Video Based Vehicle Detection and Tracking using Image Processing and Deep Learning

A thesis

Submitted in partial fulfillment of the requirements for the Degree of

Bachelor of Science in Computer Science and Engineering

Submitted by

Sabbir Ahmed	**170104011**
Lamia Anjum	**170104023**
Farhana Akter Tumpa	**170104042**
Sabiha Benta Sayed Badhon	**170104051**

Supervised by

Dr. -Ing. Nusrat Jahan Lisa

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Dhaka, Bangladesh

December 26,2021

CANDIDATES' DECLARATION

We, hereby, declare that the thesispresented in this report is the outcome of the investigation performed by us under the supervision of Dr. -Ing. Nusrat Jahan Lisa, Assistant Professor, Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh. The work was spread over two final year courses, CSE4100: Project and Thesis I and CSE4250: Project and Thesis II, in accordance with the course curriculum of the Department for the Bachelor of Science in Computer Science and Engineering program.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Sabbir Ahmed
170104011

Lamia Anjum
170104023

Farhana Akter Tumpa
170104042

Sabiha Benta Sayed Badhon
170104051

CERTIFICATION

This thesis titled, **"Video Based Vehicle Detection and Tracking using Image Processing and Deep Learning"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in December 26,2021.

Group Members:

Sabbir Ahmed	**170104011**
Lamia Anjum	**170104023**
Farhana Akter Tumpa	**170104042**
Sabiha Benta Sayed Badhon	**170104051**

Dr. -Ing. Nusrat Jahan Lisa
Assistant Professor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Prof. Dr.Mohammad Shafiul Alam
Professor & Head
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

ACKNOWLEDGEMENT

We would like to acknowledge and express our sincere gratitude to our supervisor Dr. -Ing. Nusrat Jahan Lisa Ma'am for her guidance, motivation and enthusiam. Her contribution helped us in every step of the way, for which we are immensely grateful.

We would like to thank the Almighty and our parents for making this possible.

Dhaka
December
26,2021

Sabbir Ahmed

Lamia Anjum

Farhana Akter Tumpa

Sabiha Benta Sayed Badhon

ABSTRACT

In the subject of highway management, intelligent vehicle system and traffic management are becoming increasingly crucial in today's time. As various vehicles are seen on the street daily, the number of these vehicles is increasing at a high rate. With this huge amount of vehicles, detection and tracking become a difficult job especially in developing and underdeveloped countries. With this thought, detecting and tracking vehicles from video frames is discussed and addressed in this work. In the realm of object detecting and tracking, the deep learning method has been widely applied. This research work proposes video-based vehicle detection and tracking. A new high definition highway vehicle data set containing around 8,000 images extracted from videos with proper annotation is made from the perspective of Bangladesh which provides a complete data foundation for vehicle detection and tracking based on deep learning. For making this data, different types of image processing methods had been used. For detection purposes, the YOLO v5 model, which is the most latest version of the YOLO model is being used; Also the MASK R-CNN model and SSD(Single Shot Detection) have been used for detection purposes. For tracking, we have used the YOLO v5 model, DeepSORT framework and GOTURN method. After detection and tracking, vehicle count and speed estimation are done. For vehicle counting and speed estimation, the YOLO v5 model is used. Classification of the vehicle is done in 10 categories. This information will aid in determining the priority and maximum users of a route and designing traffic patterns that will benefit the most people. Several highway surveillance videos based on different scenes are used to verify the proposed method.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

In today's world, the intelligent transportation system plays a crucial role in the field of traffic management to provide an efficient and reliable transportation system. One of the applications of the intelligent transportation system is to detect and track vehicles accurately. Smart detection and tracking systems require the collection of processed data from respective procedures for the regulation of classifying. In this regard, surveillance cameras have been installed in monitoring and control of traffic in the last few years. Image processing algorithms have been widely developed to monitor the motion of vehicles, humans, or any other objects. Video processing of traffic data obtained through prerecorded video is an instance of applications for advanced cautioning or data extraction for real-time analysis of vehicles. However, the traditional vehicle systems may be declined and not recognized well due to the vehicles being occluded by other vehicles or by background obstacles such as road signals, trees, weather conditions and the performance of these systems depend on a good traffic image analysis approaches to detect, track and classify the vehicles. Here we have studied and analyzed previous works done on this area, identified the research scope, understand the process, methods used, and finally, propose a model that might help us in vehicle detection and tracking with great accuracy.

1.2 Motivation

Vision-based video monitoring systems offer several advantages. Tracking moving vehicles in video streams has been an active area of research in computer vision. The main aim is to detect and recognize moving objects vehicles from real surveillance videos to avoid congestion on highways and parking areas for the prevention of accidents In addition to

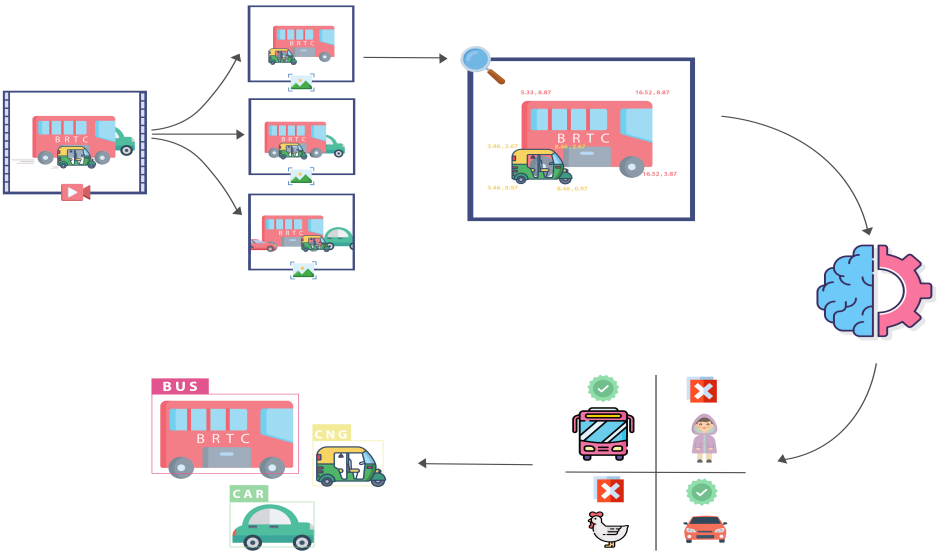

Figure 1.1: General Workflow of Object Detection and tracking

vehicle counts, a much larger set of traffic parameters such as vehicle classifications, lane changes, etc., can be measured. Vehicle classification is important in the computation of the percentages of vehicle classes that use state-aid streets and highways. Even in large metropolitan areas, there is a need for data about vehicle classes that use a particular street. A classification system like the one proposed here can provide important data for a particular design scenario. To keep an eye on all the vehicles in a particular time domain, finding out accidents caused by overtaking, monitoring the speed of the vehicles are the motivation for this study. From Figure 1.1, we can have a general overview of our whole work.

Chapter 2

Background Studies

2.1 What is Image?

An image is an array, or a matrix, of square pixels (picture elements) arranged in columns and rows. An image is composed of picture elements, also known as pixels, each with finite, discrete quantities of numeric representation for its intensity or gray level that is an output from its two-dimensional functions fed as input by its spatial coordinates denoted with x, y on the x-axis and y-axis.(From Figure 2.1)

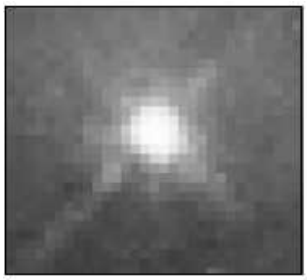

Figure 2.1: An image — picture elements arranged in columns and rows. [1]

In a (8-bit) greyscale image each picture element has an assigned intensity that ranges from 0 to 255. A grey scale image is what people normally call a black and white image, but the name emphasizes that such an image will also include many shades of grey. Each pixel has a value from 0 (black) to 255 (white). The possible range of the pixel values depend on the colour depth of the image, here 8 bit = 256 tones or greyscales. (From Figure 2.2) A true-colour image assembled from three greyscale images coloured red, green and blue. Such an image may contain up to 16 million different colours.(From Figure 2.3)

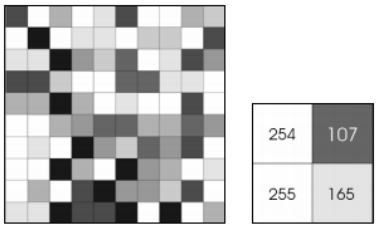

Figure 2.2: Greyscale image. [1]

Figure 2.3: A true-colour image [1]

Some greyscale images have more greyscales, for instance 16 bit = 65536 greyscales. In principle three greyscale images can be combined to form an image with 281,474,976,710,656 greyscales. There are two general groups of 'images': vector graphics (or line art) and bitmaps (pixel-based or 'images'). Some of the most common file formats are:

- **GIF:** An 8-bit (256 colour), non-destructively compressed bitmap format. Mostly used for web. Has several sub-standards one of which is the animated GIF.
- **JPEG:** A very efficient (i.e. much information per byte) destructively compressed 24 bit (16 million colours) bitmap format. Widely used, especially for web and Internet (bandwidth-limited).
- **TIFF:** The standard 24 bit publication bitmap format. Compresses nondestructively with, for instance, Lempel-Ziv-Welch (LZW) compression.
- **PS:** Postscript, a standard vector format. Has numerous sub-standards and can be difficult to transport across platforms and operating systems.
- **PSD:** A dedicated Photoshop format that keeps all the information in an image including all the layers.

2.2 Image Processing

Image processing aims to transform an image into digital form and performs some process on it, to get an enhanced image or take some utilized information from it. It is a method

that develops to convert the image into digital form and perform some operations to obtain specific models or to extract useful information from it. The input of this method is a video section or an image, such as a photograph. The output corresponds to the desired or attention part of the picture. Digital image processing techniques help in manipulation of the digital images by using computers. The three general phases that all types of data have to undergo while using digital technique are pre-processing, enhancement and display, information extraction.(From Figure 2.4)

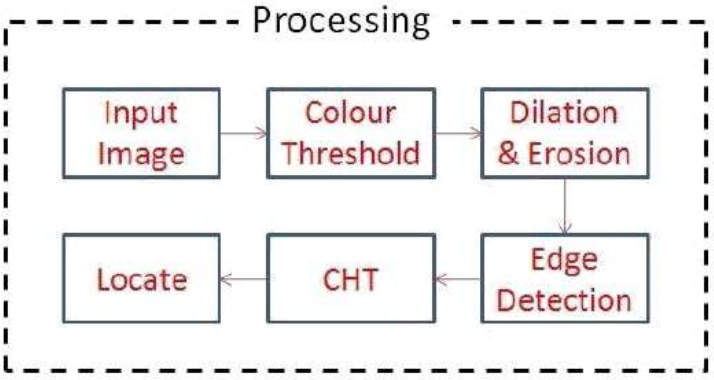

Figure 2.4: Basic Structure of Image Processing. [2]

- **Threshold:** In digital image processing, thresholding is the simplest method of segmenting images. From a grayscale image, thresholding can be used to create binary images.

- **Dilation:** Dilation adds pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries. In the dilation and erosion operations, the state of any given pixel in the output image is determined by applying a rule to the corresponding pixel and its neighbors in the input image. The rule used to process the pixels defines the operation as a dilation or an erosion. For dilation operation, the value of the output pixel is the maximum value of all pixels in the neighborhood. In a binary image, a pixel is set to 1 if any of the neighboring pixels have the value 1. For erosion operation, The value of the output pixel is the minimum value of all pixels in the neighborhood. In a binary image, a pixel is set to 0 if any of the neighboring pixels have the value 0.

- **Edge Detection:** Edge Detection is a method of segmenting an image into regions of discontinuity. It is a widely used technique in digital image processing like pattern recognition,image morphology,feature extraction. Edge detection allows users to observe the features of an image for a significant change in the gray level. This texture indicating the end of one region in the image and the beginning of another. It reduces the amount of data in an image and preserves the structural properties of an image.

- **CHT:** The circle Hough Transform (CHT) is a basic feature extraction technique used in digital image processing for detecting circles in imperfect images.
- **Locate:** Object localization refers to identifying the location of one or more objects in an image and drawing abounding box around their extent. Object detection combines these two tasks and localizes and classifies one or more objects in an image.

2.3 What is Object Detection?

Object detection is a computer vision technique that allows us to identify and locate objects in an image or video (From Figure 2.5).With this kind of identification and localization, object detection can be used to count objects in a scene and determine and track their precise locations, all while accurately labeling them.Specifically, object detection draws bounding boxes around these detected objects, which allow us to locate where objects are in a given scene.

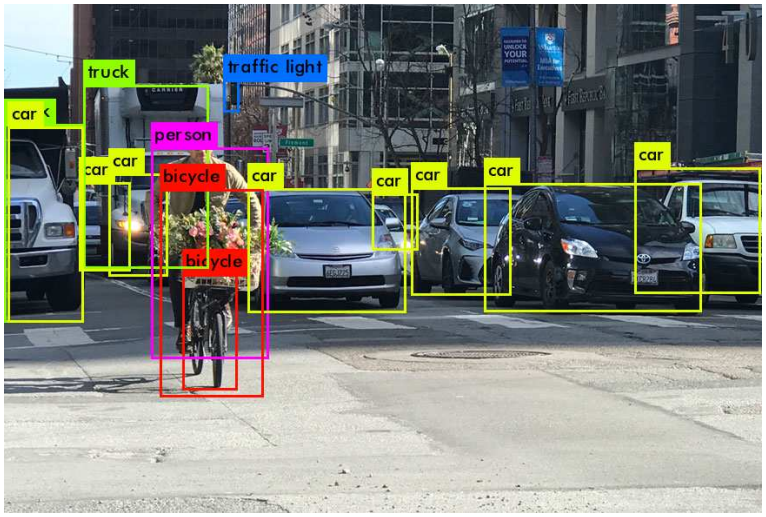

Figure 2.5: Object Detection [3]

2.4 Why is object detection important?

Object detection is linked to other similar computer vision techniques like image recognition and image segmentation, in that it helps us understand and analyze scenes in images or video. But there are important differences. Image recognition only outputs a class label for an identified object, and image segmentation creates a pixel-level understanding of a scene's elements. What separates object detection from these other tasks is its unique ability to locate objects within an image or video.

2.5 How does object detection work?

object detection can be broken down into machine learning-based approaches and deep learning-based approaches. In more traditional ML-based approaches, computer vision techniques are used to look at various features of an image, such as the color histogram or edges, to identify groups of pixels that may belong to an object. These features are then fed into a regression model that predicts the location of the object along with its label. On the other hand, deep learning-based approaches employ convolutional neural networks (CNNs) to perform end-to-end, supervised object detection, in which features don't need to be defined and extracted separately.

2.6 Why it is challenging?

Object detection and tracking is still a great Challenge today. The difficulty level of this problem highly depends on how one defines the object to be detected and tracked.

Illumination Changes: Illumination strongly affects the appearance of background, and cause false positive detections. The background model should take this into consideration.

Dynamic Background: Some parts of the scenery may contain movement (a fountain, movements of clouds, swaying of tree branches, wave of water etc.), but should be regarded as background, according to their relevance. Such movement can be periodical or irregular (e.g., traffic lights, waving trees). Handling such background dynamics is a challenging task.

Occlusion: Occlusion may affect the process of computing the background frame.

Speed of the Moving Objects and Intermittent Object Motion: If the object is moving very slowly, the temporal differencing method will fail to detect the portions of the object preserving uniform region. On the other hand a very fast moving object leaves a trail of ghost region behind it in the detected foreground mask. Objects move then stop for a short while, after which they start moving again causes difficulty in detecting and tracking.

Presence of Shadows: Shadows cast by foreground objects often complicate further processing steps subsequent to background subtraction.

Challenging Weather: Detection of moving object becomes a very difficult job when videos are captured in challenging weather conditions (winter weather conditions, i.e., snow storm, snow on the ground, fog), air turbulence etc.

2.7 Deep Learning in Object Detection

As deep learning methods have become the state-of-the-art approaches to object detection, we have focused on these techniques for our work. Neural networks have become the state-of-the-art methods for object detection. [4] Deep learning-based object detection models typically have two parts. An encoder takes an image as input and runs it through a series of blocks and layers that learn to extract statistical features used to locate and label objects. Outputs from the encoder are then passed to a decoder, which predicts bounding boxes and labels for each object.

The simplest decoder is a pure regressor. The regressor is connected to the output of the encoder and predicts the location and size of each bounding box directly (From Figure 2.6). The output of the model is the X, Y coordinate pair for the object and its extent in the image. Though simple, this type of model is limited. The number of boxes need to specified. If the image has two dogs, but the model was only designed to detect a single object, one will go unlabeled. However, if the number of objects need to predict is known in each image ahead of time, pure regressor-based models may be a good option.

