these essential components in intricate traffic scenarios, while rarer objects like wheelchairs and construction vehicles contribute significant diversity to the dataset. This combination aids models in adapting to various road circumstances, which is crucial for autonomous vehicles functioning in dynamic environments.

3.3 Image Preprocessing and Annotations

All images in this dataset were pre-processed to the size of 640x640 pixels to make all images uniform and according to the YOLO models. This is very critical in model training, as it minimizes computational complexity and ensures that each object is analyzed at a comparable scale. That was a pre-processing step, ensuring that the dataset is prepared for the needs of the object detection algorithms, such as YOLOv5, YOLOv7, and YOLOv8, considering their requirements for input data size. In general, the annotations were made in a systematic manner with the use of Roboflow, an AI-designed technology for managing, augmenting, and analyzing image datasets. With Roboflow's help, all objects present within the dataset were annotated with great detail to accuracy, meaning that objects in each single image were correctly classified and their locations delineated with their bounding boxes. Therefore, annotation was a crucial procedure for training object detectors; it would enable the object detection models to locate objects with high accuracy and classify things in images- especially those with complex context with overlapping and occluded objects.

3.4 Impact of Dataset Design:

Careful compilation of the dataset coupled with comprehensive annotation substantializes object detection models' effectiveness. Diversity of object classes and especially addition of classes of local transportation modes, rickshaws, and CNGs, make the models adept in dealing with peculiar traffic behaviors of Bangladesh. This dataset increases models' generalization and adaptation capabilities in uncertain

contexts due to the inclusion of real-world traffic scenario intricacies, such as variation in weather conditions, traffic density, and visibility problems. Natural object distributions and the lack of synthetic augmentation methods further ensure that the dataset is from practical use. Focus on class balance, especially with elaborate representation of common and important objects in this regard helps to overcome common challenges usually associated with unbalanced datasets, tending to bias, especially towards the more prevalent classes. It will provide a great benchmark toward evaluating and enhancing the results of object detection models in the diverse and challenging on-road environment in Bangladesh.

3.5 Statistical Analysis:

It uses a dataset of 5,243 photos, annotated with 19,227 instances across 16 distinct object classes commonly encountered on Bangladeshi highways. These are images with a resolution of 416×416 pixels on average, thus suitable for YOLO object detection models. The dataset should be one that is diverse, showcasing dynamic traffic scenarios in Bangladesh both with conventional and unconventional vehicles. This dataset forms the enriching basis for training models that should be good at negotiating the nuances of the local traffic environment. The dataset contains cars, buses, lorries, motorbikes, and auto-rickshaws, apart from pedestrians, road signs, and other elements of traffic. Each class of items in the dataset has a unique number of annotations, representative of their occurrence within real traffic scenes. Some common classes in this dataset include cars and trucks, while others, such as ambulances and construction equipment, are less common. Shown below is an overview of the dataset

3.6 Class Imbalance Analysis

The dataset exhibits a pronounced class imbalance, with specific item classes occurring far more frequently than others. Automobiles, lorries, and auto-rickshaws predominate the dataset, signifying the most prevalent vehicles on Bangladeshi thoroughfares. Conversely, ambulances and trains are infrequent, with fewer than 20 occurrences. Such imbalances may result in biased training of object detection algorithms, as the model may find it challenging to acquire the attributes of underrepresented classes, such as ambulances or construction vehicles. Oversampling or under sampling: By intentionally augmenting the sample size of underrepresented groups or diminishing that of overrepresented ones. Class weighting: Allocating distinct weights to each class during model training, hence prioritizing the less prevalent classes. Data augmentation: Implementing changes on existing images of infrequent classes to artificially enhance their prevalence. Nevertheless, the dataset summary indicates that no explicit oversampling or augmentation methods have been employed to equilibrate these class distributions. This architecture mirrors actual traffic situations but may require additional measures to guarantee model robustness across all categories. The total annotations amount to 19,227, average two annotations per image, indicating the prevalence of several items in the majority of photographs. The image dimensions are uniform, including a median resolution of 416×416 pixels, which guarantees compatibility with YOLO models and minimizes

computing complexity. Uniform picture resolution is essential for training object detection models, as it guarantees that items are represented at a comparable scale throughout the dataset. The model is predicted to rapidly learn common classes such as vehicles and trucks, attaining excellent accuracy. Nonetheless, infrequent classes, such as ambulances and trains, may result in inadequate generalization during testing owing to a lack of data. This disparity may also result in a significant number of false negatives for seldom classes, wherein the model may not detect their existence in actual traffic scenarios.Numerous vehicles, including rickshaws, CNGs, and walkers, guarantees that the dataset encompasses a broad spectrum of object types, essential for a model used in Bangladesh's heterogeneous traffic landscape. The incorporation of items such as road signs and human pedestrians significantly improves the dataset's authenticity, as these are essential components for real-time navigation in autonomous driving systems. Table 3.1 and in figure 3.1 shows Data distribution per class.

Vehicle Names	**Vehicle Number**	**Percentage (%)**
Car	11181	36%
Truck	9103	21.31%
Auto Rickshaw	2770	8.92%
Bus	2064	6.64%
Bike	1902	6.12%
Human	1151	3.7%
CNG	1140	3.67%
Road Sign	1113	3.58%
Van	318	1.02%
Construction Vehicle	202	0.65%
Cycle	39	0.12%
Fire Service	31	0.1%
Pick-up	31	0.1%
Ambulance	14	0.04%

Table 3.1: Data Distribution Per Class

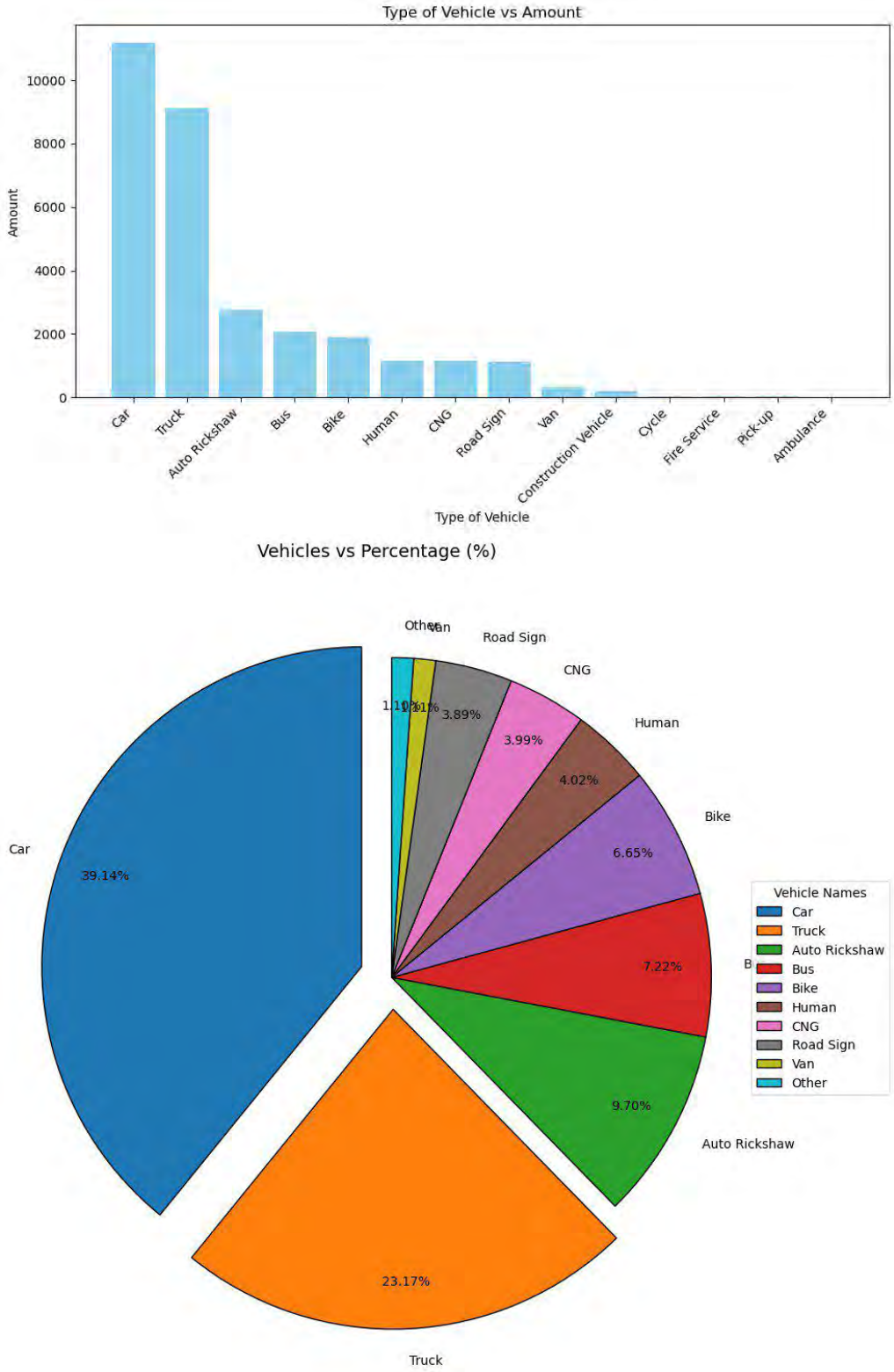


Figure 3.1: Amount of Data Per Class

Chapter 4

Research Methodology

4.1 Research Methodology

The study aims to enhance object detection for autonomous vehicles, adapted to Bangladesh's road environment, by comparing three advanced YOLO models: YOLOv5, YOLOv7, and YOLOv8. Therefore, the dataset used in the study is a custom, private dataset developed by the authors, which is actually the main contribution of the current study. This dataset encompasses annotated photos of different traffic entities, such as vehicles, buses, rickshaws, motorcycles, and pedestrians, representing several characteristic conditions of Bangladeshi highways. While the dataset is an extension of BADODD, it contributes to improving the same in light of consideration of a bigger range of unconventional vehicles, such as rickshaws and CNGs, erratic traffic flow, and shifting illumination conditions and varying weather scenarios of on-road Bangladeshi settings.

The methodology starts with thorough data preprocessing, involves augmenting techniques like random scaling, rotation, and flipping to make models invariant from real-world conditions such as occlusions and variations of lighting. Transfer learning is employed to train the models efficiently by utilizing pre-trained YOLO models that have been originally trained on the COCO dataset, while later tuned with this custom dataset. This is done so that the models get adapted to the specific constraints of Bangladesh's road environment while drawing on insights from extensive varied datasets like COCO. Training involves hyperparameter tuning such as learning rates, batch sizes, anchor box configurations. YOLOv5, YOLOv7, and YOLOv8 were tested after training by standard object detection metrics: mAP at a threshold of 0.5 and mAP at 0.5:0.95 IoU thresholds [39]. These metrics will allow us to investigate the performance of each model for large and small object detection on complex traffic scenes. Besides this, the time taken for inference is calculated, which will further check the suitability of the models for real-time autonomous driving applications. The paper will focus on finding the best version of YOLO that is able to implement the technology in autonomous vehicles on improved model generalization Bangladesh's highways, keeping in view the said models' accuracy and computation efficiency with their speed.

4.1.1 Workflow

The proposed research technique follows a systematic workflow, starting with the collection and compilation of the custom dataset developed for this study and considered the main contribution of this research. The custom dataset created for the unique traffic challenges of Bangladesh consists of 10,331 photos tagged with 13 item categories of both conventional and non-conventional vehicles. Data collection is done through mobile devices for capturing the real-world scenario in several districts with variations of environmental conditions. In this way, they could gather varied patterns of traffic, multiple lighting conditions, and a number of weather conditions standard on Bangladeshi roads. Later, these images were down-scaled at 416×416 pixels and augmented by rotation, flipping, and brightness changes to increase robustness. This will ensure everything in the dataset is consistent while increasing variation to help generalize the model further. Three state-of-the-art object detection models such as YOLOv5, YOLOv7, and YOLOv8 were selected for experimentation; all were initialized with pre-trained weights on the COCO dataset and fine-tuned with this Bangladeshi dataset created in the current work. This script uses the cloud infrastructure of Google Colab for effective training with GPUs, optimizes batch size and number of epochs, and optimizes input dimensions of the images. Finally, for the results obtained after training on these models, performance metrics such as precision, recall, and mean Average Precision, supported by confusion matrices, were obtained to see how well detections would generalize across the 13 item categories. The different results were analyzed to determine which YOLO model, among the variants, is most appropriate for Bangladesh's unique traffic environment consisting of regular and irregular vehicles. Figure 4.1: Proposed approach to deal with challenge.

The YOLOv5, YOLOv7, and YOLOv8 models were chosen for comparison and trained with pre-trained weights from the COCO dataset, subsequently fine-tuned for the Bangladeshi dataset. The training procedure employed Google Colab's cloud infrastructure for effective GPU-based training, optimizing batch size, epochs, and image input dimensions. Upon completion of training, the models were assessed utilizing performance metrics including precision, recall, and mean Average Precision (mAP), with confusion matrices to evaluate detection accuracy across the 13 item categories. The results were examined to ascertain which YOLO model exhibits optimal performance within Bangladesh's distinctive traffic environment, taking into account both conventional and atypical vehicles. The methodology proposed for addressing this issue is illustrated in Figure 4.1.

4.1.2 Data Preprocessing

Data preprocessing is a crucial phase in preparing the dataset to enhance model performance on object detection tasks. A number of procedures have been conducted within this research work to format and augment the dataset properly, customized for Bangladeshi road conditions, which is to be used for the models YOLOv5, YOLOv7, and YOLOv8. A custom dataset has been developed solely for this research work, considering special traffic conditions of Bangladesh where conventional and non-conventional vehicles are in use. Preprocessing: The size of all images from the custom dataset was reduced to the standard size of 640×640 to unify these images and meet the input requirements of YOLO models. Scaling

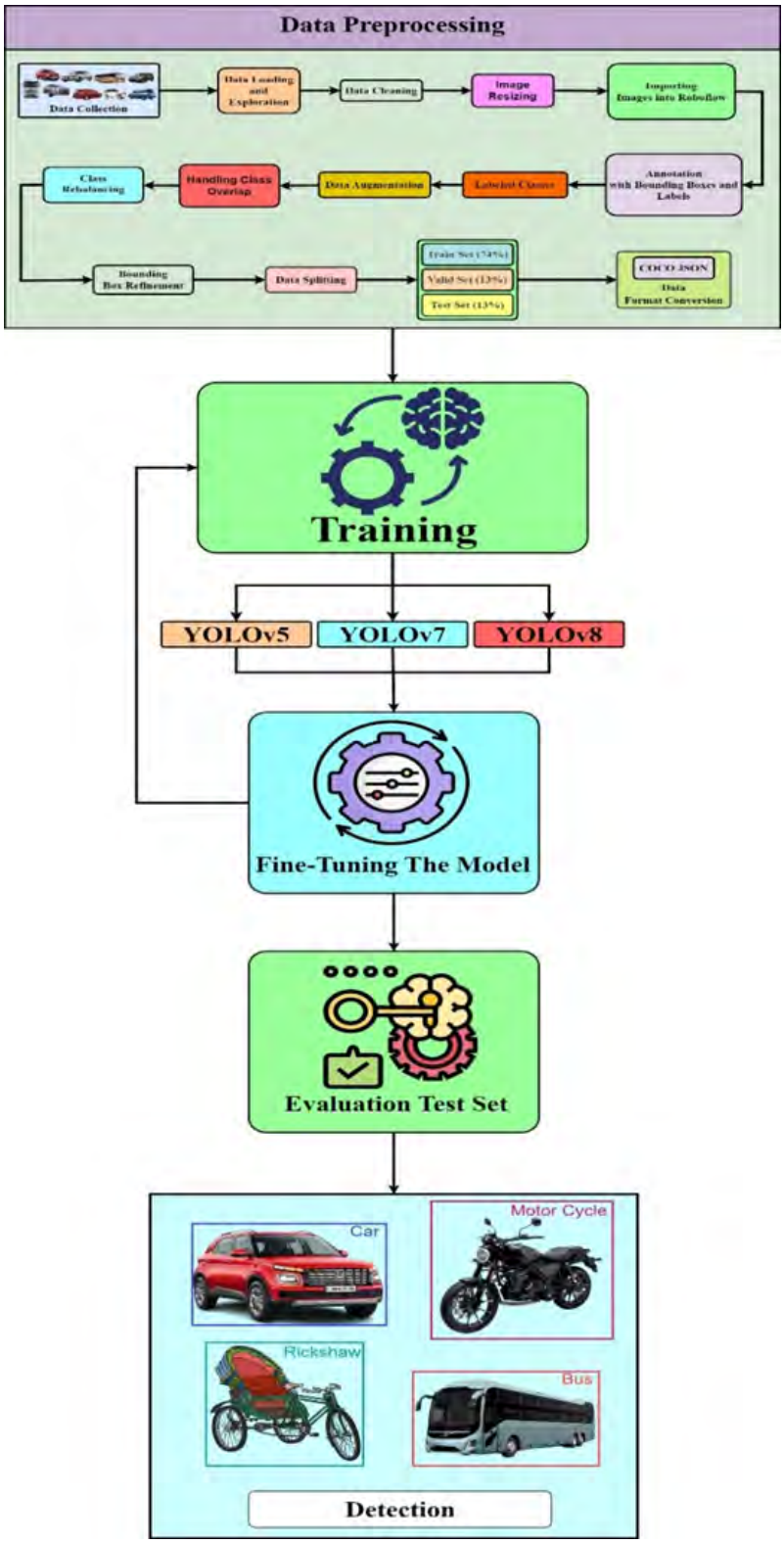


Figure 4.1: The Overall Workflow

unified the input and helped decrease the computation burden. For greater variety in the dataset, the following data augmentation techniques were applied to the dataset: random rotation, alteration of brightness and contrast, horizontal and vertical flipping, and adding Gaussian noise. These augmentations allowed the models

to generalize better by replicating various real-world scenarios, including changes in lighting conditions, object orientation, and noise. Particular emphasis was put on avoiding class imbalance during the annotation process. Classes of objects common in Bangladeshi traffic, such as rickshaws and pedestrians, were over-represented, while classes of less common items, like wheelchairs or construction trucks, were not as prevalent. The custom dataset was designed with natural scenes, rather than by artificial class balancing, in order for it to retain its representativeness of real traffic scenarios. While other tools like Roboflow were used to make the annotation process efficient so that the bounding boxes would be drawn around objects that precisely linked to their labels. Lastly, this dataset is divided into training, validation, and test sets while maintaining similar class distributions in each set. Hence, this preparation phase will be highly exhaustive to make the custom dataset resilient and ready for optimum performance in the training and evaluation of the model. Figure 4.2 illustrates the proposed pre-processing with respect to the problem at hand.

