Mask and Helmet Detection in Two-wheelers using YOLOv3 and Canny Edge Detection

Submitted in partial fulfilment of the requirements

of the degree of

B. E. Information Technology

By

Jayesh Rane 06

Shravani Maliye 36

Jayom Oza 48

Supervisor:

Prof. Nileema Pathak

Assistant Professor

Department of Information Technology

Atharva College of Engineering

Malad West Mumbai

University of Mumbai

2020-2021

CERTIFICATE

This is to certify that the project entitled **“Mask and Helmet Detection in Two-wheelers using YOLOv3 and Canny Edge Detection”** is a bonafide work of **“Jayesh Rane (06), Shravani Maliye (36) and Jayom Oza (48)”** submitted to the University of Mumbai in partial fulfilment of the requirement for the award of the degree of B.E. in Information Technology.

Prof. Nileema Pathak

Supervisor/ Guide

Prof. Deepali Maste	**Dr. S. P. Kallurkar**
Head of Department	**Principal**

Project Report Approval for B.E.

This project report entitled ***"Mask and Helmet Detection in Two-wheelers using YOLOv3 and Canny Edge Detection"*** by ***Jayesh Rane, Shravani Maliye and Jayom Oza*** is approved for the degree of ***B.E. in Information Technology.***

Examiners

1.--

2.--

Date:

Place:

Declaration

I declare that this written submission represents my ideas in my own words and where others' ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Jayesh Rane

Shravani Maliye

Jayom Oza

Date:

Abstract

There is no automated existing system which can detect motorcyclists who are not wearing helmet or face mask or both due to which the traffic policemen have to manually keep records of such traffic rules violators either by remembering the number plate or by capturing a photo of the number plate. This manual administration can sometimes lead to errors. In order to overcome these drawbacks, we have designed an automated face mask and helmet detection system which is able to catch all the motorcyclists who are not wearing helmet or mask or both just by storing the number plate of those bike riders. The "Face Mask Helmet Detection System" is divided into four modules. It begins by classifying the Two Wheelers from an image consisting of four wheelers and other vehicles. Once the two wheelers are classified by the system then it checks for the presence of a helmet on the bike rider. In third module it checks if rider is wearing mask or not. If the bike rider is found without a helmet or mask or the rider is not wearing both, then it extracts the number plate from the image and stores it in the database. The system works properly even when the number plate is horizontally not straight as the bike rider may not be driving in a straight direction. This system will help to catch traffic violators thereby creating more strictness in traffic and thus reducing risk of injuries.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Sr. No.	Abbreviation	Expanded Form
i	YOLOv3	You Only Look Once version 3
ii	OIDv4	Open Image Dataset version 4
iii	CNN	Convolutional Neural Network
iv	WHO	World Health Organisation
v	GPU	Graphics Processing Unit
vi	CUDA	Compute Unified Device Architecture

CHAPTER 1

INTRODUCTION

1.Introduction

Two-wheeler vehicles such as scooters and motorcycles are the most popular type of vehicles in India because of their availability and convenience. They are owned by a major part of the population in the country according to the statistics provided by the Department of Statistics and Implementation Systems (MSPI), the Government of India. As two-wheelers become more and more popular due to the ease of travel, the number of road accidents involving two-wheelers have increased. According to a report released by the Department of Transport and Highways, more than 37% of people die in road accidents (56,136) or six every hour on an average - including two wheelers. More crashes and deaths are caused by faulty licensing rules, no training, poor roads and unsafe protective helmets. The use of a proper helmet can reduce the risk of injury by 42% and head injuries by 69%, according to a World Health Organization report. It is highly recommended for two-wheeler riders to use a safety helmet to reduce the risk of injury. Most of the deaths in the last few years in accidents are due to head injuries. It is a criminal offense to ride a two-wheeler without a helmet and many hand-to-hand tactics have been adopted to catch violators because of their importance. Still a large number of bike riders do not follow the law. The automation of this process in real time is a need of the hour and will help to accurately monitor passengers who break the rules, and greatly reduce the number of human interventions. To perform the face-to-face identification process, we need to find two wheelers, a mask, a helmet and a number plate.

1.1 Description

In the existing system, the police officer has to manually capture the image of the number plate of the bike rider who is not wearing the helmet and the face mask. A very big disadvantage of this method is that often such rule-violators speed up and run off and are not penalized for their actions. To overcome the disadvantages of the existing system, we have proposed an automated system which is more accurate and requires minimum human efforts. The main application of this system is to catch all the motorcyclists who are not wearing helmets and face masks. Now let us understand the working of the system. The system will divide the traffic video into various images(frames) and from those images the system is able to detect bike riders from an image consisting of both two wheelers and other vehicles using YOLOv3. Once the bike riders are detected, the system is able to detect if the motorcyclist is wearing a helmet or not using YOLOv3, also the system is able to detect if the motorcyclist is wearing a mask or not using

YOLOv3. The system will then record the number plate of those bike riders who are not wearing helmet or face mask or both using the Canny Edge Detection Method.

1.2 Problem Formulation

Today, road traffic safety is one of the prime concerns of any nation. Our country, India being one such nation has to ensure that the citizens abide by road safety rules. India has an estimated 3.7 crores population with motorcycles/bikes, which is the largest number in the world. In the existing system, it becomes a tedious task to regulate traffic violations without a helmet or face mask by the bike riders, by the traffic police. The police officer has to manually capture the image of the number plate of the bike rider who is not wearing the helmet or face mask. A very big disadvantage of this method is that often such rule-breakers speed up and run off and are not penalized for such violations. Another aspect is that the existing traffic cameras are used only to check for parking and signal related violations. The main aim is to automate this task, so we can avoid any human errors and execute it efficiently. The Helmet Detection System aims to work on the traffic camera feed and introduce this new added feature.

1.3 Motivation

The road accident is one of the major problems all over the world. The recent report says that the annual average road accidents is estimated to be about 7,00,000 of which 10 percentage occur in India which has overtaken China. The annual statistics revealed by the World Health Organization (WHO) in its Global status report on road safety says that around 80,000 people are killed on Indian roads due to rush driving, drunken driving and less usage of helmets. Also, most of the countries are forcing the motor riders to wear the helmet and not to use the vehicles when the person is in drunken condition. Also, during this pandemic situation face mask is also necessary for all the bike riders has it provides safety towards the viruses. To overcome problem, a system called Face Mask and Helmet Detection using Machine learning is introduced

1.4 Proposed Solution

The Helmet and face mask Detection System has been designed to create an automated system which could detect the bike riders who are travelling without helmet and the face mask. The system will be useful for traffic policemen to catch bike riders who are not wearing a helmet and face mask. The system will divide the traffic video into various images(frames) and from those images the system is able to detect bike riders from an image consisting of both two wheelers and other vehicles using YOLOv3. Once the bike riders are detected, the system is able to detect if the motorcyclist is wearing a face mask or not using YOLOv3, also the system is able to detect if the motorcyclist is wearing a helmet or not using YOLOv3. The system will then record the number plate of those bike riders who are not wearing helmet or face mask or both using Canny Edge Detection Method. With the help of this system, proper emphasis is being made on the safety rules and regulations that have to be followed in order to reduce the number of accidents that occur very frequently with two wheelers. Also, it reduces the burden of manual work and the man power can then be used in the places where it is absolutely necessary.

1.5 Scope of the Project

The main application of the system is to catch all the motorcyclists who are not wearing helmet or face mask or both. This system can be used by almost any country provided that they provide the dataset in an appropriate format and train the model efficiently. Apart from this application this system can be used for other applications as well. Since this system is able to detect and classify two wheelers and four wheelers this system can be used for vehicle classification i.e it can be used to detect whether a vehicle is a car, a truck, a bike etc. By doing this vehicle classification the system will be able to keep track of the information about vehicles which can be used by the government for planning purposes. Since this system is able to detect the helmet on bike riders so it can be used in various other places where the helmet is a very important part of safety like construction sites, power plants, etc. In construction sites, the helmet is very important for people working there to protect their head from any injury. In such a case this system is useful to ensure each and every worker working near the construction site is wearing a helmet to avoid any fatal injury. As this system has a number plate recognition system in it so it can be used in various other places where the number plate is to be recorded for example in case of robbery, accidents etc.

CHAPTER 2

REVIEW OF LITERATURE

2. Review of Literature

Paper No.	Year of Publication	Journal Name	Paper Details
[1]	2020	IEEE	The system uses You Only Look Once (YOLO)-Darknet deep learning framework which consists of Convolutional Neural Networks trained on Common Objects in Context (COCO) and combined with computer vision. YOLO's convolutional layers are modified to detect specified three classes and it uses a sliding-window process. The map (Mean Average Precision) on validation dataset achieved 81% by using training data.
[2]	2019	IOSRJEN	In this system the motorbikes are detected using the feature vector HOG. Once the motorbike is detected, by means of convolutional neural network, it is determined whether the motorcyclist is wearing a helmet or not. If the motorcyclist is identified without helmet, then the license plate of the motorcycle is detected using tesseract OCR.
[3]	2019	IJRTE	In this paper authors use R-CNN, Fast R-CNN and Faster RCNN for object detection and compare all three algorithms and also explained the YOLOv2 algorithm and why it is better from traditional approach. They concluded that FastRCNN have mean average precision up to 76.4 but it has between 5 to 18 frames per second(fps). On the other hand, YOLOv2 algorithm has mean average precision up to 78.6, and can attained speed up to 155 frame per second(fps). YOLOv2 attains an outstanding trade-off between accuracy and speed and also as a detector possessing powerful generalization capabilities of representing an entire image.

[4]	2019	IEEE	In this paper the authors intend to make an automated system to distinguish whether a biker is wearing a helmet or not and to impose fine to defaulters as a part of law enforcement. For execution of the following idea, they have used Caffe model and InceptionV3. According to our research, such a system is not currently used by the police or by any other authority. Executing the proposed framework can convey more mindfulness and need to wear a helmet at any rate with the goal that they don't get captured on camera and avoid fine. This project intends to make deep learning based automated detection system for helmet identification using trained models and datasets that would be useful for the police department to enforce the law for the betterment of the society.
[5]	2020	IEEE	The COVID-19 epidemic caused by the novel coronavirus continues to spread worldwide. The impact of COVID-19 has fallen on almost all development sectors. The health care system is in trouble. Many precautionary measures have been taken to reduce the spread of the disease when wearing a mask is one of them. In this paper, they propose a program that finds naked people who are not wearing face mask on a smart city network where all public areas are monitored by Closed-Circuit Television (CCTV) cameras. When a person without a mask is found, the corresponding officer is informed. Deep architecture is trained in a database that contains images of people with and without masks collected from various sources. Professional facilities have found 98.7% accuracy in distinguishing people with face masks for previously unseen test data.

[6]	2020	IEEE	The proposed method detects the face from the image correctly and then identifies if it has a mask on it or not. As a surveillance task performer, it can also detect a face along with a mask in motion. The method attains accuracy up to 95.77% and 94.58% respectively on two different datasets. We explore optimized values of parameters using the Sequential Convolutional Neural Network model to detect the presence of masks correctly without causing over-fitting.
[7]	2015	IEEE	The main objective of the proposed segmentation-based system includes significant reduction in latency, better edge preservation. The simulation of proposed system with MATLAB version R2012a is carried out. The experimental result indicates the effect of mask size on the latency and effect of PSNR on quality of edges which quantify the overall system performance.
[8]	2020	IEEE	In this paper, a transfer learning model is proposed to automate the process of identifying the people who are not wearing mask. The proposed model is built by fine-tuning the pre-trained state-of-the-art deep learning model, InceptionV3. The proposed model is trained and tested on the Simulated Masked Face Dataset (SMFD). Image augmentation technique is adopted to address the limited availability of data for better training and testing of the model. The model outperformed the other recently proposed approaches by achieving an accuracy of 99.9% during training and 100% during testing.
[9]	2020	IEEE	The approach has been tested on a set of 1211 facial images extracted from group of people wearing or not wearing a mask, by considering a 2 - class problem, where the mask class represents the positive examples, where the non-masked faces are negative examples. Part of the image data set is used to train the support vector

			machines for learning discriminant features for each class, followed by a prediction for each test sample. The image set for the mask class ranges from simple and common one-colored surgical masks to complex and challenging patterned masks. Cross-validation approach is adopted to test the approach, leading to 97.25 % as recognition rate.
[10]	2020	IEEE	They have used datasets to develop approaches are limited in terms of traffic environments and traffic density variations. In this paper, we propose a CNN-based multi-task learning (MTL) method for identifying and tracking individual motorcycles, and register rider specific helmet use. They further release the HELMET dataset, which includes 91,000 annotated frames of 10,006 individual motorcycles from 12 observation sites in Myanmar. Along with the dataset, we introduce an evaluation metric for helmet use and rider detection accuracy, which can be used as a benchmark for evaluating future detection approaches. They have shown that the use of MTL for concurrent visual similarity learning and helmet use classification improves the efficiency of our approach compared to earlier studies, allowing a processing speed of more than 8 FPS on consumer hardware, and a weighted average F-measure of 67.3% for detecting the number of riders and helmet use of tracked motorcycles.

CHAPTER 3

SYSTEM ANALYSIS

3. System Analysis

3.1 Functional Requirements

3.1.1 Critical Requirements

1. The system should be able to differentiate between different objects in the image such as a bike, helmet, number plate, mask.

2. The system should detect the number plate of only those bike riders who are violating any rule.

3. The system should be able to detect objects any day and any time.

4. The system should be able to detect objects in low lighting conditions as well.

5. The system should also be able to detect motorcycles from faraway.

6. The system should be able to distinguish between objects which look similar and should be able to correctly distinguish between them.

3.2 Non-Functional Requirements

3.2.1 Performance

The response time for detecting the number plate of traffic rule violations is very fast. At a time, the system can detect multiple traffic rule violators in a single frame with high accuracy.

3.2.2 Usability

The system has been designed in such a way that any person with not so much technical skills can use it, thus providing everyone a good user experience.

3.2.3 Security

The images of the number plate are only used for keeping record of traffic rule violators and will not be shared with anyone.

3.2.4 Capacity

The system can work optimally when the computer has good configurations such as having ram more than 2 GB.

3.2.5 Compatibility

The system has been designed in such a way that it can work on computers having windows platform from Windows 10 onwards.

3.2.6 Network Connectivity

The system is required to be connected to the internet all the time so that it can carry out its operations of object detection.

