# Dhaka Traffic Detection

*Capstone project report to be submitted in partial fulfillment of the requirements for the degree*

*of*

**Bachelor of Science in Computer Science and Engineering**

*by*

**Mohammad Navid Nayyem**
**172014003**

**KH. Nafiu Nur Rashid**
**172014026**

**Fabliha Bushra**
**163014029**

Under the supervision of

**Mohammad Rifat Ahmed Rashid**


**COMPUTER SCIENCE AND ENGINEERING**

**UNIVERSITY OF LIBERAL ARTS BANGLADESH**

**SPRING 2021**

# DECLARATION

| | |
|---|---|
| **Project Title** | Dhaka Traffic Detection |
| **Authors** | *Mohammad Navid Nayyem*, *KH. Nafiu Nur Rashid*, and *Fabliha Bushra* |
| **Student IDs** | 172014003, 172014026, and 163014029 |
| **Supervisor** | Mohammad Rifat Ahmed Rashid |

We declare that this capstone project report entitled *Dhaka Traffic Detection* is the result of our own work except as cited in the references. The capstone project report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.


**Mohammad Navid Nayyem**
**172014003**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

**KH. Nafiu Nur Rashid**
**172014026**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

**Fabliha Bushra**
**163014029**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh


**Date: June 13, 2021**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh
Mohammadpur, Dhaka - 1207

# CERTIFICATE

This is to certify that the capstone project report entitled **Dhaka Traffic Detection**, submitted by **Mohammad Navid Nayyem** (Student ID: 172014003), **KH. Nafiu Nur Rashid** (Student ID: 172014026) and **Fabliha Bushra** (Student ID: 163014029) are undergraduate students of the **Department of Computer Science and Engineering** has been examined. Upon recommendation by the examination committee, we hereby accord our approval of it as the presented work and submitted report fulfill the requirements for its acceptance in partial fulfillment for the degree of *Bachelor of Science in Computer Science and Engineering*.

**Mohammad Rifat Ahmed Rashid**
**Assistant Professor**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

**Dr. Syed Akhter Hossain**
**Professor and Head**

Department of Computer Science and Engineering
University of Liberal Arts Bangladesh

**Place: Dhaka**
**Date: June 13, 2021**

# ACKNOWLEDGEMENTS

We would like to express our deep and sincere gratitude to our research supervisor, *Mohammad Rifat Ahmed Rashid*, for giving us the opportunity to conduct research and providing invaluable guidance throughout this work. His dynamism, vision, sincerity and motivation have deeply inspired us. He has taught us the methodology to carry out the work and to present the works as clearly as possible. It was a great privilege and honor to work and study under his guidance.

We are greatly indebted to our honorable teachers of the Department of Computer Science and Engineering at the University of Liberal Arts Bangladesh who taught us during the course of our study. Without any doubt, their teaching and guidance have completely transformed us to the persons that we are today.

We are extremely thankful to our parents for their unconditional love, endless prayers and caring, and immense sacrifices for educating and preparing us for our future. We would like to say thanks to our friends and relatives for their kind support and care.

Finally, we would like to thank all the people who have supported us to complete the project work directly or indirectly.

**Mohammad Navid Nayyem**, **KH. Nafiu Nur Rashid** and **Fabliha Bushra**
University of Liberal Arts Bangladesh
Date: June 13, 2021

To my mother
**Nurun Nahar Begam**
and father
**Late Md. Abdur Rouf**

–Mohammad Navid Nayyem

To my mother
**Nadira Sultana**
and father
**KH. Harunar Rashid**

–KH. Nafiu Nur Rashid

To my mother
**Shamima Umme Kulsum**
and father
**Md. Jahangir Alam**

–Fabliha Bushra

# ABSTRACT

Dhaka is the capital city of Bangladesh. Among the most populated city in the world, Dhaka is one of them. **Traffic jam** of this city is a big problem and sometimes it seems that this city is cursed with traffic jam. The traffic jam of Dhaka is very complicated and it is a new complex challenge in the sector of automated traffic detection. This problem can solve using **Artificial Intelligence based technology**. The detection of an object is associated with computer vision. Computer vision works with digital images and videos to understand the contents or objects presented in the images or videos. Rectangular or square bounding box is used to **detect and count vehicles** from the images or videos according to vehicles categories by estimated distance from the video recording of closed-circuit television (CCTV) camera. Here we need to mention that the "Counting Vehicles" part have been included in the "Scope for Future Work" part.

**Keywords**: Traffic jam, Artificial Intelligence based technology, detect and count vehicles

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

Dhaka, the capital city of Bangladesh is one of the most populated city in Asia. The populations of Bangladesh are increasing every day and mainly Dhaka is affected by a huge amount of traffic jams. The traffic jam of Dhaka is very complicated and it is a new complex challenge in the sector of automated traffic detection. This problem can solve using Artificial Intelligence-based technology. The detection of an object is associated with computer vision. It describes a system that can detect the location of the desired object in an image. Computer vision works with digital images and videos to understand the contents or objects presented in the images or videos. The rectangular or square bounding box is used to detect and count vehicles from the images or videos according to vehicle categories by estimated distance from the video recording of closed-circuit television (CCTV) camera. This application-based system is beneficial for maintaining many systems such as managing traffic, controlling traffic and a better parking management system.

## 1.1 Problem Statement

The application-based system is used to maintain traffic in Bangladesh. The traffic jam can reduce as well as the savings of time is increased. This is the main motto of developing this system. The objectives of this project are: vehicle images annotation, create dataset using annotated image with XML file, train a model using dataset, integrate the model in the android app and develop a fully functioning Android app to detect the vehicle classes perfectly from video or images, easy to use, the wastage of time will reduce as the traffic jam can control, provide good

service to the users, reduce the difficulty to manage traffic and beneficial for better parking management system.

## 1.2 Aims and Objectives

The aims and objectives of this project are:

- Vehicle Images Annotation.
- Create dataset using annotated image with XML file.
- Train a model using dataset.
- Exporting our Model for Android App
- Develop a fully functioning Android App to detect the vehicle classes perfectly from video or images.
- Easy to use.
- The wastage of time will reduce as the traffic jam can control.
- Provide good service to the users.
- Reduce the difficulty to manage traffic.
- Beneficial for better parking management system.

## 1.3 Motivation

Transport is an important part of our day-to-day life. Now a days, traffic jam is the most unbearable problem in Bangladesh. People in our country face a lot of problems on the roads when they go out to go somewhere they need to and most of the time they get frustrated because of this traffic jam as there are too many vehicles out there in the road.

**Dhaka Traffic Detection** is an Android application, where the user is able to detect vehicles using this app. By using Artificial Intelligence, the critical condition of the traffic of Bangladesh can solve. It is beneficial for Bangladeshi people, as they want a better solution for the traffic system. The traffic jam can control and the valuable times of people can save.

# Chapter 2

# Literature Review

## 2.1 Object Detection

In computer vision field, object detection plays a vital role. Object detection means locating or identifying objects in frame of video sequence. The main application areas of object detection: Motion based recognition, automated surveillance, video indexing, human-computer interaction, traffic monitoring, vehicle navigation and etc. Most of the object detection mechanism used information from single frame for detecting an object. But some of the object detection mechanism used temporal information which is computed from sequence of frames. The first step of real time object detection is to identify the region of interest of the video. Some of the object detection methods are: Point Detector, Background Modelling, Segmentation, Optical Flow and Supervised Classifier. Background modelling has two main approaches. One is Recursive algorithm and another one is Non-recursive algorithm. **Object Detection Algorithms:** An accuracy of detecting object in background subtraction-based object detection algorithms, an optical flow-based object detection algorithms and frame differencing based object detection algorithm. **Background Subtraction Based Motion Detection and Agent Based Tracking:** A Scene feature based algorithm was proposed in for detecting counter flow in security related surveillance in airport. They addressed the two main problems: (i) Most of the cameras deployed in security surveillance networks have poor resolution. It will create negative effects on tracking algorithm. (ii) 24/7 basis operation of automatic video analytics algorithm some time will provide higher false positive rate in tracking. To avoid such problems, they used novel classifier to identify scene

feature in the image and KLT optical flow tracking algorithm. **Background Subtraction Based Object Detection Using Canny Edge Operator:** When the number of discretely moving object increases, the understanding of video scene becomes more difficult. Motion feature (viz location, scale, score (magnitude), direction and velocity) filtering based event detection was proposed in to detect event in crowded area. When the moving objects are relatively fast to frame rate, the detection can be especially difficult. To overcome these problem, Manisha Chaple, proposed background subtraction-based detection and optical flow algorithm for tracking in. And also, they used centroid in frame to detect the distance and velocity of the moving object. They established statistical based reliable background model and used dynamic optimization threshold method to detect moving object. In order to eliminate noise and eliminate the background disturbance problem, they applied dilation and erosion processing. Due to occlusion, lightning changes and other factor, abandoned object detection in complex video surveillance is too difficult. Yingli Tian, modeled three Gaussian mixture-based Background subtraction. Among upper all of these, optical flow-based detection provides more accuracy than other techniques.[1]