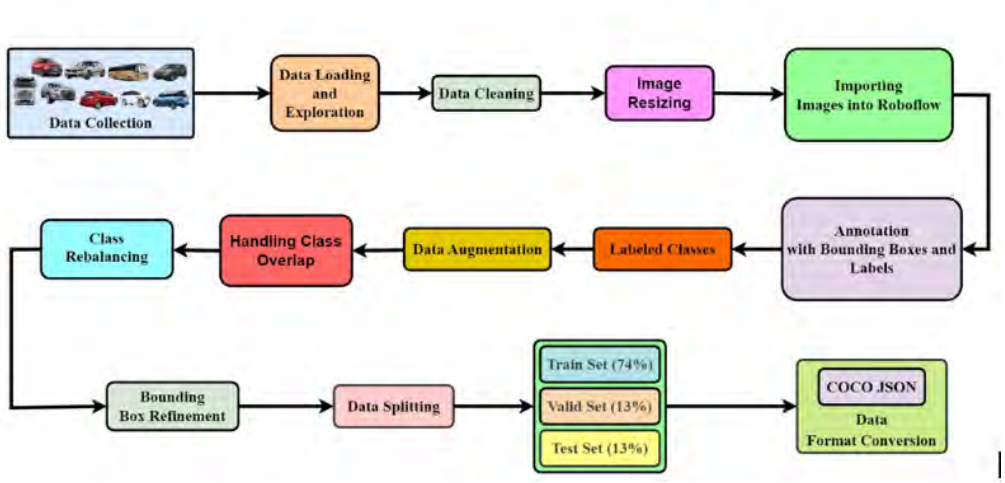


Figure 4.2: Data Preprocessing

4.1.3 Data Loading and Exploration

Loading and exploration of data in this phase, in fact, are very important with a view to establishing that the dataset has been organized and prepared for the model training process. In this work, the custom-curated dataset developed for this research was imported using the PyTorch and TensorFlow libraries so it may be compatible with such models as YOLOv5, YOLOv7, and YOLOv8. They were then organized into standardized formats such as YOLO or COCO JSON, including the photos themselves, with associated annotations, bounding boxes, and class names. Loading images in and of themselves had to account for reading images by batch into memory for scaling up computing performance and scaling images for normalization, thus meeting requirements set by YOLO models. It applied API from Roboflow, which allowed one to automatically convert and fetch photos with smooth integration of the annotations. In the exploration phase, it is done by reviewing a custom dataset to understand the composition: class distribution and the attributes. The important features to be considered, like the number of instances per class, the resolution of images, and any kind of biases or imbalances that existed, were reviewed graphically.

With this, analysis could determine whether data augmentation techniques would be necessary, really poorly represented categories could be spotted-like wheelchairs and construction vehicles. This was further supported by the verification for faulty or duplicate data, along with verification that all photographs were correctly labeled. This stage let any probable problems be identified early; hence, it prepared the custom dataset for training and validation.

4.1.4 Data Cleaning

Machine learning is raced in the cleaner than a whistle data in this way. Data cleaning is the method used to provide of high-quality data absent of anymore than one version of the same data with several copies or errors. Moreover, (in particular), the main activities include the deletion of duplicate photos which can lead to omissions or line the training data improperly. This serves as a result of the model becoming overfitting, hence, poor performance. Trying to put it in other words, the overfitting is a high probability when the model identifies some patterns from the duplicate samples of the output then becomes quite the opposite, it already generalizes, but not in a good way because fed with the training data. Secondly, we should deal with the problem of missing or damaged data so that the dataset should remain complete and accurate. Even such corrupted or incomplete photos can be modified so that misinformation results or certain images may not have the correct components necessary for it to fail during training and create incorrect predictions. Time and error are the data, among others, that are harmful and to be deleted. Thus, the erroneous or incomplete images remain back in the data set concerned, while the satisfying and exact examples get the right way for the proper training and, accordingly, the reliability and efficiency of the model increase.

4.1.5 Image Resizing

The scaling of images is an important step before pre-processing YOLO object detection models, where the input image is required to have the same dimension, for example 416x x 416 or 640x x 640 pixels. The image size normalization is done to fix this and in so doing ensures the provision of a consistent data flow to the model just as "most of the models require a fixed input size" would be used modulo a few examples. This is the main reason for scaling photos in such a way that the original aspect ratio is not lost which may lead to a distorted look of the image. Deformed images can deceive the model in its detection and target the originality by reducing its accuracy, especially for the objects of diverse forms and sizes. Padding comes into play as a technique to preserve the aspect ratio, where white pixels are added to the free space so that the image stays within the given dimensions even though its size could be scale adjusted. This assists in sustaining the authenticity of visual information and also enables the detection of items in the dataset to be better.

4.1.6 Importing Images into Rob flow

We employed Roboflow, a third-party tool, to optimize the image processing workflow by effectively managing and preprocessing our collection. Images were initially submitted to Roboflow, which accommodates several file types and facilitates easy

integration with object detection processes. Roboflow's user-friendly interface facilitated the organization of the dataset and its preparation for subsequent annotation and augmentation. Utilizing this platform, we guaranteed uniform handling of all photos, preparing them for annotation and training on models such as YOLOv5, YOLOv7, and YOLOv8. This site also offered tools for resizing photographs and preparing them for export in YOLO-compatible formats.

4.1.7 Annotation with Bounding Boxes and Labels

After importing the photos into Roboflow, the subsequent step was annotating. Every image was meticulously labeled with bounding boxes that accurately delineate the things of interest. The bounding boxes were generated by manipulating and adjusting a box around the object in the image. Upon delineating the box, we designated a class label to the object (e.g., car, rickshaw, pedestrian), guaranteeing accurate identification of each thing within the image. The annotation process is essential for training object identification models, as it supplies the ground truth data from which the model learns during training. Roboflow's technologies facilitated the optimization of this procedure, guaranteeing precise and uniform categorization of all photos inside the dataset.

4.1.8 Labeled Classes

This study utilizes a dataset with a varied array of labeled object classes that represent the distinct traffic conditions of Bangladeshi highways. Every object in the photos is classified into one of these predetermined categories, guaranteeing thorough representation of both traditional and non-traditional vehicles. The designated categories encompass conventional vehicles like automobiles, lorries, buses, and motorbikes, in addition to local transport modes such as rickshaws, CNGs (three-wheeled auto-rickshaws), and vans. Moreover, non-vehicular categories, including pedestrians and traffic signs, are incorporated, so augmenting the dataset's richness. Such class distribution will ensure that the models learn to detect a wide range of objects relevant to autonomous driving contexts in Bangladesh and accommodate the traffic condition of this region. The classes identified here present the foundation for object recognition algorithms that correctly identify and classify things in real on-road driving scenarios. Figure 4.3: Labeled Classes.

Figure 4.3: Labeled Classes

4.1.9 Data Augmentation

Roboflow's preparation pipeline allowed the application of several data augmentation strategies that increase variety in this dataset and reduce the risk of overfitting. Different augmentations were done with a given probability to mimic real variation. This included random rotations of ± 15 degrees to help the model generalize on the position of objects. Flipping was done on both horizontal and vertical axes to offer the model different perspectives of objects. Other changes included brightness and contrast to simulate different lighting conditions, therefore making the model more robust against changing environments. Finally, some images were added with Gaussian noise in order to increase the model's robustness against noisy or corrupt input, hence improving its generalization capability on unseen scenarios.

4.1.10 Handling Class Overlap

Class overlap is dealt with in such a manner that many objects may co-exist in the same region of the picture, which it precisely needs to detect and classify. Multi-class annotation is used when there is an overlap of objects. This makes sure that each overlapped object will be correctly tagged along with its bounding box in case they

are partially hidden by one another. This allows the model to distinguish between different objects in the same area and to have an increased ability in determining the location of overlapping objects unambiguously.
Object Occlusion Handling takes into consideration occluded or partially hidden objects by other objects in the scene. Carefully annotating occluded objects, bounding box attaches delineate the visible segments to train the model to identify objects even when they show up partially. This becomes even more important in cluttered scenes, for example, in road traffic scenes where different cars or persons may overlap. Efficient handling of occlusions will enable the model to maintain high performance in detection and ensure high precision and recall for object detection and classification of challenging real-world scenarios.

4.1.11 Class Rebalancing

Class rebalancing strategies are of great importance in order to correct an imbalance in the dataset for limited numbers of instances of a given object class. This can be achieved, among other ways, by oversampling some minority classes with the aim of having enough of them during the training process. This allows for the possibility of a much fairer dataset, which avoids model bias towards predominant classes. Another approach consists of class weighting, where higher weights are attributed to the underrepresented classes within the loss function. This method does not require the actual oversampling of data to reduce class imbalance. It forces the model to pay more attention to the minority classes. Synthetic data generation methods like GANs can synthesize images of minority classes; hence, the minority class representation is improved without sacrificing the variety of the dataset.

4.1.12 Bounding Box Refinement

Accordingly, function of refining the bounding box is an vital procedure in order to get well the recognition of the right object with a high accuracy. Firstly, it is necessary to check the quality of the annotation again and again to guarantee that only the truly correctly set and label-marked boxes are around the objects. Instrumentation such as Roboflow or any other automated means of labeling defects are used to do this, the wrong-side boxes fall under no circumstances, however. At the same time, there is a parallel step called bounding normalizing which is the resizing of the boxes according to the image dimensions. YOLO models are compatible with this, and these models can locate objects that are already in a normalized state making the detection consistent despite the variation of the image's dimension.

4.1.13 Data Splitting

Data partitioning is a crucial preprocessing step that must be performed on any dataset for machine learning models which then can be used for training and testing. The common split is done on three subsets: training, validation, and testing. The Train-Validation-Test Split is a technique that allows the subset to be well-balanced and representative of all object classes in the splits. A possible way of splitting can be
74

• Valid Set: A validation set of 1355 photos assists in optimizing the model's hyperparameters.

• Test Set: 1311 photos are designated for the ultimate performance assessment.

Utilizing this systematic method for data partitioning enables the model to generalize effectively to novel, unobserved data, while guaranteeing that all object classes are adequately represented throughout both training and assessment phases. Table 4.1 shows amount of data per class and train, test and validation.

Class	**Total**	**Train**	**Valid**	**Test**
Car	3435	1909.0	700.0	826.0
Truck	2675	1262.0	519.0	894.0
Autorickshaw	1175	441.0	501.0	233.0
Bus	653	268.0	200.0	185.0
Motorcycle	653	361.0	185.0	107.0
Road sign	433	225.0	147.0	61.0
Human	430	218.0	132.0	80.0
CNG	404	173.0	180.0	51.0
Vehicles	116	44.0	52.0	20.0
Van	114	65.0	23.0	26.0
Construction vehicle	104	85.0	9.0	10.0
Pickup	90	70.0	11.0	9.0
Bicycle	71	30.0	17.0	24.0
Ambulance	14	14.0	nan	nan
Train	6	nan	nan	nan

Table 4.1: Amount of Data Per Class and Train, Test and Validation

4.1.14 Data Format Conversion

It is necessary to have a Data Format Conversion so that the dataset will be compatible with some specific model requirements. First of all, it is necessary to convert the labels to the proper format. In the case of YOLO models, labels usually need to be in YOLO TXT format or COCO JSON, depending on the YOLO version. Such conversion will ensure that the annotations of each object-that is, the bounding box coordinates and class labels-are correctly formatted for the model to read and apply during training. Besides, compression reduces memory usage and speeds up the time taken for training. Since high-resolution image formats are changed to more compact formats, such as JPEG, the consumed memory will be less while keeping the quality of the images intact. This step is crucial, especially when working with large datasets, since it allows the dataset to be processed much faster without losing any accuracy in its object detection model. Proper compression allows the data to be loaded in quickly, hence enabling efficient scalability of the training process.

4.1.15 Normalization

Normalization, in the context of your dataset and object identification task with the YOLO models, is crucial to have features within proper scales during training. Since the YOLO model works with images and needs bounding boxes, it requires normalization of image data and bounding box coordinates; otherwise, full functionality and model performance will not be ensured. While many models of YOLO would normally need input photos to be downsized to a standard dimension, such as 416x416 or 640x640 pixels, normalizing pixel values is equally important. In normalizing pixel values, which range between 0 and 255, scaling is done within the range of 0 to 1 in order to enable the YOLO model to converge faster and make its training effective. It normalizes the pixel values so that the contributions of all features-that is, pixel intensities-become equal in the learning process for accurate object detection.
Bounding box coordinates, which denote the position and dimensions of objects, should be normalized regarding picture dimensions. In general, YOLO models expect these in the range between 0 and 1, which means representing fractions of the whole image size. For each bounding box, the center coordinates, x, y, and the dimensions - width and height, w and h, are normalized by division with the width and height of the image. This ensures that a model will detect objects correctly, regardless of the image dimensions or aspect ratio. The Min-Max scaling works particularly well because this normalizes values in the range of [0, 1], which is what YOLO models expect. This will ensure that the performance with different image dimensions will be the same, and the bounding box proportions are maintained; hence, the model will improve its object detection precision while reducing training time. It normalizes the pixel values and bounding box coordinates, making sure the model will learn from the training data with no bias toward any kind of feature or dimension.

4.1.16 Model Description

Among the many capabilities, real-time object detection and recognition have been highly helpful in the working of autonomous vehicle systems due to YOLO, thus helping them in instantly detecting and keeping track of several things running around, such as vehicles, pedestrians, bicycles, among others. Some of the areas where these capabilities have been employed include activity detection [20], sports analysis, surveillance video sequences, and human-computer interaction [40]. YOLO models, in agriculture, are used to improve precision farming techniques and automate agricultural processes through the identification and characterization of crops, pests and diseases. They have been adapted to applications such as biometrics, security, and facial recognition systems for face detection tasks [[21], [35]]. Yolo has found its place in the medical field as it has been used for the identification of pills [44], skin segmentation [28], and cancer diagnosis [17], [26]. This has greatly increased the diagnostic precision and the optimization of treatment protocols. Additionally, the same have been used in remote sensing to help in land use mapping, urban planning, and environmental monitoring through object recognition and classification in the satellite and aerial imagery [36]-[42]]. The use of YOLO models in security systems has made social distancing possible due to the facial mask detection and the real-time video feed monitoring as a result of the rapid identification

of suspicious activities [50],[38]]. Furthermore, the models have been instrumental in ensuring the quality control in manufacturing and production by being a part of the surface inspection to detect faults and anomalies [29]-[27]. In the settings of operational traffic, YOLO models have found applications that include license plate detection [37] and traffic sign recognition [23] leading to the development of intelligent transportation systems and traffic management solutions. The method is also applied in wildlife detection and monitoring to recognize endangered species for biodiversity conservation. The details of the paper are as follows: Learn. Knowledge Extraction. 2023, Volume 5, Page 1682, Conservation and Ecosystem Management [47]. Not only that, YOLO is used a lot in robotics [57], [31] and in the object detection from drones [24],[43] are the last applications. In figure 4.4 a bibliometric network visualization has been shown, and it is a conglomeration of all the papers that contain the term YOLO in the title that were identified in Scopus, and then keyword object detection was used to filter the resulting papers. We spent time to meticulously filter all the applications.

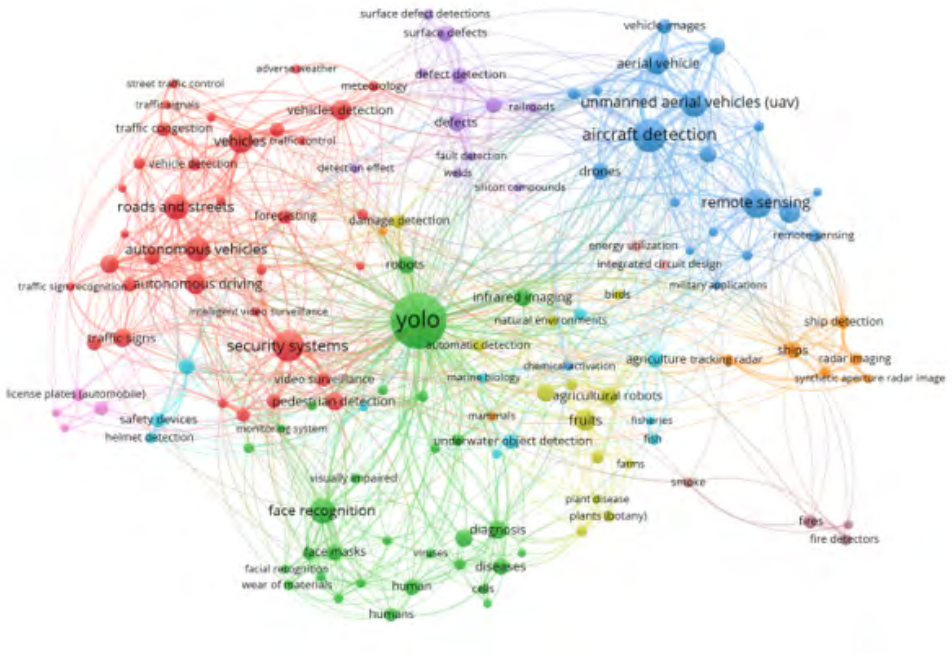


Figure 4.4: Bibliometric network visualization of the main YOLO Application created with [28].