3.3 Specific Requirements

Table 3.3.1 Hardware

Hardware
Intel core i5 or higher
2 GB DDR4 Memory or more
500 GB HDD or more

Table 3.3.2 Software

Software
Google Colab
Python
Operating System Windows 10

CHAPTER 4

ANALYSIS MODELLING

4. Analysis Modelling

4.1 Entity Relationship Diagram

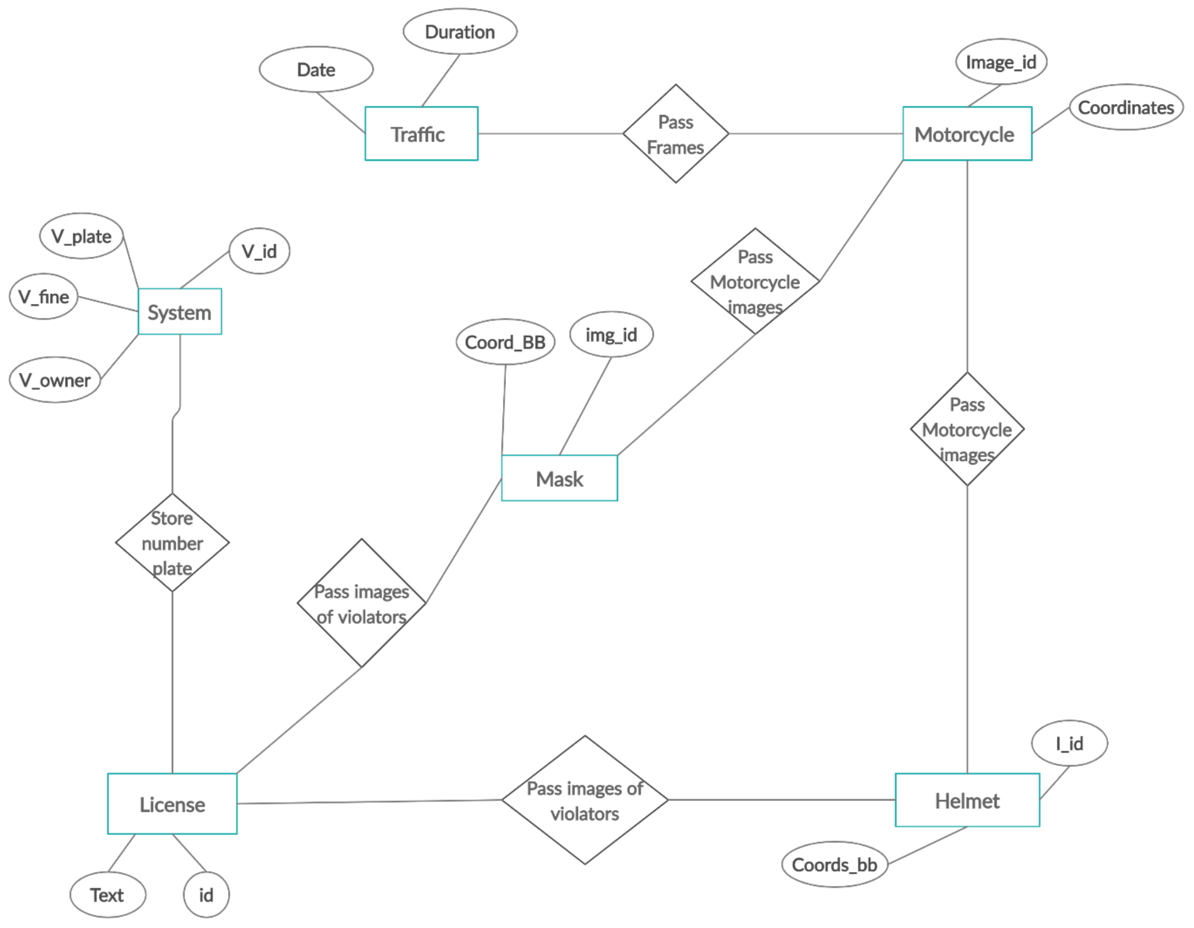


Figure 4.1 Entity Relationship Diagram

The E-R diagram consists of six main entities: Traffic, Motorcycle, Helmet, License System. The Traffic table has the following attributes: Duration and Date. The Motorcycle table consists of the Image_id and Coordinates. The Helmet table consists of I_id and Coords_bb attributes. The Mask Detector table consists of img_id and Coord_BB attributes. The License table has the following attributes: Text and id. Finally, the System has the following attributes: V_id, V_plate, V_fine amd V_owner. The Traffic and Motorcycle tables are related through pass frames relationship. The Motorcycle and Helmet tables are related through pass motorcycle images relationship. The Motorcycle and Mask tables are connected through pass motorcycle images relationship. The Helmet and License tables are connected through pass images of violators relationship. The Mask and License tables are related through pass images of violators relationship. The License and System tables are connected through store number plate relationship.

4.2 Activity Diagram

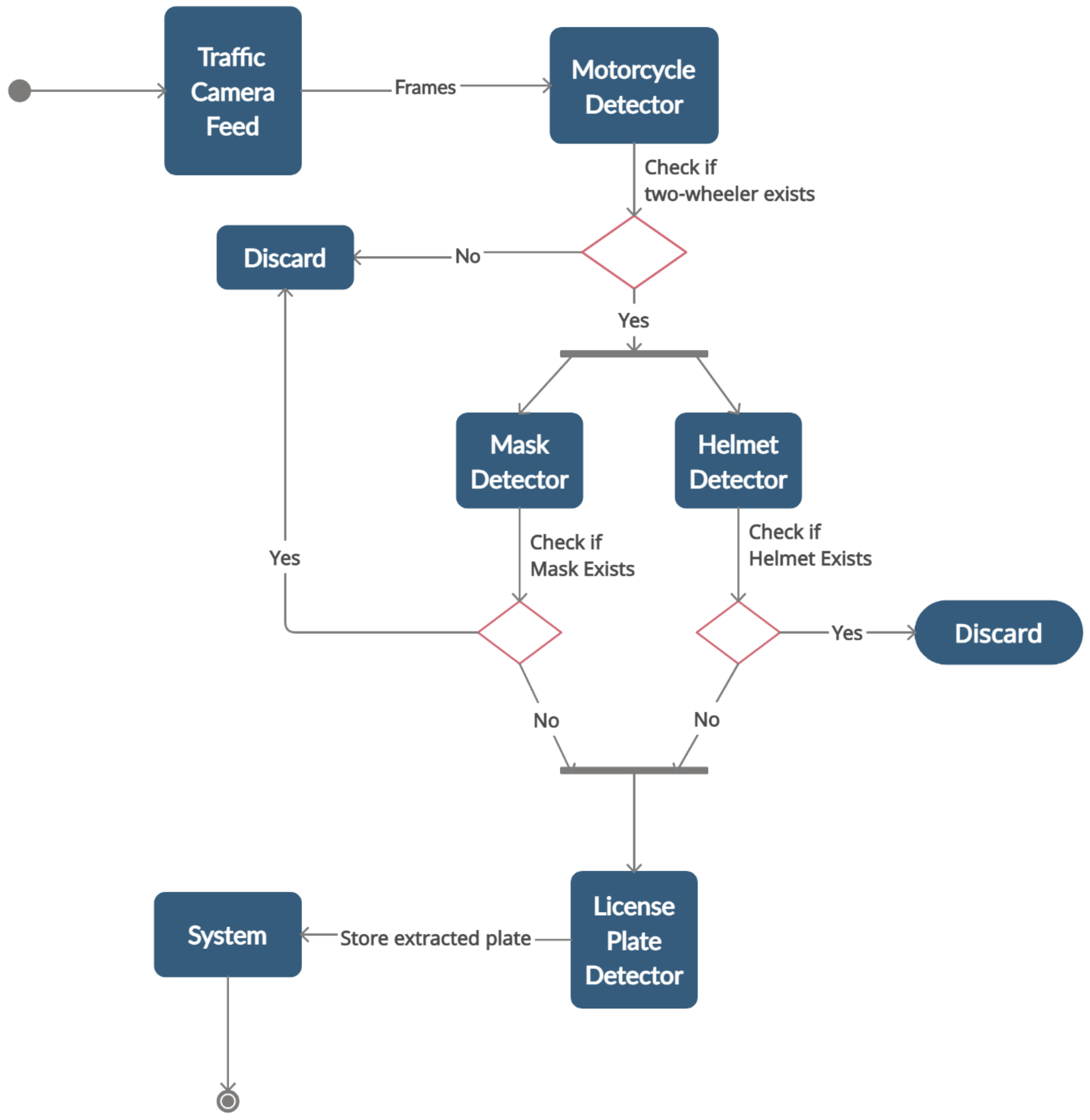


Figure 4.2 Activity Diagram

The above diagram depicts the working of the system in a sequential manner. The frames from the traffic camera feed are sent to the Motorcycle Detector in order to check if a two-wheeler exists in the current frame. If it does, the frame is sent to the Helmet Detector as well as Mask Detector. If not, then the frame is simply discarded. The frame is then checked to see if the rider is wearing a helmet or not- if they are wearing the helmet, then the frame is discarded, else the image of the violator is sent to the License Plate Detector. If the rider does not wear a mask, then the image is sent to the License Plate Detector or else it is discarded. The information of the violating vehicle is then stored in the system for maintaining records for future references.

4.3 Functional Modelling

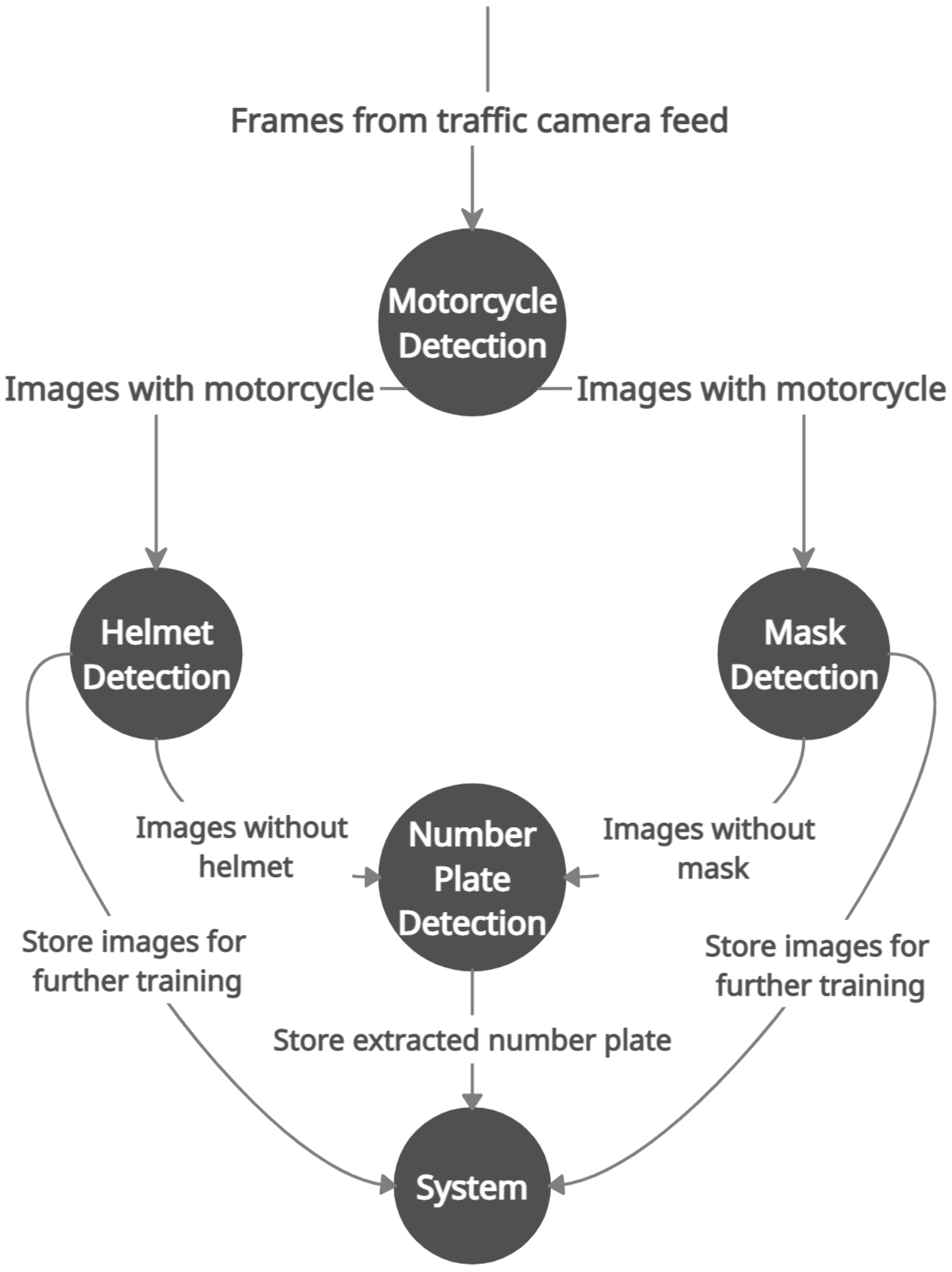


Figure 4.3 Data Flow Diagram

The above figure depicts the data flow diagram for the system. After receiving the appropriate data, the respective modules begin to detect the required information. The images of violators from various modules are stored for two main reasons: first, to increase the training period for the system and second, increase the accuracy provided by the system. The unnecessary images are not stored but are simply removed from the system. Also, the extracted number plates are stored for the purpose of maintaining records.

4.4 TimeLine Chart

A Gantt Chart is used to provide the time taken for a particular activity in a pictorial or graphical format.