An extension of the regressor approach is a region proposal network. In this decoder, the model proposes regions of an image where it believes an object might reside. The pixels belonging to these regions are then fed into a classification sub network to determine a label. It then runs the pixels containing those regions through a classification network. The benefit of this method is a more accurate, flexible model that can propose arbitrary numbers of regions that may contain a bounding box. The added accuracy, though, comes at the cost of computational efficiency.

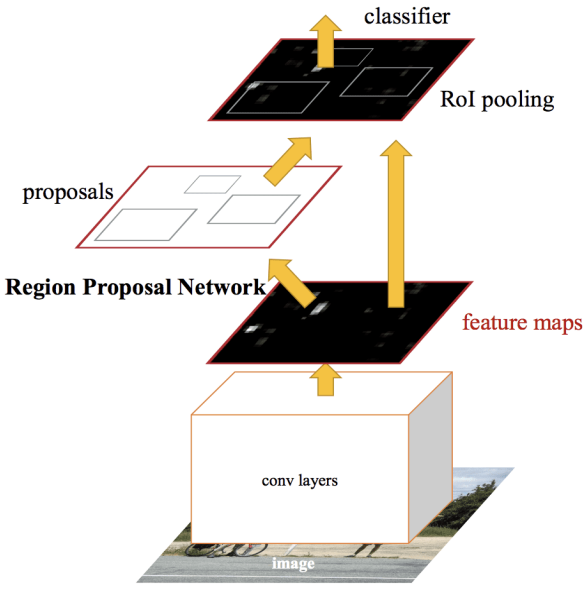

Figure 2.6: Basic Structure of Object Detection [4]

2.8 Deep Learning Algorithms for Object Detection

There are some popular algorithms which is used to carry out object detection include convolutional neural networks (R-CNN, Region-Based Convolutional Neural Networks), Fast R-CNN, and YOLO (You Only Look Once). The R-CNN's are in the R-CNN family, while YOLO is part of the single shot detector family. Some of the popular deep learning algorithms are discussed below:

2.8.1 YOLO

You Only Look Once (YOLO) [5] is one of the popular deep learning algorithms in object detection. The YOLO system predicts bounding boxes using dimension clusters as anchor boxes. Most bounding boxes have certain height-width ratios, so instead of directly predicting a bounding box, YOLO predicts off-sets from a predetermined set of boxes with particular height-width ratios, called anchor boxes. The anchor boxes are generated by clustering the dimensions of the ground truth boxes from the original dataset to find the most common shapes and sizes. The network predicts an objectness score for each bounding box using logistic regression which occurs when the bounding box prior overlaps a ground truth object by more than any other bounding box prior. Unlike sliding window and region proposal-based techniques, YOLO sees the entire image during training and test time so it implicitly encodes contextual information about classes as well as their appearance. YOLO algorithm works using the following three techniques:

- **Residual blocks:** The image is divided into various grids. Each grid has a dimension of S x S. The following image shows how an input image is divided into grids.
- **Bounding box regression:** A bounding box is an outline that highlights an object in an image. Every bounding box in the image consists of the following attributes: Width (bw), Height (bh), Class, Bounding box center (bx,by)
- **Intersection Over Union (IOU)** IOU is a phenomenon in object detection that describes how boxes overlap. YOLO uses IOU to provide an output box that surrounds the objects perfectly. Each grid cell is responsible for predicting the bounding boxes and their confidence scores. The IOU is equal to 1 if the predicted bounding box is the same as the real box. This mechanism eliminates bounding boxes that are not equal to the real box.

This model is implemented as a convolutional neural network. The detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 X 1 convolutional layers reduce the features space from preceding layers. The convolutional layers are

pretrained on the ImageNet classification task at half the resolution (224 X 224 input image) and then double the resolution for detection (From figure 2.6).

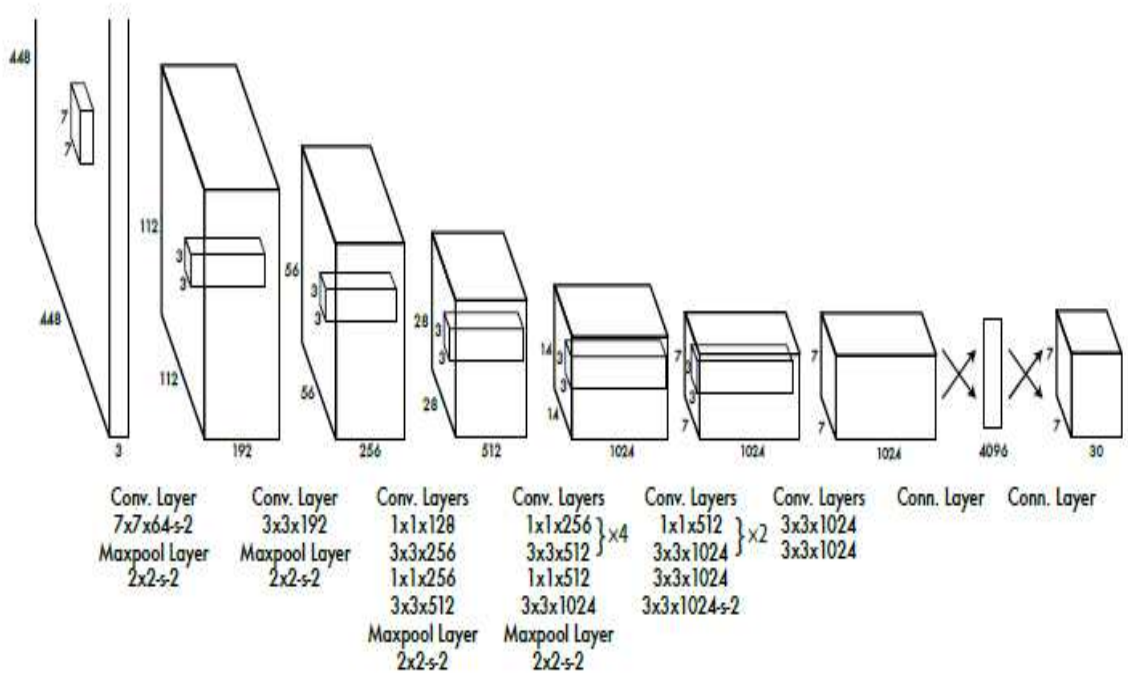

Figure 2.7: The Architecture of YOLO [5]

2.8.2 Mask R-CNN

Region-based convolutional neural networks or regions with CNN features (R-CNNs) [6] are a pioneering approach that applies deep models to object detection. R-CNN models first select several proposed regions from an image and then label their categories and bounding boxes. These labels are created based on predefined classes given to the program. They then use a convolutional neural network to perform forward computation to extract features from each proposed area. The input image is first divided into nearly two thousand region sections and then a convolutional neural network is applied for each region respectively. The size of the regions is calculated and the correct region is inserted into the neural network. There is a set of models in the R-CNN family. Fast R-CNN is one of them. Mask R-CNN is an advancement of Fast R-CNN.

The work flow of the model (From Figure 2.8) is given below:

- Pre-train a CNN network on image classification tasks.
- Fine-tune the RPN (region proposal network) end-to-end for the region proposal task, which is initialized by the pre-train image classifier. Positive samples have IoU $>$ 0.7,

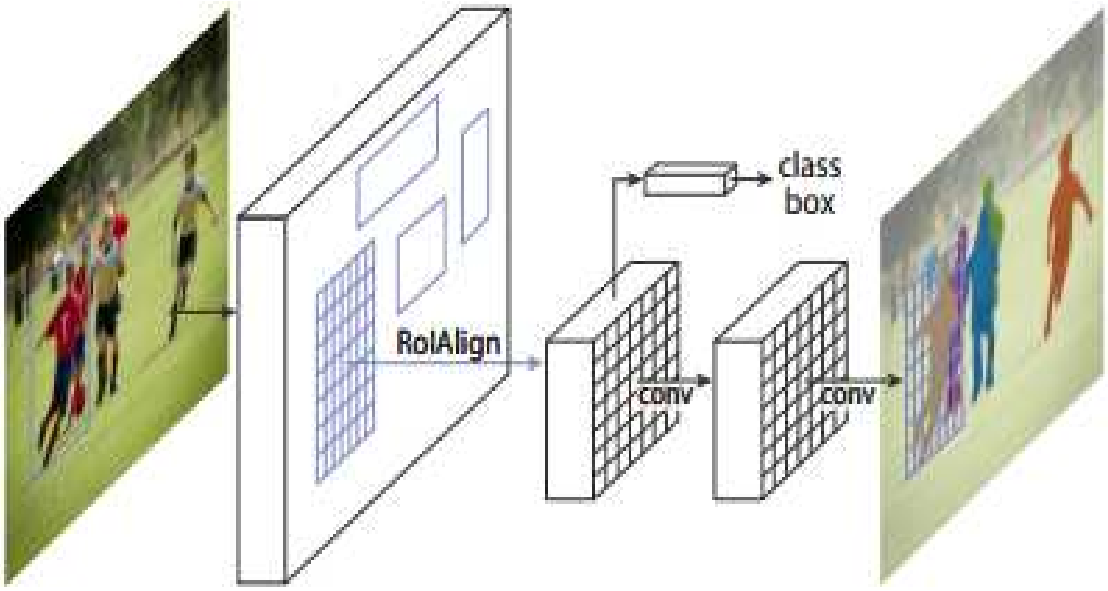

Figure 2.8: The Mask R-CNN framework for instance segmentation [6]

while negative samples have IoU < 0.3.

- Train a Mask R-CNN object detection model using the proposals generated by the current RPN
- A third branch for predicting an object mask is added in parallel with the existing branches for classification and localization.

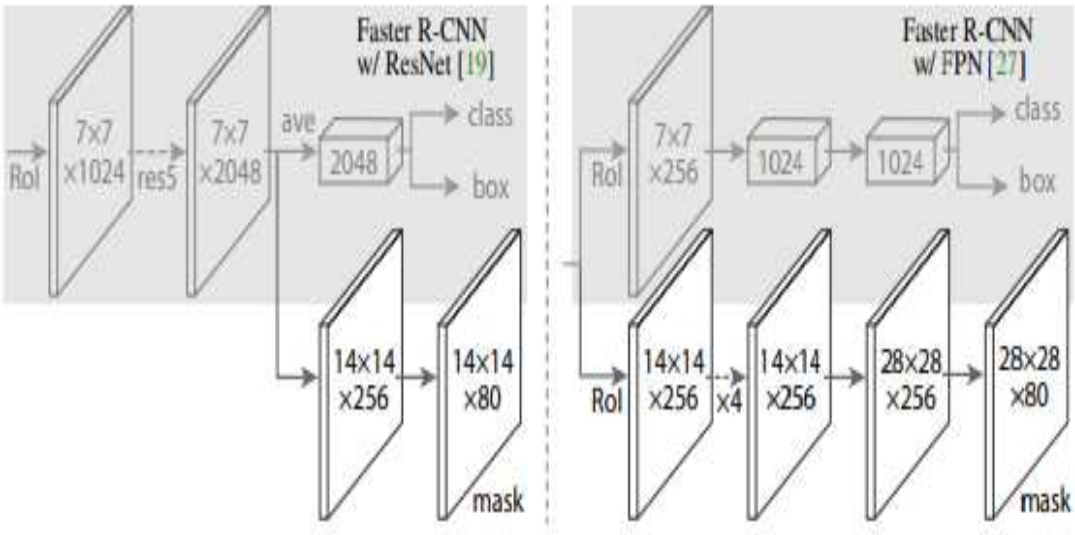

Figure 2.9: Network architecture of Mask R-CNN [6]

The multi-task loss function of Mask R-CNN combines the loss of classification, localization and segmentation mask: L=L(cls)+L(box)+L(mask), where L(cls) is the log loss function over two classes and L(mask) is the average binary cross-entropy loss.

Mask R-CNN is conceptually simple (Figure 2.9) : Faster R-CNN has two outputs for each candidate object, a class label and a bounding-box offset; to this we add a third branch that outputs the object mask. The additional mask output is distinct from the class and box outputs, requiring extraction of much finer spatial layout of an object. Next, we introduce the key elements of Mask R-CNN, including pixel-to-pixel alignment, which is the main missing piece of Fast/Faster R-CNN.

2.8.3 SSD: Single Shot Detection

Single Shot Detector (SSD) [7] is a method for detecting objects in images using a single deep neural network. The SSD approach discretises the output space of bounding boxes into a set of default boxes over different aspect ratios. After discretising, the method scales per feature map location. The Single Shot Detector network combines predictions from multiple feature maps with different resolutions to naturally handle objects of various sizes.

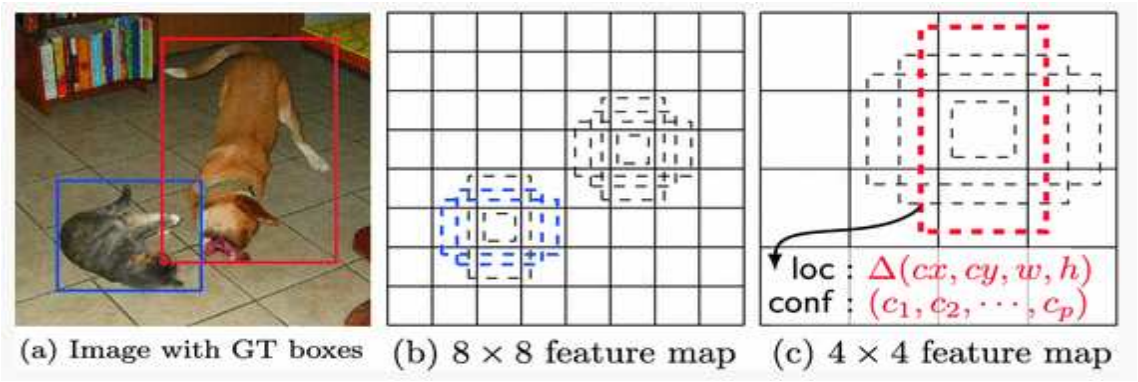

Figure 2.10: The SSD framework [7]

SSD only needs an input image and ground truth boxes for each object during training (Figure 2.10). In a convolutional fashion, we evaluate a small set of default boxes of different aspect ratios at each location in several feature maps with different scales (8×8 and 4×4). For each default box, we predict both the shape offsets and the confidences for all object categories. At training time, we first match these default boxes to the ground truth boxes. For example, we have matched two default boxes with the cat and one with the dog, which are treated as positives and the rest as negatives. The model loss is a weighted sum between localization loss.

The SSD approach is based on a feed-forward convolutional network (Figure 2.11) that produces a fixed-size collection of bounding boxes and scores for the presence of object class instances in those boxes, followed by a non-maximum suppression step to produce the final detections. The key difference between training SSD and training a typical detector that uses region proposals, is that ground truth information needs to be assigned to specific outputs in the fixed set of detector outputs.The loss function and back propagation are

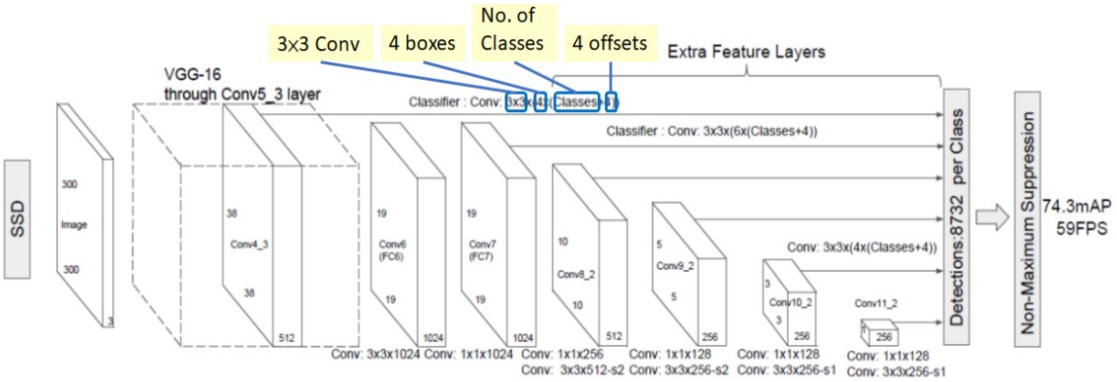

Figure 2.11: The SSD framework [7]

applied end-to-end. Training also involves choosing the set of default boxes and scales for detection as well as the hard negative mining and data augmentation strategies.