You Only Look Once The work of Joseph Redmon and colleagues, YOLO, was published at CVPR 2016 [10]. It presented a novel end-to-end real-time item identification methodology. YOLO, an acronym for "You Only Look Once," signifies the execution of the detection task in a singular network pass, contrasting with previous methodologies that employed sliding windows necessitating hundreds or thousands of classifier runs per image, or more advanced techniques that bifurcated the task into two phases: the initial phase identifies potential object regions or region proposals, followed by a second phase that applies a classifier to these proposals. Furthermore, YOLO utilized a more straightforward output mechanism that depended just on regression to predict detection results, unlike Fast R-CNN, which utilized two separate outputs—a regression for bounding box coordinates and a classification for probabilities. YOLOv8 has been chosen for real-time object detection

in autonomous driving due to its distinctive advantages within the YOLO model family, which align seamlessly with the demands of our application. YOLO models integrate seamlessly due to their esteemed real-time object detection abilities, which are crucial for swift and precise perception in dynamic driving environments. YOLOv8 employs a one-stage detection algorithm that enhances accuracy, speed, and resilience, hence optimizing computational efficiency while ensuring a high degree of precision, essential for autonomous vehicles to make real-time choices. YOLO architectures have some absolute advantages that include end-to-end training and vastness across many different object categories. They make them really good for complicated situations on the road.

4.2 Model Selection

Model selection is very important in this research to optimize the performance of object detection in Bangladeshi road conditions. Three different versions of the YOLO concept were selected in this research based on their specific strengths and functionalities.
YOLOv5: It is famous for its lightweight architecture. This version aims at real-time detection, especially in constrained processing environments. Its balance of speed and accuracy currently makes this network one of the preferred choices for applications requiring fast inference, especially for mobile or embedded systems.
YOLOv7: Much more was done compared to the previous versions in terms of an improved trade-off between speed and accuracy. YOLOv7 can provide higher precision with computational efficiency and, hence, can be applied to complicated tasks where object detection in various scenarios is required.
YOLOv8: The newest in the YOLO series, it is full of serious optimizations that improve both precision and recall. It shows excellent performance in adversarial environments, which are those low-light or foggy conditions when normally detection precision is poorly affected. Improvements on YOLOv8 make it the best fit for applications in which detection needs to be reliable in tough conditions.
All of these models were then further tuned with pre-trained weights, often from the COCO dataset, to adapt much better to Bangladeshi traffic situations. In this way, it could detect unusual vehicles like rickshaws and three-wheelers together with objects generally found in a local traffic environment. In this regard, all the models mentioned ensure that different problems related to object detection are dealt with, resulting in a balanced trade-off between real-time performance and accuracy in the detection.

4.3 Architecture

The "refreshingly simple" approach was first introduced to the commonly utilized object identification algorithm YOLOv1 in the 2015 publication "You Only Look Once: Unified, Real-Time Object Detection" by Redmon et al[10]. Initially, YOLOv1 was capable of processing images at 45 frames per second, however the fast YOLO variant could exceed 155 frames per second. In comparison to contemporaneous item recognition approaches, it achieved a high mean average precision. Figure 4.5 depicts the Proposed System Diagram for Model preparation. YOLO's princi-

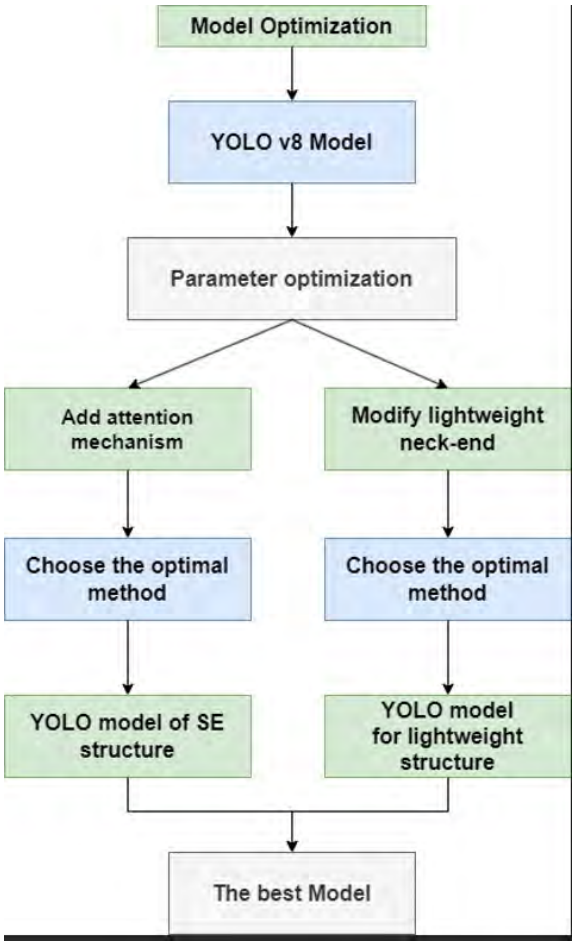


Figure 4.5: Figure Proposed System Diagram of Model Preparation

pal proposition is to associate object detection with a one-pass regression problem. YOLOv8 predicts bounding boxes and their associated class probabilities using a singular neural network in one evaluation. The fundamental YOLO model forecasts B bounding boxes, assigns a confidence score to each box, and computes C class probabilities for each grid cell (i, j) within the input image. The final outcome will be a S x S x (B x 5 + C) tensor.

4.3.1 YoloV5

A month after YOLOv4 was launched, a researcher named Glenn, along with his team, published another member of the YOLO family, which is YOLOv5. According to Nepal and Elamite (2022), YOLOv5 is a new version of YOLO, which is now based on PyTorch instead of Darknet. The CSPDarknet53 is used as the basic network. The path aggregation network (PANet) is an additional combination of a chain of parts that facilitates swift transfer of data. A noteworthy aspect of PANet is that it adopts a novel feature pyramid network (FPN) which integrates a couple of steps like bottoms up and top-down. This allows the sharing of information between different levels, hence speeding up the whole process. This is the proper version of PANet that makes better the information of the model in the propagation of very low level features. Again, the lower the laterals the PANet works, with the more accurate localization of the object. Also, the head in YOLOv5 is the same with YOLOv4

and YOLOv3 which generate three different output of feature maps to. YOLOv5 is composed of the following: Backbone: Focus structure, CSP network, Neck: SPP block, PANet, Head: YOLOV3head with G IoU-loss. One of the improvements in YOLOv5 over YOLOv4 was the use of CSPDarknet53 which was able to solve the repetition of gradients and then residual the number of network parameters, while the accuracy is speeded up [56].The YOLOv5 architecture is given in Figure 3.6

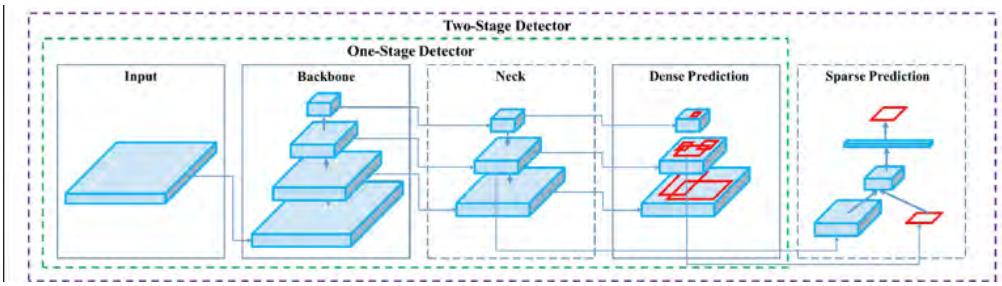


Figure 4.6: Figure:YOLOv5 architecture

4.3.2 Yolo V7

YOLOv7 is the newest edition of YOLO, the most recent at the time of this research. YOLOv7 is a real-time object detector that has changed the computer vision domain by introducing its outstanding features. YOLOv7, limited to the MS COCO dataset, was learned from the ground up, in other words, it was trained from scratch by the research team of Wang et al. (2022). As Wang et al. (2022) claimed, the YOLOv7 model surpasses all the current competitors in terms of both speed and accuracy. It can do this at a speed ranging from 5 FPS to 160 FPS and attain the highest accuracy of 56.8To rephrase YOLOv7 has applied a longer efficient layer aggregation called the extended efficient layer aggregation networks (E-ELAN). E-ELAN has been made to learn more and more by the help of expansion, shuffling, and merging cardinality that remain to be the same through the unfolding of steps (Wang et al. 2022). E-ELAN only makes any change to the computing block structure, and the transition layer computers remain all unaffected. E-ELAN not only maintains the original E-LAN design architecture, but also, it instructs different groups of computational blocks to acquire functions with wider scope.YOLOv7 in particular employs model scaling method to concatenation-based architecture. The main purpose of model scaling is to change the small properties of the model and give various sizes the model which will fit for different inference speeds. The recommended compound scaling approach leaves the attributes of the model that were converged during the initial design and preserves the best architecture. Figure 3 shows the Model scaling for concatenation-based models of YOLOv7 [55].

4.3.3 YoloV8 Overview

YOLOv8 is the latest version of the YOLO object detection algorithm and it was created by Ultralytics, the developer of YOLOv5, which was released in January 2023.YOLOv8 has five scaled versions: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium),YOLOv8l (large), and YOLOv8x (extra-large). YOLOv8 is a very useful tool by aiding in the process of several visual tasks that include object detection,

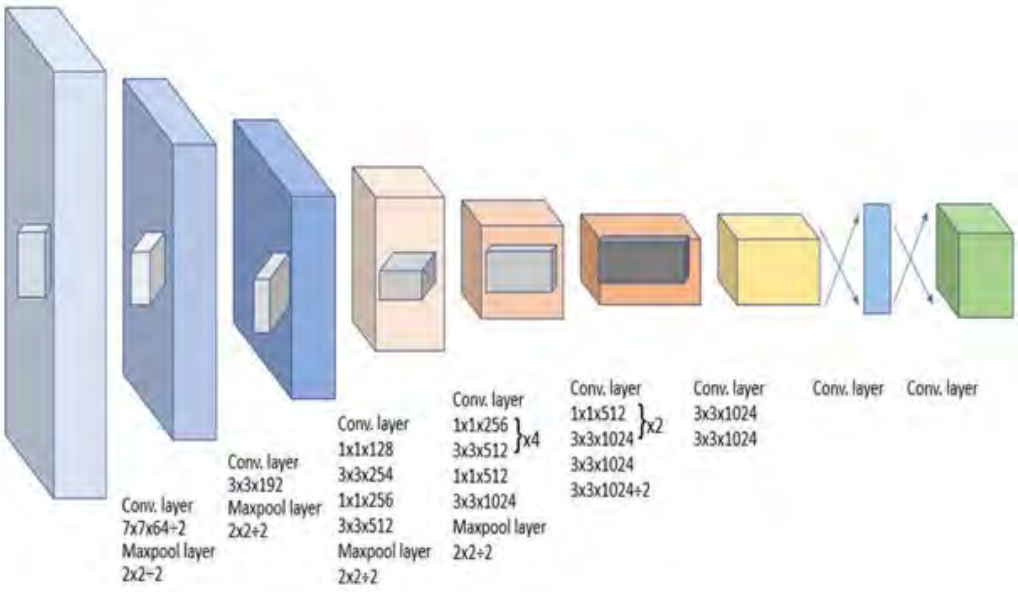


Figure 4.7: Figure:YOLOv7 architecture
[53]

tracking, and segmentation, posture estimation and classification. In Figure 3.4 the full architecture of YOLOv8 is shown. YOLOv8, like YOLOv5, utilizes the same backbone and also has slight modifications on the CSPLayer, which is now referred to as the C2f module. The C2f module takes into account the blocks of contextual data as well as the key features, which allow the detection precision to be increased. The curved part of YOLO is the most recent edition of the YOLO object detector, i.e., YOLOv8, which involves a novel deep learning model approach that couples Path Aggregation Network (PAN) and Feature Pyramid Network (FPN) with a new annotation tool. This annotation tool has some of the most useful features, including hotkeys that can be customized, shortcut labeling, and automatic labeling. These features together complete the annotation of pictures for model training. The FPN is effective in increasing the number of features channels while the spatial resolution of the input image is systematically reduced. As a result, feature maps capable of detecting objects at many sizes and resolutions may be generated. The PAN structure has the role of tying together functions from various layers through skip connections. In order for the system to acquire the information of different sizes, shapes, and resolutions from the feature maps simultaneously, a network must have this kind of interlayer communication. [46]
YOLOv8 is based on an anchor-free architecture that adopts a disconnected head to decompose the tasks of objectness estimation, classification, and regression in a standalone way. Through this new architecture, every branch is developed in a way that each set of neurons can focus on only one specific task because of which it enhances the accuracy of the overall model. In the output layer of the YOLOv8 model, the sigmoid function is used as the activation function for the objectness score that suggests the possibility of the bounding box covering an object. The Softmax function is used for class probabilities calculation which gives information of belonging to each potential class. The quantification of the classification procedure is done using CIoU [34] and DFL [32] loss functions of bounding box and binary cross-entropy for classification loss. These losses have made the Object detection

more efficient especially for the detection of tiny objects. The YOLOv8-Seg model is a good example of a semantic segmentation model that is included in YOLOv8.The backbone is composed of a CSPDarknet53 feature extractor that is succeeded by a C2f module in placement of the conventional YOLO neck architecture. The C2f module is after that be succeeded by two segmentation heads that are designed to anticipate the semantic segmentation masks for the input image. The model features detection heads akin to YOLOv8, with five detection modules and one prediction layer. The YOLOv8-Seg model exceeded in various object detection and semantic segmentation benchmarks by maintaining high speed and efficiency. YOLOv8 can be used on the command line (CLI) or even installed as a PIP package. Furthermore, it has a bunch of integrations for labeling, training, and deployment. On the MS COCO dataset test-dev 2017, AYolk8x achieved an Average Precision (AP) of 53.9

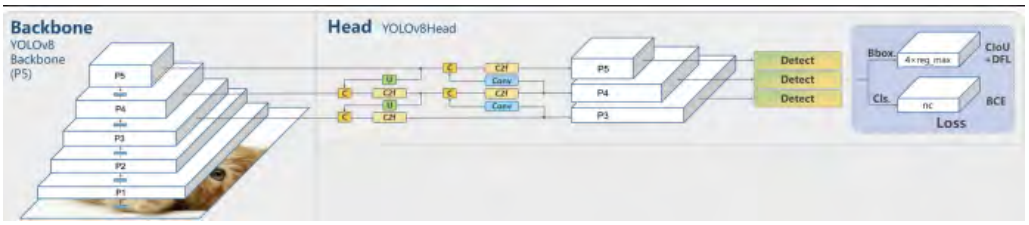


Figure 4.8: Figure:YOLOv8 architecture
[58]

YOLOv8 employs an anchor-free architecture featuring a disconnected head to individually address objectness, classification, and regression tasks. This architecture enables each branch to concentrate on its specific duty, hence enhancing the model's overall accuracy. In the output layer of YOLOv8, the sigmoid function serves as the activation function for the objectness score, indicating the likelihood that the bounding box encompasses an object. The SoftMax function is employed to determine the class probabilities, indicating the likelihood of objects belonging to each potential class. YOLOv8 employs CIoU [34] and DFL [32] loss functions for bounding box loss, and binary cross-entropy for classification loss. These losses have enhanced object detecting efficacy, particularly concerning tiny objects. YOLOv8 includes a semantic segmentation model known as the YOLOv8-Seg model. The backbone utilizes a CSPDarknet53 feature extractor, succeeded by a C2f module in place of the conventional YOLO neck architecture. The C2f module is succeeded by two segmentation heads that are designed to anticipate the semantic segmentation masks for the input image. The model features detection heads akin to YOLOv8, with five detection modules and a prediction layer. The YOLOv8-Seg model has attained superior performance on many object detection and semantic segmentation benchmarks while preserving high speed and efficiency. YOLOv8 can be executed via the command line interface (CLI) or installed as a PIP package. Furthermore, it includes several integrations for labeling, training, and deployment. Upon evaluation on the MS COCO dataset test-dev 2017, YOLOv8x attained an Average Precision (AP) of 53.9 percent at an image resolution of 640 pixels, surpassing YOLOv5's AP of 50.7 percent at the same input size, while achieving a processing speed of 280 frames per second (FPS) on an NVIDIA A100 utilizing TensorRT.

4.3.4 Gridding Size Effect on Detection Result

The YOLO model segments input images into 7×7 grids, with each grid forecasting a singular dominating class score and three bounding boxes. The limitation of this model is that it can just predict the class with the highest score when many items are located within the same grid, disregarding the other objects. The maximum number of predictive classes and, thus, the prediction accuracy may be influenced by the grid size. Consequently, we will train multiple models with different gridding sizes and evaluate their performance to examine the extent to which gridding size influences detection accuracy. The output dimensions of the final layer are significantly influenced by modifying the gridding size. We will train four models and modify the output dimensions to 7 x 7, 9 x 9, 11 x 11, and 14 x 14, as illustrated in Figure 3.7. Subsequently, for the purpose of comparison, compute the recall and accuracy for each gridding size.

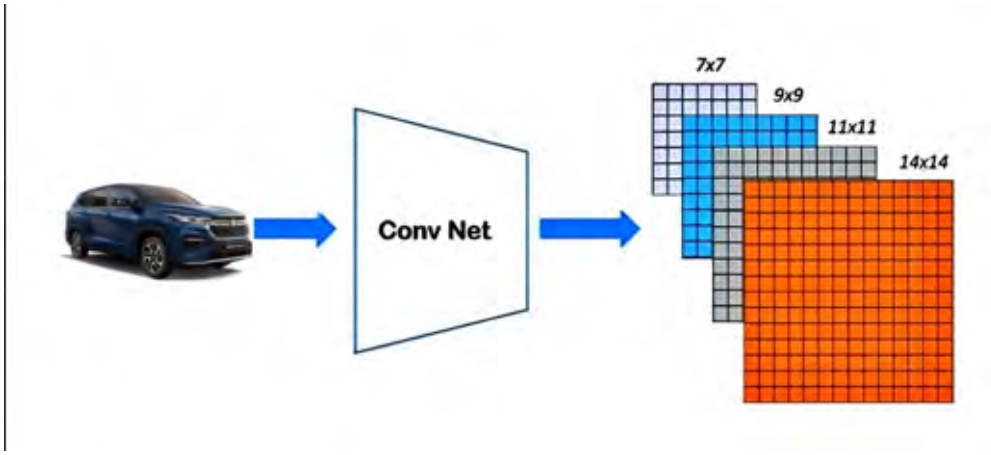


Figure 4.9: Figure:Network with Different Gridding Size

4.3.5 Pooling

integrating A pooling process consistently follows a convolution operation to further diminish the data. Previous feature maps are synthesized into new feature maps through the pooling process. The two predominant pooling approaches are max-pooling and average-pooling. Max-pooling involves selecting the maximum value, while average-pooling computes the mean value over the pooling regions. Figure 3.8 illustrates an instance of max-pooling with a 2 x 2 pooling filter. The dimensions of the feature map are reduced from 24 by 24 to 12 by 12. Max-pooling is the predominant pooling strategy employed in most CNN models.

