Deep Learning Based Vehicle Detection with Modified YOLOv5

Submitted to the

Department of Computer Science and Engineering,
Mawlana Bhashani Science and Technology University
in partial fulfillment of the requirements for the degree of
B.Sc. in CSE.

Submitted by

Sayed Sohan, *(CE-16018)*

Arafat Hasan, *(CE-16024)*

Under the Supervision of

Professor Dr. Mohammad Motiur Rahman
Department of Computer Science and Engineering

Department of Computer Science and Engineering
Mawlana Bhashani Science and Technology University
Santosh, Tangail-1902, Bangladesh

September, 2021

Deep Learning Based Vehicle Detection with Modified YOLOv5

Submitted by
Sayed Sohan, *(CE-16018)*
Arafat Hasan, *(CE-16024)*

Under the Supervision of
Professor Dr. Mohammad Motiur Rahman
Department of Computer Science and Engineering

Submitted to the
Department of Computer Science and Engineering,
Mawlana Bhashani Science and Technology University
in partial fulfillment of the requirements for the degree of
B.Sc. in CSE.

Project / Thesis Evaluation Committee:

Teacher Name 1 .....

Teacher Name 2 .....

Teacher Name 3 .....

Thesis Approval

Sayed Sohan, *(CE-16018)*
Arafat Hasan, *(CE-16024)*

Deep Learning Based Vehicle Detection with Modified YOLOv5

We the undersigned, recommend that the thesis completed by the students listed above, in partial fulfillment of B.Sc. in CSE degree requirements, be accepted by the Department of Computer Science and Engineering, Mawlana Bhashani Science and Technology University for deposit.

Supervisor Approval*

.....

Professor Dr. Mohammad Motiur Rahman
Department of Computer Science and Engineering

Departmental Approval*

.....

Dr. Md. Sazzad Hossain
Associate Professor & Chairman
Department of Computer Science and Engineering

Mawlana Bhashani Science and Technology University
Santosh, Tangail-1902, Bangladesh

Declaration

We hereby declare that this thesis titled, ***"Deep Learning Based Vehicle Detection with Modified YOLOv5"*** is our own research work and we confirm that any part of this thesis has not been submitted yet for a degree or any other qualification at this university or any other institution.

.....
Sayed Sohan
Student ID: CE-16018

.....
Arafat Hasan
Student ID: CE-16024

dedicated to our
beloved parents

Abstract

Intelligent vehicle detection and vehicle counting are becoming increasingly important in the field of vehicular automation specially in traffic controlling, road management and crime control. It is possible to solve many other traffic related problems which are generally not possible by humans. We propose a deep learning method by modifying YOLOv5(You Only Look Once) object detection model and by preprocessing the dataset using ORB(Oriented FAST and rotated BRIEF) . Our proposed method beats the existing detection model results in our current Dhaka AI dataset. Our output of the proposed method looks promising compared to other methods. This work can be used in many modern tasks which require classification, counting and detecting vehicles in the roads of Bangladesh.

Keywords: YOLOv5, ORB, Automation, Deep Learning, Classification, Detection, Counting , Feature Extractor.

Acknowledgment

First and foremost, at this very special moment, we would like to express our sincere gratitude to the Almighty Allah for helping us to complete this Bachelor's degree. We are very grateful not only throughout our study time, but also during our lives, for the tremendous blessings the Almighty has conferred upon us.

We have been through the experiences with and assistance of other people in achieving this goal, and would like to express our heartfelt gratitude to those who have contributed to this work.

Sayed Sohan
Arafat Hasan
September, 2021

Contents

List of Figures

List of Tables

Chapter 1
Introduction

Chapter 1

Introduction

1.1 Overview

The world is progressing fast enough on technology in the last years and computer science is not an area that falls behind. Computer science can be divided into many areas of study, but we can divide them into three big fields. Software development, information technology and security. We will be focusing on information technology such as artificial intelligence, machine learning and deep learning. In computer science, Artificial Intelligence (AI) is the intelligence carried out by machines. Its goal is to build smart machines capable of performing tasks that normally are made by humans. AI is just software; it is an application written to do a task. For example, on Facebook with suggestions of names of people to tag on pictures, for fraud detection on credit cards or for self- driving cars that can manage their shelves to drive through the city.

As use of AI and deep learning become popular many fields start using these methods to improve many important fields. A vehicular automation system is one of them which is becoming popular recently to create an auto driving car, to detect cars on the road and also making improvements in urban designing for roads.

1.2 Motivation

Intelligent vehicle detection and vehicle counting are becoming increasingly important in the field of vehicular automation. However, due to the different types of vehicles, their detection remains a challenge that directly affects the accuracy of vehicle classification and detection. Much research has been conducted in this field of vehicular automation but it is mostly done in Europe, America and other rich countries. Unfortunately this kind of research is new from the perspective of Bangladesh and every country contains different sets of vehicles which is why it is hard to conduct research in these fields. Each year the number of vehicles

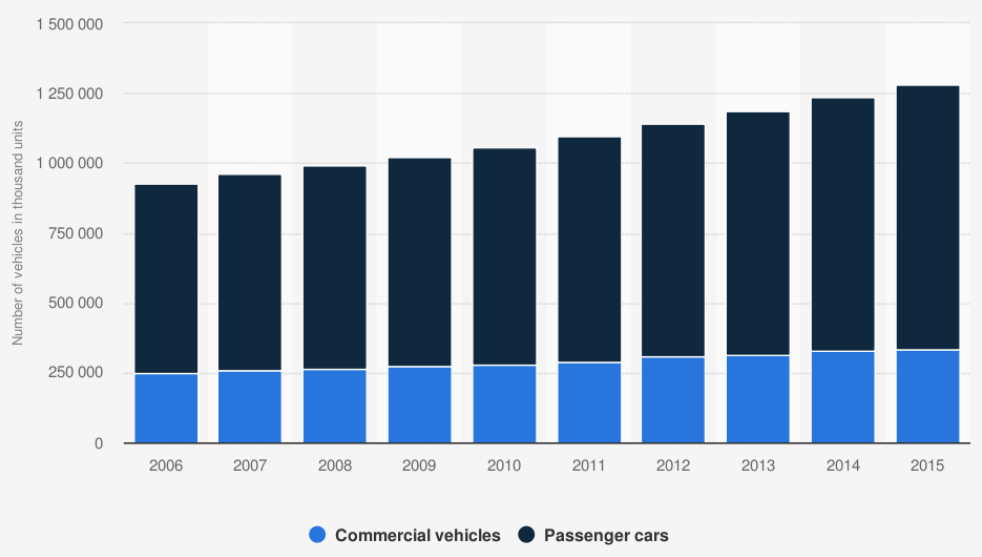

FIGURE 1.1. World vehicles in operations [1]

is increasing and keeping track of them is getting harder. From automatic cars to designing the modern road, we need information related to vehicles. For this, detecting vehicles is important. We know that it is hard but using deep learning and other modern methods for detecting objects we tried to challenge this task.

1.3 Research Questions

This thesis research questions are focused on vehicle detection and it's future. The main research questions are described below:

- **Research Question 1:** How can we propose a better deep learning model to detect different types of vehicle?
- **Research Question 2:** How can we more accurately classify different types of vehicle and increase classification accuracy?
- **Research Question 3:** How can we improve existing vehicle counting methods?

1.4 Objectives

The goal of this study is to build a methodology to detect vehicles from the road of Bangladesh with improved methods which will also solve the vehicles counting problem. Our work is to find a better solution than the existing method which will give better results on our custom dataset.

- To propose a better deep learning based model to detect different types of vehicles.
- To classify different types of vehicles precisely and increase classification accuracy significantly.
- To count different types of vehicle authentically with precise aim.

1.5 Organization

The remainder of this research is organized as follows:

- **Related Works** Highlights the previous research works on vehicles detection system.
- **Object Detection With YOLOv5** Describing how YOLOv5 model works and how it detect objects using deep learning.
- **Methodology** Illustrates our proposed method with proper example and diagrams.
- **Results & Analysis** Results of our proposed method and analysis regarding it.
- **Conclusion and Future Work** Concludes the whole research work and focuses on future research directions.

Chapter 2
Related Works

Chapter 2

Related Works

In this section, We have mentioned some literature reviews of recent works which are given below:

A vehicle (from Latin: vehiculum [2]) is a machine that transports people or cargo. Vehicles include wagons, bicycles, motor vehicles (motorcycles, cars, trucks, buses), railed vehicles (trains, trams), watercraft (ships, boats), amphibious vehicles (screw-propelled vehicle, hovercraft), aircraft (airplanes, helicopters, aerostat) and spacecraft [3]. Land vehicles are classified broadly by what is used to apply steering and drive forces against the ground: wheeled, tracked, railed or skied. ISO 3833-1977 is the standard, also internationally used in legislation, for road vehicles types, terms and definitions. Vehicle detection and vehicle type recognition is a practical application of machine learning concepts and is directly applicable for various operations in a traffic surveillance system contributing to an intelligent traffic surveillance system. We will introduce the processing of automatic vehicle detection and recognition using static image datasets. Further using the same technique, we shall improvise vehicle detection by using live CCTV surveillance. The surveillance system includes detection of moving vehicles and recognizing them, counting number of vehicles and verification of their permit with the organization [4].