Intelligent vehicle detection and counting are very important in the field of highway management. Due to the different sizes of vehicles, the detection remains a challenge that directly affects the accuracy of vehicle counting. Vision-based vehicle object detection is divided into traditional machine vision methods and complex deep learning methods. Traditional machine vision can distinguish the from a predetermined background and this can be divided into three categories: background subtraction; continuous video frame; optical flow. CNNS has surprising success in the field of vehicles objects detection due to its solid ability to learn image features while simultaneously performing many related tasks. The detection can be primarily divided into two categories: one stage and two stages. In the One stage method the positioning problem the object bounding box is converted into regression problem processing. The two-stage method makes a candidate box of object thereby classifying the object by a convolutional neural network. The single shot multibox Detection (SSD) and you only look once (YOLO) one among the most important component of stage one method. For effective image detection traffic image dataset can be used according to how the image is collected,

it is divided in to three categories: car camera, by surveillance camera and non-monitoring camera. To initiate the system they, at first, need video data of the traffic scene, then find the road surface and divide it. ORB feature tracks multiple objects and acquires vehicle traffic data. YOLOv3 is used to detect. Object in highway traffic scene. the area of surveillance is great, and given that their goal is vehicle detection, they use road surface segmentation to aid their agenda. The most common segmentation method is Gaussian mixture model, which extracts the background in first 500 frames, helping to extract vehicles from backgrounds having multiple peak characteristics. In practice, of the entire dataset, 80 percent is training set and 20 percent is test set; high rate of training set gives an accurate model.[2]

## 2.2 Traffic Detection

We live in era of technology where we moving towards making a smart city. A smart city with a smart AI based traffic monitoring system. The author of this paper called **"Automated AI Based Road Traffic Accident Alert"** they are proposing a more advance traffic monitoring system which can identify and detect moving objects like cars, bikes etc. in live camera feeds. We make an **advance Artificial intelligence-based system** can detect occurrence of accident and alert to nearby hospitals/ambulance or Traffic policemen in real-time. Their system is based on **Neural Network** and **Deep Learning** of object detection along computer vision technology and several methods and algorithms. They **detect, classify, track** and compute moving object **velocity** using **convolution neural network (CNC)**. Accident detection system trained by using **Regression** base algorithm called Yolo algorithm. In this system first they create their own dataset designing CNN. **RCNN** is used to extract region by applying selective search. The method is named RCNN because Regional proposals are combined with CNNs. Here two basic concepts are combined and applied in the R-CNN. The first concept is to apply an efficient convolutional neural network from bottom to up region proposals to locate and the second concept is to apply supervised training for field-specific tuning task when insufficient training images in entered into the system, resulting in significant improvement of performance. They use CNN (Convolutional Neural Network) for Feature extraction. Take an image as input **extract region proposal** using **R-CNN**.

After attribute calculation using CNN, they classify region using. Here **Caffe Deep** Learning framework was used to carry out this training. **YOLO (you only look once)** algorithm achieves its result by applying a neural network on an image.[3]

# Chapter 3

# Project Specification

## 3.1 Dataset Description

The capital city of Bangladesh, Dhaka, conveys merely 7 percent of traffic roads (in comparison to 25 percent of urban standard) in presence of approximately 8 million computers per day with 306 sq km of the total area. The scenario of capital' traffic is unique in the sense of possessing new complex challenges in terms of automated traffic detection. As a solution, advances in Artificial Intelligence-based technologies and ICT solutions are used for Dhaka's automatic road acquisition problems in visual images. This AI-Based Dhaka Traffic Detection is newer what challenges to aim at accessing the capability of detecting and recognizing the vehicles in Dhaka city. Modern cities met this solution and here in modern cities, culture of differences is living and communicating with each other. And the users watch languages, scripts in that way which prevents to use such a prior knowledge. And at this same moment, the researches from those regions, experts in Artificial Intelligence and has the interest to explore possibilities, can bring under a community named "network community". Here the method of objective is that, this dataset is containing images of different categories of vehicles, and an image may carry more than one categories of vehicles. Here in this dataset, the vehicles are categorized into 21 different classes. This arrangement builts the dataset smoothly useful for multiple vehicles detection as well as to recognize them and what worth it much. LabelImg tool is a graphical image annotation tool which is used to annotate images with a bounding box of this dataset. The extension of the annotation files are in XML format. All images has been captured by using Android Smartphone or iPhone. According to the project the considered classes of ve-

hicles are: ambulance, army vehicle, auto rickshaw, bicycle, bus, car, garbagevan, human hauler, minibus, minivan, motorbike, pickup, policecar, rickshaw, scooter, suv, taxi, three wheelers (CNG), truck, van and wheelbarrow. The image of the classes of vehicles is given below:


Figure 3.1: Classes of Vehicles

## 3.2 Deep Learning Approach

In this era, real time object detection technique is becoming very much popular in detecting objects from videos or images. There are many frameworks and YOLO (You Only Look Once) is one of them. It is now the first choice in machine learning engineering and data scientists are very much fond of it. There are total five versions of YOLO including YOLO, YOLOv2, YOLOv3, YOLOv4 and YOLOv5. For this project, the model is trained by using YOLOv5s, subsector of YOLOv5. It is the latest version of YOLO.

As YOLO is real time object detection framework, it is very much capable of detecting vehicle correctly. It is a single stage object detector. That means it doesn't undergo the region purpose step and only predict over a limited number of bounding boxes.

YOLO algorithms divide the input images into the SxS grid cells and this grid is responsible for object detection. This grid system predict the bounding boxes for the detected object.

The number of epochs is a hyper parameter that defines the number times that the learning algorithm will work through the entire dataset. The batch size is a number of samples processed before the model is updated.[4] In this project, at the time of training the model, the batch size was 16.

By following the below steps, YOLOv5 implemented in this project.

(i) Clone the YOLOv5 repository from GitHub to "Kaggle Notebook".
(ii) Install the dependencies using the pip command.
(iii) Import torch module and display to display our output image inside the notebook.
(iv) 100 epochs passes through the training dataset.
(v) The trained image size was 640*640.
(vi) Using a command, detect.py runs inference on a variety of sources and it automatically downloaded the latest model.

## 3.3 Application Specification

The application is an Android-Based application. It can detect different categories of vehicles from videos or images. In the application, there are four different types of operations. Among them, one operation is to detect the vehicles from three built in test images. Another operation is to choose images from android phone's gallery and detect vehicles from them and the last operation is to detect vehicles from live video or live image. Mobile camera is used to perform this operation. In live video, the system can detect vehicles by its own but to detect the vehicles from images, the "Detect Vehicle" operation is used.

## 3.4 Specific Requirements

### 3.4.1 Software Requirements(for user)

(i) Anaconda Prompt
(ii) Python v3.8.3
(iii) Android Studio v4.1.2
(iv) Git Bash
(v) Kaggle Notebook
(vi) Adobe XD

### 3.4.2 Hardware Requirements(for user)

(i) Acer/Dell/HP Laptop (GPU Included)
(ii) Android Smartphone with Minimum Android 11.0 Supported
(iii) Closed-Circuit Television(CCTV) Camera

### 3.4.3 System Attributes

(i) **Test Image:** These images are built in images. There are three built in images in the system.

(ii) **Select Image:** This operation can choose images from mobile's gallery.

(iii) **Go Live:** Using the mobile camera, this operation can detect vehicles from a live video or image.

(iv) **Detect Vehicles:** To detect the vehicles from images, this operation is used.

# Chapter 4

# System Design

## 4.1 Empirical Data Analysis

In our dataset, there are 4507 images for training. And we have trained all of these images in Kaggle notebook. There are 21 classes in our datasets. The name of the classes are: ambulance, army vehicle, auto rickshaw, bicycle, bus, car, garbagevan, human hauler, minibus, minivan, motorbike, pickup, policecar, rickshaw, scooter, suv, taxi, three wheelers (CNG), truck, van and wheelbarrow.

**Challenges:**
1. Mislabeled Images
2. Class imbalance
3. Insufficient samples for some classes
4. Different camera angle
5. Highly congested vehicles
6. Variance in aspect ratio between train and test images

**Mislabeled images:** As we have 4507 images, but on some images, we have found that they were not labeled perfectly. Some vehicles would remain unlabeled. It was our challenge to find those images and make them label perfectly so that at the time of our model train, it may train as perfectly as possible. And here we used roboflow.ai to find out the images.

**Class imbalance:** There are 21 classes in our dataset. But there was an extreme class imbalance in our dataset. It means we had found some overrated vehicles

category as well as some underrated vehicles in our dataset. So we collected more images of those underrated vehicles. But capturing images might conflict with some privacy issues. That is why sometimes we took videos and extract images from those videos. That is how we tried to enrich our dataset and made the imbalance classes balanced.

**Insufficient samples:** For some classes, there were very few samples in our dataset. And this problem was also solved by collecting more images of those vehicles.

**Different camera angle:** The most challenging part for creating our dataset was different camera angle. It was solved by using augmentation named "Mosaic Augmentation" and "Multi-scale Training".

· **Mosaic Augmentation:** The mosaic data augmentation, which combines four images into four tiles of random ratio. We use data augmentation to reduce overfitting. It was very much helpful using mosaic augmentation because Mosaic augmentation helps the model learn to address the well-known "small object problem" - where small objects are not as accurately detected as larger objects. That means it can detect very much smaller objects correctly.

· **Multi-scale training:** In order to train the model to be robust to input images of different sizes, a new size of input dimension is randomly sampled every 10 batches.

**Highly congested vehicles:** There are some images where the vehicles are highly overlapping and for the model it was difficult to detect the vehicles individually.

Figure 4.1: Highly congested vehicles

**Aspect ratio variance:** A noticeable variance in aspect ratio was resolved by using the "Multi-scale Training".

YOLOv5 was released by ultralytics in which they said that YOLOv5 model performs faster than EfficientDet Model like D0,D1,D2,D3 or D4 and it required lesser GPU memory so we used YOLOv5s model for training our model.

