VEHICLE DETECTION USING DEEP LEARNING

A PROJECT REPORT

Submitted by

ATHUL R ASHOK

REG NO : TKM19MCA008

In partial fulfillment for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

Thangal Kunju Musaliar College of Engineering
Kerala

MAY 2022

DECLARATION

I undersigned hereby declare that the project report DETECTION OF VEHICLE USING DEEP LEARNING, submitted for partial fulfillment of the requirements for the award of degree of Master of Computer Applications of the APJ Abdul Kalam Technological University, Kerala is a bonafide work done by me under supervision of Prof.Jasmin MR. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute, or by the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree, diploma or similar title of any other University.

Place: Kollam

Date: 21-06-22

ATHUL R ASHOK

DEPARTMENT OF COMPUTER APPLICATIONS
TKM COLLEGE OF ENGINEERING

C E R T I F I C A T E

This is to certify that, the report entitled "**DETECTION OF VEHICLE USING DEEP LEARNING"** submitted by **ATHUL R ASHOK**, to the **APJ Abdul Kalam Technological University** in partial fulfillment of the requirements for the award of the Degree of **Master of Computer Applications** is a bonafide record of the project work carried out by him under our guidance and supervision.This report in any form has not been submitted to any other University or Institute for any purpose.

Internal Supervisor | Head of the Department | External Examiner

ACKNOWLEDGEMENT

First and foremost I thank GOD almighty and my parents for the success of this project. I owe sincere gratitude and heart full thanks to everyone who shared their precious time and knowledge for the successful completion of my project.

I am extremely grateful to **Dr Fousia M Shamsudeen**, Head of the Department for providing us with best facilities.

I would like to thank project guide **Prof. Jasmin MR**, Department of Computer Applications, who motivated me throughout the project.

I profusely thank all other faculty members in the department and all other members of TKM College of Engineering, for their guidance and inspirations throughout my course of study.

I owe my thanks to our friends and all others who have directly or indirectly helped us in the successful completion of this project.

ATHUL R ASHOK

ABSTRACT

The deep learning object detection algorithms have become one of the powerful tools for road vehicle detection in autonomous driving where vehicles are capable of sensing its environment and operating without human involvement where they can go anywhere a traditional car goes and do everything that an experienced human driver does. However, the limitation of the number of high-quality labeled training samples makes the single-object detection algorithms unable to achieve satisfactory accuracy in road vehicle detection. In this paper, by comparing the pros and cons of various object detection algorithms, an ensemble is attenuated with multiple models and they are selected for a method where the first step groups the overlapping regions. Subsequently, a voting strategy is applied to discard some of those groups, this can further reduce the vehicle misdetection of the target detection algorithm, helps in obtaining a better detection result with a Non-maximum Suppression algorithm for the final prediction.

Contents

List of Figures

List of Tables

List of Abbreviations

KITTI	– Karlsruhe Institute of Technology and Toyota Technological Institute
RCNN	– Region-based Convolutional Neural Network
DPM	– Deformable Part Model
CRF	– Conditional Random Field
UAV	– Unmanned Aerial Vehicle
SSD	– Single Shot Detectio
YOLOV3	– You Only Look Once V3
COCO	– Common Objects In Context
ResNet	– Residual Neural Network
API	– Application Programming Interface
GUI	– Graphical user interface
OS	– Operating System
NMS	– Non Maximum Suppression Algorithm
XML	– Extensible Markup Language

Chapter 1

INTRODUCTION

Environmental awareness such as awareness about the surrounding vehicles and pedestrians is a crucial part of intelligent vehicles technology development. To perceive the surrounding environment more accurately, intelligent vehicles often adopt a multi-sensor fusion scheme, which combines the millimeter-wave radar, lidar, camera, and other equipment. After the calibration, the incoming data and detection results are fused at different levels to obtain a more reliable perceptual result. However, the multisensor solutions are often too costly, so there is still a long way to go before mass marketization. Perceiving the surrounding environment only by a camera is undoubtedly the lowest-cost and the most feasible sensing strategy.

Computer vision algorithms based on deep learning have recently achieved great success in various computer vision tasks, such as object classification, object detection, semantic segmentation, and so on, as a rising technology in the computer vision field, allowing for the continuous improvement of data volume and rapid hardware advancement. There are many object detection algorithms available today which have good effects and real-time performance, so making use of them to detect vehicles and make autonomous driving more effective and safe.

1.1 Problem Definition

Deep learning object detection algorithms have become one of the most powerful tools for autonomous driving road vehicle detection. The achieved method is one of the best for VEHICLE

detection, as it combines a number of detection models for improved detection accuracy. Every object has discrete features in an object detection model that help distinguish it in a photo or video frame. Object detection methodologies employ these characteristics in determining, identifying, and labeling objects and they've been used to classify various types of vehicles and address illumination issues in real-world scenarios. The detection result is obtained using a set of single-stage detection models, which are much faster in detection and are well suited for road vehicle detection. Single-stage models are good, but their detection accuracy is poor because there is no targeted measure, so the idea is to use an ensemble model that combines multiple models to improve road vehicle detection accuracy.

1.2 Objective

1.2.1 Illumination Problems

- In real world there are different light sources where the day light and the artificial lights in the night perform differently according to the time. So the main idea here is to achieve admissible vehicle detection in different lighting conditions.