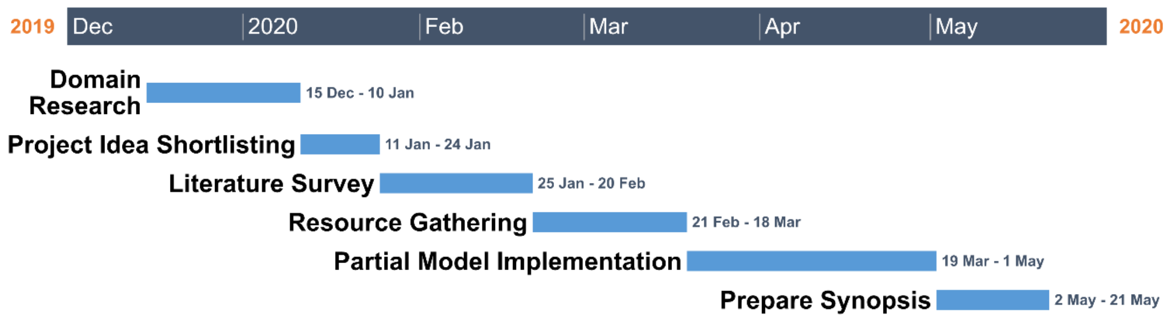


Figure 4.4.1 Gantt Chart for Phase 1

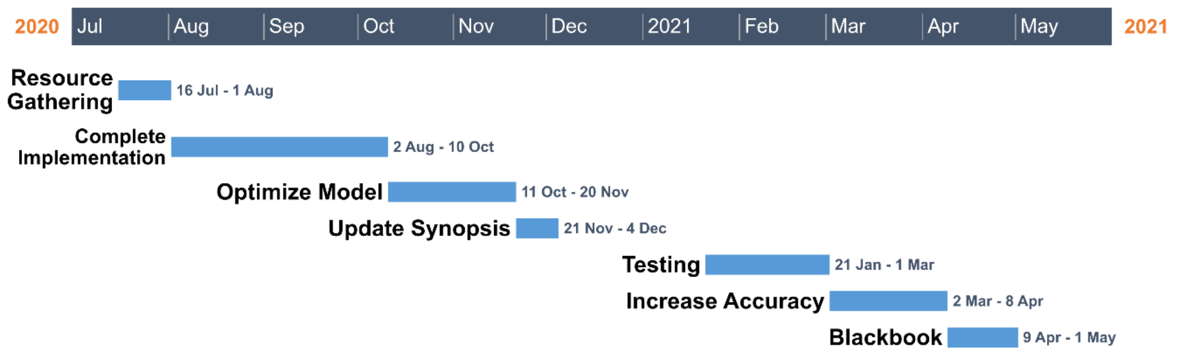


Figure 4.4.2 Gantt Chart for Phase 2

CHAPTER 5

DESIGN

5. Design

5.1 Architectural Design

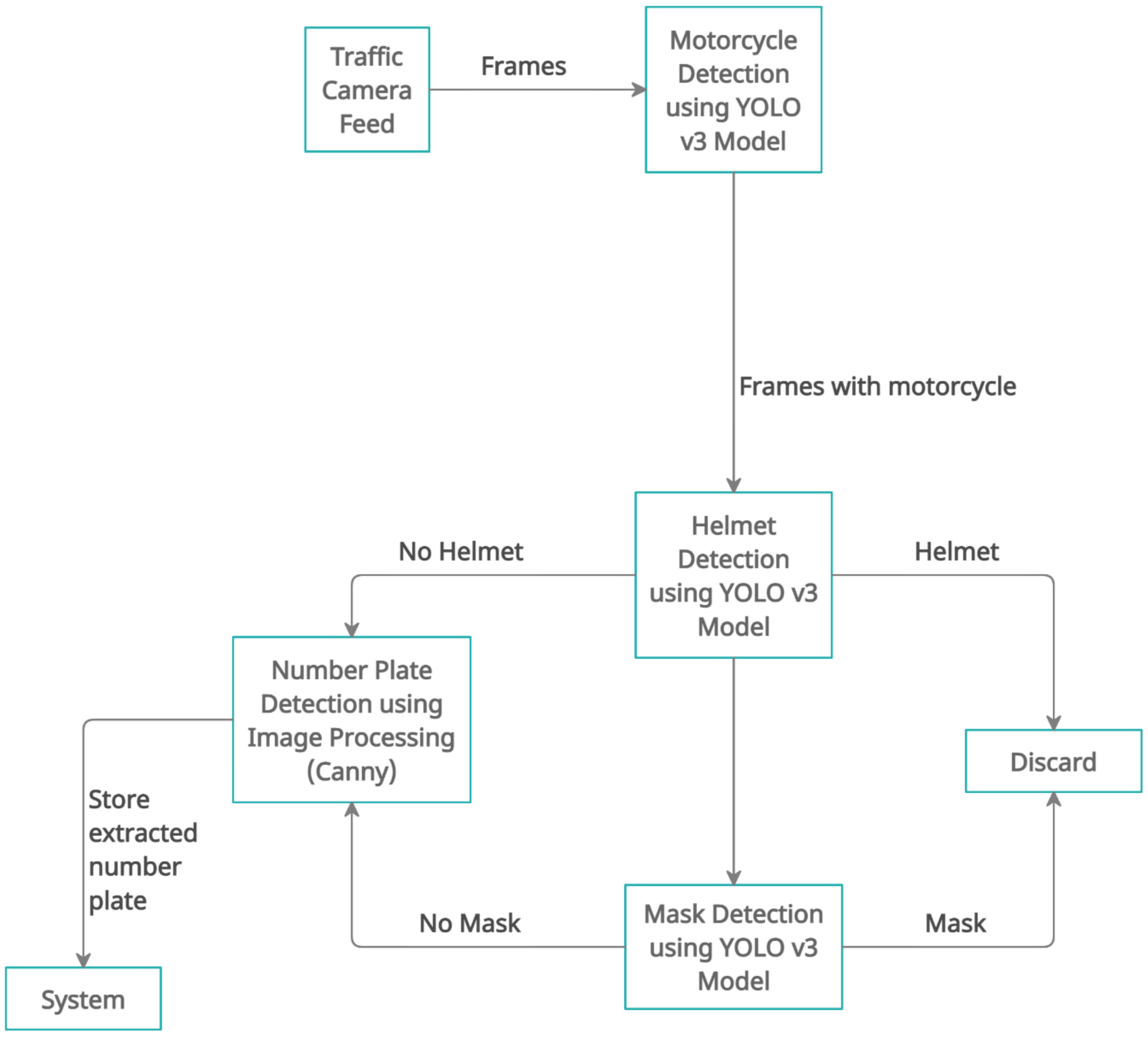

Traffic Camera Feed
Frames
Motorcycle Detection using YOLO v3 Model
Frames with motorcycle
Helmet Detection using YOLO v3 Model
No Helmet
Helmet
Number Plate Detection using Image Processing (Canny)
Discard
Store extracted number plate
No Mask
Mask Detection using YOLO v3 Model
Mask
System

Figure 5.1 Block Diagram

The first thing is to pass the existing traffic camera feed to the system which has been built specifically for the detection. The feed is then divided into frames so that they can be sent to the first detection module- Motorcycle Detection. The frames are passed to the first module which detects two-wheelers. If a two-wheeler exists then the frame is sent to the second detection module- Helmet Detection.

The second detection module detects helmet on the rider’s head. If a helmet is detected, then the frame is discarded else, a violation is found and the same frame is sent to the Number Plate

Detection module, wherein the number plate of the violating vehicle is extracted and simply stored into the system. The same frame is also sent to the third module- Mask Detection.

The third and most important module is the Mask Detection module which is a very important concept to implement some semblance of safety in the current pandemic situation. The frame is scanned and if a mask is detected then it is simply discarded, if not, then a safety violation is found. The frame consisting of the violating vehicle is sent to the final module which is the Number Plate Detection module. The number plate detection module is used to extract the license plate information of the violators so that stricter measures for safety can be imposed and the number of accidents and deaths caused by them can be reduced.

CHAPTER 6

IMPLEMENTATION

6. Implementation

6.1 Algorithms

Object Detection is a concept in Computer Vision which is gaining momentum in recent times. The need of the hour was to develop exemplary object detection algorithms. YOLO is a very efficient object detection algorithm which was developed keeping the requirements in mind. It only needs to look once at the image which in simpler terms means that it only needs a single forward propagation through the neural network to make the appropriate predictions.

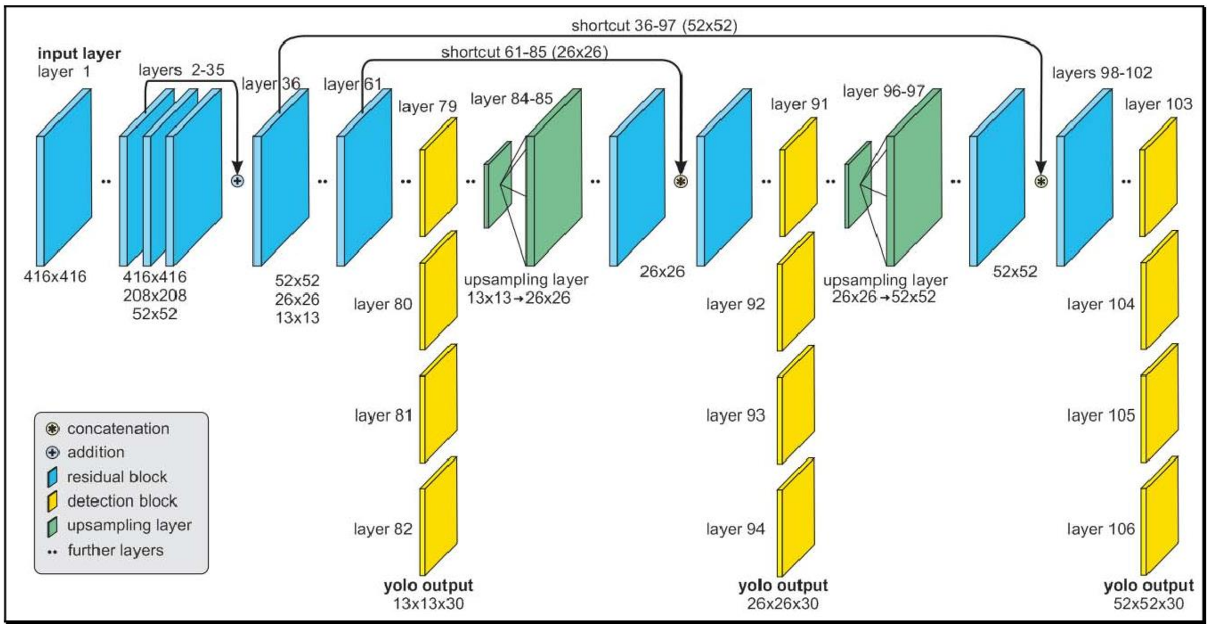


Figure 6.1.1 YOLO architecture

The above figure shows the architecture of a YOLO model. In this model, a single neural network is applied to the concerned image. In this model, the image is divided into multiple parts and probabilities are calculated and depending on their values bounding boxes are mapped. Non max suppression is deployed so that each object is detected only once and the overall accuracy of the model is increased. The output is then provided in the format that it shows the bounding box along with the appropriate class label and the probability calculated for that label.

The version of YOLO used in the proposed system is version 3. YOLOv3 can detect multiple objects at once, predict the appropriate classes and identify the locations. The model applies a single neural network to the image and divides it into a number of grid cells. The probability for each individual cell is then calculated and depending on its value, the boxes are predicted.

There are no pooling layers in this model, instead, they have Convolutional Neural Networks. It prevents the loss of low-level features and hence, is able to detect even small objects accurately.

A Darknet variant of 53 layers is used in this version in such a way that the architecture of this model consists of 106 fully functional convolutional layers. The most astounding features of this version is that it makes predictions at three different scales. The detection happens at the following three layers: 82, 94 and 106. The shape of the detection kernel is calculated using the below formula:

$$b * (5 + c) \tag{1}$$

where b is the number of bounding boxes = 3 and c is the number of classes, which depends on the individual modules.

The essential elements of a YOLO model is: Residual blocks and Skip Connections. In residual blocks, the activation value of a particular layer is forwarded to another deeper layer in the network. In deep learning networks, spik connections basically meaning skipping a few layers and providing the output of one layer to another layer which is not the next one.

The input can be of any size as it is then adjusted according to the network. The bounding boxes are predicted via a number of calculations. Anchors are calculated using k-means clustering. To predict the bounding box, further the offsets to the anchors are calculated in order to remove unstable gradients. The objectness score for each bounding box is calculated using the below formula:

$$P_0 = P_{object} * IoU \tag{2}$$

Where, P_0 is objectness score, P_{object} is the predicted probability and IoU stands for the Intersection over Union values.

The speed of YOLOv3 is slower as compared to the previous reasons as it has 106 layers, but the advantage of this model is that it provides a higher accuracy. The choice of anchor boxes is more and the number of bounding boxes per image is also more.

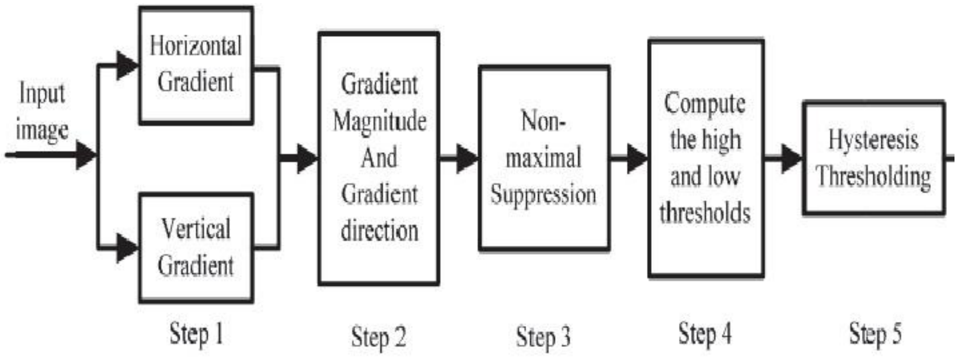


Figure 6.1.2 Canny Edge Detection [7]

Canny Edge Detection model is an algorithm used to detect edges in various images. The working of this model is pretty straightforward. [7] The original image is supplied to the algorithm as an input. This image is then converted to a greyscale format. The image is susceptible to a lot of noise; hence Gaussian Blur is performed on the image to smooth it out and get rid of all the noise. The Intensity Gradient is calculated for two important reasons- first, a sudden change in intensity means that an edge is detected at that particular location, second, it also specifies the direction of the edge. The final output that us received should consist of thin edges, hence, the step of non-maximum suppression is done. It is followed by Hysteresis Thresholding. If the calculated intensity lies between two values of minimum and maximum, then it is classified as an edge otherwise it is simply discarded. The final image is then cleaned appropriately and provides the necessary accurate output.

In the proposed system, YOLOv3 has been used for Motorcycle Detection, Helmet Detection and Mask Detection. The annotated images have been placed into the respective folders and all the specific parameters have been fine tuned to train the model. Pre-existing weights have been used so that the model need not be built from scratch which saves a lot of time and guarantees more accuracy. The Number Plate Detection module uses the Canny Edge Detection algorithm to extract the number plate. It is a very efficient method and edge localization is highly accurate.

The algorithm for our proposed system is explained below:

1. The most important part of a detection system is the dataset. In the proposed system, we need images that are labelled with the required classification. The first step is to gather the annotated images. The images and the text files that contain the coordinates should be moved into a single folder.
2. The folder is then split into the following manner: 80% of the images are used for training while the remaining 20% is used for testing.

3. Certain parameters for the YOLOv3 algorithm are tweaked- the classes are changed from the default value 80 to the value required for the particular module, the filters are changed according to the number of classes, and the training parameters are uncommented, also the number of iterations are adjusted depending on the number of images provided.
4. Once all the parameters are changed, the training begins. The training uses the pre-defined YOLO weights already available which prevents the need of building the model from scratch.
5. The training occurs for the number of iterations that are specified. Also, simultaneously the Mean Average Precision (MAP) is also calculated.
6. The training ends, once the model completes all iterations. The model can then be run on the videos, or other images.

6.2 Working of the Project

6.2.1 YOLOv3

YOLO (You Only Look Once) is a framework which is used in real-time object detection systems. YOLO makes use of the darknet framework as the main backbone for feature extraction. Darknet is basically a neural network framework which has been written using C language and CUDA. The darknet used in earlier versions of YOLO was Darknet-19 which meant that it had 19 convolutional layers. The Darknet used in YOLOv3 has 53 fully convolutional layers which have been stacked in such a way that the model has 106 layers and the detection happens at 3 different layers. These layers therefore make YOLOv3 a very fast and accurate algorithm used for object detection.