Intelligent vehicle detection and counting are becoming increasingly important in the field of highway management. However, due to the different sizes of vehicles, their detection remains a challenge that directly affects the accuracy of vehicle counts. To address this issue, this paper proposes a vision-based vehicle detection and counting system. In the proposed vehicle detection and counting system, the highway road surface in the image is first extracted and divided into a remote area; the method is crucial for improving vehicle detection. Then, the vehicle trajectories are obtained by the ORB algorithm. Finally, the above two areas are placed into the YOLOv5 network to detect the type and location of the vehicle. Several highway surveillance videos based on different scenes are used to verify the proposed methods. The experimental results verify that using the

proposed segmentation method can provide higher detection accuracy, especially for the detection of small vehicle object [4].

Vehicle detection and statistics in highway monitoring video scenes are of considerable significance to intelligent traffic management and control of the highway. With the popular installation of traffic surveillance cameras, a vast database of traffic video footage has been obtained for analysis. Generally, at a high viewing angle, a more-distant road surface can be considered. The object size of the vehicle changes greatly at this viewing angle, and the detection accuracy of a small object far away from the road is low. In the face of complex camera scenes, it is essential to effectively solve the above problems and further apply them [5].

TABLE 2.1. Literature Review

References	Methods	Merits	limitations
Huansheng Song et al.[5]	YOLOv3 object detection and ORB algorithm	Can accurately detect driving direction of vehicle	Counting vehicle and detection is not in state of art
Tang. Y. Zhang et al.[6]	Haar-like features and AdaBoost algorithm	Can accurately detect vehicle in road	High false rate on detected vehicle
Ahmad Arinaldi et al.[7]	Faster-RCNN compared with MOG+SVM algorithm	It shows comparison of vehicle classification and detection	Many latest object detection model is not concerned
Lili Jia et al.[8]	Road and vehicle segmentation and camshift algorithm	Can detect vehicle from road CCTV camera accurately	Can not detect vehicle accurately from moving camera
Raad Ahmed Hadi et al.[9]	Road and vehicle segmentation and Shadow Identification algorithm	Can detect vehicle from CCTV and satellite camera	Can not detect vehicle accurately from moving camera

A CCTV camera is a very essential part of an intelligent traffic surveillance framework. It is simply the automated process of monitoring the traffic in a particular area and detecting vehicles for further action, as shown in the diagram. The captured images can provide valuable clues to the cops and other public essential tracking services, such as vehicle's license plate number, time and motion of vehicle, details associated with the driver, etc.. which all may lead to evidence of some crime or any unforeseen or unfortunate incidents. Earlier people used to process images manually. In fact, this system is still going on in India, whereas countries like the USA also have implemented automated machines- CCTVs that function 24x7 and take immediate action via signaling too. Manual work has always been proven slower and less efficient due to human errors and many other factors that affect living beings. Keeping these points in mind and moving with the advancement of technologies, many innovative thinkers have developed certain intelligent traffic control systems using various techniques. This research is based

upon the combination of two prior-made researches by scholars whose works have been published [10].

Chapter 3
Object Detection With YOLOv5

Chapter 3

Object Detection With YOLOv5

3.1 Introduction

YOLO (You Only Look Once) is a state-of-art algorithm devoted to object detection, as the name implies it can predict objects just by looking once at the image in a clever way. YOLO makes the prediction by classifying the object and locating it in the image. It uses deep learning and CNN techniques to detect objects, and distinguishes itself from its competitors because, as its name indicates, it requires to see the image only once, allowing it to be the fastest of all although it sacrifices a little accuracy. This speed allows users to easily detect objects in real time in videos (up to 30 FPS). In other words, YOLO takes as input value an image and passes through the neural network that looks like a CNN and returns a vector of bounding boxes and class prediction. To understand better how YOLO makes the prediction we will explain it with an example of an image as it can be seen in Figure 3.1:

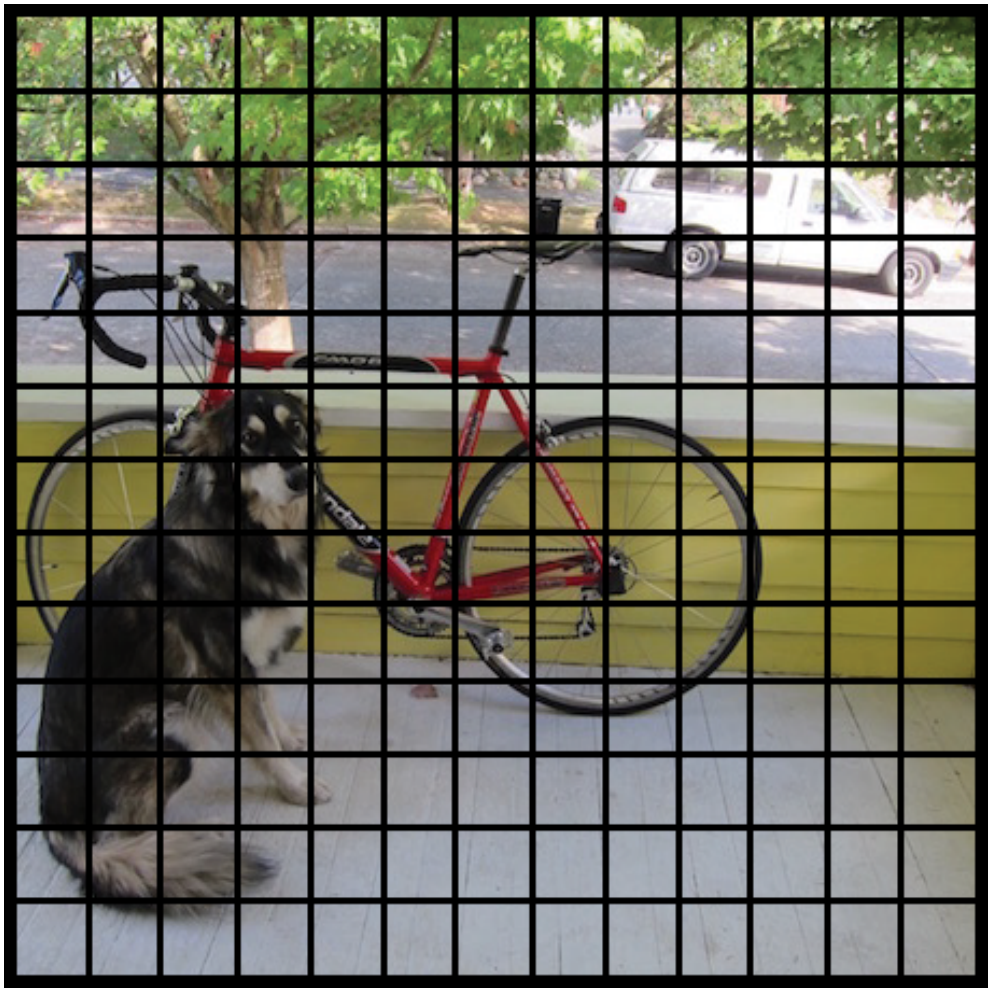

FIGURE 3.1. YOLO divides up the image into a grid of 13 by 13 cells: Each of these cells is responsible for predicting 5 bounding boxes. A bounding box describes the rectangle that encloses an object.

3.2 YOLO in Brief

To perform de detection YOLO first divides the image into an S x S grid of cells, in Figure 3.1 we can see that the image is divided into a 13 x 13 grid of cells. In each of the cells it predicts N possible bounding boxes and calculates the level of certainty (probability) of each of them, that is, S x S x N different bounding boxes are calculated in the image, the vast majority of them with a very low level of certainty. The score of each BB does not classify what type of object is or it does not know what object is fitting the bounding box, it just gives a confidence

value or a probability value of how good the BB is surrounding an object in each cell. In case of Figure 3.1 for each grid cells 5 bounding boxes are predicted and are graphically shown in Figure 3.2 where the higher the probability is the fatter the box is. Each cell makes N bounding box predictions and M class probabilities.

FIGURE 3.2. The predicted bounding boxes may look something like this, the higher the confidence score, the fatter the box is drawn.

The bounding box prediction is formed by 5 components: x, y, width, height, confidence score.