1.2.2 Motorcycle Detection

- Motorcycles are the most common used way of transportation where they plan an important role in cost effective way of transportation so people use it more on a regular basis so the detection of motorcycles is an important factor of a better autonomous driving feedback so the aim is to improve motorcycle detection.

1.2.3 Vehicle classification

- In real world there are different types of vehicles are filled with so its important to differentiate vehicle according to to its size and type so classification of light,heavy vehicles and control over the traffic was a good possibility of an effective vehicle detection.

Chapter 2

LITERATURE SURVEY

2.1 Purpose of the Literature Review

1. It gives readers easy access to research on a particular topic by selecting high quality articles or studies that are relevant, meaningful, important and valid and summarizing them into one complete report.
2. It provides an excellent starting point for researchers beginning to do research in a new area by forcing them to summarize, evaluate, and compare original research in that specific area.
3. It ensures that researchers do not duplicate work that has already been done.
4. It can provide clues as to where future research is heading or recommend areas on which to focus.
5. It highlights the key findings.
6. It identifies inconsistencies, gaps and contradictions in the literature.
7. It provides a constructive analysis of the methodologies and approaches of other researchers.

2.2 Related Works

H. Wang et al.[1]- used an ensemble model combining the RetinaNet and Cascade-RCNN is capable of achieving an average detection accuracy of 94.75 percent for easy-level detection targets, being the third-ranked on the KITTI list for easy-level targets detection. onsidering that in the both KITTI test set and real road scene, the easy-level target vehicles represent the majority, therefore this paper first determines the fusion of the RetinaNet and Cascade-RCNN Then, the model-combination scheme is used for detection. [1].

Cai et al.[2]- proposes a probabilistic framework combining a scene model with a pattern recognition method for vehicle detection by a stationary camera. Then, the possible vehicle location and the probability distribution around the viewpoint in a fixed location are calculated. For each viewpoint, the vehicle model described by a deformable part model (DPM) and a conditional random field (CRF) is learned. Scores of root and parts and their spatial configuration generated by the DPM are used to learn the CRF model. The occlusion states of vehicles are defined based on the visibility of their parts and considered as latent variables in the CRF. In the online procedure, the output of the CRF, which is considered as an adjusted vehicle detection result compared with the DPM, is combined with the probability of the apparent viewpoint in a location to give the final vehicle detection result. [2].

Y. Xu et al.[3] - develops an advanced vehicle detection method, which improves the original Viola-Jones (V-J) object detection scheme for better vehicle detections from lowaltitude unmanned aerial vehicle (UAV) imagery. The original V-J method is sensitive to objects' in-plane rotation, and therefore has difficulties in detecting vehicles with unknown orientations in UAV images. To address this issue, this research proposes a road orientation adjustment method, which rotates each UAV image once so that the roads and on-road vehicles on rotated images will be aligned with the horizontal direction and the V-J vehicle detector. Then, the original V-J can be directly applied to achieve better efficiency and accuracy. The enhanced V-J method is further applied for vehicle tracking. Testing results show that both vehicle detection and tracking methods are competitive compared with other existing methods. Future research will focus on expanding the current methods to detect other transport modes, such as buses, trucks, motorcycles, bicycles, and

pedestrians [3].

A. Mukhtar et al.[4] - provide a comprehensive survey in a systematic approach about the state-of-the-art on-road vision-based vehicle detection and tracking systems for collision avoidance systems (CASs). This paper is structured based on a vehicle detection processes starting from sensor selection to vehicle detection and tracking. Techniques in each process/step are reviewed and analyzed individually. Two main contributions in this paper are the following: survey on motorcycle detection techniques and the sensor comparison in terms of cost and range parameters. Finally, the survey provides an optimal choice with a low cost and reliable CAS design in vehicle industries. [4].

Chapter 3

METHODOLOGY

Vehicle detection is an important aspect of autonomous driving, the paper aims for:
Faster detection – Its crucial to detect vehicles faster and accurate in real world for a safer environment.
Motorcycle detection – Detecting motorcycles is a challenging task due to its low visibility and high velocity
Vehicle Classification – Vehicle classification is to categorize the detected vehicles into their respective types
Illumination problems – Real world scenario consist of different light sources.
Detection model – Single detection models are more likely to fail in better detection.

3.1 Proposed system

To overcome the limitations of traditional VEHICLE DETECTION system a new ensemble method consist of multiple deep learning models are been used. The method of real-time detection considering the high-quality road vehicle training data is often not enough to realize a detection model with a strong generalization ability, resulting in a severe over fitting phenomenon.

Deep learning-based image processing has been shown to be effective in automated detection of VEHICLE. However, deep learning typically requires a large number of labelled samples for training, which is time consuming and expensive to implement as it requires high end computational power, so the use of pre-trained models are more feasible. In the paper, three models are presented based on the applications of deep learning and are capable of classifying and detect vehicles with better accuracy and reduce over fitting.

The proposed methodology uses three deep learning models Single Shot Detection, You Only Look Once V3 and RetinaNet deep transfer learning algorithms that extracts the features from the input image that describes the presence of vehicles automatically and reports whether it is a vehicle or not from the best of these algorithms.