2.9 Evaluating Object Detection Models Using Mean Average Precision (mAP)

To evaluate object detection models like R-CNN and YOLO, the mean average precision (mAP) [12] is used. The mAP compares the ground-truth bounding box to the detected box and returns a score. The higher the score, the more accurate the model is in its detections. First of all, we need to know about precision and recall.

- **Precision:**The precision is calculated as the ratio between the number of Positive samples correctly classified to the total number of samples classified as Positive (either correctly or incorrectly). The precision measures the model's accuracy in classifying a sample as positive. When the model makes many incorrect Positive classifications, or few correct Positive classifications, this increases the denominator and makes the precision small. On the other hand, the precision is high when: The model makes many correct Positive classifications (maximize True Positive) and The model makes fewer incorrect Positive classifications (minimize False Positive).

- **Recall:** The recall is calculated as the ratio between the number of Positive samples correctly classified as Positive to the total number of Positive samples. The recall measures the model's ability to detect Positive samples. The higher the recall, the more positive samples detected. The recall cares only about how the positive samples are classified. This is independent of how the negative samples are classified, e.g. for the precision. When the model classifies all the positive samples as Positive, then the recall will be 100 percentage even if all the negative samples were incorrectly classified as Positive.

The decision of whether to use precision or recall depends on the type of problem being solved. If the goal is to detect all the positive samples (without caring whether negative samples would be misclassified as positive), then recall is used. Precision is used if the problem is sensitive to classifying a sample as Positive in general, i.e. including Negative samples that were falsely classified as Positive.

- **Precision-Recall Curve:** From the definition of both the precision and recall, the higher the precision, the more confident the model is when it classifies a sample as Positive. The higher the recall, the more positive samples the model correctly classified as Positive. When a model has high recall but low precision, then the model classifies most of the positive samples correctly but it has many false positives (i.e. classifies many Negative samples as Positive). When a model has high precision but low recall, then the model is accurate when it classifies a sample as Positive but it may classify only some of the positive samples. Due to the importance of both precision and recall, there is a precision-recall curve the shows the tradeoff between the precision and recall values for different thresholds. This curve helps to select the best threshold to maximize both metrics.

- **Average Precision (AP):** The average precision (AP) is a way to summarize the precision-recall curve into a single value representing the average of all precisions. The AP is calculated according to the next equation. Using a loop that goes through all precisions/recalls, the difference between the current and next recalls is calculated and then multiplied by the current precision. In other words, the AP is the weighted sum of precisions at each threshold where the weight is the increase in recall.

- **Mean Average Precision (mAP):** To calculate the mAP, the AP for each class is calculated. The mean of the APs for all classes is the mAP.

2.10 What is Object Tracking?

Object tracking is an application of deep learning where the program takes an initial set of object detection and develops a unique identification for each of the initial detection and then tracks the detected objects as they move around frames in a video. (Figure 2.11)

In other words, object tracking is the task of automatically identifying objects in a video and interpreting them as a set of trajectories with high accuracy. Often, there's an indication around the object being tracked, for example, a surrounding square that follows the object, showing the user where the object is on the screen.

The basic idea behind tracking objects in digital video is to derive object or even camera motion information. Based on the results derived for a video frame, we can predict (or es-

timate) rather than detect the object's position and/or orientation. This is in contrast with frame-based video object detection, where the goal is to find the location of the object of interest in the scene without using its motion information. An obvious advantage of object tracking over object detection is that, in the case of multiple objects, the former can often provide automatic object labeling as they move over time. Additionally, one of the motivations behind object tracking is that object detection is frequently computationally slow or prone to detection errors. Even in the case, that object detection is the final goal, tracking can significantly reduce the search region within a frame and, hence, the computations required.

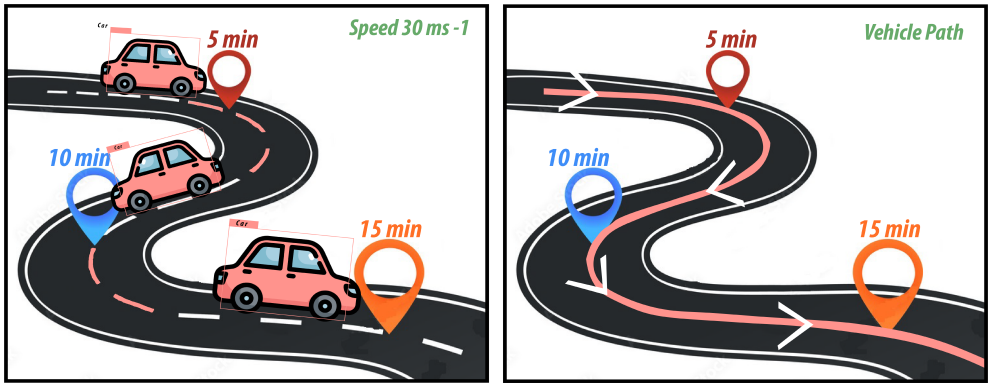

Figure 2.12: Object tracking

2.11 Why Object Tracking is challenging ?

There are some factors which makes object tracking difficult. The main challenges usually stem from issues in the image that make it difficult for object tracking models to effectively perform detection on the images. Here, we will discuss the few most common issues with the task of tracking objects:

- **Background Distractions:** The backgrounds of inputted images or images used to train object tracking models also impact the accuracy of the model. Busy backgrounds of objects meant to be tracked can make it harder for small objects to be detected.

  With a blurry or single color background, it is easier for an AI system to detect and track objects. Backgrounds that are too busy, have the same color as the object, or that are too cluttered can make it hard to track results for a small object or a lightly colored object.

- **Occlusion:** A common issue with tracking an object in an environment with many moving objects is occlusion. Occlusion can cause the system to lose track of the object

being tracked or after overlapping, the wrong object will be tracked instead. Occlusion often occurs when two or more objects come too close and seemingly merge or combine with each other. Image processing system with object tracking often wrongly track the occluded objects. Sometimes, after occlusion, the system will wrongly identify the initially tracked object as a new object.

- **Multiple Spatial Scales:** Objects meant to be tracked can come in a variety of sizes and aspect ratios. These ratios can confuse the object tracking algorithms into believing objects are scaled larger or smaller than their actual size. The size misconceptions can negatively impact detection or detection speed.

  To combat the issue of varying spatial scales, programmers can implement techniques such as feature maps, anchor boxes, image pyramids, and feature pyramids.

  Anchor Boxes: Anchor boxes are a compilation of bounding boxes that have a specified height and width. The boxes are meant to acquire the scale and aspect ratios of objects of interest. They are chosen based on the average object size of the objects in a given dataset. Anchor boxes allow various types of objects to be detected without having the bounding box coordinates alternated during localization.

  Feature Maps: A feature map is the output image of a layer when a Convolutional Neural Network (CNN) is used to capture the result of applying filters to that input image. Feature maps allow a deeper understanding of the features being detected by a CNN. Single-shot detectors have to take into account the issue of multiple scales because they detect objects with just one pass through a CNN framework. This will occur in a detection decrease for small images. Small images can lose signal during downsampling in the pooling layers, which is when the CNN was trained on a low subset of those smaller images. Even if the number of objects is the same, downsampling can occur because the CNN wasn't able to detect the small images and count them towards the sample size. To prevent this, multiple feature maps can be used to allow single-shot detectors to look for objects within CNN layers – including earlier layers with higher resolution images. Single-shot detectors are still not an ideal option for small object tracking because of the difficulty they experience when detecting small objects.

2.12 Levels of Object Tracking

Object Tracking consists of multiple subtypes because it is such a broad application. Levels of object tracking differ depending on the number of objects being tracked. Object tracking levels can be classified into two categories: (1) Single Object Tracking and (2) Multiple Object Tracking (MOT)

2.12.1 Single Object Tracking(SOT):

Single Object Tracking creates bounding boxes that are given to the tracker based on the first frame of the input image.

SOT implies that one singular object is tracked, even in environments involving other objects. Single Object Trackers are meant to focus on one given object rather than multiple. The object of interest is determined in the first frame, which is where the object to be tracked is initialized for the first time. The tracker is then tasked with locating that unique target in all other given frames.SOT falls under the detection-free tracking category, which means that it requires manual initialization of a fixed number of objects in the first frame. These objects are then localized in consequent frames. A drawback of detection-free tracking is that it cannot deal with scenarios where new objects appear in the middle frames. SOT models should be able to track any given object.

2.12.2 Multiple Object Tracking (MOT):

Multiple object tracking is defined as the problem of automatically identifying multiple objects in a video and representing them as a set of trajectories with high accuracy.

Hence, multi-object tracking aims to track more than one object in digital images. It is also called multi-target tracking, as it attempts to analyze videos to identify objects ("targets") that belong to more than one predetermined class.Multiple object tracking is of great importance in autonomous driving, where it is used to detect and predict the behavior of pedestrians or other vehicles. Multiple object tracking often has little to no prior training regarding the appearance and number of targets. Bounding boxes are identified using their height, width, coordinates, and other parameters.

2.13 Object Tracking Algorithms:

Most multiple object tracking algorithms [13] incorporate an approach called tracking-by-detection. The tracking-by-detection method involves an independent detector that is applied to all image frames to obtain likely detections, and then a tracker, which is run on the set of detections.The algorithms can be classified into two methods including batch method and online method.

Batch tracking algorithms use information from future video frames when deducing the identity of an object in a certain frame. Batch tracking algorithms use non-local information regarding the object. This methodology results in a better quality of tracking.While batch

tracking algorithms access future frames, online tracking algorithms only use present and past information to come to conclusions regarding a certain frame. They perform worse than batch methods because of the limitation of online methods staying constrained to the present frame. They perform better in real-time problems requiring the tracking of objects.

Now some popular object tracking algorithms are going to be discussed below:

2.13.1 OpenCV Object Tracking

OpenCV object tracking is a popular method because OpenCV has so many algorithms built-in that are specifically optimized for the needs and objectives of object or motion tracking.

Specific Open CV object trackers include the BOOSTING, MIL, KCF, CSRT, MedianFlow, TLD, MOSSE, and GOTURN trackers. Each of these trackers is best for different goals. For example, CSRT is best when the user requires a higher object tracking accuracy and can tolerate slower FPS throughput.

The selection of an OpenCV object tracking algorithm depends on the advantages and disadvantages of that specific tracker and the benefits:

- **BOOSTING Tracker:** This tracker is based on the same algorithm used to power the machine learning behind Haar cascades (AdaBoost), but like Haar cascades, is over a decade old. This tracker is slow and doesn't work very well. Interesting only for legacy reasons and comparing other algorithms. (minimum OpenCV 3.0.0)
- **MIL Tracker:** It has better accuracy than BOOSTING tracker but does a poor job of reporting failure. (minimum OpenCV 3.0.0)
- **KCF Tracker:** The KCF tracker is not as accurate compared to the CSRT but provides comparably higher FPS. It is also faster than BOOSTING and MIL. Similar to MIL and KCF, does not handle full occlusion well. (minimum OpenCV 3.1.0)
- **CSRT Tracker:** It includes channel and spatial reliability. It tends to be more accurate than KCF but slightly slower. (minimum OpenCV 3.4.2)
- **MedianFlow Tracker:** It does a nice job reporting failures; however, if there is too large of a jump in motion, such as fast moving objects, or objects that change quickly in their appearance, the model will fail. (minimum OpenCV 3.0.0)
- **Mosse Tracker:** The MOSSE tracker is very fast, but its accuracy is even lower than tracking with KCF. Still, when it comes to looking for the fastest object tracking OpenCV method, MOSSE is a good choice. (minimum OpenCV 3.4.1)

- **GOTURN Tracker:** The only deep learning-based object detector included in OpenCV. It requires additional model files to run. The original implementation of GOTURN is in Caffe, but it has been ported to the OpenCV Tracking API. (minimum OpenCV 3.2.0)

2.13.2 DeepSORT

DeepSORT is a good object tracking algorithm choice, and it is one of the most widely used object tracking frameworks. It is an extension to SORT (Simple Real time Tracker). Basically we do not track just distance and velocity, but also appearance. DeepSORT enables us to add this feature by computing deep features for each bounding box and factoring in the tracking algorithm based on deep feature similarity. The model's goal is to simply track a specific object from a specified image crop. To do so, they employ a two-frame CNN architecture that accurately regresses onto the object using both the current and prior frames. DeepSORT is one of the quickest trackers, focusing on simple, effective algorithms and taking a realistic approach to multiple item tracking. DeepSORT accelerates tracker speed by using Kalman Filtering and the Hungarian Algorithm. It produced 16 FPS on average while still maintaining good accuracy, definitely making it a solid choice for multiple object detection and tracking. The model has four stages – detection stage, feature extraction/motion prediction stage, affinity stage, and association stage.

- **Detection stage:** In this stage, objects are detected from each input frame and construct bounding boxes around them.
- **Feature extraction/motion prediction stage:** A feature extractor extracts feature vectors of target objects and a motion predictor forecasts each tracked target's upcoming position.
- **Affinity stage:** In this stage, a similarity/distance score is computed between pairs of detections and/or tracks using feature vectors and motion predictions.
- **Association stage:** In this stage, the similarity/distance measures are used to associate detections and tracks belonging to the same target by assigning the same ID to detections that identify the same target.

2.13.3 Object Tracking MATLAB

MATLAB is a numeric computing platform, which makes it different in its implementation compared to DeepSORT and OpenCV, but it is nevertheless a fine choice for visual tracking tasks.The Computer Vision Toolbox in MATLAB provides video tracking algorithms, such as

continuously adaptive mean shift (CAMShift) and Kanade-Lucas-Tomasi (KLT) for tracking a single object or for use as building blocks in a more complex tracking system.

2.13.4 MDNet

MDNet is a fast and accurate, CNN-based visual tracking algorithm inspired by the R-CNN object detection network. It functions by sampling candidate regions and passing them through a CNN. The CNN is typically pre-trained on a vast dataset and refined at the first frame in an input video. Therefore, MDNet is most useful for real-time object tracking use cases. However, while it suffers from high computational complexity in terms of speed and space, it still is an accurate option. The computation-heavy aspects of MDNet can be minimized by performing RoI (Region of Interest) Pooling, however, which is a relatively effective way of avoiding repetitive observations and accelerating inference.

Chapter 3

Literature Review

We have studied some existing research paper on this topic. In this section we will summarize and review them

3.1 A Survey On Vehicle Detection And Tracking Algorithms In Real Time Video Surveillance

Objective: In this paper [14], the detailed overview of object motion detection, classification, and tracking algorithms are presented and also their strengths and weakness of the various algorithms are discussed.

Summery: The authors classified the whole concept of vehicle detection and tracking into four steps mentioning: detection, classification, tracking and recognition.

In this review paper some algorithms are mentioned for moving object detection like background subtraction techniques, optical flow methods, statistical methods, frame differencing, temporal differencing, shadow and light change detection which are the subsequently used techniques. The background subtraction technique finds the moving regions in images by subtracting the initial image of pixels from a referenced background image which is formed by averaging images. In the statistical method, the characteristic of individual pixel or group of pixels are considered to construct the background frame and statistics of background can automatically update during processing. In temporal differencing, a new image is obtained when the difference between the previous frame and the current frame is greater than the threshold value. In background subtraction and shadow detection method, pixels are represented by a color model that separates brightness from the chromaticity component. The two methods by which the shadows can be detected are statistic and video based method.

Here mainly five types of tracking methods are mentioned: Region- Based Tracking Methods, Contour Tracking Methods, 3D Model-Based Tracking Methods, Feature-Based Tracking Methods, Color and Pattern-Based Methods. In region based tracking methods, the particular region of the moving objects like vehicles (blobs) are tracked for locating the vehicles. These regions are segmented by subtracting the current image and previous image. Contour tracking method consists of two techniques: color contour based matching and gradient based matching. The feature based tracking method uses a feature descriptor of SURF (Speeded Up Robust Features) for a large region of feature sets to classify the vehicles in smart surveillance videos. In color and pattern based methods, there are three important main levels in this system that used are 1D shape patterns, tracking level, 2D pattern verification

3.2 Vehicle Detection and Tracking Techniques: A Concise Review

Overview: In this review paper [15], a concise overview of image processing methods and analysis tools which are used in building the applications that involved developing traffic surveillance systems is represented.

Summery: Here the traffic image analysis comprises of three parts: (1) Motion Vehicle detection and Segmentation Approaches (2) Camera Calibration Approaches and (3) Vehicle Tracking Approaches. The detection of moving objects consists of three main approaches to detect and segment the vehicle, as mentioned below: 1. Background Subtraction Methods. 2. Feature Based Methods. 3. Frame Differencing and Motion Based methods.