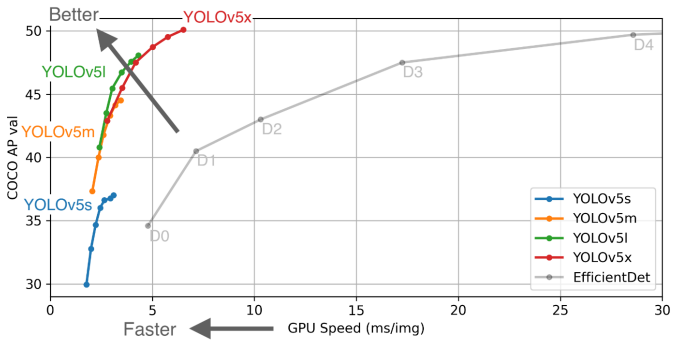


Figure 4.2: Comparison between YOLOv5 and EfficientDet

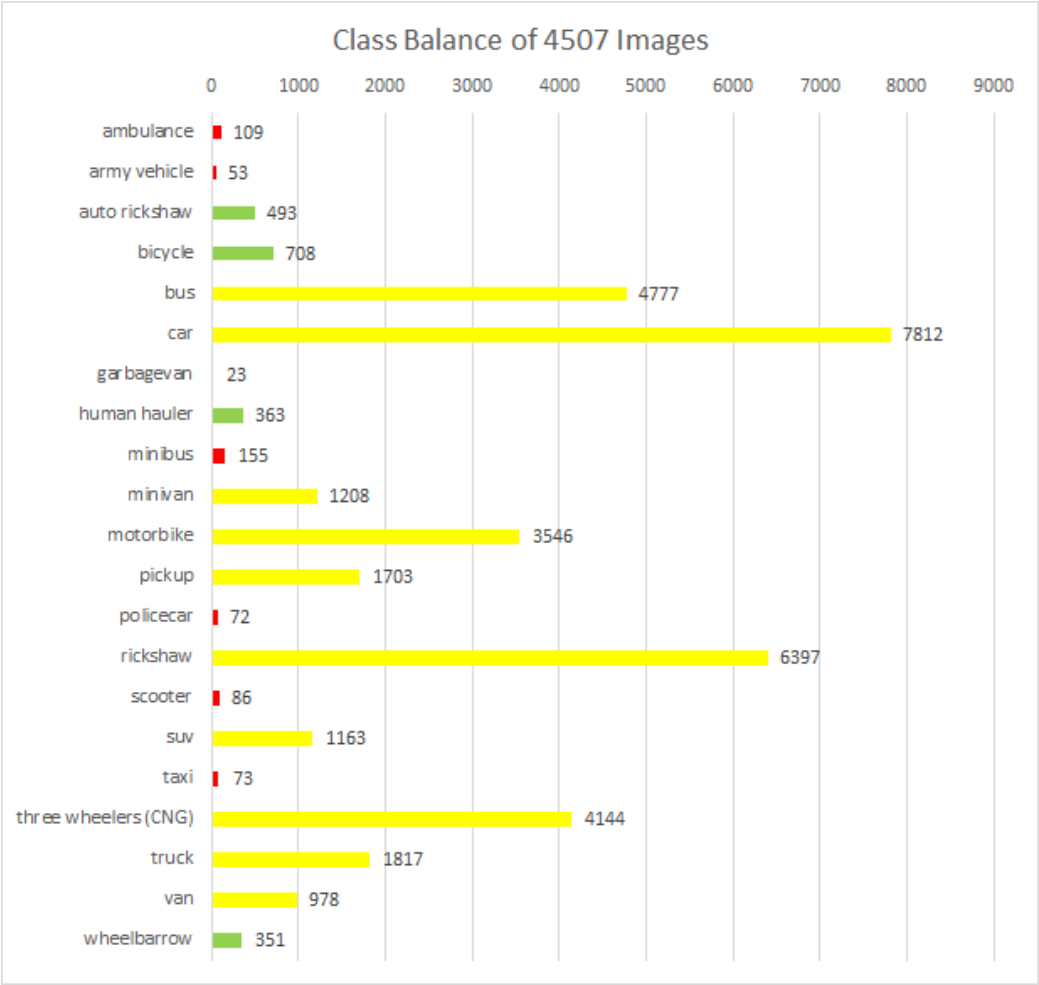


Figure 4.3: Class Balance

The sample statistic of 4507 images is shown in **Figure 4.3**. There are some privacy issues while capturing pictures of army vehicles from roads and also, we didn't saw that amount of army vehicles, policecar, garbagevan, taxi on the road. In the case of army vehicles, if we wanted to capture more pictures of it, we needed to go to the cantonment area and also needed to take permission from the officers. In this way we can enrich this category in our dataset.

## 4.2 Use Case Diagram

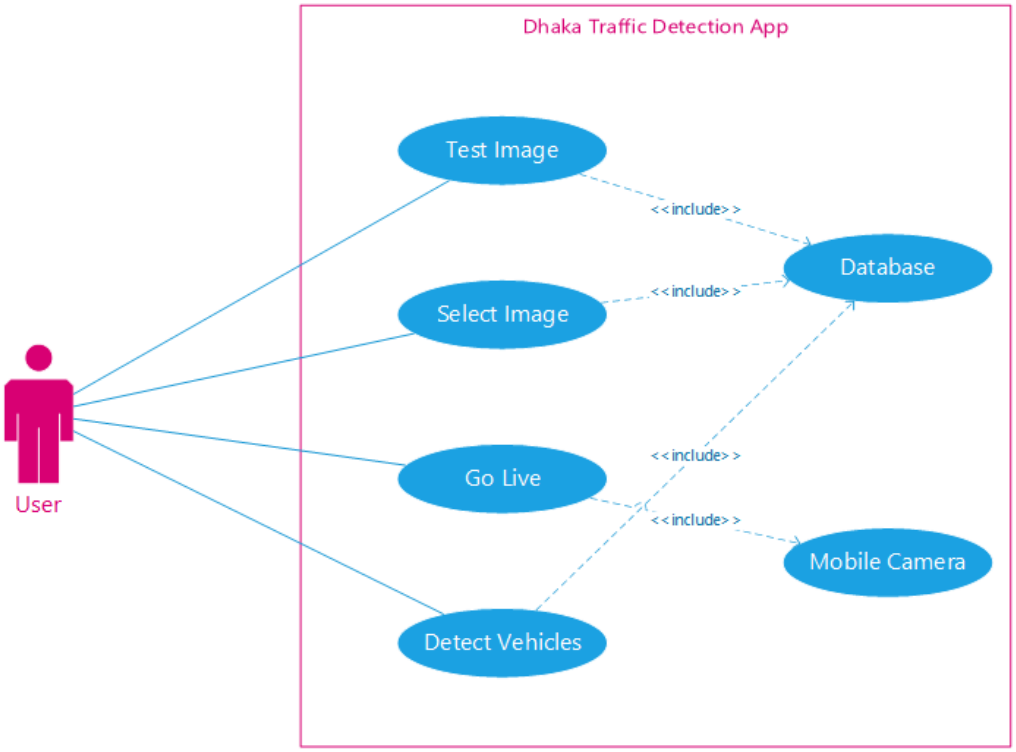


Figure 4.4: Use Case Diagram

This is the Use Case Diagram. When the user enters the Dhaka Traffic Detection App, he can find four types of base use cases. The base use cases are Test image, Select Image, Go Live and Detect Vehicles. Database and mobile camera are included use case. Base use case is always executed when included use case is executed. Test Image, Select Image and Detect Vehicles are executed while database is executed. Go Live is executed when the mobile camera is executed. After executing all of this, user can access to the photo gallery and then train model run on the image to detect vehicles.

## 4.3 Data Flow Diagram (DFD)


Figure 4.5: Context Level Data Flow Diagram

This is the Context Level Data Flow Diagram as the final outcome is an android-based application, the user have to install the app then from the mobile camera the vehicles will be detected in real time with class name and probability value and store in the app and then from the app the total number of vehicles detected with class name by bounding boxes will be shown to the user.

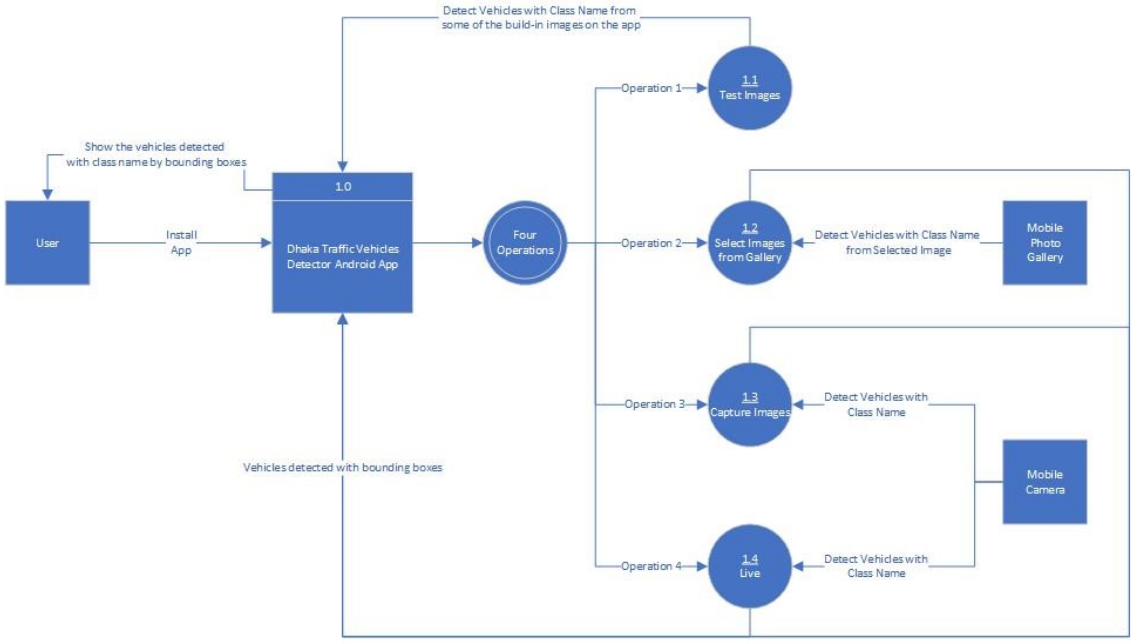


Figure 4.6: Level 1 Data Flow Diagram

This is the Level 1 Data Flow Diagram. In this diagram at first user will install the application. After installing the application user will enter in this application, they