3.2 System Architecture

The main purpose of the project is vehicle detection. So when given some images of a real world road scenario, the proposed algorithm can detect whether it is a vehicle, person or even animals can be expected as a part of real world road scene.
This study uses three well-known models Single Shot Detection, You Only Look Once V3 and RetinaNet for the detection of VEHICLE. This demonstrates that using pre trained models for image classification is an effective way to detect VEHICLE. The figure 3.1 below shows system architecture. The proposed system consist of four major phases:

- Data preprocessing and data augmentation
- Training the data set
- Evaluation of the model
- Deploying the model using Flask API

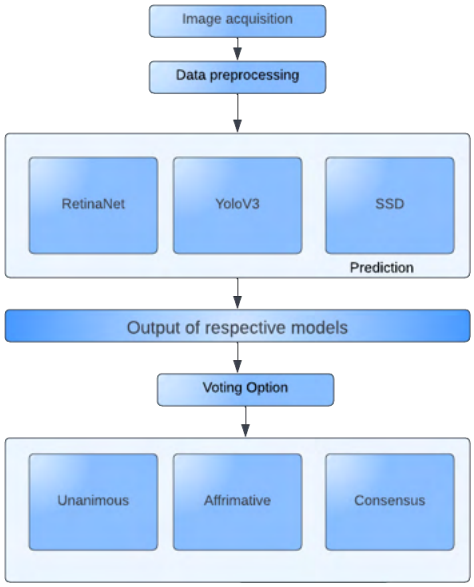


Figure 3.1: System Architecture

3.2.1 Data set

The data set used here for VEHICLE detection is COCO Dataset which is a large-scale object detection, segmentation, and captioning data set which contains 33000 images. The Convolutional Neural Network is built and trained using the Keras open-source deep learning framework and the TensorFlow backend. The training, testing, and validating images were included in the dataset. To improve the system and increase efficiency, the data is modified into the training and validation set. The training set contains around 33000 images in total. 1.5 million instances of objects There are 80 object categories and 91 item categories. There are five captions per image. To improve the overall accuracy, 250,000 people with key points are assigned to the validation set.

3.2.2 Data Preprocessing and Augmentation

To improve the performance of deep learning algorithms, augmentation techniques are widely used. The primary goal of data augmentation is to improve the performance of the convolutional neural network model that classifies the input images. To improve and augment the quality, adeptness, and size of the data, data augmentation is used on the training data, which in this case is

images.

To help the Deep Learning model capture multiple nuances in the training images, a number of operations are performed to increase the size of the data. Convolutional Neural Networks are fed augmented data to avoid data overfitting and to improve the model's performance. Overfitting occurs when a machine learning or deep learning model performs well on the training set but fails to perform well on the testing and validation sets. As a result, excessive model fitting is not recommended and should not occur during implementation. Deep Learning and Computer Vision algorithms can fit models more efficiently when the training dataset is larger. The model has been trained using a variety of data augmentation techniques. This effectively does not alter the original dataset and is only implemented during runtime, preventing the use of unnecessary disc space to store the modified and augmented images. Flipping, rotation, and shearing are examples of data augmentation.

3.2.3 Training the model

Another performance enhancing method in deep models especially in CNNs which is named as transfer learning. Transfer learning is the idea of overcoming the isolated learning paradigm and utilizing knowledge acquired for one task to solve related ones. There are three different transfer learning approaches in CNNs. These are feature extractor, fine-tuning and pre-trained models. The study, used pre-trained approach. The models used in the propoed system Single Shot Detection, You Only Look Once V3 and RetinaNet. Both Single Shot Detection, You Only Look Once V3 and RetinaNet are predicted separately and these models are saved as xml files.

3.2.4 Testing the model

The proposed Single Shot Detection, You Only Look Once V3 and RetinaNet models are tested with testing data images for classifying the VEHICLES from a road scene as images.

3.2.5 Deploying the model in Flask API

Developed a basic API using Flask where user can upload real road scenario image which is to be detected and the system predicts whether it is VEHICLE or not.

3.3 RetinaNet Architecture

RetinaNet is one of the best one-stage object detection models, having proven to work well with dense and small scale objects. It was created by combining two existing single stage object detection models.

There are four major components of a RetinaNet architecture :

a) Bottom-up Pathway - The backbone network (e.g. ResNet) calculates feature maps at various scales, regardless of the size of the input image or the backbone.

b) Top-down pathway and Lateral connections - The top-down pathway upsamples the spatially coarser feature maps from higher pyramid levels, while the lateral connections combine top-down and bottom-up layers of the same spatial size..

c) Classification subnetwork - For each anchor box and object class, it predicts the likelihood of an object being present at each spatial location..

d) Regression subnetwork - It regresses the offset for each ground-truth object's bounding boxes from the anchor boxes.

The figure 3.1 below shows different layers of RetinaNet architecture.