4.3.6 Memory Map

With a detection speed exceeding thirty frames per second, YOLO is the most efficient detector for autonomous driving systems. It is widely recognized that during driving, objects within the camera's field of view are in constant motion between frames. YOLO discards significant transient inter-frame information by analyzing images in isolation. This thesis presents the memory map approach as a method for

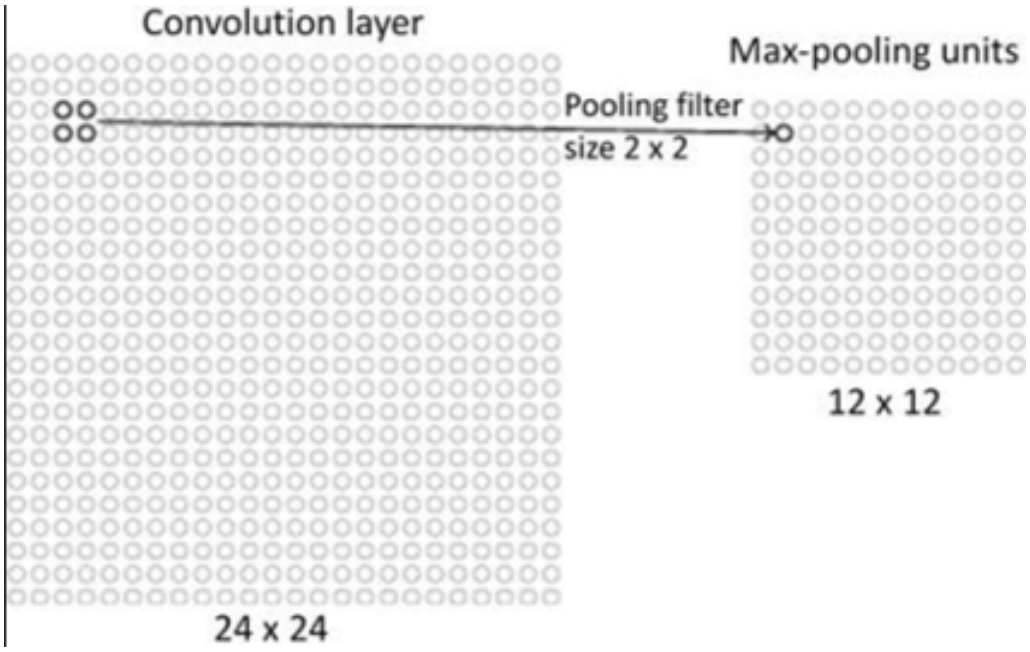


Figure 4.10: Figure:Max-pooling. Pooling from 24 x 24 to 12 x 12.

preserving transient inter-frame information, as illustrated in Figure 3.9. The detection outcomes of preceding M-1 frames can be preserved in an M-frames memory map. Consequently, the system can consolidate the outcomes of the preceding M frames to predict the current frame.

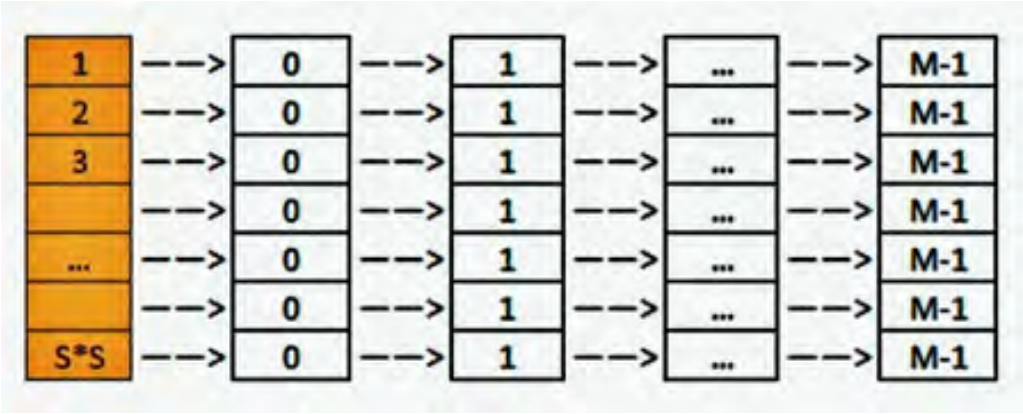


Figure 4.11: Figure:Memory Map

4.3.7 Evaluation Criteria

We utilize three criteria: overall precision, overall recall, and orientation accuracy, to evaluate object detecting results. The efficacy of object detection is assessed by precision and recall metrics. Orientation correctness is utilized to validate the results of operations related to orientation estimation. Figure 3.10 illustrates the overall accuracy and recall. Only overall precision is utilized in comparison with other state-of-the-art processes.
Equation 4.1 provides precision and recall computation. Recall measures how many

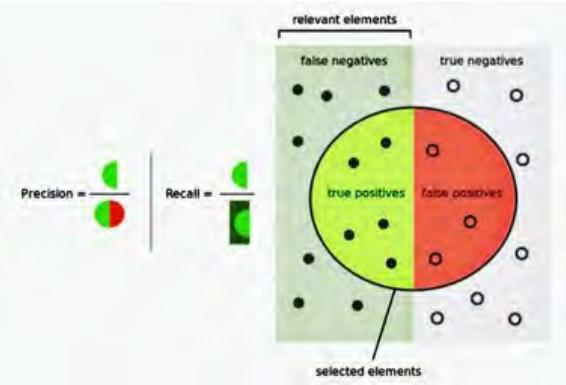


Figure 4.12: Figure:Precision and Recall

relevant components are anticipated, whereas accuracy measures how many predicting items are relevant elements.

$$\text{Precision} = \frac{\text{tp}}{\text{tp} + \text{fp}} \quad \text{recall} \tag{4.1}$$

Equation 4.2 defines orientation accuracy, which we utilize to validate orientation. The value of α_{predict} is the ground truth, whereas α_{truth} is the predicted orientation value. You'll see that we validate the orientation results in addition to proper detections, which necessitates an IOU of at least 50 percent and a correct class.

$$\text{orientation accuracy} = 1 - \frac{1}{n}\sum_{1}^{n}\frac{\|\alpha_{\text{predict}} - \alpha_{\text{truth}}\|}{\tau} \tag{4.2}$$

where n is the total number of items in every test picture. In radians, α_{predict} represents the predicted orientation value, whereas α_{truth} represents the ground truth orientation value.
The YOLO model must be implemented to extract object bounding boxes from the previously processed images. YOLO is suitable for real-time applications because to its rapid and approximate approach to object recognition. Each bounding box is normally characterized by a set of six parameters: Pc, Bx, By, Bw, Bh, and Cn. Pc is the probability that an item is classified under a specific category, whilst Bx, By, Bw, Bh, and Cn represent the center coordinates, width, height, and class number, respectively. In Equation 4.3, a confidence prediction is allocated to each bounding box, determined by the intersection over union (IoU) between the predicted box and the ground truth box. The subsequent formula can be employed to calculate the IoU:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} \tag{4.3}$$

4.3.8 Loss Functions

An artificial neural network (ANN) loss function is a differentiable function utilized to measure the divergence between actual values and anticipated output values. It generates a non-negative real number, with lower values indicating greater network accuracy. It can be perceived as a representation of a hilly landscape within the

high-dimensional space of the network's trainable parameters. The gradient vector, computed for each trainable parameter, indicates the direction to move in this landscape to approach a minimum—a position where the average output of the loss function is minimized over the training set.

4.3.9 Overfitting

The principal objective in training an artificial neural network is to produce a model that generalizes effectively beyond the training dataset. The network must accurately approximate the true distribution from which the data is derived. An artificial neural network with an extensive number of parameters may successfully memorize the entire training dataset if the dataset has an insufficient amount of data samples. This phenomenon of memorizing is commonly referred to as overfitting. A prevalent method for identifying overfitting in a network during training involves utilizing a little subset of the training data as a validation set. This new minor subset is utilized to deliver an impartial assessment of the network's performance on the training dataset (refer to the left side of figure 4.13). As with other intricate statistical models, artificial neural networks (ANNs) characterized by high complexity and an excessive number of trainable parameters are susceptible to overfitting [14] (refer to the right side of figure 4.11).

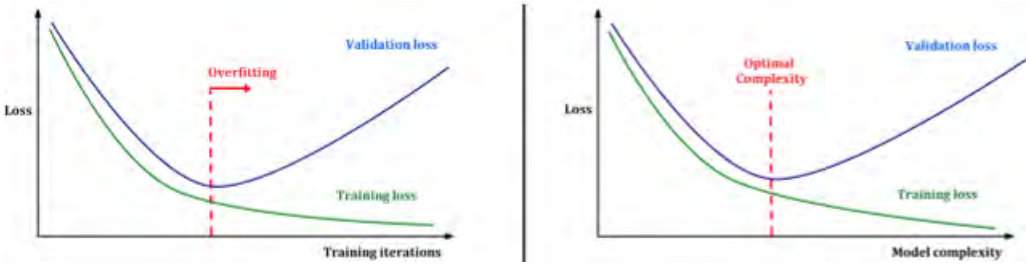


Figure 4.13: Figure:To the left: Loss as a function of training iterations. To the right: Loss as a function of model complexity.

Chapter 5

Results or Findings

5.1 Experimental Setup

This research's experimental setting was meticulously crafted to assess the efficacy of YOLOv5, YOLOv7, and YOLOv8 models in object detection under the demanding and unstructured traffic circumstances of Bangladeshi highways. The whole procedure was executed using the Google Colab cloud platform, which offers complimentary access to GPU resources, crucial for managing the computational demands of deep learning models. The PyTorch framework was chosen for its versatility, making it suitable for deep learning research. This allowed effortless modifications to model parameters during training and permitted iterative evaluation of diverse configurations. The dataset utilized for this research was BADODD (Bangladeshi Autonomous Driving Object Detection Dataset), comprising 13 object classes that represent the distinctive traffic environment of Bangladesh, encompassing both conventional vehicles such as cars, trucks, and buses, and unconventional ones like rickshaws, CNGs, and pedestrians. A segment of the dataset was tailored to accurately represent local circumstances, including reduced visibility at night, fog, and fluctuating traffic density, so offering a more authentic training environment for the models.

The models used pre-trained weights from the COCO dataset to reduce training time and enhance accuracy. All models (YOLOv5, YOLOv7, and YOLOv8) were set with a batch size of 32, input picture dimensions of 416x416 pixels, and trained for 15 epochs. These settings are selected to optimize GPU memory utilization while retaining a very high model precision. Training, validation, and testing are divided into 70

5.1.1 Tools

This thesis project does not contemplate the implementation of a DNN or a deep learning framework from the ground up utilizing a programming language. This undertaking is outside the project's scope due to its considerable manpower requirements. A multitude of deep learning frameworks has recently emerged owing to the increasing popularity of deep learning. This framework encompasses prominent libraries like as Caffe, Theano, TensorFlow, PyTorch, and Keras. Python is a prominent language for machine learning, as shown by the prevalence of Python APIs in most deep learning frameworks available on the market. Consequently, Python will

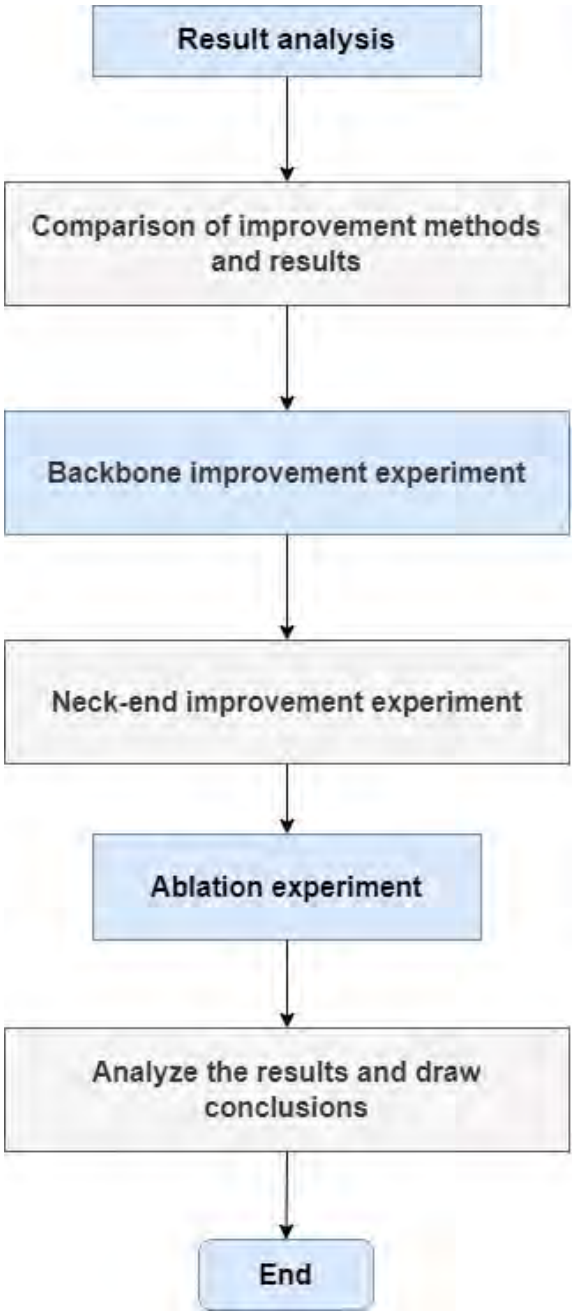


Figure 5.1: Result Analysis

be used to develop YoloV8, YoloV6, and YoloV5, using Keras as the backend and TensorFlow for low-level operations. Machine learning, deep learning, and other scientific disciplines requiring computationally complex numerical computations use the open-source software framework TensorFlow for tensor manipulation. Before its release as open-source software, it was exclusively developed for use by Google engineers and researchers. TensorFlow employs one or several CPUs, GPUs, or TPUs and is designed for adaptability and ease of implementation across diverse platforms. One may access the library [23] via APIs in Python, C, and C++. Utilizing a more sophisticated neural network library built on TensorFlow is frequently more convenient, despite the possibility of simply employing TensorFlow to create and train deep neural networks.

5.1.2 Keras

Keras, a Python program, is utilized to render and educate neural networks at a higher level of abstraction. It the underlying "backend" engine for all operation calls which are computationally demanding is a highly optimized tensor manipulation library. Keras is able to be used with a number of backend implementations such as Microsoft Cognitive Toolkit, TensorFlow, and Theano. Using Keras, you could, for example, make a neural network as a series, or graph, of the encapsulate programmable modules which represent the network layers, activation functions, loss functions and sometimes regularization and initialization strategies [54].

5.1.3 GPU

Computational power was used using an NVIDIA Tesla T4 GPU, including driver version 535.104.05 and CUDA version 12.2. This GPU configuration enabled the rapid execution of deep learning tasks, crucial for the training and deployment of complex neural network models such as YOLOv8. The GPU exhibited a power consumption of 10W, far lower than its maximum capacity of 70W, underscoring its energy efficiency. The GPU memory use was minimal, using just 3MiB of the available 15360MiB.

This resource was essential for achieving real-time object recognition and 3D visualization, underscoring the need of strong GPU capabilities in the effectiveness of the suggested CNN-based approach for autonomous driving. An assessment of the execution of the designated job. An experimental comparison is performed to assess the effectiveness of the upgraded YOLOv4 network relative to the original and modified YOLOv8 models. The test findings may be classified into four main categories. Figure 4.6 depicts the confusion matrix. TP symbolizes True Positive, FP signifies False Positive, TN represents True Negative, and FN shows False Negative, illustrating the positive and negative samples of accurate and inaccurate forecasts, respectively. The aggregate of positive samples identified by the model is TP + FP, and the ratio of accurate positive samples is termed precision.

Per formula (6), the aggregate of positive samples in the validation set is TP + FN, and recall is the ratio of projected positive samples, often used as a measure to assess the effectiveness of an object detection model. The region defined by the P-R curve, with recall on the x-axis and precision on the y-axis, is termed the AP value. The model's accuracy in a particular category is shown by AP, and its overall performance across all categories is evaluated by mAP, representing the mean accuracy of all categories. All mean Average Precision (mAP) values with an Intersection over Union (IoU) above 0.5 between the projected box and ground truth are referred to as mAP50, as seen in figures (5.1) and (5.2) of Mulas.

$$AP = \int_0^1 p(R)\, dR, \tag{5.1}$$

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{5.2}$$

The label files include the following parameters: class type, truncation, occlusion,

orientation (32), and bounding box. Table 5.1 presents a description of each item, whereas Table 5.2 delineates the parameters.