Background subtraction process is extracting moving foreground objects (input image) from stored background image (static image) or generated background frame form image series (video). After that, the extracted information (moving objects) is resulted as the threshold of image differencing. In this review paper several background subtracting methods have been mentioned with their references. Also a new method has proposed for vehicle detection based on shadows underneath vehicles information. This method extracts the size features of vehicles from information that gathered from the distance between ends of front and rear tires for underneath shadow of vehicles to distinguish the existence of vehicles on the lanes. Sub-features like the edges and corners of vehicles, the set of these features from the movement between the subsequent frames supports the occlusion handling between the overlapping vehicles and compared with background subtraction method.

The frame differencing is the process of subtracting two subsequent frames in image series to segment the foreground object (moving object) from the background frame image. An

interframe and tracking levels are suggested framework to recognize and manipulate occlusion vehicles. Camera calibration is a vital procedure for well video-based surveillance system. Here it is also mentioned that a new automatic method for segmenting and tracking vehicles applied on a video taken by camera at low angle level relatively to the ground on highway road.

3.3 Region-based Convolutional Networks for Accurate Object Detection and Segmentation

Overview: In this paper [8], a simple and scalable detection algorithm is proposed that improves mean average precision.

Summery: The region-based approach led to a big wave of object detection research with its two-stage framework, i.e, region proposal stage, and region classification and refinement stage.

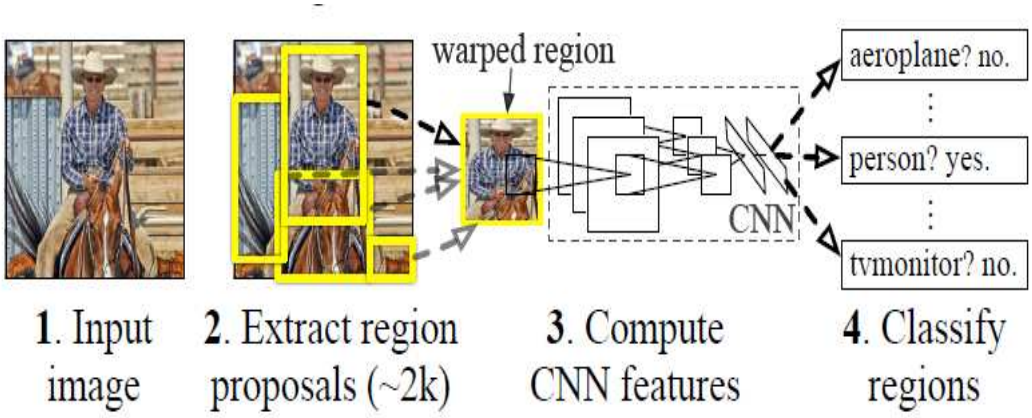

Figure 3.1: R-CNN: Region-based Convolutional Network. [8]

In the above diagram (Figure 3.1), R-CNN first extracts potential regions of interest from an input image by using a technique called selective search. Next, R-CNN warps these region proposals into fixed-size images with some paddings, and feed these images into the second stage of the network for more fine-grained recognition. Unlike those old methods using selective search, R-CNN replaced HOG with a CNN to extract features from all region proposals in its second stage. Region proposal from selective search highly depends on the similarity assumption, so it can only provide a rough estimate of location. To further improve localization accuracy, R-CNN borrowed an idea from “Deep Neural Networks for Object Detection” (aka DetectorNet), and introduced an additional bounding box regression to predict the center coordinates, width and height of a box. This regressor is widely used in the future object detectors. However, a two-stage detector like R-CNN suffers from two big

issues: 1) It's not fully convolutional because selective search is not E2E trainable. 2) region proposal stage is usually very slow compared with other one-stage detectors like OverFeat, and running on each region proposal separately makes it even slower.

3.4 You Only Look Once: Unified, Real-Time Object Detection

Overview: In this paper [5], a approach YOLO is proposed for object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. YOLO learns very general representations of objects. It outperforms other detection methods, including DPM and R-CNN.

Summery: Unlike R-CNN, YOLO decided to tackle region proposal and region classification together in the same CNN. In other words, it treats object detection as a regression problem, instead of a classification problem relying on region proposals.

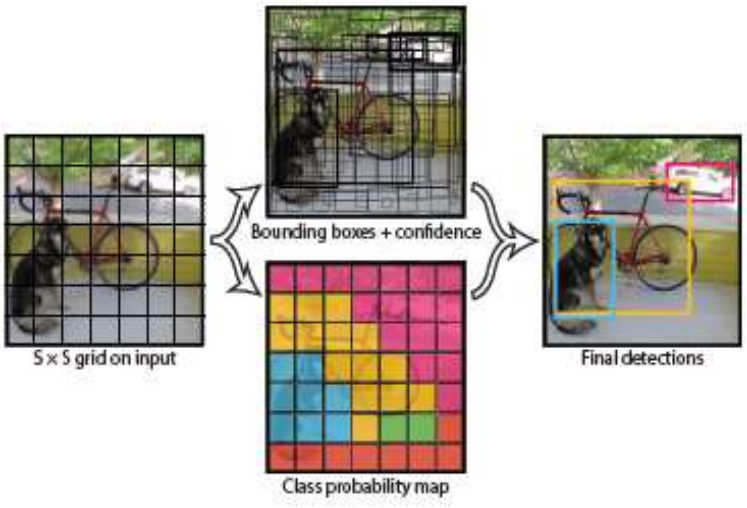

Figure 3.2: The YOLO Detection System. [5]

The general idea is to split the input into an SxS grid and having each cell directly regress the bounding box location and the confidence score if the object center falls into that cell (Figure 3.2). During training, the regressor with the highest IOU will be assigned to compare with the ground-truth label, so regressors at the same location will learn to handle different scales over time. In the meantime, each cell will also predict C class probabilities, conditioned on the grid cell containing an object. This approach is later described as dense predictions because YOLO tried to predict classes and bounding boxes for all possible locations in an image.

3.5 High-Speed Tracking-by-Detection Without Using Image Information

Overview: In this paper [9], a simple but successful tracker algorithm IoU-Tracker has been showed which can easily run at 100K fps performs better than other traditional vehicle tracking approach.

Summery: IoU (Intersection over Union) is an evaluation metric used to measure the accuracy of an object detector on a particular dataset (Figure 3.3). In order to apply Intersection over Union to evaluate an (arbitrary) object detector we need: The ground-truth bounding boxes (i.e. the hand labeled bounding boxes from the testing set that specify where in the image the object is) and secondly, The predicted bounding boxes from the model. IoU is simply a ratio of area of overlap between the predicted bounding box and the ground-truth bounding box (numerator) and the area of union, or more simply, the area encompassed by both the predicted bounding box and the ground-truth bounding box (denominator).

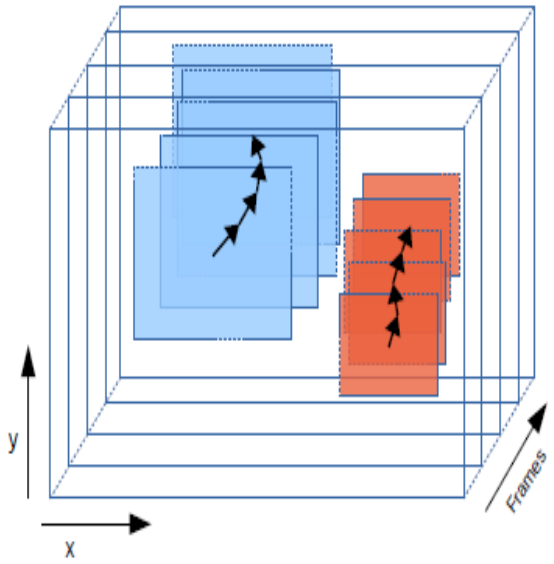

Figure 3.3: Basic principle of the IOU Tracker: with high accuracy detections at high frame rates. [9]

In this paper, the method is based on the assumption that the detector produces a detection per frame for every object to be tracked, i.e. there are none or only few gaps in the detections. Furthermore, it is assumed that detections of an object in consecutive frames have an unmistakably high overlap IOU (intersection-over-union) which is commonly the case when using sufficiently high frame rates.

3.6 Extending IOU Based Multi-Object Tracking by Visual Information

Overview: In this paper [10], a visual single-object tracker is incorporated into the IoU tracking scheme to increase the robustness against missing detections.

Summery: A major drawback of the simple IoU approach is its requirement for a high recall of the underlying detector. Every gap caused by a single or few missing detections leads not only to false negatives but also to the termination and restart of the track, causing high rates of fragmentation and ID switches. The idea was to continue each track by a visual tracker if no new detection can be associated and thus fill the gaps between tracks. This reduces the fragmentation of the tracks and number of ID switches. The accuracy and speed of the proposed approach depends primarily on the object detectors and visual single-object trackers performances.

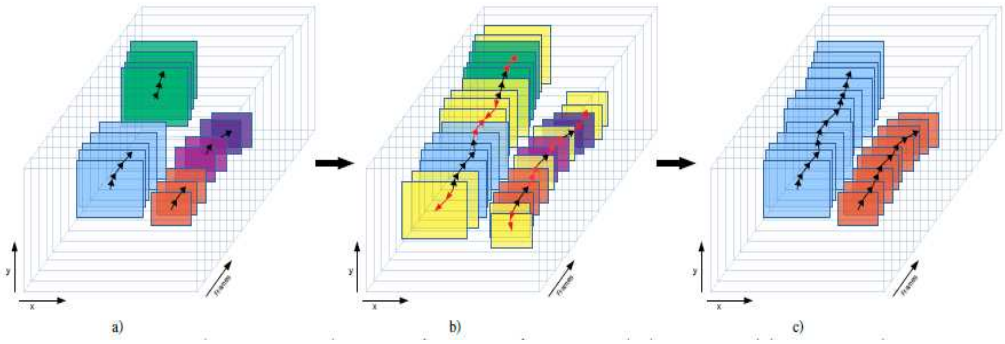

Figure 3.4: Basic principle of the extension for the IOU Tracker: IOU tracks are usually highly fragmented due to missing detections (left). Gaps can be filled by employing a visual tracker to compensate for missing object locations (yellow, middle). The resulting tracks are less fragmented (right). [10]

Visual tracking is performed in two directions(Figure 3.4): First, if no detection satisfies the threshold of simple IoU for a track, the visual tracker is initialized on the last known object position (the detection at the previous frame) and used to track the object for a maximum of frames. If a new detection satisfies the threshold within these frames, visual tracking is stopped and the IoU tracker is continued. Otherwise, the track is terminated. This is usually sufficient to compensate reliably for few missing detections. However, with increasing number of visual tracked frames it becomes more likely that the visual tracker looses the track or jumps over to another object. To limit the number of consecutive frames where the object is only tracked by visual indication, we perform the visual tracking also backwards through the last frames for each new track. Although the visual forward and backward tracking of the object helps to merge discontinued tracks it also adds visually tracked stubs at the beginning and end of each completed track as shown in Figure.

3.7 Efficient vehicle tracking and classification for an automated traffic surveillance system

Overview: In this paper [16], a traffic surveillance system is proposed that works without prior, explicit camera calibration and has the ability to perform surveillance tasks in real time. The authors used the knowledge of camera parameters to detect the pose of a vehicle in the 3D world.

Summary: Here two techniques color contour based matching and gradient based matching is used. A simple recursive learning method is used to model the background. The overall accuracy of the system depends on robust foreground object detection. Background modeling methods can be divided into two types: parametric [3] and non-parametric. The approach involves learning a statistical color model of the background, and process a new frame using the current distribution in order to segment foreground elements. The algorithm has three distinguishable stages: learning, classification and post processing. In learning stage, a background model is established using recursive learning. In the classification stage, the image pixels are classified into foreground and background pixels based on background model. To remove unwanted foreground, ROI template is used. Both of the above mentioned classification algorithms proposed in this work try to match object edges with the 3D wire-frame models. Initially, the edge template is created by detecting the object edges. It is then matched with the 3D wire-frame models of the four vehicle classes. To create a color contour template for the object edge template, initially drawing filled circles of blue color at all the edge pixels is detected. Then, reducing the radius of the circle gradually and repeat the procedure for green, red and finally for black color. While creating a color contour template for a model edge template, the authors used only black color. Matching score is calculated by counting the number of different color pixels present in the matching template. Each black pixel in the matching template represents perfectly matched edge pixel and contributes the highest to the matching score. Each red pixel contributes more than green and blue pixel, but less than black pixel. In gradient based matching algorithm, initially the gradient of the edges in both templates (object edge template and model edge template) is calculated using a 3×3 Prewitt mask. Then matching is done on the basis of gradient magnitude and direction.

3.8 Learning to Track at 100 FPS with Deep Regression Networks

Overview: In this paper [11], a method for training neural networks offline so that they can track novel objects at 100 frames per second during testing is proposed. Generic Object Tracking Using Regression Networks, which is called GOTURN is used to achieve this purpose. GOTURN is the first generic object neural-network tracker that can run at 100 fps. The tracker is far faster than prior approaches for tracking that used neural networks, which are often slow to run and unsuitable for real-time applications.

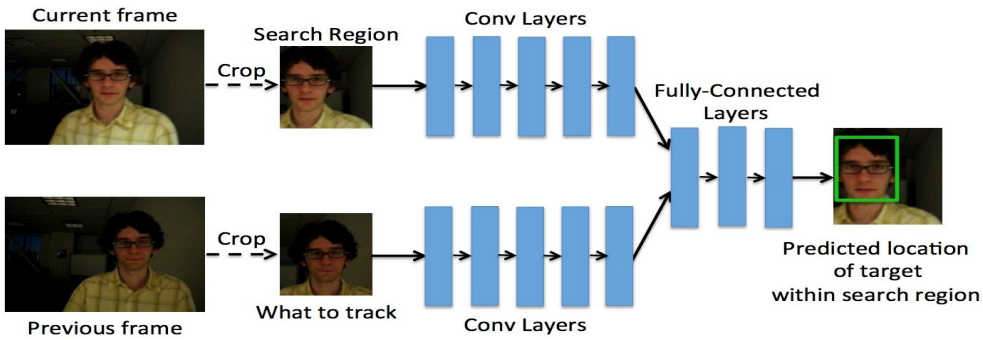

Figure 3.5: GOTURN-architecture. [11]

Summery: GOTURN tracker employs a regression-based technique, which requires only one feed-forward trip through the network to regress directly to the target object's location. There is no need for class-level labeling or information about the types of objects are being tracked. GOTURN introduces a new tracking system in which the relationship between appearance and motion is learned in a general manner offline.

In GOTURN tracking architecture (Figure 3.5), a search region from the current frame and a target from the previous frame is being inputted to the network, then the network learns to compare these crops to find the target object in the current image. The tracker must know where the target object was previously situated to find it in the current frame. As things travel smoothly through space, the object's former location will give a solid indication of where the network should anticipate to find it now. This is accomplished by selecting a search region in our current frame depending on the object's previous position. Using the search region, the current frame is being cropped and fed into the network. These convolutional layer outputs are subsequently fed into a series of fully connected layers. The fully linked layer's job is to compare the target object's features to the features in the current frame to figure out where the target object has moved. The item may have been translated, rotated, lit differently, occluded, or deformed between these frames. The fully connected layer's function is doing a sophisticated feature comparison that is taught through

numerous examples to be robust to these many aspects while outputting the relative motion of the tracked object. In this paper, for each subsequent frame t crop from frames t-1 and t is being inputted. Between these two crops, the target item appears to have been translated and scaled-down, and then it will track the movement of the target object throughout the entire video sequence.

3.9 Real Time Pear Fruit Detection and Counting Using YOLOv4 Models and DeepSORT

Overview: In this paper [17], DeepSORT is a tracking method that includes a useful method for dealing with numerous object tracking issues. It employs appearance data to track objects through occlusions, minimizing the number of identity switches. In this paper, the DeepSORT algorithm was used in conjunction with several state-of-the-art object detectors like YOLO, SSD, and FasterRCNN to evaluate and compare pedestrian tracking performance shortcoming of each detector is discussed.

Summery: In Deep Sort, A classifier is created, then it is being trained on the dataset with higher accuracy, and removal of the final classification layer is done. The feature vector is then categorized using a dense layer. This feature vector is used to represent the object's appearance description.