will find four operations. In first operation there will be some test images. In this operation they will detect vehicles with class name some of the build in images on the app. Then user will be shown the vehicles detected with class name by bounding box. The second operation of this application is to select images from gallery. In this operation user will be entered in their mobile gallery and selected image. The selected image will be processed and will give a result inside the bounding box which will be shown to the users. The third operation is capturing image. User can capture photo by using his/her mobile camera and it will detect vehicles with class name and shown the result by bounding boxes to the user. Live is fourth operation of this application. User can do live by his/her mobile camera and this will detect vehicles with class name and user can see them by bounding boxes from the real-time video.

## 4.4 Sequence Diagram

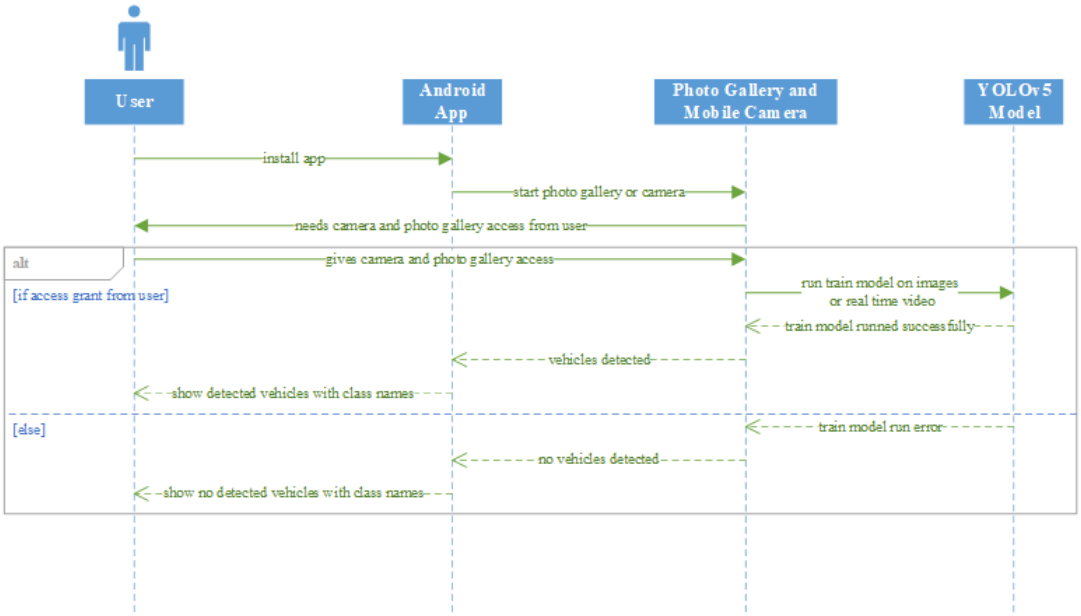


Figure 4.7: Sequence Diagram

This is the Sequence Diagram. Here, Actor is an user and the classes are arranged

in an order - Android App, Photo Gallery and Mobile Camera And lastly YOLOv5 model. First, the user will install the application and then will send a message to Android App class. Android App class will then handover a message to Photo Gallery or Camera. Photo Gallery will finally send a message to the user to confirm the access to open the Photo Gallery or camera. If the access is granted by user, then the user can select images from Photo Gallery or go to live by camera and start real-time video. The trained model is then run on the images to detect vehicles from it. Here, the trained model is YOLOv5 Model. After train model run on images, the vehicles are detected with class name and show it to the user. If the access is not granted by user, then the user cannot select images from Photo Gallery or cannot go to live by camera and start real-time video. The trained model will not run as well as the vehicles will not be detected and user will see nothing.

# Chapter 5

# Methodology

## 5.1 Overview

The steps for machine learning pipeline have been discussed.

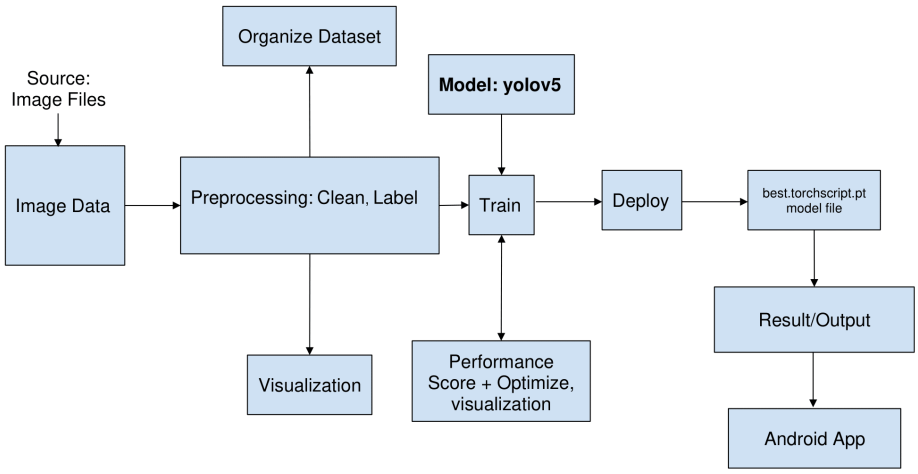


Figure 5.1: Machine Learning Pipeline

In the machine learning pipeline, our source is image files. The image data is then preprocessed by cleaning and labelling and we can visualize the statistics. From preprocessing we generate the organize datasets. After preprocessing the model is trained with organized dataset by using yolov5. The performance score can be visualized during model train and optimized it. Then the train model is deployed with exported torchscript.pt model file. After inserting the model file in the android app, the model runs on the images to detect vehicles with bounding boxes. Then the output is visualized by using the android app. In the android app, there are some of the inbuild images which can detect vehicles after running the model on it and the user can capture an image and upload it or go to live camera for real time video and then the result will be shown to the user.

YOLO ("You Only Look Once") is an effective real-time object recognition algorithm. [5] There are total five versions of it including YOLO, YOLOv2, YOLOv3, YOLOv4 and YOLOv5. For our model train, we used YOLOv5. It is the latest version of YOLO.

## 5.2 Deep Learning Architecture

**Architecture of YOLOv5**

As yolo is real time object detection framework, it is very much capable of detecting vehicle correctly. It is a single stage object detector. That means it doesn't undergo the region purpose step and only predict over a limited number of bounding boxes. Whereas RCNN, R-FCN these algorithms cannot do this. There are three major part of YOLOv5.

**Model backbone:** It extract important features from the given input image. It is a convolution neural network that aggregates and forms image features at different granularities.[6] The work of backbone is to extract the important features from the images. There are many kinds of backbones such as darknet, CSPNet etc.

**Model Neck:** It generates features pyramids that helps to identify the same object with different sizes and scales. A series of layers to mix and combine image

features to pass them forward to prediction.[6] The work of neck is to generate features pyramid. Which can deal with multiscale correctly. For feature pyramid, there are many modules are suggested. We can add here Path Aggregation Network as an example.

**Model Head:** Consumes features from the neck and takes box and class prediction steps.[6] Applies anchor boxes on features and generates final output vectors with class probabilities, object ness scores and bounding boxes. Well Anchor boxes is one of the most important parameters we can tune for improved performance on our dataset. In fact, if anchor boxes are not tuned correctly, our neural network will never even know that certain small, large or irregular objects exist and will never have a chance to detect them.

The post processing task similar to yolov3. The diagram of Object Detection Process is given below: [7]

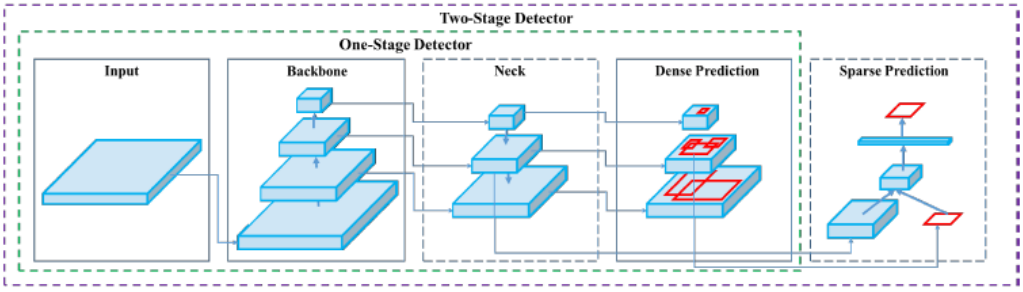


Figure 5.2: Object Detection Process

**Other Aspects of YOLOv5**

**Augmentation used:**
First of all, we need to know what data augmentation is. Data augmentation makes transformations to the base training data to expose the model to a wider range of semantic variation than the training set in isolation.[5]

Here we used mosaic augmentation. It was very much helpful using mosaic augmentation because Mosaic augmentation helps the model learn to address the well

know ”small object problem” - where small objects are not as accurately detected as larger objects. That means it can detect very much smaller object correctly.