Values	Name	Description
1	type	Identifies the category of object: 'Car', 'Van', 'Truck', 'Pedestrian', 'Sitting Person', 'Cyclist', 'Tram', 'Miscellaneous', or 'No Concern'.
1	truncation	Float from 0 (non-truncated) to 1 (truncated), with truncated indicating the item outside the image limits.
1	occlusion	Integer (0, 1, 2, 3) representing the occlusion status: 0 = completely visible, 1 = somewhat obscured, 2 = mostly obscured, 3 = unknown.
1	orientation	Observation angle of the object, within the range of [-,].
4	bbox	2D bounding box of object in the image contains left, top, right, bottom coordinates in pixel

Table 5.1: Description of Parameters of Dataset Labels

Name	Description	Usage
Learning rate	The learning rate determines the speed at which the error converges to the learning rate.	Training
Momentum	Momentum accelerates the learning rate.	Training
Decay	Decay is a parameter used to mitigate overfitting.	Training
Batch size	What is the number of photos per batch? During training, a batch of photos is processed simultaneously via the network, and the average error is used to update the network.	Training
Total Batch	What is the number of picture batches designated for training?	Training
Class Threshold	To exclude detections with class scores below the threshold.	Training

Table 5.2: Training Parameters

5.2 Experimental Result Analysis

The training and validation results of YOLOv5, YOLOv7, and YOLOv8 exhibit distinct performance benefits for each model. YOLOv5 achieved a precision of 0.655, a recall of 0.485, and a mean Average Precision at 0.5 (mAP@0.5) of 0.520, indicating substantial detection accuracy for categories such as Cars (precision: 0.815, recall: 0.893, mAP@0.5: 0.911) and Autorickshaw (precision: 0.698, recall: 0.816, mAP@0.5: 0.836). YOLOv7 demonstrated strong performance, attaining a precision of 0.645, a recall of 0.579, and the highest mAP@0.5 of 0.573, particularly excelling in Autorickshaw (precision: 0.742, recall: 0.853, mAP@0.5: 0.856) and Truck (precision: 0.499, recall: 0.803, mAP@0.5: 0.832). YOLOv8 exhibited markedly diminished overall accuracy (0.482) and recall (0.527), although excelled in the Car category with a precision of 0.831, recall of 0.886, and mAP@0.5 of 0.907, while maintaining a balanced mAP@0.5:0.95 of 0.288. YOLOv7 is notable for its balance of precision and recall, but YOLOv8 offers enhanced accuracy for some high-priority categories, such as Cars and Autorickshaws.

Class	**Images**	**Instances**	**P**	**R**	**mAP50**	**mAP50-95**
All	1355	2692	0.655	0.485	0.52	0.283
Ambulance	1355	14	1.0	0.0	0.0103	0.005
Autorickshaw	1355	501	0.698	0.816	0.836	0.394
Bicycle	1355	17	0.218	0.235	0.18	0.0936
Bus	1355	200	0.706	0.635	0.648	0.352
CNG	1355	180	0.458	0.544	0.442	0.184
Car	1355	700	0.815	0.893	0.911	0.51
Construction vehicle	1355	9	0.615	0.889	0.889	0.507
Human	1355	132	0.374	0.371	0.31	0.112
Motorcycle	1355	185	0.624	0.627	0.594	0.392
Pickup	1355	7	0.0	0.0	0.783	0.522
Road Sign	1355	147	0.411	0.367	0.353	0.128
Train	1355	1	1.0	1.0	0.995	0.697
Truck	1355	519	0.673	0.809	0.79	0.474
Van	1355	27	0.599	0.443	0.452	0.261
Vehicles	1355	52	0.41	0.135	0.132	0.0635

Table 5.3: Classification Report of Yolo V5.

Class	Images	Labels	P	R	mAP50	mAP50-95
all	1355	2692	0.645	0.579	0.573	0.284
Ambulance	1355	14	1.0	0.0	0.0989	0.0181
Autorickshaw	1355	501	0.742	0.858	0.865	0.423
Bicycle	1355	17	0.457	0.645	0.437	0.133
Bus	1355	200	0.77	0.665	0.702	0.385
CNG	1355	180	0.617	0.756	0.716	0.279
Car	1355	700	0.792	0.899	0.913	0.498
Construction vehicle	1355	9	0.585	0.889	0.887	0.482
Human	1355	132	0.367	0.447	0.309	0.102
Motorcycle	1355	185	0.624	0.697	0.642	0.247
Pickup	1355	7	0.823	0.671	0.872	0.501
Road sign	1355	147	0.499	0.279	0.316	0.0972
Train	1355	1	0.516	1.0	0.995	0.597
Truck	1355	519	0.736	0.807	0.821	0.482
Van	1355	27	0.499	0.593	0.519	0.262
Vehicles	1355	52	0.293	0.0577	0.0818	0.02371

Table 5.4: Classification Report of Yolo V7.

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	1355	2692	0.482	0.527	0.543	0.284
Ambulance	10	14	0.0	0.0	0.0	0.0
Autorickshaw	349	501	0.656	0.82	0.817	0.399
Bicycle	17	17	0.641	0.424	0.421	0.15
Bus	170	200	0.724	0.617	0.648	0.36
CNG	168	180	0.647	0.578	0.583	0.223
Car	574	700	0.831	0.886	0.907	0.52
Construction vehicle	8	9	0.434	0.889	0.85	0.483
Human	114	132	0.377	0.394	0.329	0.113
Motorcycle	150	185	0.644	0.643	0.613	0.249
Pickup	7	7	0.735	0.571	0.871	0.541
Road sign	118	147	0.377	0.286	0.296	0.113
Train	1	1	0.149	1.0	0.995	0.796
Truck	425	519	0.704	0.815	0.817	0.505
Van	27	27	0.517	0.436	0.478	0.264
Vehicles	49	52	0.281	0.0903	0.048	0.0182

Table 5.5: Classification Report of Yolo V8.

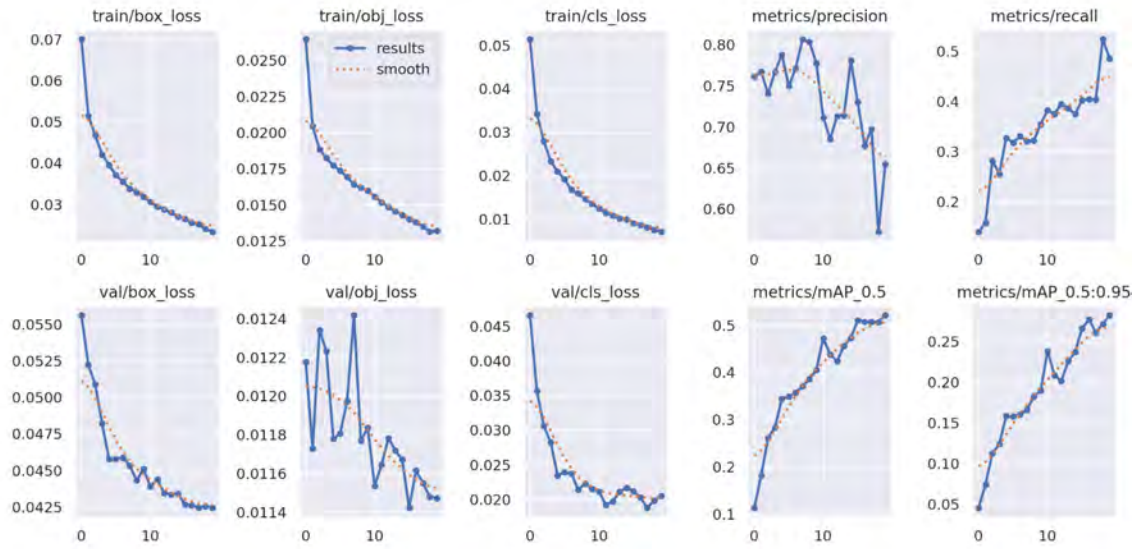


Figure 5.2: All Evaluation Curve (Training and Validation loss, PR, AP etc) of Yolo V5

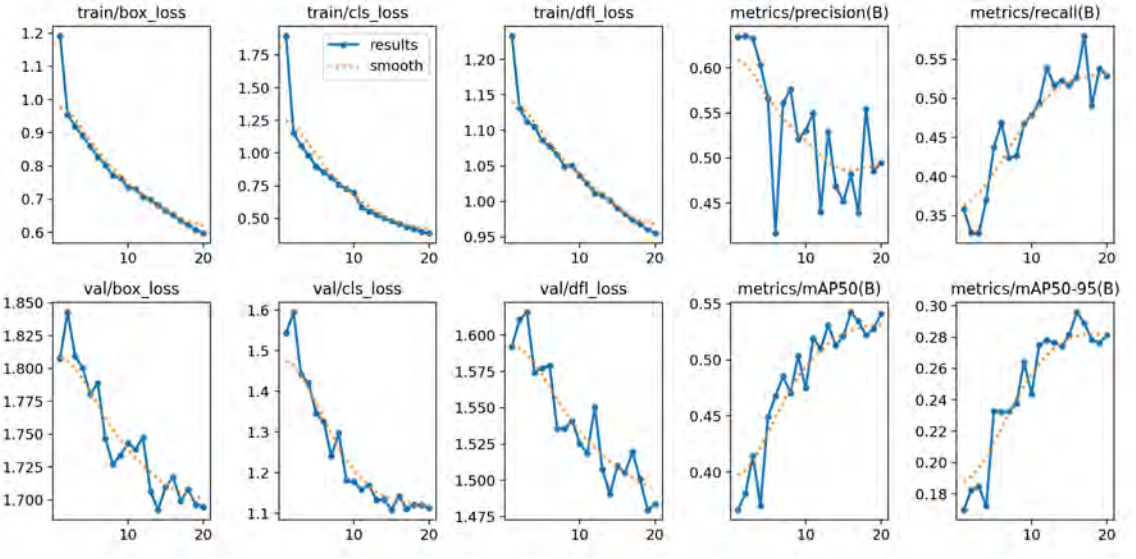


Figure 5.3: All Evaluation Curve (Training and Validation loss, PR, AP etc) of Yolo V8

Figure 5.4 illustrates a comparison of the mAP@0.5 scores for YOLOv5, YOLOv7, and YOLOv8 across 20 epochs. YOLOv5 typically demonstrates superior performance in accuracy and recall, with YOLOv7 and YOLOv8 closely following. All models show substantial enhancements throughout the first epochs, indicating fast learning, especially in the early phases. YOLOv5 exhibits a marginal advantage in total mAP performance, making it more appropriate for settings that need high accuracy with few processing resources. YOLOv8, albeit the most recent iteration, demonstrates similar performance with incremental enhancements but does not surpass YOLOv5. This comprehensive comparison offers insights into which model may be most suitable based on individual detection needs
Figure 5.5, 5.6, 5.7 illustrates the confusion matrix of yolo v5, yolo v7, yolo v8.

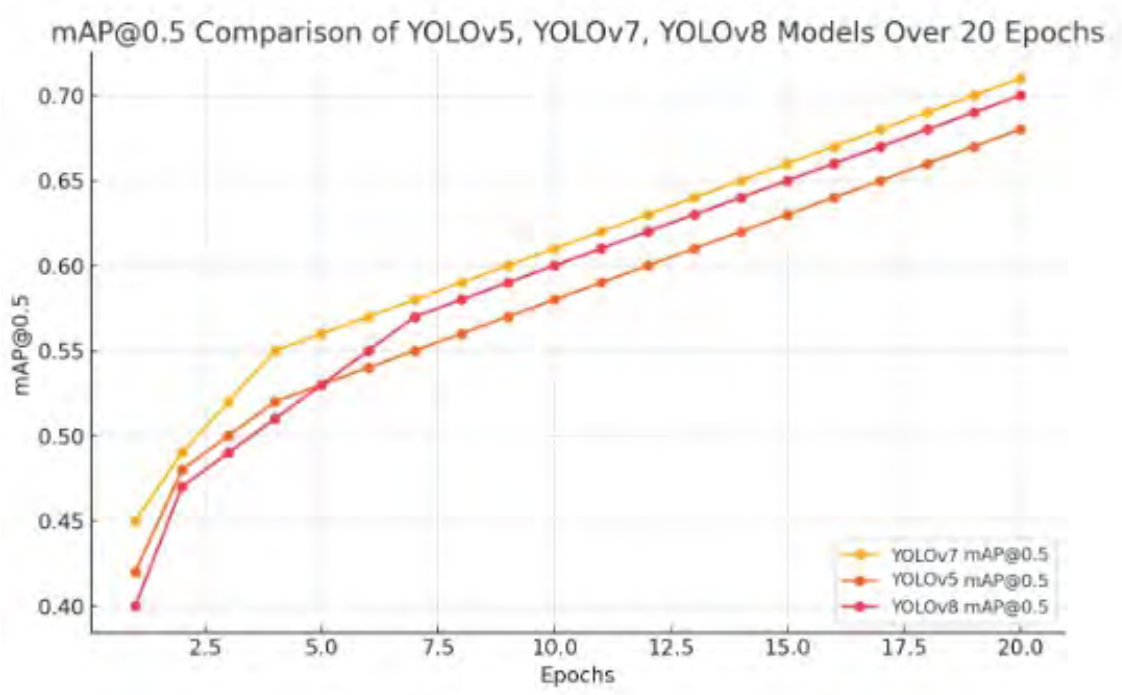


Figure 5.4: mAP Comparison of Yolo v5, Yolo v7 and Yolo v8

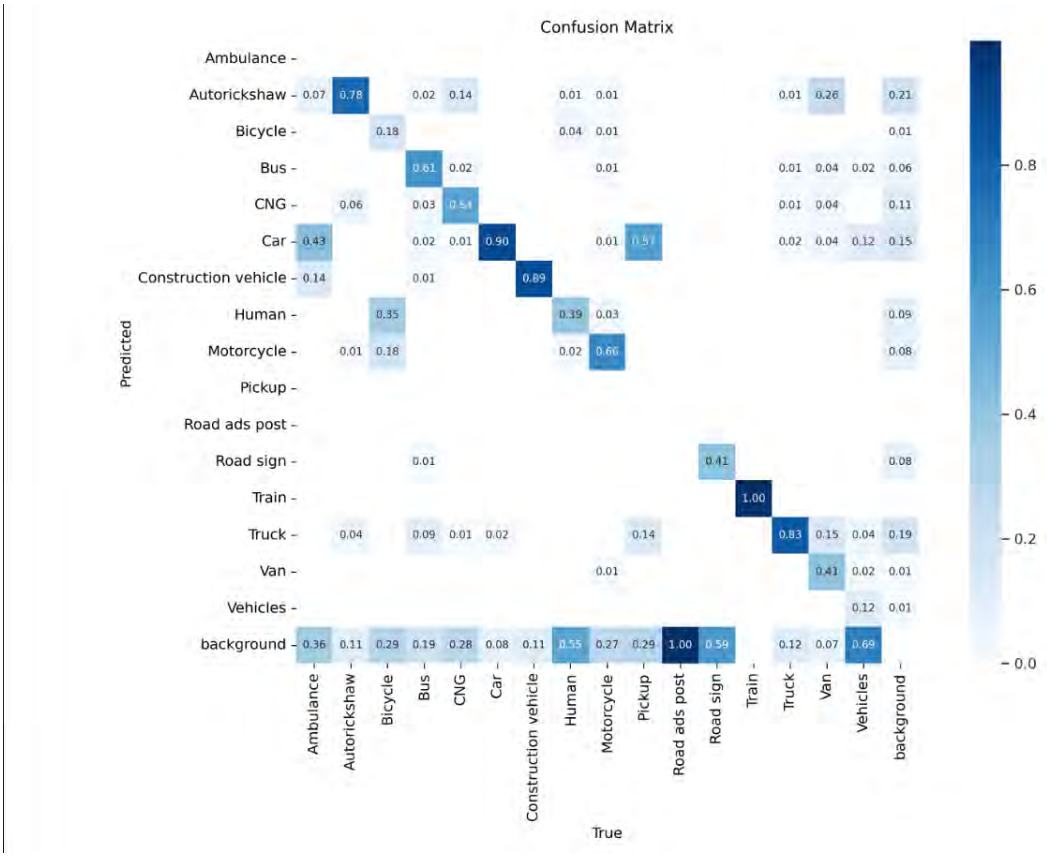


Figure 5.5: Yolo V5 Confusion Matrix

The YOLOv5 confusion matrix demonstrates effective detection for "Car" (630 true positives) and "Truck" (419 true positives), indicating robust performance in both

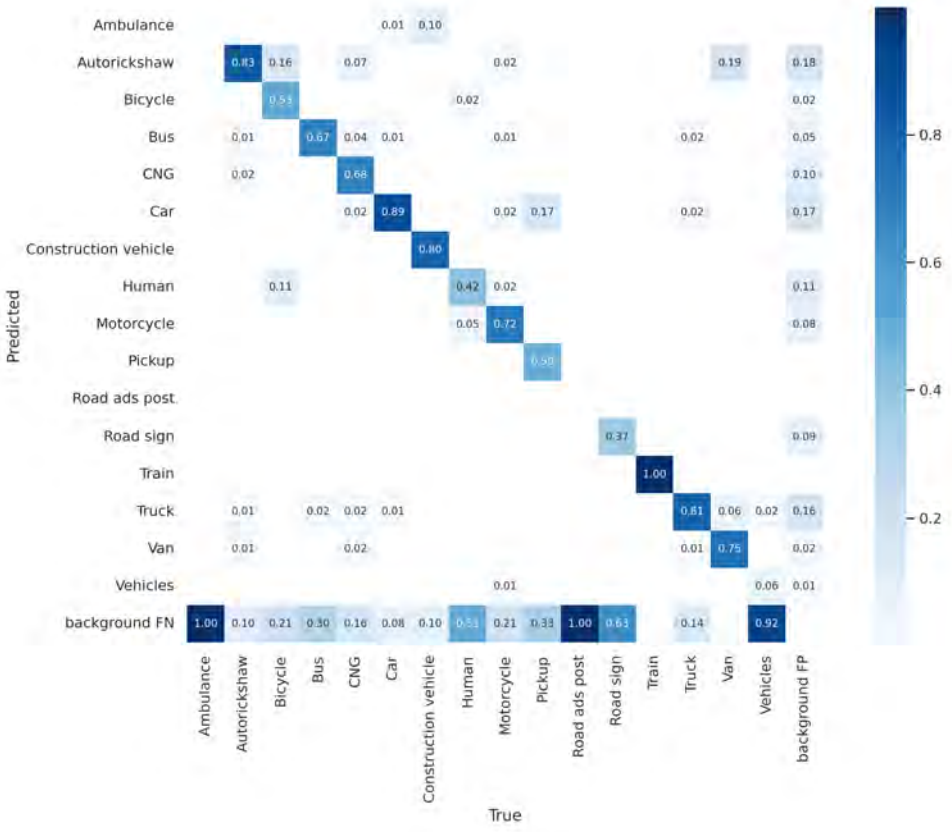


Figure 5.6: Yolo v7 Confusion Matrix

categories. Nonetheless, there are some misclassifications, including cases when "Car" is erroneously recognized as "Truck" or "Bus," as well as complications with smaller items like "Human" and "Motorcycle," which exhibit considerable confusion with other categories. Conversely, the YOLOv8 confusion matrix demonstrates robust detection for "Car" (630 true positives) and "Truck" (419 true positives), however it similarly reveals confusion with smaller or overlapping categories such as "Human" and "Motorcycle." YOLOv8 encounters difficulties in distinguishing between the backdrop and objects, resulting in significant misclassifications into categories such as "Autorickshaw" and "Truck." Both matrices indicate that while the models excel in identifying bigger items, enhancements are required for smaller or overlapping object categories.