The darknet variant can be cloned from Github easily from AlexeyAB’s resources. Darknet is basically used for GPU computations. The configuration files and the weight files are available in the darknet files and can be easily modified depending on the model we want to create.

OpenCV is a package which is needed to train the model. OpenCV requires the yolov3.cfg, yolov3.weights and obj.names files so that the appropriate training can take place. Certain parameters need to be tweaked in the code. OPENCV is set from 0 to 1. The same changes are required for the GPU, CUDNN, CUDNN_HALF and LIBSO parameters.

The file containing the training and testing data is zipped for convenience and the same file has to be unzipped so that the training can take place. All the necessary files needed for training are also created and uploaded to the drive. Certain tweaks in the parameters are also required here. Depending on the object, the classes and number of filters have to be changed. Also, for the process of training, the batch size is set to 64 and once the model is ready for the images, the batch size is set to 1. The formula for the number of filters has already been given. In order to increase the accuracy, the dataset has to be large to begin with. Depending on the number of images, the max_batches size is decided and also the steps are set to 80% and 90% of the max_batches as well. The image dataset is divided in sucha way that 80% of the images are used for training purposes and the remaining 20% for testing purposes.

Before the actual training begins, we require the weights on which our training takes place. These weights are provided by the following command:

```
!wget http://pjreddie.com/media/files/darknet53.conv.74
```

The most important part of any object detection algorithm is the training process. The accuracy of YOLO models is calculated using the MAP (Mean Average Precision) which is a very important parameter. The value of MAP is calculated after some iterations are completed. The greater the number of iterations, the greater is the accuracy obtained for the model. The loss value is generally below 0.2 if the model is very accurate. The highest accuracy is regularly updated in the best weights file. The last weights file is also updated regularly after each calculation of MAP value. The following code is used to begin the training process:

```
!./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show -
map
```

The below line of code is used to check the output of the object detection on a particular image:

```
!./darknet detector test data/obj.data cfg/yolov3_custom.cfg
/mydrive/yolov3/training/yolov3_custom_last.weights /mydrive/images/t2.jpg
```

6.2.2 Canny Edge Detection

The edge detection process using the Canny algorithm includes various steps which require the appropriate packages to be imported. All these packages are listed below with their uses.

1. pytesseract: pytesseract is an OCR (Optical Character Recognition) Tool available in Python. The main use of this package is to read and recognize the text present in images. It basically writes down the text found in the images. It is an integral part of the License

Plate Detection Module as once, the license plate is extracted, the same image has to be converted into text, so that the records can be maintained.

2. cv2: cv2 denotes the OpenCV library available in Python. It aids in solving problems that come under computer vision. The functions present in this library are used to read input images and show the output images, etc. It is used to analyse images as well as video files. The functions used from this library are explained below:

   a. cvtColor(): The cvtColor() function is used to convert an image into any specific color format.

   b. bilateralfilter(): This function is used to remove any unwanted and unnecessary noise from the image while still keeping the required edges clean.

   c. canny(): canny() function is used to detect the edges in the given image. It uses a multistage algorithm on an image in a greyscale format.

   d. drawContours(): The function is used to draw any possible shape if and only if the boundary points for that image are provided.

   e. arclength(): This function is used to find the arc length of the concerned contour.

   f. approxPolyDP(): This function is used for the approximation of the polygons in an image. If the particular image contains a polygon, then it will be detected accurately in combination with cv2.

   g. boundingRect(): This function can be used to draw a rectangle in the image provided in a binary format.

3. Numpy: NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray , it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

4. Matplotlib: matplotlib. pyplot is a collection of functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a

figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information.

5. VideoCapture(): The function is used to capture livestream with the camera or webcam. It converts the video into a greyscale format. You can either provide the index of the device or the name of the video file so that the appropriate processing can take place.

CHAPTER 7

TESTING

7.Testing

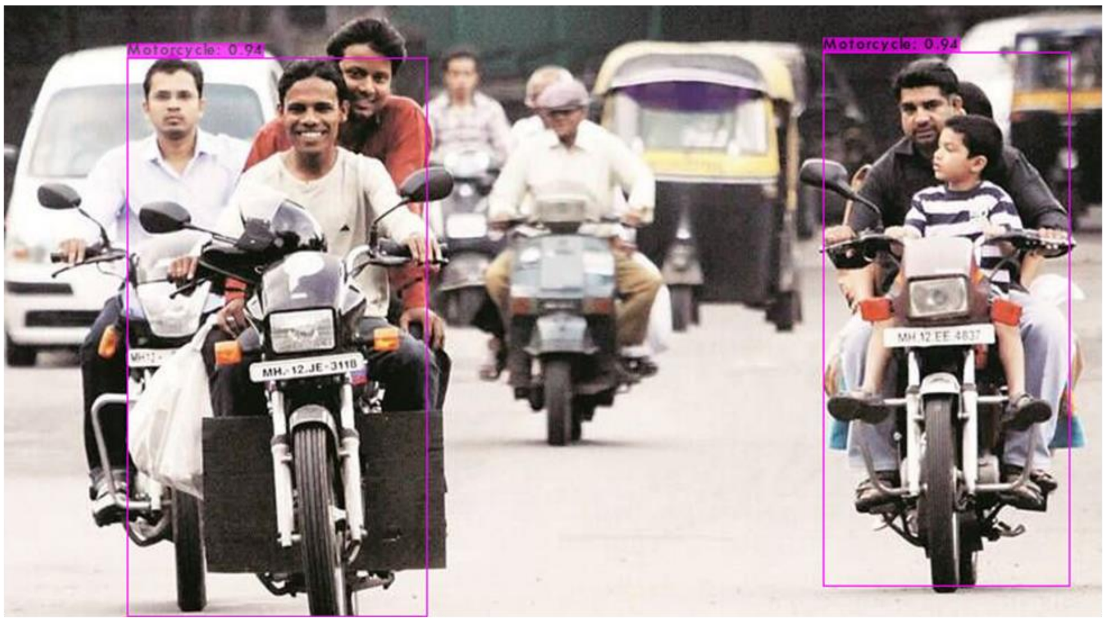
7.1 Test cases

1. Test case Name: Motorcycle Detection 1

   Input:

Expected Output: Image containing all the labelled motorcycles.

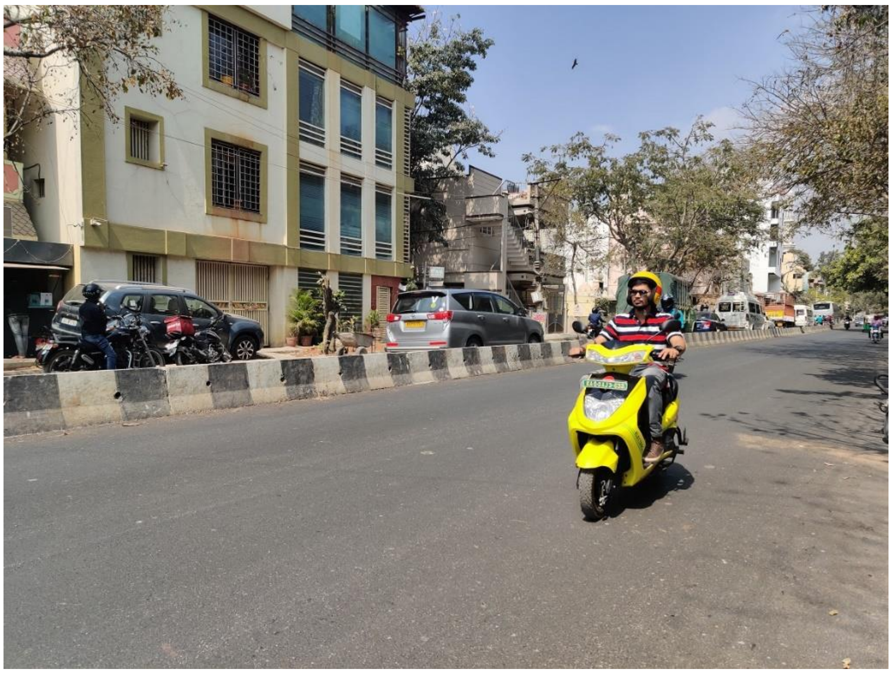
Actual Output:



Status: Partial Success

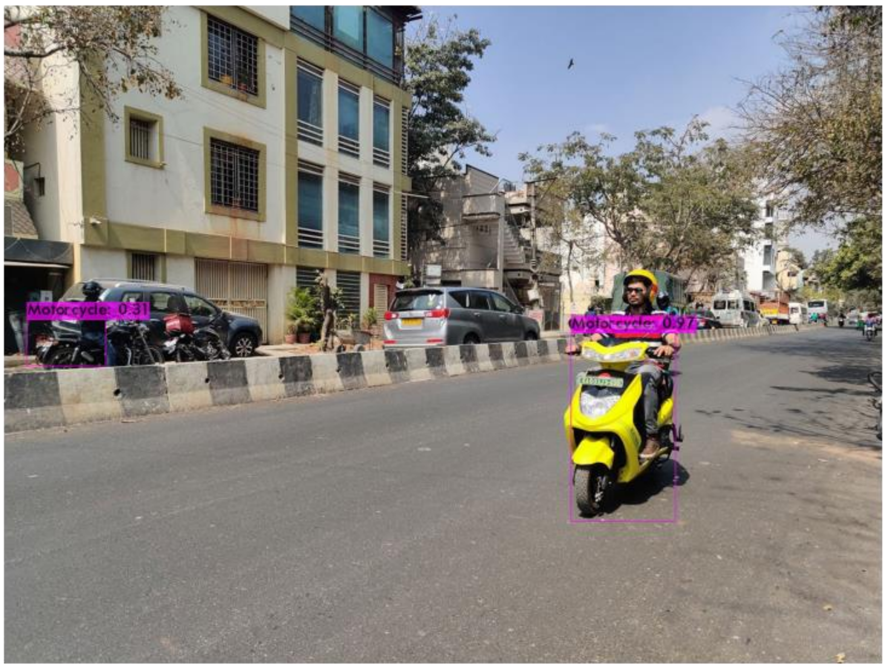
2. Test case Name: Motorcycle Detection 2

   Input:



   Expected Output: Image with all labelled motorcycles

   Actual Output:



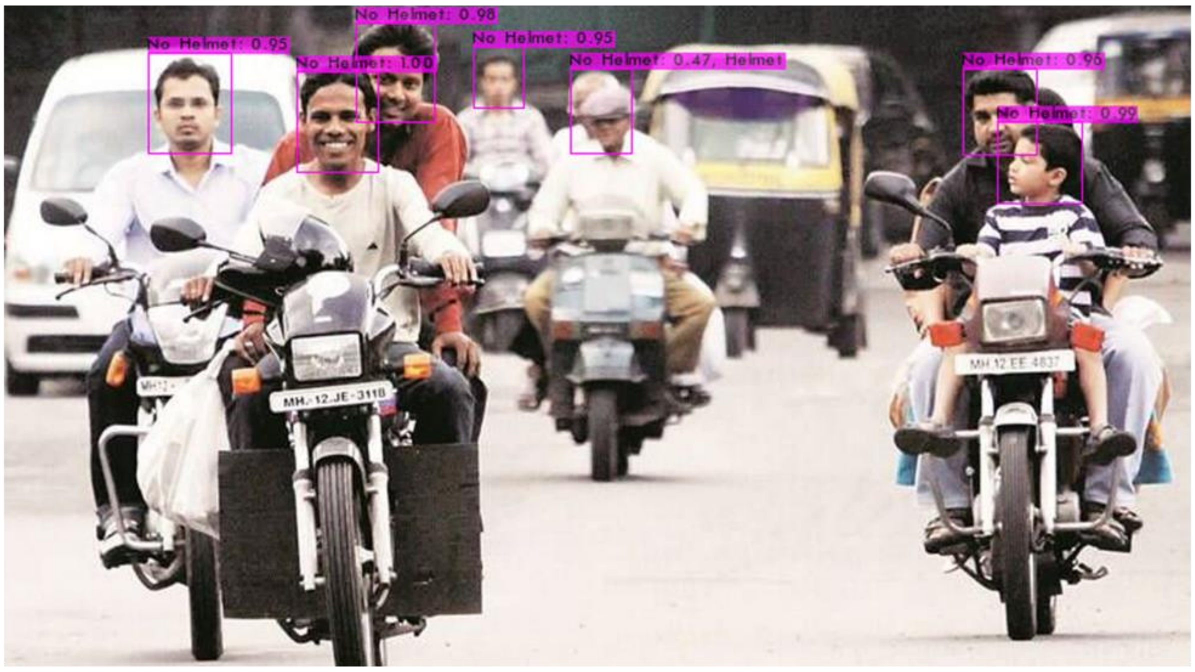
   Status: Success

3. Test case Name: Helmet Detection 1

   Input

   Expected Output: Image with accurate labels of helmet and non-helmet riders.

   Actual Output:



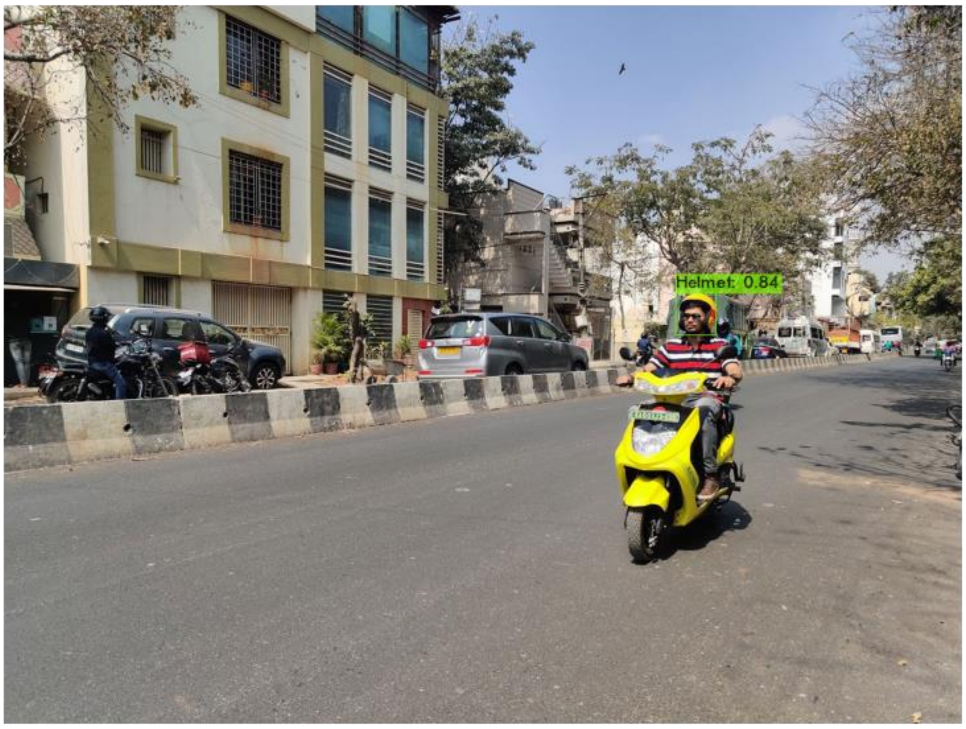
   Status: Success

4. Test case Name: Helmet Detection 2

   Input:

Expected Output: Images with accurate labels of helmet and non-helmet riders.