**Bounding boxes anchor:**
It learned from dataset automatically with K-means and genetic learning algorithm. Well in yolov3 and yolov4, the bounding boxes were calculated manually, but here in yolov5, bounding boxes are calculated automatically.

## 5.3 Software Development Process

The V-Model denotes verification and validation model where each phase has a testing step associated with developing stages. There are four design phases in the V-Model. They are: Requirement Analysis, System Design, Architectural Design and Module Design. There are four testing activities proposed in the V-Model. They are: Unit Testing, Integration Testing, System Testing and User Acceptance Testing.

Requirement Analysis consists of a detailed communication with the customer to understand their needs and expectations. System Design contains system configuration as well as complete hardware configuration and product development enhancement. Architectural Design is divided into advanced modules that take different functions. Data transfer and communication between internal and external modules is well understood. In Module Design, the program enters small modules. The detailed design of the modules is specified, also known as Low-Level Design.

Unit testing plans are developed during module phase. It is executed to eliminate bugs at code. Integration Testing comes after completing unit testing which is integrated and the system is tested. System Testing test the complete application with its functionally, inter dependency and communication. It tests the functional and non-functional requirements of the developed application. User Acceptance Testing is performed in a user environment that resembles the production environment.

The V-Model is used for the development of this project. For this project, the Unit Testing activities that the team should write a segment of code to test a specific function in application. They could also separate the function to test it more thoroughly and try to remove the bugs from code. The Integration Testing activities that the team should test the whole system. It tests all of the modules when they work together. To run the system, all of the modules need to work together. The purpose of integration testing is that if all these modules can communicate or interact with each other properly so that the system can run. The System Testing activities that the team should check and test the complete application, verifies functional and non-functional requirements of the application. Their goal is to detect the defects and bugs both within the interfaces and the software. System testing ensures the complete feasibility of the ride sharing system. They should focus on the design of the software, performance and the believed expectations of the customer. System testing means that if the whole program is working when all of the modules are functioning. The SDLC V-Model figure is given below: [8]

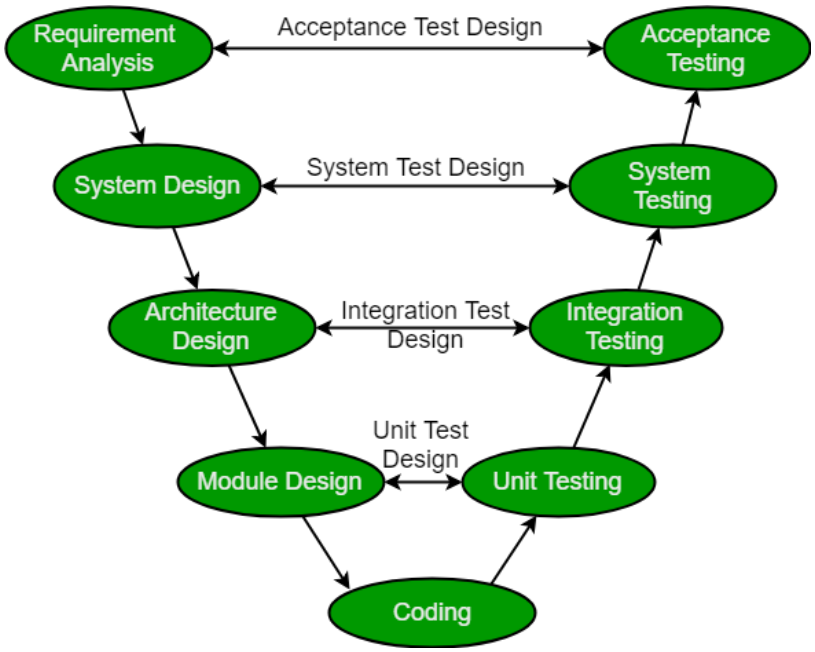


Figure 5.3: SDLC V-Model

## 5.4 Design Pattern

### 5.4.1 Model-view-controller

Model-View-Controller is an architectural pattern that divides the system into three logical elements: model, view, and controller. Each of these features is designed to handle specific aspects of the app's development.

The model stores the application data. It has no information about the interface. The model communicates with the database and layers of the network. The view is the User Interface layer that handles visual elements on the screen. In addition, it provides visibility of the data stored in the Model and provides connectivity to the user. The controller create a relationship between View and Model. It has the original concept of the app and is informed of the user's behavior and updates the Model according to each need. The Model-View-Controller pattern of this project is shown below:

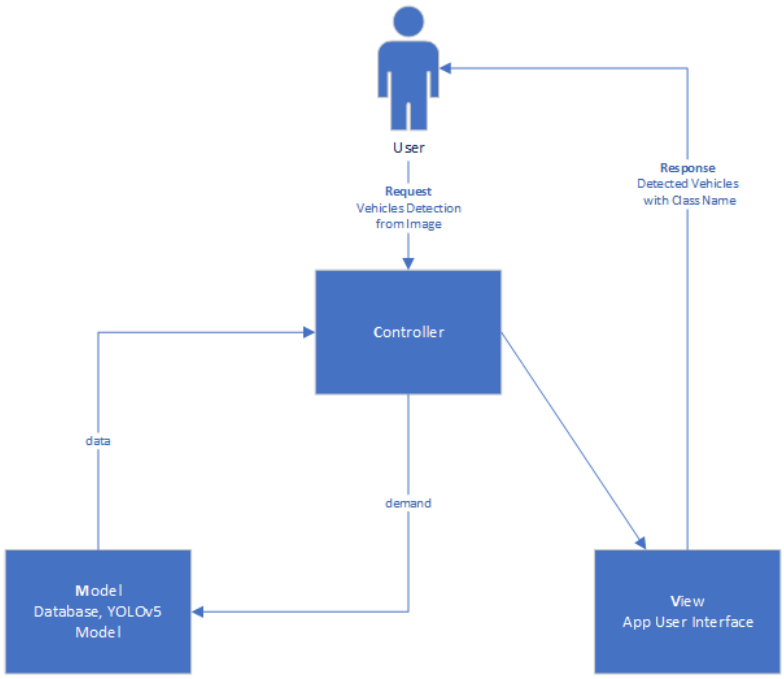


Figure 5.4: The MVC Pattern

This is the MVC Pattern. Here the user send requests to the controller for vehicles detection from image and the controller create a relationship with the View and the Model. The controller demand the data from the Model which communicated with the database. From the model, the data go to the controller and the vehicles is detected from YOLOv5 trained model and show it to the View which is User Interface. At last, the user get response of detected vehicles with class name and see it from View.

## 5.5 Technology Used

Java Programming Language has been used for developing this Android application of this project. Java is the default development programming language to write Android application. This programming language was first introduced back in 1995.

# Chapter 6

# Project Contribution

## 6.1 Overview

The project is all about making an android application by which an automatic traffic detection system can be maintained. The traffic jam of Dhaka is very complicated and it is a new complex challenge in terms of automated traffic detection. This problem can be solved using Artificial Intelligence based technology. The detection of an object is associated with computer vision. It describes a system that can detect the location of a desired object in an image.

## 6.2 Proposed System

The project is an automatic traffic detection system and it can use in various sector in traffic system. This system can detect the specific vehicle from images or videos or live camera. The system only can detects the vehicles perfectly because of the strong dataset. Besides, the dataset collection is remained continued. On the upper part of this paper, it is mentioned that this application-based system is beneficial for maintaining many systems such as managing traffic, controlling traffic and a better parking management system.

As the system can detect specific vehicles from live video, it can use to find the maximum number of specific vehicles on a particular road and a restriction can make on those specific vehicles so that the traffic jam can reduce.