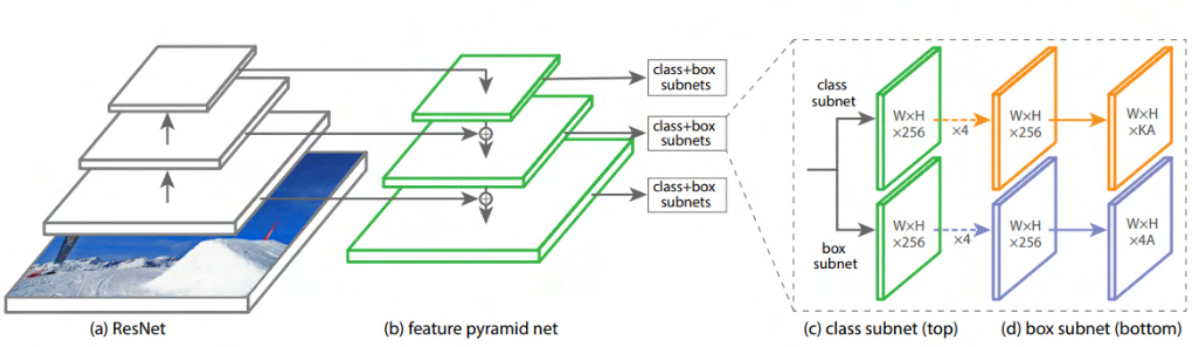


Figure 3.2: Architecture of RetinaNet

3.4 YOLOv3 Architecture

YOLOv3 is an object detection algorithm that detects specific objects in videos, live feeds, or images in real time. To detect an object, YOLO uses features learned by a deep convolutional neural network. The YOLOv3 feature detector's architecture was influenced by well-known architectures such as ResNet and FPN. The YOLOv3 feature detector, known as Darknet-53, had 52 convolutions with skip connections, similar to ResNet, and a total of three prediction heads, similar to FPN, allowing it to process images at different spatial compressions.

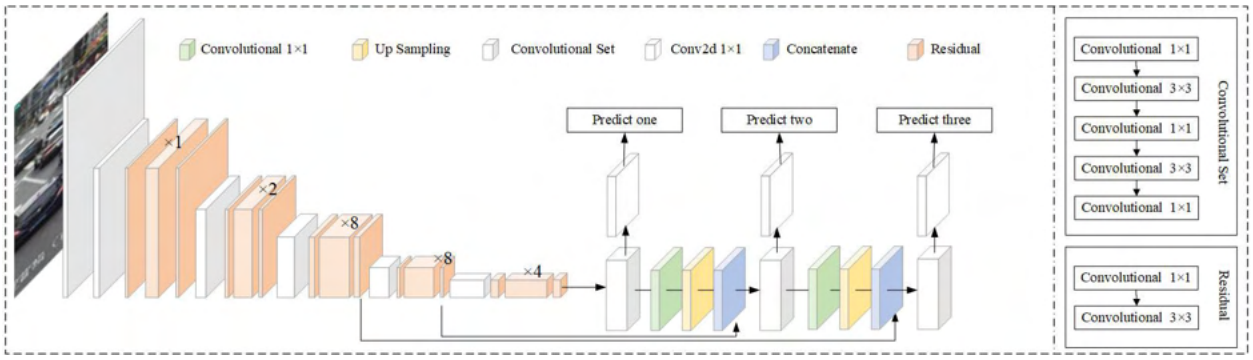


Figure 3.3: Architecture of YoloV3

3.5 SSD Architecture

SSD is a single-stage object detection method that discretizes the output space of bounding boxes into a set of default boxes for each feature map location using different aspect ratios and scales. A backbone model and an SSD head make up an SSD. As a feature extractor, the backbone model is usually a pre-trained image classification network. This is typically a ResNet network trained on ImageNet that has had the final fully connected classification layer removed. As a result, considering the deep neural network that can extract semantic meaning from an input image while preserving the image's spatial structure, albeit at a lower resolution. The backbone of ResNet34 produces 256 7x7 feature maps for an input image. The SSD head is simply one or more convolutional layers added to this backbone, with the outputs being interpreted as bounding boxes and classes of objects in the spatial location of the final layer activations.

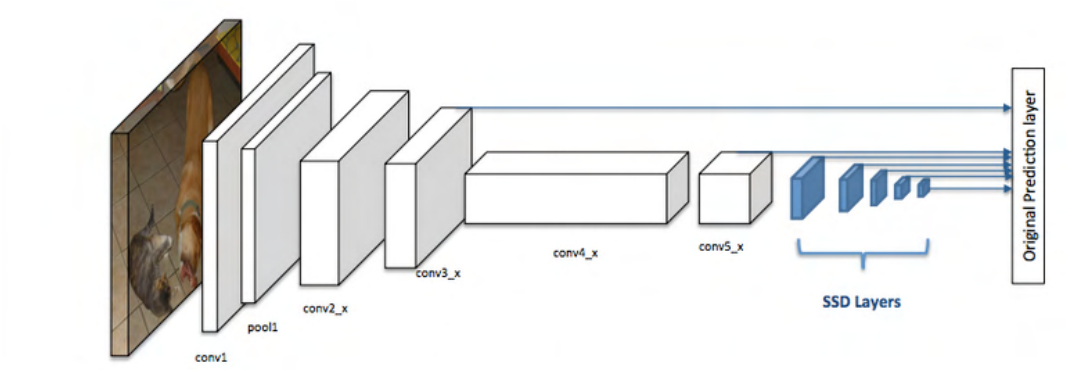


Figure 3.4: Architecture of SSD

3.6 Flask API

Flask is a Python-based web application framework. It is developed by Armin Ronacher, who leads the Pocco international group of Python enthusiasts. The Flask template engine is based on the Werkzeug WSGI toolkit and the Jinja2 template engine. Both are Pocco initiatives. Flask typically requires Python 2.6 or higher for installation. Although Flask and its dependencies are compatible with Python 3 (versions 3.3 and up), many Flask extensions do not. As a result, it is suggested that Flask be installed on Python 2.7. Flask is a Python API that allows us to create web-based applications. Armin Ronacher was the creator. Flask's framework is more explicit than Django's, and it's also easier to learn because it requires less base code to create a simple web application. The Flask template engine is based on the WSGI (Web Server Gateway Interface) toolkit and the Jinja2 template engine. Flask was made to be simple to use and expand. Flask's goal is to provide a solid foundation for web applications of various levels of complexity. After that,can easily plug in any extensions which is in need and also free to create own modules. Flask is suitable for a wide range of projects. It's particularly useful for prototyping. The Jinja2 template engine and the Werkzeug WSGI toolkit are two external libraries that Flask relies o