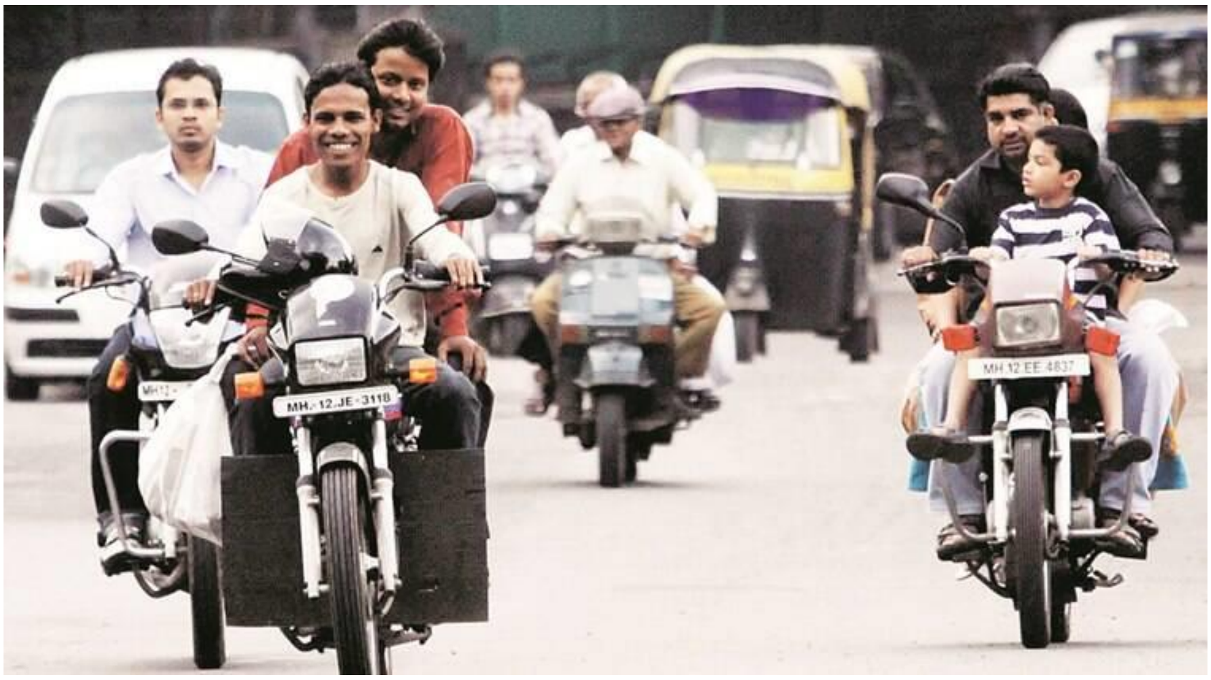
Actual Output:



Status: Partial Success

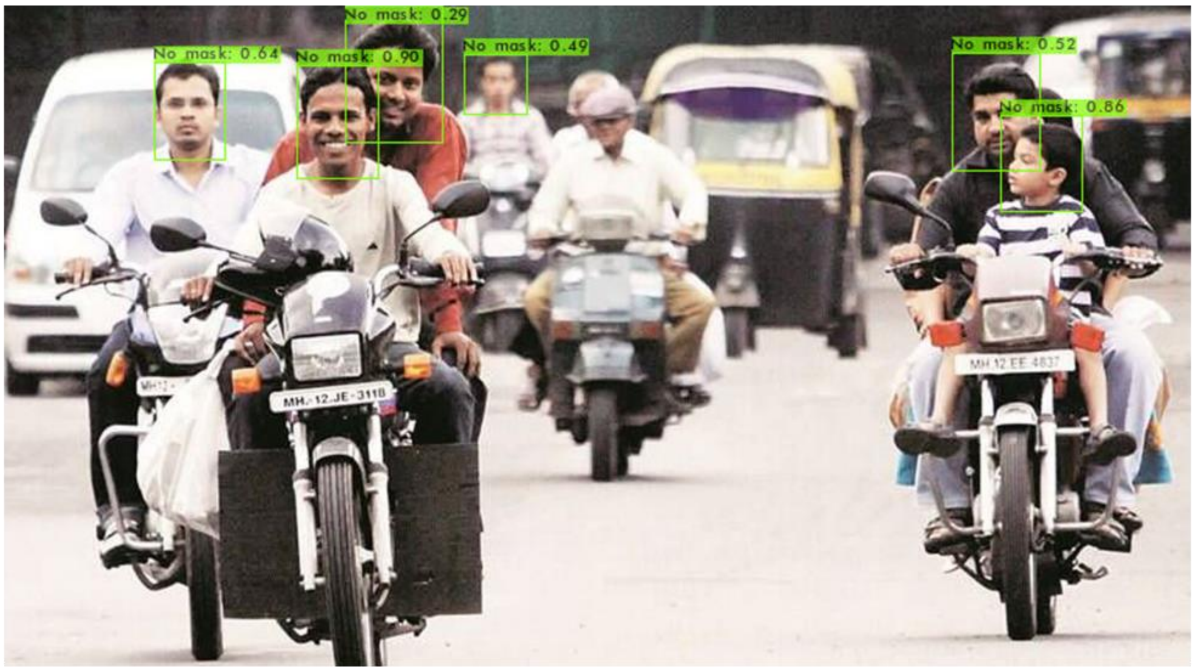
5. Test case Name: Mask Detection 1

   Input:



   Expected Output: Image with accurate labels for masks and no masks.

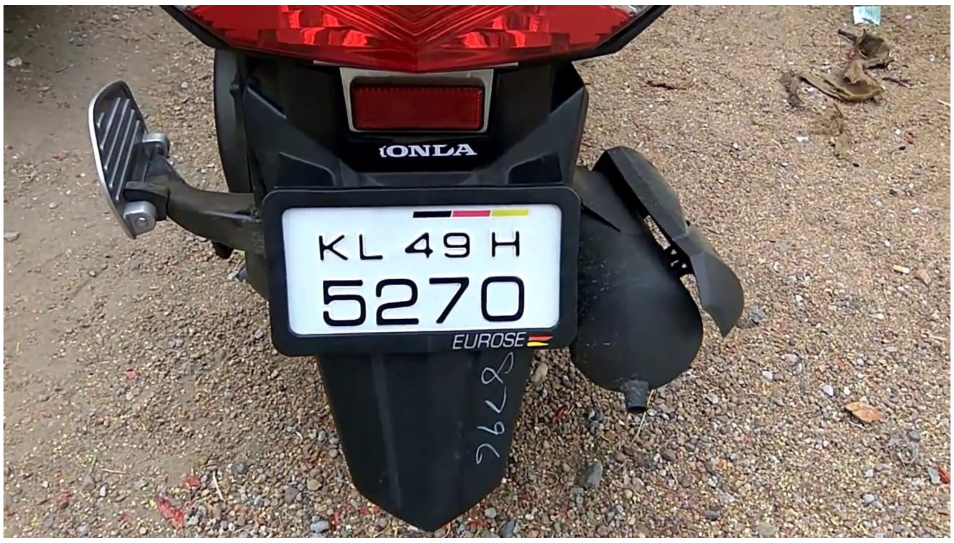
   Actual Output:



   Status: Success

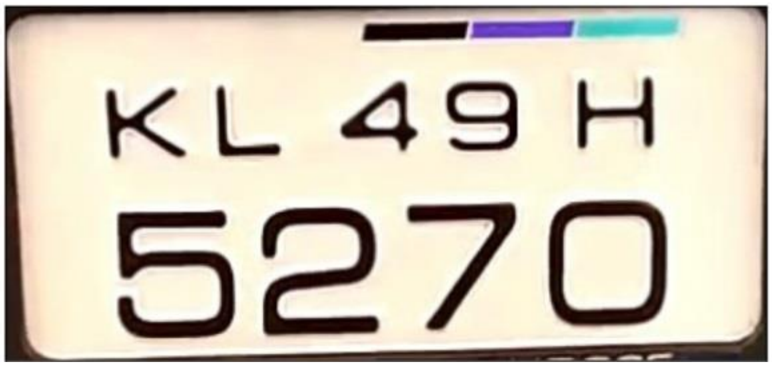
6. Test case Name: License Plate Detection

   Input:



Expected Output: Image containing only the number plate.

Actual Output:



Status: Success

7.2 Type of Testing Used

7.2.1 Unit Testing

Unit testing is the most basic form of testing in which the working of the individual units of a system are tested to ascertain whether they provide an accurate output or not. As it is already known, our system has been divided into four major parts, namely, the Motorcycle, Helmet, Mask and License Plate Detection modules. Once the training has been finished, an image is supplied to the individual modules to gauge its output.

- The motorcycle module is able to identify two wheelers from the image accurately. The accuracy can easily be increased by increasing the training data.
- The helmet detection module has two categories: people wearing helmets and not wearing helmets. Accurate output was obtained in this case as well.
- The mask module also has two categories. This model also provides the required output.
- The license plate detection module extracts the number plate, even when a license plate is at a different angle.

7.2.2 Integration Testing

Integration Testing is a type of testing in which the individual units of a system are combined together and tested to check their working as a single group. The proposed system has been checked on images but the actual working has to be done on a video feed. The required output was obtained and it can be further improved by tweaking certain parameters in the model.

- The motorcycle detection module was provided with a video feed and it provided almost all labelled images.
- The helmet detection module also worked on the same video, providing an accurate result.
- The mask detection module ran smoothly when the video was provided and no issue was faced.

The minor problems faced can be easily removed by increasing the dataset to include a wider range of images.

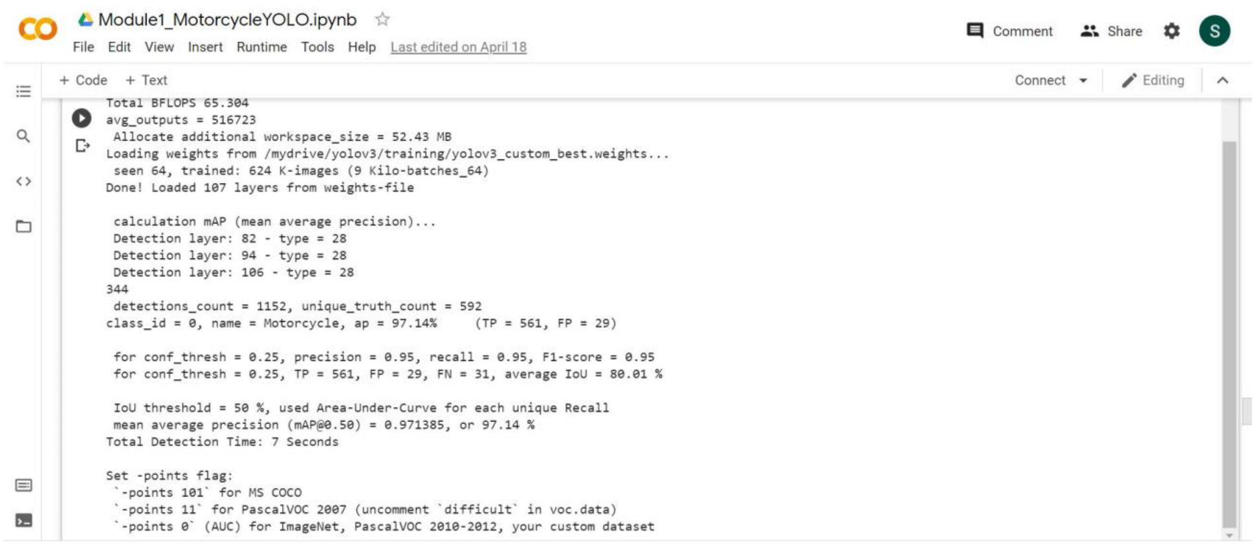
CHAPTER 8

RESULT ANALYSIS AND DISCUSSIONS

8. Results Analysis and Discussions

The accuracy of the models depends on the following factors- the training period for the model and the quantity of the dataset. If the number of iterations for the model is increased, then greater levels of accuracy can be achieved. When a variety of images are added regularly to the dataset, then it becomes relatively easy to attain the desired accuracy for the model.

The YOLOv3 model is first trained to detect motorcycles. Only one class label is used here for classification. The number of filters is set to 18 for the three scales and classes is set to 1 as compared to the original number of 80. A random image was downloaded to check the extent of accurate detection in this model. The Figure 5 below shows the output that was given by the system. The precision value is 96% and recall is around 92%. The MAP (Mean Average Precision) comes out to be 95.92%. The average loss of this model is around 0.2614 which is a great value and thus assures highest accuracy.