- The (x, y) components are like a mathematical graph coordinates that refer to the centre of the box, relative to the grid cell location. If the centre of the bounding box does not fall inside the grid cell this means that the grid cell

is not responsible of the object that has predicted the bounding box. This happens because each object that appears in the image is related to a single grid cell that is responsible for predicting the object. (x, y) are normalized between 0 and 1.

- (Width, height) represent the dimension of the box that contains an object, they are also normalized to values from 0 to 1 and they are fundamental to locate the object in the image.
- Confidence score is a real value that represents the assurance the algorithm has that the box contains an object of any class. The method how the confidence is calculated will be explained later.

3.3 The Predictions Vector

The first step to understanding YOLO is how it encodes its output. The input image is divided into an S x S grid of cells. For each object that is present on the image, one grid cell is said to be "responsible" for predicting it. That is the cell where the center of the object falls into [11].

Each grid cell predicts B bounding boxes as well as C class probabilities. The bounding box prediction has 5 components: (x, y, w, h, confidence). The (x, y) coordinates represent the center of the box, relative to the grid cell location (remember that, if the center of the box does not fall inside the grid cell, then this cell is not responsible for it). These coordinates are normalized to fall between 0 and 1. The (w, h) box dimensions are also normalized to [0, 1], relative to the image size [11].

3.4 The Network

The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 ×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 ×224 input image) and then double the resolution for detection.

The network structure looks like a normal CNN, with convolutions and max pooling layers, followed by 2 fully connected layers in the end: Note that the architecture was crafted for use in the Pascal VOC dataset, where the authors used S= 7, B = 2 and C = 20. This explains why the final feature maps are 7×7, and also explains the size of the output (7x7x (2×5+20)). Use of this network with

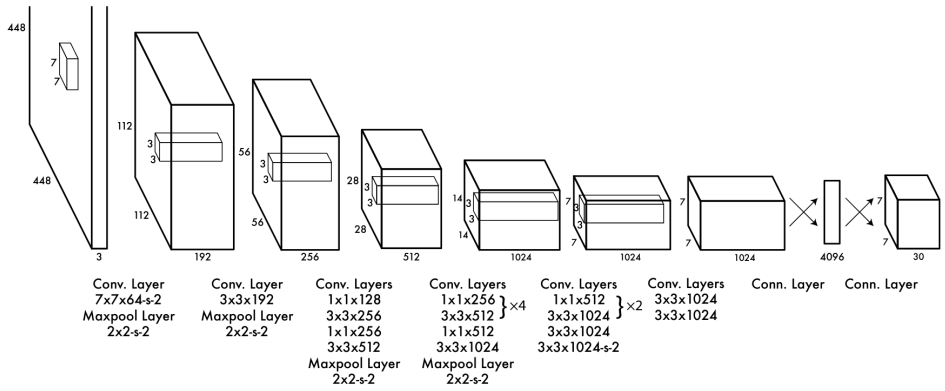

FIGURE 3.3. The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

a different grid size or different number of classes might require tuning of the layer dimensions. The authors mention that there is a fast version of YOLO, with fewer convolutions layers. We can see that in Table 3.1

Note that the architecture was crafted for use in the Pascal VOC dataset, where the authors used $S = 7$, $B = 2$ and $C = 20$. This explains why the final feature maps are 7×7, and also explains the size of the output ($7x7x(2 \times 5 + 20)$). Use of this network with a different grid size or different number of classes might require tuning of the layer dimensions. The authors mention that there is a fast version of YOLO, with fewer convolutional layers. The Table 3.1, however, display the full version. The sequences of 1x1 reduction layers and 3x3 convolutional layers were inspired by the GoogLeNet (Inception) model. The final layer uses a linear activation function. All other layers use a leaky RELU ($\phi(x) = x, if x > 0; 0.1x\ otherwise$).

3.5 The Loss Function

The loss function start like this:

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \tag{3.1}$$

This equation computes the loss related to the predicted bounding box position (x, y). Don't worry about λ for now, just consider it a given constant. The

TABLE 3.1. YOLO Network Structure

Name	Filters	Output Dimension
Conv 1	7 x 7 x 64, stride=2	224 x 224 x 64
Max Pool 1	2 x 2, stride=2	112 x 112 x 64
Conv 2	3 x 3 x 192	112 x 112 x 192
Max Pool 2	2 x 2, stride=2	56 x 56 x 192
Conv 3	1 x 1 x 128	56 x 56 x 128
Conv 4	3 x 3 x 256	56 x 56 x 256
Conv 5	1 x 1 x 256	56 x 56 x 256
Conv 6	1 x 1 x 512	56 x 56 x 512
Max Pool 3	2 x 2, stride=2	28 x 28 x 512
Conv 7	1 x 1 x 256	28 x 28 x 256
Conv 8	3 x 3 x 512	28 x 28 x 512
Conv 9	1 x 1 x 256	28 x 28 x 256
Conv 10	3 x 3 x 512	28 x 28 x 512
Conv 11	1 x 1 x 256	28 x 28 x 256
Conv 12	3 x 3 x 512	28 x 28 x 512
Conv 13	1 x 1 x 256	28 x 28 x 256
Conv 14	3 x 3 x 512	28 x 28 x 512
Conv 15	1 x 1 x 512	28 x 28 x 512
Conv 16	3 x 3 x 1024	28 x 28 x 1024
Max Pool 4	2 x 2, stride=2	14 x 14 x 1024
Conv 17	1 x 1 x 512	14 x 14 x 512
Conv 18	3 x 3 x 1024	14 x 14 x 1024
Conv 19	1 x 1 x 512	14 x 14 x 512
Conv 20	3 x 3 x 1024	14 x 14 x 1024
Conv 21	3 x 3 x 1024	14 x 14 x 1024
Conv 22	3 x 3 x 1024, stride=2	7 x 7 x 1024
Conv 23	3 x 3 x 1024	7 x 7 x 1024
Conv 24	3 x 3 x 1024	7 x 7 x 1024
FC 1	-	4096
FC 2	-	7 x 7 x 30 (1470)

function computes a sum over each bounding box predictor ($j = 0..B$) of each grid cell ($i = 0..S^2$). $\mathbb{1}obj$ is defined as follows:

- 1, If an object is present in grid cell i and the jth bounding box predictor is "responsible" for that prediction

- 0, otherwise

But how do we know which predictor is responsible for the object? Quoting the original paper:

> YOLO predicts multiple bounding boxes per grid cell. At training time we only want one bounding box predictor to be responsible for each object. We assign one predictor to be "responsible" for predicting an object based on which prediction has the highest current IOU with the ground truth.

The other terms in the equation should be easy to understand: (x, y) are the predicted bounding box position and $(\hat{x}, \hat{y})$ hat are the actual position from the training data.

Let's move on to the second part:

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} [(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2] \tag{3.2}$$

This is the loss related to the predicted box width / height. The equation looks similar to the first one, except for the square root. Quoting the paper again:

> Our error metric should reflect that small deviations in large boxes matter less than in small boxes. To partially address this we predict the square root of the bounding box width and height instead of the width and height directly.

Moving on to the third part:

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2 \tag{3.3}$$

Here we compute the loss associated with the confidence score for each bounding box predictor. C is the confidence score and $\hat{C}$ is the intersection over union of the predicted bounding box with the ground truth. $\mathbb{1}obj$ is equal to one when there is an object in the cell, and 0 otherwise. $\mathbb{1}noobj$ is the opposite.

The λ parameters that appear here and also in the first part are used to differently weight parts of the loss functions. This is necessary to increase model stability. The highest penalty is for coordinate predictions ($\lambda coord = 5$) and the lowest for confidence predictions when no object is present ($\lambda noobj = 0.5$).

The last part of the loss function is the classification loss:

$$\sum_{i=0}^{S^2} \mathbb{1}_i^{obj} \sum_{c \in classes} (p_i(c) - \hat{p}_i(c))^2 \tag{3.4}$$

It looks similar to a normal sum-squared error for classification, except for the $\mathbb{1}obj$ term. This term is used because so we don't penalize classification error when no object is present on the cell (hence the conditional class probability discussed earlier).

Note that the loss function only penalizes classification error if an object is present in that grid cell (hence the con-ditional class probability discussed earlier). It also only pe-nalizes bounding box coordinate error if that predictor is "responsible" for the ground truth box (i.e. has the highest IOU of any predictor in that grid cell). [11].

Chapter 4
Methodology

Chapter 4

Methodology

4.1 Introduction

In this chapter will overview our proposed method and will further discuss prepossessing, analysing, post-processing and interference in our deep learning model. Later in this chapter we will discuss about our mixed datasets.