Because Flask is lightweight and modular, it's simple to turn it into the web framework needed with a few extensions. The documentation for Flask is thorough, well-organized, and full of examples which can even tried out a sample application to get a better understanding of Flask. Flask

is incredibly simple to deploy in production (it's "100The configuration is even more flexible than Django's, allowing to find a solution for any production requirement.

3.7 Software Requirement and Specification

The software used for the project:

• Python

• Visual Studio Code

• Jupyter Notebook

• System Specification

3.7.1 Python

Guido Rossum created Python, an object-oriented programming language, in 1989. It's ideal for quickly prototyping complex applications. It's protractile to C or C++ and has interfaces to several OS system calls and libraries. NASA, Google, YouTube, BitTorrent, and other large companies use the Python programming language. Python programming is widely used in fields such as artificial intelligence, natural language generation, neural networks, and other advanced computer science. Python is a high-level programming language that was created by Guido van Rossum in the late 1980s and is now managed by the Python Software Foundation.

It was inspired by the ABC language, which he helped create early in his career. Python is a powerful programming language that can be used to create games, GUIs, and web applications. It's a very advanced language. Python allows to read and write code in the same way that in English. As a result of not being written in a machine-readable language, Python programmes must be processed before being executed by machines. Python is a well-known programming language. This means that the interpreter runs through the code and converts it to machine-readable byte code each time the programme is run.Python is an object-oriented programming language that allows users to create and run programmes by controlling data structures and objects. Python has it all. Python places equal importance on all objects, data types, functions, methods, and classes. Programming languages are designed to meet the needs of programmers and users who need a quick way to build applications that affect people's lives, lifestyles, economies, and societies.

They help people live better lives by improving productivity, communication, and potency. When languages fall short of expectations, they die and are replaced by more powerful languages. Python is a programming language that has stood the test of time and is still used by programmers and individual users across industries and businesses. It's a living, thriving, and extremely useful language that comes highly recommended as a primary programming language for those who want to get started with programming.

3.7.2 Visual Studio Code

Microsoft's Visual Studio Code (also known as VS Code) is a free open source text editor. For Windows, Linux, and macOS, VS Code is available. VS Code includes some powerful features that have made it one of the most popular development environment tools in recent years, despite its light weight. Visual Studio Code combines the ease of use of a source code editor with advanced developer features such as IntelliSense code completion and debugging. Visual Studio Code is built around a lightning-fast source code editor that is ideal for everyday use. Syntax highlighting, bracket matching, auto-indentation, box selection, snippets, and more are all available in VS Code, which supports hundreds of languages. Easy customization, intuitive keyboard shortcuts, and community-contributed keyboard shortcut mappings make it simple to navigate code. Use tools that understand more code than just blocks of text for serious coding. IntelliSense code completion, rich semantic code understanding and navigation, and code refactoring are all built into Visual Studio Code.

Visual Studio Code combines the best of web, native, and language-specific technologies in its architecture. VS Code uses Electron to combine web technologies like JavaScript and Node.js with native app speed and flexibility. The same industrial-strength HTML-based editor that powers the "Monaco" cloud editor, Internet Explorer's F12 Tools, and other projects is used in VS Code. VS Code also uses a tools service architecture that allows it to integrate with many of the same technologies that power Visual Studio, such as Roslyn for.NET, TypeScript, the Visual Studio debugging engine, and others.

Instead of Anaconda or Miniconda, can use pip to create a Python virtual environment and install the necessary packages for the tutorial. Pandas, jupyter, seaborn, scikit-learn, keras, and tensorflow

are needed.

3.7.3 Jupyter Notebook

The Jupyter Notebook is a free and open source web application that lets us create and share documents with live code, equations, visualisations, and text. Project Jupyter is in charge of maintaining Jupyter Notebook. Jupyter Notebooks is a fork of the IPython project, which used to have its own IPython Notebook project. Jupyter gets its name from the three main programming languages it supports: Julia, Python, and R. Jupyter comes with the IPython kernel, which allows to write Python programmes, but there are currently more than 100 other kernels available. Jupyter notebooks are designed to make code used in digitally supported research or pedagogy more accessible. Jupyter notebooks, for example, are less useful to learn or teach about in isolation because they don't do anything to directly advance research or pedagogy. As an open-source environment compatible with a variety of programming languages, Jupyter has gained traction in a variety of fields. The name Jupyter refers to the project's three core languages (Julia, Python, and R), but kernels exist that make Jupyter compatible with dozens of others, including Ruby, PHP, Javascript, SQL, and Node.js. Although it may not be practical to implement projects in all of these languages using Jupyter notebooks (for example, Omeka will not let allow to install a Jupyter notebook), the Jupyter environment can still be useful for documenting code, teaching programming languages, and providing students with a space where they can easily experiment with provided examples. Jupyter Notebook (not to be confused with the Jupyter notebook files themselves, which have an.ipynb extension) and the newer Jupyter Lab are the two major environments for running Jupyter Notebooks as of late 2019. Jupyter Notebook is a popular and well-documented programme that includes a simple file browser as well as a workspace for creating, editing, and running notebooks. Jupyter Lab is more complicated, with a user interface that resembles an Integrated Development Environment (discussed in previous Programming Historian tutorials for Windows, Mac, and Linux). The Jupyter Notebook is useful not only for learning and teaching Python, but also for sharing data can make a slideshow out of Notebook or share it on GitHub. Binder is a useful tool for sharing a Notebook without requiring users to install anything. Google Colaboratory and Microsoft Azure Notebooks each have their own version of the Notebook that can use to create and