Module1_MotorcycleYOLO.ipynb
File Edit View Insert Runtime Tools Help Last edited on April 18
Comment
Share
S
+ Code + Text
Connect
Editing

```
Total BFLOPS 65.304
avg_outputs = 516723
 Allocate additional workspace_size = 52.43 MB
Loading weights from /mydrive/yolov3/training/yolov3_custom_best.weights...
 seen 64, trained: 624 K-images (9 Kilo-batches_64)
Done! Loaded 107 layers from weights-file

 calculation mAP (mean average precision)...
 Detection layer: 82 - type = 28
 Detection layer: 94 - type = 28
 Detection layer: 106 - type = 28
344
 detections_count = 1152, unique_truth_count = 592
class_id = 0, name = Motorcycle, ap = 97.14%     (TP = 561, FP = 29)

 for conf_thresh = 0.25, precision = 0.95, recall = 0.95, F1-score = 0.95
 for conf_thresh = 0.25, TP = 561, FP = 29, FN = 31, average IoU = 80.01 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.971385, or 97.14 %
Total Detection Time: 7 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset
```

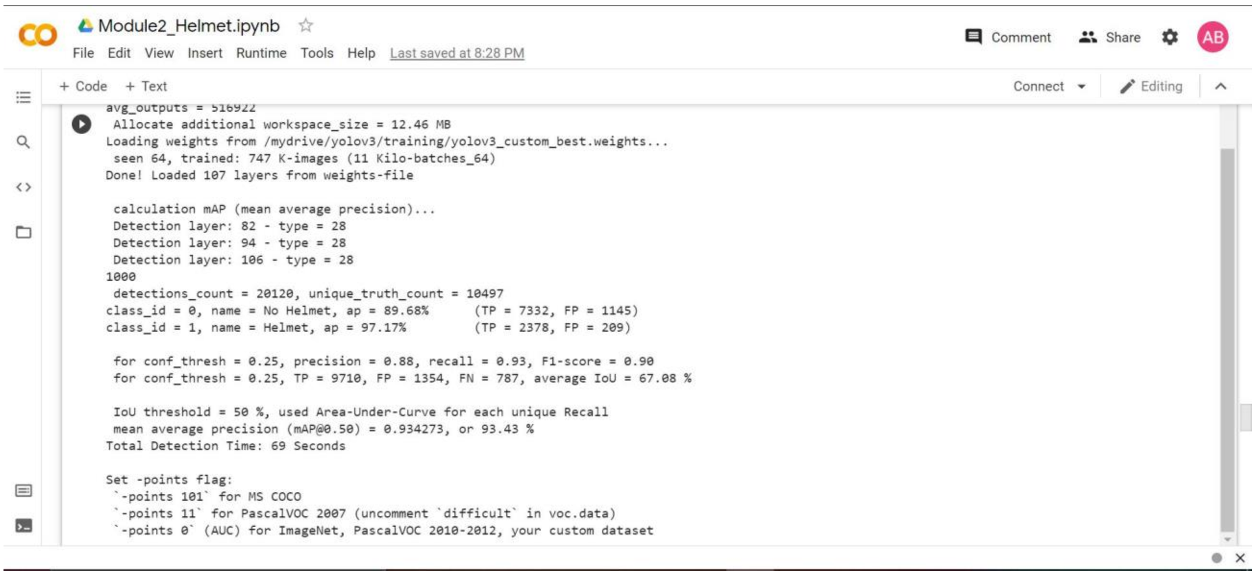
Figure 8.1.1 Motorcycle Detection Accuracy

Table 1 below compares the accuracy of the proposed system with other algorithms.

Table 8.1.1 Accuracy of methods in Vehicle Detection

Sr. No.	Algorithm	Detection accuracy (in %)
1.	YOLOv3 (Proposed system)	97.14
2.	HOG (feature vector) [5]	93
3.	Faster R-CNN (VGG16) [7]	73.2
4.	Fast YOLO [7]	52.7
5.	SSD300 [7]	74.3
6.	Caffe Model [10]	76

The next part of the system is the Helmet Detection which is trained in such a way that it catches all the violators who fail to abide by the road safety rules. By imposing fines on such rule-breakers, a greater emphasis is laid on safety norms. Helmet and No helmet are the two categories in this module. The average precision for Helmet classification is 94.31% and for No helmet classification is 87.37%. The precision value is 86% and recall value is 90%. The MAP (Mean Average Precision) is calculated to be around 90.84%.


Module2_Helmet.ipynb
File Edit View Insert Runtime Tools Help Last saved at 8:28 PM
Comment
Share
AB
+ Code + Text
Connect
Editing

```
avg_outputs = 516922
 Allocate additional workspace_size = 12.46 MB
Loading weights from /mydrive/yolov3/training/yolov3_custom_best.weights...
 seen 64, trained: 747 K-images (11 Kilo-batches_64)
Done! Loaded 107 layers from weights-file

 calculation mAP (mean average precision)...
 Detection layer: 82 - type = 28
 Detection layer: 94 - type = 28
 Detection layer: 106 - type = 28
1000
 detections_count = 20120, unique_truth_count = 10497
class_id = 0, name = No Helmet, ap = 89.68%      (TP = 7332, FP = 1145)
class_id = 1, name = Helmet, ap = 97.17%         (TP = 2378, FP = 209)

 for conf_thresh = 0.25, precision = 0.88, recall = 0.93, F1-score = 0.90
 for conf_thresh = 0.25, TP = 9710, FP = 1354, FN = 787, average IoU = 67.08 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.934273, or 93.43 %
Total Detection Time: 69 Seconds

Set -points flag:
 `-points 101` for MS COCO
 `-points 11` for PascalVOC 2007 (uncomment `difficult` in voc.data)
 `-points 0` (AUC) for ImageNet, PascalVOC 2010-2012, your custom dataset
```

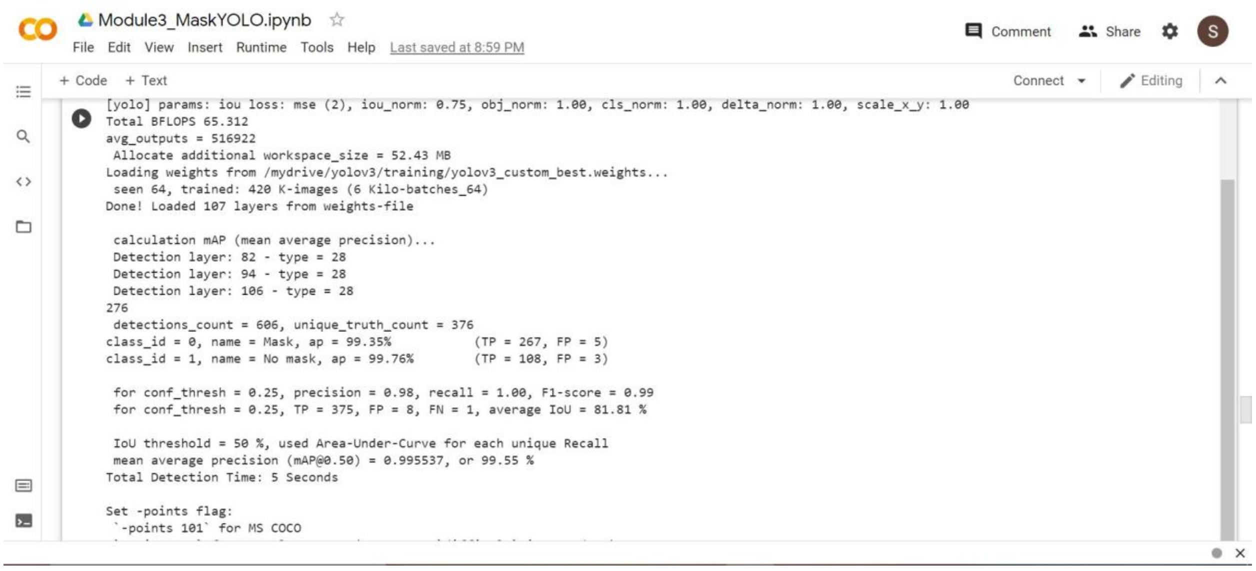
Figure 8.1.2 Helmet Detection Accuracy

Table 2 below shows the accuracy percentage of the helmet detection models.

Table 8.1.2 Accuracy of methods in Helmet Detection

Sr. No.	Algorithm	Accuracy (in %)
1.	YOLOv3 (proposed system)	93.43
2.	YOLO [2]	81
3.	CNN [5]	85
4.	InceptionV3 [10]	81
5.	CNN based MTL [22]	80.7

Mask Detection is the most important part of the system as it includes a new functionality which is an advantage when it comes to the pandemic situation. The dataset containing images of masks as well as images without masks is used for training. There are two classes for this classification: Mask and No mask. The average precision of Mask classification is 99.20% and that of No mask classification is 99.55%. The precision value comes around 95% and recall is calculated around 98%. The MAP value for this module is 99.36%.


Module3_MaskYOLO.ipynb
File Edit View Insert Runtime Tools Help Last saved at 8:59 PM
Comment
Share
+ Code + Text
Connect
Editing

```
[yolo] params: iou loss: mse (2), iou_norm: 0.75, obj_norm: 1.00, cls_norm: 1.00, delta_norm: 1.00, scale_x_y: 1.00
Total BFLOPS 65.312
avg_outputs = 516922
 Allocate additional workspace_size = 52.43 MB
Loading weights from /mydrive/yolov3/training/yolov3_custom_best.weights...
 seen 64, trained: 420 K-images (6 Kilo-batches_64)
Done! Loaded 107 layers from weights-file

 calculation mAP (mean average precision)...
 Detection layer: 82 - type = 28
 Detection layer: 94 - type = 28
 Detection layer: 106 - type = 28
276
 detections_count = 606, unique_truth_count = 376
class_id = 0, name = Mask, ap = 99.35%           (TP = 267, FP = 5)
class_id = 1, name = No mask, ap = 99.76%        (TP = 108, FP = 3)

 for conf_thresh = 0.25, precision = 0.98, recall = 1.00, F1-score = 0.99
 for conf_thresh = 0.25, TP = 375, FP = 8, FN = 1, average IoU = 81.81 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.995537, or 99.55 %
Total Detection Time: 5 Seconds

Set -points flag:
 `-points 101` for MS COCO
```

Figure 8.1.3 Mask Detection Accuracy

Table 8.1.3 Accuracy of methods in Mask Detection

Sr. No.	Algorithm	Accuracy (in %)
1.	YOLOv3 (proposed system)	99.55
2.	CNN [13]	98.7
3.	Sequential CNN [15]	95.77 / 94.58
4.	SVM and MobileNetV2 [20]	97.1
5.	DNN based on SSD [21]	97.25

Number Plate Detection makes use of the Canny Edge Detection Model which detects all the edges accurately and crops out only the part of the image that contains a number plate. Also, the image of the number plate can be worked upon to retrieve the number plate in a text format. Figure 10 is the input provided to this module, while Figure 11 is the output generated after all the calculations.

CHAPTER 9

CONCLUSION AND FUTURE SCOPE

9. Conclusion & Future Scope

A system has been developed for bike riders who violate laws like not wearing a helmet and mask. This project helps in imposing the rules like wearing a helmet and mask for a bike rider in a stricter manner as it is difficult for a traffic policeman to keep record of all the bike riders who violate such laws as the bike riders speeds up and runs away. In such scenarios, this system helps in imposing more strictness in the rule of wearing helmet and mask as all those bike riders who violate such laws can be detected with the help of their number plate and they can’t run away from this system. This helps in reducing any kind of possibility of facing an injury for not wearing a helmet or coming in contact with any kind of virus for not wearing a mask. This system mainly consists of four parts - Detection of Bike, Detection of Helmet, Detection of Mask and Detection of Number Plate of those bike riders who don't wear helmet or mask. Firstly, the system checks if the vehicle in the image is a bike or not using YOLOv3. Then the detection of helmet and mask on the bike rider is checked using YOLOv3. Those bike riders who violate laws like not wearing a helmet or mask then the number plate of those bike riders is extracted using canny edge detector. The accuracy obtained for detecting a bike is 95.92% and the accuracy obtained for detecting a helmet is 90.84% and the accuracy for detecting a mask is 99.36%. The accuracy of the system can be increased further by increasing the training dataset and training the algorithm for more time.

Following are the Future Scope of the proposed system:

- The proposed system can be used in all those places where motorcycle detection is needed.
- It can also be implemented near construction sites to check and detect whether the bike riders are wearing a helmet or not.
- The proposed system can also be implemented to check whether the people walking on the roads or any other place are wearing masks or not.
- Further, a payment module could be added into the system so that the traffic rule violators could be fined.

APPENDIX

Appendix

1. Code to generate frames

```
import cv2

vidcap = cv2.VideoCapture('/content/drive/MyDrive/images/8.mp4')

def getFrame(sec):

vidcap.set(cv2.CAP_PROP_POS_MSEC,sec*1000)

hasFrames,image = vidcap.read()

if hasFrames:

cv2.imwrite("image"+str(count)+".jpg", image)     # save frame as JPG file

return hasFrames

sec = 0

frameRate = 0.5 #//it will capture image in each 0.5 second

count=1

success = getFrame(sec)

while success:

count = count + 1

sec = sec + frameRate

sec = round(sec, 2)

success = getFrame(sec)
```

2. Code for Motorcycle Classification:

```
!git clone https://github.com/AlexeyAB/darknet

!ln -s /content/drive/MyDrive/ /mydrive

!ls /mydrive
```

```
!sed -i 's/OPENCV=0/OPENCV=1/' Makefile

!sed -i 's/GPU=0/GPU=1/' Makefile

!sed -i 's/CUDNN=0/CUDNN=1/' Makefile

!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile

!sed -i 's/LIBSO=0/LIBSO=1/' Makefile

!make

!find -maxdepth 1 -type f -exec rm -rf {} \;

!cp /mydrive/yolov3/obj.zip ../

!unzip ../obj.zip -d data/

!cp /mydrive/yolov3/yolov3_custom.cfg ./cfg

!cp /mydrive/yolov3/obj.names ./data

!cp /mydrive/yolov3/obj.data  ./data

!cp /mydrive/yolov3/process.py ./

!python process.py

!ls data/

!wget http://pjreddie.com/media/files/darknet53.conv.74

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show -
map

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg
/mydrive/yolov3/training/yolov3_custom_last.weights -dont_show -map

def imShow(path):

import cv2

import matplotlib.pyplot as plt

image = cv2.imread(path)

height, width = image.shape[:2]
```

```
resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

fig = plt.gcf()

fig.set_size_inches(18, 10)

plt.axis("off")

plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))

imShow('chart.png')

!./darknet detector map data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_best.weights -points 0

!sed -i 's/batch=64/batch=1/' yolov3_custom.cfg

!sed -i 's/subdivisions=16/subdivisions=1/' yolov3_custom.cfg

!./darknet detector test data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_last.weights /mydrive/images/t2.jpg

imShow('predictions.jpg')

!./darknet detector demo data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_best.weights -dont_show /mydrive/images/8.mp4 -i 0 -out_filename /mydrive/images/result.avi
```

3. Code for Helmet Detection:

```
!git clone https://github.com/AlexeyAB/darknet

!ln -s /content/drive/MyDrive/ /mydrive

!ls /mydrive

!sed -i 's/OPENCV=0/OPENCV=1/' Makefile

!sed -i 's/GPU=0/GPU=1/' Makefile

!sed -i 's/CUDNN=0/CUDNN=1/' Makefile

!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile

!sed -i 's/LIBSO=0/LIBSO=1/' Makefile
```

```
!make

!find -maxdepth 1 -type f -exec rm -rf {} \;

!cp /mydrive/yolov3/obj.zip ../

!unzip ../obj.zip -d data/

!cp /mydrive/yolov3/yolov3_custom.cfg ./cfg

!cp /mydrive/yolov3/obj.names ./data

!cp /mydrive/yolov3/obj.data  ./data

!cp /mydrive/yolov3/process.py ./

!python process.py

!ls data/

!wget http://pjreddie.com/media/files/darknet53.conv.74

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show -
map

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg
/mydrive/yolov3/training/yolov3_custom_last.weights -dont_show -map

def imShow(path):

import cv2

import matplotlib.pyplot as plt

image = cv2.imread(path)

height, width = image.shape[:2]

resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

fig = plt.gcf()

fig.set_size_inches(18, 10)

plt.axis("off")

plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))
```

```
imShow('chart.png')

!./darknet detector map data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_best.weights -points 0

!sed -i 's/batch=64/batch=1/' yolov3_custom.cfg

!sed -i 's/subdivisions=16/subdivisions=1/' yolov3_custom.cfg

!./darknet detector test data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_last.weights /mydrive/images/t2.jpg

imShow('predictions.jpg')

!./darknet detector demo data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_best.weights -dont_show /mydrive/images/8.mp4 -i 0 -out_filename /mydrive/images/result.avi
```