4.2 Proposed Methodology

In our proposed method we have used BottleneckCSP and SSP multiple times in the middle of the layers. For this, the model has been gone mush deep, and works well. Training process has slowed down due to these running in parallel, but the obtained model has accelerated. We have processed the data before feeding it to our modified core model. We have got better results using ORB and Hog combined. The model was able to easily objects due to the ORB making corners and edges distinguished.

4.2.1 Oriented FAST and Rotated BRIEF (ORB)

Oriented FAST and Rotated BRIEF (ORB) performs as well as SIFT on the task of feature detection (and is better than SURF) while being almost two orders of magnitude faster. ORB builds on the well-known FAST key-point detector and the BRIEF descriptor. Both of these techniques are attractive because of their good performance and low cost [12]. ORB's main contributions are as follows:

- The addition of a fast and accurate orientation component to FAST
- The efficient computation of oriented BRIEF features
- Analysis of variance and correlation of oriented BRIEF features

- A learning method for de-correlating BRIEF features under rotational invariance, leading to better performance in nearest-neighbor applications

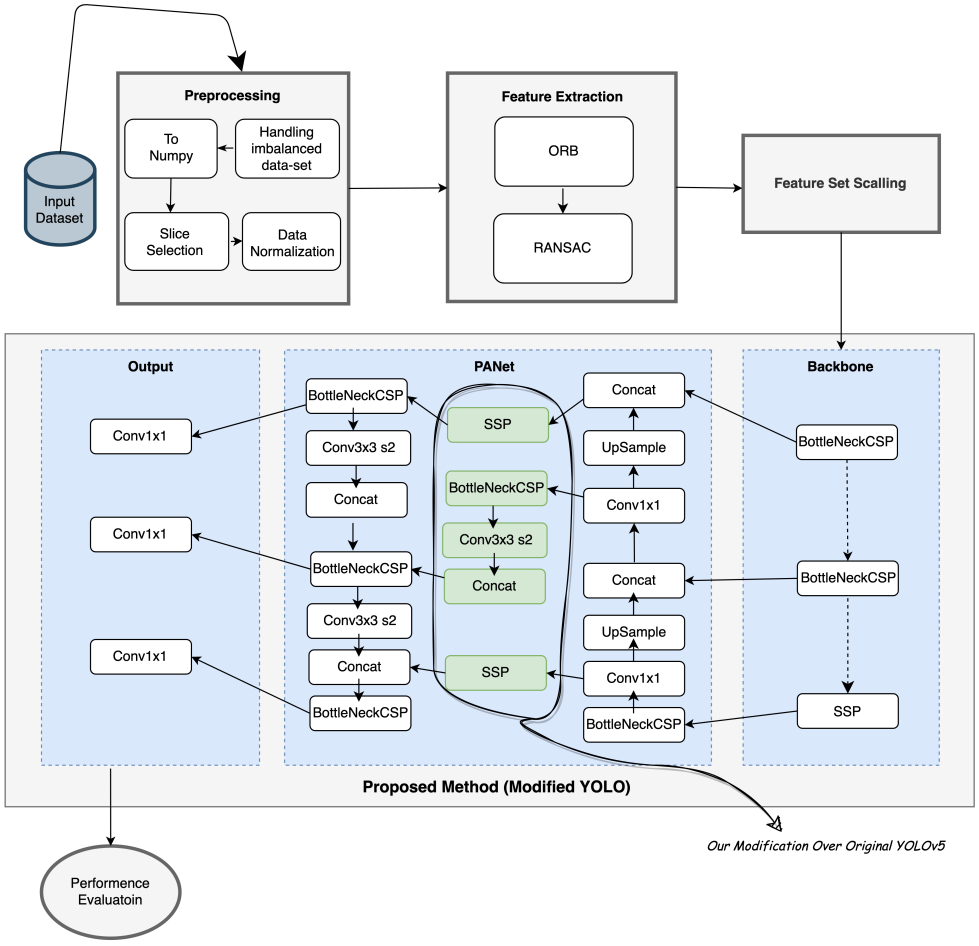

FIGURE 4.1. Our Proposed Methodology: We have added some more layers in PANet

Features from accelerated segment test (FAST) is a corner detection method, which could be used to extract feature points and later used to track and map objects in many computer vision tasks. The FAST corner detector is very suitable for real-time video processing application because of this high-speed performance.

Brief (Binary robust independent elementary feature) start by smoothing image using a Gaussian kernel in order to prevent the descriptor from being sensitive to high-frequency noise.

TABLE 4.1. Modified YOLO Network Structure

	params	module	arguments
0	8800	Focus	[3, 80, 3]
1	115520	Conv	[80, 160, 3, 2]
2	315680	BottleneckCSP	[160, 160, 4]
3	461440	Conv	[160, 320, 3, 2]
4	3311680	BottleneckCSP	[320, 320, 12]
5	1844480	Conv	[320, 640, 3, 2]
6	13228160	BottleneckCSP	[640, 640, 12]
7	7375360	Conv	[640, 1280, 3, 2]
8	4099840	SPP	[1280, 1280, [5, 9, 13]]
9	20087040	BottleneckCSP	[1280, 1280, 4, False]
10	820480	Conv	[1280, 640, 1, 1]
11	0	Upsample	[None, 2, 'nearest']
12	0	Concat	[1]
13	5435520	BottleneckCSP	[1280, 640, 4, False]
14	205440	Conv	[640, 320, 1, 1]
15	0	Upsample	[None, 2, 'nearest']
16	0	Concat	[1]
17	1360960	BottleneckCSP	[640, 320, 4, False]
18	922240	Conv	[320, 320, 3, 2]
19	0	Concat	[1]
20	5025920	BottleneckCSP	[640, 640, 4, False]
21	3687680	Conv	[640, 640, 3, 2]
22	0	Concat	[1]
23	20087040	BottleneckCSP	[1280, 1280, 4, False]
24	174954	Detect	[21, [[0, 1, 2, 3, 4, 5], [0, 1, 2, 3, 4, 5], [0, 1, 2, 3, 4, 5]], [320, 640, 1280]]

4.2.2 RANdom SAmple Consensus (RANSAC)

These descriptors between image pairs are then matched against each other to identify the best match with the minimum distance by brute force method. As the matches often contain outliers, a consistency check such as RANSAC is often used to remove inconsistent matches. Figure 4.2 shows the match points of an input image pair after adopting the RANSAC algorithm. The consistent matches are then used to model a transformation matrix for estimating a global motion for every pixel [13].

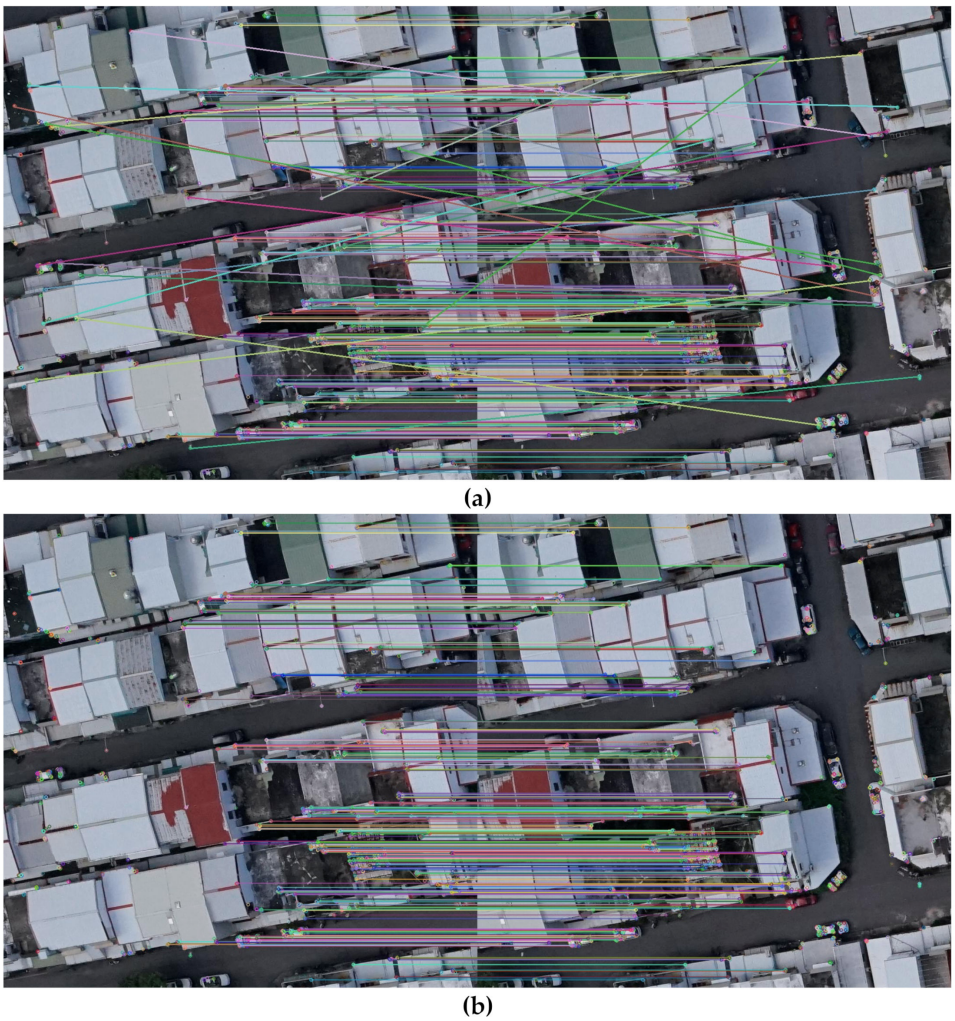

FIGURE 4.2. The match points of an input image pair before and after adopting RANSAC algorithm. (a) initial match, (b) filtered match points.