share Notebooks and can also look through some really interesting Notebooks there. JupyterLab, Project Jupyter's newest product, was just released. JupyterLab combines Jupyter Notebook with an Integrated Development Environment (IDE) that runs in browser. JupyterLab can be thought of as a more advanced version of Jupyter Notebook. In addition to Notebooks, JupyterLab allows to run terminals, text editors, and code consoles in the browser.

3.8 System specification

• Program logic : Python 3.7.9
• IDE : VS Code
• Browser : Chrome
• Designing tool : JS, CSS, HTML
• Web framework : flask
• Packages : Keras,Tensorfolw 2.8.0

3.9 Whole System Architecture

• Significance of voting
• Voting Strategies
• Non Maximum Suppression algorithm

3.9.1 Significance of voting

The voting approach is used to improve detection. Three methods were employed to detect the items in the original image, and the ensemble method was combined with a voting strategy to eliminate some of those methods. The overlapping regions are grouped in the first phase of our ensemble approach. Following that, a voting strategy is used to eliminate some of the methods. The final predictions are obtained using the NMs algorithm.

3.9.2 Voting Strategies

Affirmative : The affirmative strategy reduces the number of objects that are not detected (false negatives), because some objects that are not detected by one approach may be detected by another, but it increases the number of incorrect detection (false positives), due to the accumulation of false positives obtained by each approach.

Unanimous : The unanimous technique decreases false positives while increasing false negatives. Because all of the first detection approaches must agree to detect an object.

Consensus : The most popular voting approach is consensus. Techniques like affirmative and unanimous may appear to be too flexible and restrictive to be effective. However, depending on the detection models' efficacy, they may produce better results than consensus, but consensus avoids false positives, making it the best option.

3.9.3 Non Maximum Suppression algorithm

NMS is a technique that is utilised in a variety of computer vision tasks. It is a group of algorithms that selects one entity (for example, bounding boxes) from a large number of overlapping entities. To get the desired results, we can specify the selection criteria.

Chapter 4

RESULTS AND DISCUSSIONS

An automated system for VEHICLE detection has a significant impact on autonomous driving and helps to make current traffic conditions much better and efficient for everyday life, where better detection ensures better road safety. Convolutional Neural Networks (CNN) have proven to be quite successful in vehicle identification, and the addition of Region Proposal Networks (RPN) improves and speeds up the process. Also, image classification can be done with pre-trained CNN models, and the model's efficiency is high.
After the model is built, the Non-maximum Suppression (NMS) algorithm is used to make the final ensemble predictions. The ensemble algorithm can be used with three different voting strategies:

- Affirmative:- This means that a detection is valid if one of the methods used to generate the initial predictions says that a region contains an object.
- Consensus:- This requires the majority of the initial methods to agree that a region contains an object. The consensus strategy is similar to the majority voting strategy used in ensemble image classification methods..
- Unanimous:- This indicates that all methods must agree that a region contains an object.

4.1 Prediction results

The following are the predictions obtained using different models by detecting a single vehicle object with the same image input:

Figure : Input image for the below predictions

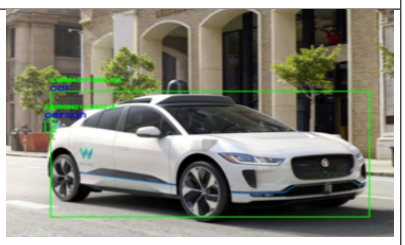
SSD	RetinaNet	YoloV3
		

Table 4.1: Detection by three algorithms

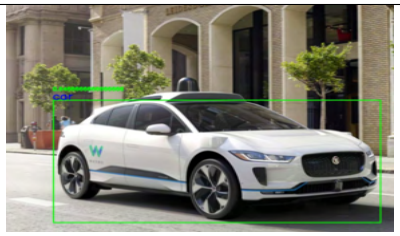
Affirmative	Consensus	Unanimous
		

Table 4.2: Ensemble by three voting strategies

Affirmative:- Accuracy of 92% for SSD (Object misdetected), 94% RetinaNet, 99% for YoloV3 and ensemble failed in correct detection according to the voting strategy.

Consensus:- Accuracy of 92% for SSD (Object misdetected), 64% RetinaNet, 99% for YoloV3 and an ensemble accuracy of 64% according to the voting strategy.

Unanimous:- Accuracy of 92% for SSD (Object misdetected), 94% RetinaNet, 99% for YoloV3 and ensemble predicted nothing according to the voting strategy.

The given table 4.1 shows triumvirate models prediction accuracy.