## 6.3 User Interface

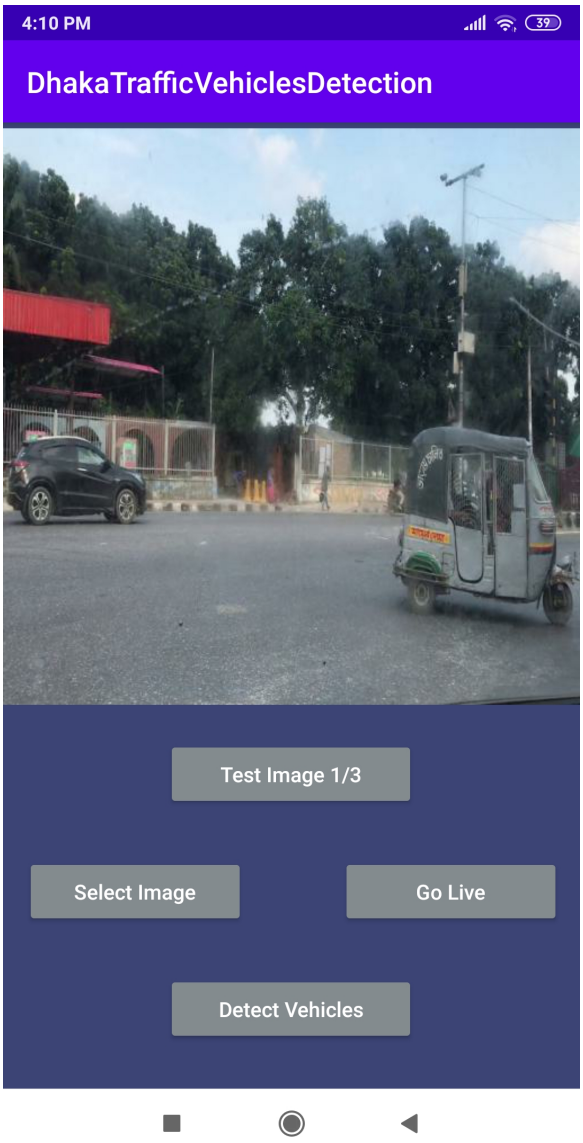


Figure 6.1: Android App User Interface

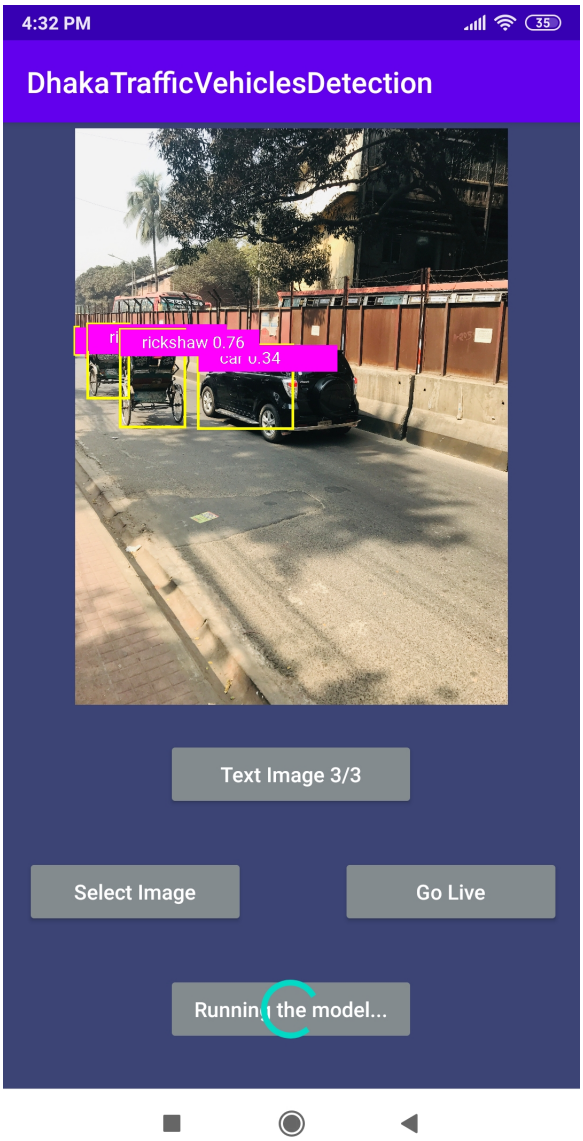


Figure 6.2: Android App User Interface during Model Run

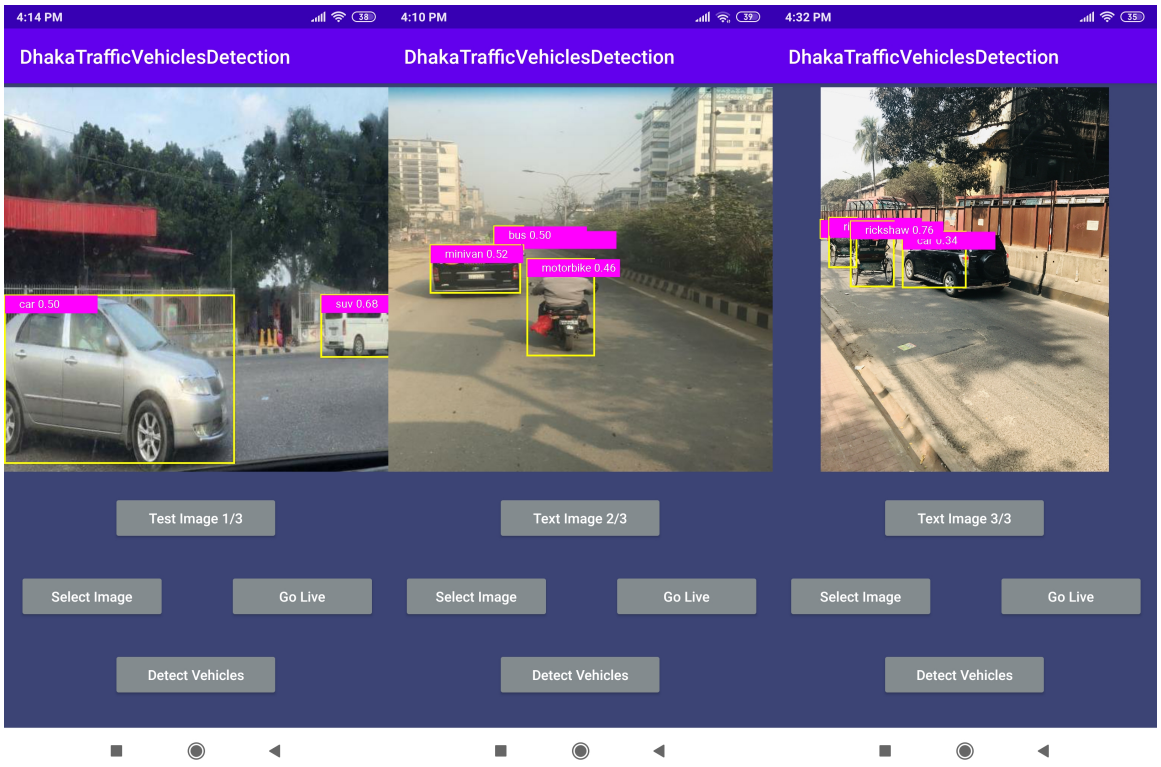


Figure 6.3: Android App User Interface after Model Run on Three Test Image

## 6.4 Results

### 6.4.1 Deep Learning Model Results

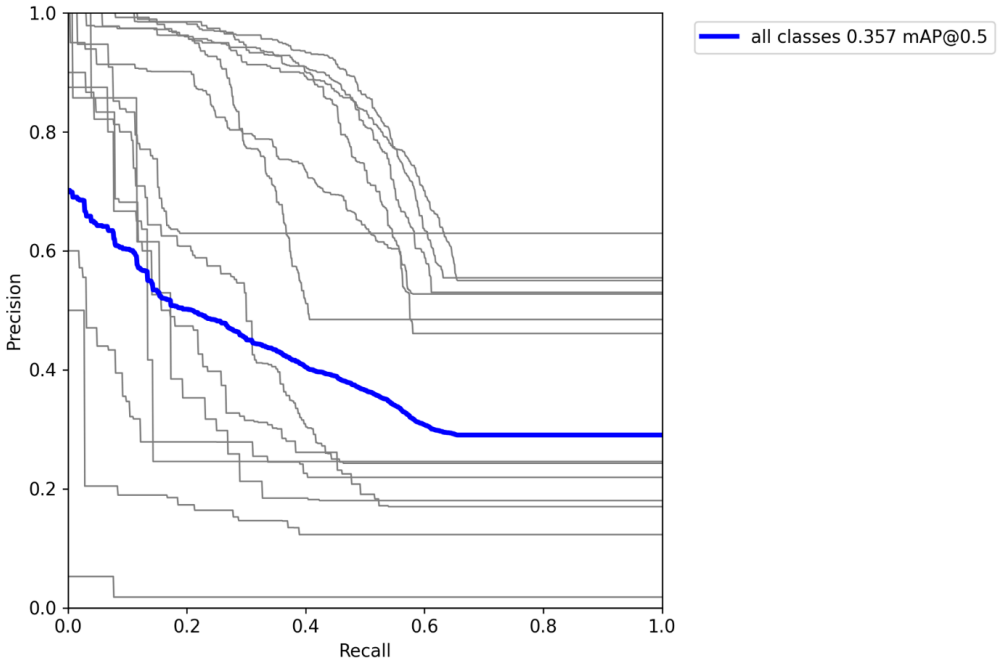


Figure 6.4: Recall vs Precision graph (Best Possible Recall (BPR) = 0.9977)

Recall is proportion of correct positive classifications (True Positive) from cases that are actually positive. Precision is proportion of correct positive classifications (True Positive) from cases that are predicted as positive.