4.2.3 BottleneckCSP

A Bottleneck Residual Block is a variant of the residual block that utilises 1x1 convolutions to create a bottleneck. The use of a bottleneck reduces the number of parameters and matrix multiplications. The idea is to make residual blocks as thin as possible to increase depth and have less parameters [14]. They were introduced as part of the ResNet architecture, and are used as part of deeper ResNets such as ResNet-50 and ResNet-101.

4.2.4 Spatial Pyramid Pooling (SPP)

Spatial Pyramid Pooling (SPP) is a pooling layer that removes the fixed-size constraint of the network, i.e. a CNN does not require a fixed-size input image. Specifically, we add an SPP layer on top of the last convolutional layer. The SPP layer pools the features and generates fixed-length outputs, which are then fed into the fully-connected layers (or other classifiers) [15]. In other words, we perform some information aggregation at a deeper stage of the network hierarchy (between convolutional layers and fully-connected layers) to avoid the need for cropping or warping at the beginning.

4.3 Dataset Analysis

Our new dataset has 100000 cropped images after discarding some of the images which only containing background. Of these, 10000 contain 30000 vehicles in total. Although our source images cover much of Bangladesh and India, an imbalance still exists between different classes of vehicles in our benchmark. This is unavoidable: classes such as auto rickshaw. It can be from different viewpoints, have different ground sampling distances (gsd), different image sizes, aspect ratios, color, etc. Vehicles in different data may be significantly different in size and appearance. Figure 4.3 shows different images from four different datasets. It is obvious how different the VOC data is from any of the elevated data, while the VEDAI and AFVID data are somewhat similar, and the AF Building Camera data is more similar to the aerial and satellite data. To change the number of classes used, more than just the classes setting will need to be changed; the number of filters in the last convolutional layer must be altered to reflect the changed number of classes. The number of filters is set by $num(classes + coords + 1)$, where num is number of anchor boxes, and coords is four corresponding to the four coordinates used to define a bounding box.

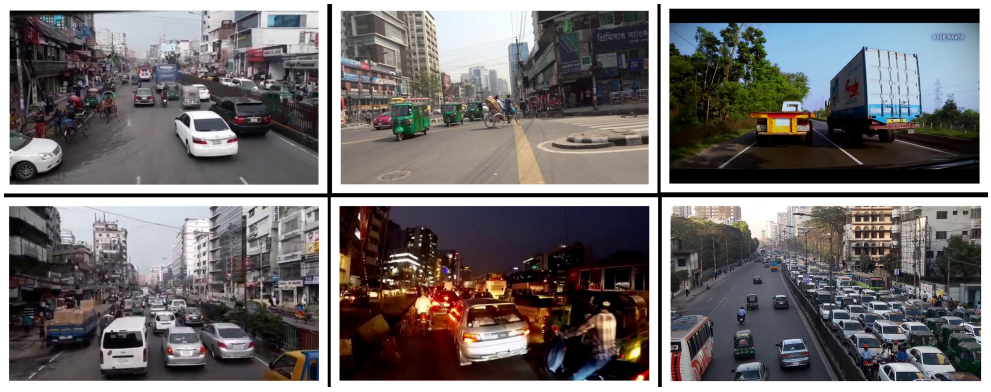

FIGURE 4.3. Some sample data from Dhaka AI dataset

4.3.1 COCO Dataset Details

- **Images per class** 1500 images per class recommended
- **Instances per class** 10000 instances (labeled objects) per class recommended

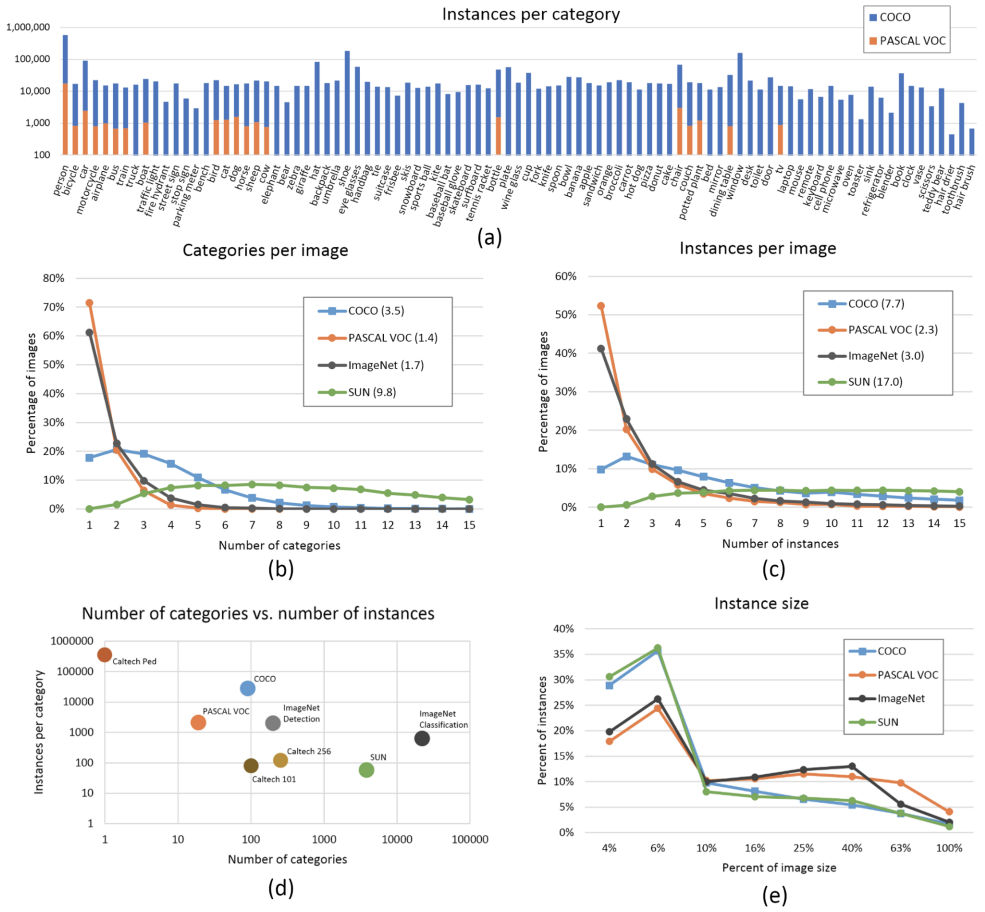

FIGURE 4.4. COCO dataset details [16]

- **Image variety** Must be representative of deployed environment. For real-world use cases we recommend images from different times of day, different seasons, different weather, different lighting, different angles, different sources (scraped online, collected locally, different cameras) etc.
- **Label consistency** All instances of all classes in all images must be labelled. Partial labelling will not work.
- **Label accuracy** Labels must closely enclose each object. No space should exist between an object and it's bounding box. No objects should be missing a label.
- **Background images** Background images are images with no objects that

are added to a dataset to reduce False Positives (FP). We recommend about 0-10% background images to help reduce FPs (COCO has 1000 background images for reference, 1% of the total). No labels are required for background images.

Figure 4.4 is showing details of COCO dataset. We used pretrained model trained by COCO dataset then fitted our model with our dataset as needed. This helps to reduce time to train model efficiently. We used COCO dataset in every model.

4.3.2 Dhaka AI Dataset

We have used a mixed dataset from Indian Driving Dataset and Dhaka AI Traffic Detection Challenge Dataset.