The network's CNN design can be classified into: A residual network, two convolutional layers, and six residual blocks make up the deep association clock. The dense layer's computed features map has a dimensionality of 128. The unit includes a batch layer and l2 normalization projects. To be compatible with the cosine appearance metric, the hypersphere must be spherical. In this paper, for detecting objects different object detection framework has been applied. The Faster -RCNN model is divided into two parts. A deep convolutional network is employed in the first stage to generate region proposals, which are then classified into different items in the second step. SSD, on the other hand, simply requires a single shot to detect things. As a result, SSD detection is much faster than Faster-RCNN. For prediction, SSD extracts feature maps and applies a 3x3 convolution filter to each cell. However, Faster R-CNN is detecting many false positives Though SSD doesn't have the problem of false positives, but it has a high miss rate. YOLO is faster than other region-based detectors since it only sends the complete image to the CNN once. The image is divided into am x m grid, with bounding boxes and class probabilities created. When compared to region-based detectors, YOLOv2 introduced batch normalization, a lower localization error, and a higher recall. On the contrary, YOLOv3 is well-known for its accuracy. Additionally, YOLOv4, which is an upgrade above YOLOv3 as well as its MAP (mean Average Precision) and FPS (frames per second) have both increased by 10 percentage and 12 percentage, respectively. Dur-

ing the evaluation, all targets that appear during starting point must be found in time, the target position should be as consistent as possible with the real target position, each target should be assigned a unique ID that remains unchanged throughout the sequence is maintained. To have good results, the detector algorithms are trained on the dataset and YoloV4 outperforms among those.

3.10 A Safety Vehicle Detection Mechanism Based on YOLO v5

Overview: In this paper [18], to quantify the accuracy and effectiveness of the detection mechanism, two main detection models, YOLOv3 and YOLOv5, were used. YOLOv5 has a faster average detection speed and is useful for making decisions. To demonstrate the efficacy of the proposed method, a software system was constructed.

Summery: Due to the complexity of the real-world traffic scene, which is influenced by weather, time, vehicle occlusion, drivers' vision, and other factors, traditional vehicle detection algorithms have the drawbacks of slow detection speed and low accuracy, as well as the frequent occurrence of false detection and missing detection, making it difficult to meet actual needs. In this paper, The overall processing network flow of the mechanism is based on the network of Yolov5, which can be divided into 3 parts: backbone, neck, and detection. The input image is preprocessed in the backbone stage, after which mosaic, mix up, color space improvement and horizontal and vertical flipping are used to enhance the data, which can generate more equivalent data to expand the dataset, raise the data size, and lower the model's overfitting. The evolution priority box is generated and mutated simultaneously using the K-means clustering and GA methods. The model's training effect and accuracy will both benefit from the proper priority box setting. The image with the dimensions of 640*640*3 is then processed by the Focus module. It consists of procedures such as slicing and convolution. The slicing procedure is utilized to create a 320 X 320 X 12 feature map, which is subsequently converted to a 320 X 320 X 32 feature map using a 3X3 convolution kernel. Following the Focus, the CBL and Bottleneck CSP basic convolution modules are utilized to eliminate the extract features and gradient information duplication.Yolov5 uses both a top-down and bottom-up feature pyramid structure in the neck stage. The feature pyramid sends strong position features from the bottom to the top, while FPN conveys strong semantic features from top to bottom. The traits of different layers can be increased and used for multi-layer detection when these two structures are combined. Due to the real application scenario in the city, this paper only chooses the 6 most common categories of an object such as bus, truck, motor, car, bike on road for drivers. Yolov5 outperforms Yolov3 in terms of efficiency due to differences in network layers and characteristics. In terms of

object detection for drivers on the road, the efficiency of detection and result is critical as it will provide effective decision-making suggestions even if the weather or environment is unpleasant and noisy. As a result, Yolov5 is used as a mechanism's basic model.

3.11 An improved Yolov5 real-time detection method for small objects captured by UAV

Overview: In this work [19], YoloV5 and four methods are proposed to improve the detection precision of small objects. The model that incorporates all of the new methods not only enhances detection precision but also significantly minimizes detection speed loss. Using VisDrone-2020,model's mAP increased from 12.7 to 37.66 percent and increased the detection speed to 55FPS which outperform the previous state of the art in terms of detection speed and advanced object detection algorithms on drone platforms.

Summery: To increase the small object identification performance of the yolov5s network, the feature fusion structure is enhanced. Large objects dominate traditional datasets, small objects are scarce or irregularly distributed, and a large number of little objects are concentrated in a small area of the photos. The trained model becomes progressively biased toward large objects due to the imbalanced sample distribution structure .In this work, The VisDrone -2020 dataset is used which is dominated by small objects .As the original anchor size is not suitable for the VisDrone dataset so improving the matching probability the object box and anchor KMeans++ clustering algorithm is used to redesign the anchor size. The squeeze-and-excitation (SE) module is a little plug-and-play block that adds an attention mechanism to application. GAP is used as a squeeze operation; then, two fully connected (fc) layers form a bottleneck structure. As the difference between the two cases is small, and the effect integration after the basic block is slightly better so our SELayer is added after the 4th, 6th, and 8th layers of the original backbone network (with a focus layer as the 0th layer). By inserting the difference between the center point distance and the aspect ratio of the prediction box and the gt box as a penalty item, to optimize the bounding box loss using the CIoU loss function is used . On the basis of the yolov5s, in this paper four modifications in model is done they are : changing the size of the anchor, adding a SE module, adopting CIoU loss , Adding a P2 feature level. To evaluate the model five metrics are used. They are - model size, inference speed, precision, recall, and mAP. In comparison to the original yolov5s model, the final trial data show that the mAP of the model that built based on these guiding concepts grew from 12.70 to 37.66 percent.

Chapter 4

Experimental Methodology

We have tried to experiment ourselves with detecting the vehicles from a video using our custom dataset. Below we are presenting our experimental methodology as a flow chart diagram (Figure 4.1). We will later describe each process in details.

4.1 Proposed Methodology

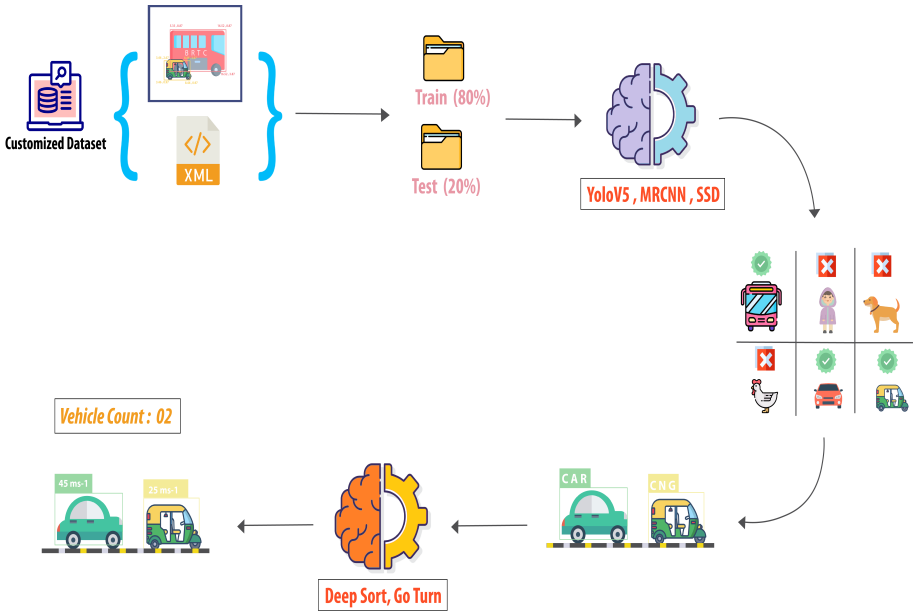

Figure 4.1: Experimental Methodology Diagram.

Object detection and Tracking is a process of detecting and tracking the moving object from a video. An object can be a human, animal, or vehicle. Our method is to be able to detect and track all kinds of vehicles in a given video using some efficient methods and algorithms. So in our proposed methodology firstly, we made our customized dataset by converting the videos into frames with proper annotation. Then we split our customized dataset where we kept 80

percentage of data for training and 20 percentage of data for testing. The detecting module extracts and identifies the desired object, then send the detecting information and object information to the learning and tracking module, respectively. For detection of our vehicles, we used YoloV5, SSD, Mask RCNN. For tracking, it is necessary to find the positions of the object in two consecutive frames to generate the trajectory. For tracking, we used Deep Sort and GOTURN and with that, we have estimated the speed of each vehicle and also counted the number of vehicles. For speed estimation, we have estimated these values manually for the current road to calculate pixels per meter (ppm). We have taken the video frame and calculated the width of the road in pixels. Then converted pixels to meter now following this formula of speed(speed = dmeters X fps X 3.6) we get the values. For counting, we generated a counter and we divided the road into polygons for identifying lanes. Any class object inside the polygon region is boxed and the counter is updated. Thus we are counting, tracking, and determining the speed of the vehicles.

4.2 Dataset

We have made our own custom dataset consists of local vehicles information of our country.

4.2.1 Data Collection

As we are going to detect and track the local vehicles so the dataset needs to be concrete and should have quality attributes. Therefore we collected data by taking video shots from different angles which have proper information regarding the vehicles.

4.2.2 Data Preprocessing

We have recorded videos of Traffics from different roads from which we first generated images from that videos. Images are selected given priority to vehicles in a frame, clear view, and not hazy objects. Images are strictly resized 64 px X 64px. And for training efficiency size is reduced to under 40kb. For keeping track, images are sorted in numeric order.

4.2.3 Data Annotation

More than 8000 frames are annotated with bounding boxes of vehicles are labeled. Our dataset contains videos with large variations in scale, pose, illumination, occlusion and background clutters. We have introduced total 10 classes including:

- Bus
- Bike
- Cycle
- Cng
- Car
- Leguna
- Rickshaw
- Truck
- Van
- Ambulance

Here we are presenting a pictorial diagram (Figure 4.2) that will help to visualize the steps that we have done for creating our custom dataset:

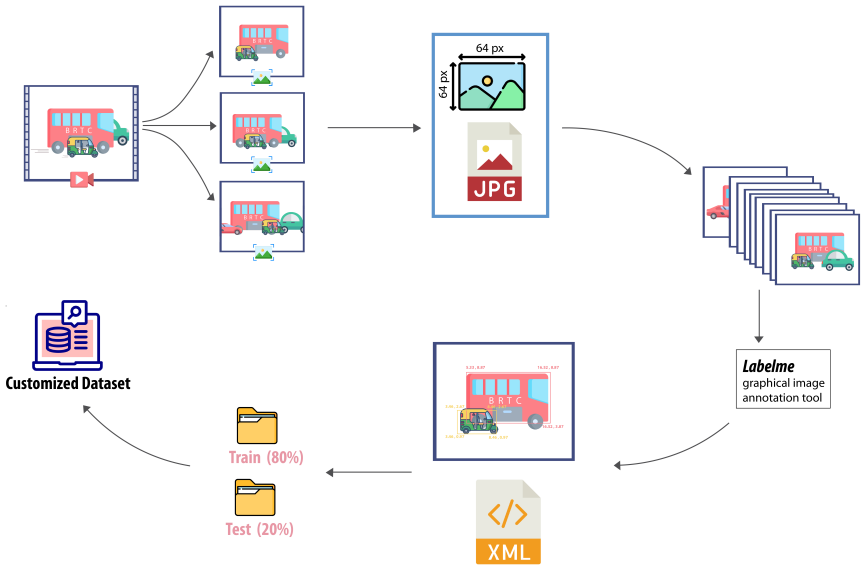

Figure 4.2: Workflow of preparing custom dataset

At first we have converted the videos into frames. The annotation has been done on the frames using MakeSense.AI [20], labelImg [21] and VGG annotator tools. As we have used the models YOLOv5, SSD and Mask R-CNN to validate our dataset, so we had to create the dataset according to the format of each model(Figure 4.3).

In YOLO labeling format, a .txt file with the same name is created for each image file in the same directory. Each .txt file contains the annotations for the corresponding image file, that

Figure 4.3: Labelling Using Makesense.Ai

is object class, object coordinates, height and width. For each object, a new line is created. For SSD, the labelling format should be in Pascal VOC which stores annotation in XML file. For Mask R-CNN, we have used VGG annotator for labelling purpose.

Our dataset is divided into training and testing sets, with 4000+ and 1000+ sequences, respectively. We have trained videos that are taken at different locations from the roads of Bangladesh as well as the testing videos, but ensuring the training and testing videos have similar traffic conditions and attributes. This setting reduces the chances of detection or tracking methods to overfit to particular scenarios.

4.3 Model Training

We have used three of the most popular deep learning algorithms for training and testing purposes. There are YOLO v5, Mask R-CNN, SSD.

- **YOLO v5:** YOLO divides an image into a grid system, and each grid detects objects within itself. The YOLO family consists of three main architectural blocks and for YOLOv5 those are YOLOv5 Backbone: It employs CSPDarknet as the backbone for feature extraction from images consisting of cross-stage partial networks. YOLOv5 Neck: It uses PANet to generate a feature pyramids network to perform aggregation

on the features and pass it to Head for prediction. YOLOv5 Head: Layers that generate predictions from the anchor boxes for object detection. Apart from this YOLOv5 uses the below choices for training – Activation and Optimization: YOLOv5 uses leaky ReLU and sigmoid activation, and SGD and ADAM as optimizer options. Loss Function: It uses Binary cross-entropy with logits loss. There are different varieties of a pre-trained version of YOLOv5.The difference between them is the trade-off between the size of the model and inference time. The lightweight model version YOLOv5s is just 14MB but not very accurate. On the other side of the spectrum, we have YOLOv5x whose size is 168MB but is the most accurate version of its family. As for this paper, YOLOv5s has been used for detection purpose. There are some steps that are need to follow to train the YOLO v5 model including:

- Set up the code.
- Prepare the data.
- Partition the dataset
- Data Config File
- Hyper-parameter Config File
- Custom Network Architecture
- Train the Model
- Computing the mAP on test dataset

For Mask R-CNN the steps are given below:

- Clone the Mask R-CNN GitHub Repository
- Install the Mask R-CNN Library
- Download Model Weights
- Load Model and Make Prediction
- Computing the mAP on test dataset

For SSD the steps are given below:

- Create a config file to store all parameters
- Construct DefaultBoxes and L2 Normalization Layer

- Construct the SSD Network

After detection, our next target is to track the vehicles. We have executed this task by using DeepSORT framework, YOLO v5 and GOTURN. Here we are going to discuss how each of them works individually:

- **DeepSORT Methodology:** The Simple Online and Realtime Tracking with a Deep Association metric (Deep SORT) enable multiple object tracking by integrating appearance information with its tracking components. If the Yolov5 can not detect any bounding box of vehicles in our dataset, The DeepSort algorithm won't be able to track the object. For tracking our vehicles with DeepSort Tracking Algorithm we need to go through some steps:

- As input video is divided and processed frame by frame, The Yolov5 detection is done to get information about the bounding box, the class including the features of the detected objects.

- The information of each vehicle (object) is then fed to the DeepSort model through a queue.

- The video frame and the information about the discovered objects are the two inputs in the Deep SORT method respectively. The Deep SORT algorithm's preprocessing removes overlapping bounding boxes.

- If no previous object ID can be attributed to a bounding box, a new object ID will be assigned to this bounding box. DeepSORT tracks items using two metrics: the position metric (IoU) and the appearance meter, as shown (feature).

- The bounding boxes will be processed in the features component if the IOU metric cannot assign the bounding box to any item ID. If the bounding box has already been assigned to an object ID in the checking component, the information about the object ID and the accompanying bounding box will be given to the resolution.

- **GOTURN Methodology:** GOTURN, short for Generic Object Tracking Using Regression Networks, is a Deep Learning-based tracking algorithm.The GOTURN is a single object tracker that is trained on thousands of video sequences and does not need to perform any learning at runtime. For tracking our moving vehicles through GOTURN tracker we need to go through some steps (Figure 4.4) :

- For Fine-tuning our dataset should be kept ready. At first, The location of the vehicle in the previous frame cropped to two times the size of the bounding box surrounding the object. The first clipped frame's vehicle is always centered. This is called target image.

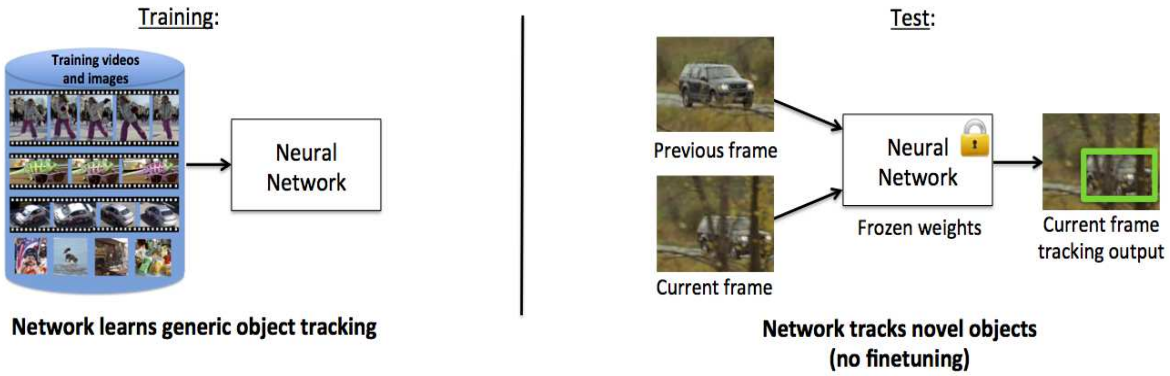

Figure 4.4: GOTURN methodology [11]

- The vehicle's location in the current frame must be predicted. Cropping the second frame is done with the same bounding box that was used to crop the first frame. The object is not centered in the second frame because it may have moved. This is called searching images.