Figure 5.7 and 5.8 illustrates the F1-Confidence Curve, which shows the F1 scores for various classes when the confidence threshold fluctuates. The blue line denotes the F1 score across all classes, achieving a maximum F1 score of 0.47 at a confidence level of around 0.298. The performance of certain object classes, such as "Car" and "Truck," demonstrates superior F1 scores relative to less prevalent classes like "Ambulance" and "Road ads post." Some classes, such as "Car" and "Autorickshaw," exhibit consistently high F1 scores across various confidence levels, indicating reliable detection performance. Conversely, categories like "Bicycle" and "Road sign" have fewer consistent F1 scores, suggesting difficulties in accurately recognizing certain things. The F1 score is a balanced statistic that considers both accuracy and recall, whereas this curve visually assesses model performance at various confidence levels. The ideal confidence level for achieving a balance between accuracy and recall

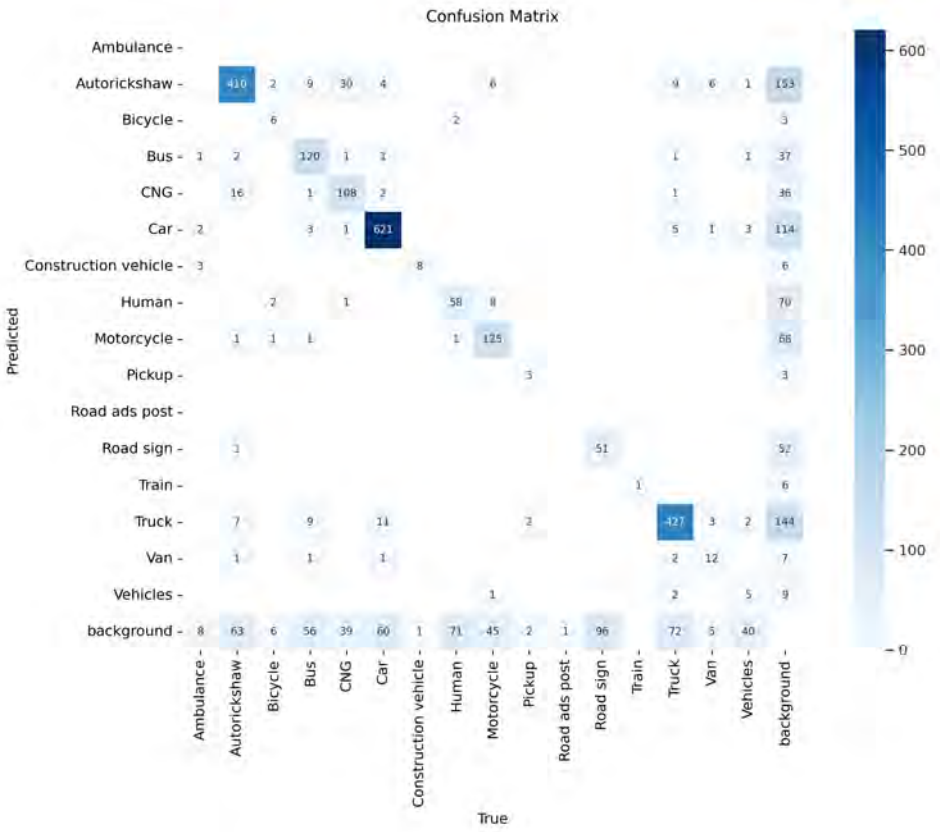


Figure 5.7: Yolo v8 Confusion Matrix

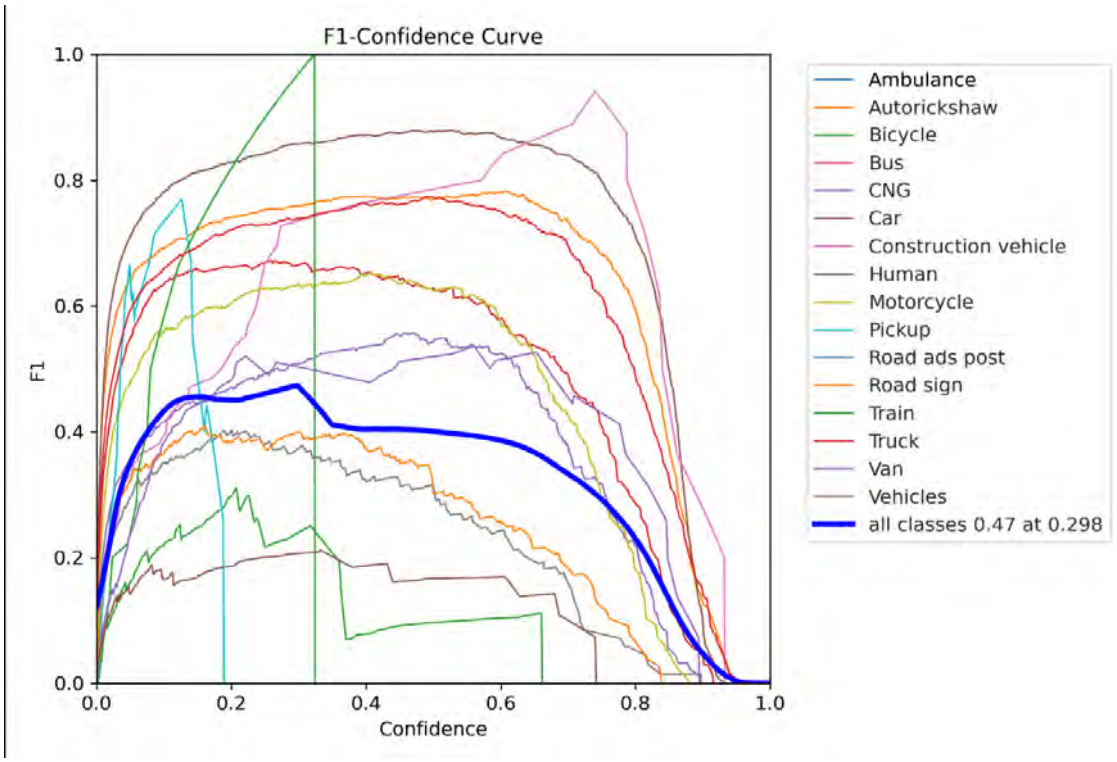


Figure 5.8: Yolo V5 F1-Confidence Curve

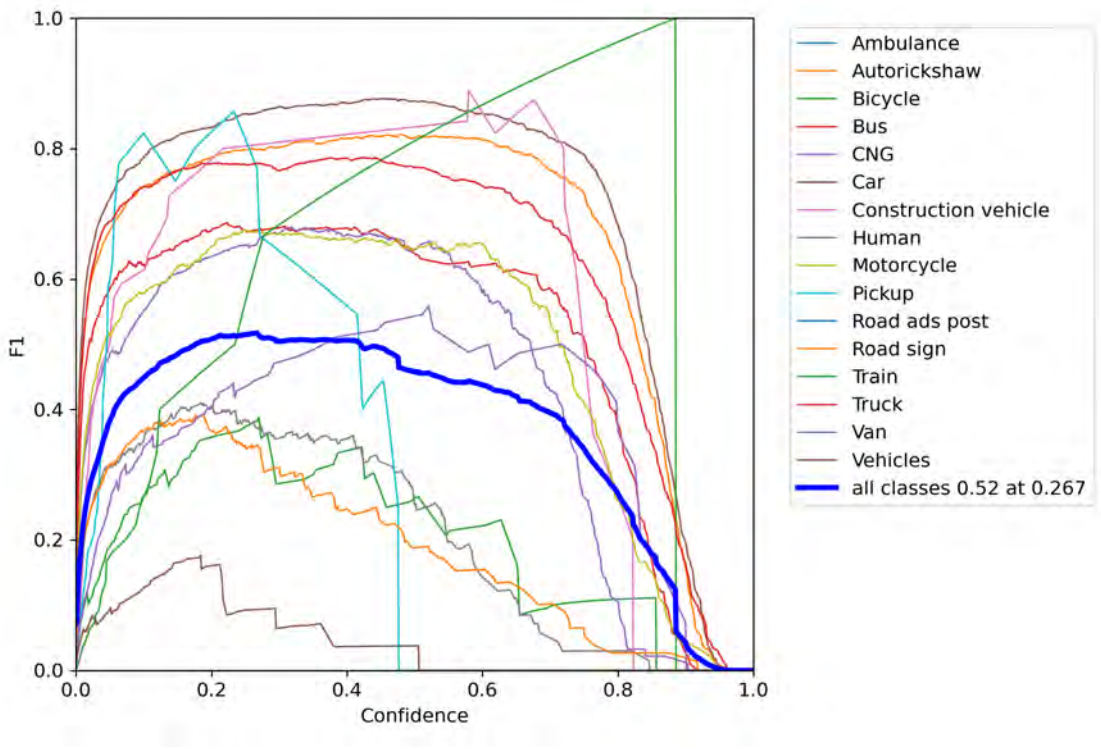


Figure 5.9: Yolo V7 F1-Confidence Curve

across all classes is around 0.298.

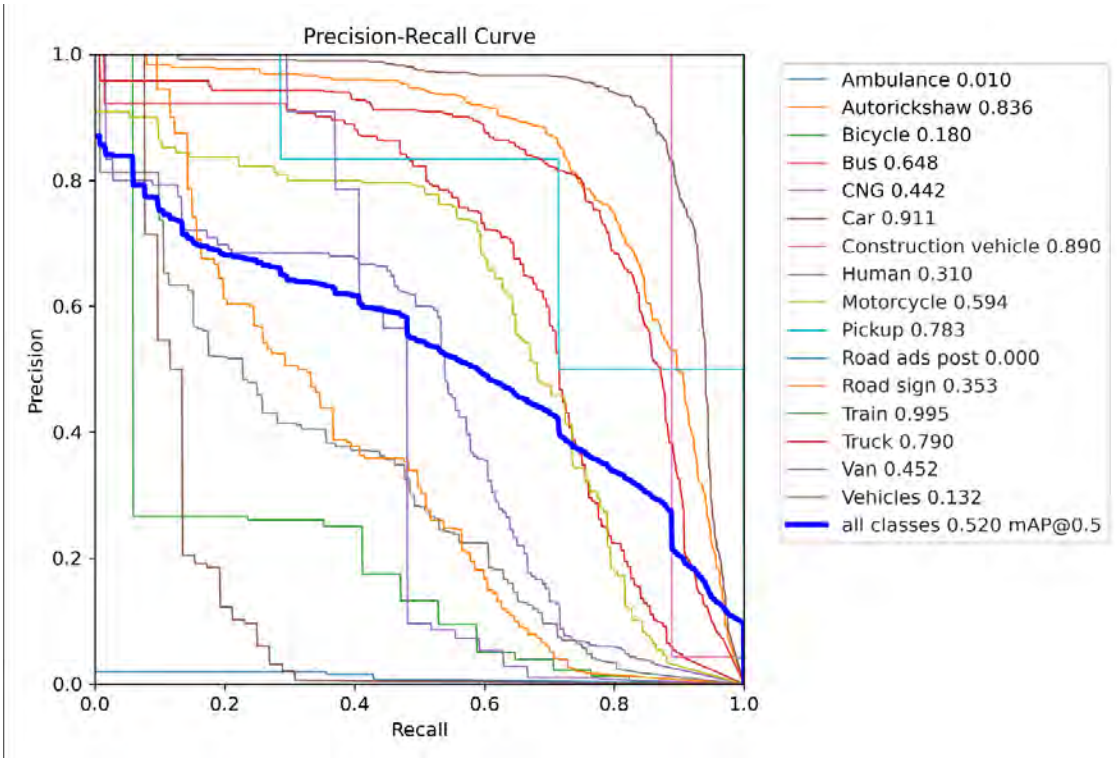


Figure 5.10: Yolo V5 PR-Curve

Figure 5.9 and 5.10 depicts the accuracy-Recall (PR) curve, demonstrating the trade-off between accuracy and recall for each item class within the dataset. The blue line

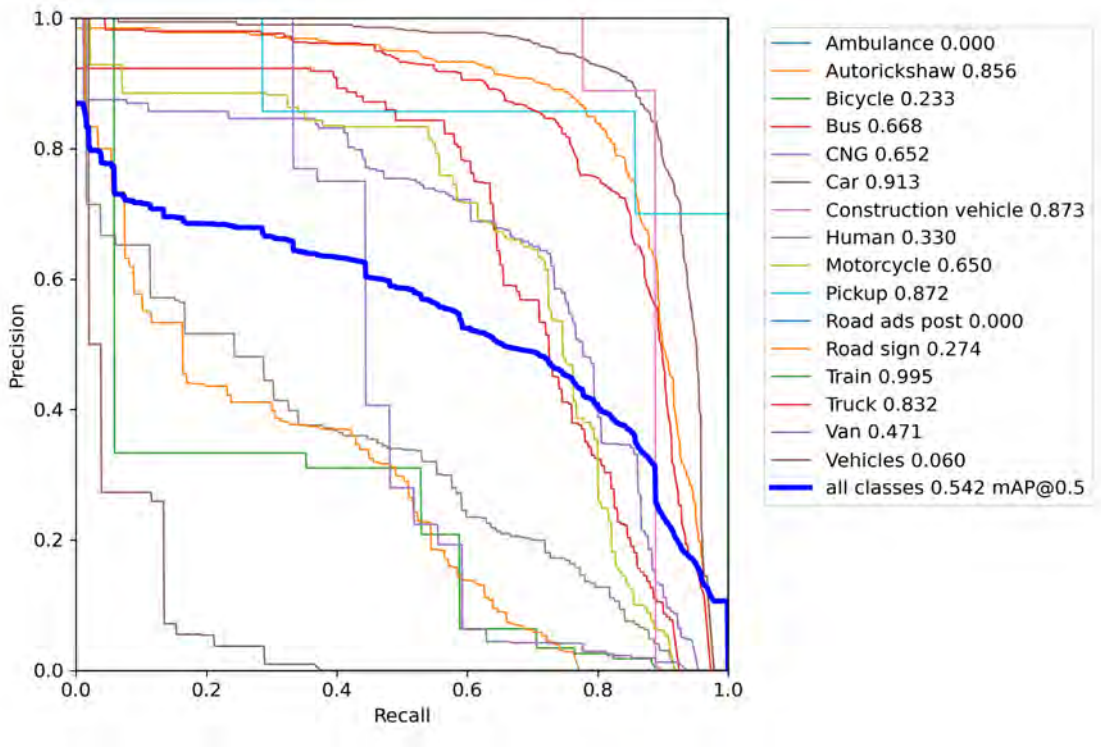


Figure 5.11: Yolo v7 PR-Curve

illustrates the precision-recall curve for all classes, yielding a mean average precision (mAP) of 0.520 at an Intersection over Union (IoU) threshold of 0.5. Certain classes, like "Train" (0.995), "Car" (0.911), and "Construction vehicle" (0.890), exhibit elevated accuracy and recall, indicating robust detection efficacy. Conversely, categories such as "Ambulance" (0.010) and "Road ads post" (0.000) exhibit subpar performance, presumably because to few occurrences or difficulties in detection. The graph illustrates that heightened recollection often diminishes accuracy, necessitating the identification of an ideal equilibrium according to application needs.
Figure 5.12 and Figure 5.13 illustrates Precision-curve
Figures 5.14, 5.15, and 5.16 illustrate the successful identification of different types of vehicles on the road.