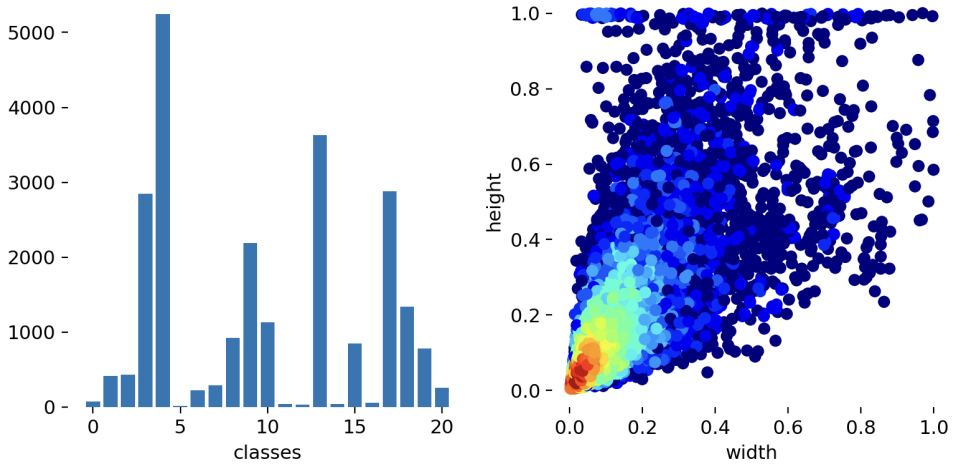

FIGURE 4.5. Class Distribution of Dhaka AI+IDD dataset

4.3.2.1 List of Classes

Our datasets contains 21 classes where some classes don't have much data compared to other classes.

Here buses and cars have the highest number of data compared to others. For this accuracy of detecting other classes became difficult. We can reduce this by gathering more data but it is a hard job. We can also train models with only a few classes which will increase the accuracy rate of the model but we will lose the ability to detect other vehicles.

1. Ambulance
2. Auto Rickshaw
3. Bicycle
4. Bus
5. Car
6. Garbage Van
7. Human Hauler
8. Minibus
9. Minivan
10. Motorbike
11. Pickup
12. Army Vehicle
13. Police Car
14. Rickshaw
15. Scooter
16. SUV
17. Taxi
18. Three Wheelers (CNG)
19. Truck
20. Van
21. Wheelbarrow

Above some of the classes have very low data like Wheelbarrow, Ambulance Garbage van etc. We tested our model by removing these classes and it gave good results but remove the ability to detect those classes.

Chapter 5
Results & Analysis

Chapter 5

Results & Analysis

5.1 Introduction

In this chapter we will discuss about our research findings and will compare it to the existing method. In this study we tried to detect vehicles from a mixed dataset which contained 21 vehicles from Bangladesh and India. We will show how our proposed methods perform compared to other well known methods and models and how it can be used in modern tasks.

5.2 Result Metrics Analysis

Here is the definition of the result metrics we are finding from our study.

5.2.1 Classification

In machine learning, classification refers to a predictive modeling problem where a class label is predicted for a given example of input data. Classification measured with confidence score also known as accuracy or precision which is defined in percentage(%) or between 0 to 1. We used both measuring method in our study. The equation for calculating precision score is:

$$precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{5.1}$$

In our dataset there are 21 classes and we have a very high accuracy rate but the detection rate is low because of low data and ground truth bounding boxes. In Table 5.2 we can see classification accuracy for 6 classes: car, rickshaw, bus, SUV and truck. We can see the accuracy is gradually increasing but both wheelbarrow and SUV have very low accuracy compared to other 4 classes. This is because of very low data related to these 2 classes. We identified that there are a total 10 classes which have very low data compared to other 11 classes.

Table 5.1. Evaluation of the model on the validation set with an IoU threshold of 45% that is the Area-Under-Curve for each unique Recall.

Class/Epoch	3 (%)	6 (%)	9 (%)	12(%)	15(%)
Ambulance	00.27	00.89	01.04	01.67	02.05
Auto Rickshaw	01.16	02.41	05.96	08.15	08.69
Bicycle	03.01	03.87	07.11	09.83	11.27
Bus	07.25	20.59	65.26	81.26	89.28
Car	09.57	23.72	59.05	85.11	94.60
Garbage Van	00.03	00.15	01.09	01.53	01.87
Human Hauler	00.75	01.04	03.43	07.29	06.49
Minibus	01.03	01.97	05.18	06.34	07.20
Minivan	06.30	08.29	19.07	29.47	37.91
Motorbike	05.21	13.62	21.46	50.37	69.33
Pickup	04.38	12.45	14.53	36.76	53.34
Army Vehicle	00.06	00.14	00.12	00.21	00.88
Police Car	00.03	00.87	00.64	00.87	01.02
Rickshaw	04.77	19.43	32.45	65.03	82.11
Scooter	00.65	01.11	01.94	02.14	02.45
SUV	03.86	12.32	23.76	32.54	41.34
Taxi	00.43	01.54	02.86	03.95	04.44
Three Wheelers (CNG)	01.54	17.25	36.24	57.23	87.26
Truck	05.49	16.38	31.86	56.67	76.63
Van	03.76	13.74	27.97	43.17	53.91
Wheelbarrow	01.49	09.32	15.34	22.39	23.29

5.2.2 Counting

Counting in machine learning means how many same objects are in the predicted data. Though It is the sub product of classification, it can be used in many useful tasks. Like counting how many cars are passing by everyday and how many buses and trucks are there. The calculation of counting is simple. For this we need to calculate the classification result and store it as data so that we can use it later.

Table 5.2. Vehicle count

Class ID	Class Name	Class count
5	car	3
9	minival	2
10	motorbike	3
19	truk	1
21	wheelbarrow	1

In Table 5.2 we showing counting of vehicles from Figure 5.1.

FIGURE 5.1. Inference from the model of our proposed architecture

5.2.3 Detection

Detection known as Image Localization, Image Localization will specify the location of single object in an image whereas Object Detection specifies the location of multiple objects in the image. Detection mostly use bounding box to detect object from image. Bounding box is a rectangle. The measuring method for detection is mAP(mean average precision) which is calculated by :

$$mAP = \frac{1}{n}\sum_{i=0}^{n}\frac{True\ Positive\ i}{(True\ Positive + False\ Positive)\ i} \tag{5.2}$$

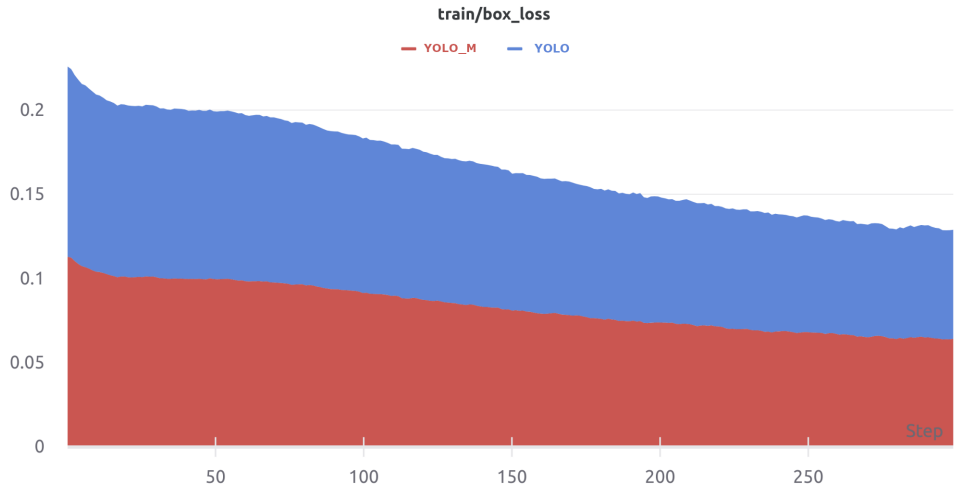

FIGURE 5.2. Bounding box loss comparison of our proposed model and YOLOv5

5.2.4 Comparison of Loss

In Figure 5.2 we can see comparison of detection box loss during training our model and YOLOv5 model. Loss is the penalty for a bad prediction. That is, loss is a number indicating how bad the model's prediction was on a single example. If the model's prediction is perfect, the loss is zero; otherwise, the loss is greater. The goal of training a model is to find a set of weights and biases that have low loss, on average, across all examples. Above blue is a loss for YOLOv5 model and red is a loss for your model. We can see that our model is almost 2x better compare to existing methods for detecting vehicles in our mixed dataset. We can see both model's loss is decreasing after each epoch but the initial difference is almost the same for all steps. By losing metrics we can see that our model is doing better.