4. Code for Mask Detection:

```
!git clone https://github.com/AlexeyAB/darknet

!ln -s /content/drive/MyDrive/ /mydrive

!ls /mydrive

!sed -i 's/OPENCV=0/OPENCV=1/' Makefile

!sed -i 's/GPU=0/GPU=1/' Makefile

!sed -i 's/CUDNN=0/CUDNN=1/' Makefile

!sed -i 's/CUDNN_HALF=0/CUDNN_HALF=1/' Makefile

!sed -i 's/LIBSO=0/LIBSO=1/' Makefile

!make

!find -maxdepth 1 -type f -exec rm -rf {} \;

!cp /mydrive/yolov3/obj.zip ../

!unzip ../obj.zip -d data/

!cp /mydrive/yolov3/yolov3_custom.cfg ./cfg
```

```
!cp /mydrive/yolov3/obj.names ./data

!cp /mydrive/yolov3/obj.data  ./data

!cp /mydrive/yolov3/process.py ./

!python process.py

!ls data/

!wget http://pjreddie.com/media/files/darknet53.conv.74

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg darknet53.conv.74 -dont_show -
map

!./darknet detector train data/obj.data cfg/yolov3_custom.cfg
/mydrive/yolov3/training/yolov3_custom_last.weights -dont_show -map

def imShow(path):

import cv2

import matplotlib.pyplot as plt

image = cv2.imread(path)

height, width = image.shape[:2]

resized_image = cv2.resize(image,(3*width, 3*height), interpolation = cv2.INTER_CUBIC)

fig = plt.gcf()

fig.set_size_inches(18, 10)

plt.axis("off")

plt.imshow(cv2.cvtColor(resized_image, cv2.COLOR_BGR2RGB))

imShow('chart.png')

!./darknet detector map data/obj.data cfg/yolov3_custom.cfg
/mydrive/yolov3/training/yolov3_custom_best.weights -points 0

!sed -i 's/batch=64/batch=1/' yolov3_custom.cfg

!sed -i 's/subdivisions=16/subdivisions=1/' yolov3_custom.cfg
```

```
!./darknet detector test data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov3_custom_last.weights /mydrive/images/t2.jpg

imShow('predictions.jpg')
!./darknet detector demo data/obj.data cfg/yolov3_custom.cfg /mydrive/yolov3/training/yolov
3_custom_best.weights -dont_show /mydrive/images/8.mp4 -i 0 -
out_filename /mydrive/images/result.avi
```

5. Code for License Plate Detection:

```
!sudo apt install tesseract-ocr
!pip install pytesseract
import cv2
import random
import os
import numpy as np
import matplotlib.pyplot as plt
images_dir = "/content/drive/MyDrive/images/"
image_files = os.listdir(images_dir)
image_path = "{}/{}".format(images_dir, "avbcd.jpg")
image = cv2.imread(image_path)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
def plot_images(img1, img2, title1="", title2=""):
fig = plt.figure(figsize=[15,15])
ax1 = fig.add_subplot(121)
ax1.imshow(img1, cmap="gray")
ax1.set(xticks=[], yticks=[], title=title1)
ax2 = fig.add_subplot(122)
ax2.imshow(img2, cmap="gray")
ax2.set(xticks=[], yticks=[], title=title2)
plot_images(image, gray)
blur = cv2.bilateralFilter(gray, 11,90, 90)
```

```
plot_images(gray, blur)
edges = cv2.Canny(blur, 30, 200)
plot_images(blur, edges)
cnts, new = cv2.findContours(edges.copy(), cv2.RETR_LIST, cv2.CHAIN_APPROX_SIMPLE)
image_copy = image.copy()
_ = cv2.drawContours(image_copy, cnts, -1, (255,0,255),2)
plot_images(image, image_copy)
cnts = sorted(cnts, key=cv2.contourArea, reverse=True)[:30]
image_copy = image.copy()
_ = cv2.drawContours(image_copy, cnts, -1, (255,0,255),2)
plot_images(image, image_copy)
plate = None
for c in cnts:
perimeter = cv2.arcLength(c, True)
edges_count = cv2.approxPolyDP(c, 0.02 * perimeter, True)
if len(edges_count) == 4:
x,y,w,h = cv2.boundingRect(c)
plate = image[y:y+h, x:x+w]
break
cv2.imwrite("plate.png", plate)
plot_images(plate, plate)
import pytesseract
text = pytesseract.image_to_string(plate, lang="eng")
print(text)
```

LITERATURE CITED

Literature Cited

[1] F. A. Khan, N. Nagori and A. Naik, "Helmet and Number Plate detection of Motorcyclists using Deep Learning and Advanced Machine Vision Techniques," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020.

[2] Emy Barnabas, Amritha B.J. “Helmet Detection and License Plate Recognition Using CNN”, 2019 IOSR Journal of Engineering (IOSRJEN), vol. 09, no. 06.

[3] K G Shreyas Dixit, Mahima Girish Chadaga, Sinchana S Savalgimath, G Ragavendra Rakshith, Naveen Kumar M R, “Evaluation and Evolution of Object Detection Techniques YOLO and R-CNN”, 2019, vol. 8.

[4] C. A. Rohith, S. A. Nair, P. S. Nair, S. Alphonsa and N. P. John, "An Efficient Helmet Detection for MVD using Deep learning," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019.

[5] M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J. -H. Kim, "An Automated System to Limit COVID-19 Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, 2020.

[6] A. Das, M. Wasif Ansari and R. Basak, "Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV," 2020 IEEE 17th India Council International Conference (INDICON), New Delhi, India, 2020.

[7] A. P. Thombare and S. B. Bagal, "A distributed canny edge detector: Comparative approach," 2015 International Conference on Information Processing (ICIP), Pune, India, 2015.

[8] A. Oumina, N. El Makhfi and M. Hamdi, "Control The COVID-19 Pandemic: Face Mask Detection Using Transfer Learning," 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS), Kenitra, Morocco, 2020.

[9] I. Buciu, "Color quotient-based mask detection," 2020 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 2020.

[10] H. Lin, J. D. Deng, D. Albers and F. W. Siebert, "Helmet Use Detection of Tracked Motorcycles Using CNN-Based Multi-Task Learning," in IEEE Access, vol.

PUBLICATIONS

Publications

Paper Title: Mask and Helmet Detection in Two-wheelers using YOLOv3 and Canny Edge Detection.

Journal: International Research Journal of Engineering and Technology (IRJET)

Volume: Volume 8- Issue 4- April 2021

p-ISSN: 2395-0072

e-ISSN: 2395-0056

Name of the Authors: Shravani Maliye, Jayom Oza, Jayesh Rane, Prof. Nileema Pathak

International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 08 Issue: 04 | Apr 2021 www.irjet.net p-ISSN: 2395-0072

Mask and Helmet Detection in Two-wheelers using YOLOv3 and Canny Edge Detection

Shravani Maliye[1], Jayom Oza[2], Jayesh Rane[3], Nileema Pathak[4]

[1,2,3] Student, Information Technology Department, Atharva College of Engineering, Maharashtra, India
[4]Professor, Atharva College of Engineering, Maharashtra, India

---***---

Abstract - *There is no automated existing system which can detect motorcyclists who are not wearing helmets as well as masks due to which the traffic policemen have to manually keep records of such traffic rules violators either by remembering the number plate or by capturing a photo of the number plate. This manual administration can sometimes lead to errors. In order to overcome these drawbacks, we have designed an automated helmet and face mask detection system which is able to catch all the motorcyclists who are not wearing helmets and masks just by storing the number plate of those bike riders. The system is divided into four modules. It begins by classifying the Two Wheelers from a frame generated by the traffic camera feed. Once the two wheelers are classified by the system then it checks for the presence of a helmet on the bike rider first and then it finds the presence of face mask on the bike rider. If the bike rider is found without a helmet or face mask or both then it extracts the number plate from the image and stores it in the system. The Motorcycle, Helmet and Mask Detection modules use the YOLOv3 algorithm to detect their respective images. The License Plate is extracted using Canny Edge Detection method. The system demonstration has been shown on Google Colab. This system will help to catch traffic rule violators thereby creating more strictness in traffic rules and thus reducing risk of injuries.*

Key Words: **Motorcycle, Helmet, Mask, Detection, License Plate, YOLOv3, Canny Edge Detection.**

1.INTRODUCTION

Two-wheeler vehicles such as scooters and motorcycles are the most popular type of vehicles in India because of their availability and convenience. They are owned by a major part of the population in the country according to the statistics provided by the Department of Statistics and Implementation Systems (MSPI), the Government of India. According to a report released by the Department of Transport and Highways, more than 37% of people die in road accidents (56,136) or six every hour on an average - including two wheelers. It is highly recommended for two-wheeler riders to use a safety helmet to reduce the risk of injury. It is a criminal offense to ride a two-wheeler without a helmet and many hand-to-hand tactics have been adopted to catch violators because of their importance. So, the automation of this process in real time is a need of the hour and will help to accurately monitor passengers who break the rules, and greatly reduce the number of human interventions. In the existing system, the police officer has to manually capture the image of the number plate of the bike rider who is not wearing the helmet and the face mask. A very big disadvantage of this method is that often such rule-violators speed up and run off and are not penalized for their actions. To overcome the disadvantages of the existing system, we have proposed an automated system which is more accurate and requires minimum human efforts. The main application of this system is to catch all the motorcyclists who are not wearing helmets and face masks. Now let us understand the working of the system. The system will divide the traffic video into various images(frames) and from those images the system is able to detect bike riders from an image consisting of both two wheelers and other vehicles using YOLOv3. Once the bike riders are detected, the system is able to detect if the motorcyclist is wearing a helmet or not using YOLOv3, also the system is able to detect if the motorcyclist is wearing a mask or not using YOLOv3. The system will then record the number plate of those bike riders who are not wearing helmet or face mask or both using the Canny Edge Detection Method.

2. LITERATURE REVIEW

Fahad A Khan, Nitin Nagori, Dr. Ameya Naik [1]

In today's world, the increasing use of Motor-bikes has prompted increment in road accidents and injuries. Helmet not used by the motorcycle rider is one of the major causes. Currently, the police officer has to manually capture the image of the number plate of the bike rider who is not wearing the helmet. The proposed system distinguishes the difference between motorcycle rider with or without helmet from frames. Based on feature extractor the system extracts object class. The system uses You Only Look Once (YOLO)-Darknet deep learning framework which consists of CNN algorithm trained on Common Objects in Context (COCO) and combined with computer vision. YOLO's convolutional layers are modified to detect specified three classes and it uses a sliding-window process. The map (Mean Average Precision) on validation dataset achieved 81% by using training data.

Emy Barnabas, AmrithaB.J [2]

Nowadays, road accidents are one of the leading causes of death. Among them, bike accidents are common and cause serious injuries. A helmet is one of the most important aspects of a safety for a rider. However, many fails to follow the rule of wearing a helmet. They have implemented a system to find motorcyclists violating the rules of the helmet, using a system that uses image processing and a convolutional neural network. The program consists of the identification of a motor-bike, a protection helmet and the identification of the number plate. Motorcycles were found using the HOG vector. Once the motorcycle has been detected, using a convolutional neural network, it is determined whether the motorcyclist is wearing a helmet or not. When a motorcyclist is found to be without a helmet, a motorcycle license plate is obtained using the Tesseract OCR.

K G Shreyas Dixit, Mahima Girish Chadaga, Sinchana S Savalgimath, G Ragavendra Rakshith, Naveen Kumar M R [3]

In this paper authors use R-CNN, Fast R-CNN and Faster R-CNN for object detection and compare all three algorithms and also explained the YOLOv2 algorithm and why it is better from traditional approach. They concluded that Fast-RCNN have mean average precision up to 76.4 but it has between 5 to 18 frames per second(fps). On the other hand, YOLOv2 algorithm has mean average precision up to 78.6, and can attained speed up to 155 frame per second(fps). YOLOv2 attains an outstanding trade-off between accuracy and speed and also as a detector possessing powerful generalization capabilities of representing an entire image.

Rohith C A, Shilpa A Nair, Parvathi Sanil Nair, Sneha Alphonsa, Nithin Prince John [4]

In this paper the authors intend to make an automated system to distinguish whether a biker is wearing a helmet or not and to impose fine to defaulters as a part of law enforcement. For execution of the following idea, they have used Caffe model and InceptionV3. According to our research, such a system is not currently used by the police or by any other authority. Executing the proposed framework can convey more mindfulness and need to wear a helmet at any rate with the goal that they don't get captured on camera and avoid fine. This project intends to make deep learning based automated detection system for helmet identification using trained models and datasets that would be useful for the police department to enforce the law for the betterment of the society.

Mohammad Marufur Rahman, Md. Motaleb Hossen Manik, Md. Milon Islam, Saifuddin Mahmud, Jong-Hoon Kim [5]

The COVID-19 epidemic caused by the novel coronavirus continues to spread worldwide. The impact of COVID-19 has fallen on almost all development sectors. The health care system is in trouble. Many precautionary measures have been taken to reduce the spread of the disease when wearing a mask is one of them. In this paper, they propose a program that finds naked people who are not wearing face mask on a smart city network where all public areas are monitored by Closed-Circuit Television (CCTV) cameras. When a person without a mask is found, the corresponding officer is informed. Deep architecture is trained in a database that contains images of people with and without masks collected from various sources. Professional facilities have found 98.7% accuracy in distinguishing people with face masks for previously unseen test data.

3. MATERIALS

OID stands for Open Image Dataset. It is of great help when it comes to computer vision applications. There are around 9 million images in the dataset spanning over various categories. The images in the OID Dataset have been prepared by Google with labels, bounding boxes, segmentation masks, localized narratives and visualized relationships.

Kaggle is an online community where people can download datasets, share the datasets that have been created by them- in short, they have a plethora of options when it comes to dealing with data.

Image datasets spanning over three different categories were required for this project- motorcycle images, images with helmet and without helmet and people wearing face mask as well as not wearing face masks. Google's OIDv4 which stands for Open Image Dataset version 4, was used for providing images for the first two categories. Images of type Detection already have bounding boxes drawn on them corresponding to the category selected. Motorcycle images, were downloaded directly for the first detection module. Images with human face and helmets were downloaded for the second detection module. With respect to the third module, Mask Detection, the required images were downloaded from Kaggle which provided a dataset which contained images with mask and without mask. The manual work of drawing bounding boxes on each image and preparing a custom dataset was reduced to a great extent. Each category contained approximately a little above 4000 images which included test data as well as training data.