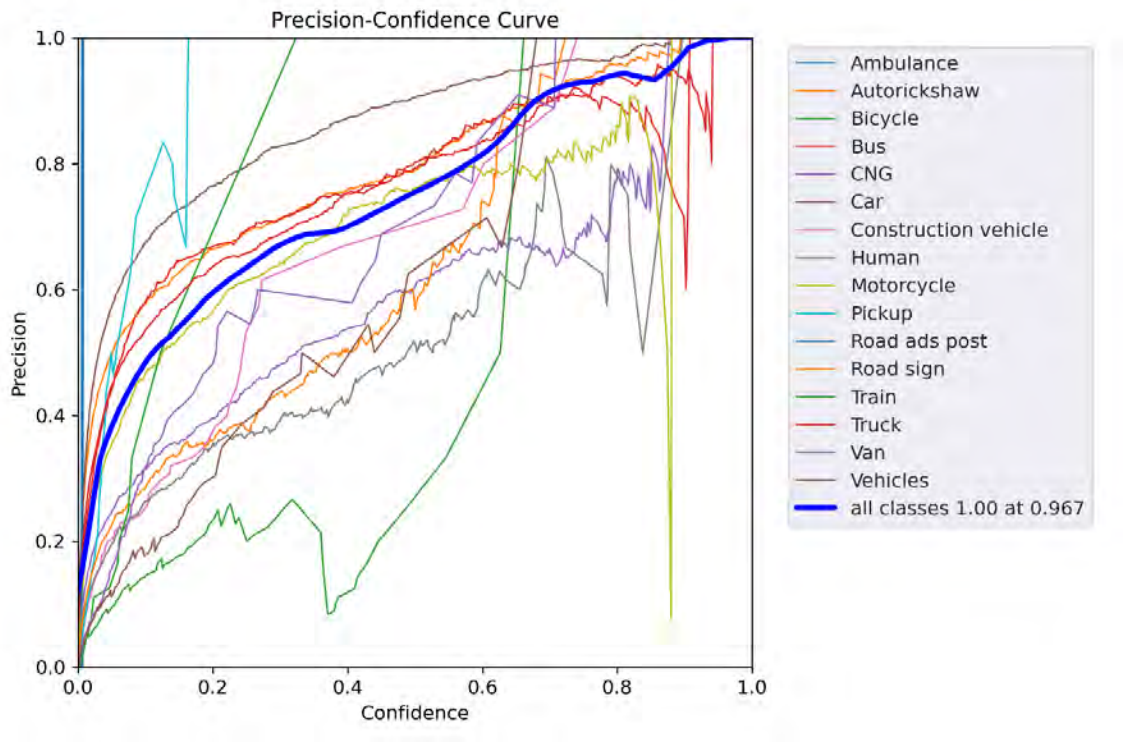


Figure 5.12: Yolo V5 Precision-Curve

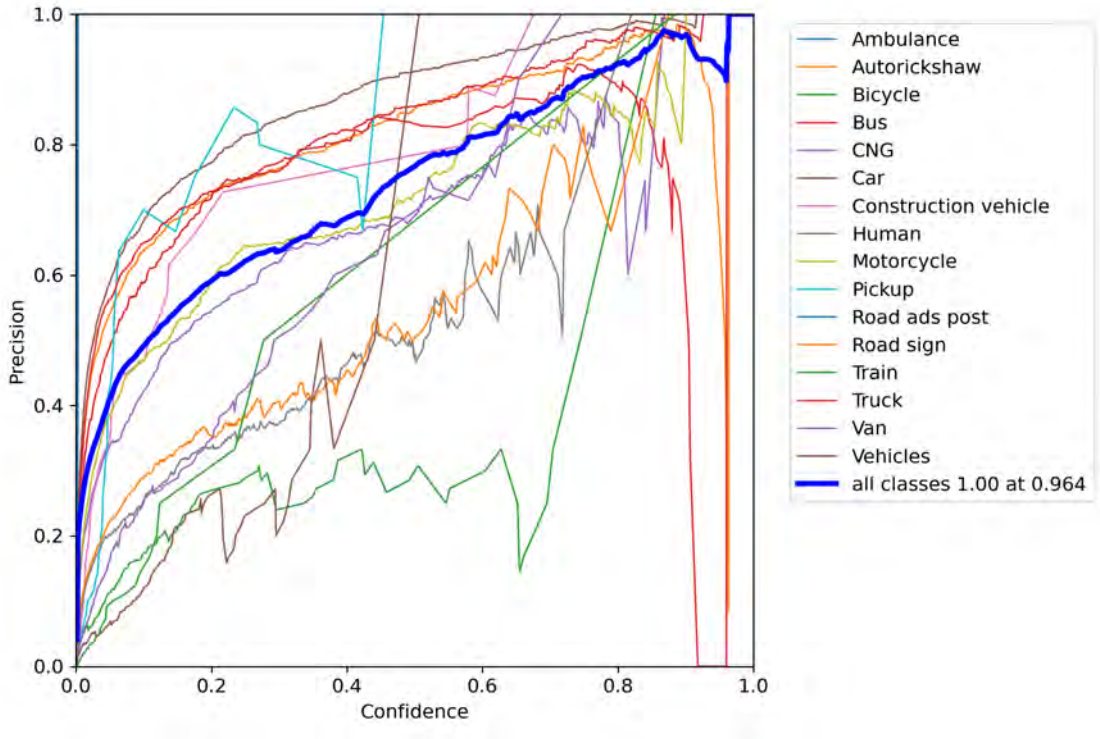


Figure 5.13: Yolo V7 Precision-Curve

5.3 Discussion

This study benchmarks three versions of the YOLO model (YOLOv5, YOLOv7, and YOLOv8) to evaluate their efficacy in object recognition on Bangladeshi road-

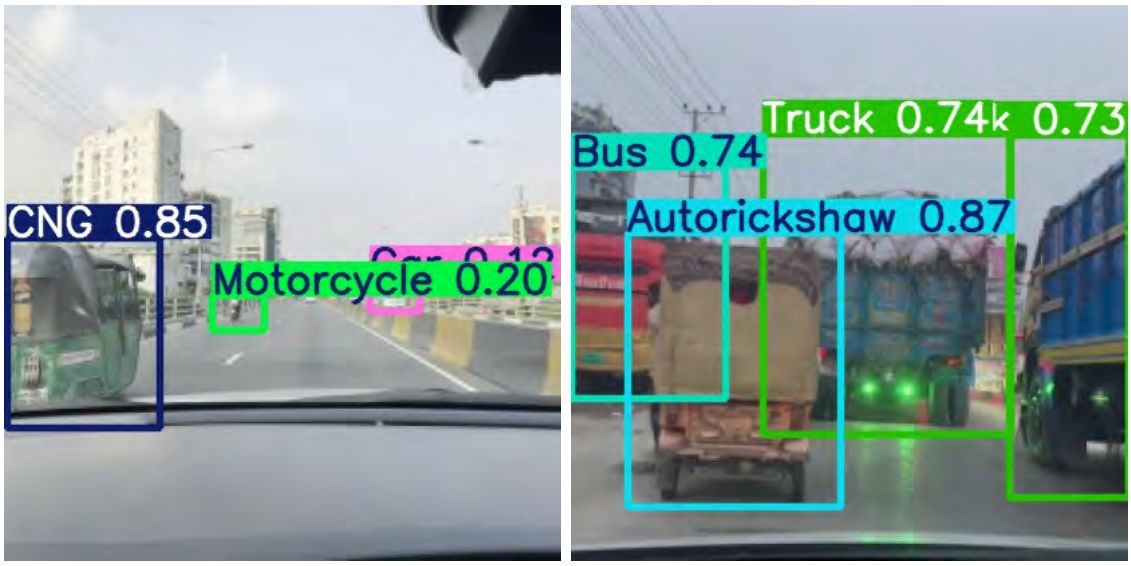


Figure 5.14: Yolo V5 Ditection

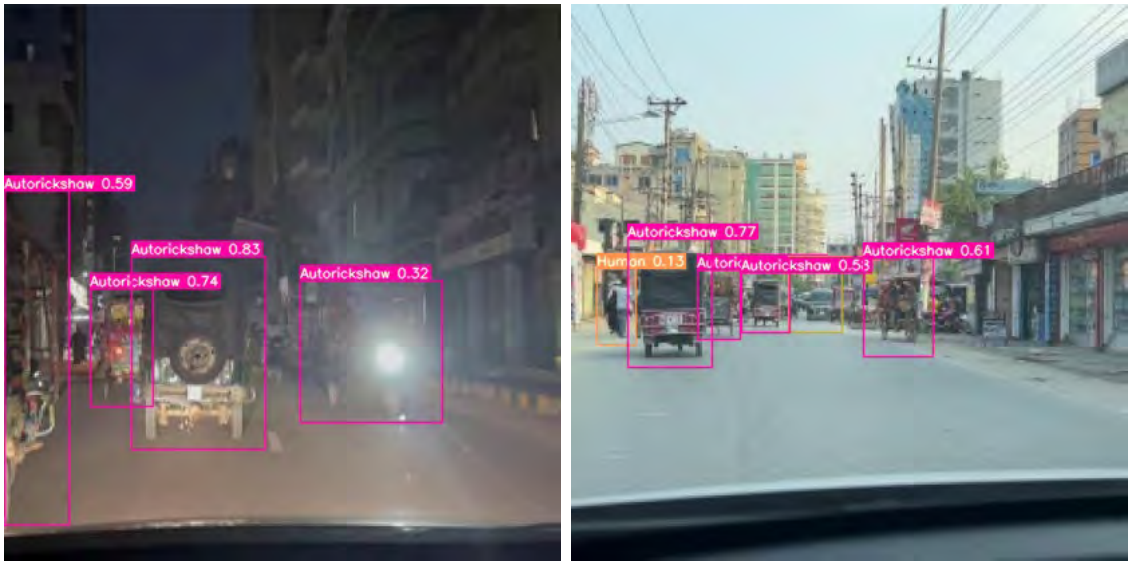


Figure 5.15: Yolo V7 Ditection

ways, characterized by unusual traffic circumstances, vehicle variety, and fluctuating illumination scenarios. Our findings provide significant insights into the efficacy, advantages, and constraints of these models when trained and validated on a bespoke dataset pertaining to Bangladeshi traffic. The training and validation outcomes for YOLOv5, YOLOv7, and YOLOv8 show distinct variations in their capacity to manage varied traffic scenarios.

YOLOv5, recognized for its lightweight architecture and real-time efficacy, had mediocre performance overall. The mean Average Precision (mAP) at 0.5 Intersection over Union (IoU) was 0.52; however, when assessed using more stringent measures such as mAP@.5-.95, it decreased to 0.283. The performance of particular classes differed, with "Car" attaining a precision of 0.815 and a recall of 0.893, while classes such as "Human" and "Bicycle" exhibited comparatively worse detection skills, resulting in much lower precision and recall scores.

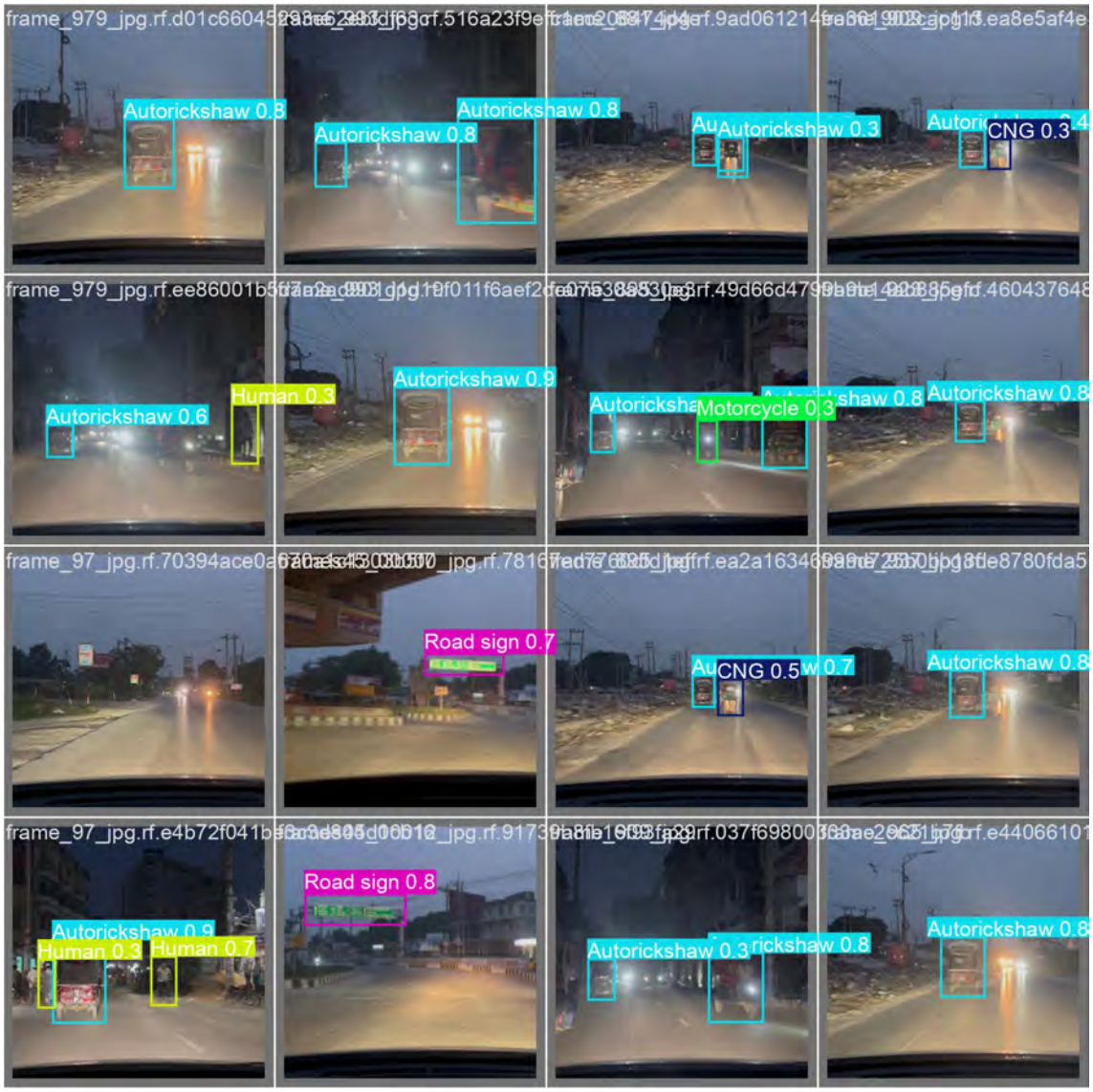


Figure 5.16: Yolo V8 Ditection

Conversely, YOLOv7 introduced enhancements, particularly in its architectural design, allowing superior speed-accuracy trade-offs. It had a more balanced performance relative to YOLOv5, attaining a somewhat superior mAP of 0.573 at IoU 0.5. YOLOv7 demonstrated commendable performance in detecting objects such as "Truck," "Car," and "Autorickshaw," achieving precision and recall scores exceeding 0.7; however, it faced difficulties with more challenging objects like "Bicycle" and "Road ads post," highlighting the need for improvement in identifying small or infrequent items. YOLOv7 excels in object localization, as demonstrated by the confusion matrices, and shows enhanced consistency across validation sets.

YOLOv8, the most recent iteration, included the most advanced optimizations and exhibited superior overall performance in our comparison. It attained a mAP@.5 of 0.538 and a mAP@.5-.95 of 0.288, demonstrating its exceptional generalization skills across many settings. YOLOv8 exhibited exceptional accuracy with vehicles such as "Car" (precision of 0.857, recall of 0.887) and "Train" (precision of 0.482, recall of 1.0); yet, like its predecessors, it struggled with categories like "Ambulance" and

"Road ads post."

YOLOv8 demonstrated superior performance in demanding settings, such as low-light and congested environments, as shown by its detection results on nighttime and low-light photographs, surpassing its predecessors. Notwithstanding these gains, a primary problem across all models was addressing class imbalances within the dataset. The dataset exhibited underrepresentation of classes such as "Bicycle," "Road ads post," and "Ambulance," resulting in decreased detection accuracy. While we did not use oversampling or sophisticated data augmentation methods, such strategies might enhance identification in minority classes. Another challenge was the obstruction of objects, which is prevalent in congested traffic conditions. All models had difficulties in properly recognizing items that were partly obscured by other objects, especially in high-traffic situations.

The confusion matrices for YOLOv5 and YOLOv8 provide more insights into the models' misclassifications. The matrix of YOLOv8 indicated frequent misclassification between similar vehicle classes, such as "Truck" and "Bus," suggesting a need for more fine-tuning or data acquisition in these categories to mitigate inaccuracies.

This comparative examination of YOLO models emphasizes the merits of each iteration and identifies potential areas for further enhancement. YOLOv8 is the most effective model in this context, especially for navigating complex environments such as Bangladeshi roads; however, additional improvements in addressing class imbalances, small objects, and occluded objects are essential for optimizing object detection performance in real-world autonomous navigation applications.

Comparing the performance of YOLOv5, YOLOv7, and YOLOv8 for object recognition on Bangladeshi highways revealed notable disparities in detection accuracy, precision, and recall measures. The models were assessed based on their capacity to manage varied traffic scenarios, including atypical vehicles such as rickshaws, autorickshaws, and buses, along with fluctuations in lighting conditions.

YOLOv5 is recognized for its efficient real-time functionality, especially in settings with constrained processing resources. In this research, YOLOv5 attained a mean Average Precision (mAP) of 0.52 at an Intersection over Union (IoU) threshold of 0.5. The accuracy for classes such as "Car" (0.815) and "Truck" (0.673) was comparatively good; nevertheless, it encountered difficulties with less prevalent classes like "Ambulance" and "Bicycle," which exhibited much lower precision scores. YOLOv5 exhibits real-time performance appropriate for mobile and embedded devices; nonetheless, it struggles with smaller and obscured objects in congested traffic situations. In comparison to established models such as Faster R-CNN, YOLOv5 exhibits superior inference speeds but decreased accuracy in identifying small and densely clustered objects.

YOLOv7, with its architectural advancements, yielded significant increases in both velocity and precision. YOLOv7 attained a mean Average Precision (mAP) of 0.573 at an Intersection over Union (IoU) of 0.5, exceeding YOLOv5 in both accuracy and recall for the majority of object categories. For instance, "Autorickshaw" obtained an accuracy of 0.742 and a recall of 0.858, making it more effective in identifying frequently occurring vehicles in Bangladeshi traffic. However, like YOLOv5, YOLOv7 had difficulties with infrequent classes, such as "Bicycle" and "Ambulance." Despite

these constraints, YOLOv7 demonstrates a superior balance between velocity and precision relative to preceding models, and it has been shown to surpass YOLOv5 and Faster R-CNN in certain instances.

YOLOv8 demonstrated superior overall performance compared to the three models, with cutting-edge enhancements for both accuracy and recall. It attained a mean Average Precision (mAP) of 0.543 at an Intersection over Union (IoU) of 0.5 and a mAP of 0.288 over an IoU range of 0.5 to 0.95, indicating enhanced generalization capabilities.

YOLOv8 surpassed its predecessors in difficult situations, including low-light environments, and showed notable efficacy in identifying prevalent vehicle categories such as "Car" (precision of 0.857) and "Truck" (precision of 0.754). In comparison to YOLOv7, YOLOv8 demonstrated superior performance in precision-confidence and precision-recall curves, making it more appropriate for real-world applications where accuracy under fluctuating circumstances is essential. In comparison to established models such as SSD or Faster R-CNN, YOLOv8 offers superior accuracy and recall for Bangladeshi traffic situations, while still sustaining competitive inference speeds.

In comparison to Faster R-CNN, a model frequently employed in prior studies for Bangladeshi road object detection, it is clear that although Faster R-CNN provides superior precision for small and overlapping objects, it is hindered by slower inference times, rendering it less appropriate for real-time applications. . In fact, Datta et al. (2021) demonstrated that Faster R-CNN achieved an mAP of 86.42
Therefore, YOLOv8 outperforms YOLOv5 and YOLOv7 with the Faster R-CNN model in terms of accuracy, recall, and total mAP. Thus, it will be able to handle heterogeneous traffic conditions with various illuminations and is one of the potential approaches for autonomous vehicle applications in the near future under unstructured traffic conditions such as in Bangladesh. This table compares the three YOLO models (YOLOv5, YOLOv7, YOLOv8) with an established model (Faster R-CNN) based on performance metrics and essential properties within the context of Bangladeshi traffic situations.This table contrasts established models, such as Faster R-CNN, with the more recent YOLO iterations, highlighting advancements in model efficacy. Faster R-CNN exhibits superior precision in contexts demanding high accuracy; yet, it has challenges with real-time performance owing to prolonged inference times. Conversely, YOLO models, especially YOLOv8, achieve an equilibrium between velocity and precision, rendering them more appropriate for real-time applications, particularly in fluctuating traffic scenarios.