We can calculate the value of these two by applying the methods which are given bellow:
**Recall = TP/ (TP+FN)**
**Precision = TP/ (TP+FP)**

Here TP means True Positives, FN means False Negative and FP means False Positive. With the help of Precision-Recall graph, we can see the balance of the classes. Result shown in **Figure 6.4**

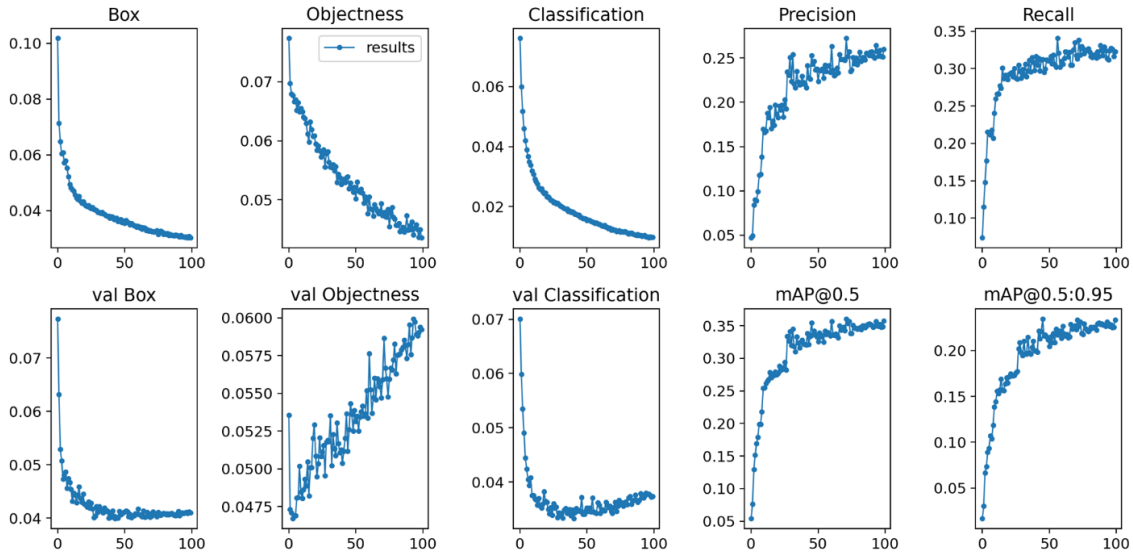


Figure 6.5: Training and Validation Results

In **Figure 6.5**, The first graph shows the box loss, the second graph shows the objectness loss, the third graph shows the classification loss. In our training the losses continue to decreasing and the precision and recall continues to increasing for the training set. The validation box and validation classification describe the losses on validation set and validation objectness describes the gaining on validation set and mean average precision on 0.5 confidence continues to increasing and mean average precision on 0.5 to 0.95 continues to increasing.

Intersection Over Union(IoU) is set to 0.5, the Average Precision of all pictures of each category is calculated, and then all categories are averaged, namely mAP (Mean Average Precision).

## 6.5 Application Outputs

Our Application is an Android-Based Application which can detect different categories of vehicles and recognize it from the videos or images. It is beneficial for maintaining the traffic jam of Bangladesh in future, so there needs of less traffic police for controlling the traffic system in Bangladesh.

# Chapter 7

# Result and Analysis

## 7.1 Deep Learning Model Performance Analysis

Deep learning model performance analysis is a sector where the comparison between two versions of YOLO algorithm has discussed. The comparisons are given below:

**Comparing between YOLOv4 and YOLOv5 Training Configurations**
YOLOv4 and YOLOv5 both specify training configuration files, .cfg and .yaml, respectively. Both of these training files require us to specify num-classes for your custom dataset, as the networks architectures are rewritten. In the Darknet config, we specify more about the training job including batch-size, subdivisions, max-batches and steps. Here's a peek at YOLO layers in each of the configuration files [9] in **Figure 7.1**:

```
                from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Focus                     [3, 32, 3]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     19904  models.common.BottleneckCSP             [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  1    161152  models.common.BottleneckCSP             [128, 128, 3]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  1    641792  models.common.BottleneckCSP             [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1    656896  models.common.SPP                       [512, 512, [5, 9, 13]]
  9                -1  1   1248768  models.common.BottleneckCSP             [512, 512, 1, False]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    378624  models.common.BottleneckCSP             [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     95104  models.common.BottleneckCSP             [256, 128, 1, False]
 18                -1  1    147712  models.common.Conv                      [128, 128, 3, 2]
 19          [-1, 14]  1         0  models.common.Concat                    [1]
 20                -1  1    313088  models.common.BottleneckCSP             [256, 256, 1, False]
 21                -1  1    590336  models.common.Conv                      [256, 256, 3, 2]
 22          [-1, 10]  1         0  models.common.Concat                    [1]
 23                -1  1   1248768  models.common.BottleneckCSP             [512, 512, 1, False]
 24      [17, 20, 23]  1     70122  models.yolo.Detect                      [21, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 5
9, 119], [116, 90, 156, 198, 373, 326]], [128, 256, 512]]
Model Summary: 283 layers, 7309034 parameters, 7309034 gradients

Transferred 362/370 items from yolov5s.pt
```

Figure 7.1: YOLO Layer in Ultralytics.yaml

**Comparing between YOLOv4 and YOLOv5 Training Time**

In YOLOv4 Darknet, training length was set based on number of iterations max-batches (not epochs).[9] It takes 16 hours in training the custom dataset. Maximum validation evaluation was at 2500 iteration. It took 3.5 hours to reach at maximum validation evaluation.

In the YOLOv5s, first 100 epochs was completed in 1.475 hours and second 100 epochs was completed in 0.944 hours. The training time comparison of YOLOv4 and YOLOv5 is shown below:

| Training Time (minute) | YOLOv4 | YOLOv5s |
|---|---|---|
| t1= 0 | 0 | 0 |
| t2=50 | 50 | 50 |
| t3=100 | 100 | 100 |
| t4=150 | 150 | 145.14 |
| t5=200 | 200 | |
| t6= 250 | 210 | |



Figure 7.2: Training Time Comparison of YOLOv4 and YOLOv5

## 7.2 Security Analysis

### 7.2.1 Privacy

Data privacy/confidentiality means protection of data from unauthorized disclosure. For our project, we have collected our data, made a dataset, annotated them and trained them into "Kaggle Notebook" by uploading. But at the time of training our dataset, we made our dataset private, so that we could maintain the confidentiality of our dataset. So there is no breaking of privacy of our data. We have successfully maintained the privacy of our dataset.

### 7.2.2 Integrity

It assurance that data received is as sent by an authorized entity. When the integrity of data is secure, the information stored in a database will remain complete, accurate, and reliable no matter how long it's stored or how often it's accessed. Data integrity also ensures that the data is safe from any outside forces.

There are many ways to reduce data integrity risk. Among them one of the way is to "Follow a Software Development Lifecycle" where few kinds of software process model are remained such as:
• Linear Sequential Model/Waterfall Model
• V-Model
• Spiral Model
• Iterative Model

We have used "V-Model" for our project. In V-Model, which is also known as verification and validation model, there are different types of testing phase. Coding is the construction phase. Though it is known as verification and validation model, but the meaning of verification and validation is different. For verification, we need the development team, the team leader, the testers, the coders. But the validation is done by the client or the users.

### 7.2.3 Security

Security is about protection of assets. As our final outcome of our project is an android app, we can mention about "Mobile Security and Smartphone Vulnerability Threats" here.

First of all we need to know that whether the mobile apps are designed for the purpose of productivity and convenience? Or whether only for security and risk minimization? The answer is basically productivity and convenience. But among all these issues, we need to try our best so that there remains low risk of security breaches.

Software security is the application of techniques that assess, mitigate, and protect software systems from vulnerabilities. These techniques ensure that software continues to function and are safe from attacks.

For our project we have used Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) tools. SAST basically analyzed the source code of our application before it compiles. DAST tools communicate with the running application to identify security vulnerabilities. By using these tools, we have tried to control the security breaches.

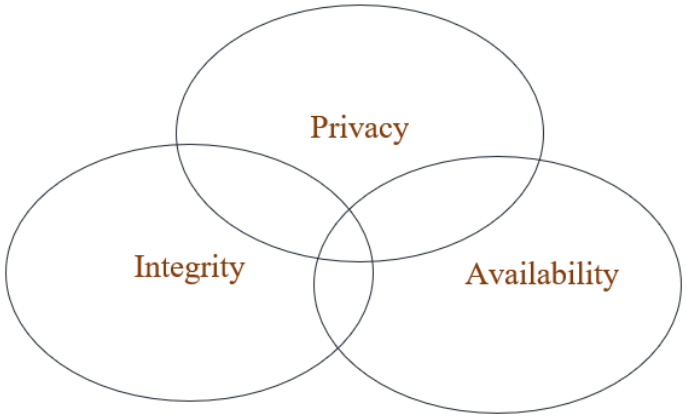


Figure 7.3: Security Goals

## 7.3 Evaluation

### 7.3.1 Interference Time

In YOLOv5 model, it takes 0.27 seconds to 0.32 seconds to detect vehicles from dataset images.

### 7.3.2 Application Performance (Time Complexity)

The android app takes 2.8 seconds to open in an Android 11.0 Supported Smartphone. From the application it takes 1.7 seconds to 2.4 seconds to detect vehicles from test images.

## 7.4 Cost Analysis

About 60 percent of the cost of software is development costs and 40 percent is testing costs. In custom software, the cost of inventory often exceeds the cost of development.

**Tangible Costs** are those that can be accurately projected by the systems analysts and the business accounting personnel.[10] **Intangible costs** are those that are difficult to estimate.[10] For this project, there is no difficulty for the estimation so there is no intangible costs.

Table 7.1 show the cost analysis of this project:

Table 7.1: Cost Analysis

| Types of Cost | Amount |
|---|---|
| **1. Hardware Cost** | |
| (i) Acer/Dell/HP Laptop (GPU Included) | 2,00,000 BDT |
| (ii) Android Smartphone with Minimum Android 11.0 Supported | 17,000 BDT |
| (iii) Closed-Circuit Television(CCTV) Camera | 1,00,000 BDT |
| **2. Software Cost** | |
| (i) Anaconda Prompt | Free |
| (ii) Python | Free |
| (iii) Android Studio | Free |
| (iv) Git Bash | Free |
| (v) Kaggle Notebook | Free |
| (vi) Adobe XD | Free |
| **3. Tangible Costs** | |
| (i) Cost of Resources | Free |
| (ii) Cost of a Single Programmer | 2,00,000 BDT |
| **4. Intangible Costs** | 0 BDT |
| **Total Amount** | **5,17,000 BDT** |

## 7.5 Social, Legal and Ethical Issues

**Social Issues:** This section can determine the social issues we have faced. For our dataset, while capturing pictures of vehicles from different places and roadside, we faced some problems such as privacy breaking of general people. And taking pictures without one's consent will be punishable.[11] The general people had no idea that the pictures were captured for project purpose only. And the pictures were only of the vehicles, not the human. So sometimes it was difficult to capture pictures because of this social issue and while using the app, the same problem was being faced. But this case was not very much concerned for us as it was for project purpose only. The student ID help a lot.