4. PROPOSED SYSTEM

4.1 System architecture

The frames created from the existing traffic camera feed is passed to the motorcycle detection module. If a motorcycle is detected, then the image is sent to the helmet detection system. If a helmet is not detected, then the license plate is extracted which is then stored in the system. Also, the same image is sent to the mask detection module. If no mask is found, then the number plate is found to store the required fines. The unnecessary frames are simply discarded, while, if any safety violations are found, then the information is stored in the system. The block diagram below highlights the system architecture:

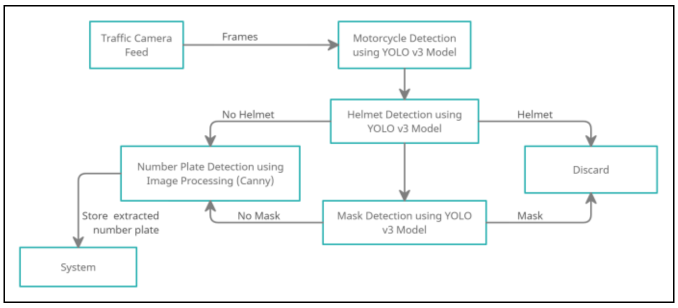

Traffic Camera Feed
Frames
Motorcycle Detection using YOLO v3 Model
No Helmet
Helmet Detection using YOLO v3 Model
Helmet
Number Plate Detection using Image Processing (Canny)
Discard
Store extracted number plate
No Mask
Mask Detection using YOLO v3 Model
Mask
System

Fig-4: Block Diagram

5. METHODOLOGY

A model of the system has been created on Google Colab to simulate the working of the actual system. The annotated images have been used to provide the appropriate data for the training as well as testing processes. The MAP (Mean Average Precision) has also been calculated. License Plate has been extracted using the Canny edge detector.

5.1 MODULES IN THE SYSTEM

a. Motorcycle Detection: This module is used to detect two-wheelers from the image passed to it. The detected frames are sent to the next module while the other frames are simply discarded.
b. Helmet Detection: If the rider doesn't wear a helmet, then the vehicle's license plate is extracted and the data is stored in the system, else, it is discarded. Also, the same frame is sent to the third module.
c. Mask Detection: If the rider doesn't wear a mask, then the license plate is extracted, otherwise, the frame is simply discarded.
d. License Plate Extraction: Using Canny edge detection technique, the license plate is easily found and stored in the system for maintaining records.

5.2 YOLOv3 (You Only Look Once version 3)

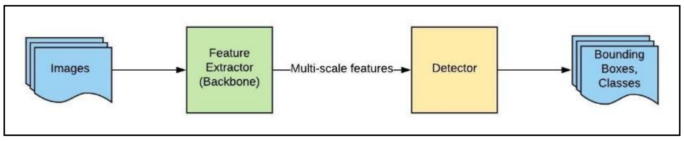


Fig-2: YOLO architecture

In this model, a single neural network is applied to the concerned image. The image is then divided into multiple regions and probabilities are calculated and depending on their values bounding boxes are mapped. Non max suppression is deployed so that each object is detected only once and the overall accuracy of the model is increased. The output is then provided in the format that it shows the bounding box along with the appropriate class label and the probability calculated for that label.

The version of YOLO used in the proposed system is version 3. A Darknet variant of 53 layers is used in this version in such a way that the architecture of this model consists of 106 fully functional convolutional layers. The most astounding features of this version is that it makes predictions at three different scales, also, it is more accurate in detecting even smaller objects.

5.3 Canny Edge Detection

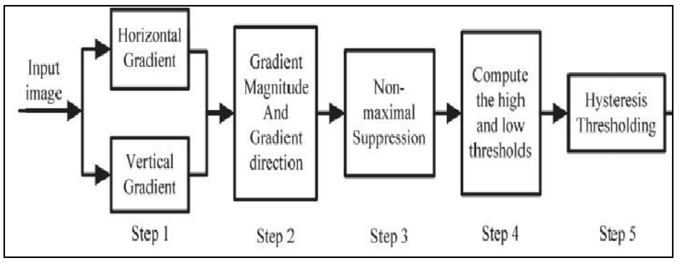


Fig-3: Canny Edge Detection [7]

Canny Edge Detection model is an algorithm used to detect edges in various images. The working of this model is pretty straightforward. [7] The original image is supplied to the algorithm as an input. This image is then converted to a greyscale format. The image is susceptible to a lot of noise; hence Gaussian Blur is performed on the image to smooth it out and get rid of all the noise. The Intensity Gradient is calculated for two important reasons- first, a sudden change in intensity means that an edge is detected at that particular location, second, it also specifies the direction of the edge. The final output that us received should consist of thin edges, hence, the step of non-maximum suppression is done. It is followed by Hysteresis Thresholding. If the calculated intensity lies between two values of minimum and maximum, then it is classified as an edge otherwise it is simply discarded. The final image is then cleaned appropriately and provides the necessary accurate output.

6. RESULTS

The accuracy of the models depends on the following factors- the training period for the model and the quantity of the dataset. If the number of iterations for the model is increased, then greater levels of accuracy can be achieved. When a variety of images are added regularly to the dataset, then it becomes relatively easy to attain the desired accuracy for the model.

The below image shows the output obtained on running the model with a particular image. The table below it compares the accuracy of other algorithms with the proposed algorithm.

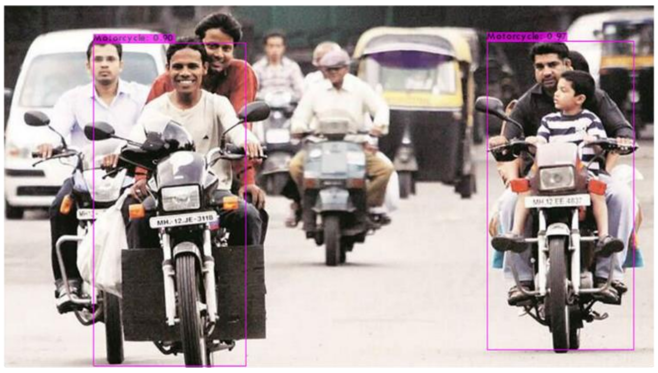


Fig-4: Motorcycle Detection Output

Table-1: Accuracy of methods in Vehicle Detection

Algorithm	Detection accuracy (in %)
YOLOv3 (Proposed system)	95.92
HOG (feature vector) [2]	93
Faster R-CNN (VGG16) [3]	73.2
SSD300 [3]	74.3
Caffe Model [4]	76

The next part of the system is the Helmet Detection which is trained in such a way that it catches all the violators who fail to abide by the road safety rules. The below image shows a properly labelled output. The table below the image, discusses the accuracy of various algorithms applied for helmet detection.

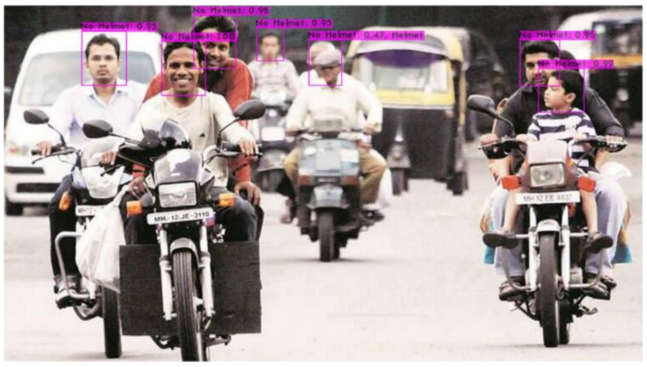


Fig-5: Helmet Detection Output
Table -2: Accuracy of methods in Helmet Detection

Algorithm	Accuracy (in %)
YOLOv3 (proposed system)	90.84
YOLO [1]	81
CNN [2]	85
InceptionV3 [4]	81
CNN based MTL [10]	80.7

Mask Detection is the most important part of the system as it includes a new functionality which is an advantage when it comes to the pandemic situation. When the same image was provided as input, perfect output was obtained and all the people without masks were labelled. The image below shows the output achieved.

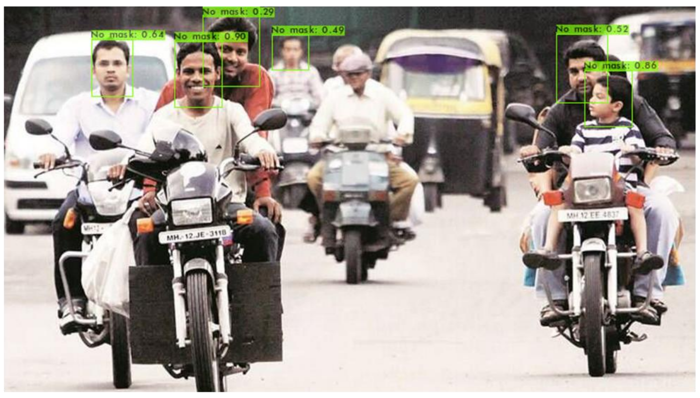


Fig-6: Mask Detection Output
Table 3 Accuracy of methods in Mask Detection

Algorithm	Accuracy (in %)

YOLOv3	99.37
CNN [5]	98.7
Sequential CNN [6]	95.77 / 94.58
SVM and MobileNetV2 [8]	97.1
DNN based on SSD [9]	97.25

Number Plate Detection makes use of the Canny Edge Detection Model which detects all the edges accurately and crops out only the part of the image that contains a number plate. Also, the image of the number plate can be worked upon to retrieve the number plate in a text format. The first image is the input provided to the module, while the image after that is the output achieved.

Fig-7: Original Input

Fig-8: Number Plate Output

7. CONCLUSIONS

A system has been developed for bike riders who violate laws like not wearing a helmet and mask. This project helps in imposing the rules like wearing a helmet and mask for a bike rider in a stricter manner as it is difficult for a traffic policeman to keep record of all the bike riders who violate such laws as the bike riders speeds up and runs away. In such scenarios, this system helps in imposing more strictness in the rule of wearing helmet and mask as all those bike riders who violate such laws can be detected with the help of their number plate and they can't run away from this system. This helps in reducing any kind of possibility of facing an injury for not wearing a helmet or coming in contact with any kind of virus for not wearing a mask. This system mainly consists of four parts - Detection of Bike, Detection of Helmet, Detection of Mask and Detection of Number Plate of those bike riders who don't wear helmet or mask. Firstly, the system checks if the vehicle in the image is a bike or not using YOLOv3. Then the detection of helmet and mask on the bike rider is checked using YOLOv3. Those bike riders who violate laws like not wearing a helmet or mask then the number plate of those bike rider is extracted using canny edge detector. The accuracy obtained for detecting a bike is 95.92% and the accuracy obtained for detecting a helmet is 90.84% and the accuracy for detecting a mask is 99.36%. The accuracy of the system can be increased further by increasing the training dataset and training the algorithm for more time.

ACKNOWLEDGEMENT

First of all, we would like to thank Prof. Nileema Pathak, who has guided and supported us in this endeavour. Her suggestions have helped us immensely to successfully complete the implementation of this system. We have benefited a lot from her immense knowledge. Her suggestions and valuable inputs have helped us at each step. Further, we would like to thank the whole IT Department who has given us this opportunity. Last but not the least, we would like to thank all the people who have constantly supported us.

REFERENCES

[1] F. A. Khan, N. Nagori and A. Naik, "Helmet and Number Plate detection of Motorcyclists using Deep Learning and Advanced Machine Vision Techniques," 2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA), Coimbatore, India, 2020, pp. 714-717, doi: 10.1109/ICIRCA48905.2020.9183287.

[2] Emy Barnabas, Amritha B.J. "Helmet Detection and License Plate Recognition Using CNN", 2019 IOSR Journal of Engineering (IOSRJEN), vol. 09, no. 06, pp. 21-24.

[3] K G Shreyas Dixit, Mahima Girish Chadaga, Sinchana S Savalgimath, G Ragavendra Rakshith, Naveen Kumar M R, "Evaluation and Evolution of Object Detection Techniques YOLO and R-CNN", 2019, vol. 8, pp 824-82, doi: 10.35940/ijrte.B1154.0782S319

[4] C. A. Rohith, S. A. Nair, P. S. Nair, S. Alphonsa and N. P. John, "An Efficient Helmet Detection for MVD using Deep learning," 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 282-286, doi: 10.1109/ICOEI.2019.8862543.

[5] M. M. Rahman, M. M. H. Manik, M. M. Islam, S. Mahmud and J. -H. Kim, "An Automated System to Limit COVID-19

Using Facial Mask Detection in Smart City Network," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, 2020, pp. 1-5, doi: 10.1109/IEMTRONICS51293.2020.9216386.

[6] A. Das, M. Wasif Ansari and R. Basak, "Covid-19 Face Mask Detection Using TensorFlow, Keras and OpenCV," 2020 IEEE 17th India Council International Conference (INDICON), New Delhi, India, 2020, pp. 1-5, doi: 10.1109/INDICON49873.2020.9342585.

[7] A. P. Thombare and S. B. Bagal, "A distributed canny edge detector: Comparative approach," 2015 International Conference on Information Processing (ICIP), Pune, India, 2015, pp. 312-316, doi: 10.1109/INFOP.2015.7489399.

[8] A. Oumina, N. El Makhfi and M. Hamdi, "Control The COVID-19 Pandemic: Face Mask Detection Using Transfer Learning," 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS), Kenitra, Morocco, 2020, pp. 1-5, doi: 10.1109/ICECOCS50124.2020.9314511.

[9] I. Buciu, "Color quotient-based mask detection," 2020 International Symposium on Electronics and Telecommunications (ISETC), Timisoara, Romania, 2020, pp. 1-4, doi: 10.1109/ISETC50328.2020.9301079.

[10] H. Lin, J. D. Deng, D. Albers and F. W. Siebert, "Helmet Use Detection of Tracked Motorcycles Using CNN-Based Multi-Task Learning," in IEEE Access, vol. 8, pp. 162073-162084, 2020, doi: 10.1109/ACCESS.2020.3021357.

ACKNOWLEDGEMENT

Acknowledgement

We would like to express our sincere thanks to our guide Prof. Nileema Pathak for taking time from her busy schedule to provide us with great deal of help, support and encourage us to work diligently at every aspect of our project. Her views have always been equitable striking a perfect balance between encouragement and constructive criticism. Her constructive tips and suggestions helped us to successfully do the project. We have benefited a lot from her immense knowledge and experience.

We express our sincere gratitude and thankfulness to Dr. S.P. Kallurkar Sir, Principal (ACE), Ms. Deepali Maste H.O.D., Department of Information Technology Engineering (ACE) and all Staff Members of Department Information Technology of Engineering (ACE) who have guided, motivated and supported us for our B.E. Project.

We are thankful to the Project Coordinator, Department of Information Technology (ACE), Prof. Nileema Pathak Ma'am, for her valuable suggestions and advice. Our project at various stages has entailed us to seek help from a variety of individuals, we would like to thank each one of them for their forbiddance and guidance. Finally, we would like to thank each one of you, for constantly supporting and encouraging our efforts.