Algorithms			Ensemble result according to voting		
SSD	**Retina-Net**	**YoloV3**	**Affirmative**	**Consensus**	**Unanimous**
Wrong prediction	Predicted 94%	Predicted 99%	Correct Prediction + Wrong Prediction	Predicted 64%	Nothing Predicted

Figure 1 : Accuracy obtained according to different strategies

4.2 Graphs

Figure 1: Input image for the below graphs

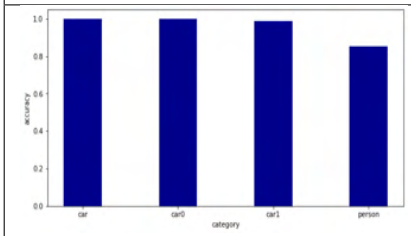
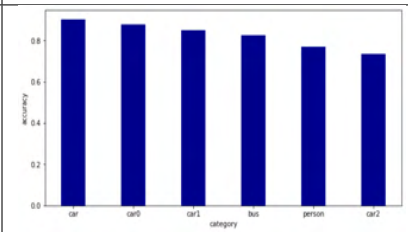
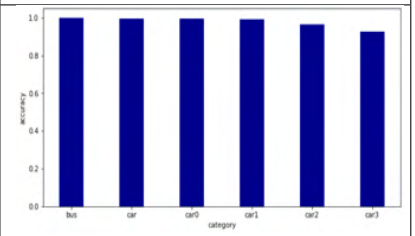
SSD	RetinaNet	YoloV3
		

Table 4.3: Detection using different Voting Strategies

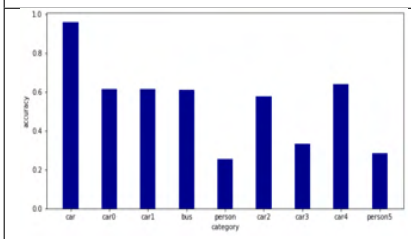
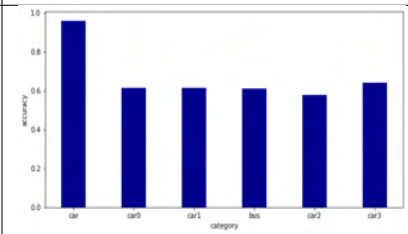
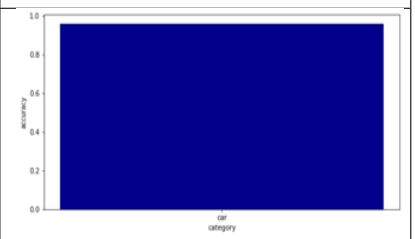
Affirmative	Consensus	Unanimous
		

Table 4.4: Detection using different Voting Strategies

4.3 Equations used

The task of determining the position and category of multiple objects in an image is known as object detection. Given an image I, returns a list of detections D = [d1,..., dN], with each di represented by a triple [bi, ci, si] that includes a bounding box, bi, the corresponding category, ci, and the corresponding confidence score, si,

The ensemble algorithm takes as input a list LD = [D1,..., Dm]. Each Di is usually derived from detection mode predictions, and each list is a list of detections for a given image. Di is a list of detections produced by a specific method Mi for a given image, the list LD is flattened into a list F = [d1,..., dk].Following that, the elements di of F are grouped together based on their bounding boxes and classes overlapping.The IoU metric is used to determine whether or not bounding boxes overlap. The IoU formula for finding the overlapped region between two bounding boxes, b1 and b2, is as follows:

$$IoU(b_1, b_2) = \frac{area(b_1 \cap b_2)}{area(b_1 \cup b_2)}$$

4.4 Testing and training

In this project Pre-trained datasets are used which consist of 178 images in training set and 60 images in testing set.

4.5 Library used

Implemented open-source library called EnsembleObjectDetection in Python and relies on several third-party libraries like Numpy and OpenCV.The library has been used with the Darknet and MxNet frameworks for any pre-trained model. Implemented an abstract class called IPredictor with a predict method that takes a folder of images as input and outputs XML files with predictions for each image in the input folder. The predict method's output can be used for ensembling and TTA with any detection model.

Chapter 5

CONCLUSION

As a result, it can be concluded that the proposed deep learning model accurately classifies and detect vehicles with an average accuracy of more than 90 percent. Deep learning models such as Retina-net, Single Shot Detector, and Yolov3 are used to automatically detect vehicles and obtained optimum accuracy, where the three methods were employed to detect the items in the original image, and a voting strategy is used to eliminate some of those methods and the final predictions are obtained using the NMs algorithm. The data augmentation and pre-processing stages ensure that convolutional neural networks and deep neural networks do not suffer from overfitting, ensuring that the results obtained are always consistent. With fewer convolutional layers, the proposed model accurately predicts whether a given sample road scenario contains vehicles or other road users such as humans, animals, and so on. This is extremely useful in the field of vehicle automation for accurate vehicle detection in various environments. In autonomous driving, faster and more accurate vehicle detection aids better and safer transportation.

5.1 Advantages

The main merits of proposed model are:

- Improves detection of vehicles in different lighting conditions.
- Increased motorcycle detection accuracy.
- Classifies light and heavy vehicles.

5.2 Future Enhancement

This project has a lot of potential in the field of autonomous driving, and it can be expanded to include other innovative ideas. The future work will focus on improving the architectures used, as well as other deep learning models, in order to improve accuracy. This research can be expanded to detect threats in traffic, such as identifying the licence plates of law-breaking vehicles, and aid in the reduction of road crimes. Furthermore, this project can be implemented anywhere in the world with little difficulty.