**Legal Issues:** The legal issue was very much higher for us. In our dataset, there is a class called "Army Vehicle". And to capture the pictures of army vehicle, we needed to take permission. Without notifying, taking pictures of these kind of vehicles are totally illegal. It would be a huge problem for us if we did not notify them as well as take permission from them. We also took videos from roadside in different angles to avoid this issue. As we took permission and also made videos to avoid this issue, there remained no illegal issues for us.

**Ethical Issues:** To maintain ethical issues, ACM code of ethics were being followed. In ACM code of ethics, there is a principle which is under section **01** and known as "PUBLIC". Where we can find many sub-sections. Among those, some sub-sections are:
"**1.01:** *Accept full responsibility for their own work.*" [12]
"**1.03:** *Approve software only if they have a well-founded belief that it is safe, meets specifications, passes appropriate tests, and does not diminish quality of life, diminish privacy or harm the environment. The ultimate effect of the work should be to the public good.*" [12]
Another principle which is under section **03** and known as "PRODUCT" and some of its sub-sections are:
"**3.01:** *Strive for high quality, acceptable cost and a reasonable schedule, ensuring significant tradeoffs are clear to and accepted by the employer and the client, and are available for consideration by the user and the public.*" [12]
"**3.02:** *Ensure proper and achievable goals and objectives for any project on which they*

*work or propose.*" [12]

These codes of ethics were strictly followed in our project. So, no ethical issues will occur in our project.

# Chapter 8

# Conclusion and Future Works

## 8.1 Scope for Future Work

We have built an android app for our project. In this app we have done the work to detect the different categories of vehicles. And we have some plans for our future development such as, count the numbers of a specific vehicles, shows the exact location of a "Traffic Image" which is selected or captured through the phone, automatically uploading system of the detected image into the cloud server. Until now, we have planned for these future implementations.

## 8.2 Conclusion

One of the biggest issues of Dhaka city is traffic jam. This problem can solve in many ways and among those ways, one of the ways is to detect vehicles. As the system can detect specific vehicles from images or live video, it can use to find the maximum number of specific vehicles on a particular road and a restriction can make on those specific vehicles so that the traffic jam can reduce. Though it won't reduce much, but an amount of reduce traffic jam can save our time. That is the main concern of this project.

## 8.3 Gantt Chart

Table 8.1 show the tasks schedule of this project:

Table 8.1: Gantt Chart Tasks Schedule

| Tasks | Start Date | End Date | Duration(Days) |
|---|---|---|---|
| **1. Planning** | | | |
| 1.1. Project Proposal | 7-Nov | 9-Nov | 2 |
| 1.2. Project Summary | 10-Nov | 12-Nov | 2 |
| 1.3. Collect Picture for Annotation | 13-Nov | 28-Nov | 15 |
| **2. Research** | | | |
| 2.1. LabelImg | 29-Nov | 30-Nov | 1 |
| 2.2. Yolov5 | 1-Dec | 3-Dec | 2 |
| 2.3. Dataset | 4-Dec | 9-Dec | 5 |
| **3. Implementation (Part - I)** | | | |
| 3.1. Model Train | 10-Dec | 9-Jan | 30 |
| **4. Design** | | | |
| 4.1. Android Application Design | 10-Jan | 17-Jan | 7 |
| **5. Planning for Adding More Images** | | | |
| 5.1. Collect Picture for Annotation | 28-Jan | 10-Feb | 13 |
| **6. Implementation (Part - II)** | | | |
| 6.1. Model Export for Android App | 11-Feb | 24-Feb | 13 |
| 6.2. Android App Development | 25-Feb | 16-Apr | 50 |
| **7. Testing Phase** | | | |
| 7.1. Android Application Testing | 17-Apr | 24-Apr | 7 |
| **8. Follow-up** | | | |
| 8.1. Final Report Writing | 25-Apr | 7-Jun | 43 |
| 8.2. Presentation and Final Defense | 8-Jun | 12-Jun | 4 |

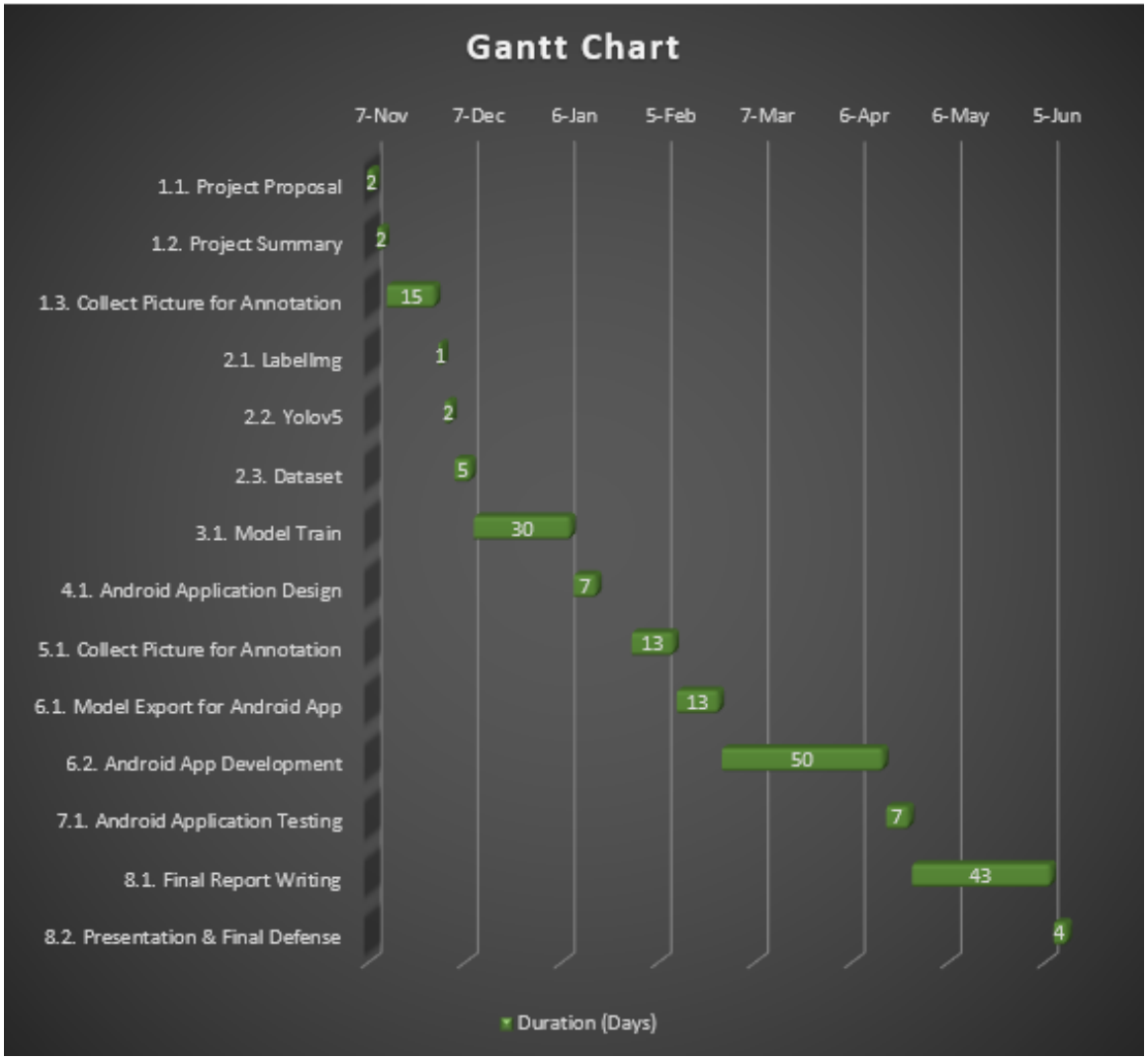


Figure 8.1: Gantt Chart

# Bibliography


[1] Gomathy Nayagam Meenakshi Sundaram. A survey on real time object detection and tracking algorithms. *International Journal of Applied Engineering Research*, 10:8290–8297, 04 2015.

[2] Huansheng Song, Haoxiang Liang, Huaiyu Li, Zhe Dai, and Xu Yun. Vision-based vehicle detection and counting system using deep learning in highway scenes. *European Transport Research Review*, 11(1):1–16, 2019.

[3] Deeksha Gour and Amit Kanskar. Automated ai based road traffic accident alert system: Yolo algorithm. *Int. J. Sci. Technol. Res*, 8(8), 2019.

[4] Jason Brownlee. Difference between a batch and an epoch in a neural network, 2021.

[5] Yolo algorithm and yolo object detection: An introduction - appsilon: End¬ to¬ end data science solutions, Mar 2021.

[6] Jacob Solawetz. Yolov5 new version - improvements and evaluation, Mar 2021.

[7] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020.

[8] Software engineering: Sdlc v-model, May 2019.

[9] Joseph Nelson. Responding to the controversy about yolov5, Mar 2021.

[10] Eco. Identifying benefits and costs, Jan 1970.

[11] The Daily Sun. Taking pictures without one's consent will be punishable, Aug 2016.

[12] Software engineering code, Dec 2018.