- We randomly cropped this image during training time and generate a text file that contains the target image, search image, and ground truth bounding boxes. The network outputs the coordinates of the object in the current frame, relative to the search region. The network's output consists of the coordinates of the top left and bottom right corners of the bounding box. At test time, the network can track novel objects without any fine-tuning. For test, we take some images as input and generate a tracking video for evaluation and also generate loss curves during training with iteration respectively.

- **Speed Estimation:** For Speed estimation, We have estimated these values manually for the current road to calculate pixels per meter (ppm). Therefore, the value will vary from road to road and have to be adjusted to be used on any other video.

  If we talk about how we estimated ppm, we need to know the actual width in meters of the road(you can use google to find the approximate width of the road in your country). Also, we have taken the video frame and calculated the width of the road in pixels digitally. Now, we have the width of the road in meters from the real world and pixels from our video frame. To map the distances between these two worlds, we have calculated pixels per meter by dividing the distance of road in pixels to meters.

  dpixels gives the pixel distance traveled by the vehicle in one frame of our video processing. To estimate speed in any standard unit first, we need to convert dpixels to dmetres.

  Now, we can calculate the speed(speed = dmeters X fps X 3.6). dmeters is the distance traveled in one frame. We have already calculated the average fps during video pro-

cessing. So, to get the speed in m/s, just (dmetres X fps) will do. We have multiplied that estimated speed by 3.6 to convert it into km/hr.

4.4 Accuracy:

Two performance metrics are applied to object detecting models for testing. These are : precision and recall. Also, mAP value is measured. Using the outcome for three model their performance were compared.

4.5 Prediction rate of Detecting vehicles:

When it comes to smaller objects, SSD's performance is much worse as compared to Mask R-CNN. The main reason for this drawback, is that in SSD, higher resolution layers are responsible for detecting small objects. Accuracy of Mask R-CNN comes at the cost of time complexity. It is significantly slower than the likes of YOLO. YOLO v5 is one of the best modifications model that had been done to an object detection system.From graphs of three model YOLO V5 shows the best overall performance and most efficient one .

4.6 Prediction rate of Tracking vehicles:

For tracking vehicles, we need to define the initial state of the target by drawing a bounding box around it and have to estimate the target position, and also need to learn visual appearance, motion estimation, and exact location of the target. We used GOTURN for tracking. It only tracks a single object but it tracks perfectly. For the prediction of multiple vehicles in our dataset we used YoloV5 and Deep Sort algorithm to track. In addition, the DeepSORT tracking cannot track the object if the YOLO cannot detect any bounding box of this object which leads to the degradation of the object tracking concerning the identity switches. On the other hand with Yolo V5 we can detect as well as track objects. In comparison, YoloV5 tracker is faster than the current real-time DeepSORT tracking algorithm and gave superior tracking accuracy.

Chapter 5

Result Analysis

In this chapter, we'll go through the statistical outcomes of our proposed methodology. After preprocessing the data, we have prepared the dataset for training and testing, after that we have used some deep learning algorithms to train the model and measured its performances by testing data. For deep learning model, we have used YOLO v5, Mask R-CNN and SSD for detection purpose and for tracking we have used YOLO v5(individually), YOLO v5+ DeepSORT(Combined) and GOTURN.

5.1 Statistical Analysis for Proposed Models

5.1.1 Analysis for Proposed Models in Vehicle Detection

We have used YOLO v5 to train our model and at first batch we have trained the model using 1000+ data and measured the performance for epoch 90. The performance for YOLO v5 for 1000+ data is expressed by a table below:

Table 5.1: Result of YOLOv5 using custom dataset for 1000+ frames.

Classes	Epoch	mAP	Precision	Recall
Bus	90	0.905	0.83	0.923
Bike	90	0.928	0.856	0.75
Cycle	90	0.685	0.925	0.517
CNG	90	0.879	0.685	0.9
Car	90	0.617	0.83	0.545
Leguna	90	0.995	1	0.
Rickshaw	90	0.662	0.738	0.571

Continued on next page

Table 5.1 – continued from previous page

Classes	Epoch	mAP	Precision	Recall
Truck	90	0.738	0.879	0.8
Van	90	0.905	0.808	0.843

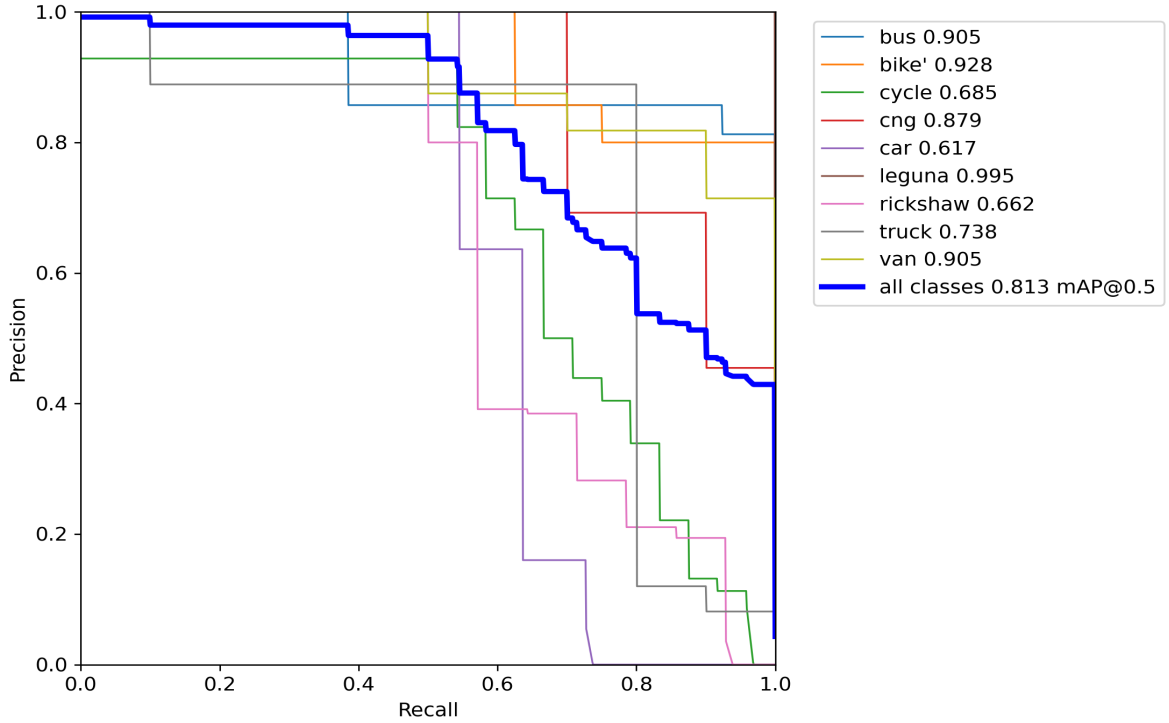

Figure 5.1: Result of YOLOv5 using custom dataset for 1000+ frames

From the above figure (Figure 5.1) we can see the performance of the model ny analysing the PR curve.

For second batch we have trained the model using 4000+ data and measured the performance for epoch 60. The performance for YOLO v5 for 4000+ data is expressed by a table below:

Table 5.2: Result of YOLOv5 using custom dataset for 4000+ frames.

Classes	Epoch	mAP	Precision	Recall
Bus	60	0.558	0.629	0.715
Bike	60	0.299	0.545	0.476
Cycle	60	0.308	0.44	0.294
CNG	60	0.529	0.51	0.475

Continued on next page

Table 5.2 – continued from previous page

Classes	Epoch	mAP	Precision	Recall
Car	60	0.754	0.455	0.756
Leguna	60	0.608	0.777	0.619
Rickshaw	60	0.358	0.839	0.303
Truck	60	0.496	0.891	0.47
Van	60	0.401	0.486	0.4

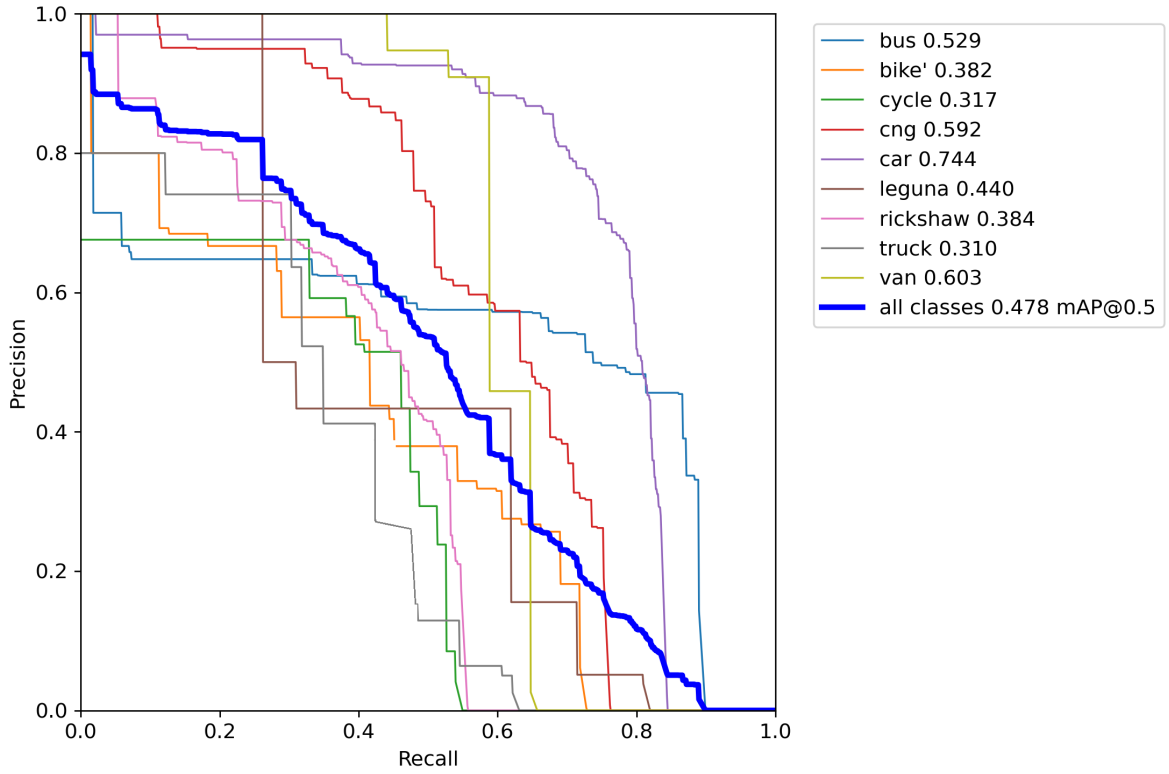

Figure 5.2: Result of YOLOv5 using custom dataset for 4000+ frames

From the above figure (Figure 5.2), we can see the performance of the model by analysing the PR curve.

For third batch we have trained the model using 8000+ data and measured the performance for epoch 60. The performance for YOLO v5 for 8000+ data is expressed by a table below:

Table 5.3: Result of YOLOv5 using custom dataset for 8000+ frames.

Classes	Epoch	mAP	Precision	Recall
Bus	60	0.529	0.495	0.495
Bike	60	0.382	0.378	0.542
Cycle	60	0.317	0.63	0.329
CNG	60	0.592	0.689	0.509
Car	60	0.744	0.81	0.687
Leguna	60	0.44	0.425	0.405
Rickshaw	60	0.384	0.718	0.289
Truck	60	0.31	0.406	0.424
Van	60	0.603	0.747	0.588

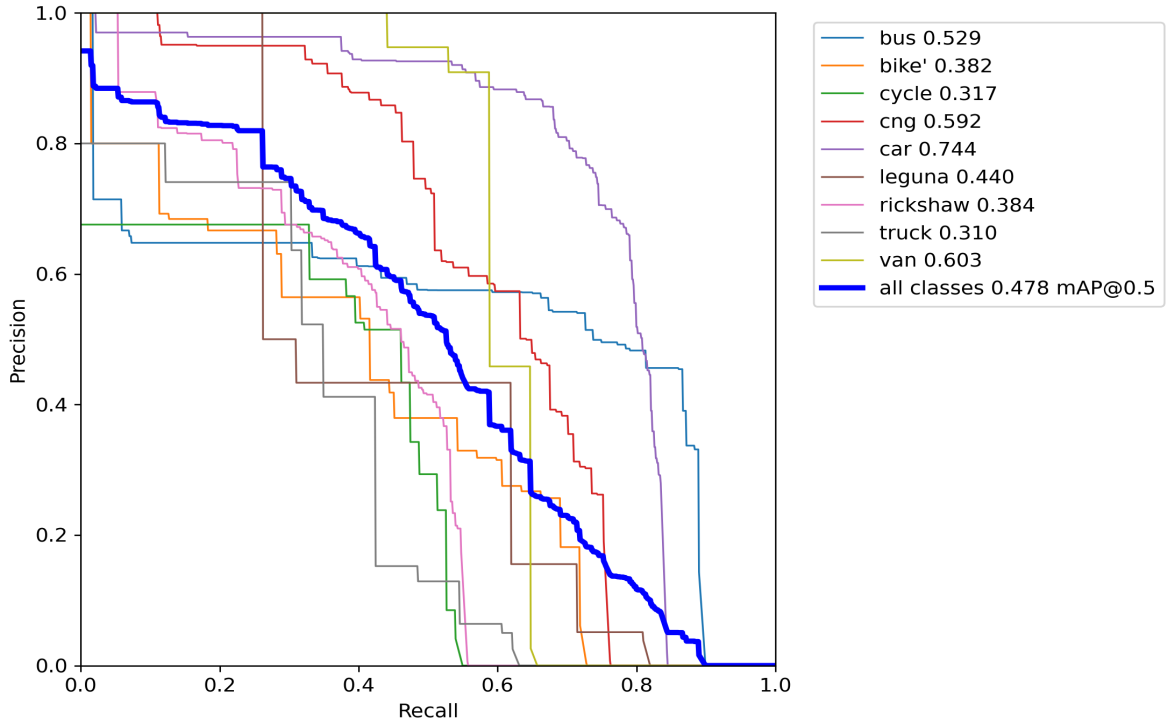

Figure 5.3: Result of YOLOv5 using custom dataset for 8000+ frames

From the above figure (Figure 5.3), we can see the performance of the model by analysing the PR curve.

Table 5.4: Comparison Result of Deep Learning models using custom dataset for 8000+ frames

Models	Epoch	mAP	Precision	Recall
YOLOv5	60	73.9	72.4	70.2
Mask R-CNN	60	55.4	37.9	54.2
SSD	60	56.4	70.1	32.9

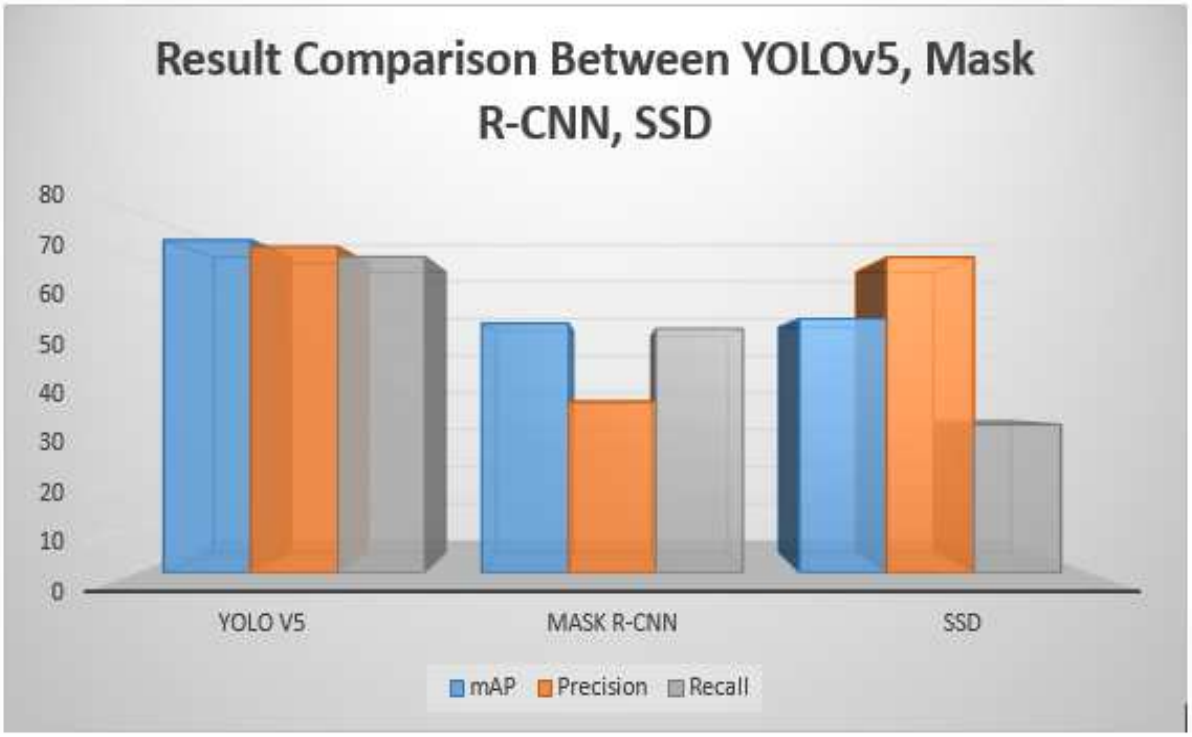

Figure 5.4: Comparison Result of Deep Learning models using custom dataset for 8000+ frames

Many research efforts in object detection focus on making standard detection pipelines fast. Their relative mAP and speed are also compared to examine the accuracy-performance tradeoffs available in object detection systems.Our training experiments show that (Figure 5.4) initially, Faster R-CNN model achieves better accuracy (in terms of mAP) than SSD model .Yolo V5 was ultimately selected as it gives higher accuracy (in terms of mAP)and runs faster while managing better performance in detecting small objects.