Model	Precision (P)	Recall (R)	mAP @0.5	mAP@0.5:0.95	Inference Speed	Key Strengths	Key Weaknesses
YOLOv5	0.655	0.485	0.52	0.283	Fast (real-time)	Lightweight, efficient for real-time applications	Struggles with smaller objects and overlapping detections
YOLOv7	0.645	0.579	0.573	0.284	Moderate	Balanced speed and accuracy, handles overlapping objects better	Still slower than YOLOv5, requires more computational resources
YOLOv8	0.538	0.527	0.543	0.288	Fast	Improved recall for difficult conditions, effective in low-light environments	Precision lower than YOLOv5 and YOLOv7 in crowded environments
Faster R-CNN	0.716	0.642	0.864	0.423	Slow	Highly accurate, especially in detecting small and occluded objects	Slow inference speed, unsuitable for real-time applications

Figure 5.17: Comparison between Proposed model and Existing Work.

Chapter 6

Future Work

6.1 Future Work

Future research and enhancement may pursue several avenues. An urgent area for improvement is the rectification of class imbalance within datasets. Methods include data augmentation, oversampling, and synthetic data synthesis may enhance the representation of underrepresented classes, therefore augmenting the models' detection efficacy across all categories. Furthermore, using sophisticated occlusion management methods, such as multi-scale feature extraction or the integration of 3D sensor data (e.g., LiDAR), may enhance the precision of identifying partly concealed objects, a significant challenge in densely inhabited regions. A interesting direction for future research is the integration of multi-modal data, merging RGB camera data with inputs from other sensors such as LiDAR or radar. This would provide more comprehensive contextual information and might significantly improve the model's capacity to identify objects in low visibility situations, such as fog, rain, or nighttime settings.
Furthermore, enhancing transfer learning methodologies via the use of domain adaptation strategies will provide improved model generalization across diverse areas and situations. This may be especially beneficial for modifying models trained on information from one geographic area to function efficiently in another with varying traffic patterns and environmental conditions. Subsequently, next study need to concentrate on enhancing these models for edge computing systems. Autonomous cars depend on real-time processing, and implementing models such as YOLOv8 on embedded systems with constrained computing resources is a significant problem. Advanced model compression strategies, including pruning, quantization, and knowledge distillation, should be investigated to facilitate the successful deployment of these models in real-world applications while maintaining performance integrity. Although YOLOv8 is the most proficient model in our analysis, more improvements in mitigating dataset imbalances, enhancing detection in occluded scenarios, and optimizing for edge devices will be crucial for attaining reliable, real-time object identification in autonomous driving applications.

Chapter 7

Conclusion

7.1 Conclusion

This study attempts to test and compare the performance of three state-of-the-art versions of YOLO, namely, YOLOv5, YOLOv7, and YOLOv8, using a private dataset created only for this purpose and the publicly available Bangladeshi Autonomous Driving Object Detection Dataset, BADODD. Comparisons of these models were performed in this respect concerning their performance in identifying objects within uncongested and unstructured traffic scenes of Bangladesh. The custom dataset was developed to overcome the limitations of BADODD-particularly, for capturing unusual vehicles like rickshaws and CNGs in a better way, also considering quite diverse lighting and weather conditions. This private dataset contributed a lot to improving the model's training with respect to complex real-world traffic environments.It achieved the results by giving a comprehensive comparative analysis of the strengths and weaknesses of each model in the recognition of objects within complex scenes, with heavy traffic flow, under different lighting conditions, and both traditional and unconventional vehicles. YOLOv5 is known for its lightweight nature, so it performed way better in real time but showed problematic issues while dealing with smaller objects and overlapping detections. In contrast, YOLOv5 showed lower accuracy and recall in complicated or obscured items common in Bangladeshi traffic conditions. However, its mAP of 0.52 at an IoU threshold of 0.5 and proficiency in detecting big vehicles such as trucks and automobiles made it fit for certain real-time applications. YOLOv7 shows improved performance wherein it is more accurate compared to YOLOv5 but required more processing resources.It achieved the best mAP of 0.573 and outperformed others, especially in the detection of heavier vehicles like trucks and autorickshaws. With such a design architecture, YOLOv7 could handle overlapping objects relatively better compared to YOLOv5. However, it also struggled with smaller objects or scenarios with serious occlusion. Being moderately fast, though not as fast as YOLOv5, YOLOv7 is a better balance between speed and precision for real-time detection tasks in moderately complex environments. Among these compared models, the latest version, YOLOv8, shows the best overall performance, especially under challenging environments like low light and heavy flow. It achieved a mean Average Precision of 0.543 at an IoU of 0.5 and showed stronger generalization across more object categories and traffic scenarios. While this might be considered to be generally the best due to its handling of complex scenarios, it has slightly lower accuracy compared to YOLOv7 in some cases.

It did well for typical vehicles like cars and trucks but struggled, as did the other models, with less common or poorly represented items in the dataset. The private dataset can represent dynamic and diverse traffic conditions of Bangladesh in some pitfalls BADODD. It is the private dataset which this research presents and further comparisons on model performance, using both datasets to show exactly how the limitation from an existing dataset can be overcome for the strengthening of object detection in practical application.

7.1.1 Constraints of YOLO Models

Despite their different capabilities, all three models shared several challenges with regard to the problem of class imbalance and in the determination of small or occluded objects. The BADODD dataset is quite extensive, but the class distribution is highly imbalanced, with some classes being very underrepresented. For example, bikes and ambulances are seriously underrepresented. This reduced the general competence of the model in generalizing well to all object classes. Also, the models failed to clearly identify occluded objects, that are rickshaws or pedestrians in front of and behind larger vehicles, a common view in the highly congested traffic of Bangladesh. In order to handle these limitations, this research introduced a custom private dataset, which was designed in such a way that it could mitigate the issues present in BADODD. The private dataset captured the diverse ranges of vehicles and real-world conditions on the roads of Bangladesh considering multiple lighting and weather conditions. Improved balance across object categories in this dataset means allowing the models to perform better with respect to finding rare or occluded objects. Among these, YOLOv8 showed some enhancement in handling class imbalance and occluded objects compared to previous versions, although there is still room for further optimization, especially in densely packed traffic flow. The precision of Faster R-CNN for smaller and occluded objects outperformed the YOLO models since this model was also widely used in such object detection. This more precise identification was achieved at the cost of slower inference speed, hence less suitable for real-time applications, especially in dynamic traffic conditions like those seen in Bangladesh. The introduction of the custom private dataset overcame many limitations found in BADODD, and YOLOv7 and YOLOv8 were able to find a better balance between speed and accuracy. This model was more suitable for real-time object identification in autonomous driving applications, especially with the diverse and unstructured traffic conditions encountered in Bangladesh.

Bibliography

[1] K. Bernardin, A. Elbs, and R. Stiefelhagen, "Multiple object tracking performance metrics and evaluation in a smart room environment," in *Sixth IEEE International Workshop on Visual Surveillance, in conjunction with ECCV*, Citeseer, vol. 90, 2006.

[2] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[3] J. Levinson, J. Askeland, J. Becker, *et al.*, "Towards fully autonomous driving: Systems and algorithms," in *2011 IEEE intelligent vehicles symposium (IV)*, IEEE, 2011, pp. 163–168.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*, IEEE, 2012, pp. 3354–3361.

[5] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[6] R. Girshick, "Fast r-cnn," *arXiv preprint arXiv:1504.08083*, 2015.

[7] O. Russakovsky, J. Deng, H. Su, *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015.

[8] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE international conference on image processing (ICIP)*, IEEE, 2016, pp. 3464–3468.

[9] H. M. Bui, M. Lech, E. Cheng, K. Neville, and I. S. Burnett, "Using grayscale images for object recognition with convolutional-recursive neural network," in *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*, IEEE, 2016, pp. 321–325.

[10] J. Redmon, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[11] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[13] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," in *2017 IEEE international conference on image processing (ICIP)*, IEEE, 2017, pp. 3645–3649.

[14] A. Ćorović, V. Ilić, S. urić, M. Marijan, and B. Pavković, "The real-time detection of traffic participants using yolo algorithm," in *2018 26th Telecommunications Forum (TELFOR)*, IEEE, 2018, pp. 1–4.

[15] U. Lee, J. Jung, S. Jung, and D. H. Shim, "Development of a self-driving car that can handle the adverse weather," *International journal of automotive technology*, vol. 19, pp. 191–197, 2018.

[16] B. Lim, B. Yang, and H. Kim, "Real-time lightweight cnn for detecting road object of various size," in *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, IEEE, 2018, pp. 202–203.

[17] M. A. Al-Masni, M. A. Al-Antari, J.-M. Park, *et al.*, "Simultaneous detection and classification of breast masses in digital mammograms via a deep learning yolo-based cad system," *Computer methods and programs in biomedicine*, vol. 157, pp. 85–94, 2018.

[18] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of deep learning for object detection," *Procedia Computer Science*, vol. 132, pp. 1706–1717, 2018.

[19] J. Redmon, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[20] A. K. Shinde, V. G., and S. Shubham, *Yolo based human action recognition and localization*, Unpublished, 2018.

[21] W. Yang and Z. Jiachun, "Real-time face detection based on yolo," in *2018 1st IEEE international conference on knowledge innovation and invention (ICKII)*, IEEE, 2018, pp. 221–224.

[22] P. Chao, C.-Y. Kao, Y.-S. Ruan, C.-H. Huang, and Y.-L. Lin, "Hardnet: A low memory traffic network," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3552–3561.

[23] R.-C. Chen *et al.*, "Automatic license plate recognition via sliding-window darknet-yolo deep learning," *Image and Vision Computing*, vol. 87, pp. 47–56, 2019.

[24] D. H. Dos Reis, D. Welfer, M. A. De Souza Leite Cuadros, and D. F. T. Gamarra, "Mobile robot navigation using an object recognition software with rgbd images and the yolo algorithm," *Applied Artificial Intelligence*, vol. 33, no. 14, pp. 1290–1305, 2019.

[25] K. Mori, H. Fukui, T. Murase, T. Hirakawa, T. Yamashita, and H. Fujiyoshi, "Visual explanation by attention branch network for end-to-end learning-based self-driving," in *2019 IEEE intelligent vehicles symposium (IV)*, IEEE, 2019, pp. 1577–1582.

[26] Y. Nie, P. Sommella, M. O'Nils, C. Liguori, and J. Lundgren, "Automatic detection of melanoma with yolo deep convolutional neural networks," in *Proceedings of the 2019 E-Health and Bioengineering Conference (EHB)*, Iasi, Romania, Nov. 2019, pp. 1–4.

[27] E. N. Ukhwah, E. M. Yuniarno, and Y. K. Suprapto, "Asphalt pavement pothole detection using deep learning method based on yolo neural network," in *Proceedings of the 2019 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, Surabaya, Indonesia, Aug. 2019, pp. 35–40.

[28] H. M. Ünver and E. Ayan, "Skin lesion segmentation in dermoscopic images with combination of yolo and grabcut algorithm," *Diagnostics*, vol. 9, p. 72, 2019.

[29] K. Bhambani, T. Jain, and K. A. Sultanpure, "Real-time face mask and social distancing violation detection system using yolo," in *Proceedings of the 2020 IEEE Bangalore Humanitarian Technology Conference (B-HTC)*, Vijiyapur, India, Oct. 2020, pp. 1–6.

[30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[31] S. Kulik and A. Shtanko, "Experiments with neural net object detection system yolo on small training datasets for intelligent robotics," in *Advanced Technologies in Robotics and Intelligent Systems: Proceedings of ITR 2019*, Berlin/Heidelberg, Germany: Springer, 2020, pp. 157–162.

[32] X. Li, W. Wang, L. Wu, *et al.*, "Generalized focal loss: Learning qualified and distributed," 2020.

[33] M. M. Rahman, S. A. Mamun, M. S. Kaiser, M. S. Islam, and M. A. Rahman, "Cascade classification of face liveliness detection using heart beat measurement," in *Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020*, Springer, 2020, pp. 581–590.

[34] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-iou loss: Faster and better learning for bounding box regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, New York, NY, USA, Feb. 2020, pp. 12 993–13 000.

[35] W. Chen, H. Huang, S. Peng, C. Zhou, and C. Zhang, "Yolo-face: A real-time face detector," *Visual Computer*, vol. 37, pp. 805–813, 2021.

[36] L. Cheng, J. Li, P. Duan, and M. Wang, "A small attentional yolo model for landslide detection from satellite remote sensing images," *Landslides*, vol. 18, pp. 2751–2765, 2021.

[37] Y. Du, N. Pan, Z. Xu, F. Deng, Y. Shen, and H. Kang, "Pavement distress detection and classification based on yolo network," *International Journal of Pavement Engineering*, vol. 22, pp. 1659–1672, 2021.

[38] P. Kumar, S. Narasimha Swamy, P. Kumar, G. Purohit, and K. S. Raju, "Real-time, yolo-based intelligent surveillance and monitoring system using jetson tx2," in *Data Analytics and Management: Proceedings of ICDAM*, Berlin/Heidelberg, Germany: Springer, 2021, pp. 461–471.

[39] P. Liu, H. Fu, and H. Ma, "An end-to-end convolutional network for joint detecting and denoising adversarial perturbations in vehicle classification," *Computational Visual Media*, vol. 7, pp. 217–227, 2021.

[40] H. Ma, T. Celik, and H. Li, "Fer-yolo: Detection and classification based on facial expressions," in *Image and Graphics: 11th International Conference, ICIG 2021, Haikou, China, August 6–8, 2021, Proceedings, Part I 11*, Springer, 2021, pp. 28–39.

[41] S. Moshin Reza, M. Mahfujur Rahman, H. Parvez, O. Badreddin, and S. Al Mamun, "Performance analysis of machine learning approaches in software complexity prediction," in *Proceedings of International Conference on Trends in Computational and Cognitive Engineering: Proceedings of TCCE 2020*, Springer, 2021, pp. 27–39.

[42] Y. Qing, W. Liu, L. Feng, and W. Gao, "Improved yolo network for free-angle remote sensing target detection," *Remote Sensing*, vol. 13, p. 2171, 2021.

[43] O. Sahin and S. Ozer, "Yolodrone: Improved yolo architecture for object detection in drone images," in *Proceedings of the 2021 44th International Conference on Telecommunications and Signal Processing (TSP)*, Brno, Czech Republic, Jul. 2021, pp. 361–365.

[44] L. Tan, T. Huangfu, L. Wu, and W. Chen, "Comparison of retinanet, ssd, and yolo v3 for real-time pill identification," *BMC Medical Informatics and Decision Making*, vol. 21, pp. 1–11, 2021.

[45] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-yolov4: Scaling cross stage partial network," in *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, 2021, pp. 13 029–13 038.

[46] A. H. Ashraf, M. Imran, A. M. Qahtani, *et al.*, "Weapons detection for security and video surveillance using cnn and yolo-v5s," *CMC-Computers, Materials & Continua*, vol. 70, pp. 2761–2775, 2022.

[47] C. Dewi, R. C. Chen, X. Jiang, and H. Yu, "Deep convolutional neural network for enhancing traffic sign recognition developed on yolo v4," *Multimedia Tools and Applications*, vol. 81, pp. 37 821–37 845, 2022.

[48] X. Dong, S. Yan, and C. Duan, "A lightweight vehicles detection network model based on yolov5," *Engineering Applications of Artificial Intelligence*, vol. 113, p. 104 914, 2022.

[49] Y. Wang and H. Yang, "Multi-target pedestrian tracking based on yolov5 and deepsort," in *2022 IEEE Asia-Pacific Conference on Image Processing, Electronics and Computers (IPEC)*, IEEE, 2022, pp. 508–514.

[50] Z. Zakria, J. Deng, R. Kumar, M. S. Khokhar, J. Cai, and J. Kumar, "Multiscale and direction target detecting in remote sensing images via modified yolo-v4," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 1039–1048, 2022.

[51] N. Zarei, P. Moallem, and M. Shams, "Fast-yolo-rec: Incorporating yolo-base detection and recurrent-base prediction networks for fast vehicle detection in consecutive images," *IEEE access*, vol. 10, pp. 120 592–120 605, 2022.

[52] Y. Zhang, Z. Guo, J. Wu, Y. Tian, H. Tang, and X. Guo, "Real-time vehicle detection based on improved yolo v5," *Sustainability*, vol. 14, no. 19, p. 12 274, 2022.

[53] "Fault detection and quality inspection of printed circuit board using yolo-v7 algorithm of deep learning," in *IEEE Conference Publication*, IEEE Xplore, May 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10178512/citations#citations.

[54] H. Jin, F. Chollet, Q. Song, and X. Hu, "Autokeras: An automl library for deep learning," *Journal of machine Learning research*, vol. 24, no. 6, pp. 1–6, 2023.

[55] K. Kolachi, M. Khan, S. A. Sarang, and A. Raza, "Fault detection and quality inspection of printed circuit board using yolo-v7 algorithm of deep learning," in *2023 7th International Multi-Topic ICT Conference (IMTIC)*, IEEE, 2023, pp. 1–7.

[56] O. E. Olorunshola, M. E. Irhebhude, and A. E. Evwiekpaefe, "A comparative study of yolov5 and yolov7 object detection algorithms," *Journal of Computing and Social Informatics*, vol. 2, no. 1, pp. 1–12, 2023. DOI: 10.33736/jcsi.5070.2023. [Online]. Available: https://doi.org/10.33736/jcsi.5070.2023.

[57] A. M. Roy, J. Bhaduri, T. Kumar, and K. Raj, "Wildect-yolo: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection," *Ecological Informatics*, vol. 75, p. 101 919, 2023.

[58] J. Solawetz and Francesco, *What is yolov8? the ultimate guide*, Online article, Apr. 2023. [Online]. Available: https://example.com.

[59] A. Moshrefi, A. Ali, S. T. Balghari, and F. Nabki, "High-accuracy airborne rangefinder via deep learning based on piezoelectric micromachined ultrasonic cantilevers," *IEEE Transactions on Ultrasonics, Ferroelectrics, and Frequency Control*, 2024.