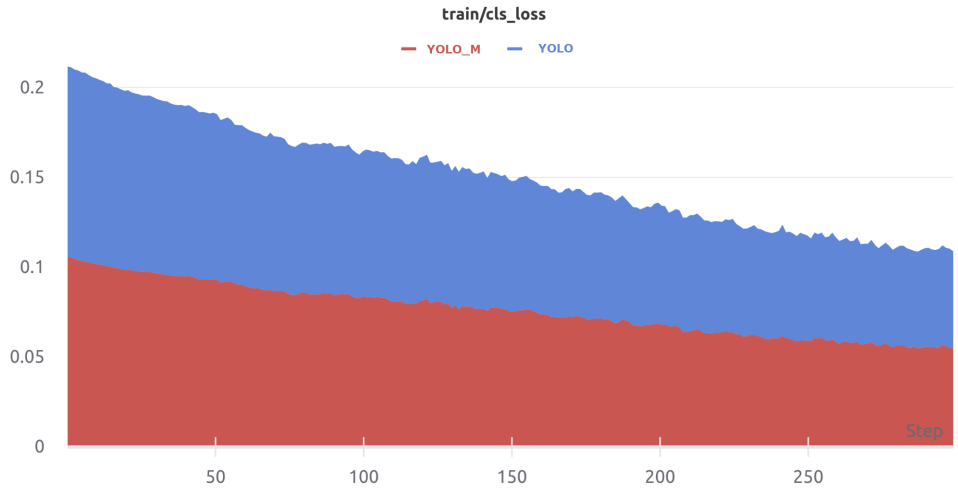

FIGURE 5.3. Classification loss comparison of our proposed model and YOLOv5

In Figure 5.3 we can see classification loss comparison between YOLOv5 and our proposed method. We can see that both classification loss and box loss are almost identical because of YOLO's feature: it both detects and classifies objects at the same time. That is why it is known as " you only look once" not like FasterRCNN or other models where we get different loss for classification and detection. Because those models use different methods for classification and detection inside the same model. So we can say our model beats YOLOv5 in both classification and detection loss.

5.2.5 Accuracy Comparison

This is the most important of any study. Here we will try to compare our model with the existing model and our parent model which we modified.

In Figure 5.4 we can see our model mAP almost touch 0.13 during epoch 300 where YOLOv5 touches only 0.12. There is almost 0.1 (10%) difference for our mixed Dhaka-Ai dataset. This result show that our model is doing better in both detection and classification. But the result of mAP is very low because of the less data in some classes. If we remove the class then the accuracy increases exponentially. But that will be researched in other studies. Currently we are studying 21 classes but only a few classes have enough data to be trained. We will be trying to reduce the classes so that we can improve our model for a few classes and use it in modern tasks. This model can easily detect bus, truck, car without much effort but when it comes to classes which have low data like wheelbarrow,

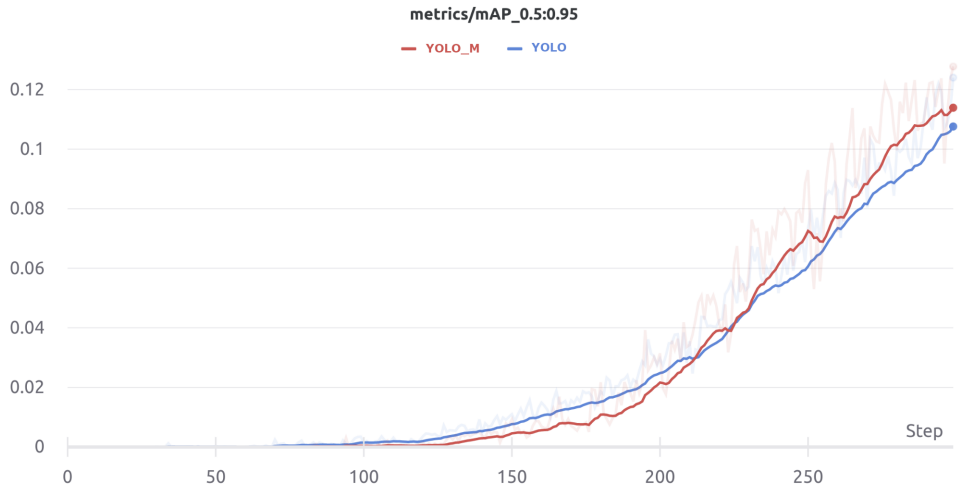

FIGURE 5.4. mAP comparison of YOLOv5 and our proposed model

ambulance, garbage van etc it can not detect these classes very well. We have plan to study this dataset and model without the minor classes which has low data compared to others.

TABLE 5.3. Performance comparison of proposed model with various models

Model Name	mAP (our dataset)	COCO mAP	Speed (ms)	Detect Small object and less overlapping
EfficientDet D4 1024x1024	12.27	48.5	133	YES
SSD ResNet152 V1 FPN 1024x1024 (RetinaNet152)	8.75	38.3	104	NO
Faster R-CNN ResNet101 V1 1024x1024	11.56	37.1	72	NO
Faster R-CNN Inception ResNet V2 1024x1024	11.89	38.7	236	NO
YOLO v5(1024x1024)	12.06	49.7	65	YES
YOLO v5(1024x1024) Modified (Proposed Method)	13.46	49.8	70	YES

This is very unique dataset none research has been made with this dhaka-ai mixed dataset. For this we run it into many different models like EfficientDet D4, SSD ResNet152 V1 FPN , Faster R-CNN ResNet101 V1 , Faster R-CNN Inception ResNet V2 , YOLO v5. After training the

5.3 Prediction Result Analysis

In this section we will discuss about prediction results of our model.

FIGURE 5.5. Sample inference of vehicles prediction near a road fence

In Figure 5.5 we can see our model classified rickshaw and bicycle with confidence score 0.69 and minivan with 0.5 confidence score which is true positive and the bounding box is also in good shape. But if we look closely we can see this model can't detect other vehicles which is other side of the fence and have low visibility. Sometimes this kind of situation gives false positive results.

FIGURE 5.6. Sample inference of multiple vehicles of same class

In Figure 5.6 we can see multiple vehicle of same class is there. We can see that it can easily identify rickshaws where confidence scores differ by the size of the vehicle. We can also see that sometimes two or more vehicles overlapped as one because of the IoU (intersection of union).

FIGURE 5.7. Sample inference of false positive and undetected vehicles

In Figure 5.7 we can see rickshaw is detected perfectly but the vehicle wheelbarrow and buses from other side of road fence is not detected. Because of the missing edges of the vehicles. Also we can see a false positive where our model detects the car as SUV with confidence score 0.32 and car as 0.30. It happens because of low data on SUV and wheelbarrow vehicles.

FIGURE 5.8. Sample inference of multiple types of vehicles

In Figure 5.8 we can see our model detecting multiple type of vehicles but missing some wheelbarrow. We can also see our model giving very low false positives and detecting vehicles in distance with a good confidence score. This image is in dusk but it is still predicating with good accuracy.

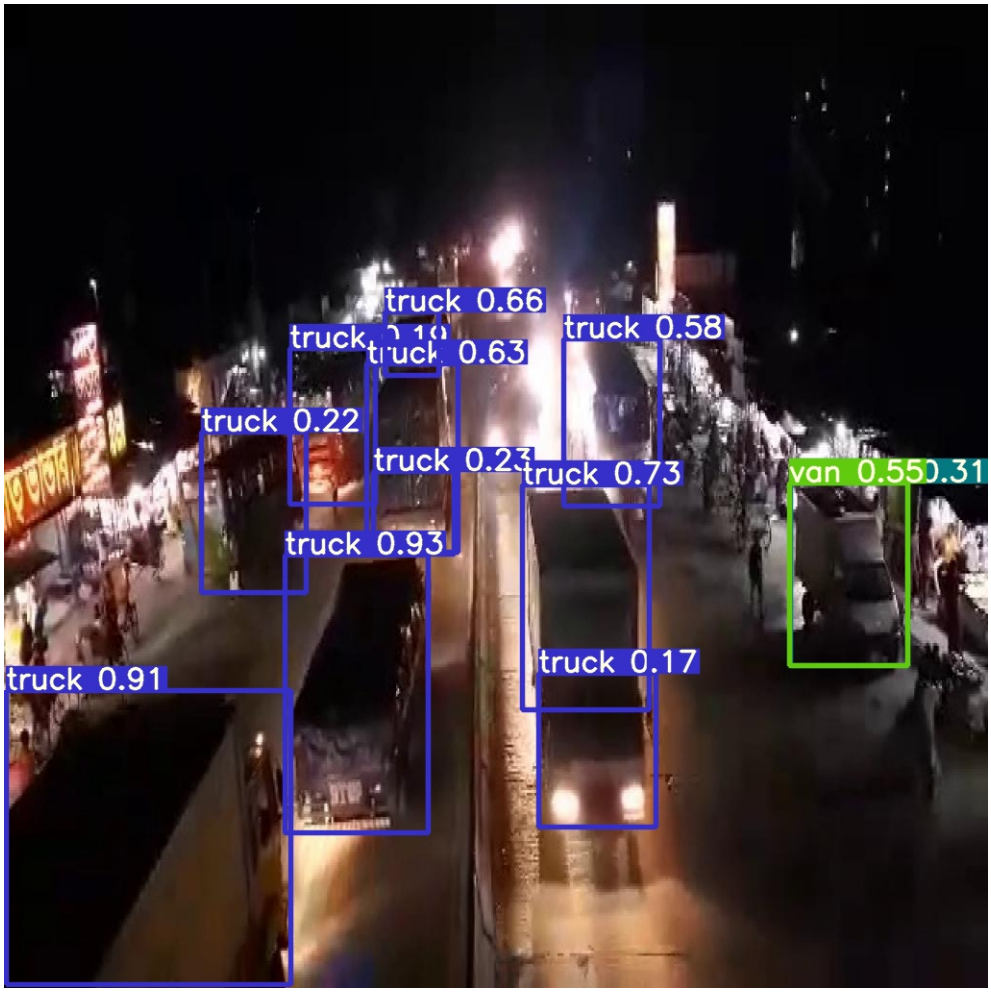

FIGURE 5.9. Sample inference vehicles at night

In Figure 5.9 our model detects vehicles during night. We can easily see that it can detect trucks with very high accuracy but fails to detect rickshaws and buses which have a high accuracy rate during day time. Because our dataset lacks images of night time. But we have many images of trucks during both night and day. That is why we can detect trucks and other related classes in night time accurately.