5.1.2 Analysis for Proposed Models in Vehicle Tracking

- **YOLO v5:** For tracking purpose, at first We have used YOLO v5 to train our model and we have trained the model using around 8000 data and measured the performance for epoch 60. The performance for YOLO v5 for around 8000 data is expressed by a table below:

Table 5.5: Result of YOLOv5 using custom dataset for 8000 frames.

Classes	Epoch	mAP@.5	mAP@.5:.95	Precision	Recall
Bus	60	0.82	0.51	0.56	0.907
Bike	60	0.482	0.232	0.393	0.69
Cycle	60	0.592	0.21	0.61	0.638
CNG	60	0.763	0.456	0.793	0.665
Car	60	0.82	0.524	0.745	0.802
Leguna	60	0.865	0.535	0.76	0.905
Rickshaw	60	0.46	0.215	0.662	0.369
Truck	60	0.612	0.304	0.635	0.561
Van	60	0.452	0.209	0.5	0.559

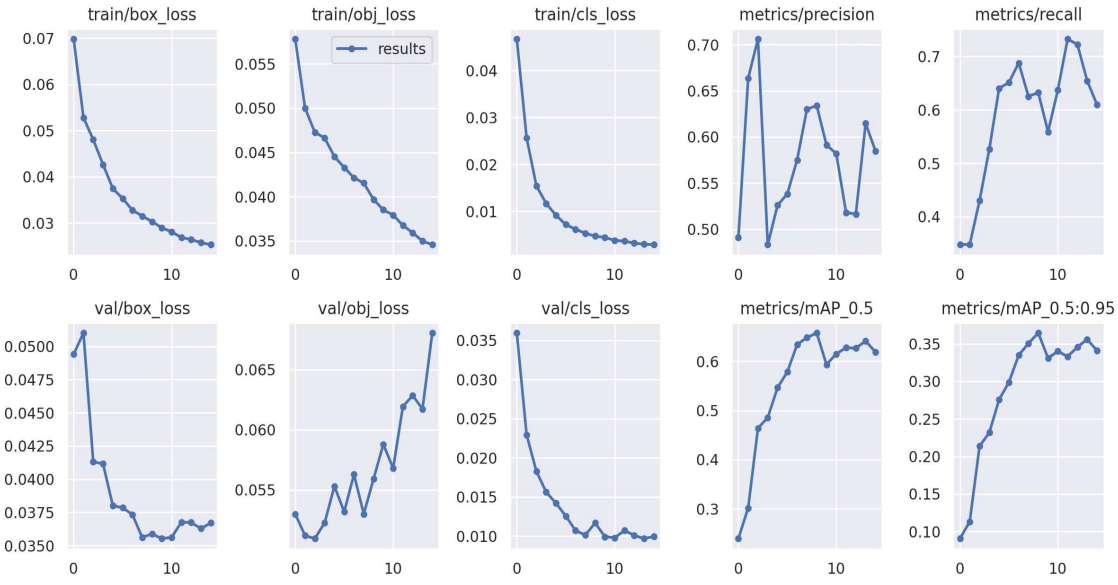

Figure 5.5: Loss curve of YOLOv5 using custom dataset for around 8000 frames

Here (Figure 5.5), we are showing the boxloss and classloss for both training period and validation period. By measuring box loss it can be identified how "tight" the predicted bounding boxes are to the ground truth object, which usually refers to regression loss. Through class loss it can be measured the correctness of the classification of each predicted bounding box:

each box may contain an object class, or a "background". This loss is usually called cross entropy loss.

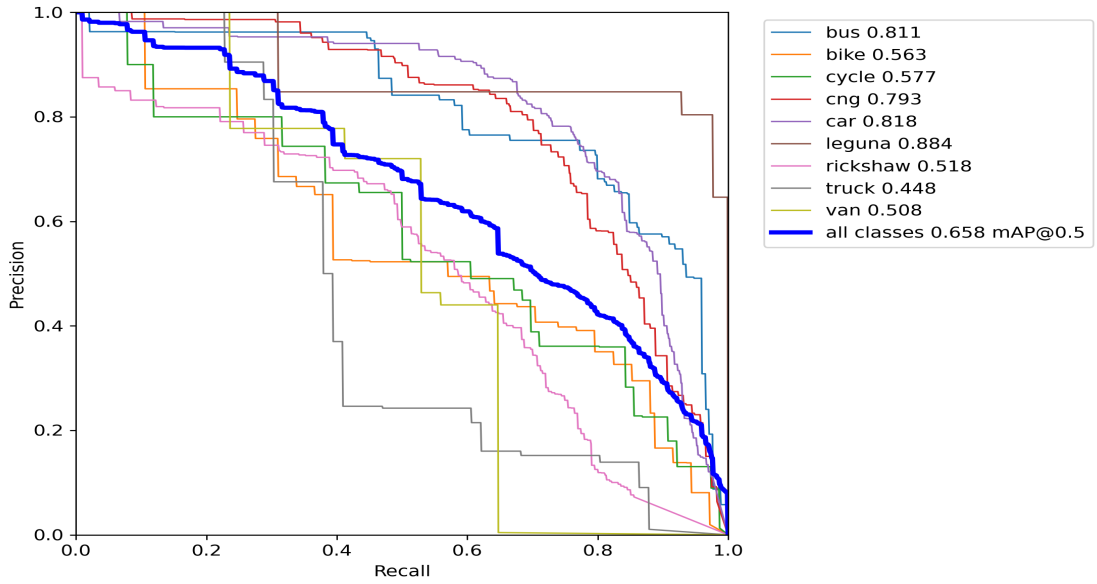

Figure 5.6: PR curve of YOLOv5 using custom dataset for around 8000 frames

From the above figure (Figure 5.6), we can see the performance of the model by analysing the Precision-Recall(PR) curve.

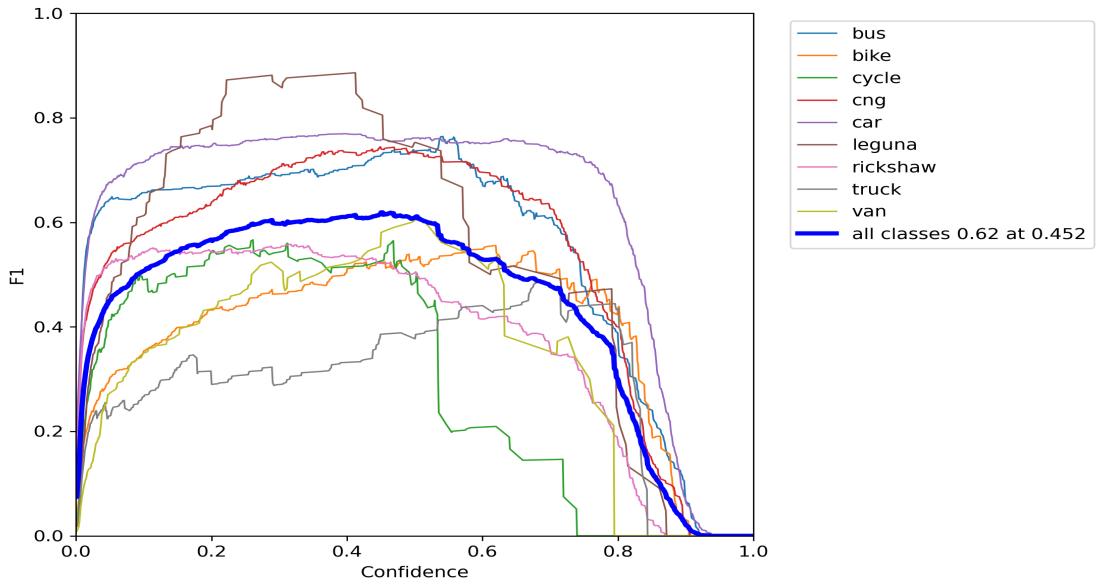

Figure 5.7: f1 score of YOLOv5 using custom dataset for around 8000 frames

Here (Figure 5.7), we are showing the f1 score through graphical representation. Accuracy can be used when the class distribution is similar while F1-score is a better metric when there are imbalanced classes as in our case. The F1-score combines the precision and recall of a classifier into a single metric by taking their harmonic mean. It is primarily used to compare the performance of two classifiers: higher recall and higher precision.

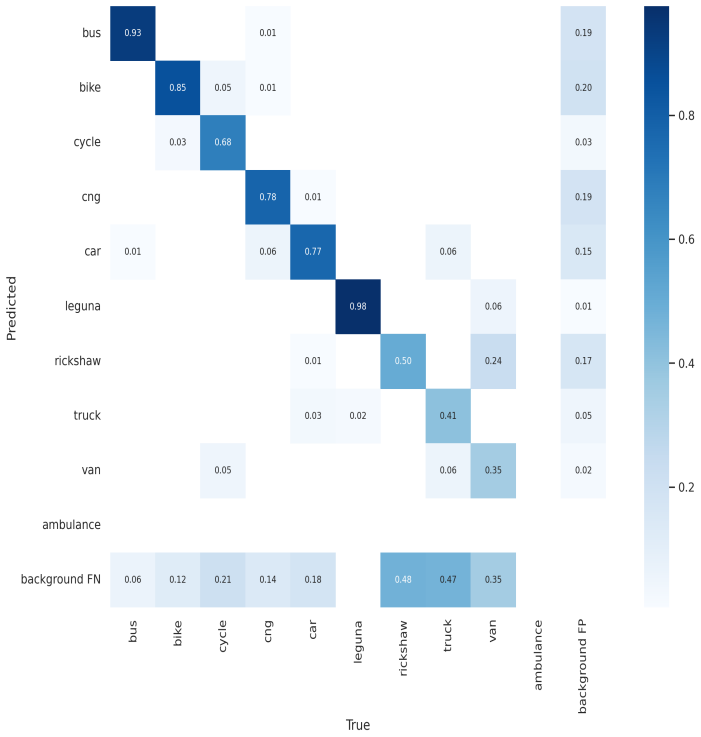

Figure 5.8: Confusion matrix for YOLOv5 using custom dataset for around 8000 frames

Here (Figure 5.8), confusion matrix has been plotted to visualise the important predictive analytics like recall, accuracy, and precision. It gives an direct comparisons of values like True Positives, False Positives, True Negatives and False Negatives.

- **DeepSORT:** Here at first we have used YOLO v5 to train our model for detection purpose and after that we have used DeepSORT framework for tracking purpose.We have trained the model using around 8000 data and measured the performance for epoch 60. The performance for this case for around 8000 data is expressed by a table below:

Table 5.6: Result of DeepSORT using custom dataset for 8000 frames.

Classes	Epoch	mAP@.5	mAP@.5:.95	Precision	Recall
Bus	60	0.721	0.421	0.585	0.758
Bike	60	0.419	0.196	0.383	0.634
Cycle	60	0.613	0.211	0.724	0.513
CNG	60	0.709	0.429	0.659	0.622
Car	60	0.826	0.532	0.782	0.752
Leguna	60	0.703	0.403	0.828	0.524
Rickshaw	60	0.533	0.243	0.695	0.428

Continued on next page

Table 5.6 – continued from previous page

Classes	Epoch	mAP@.5	mAP@.5:.95	Precision	Recall
Truck	60	0.499	0.314	0.415	0.485
Van	60	0.541	0.308	0.704	0.588

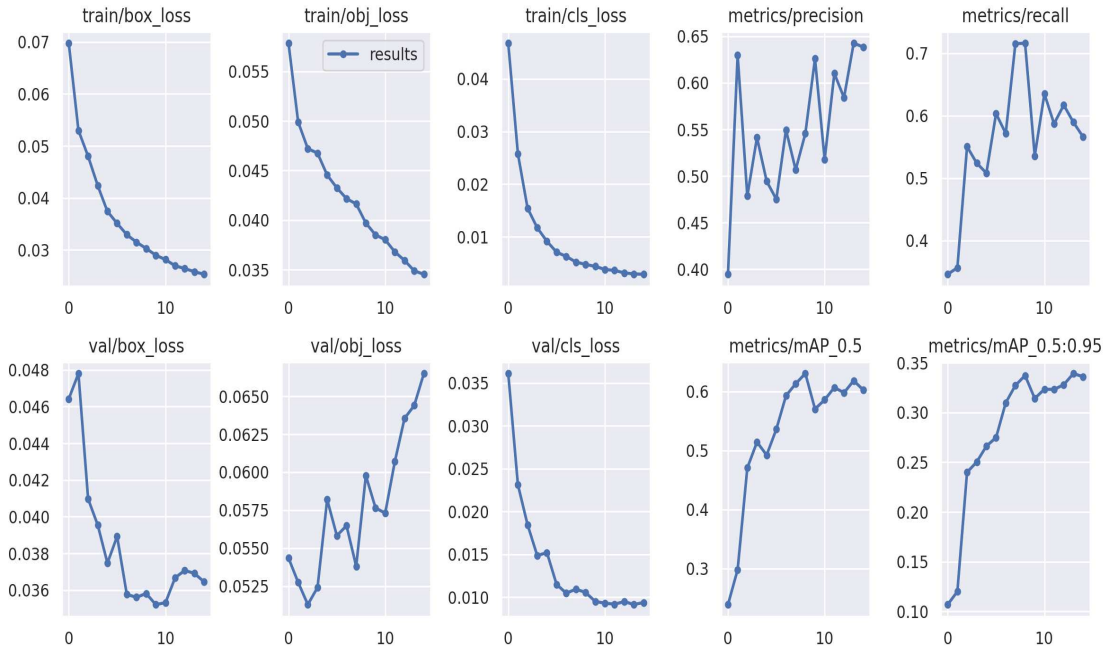

Figure 5.9: Loss curve of DeepSORT using custom dataset for around 8000 frames

Here (Figure 5.9), we are showing the boxloss and classloss for both training period and validation period.

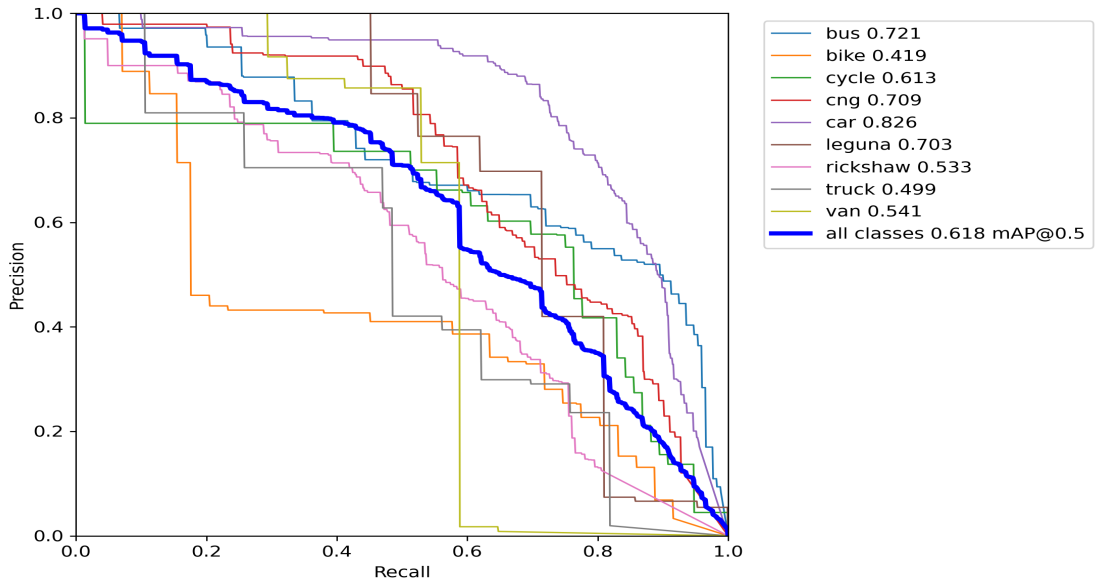

Figure 5.10: PR curve of DeepSORT using custom dataset for around 8000 frames

From the above figure (Figure 5.10), we can see the performance of the model by analysing the Precision-Recall(PR) curve.

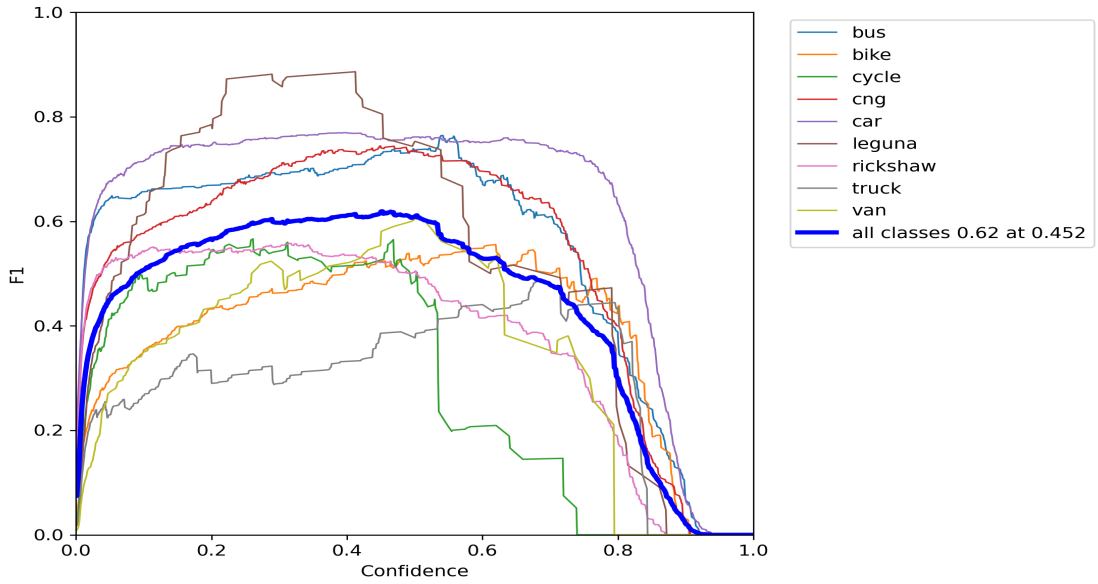

Figure 5.11: f1 score of DeepSORT using custom dataset for around 8000 frames

Here (Figure 5.11), we are showing the f1 score through graphical representation.

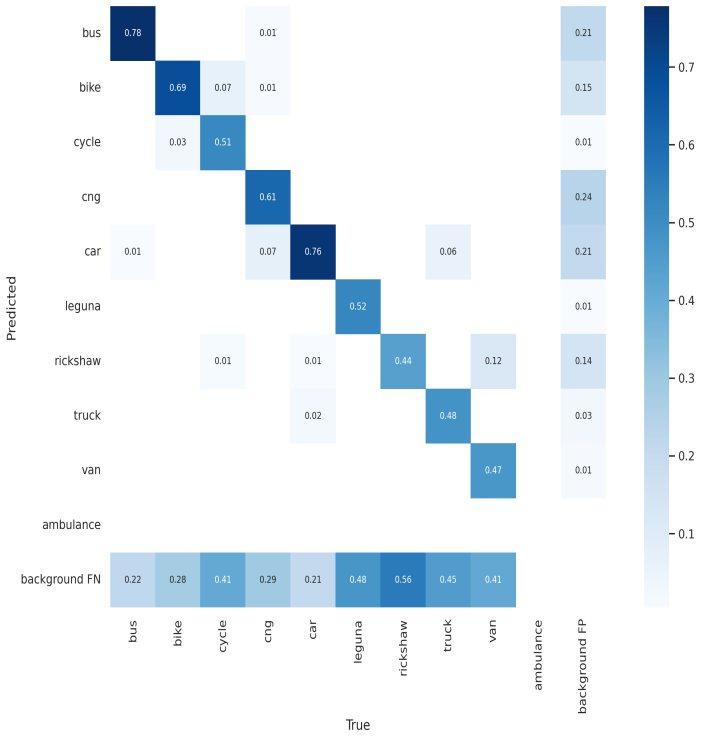

Figure 5.12: Confusion matrix for DeepSORT using custom dataset for around 8000 frames

Here (Figure 5.12), we are showing the confusion matrix for DeepSORT framework.

- **GOTURN:** Here we have used GOTURN to train our model for tracking purpose. We have trained the model using around 8000 data and measured the performance for epoch 60.

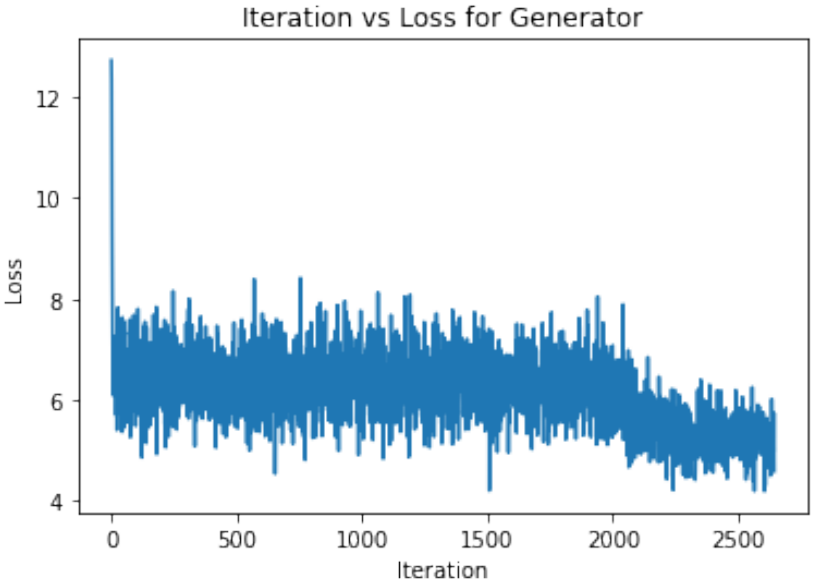

Figure 5.13: Loss Curve for GOTURN using custom dataset for around 8000 frames

During the training of GOTURN tracking algorithm in our customized dataset, the current state of the model at each step of training can be evaluated. We can get an idea of how the model is learning. From this curve (Figure 5.13), we can conclude that the training loss continues to decrease until the end of training. With the increase of iteration and time elapsed the average loss is decreasing gradually.

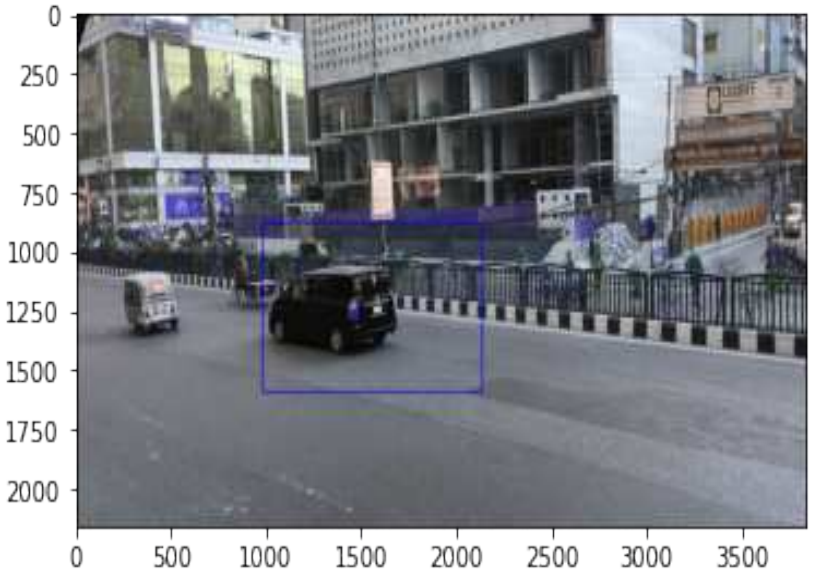

Figure 5.14: Output for GOTURN using custom dataset for around 8000 frames

After training our dataset successfully with GOTURN algorithm, we have to validate the dataset. So, we tested with the number of frames, and each time it tracks the object correctly (Figure 5.14). The tracker regresses directly to the output bounding box, so GOTURN achieves accurate tracking with no extra computational cost, enabling it to track objects at 100 fps.

5.1.3 Result of Speed Estimation and Counting

While using weights of YOLO v5, the model runs on 10 pre-defined classes. On the basis of that result, the counting vehicle part is shown. The Speed Estimation part is basically creating the frame and converting pixels into calculating values and applying them to formulas. This gives the individual result showing a tracking id, bounding box id, lane position of the vehicle and approximate speed. As the output depends on video input, the pixels and other variables, as well as vehicle speed, may differ in some cases.(Figure 5.15)

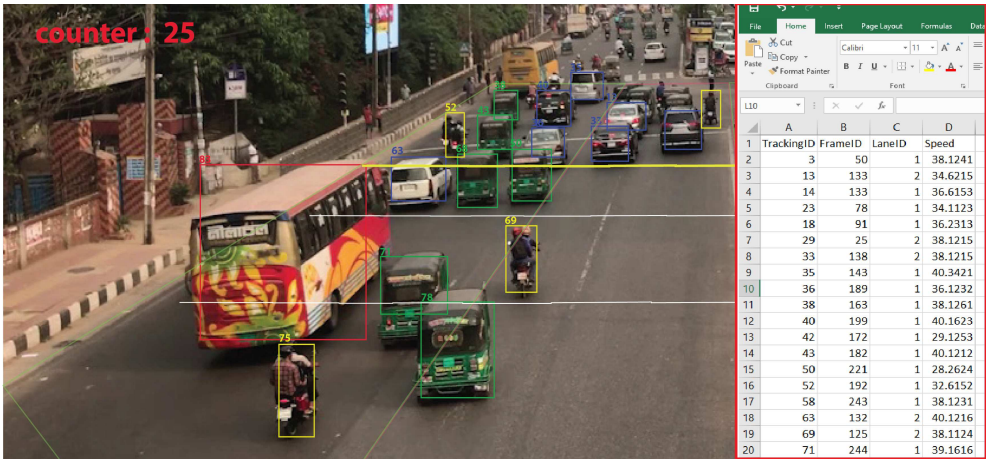

Figure 5.15: Output of Speed Estimation and Counting

Chapter 6

Conclusion and Future Work

6.1 Conclusion

For our research work, a self-built dataset is made based on the Bangladeshi local vehicle's perspective. This study has established a high-definition vehicle dataset and proposed an object detection and tracking method as well as counted vehicles and estimated the speed of the vehicles from highway surveillance video scenes. We used the Yolo V5 model, Mask R-CNN, and Single Shot Detector (SSD) model for detecting vehicles. From our experimental analysis, we came to know that SSD can make good detection with good speed whereas Mask R-CNN can make slow detection. We got different accuracy for them. YoloV5, Mask R-CNN, and SSD gave us mAP of 73.9 %, 55.4%, and 56.4% respectively. Yolo V5 model outperforms the other two object detection models Mask R-CNN and SSD. As a consequence, we proposed the YOLO V5 model for vehicle detection. For Tracking purposes, we used YOLOv5, Deep Sort, and GOTURN. As GOTURN only tracked a single object, from the perspective of Bangladesh it is not that much worthy for tracking. On contrary, Deep Sort is a recent algorithm for tracking that extends Simple Online and Real-time Tracking and has shown remarkable results in the Multiple Object Tracking (MOT) problem. Though YoloV5 can detect an object as well as tracking, for tracking with DeepSORT, at first we detected vehicles with YoloV5 and then tracked with Deep Sort algorithm. Detecting objects with YoloV5 and then tracking with Deep Sort Algorithm is efficient. But For tracking, YoloV5 and Deep Sort gave us mAP of 65.2% and 61.8% respectively. For tracking Yolo V5 gave us a satisfactory result compared to Deep Sort. To identify vehicles, counting the number of vehicles and speed estimation we also used YoloV5. In addition, the methodology and results of the vehicle detection, tracking, and counting system provided in the analysis in our work will become important references for our local transport and for ensuring a safe transport system.

6.2 Future Work

Our Future work includes building a system where we can have data of any particular road, especially where multi speeding transports run simultaneously. So, what happens that we can have a clear idea on that particular road how many local vehicles like rickshaw, CNG, leguna passes daily. Also, what's their average speed compared to the other heavy vehicles like bus, car, truck that runs along the road. If the average speeds don't add up, meaning that a particular vehicle shouldn't be moving on that road. Also, Yolo is constantly updating with more and accurate features. Particular Vehicle paths according to the speed limit can also be identified. With all this in mind, we want a constant updating tracking systems for our vehicles not to be mentioned, giving priority to our local vehicles with more enriched data to have a better and safe transport system minimizing the accidents to a great extent.

References

[1] "How MATLAB Represents Pixel Colors." https://www.mathworks.com/company/newsletters/articles/how-matlab-represents-pixel-colors.html. Accessed: 2021-06-15.

[2] "Digital Image Processing Basics." https://www.geeksforgeeks.org/digital-image-processing-basics/. Accessed: 2021-06-15.

[3] "Chegg.com." https://www.chegg.com/homework-help/questions-and-answers/object-detection-use-bounding-boxes-ident
Accessed: 2021-06-22.

[4] "Object Detection Guide." https://www.fritz.ai/object-detection/. Accessed: 2021-02-23.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 779–788, 2016.

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

[7] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*, pp. 21–37, Springer, 2016.

[8] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-based convolutional networks for accurate object detection and segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 38, no. 1, pp. 142–158, 2015.

[9] E. Bochinski, V. Eiselein, and T. Sikora, "High-speed tracking-by-detection without using image information," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, IEEE, 2017.

[10] E. Bochinski, T. Senst, and T. Sikora, "Extending iou based multi-object tracking by visual information," in *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pp. 1–6, IEEE, 2018.

[11] D. Held, S. Thrun, and S. Savarese, "Learning to track at 100 fps with deep regression networks," in *European conference on computer vision*, pp. 749–765, Springer, 2016.

[12] "Measuring Object Detection models-mAP-What is Mean Average Precision?." https://towardsdatascience.com/what-is-map-understanding-the-statistic-of-choice-for-comparing- Accessed: 2021-06-11.

[13] "What is Object Tracking? - An Introduction." https://viso.ai/deep-learning/object-tracking/. Accessed: 2021-12-24.

[14] S. Sri Jamiya and E. Rani, "A survey on vehicle detection and tracking algorithms in real time video surveillance,"

[15] R. A. Hadi, G. Sulong, and L. E. George, "Vehicle detection and tracking techniques: a concise review," *arXiv preprint arXiv:1410.5894*, 2014.

[16] A. Ambardekar, M. Nicolescu, and G. Bebis, "Efficient vehicle tracking and classification for an automated traffic surveillance system," *Signal and Image Processing*, pp. 1–6, 2008.

[17] A. I. B. Parico and T. Ahamed, "Real time pear fruit detection and counting using yolov4 models and deep sort," *Sensors*, vol. 21, no. 14, p. 4803, 2021.

[18] Y. Huang and H. Zhang, "A safety vehicle detection mechanism based on yolov5," in *2021 IEEE 6th International Conference on Smart Cloud (SmartCloud)*, pp. 1–6, IEEE, 2021.

[19] W. Zhan, C. Sun, M. Wang, J. She, Y. Zhang, Z. Zhang, and Y. Sun, "An improved yolov5 real-time detection method for small objects captured by uav," *Soft Computing*, pp. 1–13, 2021.

[20] "Make Sense." https://www.makesense.ai/. Accessed: 2021-05-10.

[21] "tzutalin/labelImg." https://github.com/tzutalin/labelImg. Accessed: 2021-06-15.

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This thesis was generated on Thursday 6th January, 2022 at 2:47pm.