FIGURE 5.10. Sample inference of false positive during dusk

In Figure 5.10 we can see our model detecting small vehicle objects with good accuracy but it detects sides of road block building as bus because of low light and having similarity with edges of bus. It is a blunder mistake but it has very low confidence with this false positive result.

Chapter 6
Conclusion and Future Work

Chapter 6

Conclusion and Future Work

6.1 Introduction

This chapter discusses the conclusion of our research. Furthermore, it conveys the completion of proposed technologies. This section provides the future direction of the next study as well.

6.2 Conclusion

After having carried out this study, we have reached the following conclusions. Every training is unique and must be evaluated and tested in order to obtain a good detecting and tracking model. YOLOv5 is the fifth version of the YOLO object detection model and the one that has been used all through this work to detect vehicles. The most important thing when training YOLOv5 is to obtain a wide dataset of vehicles in all the type of positions, point of views, image resolution, forms, brands, etc. By this way, the model learns the concept of vehicle in all its forms that would appear in real life and therefore it is going to generalize well the concept of vehicle and is going to be able to detect vehicles in new images that have not been used to train the model. It is better to train the network in the first steps with low resolution images (416×416) and then picking randomly network resolution values and changing the network configuration in order to obtain a model capable of detecting vehicles in images of high and low resolution.

It is preferable to train YOLOv5 with anchor boxes obtained from the dataset because the prediction accuracy has been increased. An attempt has been made to reduce the training time by using initial weights obtained from the pre-trained weights for the COCO and ImageNet dataset, but this has meant a worsening in all areas. We have made some attempts to see if the network can be trained with a lower resolution than 416 or with a subdivision value lower than 16 but the network fails in the training. Finally, after analysing all the training's we have concluded that with the last training we get the best values according accuracy,

recall and mAP@0.5. This training has been done with the lowest value the training accepts for the subdivision parameter and the best data augmentation parameters.

The most important thing about a vehicle tracking system is the capability the model has to detect vehicles because if it is not able to detect some vehicles it will fail to track them. Our system is very accurate at detecting vehicles and that is why we have obtained so good results in the tracking process. The model that contains the YOLOv5 object detector and the SORT tracker must be adjusted to the area that the camera is recording. For example, a model that detects vehicles very well on a highway will not work so well in a camera installed on a street of a city because the interest point will have changed and we will have to modify some parameters of the model to suit it to the interests of the recorded area. That is what has happened with the highway video and the co-driver's seat video, the model for the first video must be adjusted with a lower confidence threshold than for the second one and also, the interest area is different in both videos so the adjustment of the frame is different. In short, the model once installed in a camera must be configured at the beginning depending on the background and the interest point and once we have obtained the parameters where the accuracy and speed are the best it will work alone.

6.3 Future Research Directions

YOLO imposes strong spatial constraints on bounding box predictions since each grid cell only predicts two boxes and can only have one class. This spatial constraint limits the number of nearby objects that our model can predict. Our model struggles with small objects that appear in groups, such as flocks of birds.Since our model learns to predict bounding boxes from data, it struggles to generalize to objects in new or unusual aspect ratios or configurations. Our model also uses relatively coarse features for predicting bounding boxes since our architecture has multiple down sampling layers from the input image. Finally, while we train on a loss function that approximates detection performance, our loss function treats errors the same in small bounding boxes versus large bounding boxes. A small error in a large box is generally benign but a small error in a small box has a much greater effect on IOU. Our main source of error is incorrect localizations.

Bibliography

[1] Statista. (2021). "Number of passenger cars and commercial vehicles in use worldwide from 2006 to 2015", [Online]. Available: https://www.statista.com/statistics/281134/number-of-vehicles-in-use-worldwide (visited on 08/17/2020) (cit. on p. 3).

[2] M. Webster. (2021). "Definition of vehicle", [Online]. Available: https://www.merriam-webster.com/dictionary/vehicle (visited on 08/17/2020) (cit. on p. 7).

[3] Wikipedia. (2021). "Vehicle", [Online]. Available: https://en.wikipedia.org/wiki/Vehicle (visited on 08/17/2020) (cit. on p. 7).

[4] S. Addala, "Research paper on vehicle detection and recognition", May 2020. doi: 10.13140/RG.2.2.34908.82561 (cit. on pp. 7, 8).

[5] H. Song, H. Liang, H. Li, Z. Dai, and X. Yun, "Vision-based vehicle detection and counting system using deep learning in highway scenes", *European Transport Research Review*, vol. 11, no. 1, p. 51, 2019. doi: 10.1186/s12544-019-0390-4. [Online]. Available: https://doi.org/10.1186/s12544-019-0390-4 (cit. on p. 8).

[6] Y. Tang, C. Zhang, R. Gu, P. Li, and B. Yang, "Vehicle detection and recognition for intelligent traffic surveillance system", *Multimedia Tools and Applications*, vol. 76, no. 4, pp. 5817–5832, 2017. doi: 10.1007/s11042-015-2520-x. [Online]. Available: https://doi.org/10.1007/s11042-015-2520-x (cit. on p. 8).

[7] A. Arinaldi, J. A. Pradana, and A. A. Gurusinga, "Detection and classification of vehicles for traffic video analytics", *Procedia Computer Science*, vol. 144, pp. 259–268, 2018, INNS Conference on Big Data and Deep Learning, issn: 1877-0509. doi: https://doi.org/10.1016/j.procs.2018.10.527. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1877050918322361 (cit. on p. 8).

[8] L. Jia, D. Wu, L. Mei, R. Zhao, W. Wang, and C. Yu, "Real-time vehicle detection and tracking system in street scenarios", in. Jan. 2012, vol. 289, pp. 592–599, isbn: 978-3-642-31967-9. doi: 10.1007/978-3-642-31968-6_70 (cit. on p. 8).

[9] ——, "Real-time vehicle detection and tracking system in street scenarios", in *Communications and Information Processing*, M. Zhao and J. Sha, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 592–599, isbn: 978-3-642-31968-6 (cit. on p. 8).

[10] R. Baran, T. Rusc, and P. Fornalski, "A smart camera for the surveillance of vehicles in intelligent transportation systems", *Multimedia Tools and Applications*, vol. 75, no. 17, pp. 10 471–10 493, 2016. doi: 10.1007/s11042-015-3151-y. [Online]. Available: https://doi.org/10.1007/s11042-015-3151-y (cit. on p. 9).

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection*, 2016. arXiv: 1506.02640 [cs.CV] (cit. on pp. 14, 18).

[12] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf", Nov. 2011, pp. 2564–2571. doi: 10.1109/ICCV.2011.6126544 (cit. on p. 20).

[13] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography", *Commun. ACM*, vol. 24, no. 6, pp. 381–395, Jun. 1981, issn: 0001-0782. doi: 10.1145/358669.358692. [Online]. Available: https://doi.org/10.1145/358669.358692 (cit. on p. 23).

[14] C.-Y. Wang, H.-Y. M. Liao, I.-H. Yeh, Y.-H. Wu, P.-Y. Chen, and J.-W. Hsieh, *Cspnet: A new backbone that can enhance learning capability of cnn*, 2019. arXiv: 1911.11929 [cs.CV] (cit. on p. 24).

[15] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition", in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., Cham: Springer International Publishing, 2014, pp. 346–361, isbn: 978-3-319-10578-9 (cit. on p. 24).

[16] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, *Microsoft coco: Common objects in context*, 2015. arXiv: 1405.0312 [cs.CV] (cit. on p. 26).