REFERENCES

[1] Soft-Weighted-Average Ensemble Vehicle Detection Method Based on Single-Stage and Two-Stage Deep Learning Models, 2020

[2] Yongzheng Xu, Guizhen Yu, Member, IEEE, Xinkai Wu, Yunpeng Wang, and Yalong Ma An Enhanced Viola–Jones Vehicle DetectionMethod From Unmanned Aerial Vehicles Imagery,2019

[3] Cai Y, Liu Z, Wang H, et al. Vehicle detection by fusing part model learning and semantic scene information for complex urban surveillance[J]. Sensors, 2018

[4] S. Akcay et al., 'Using Deep Convolutional Neural Network Architectures for Object Classification and Detection Within X-Ray Baggage Security Imagery', IEEE Transactions on Information Fosics and Security, 13(9), 2203–2215, 2018

[5] J. Hosang, R. Benenson, and B. Schiele, 'Learning non-maximum suppression', CoRR, abs/1705.02950, 2017

[6] Cai Y, Liu Z, Wang H, et al. Saliency-based pedestrian detection in far infrared images[J]. IEEE Access, 2017

[7] Mukhtar A, Xia L, Tang T B. Vehicle Detection Techniques for Collision Avoidance Systems: A Review[J]. IEEE Trans. Intelligent Transportation Systems, 2015

APPENDIX

A Screenshots

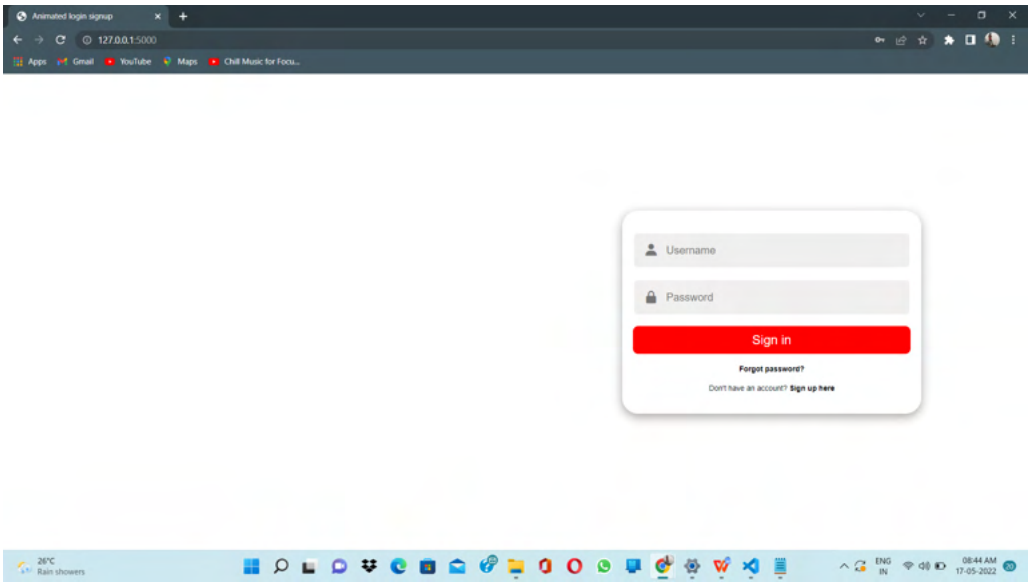


Figure 1 : Login Page

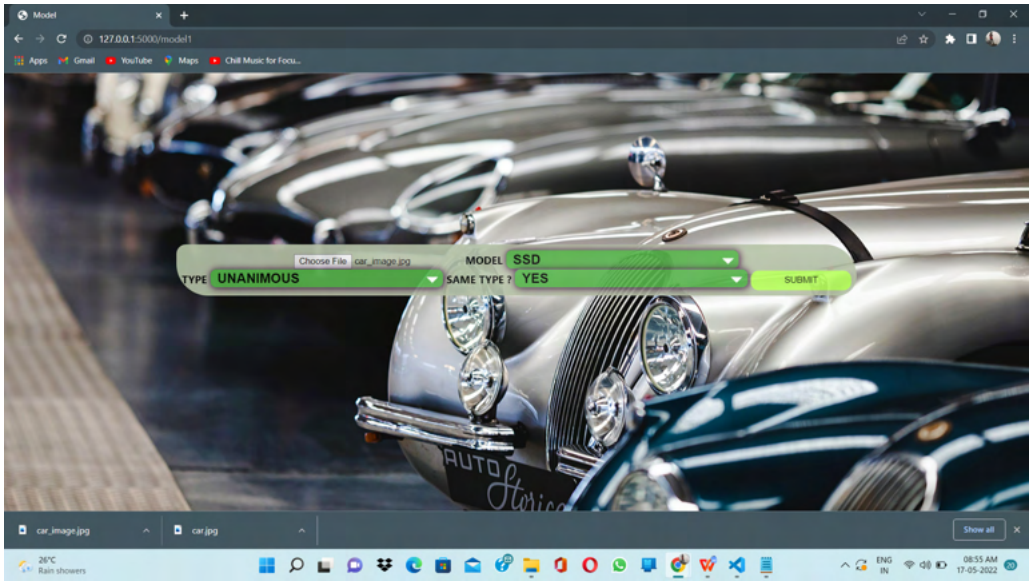


Figure 2 : Home Page

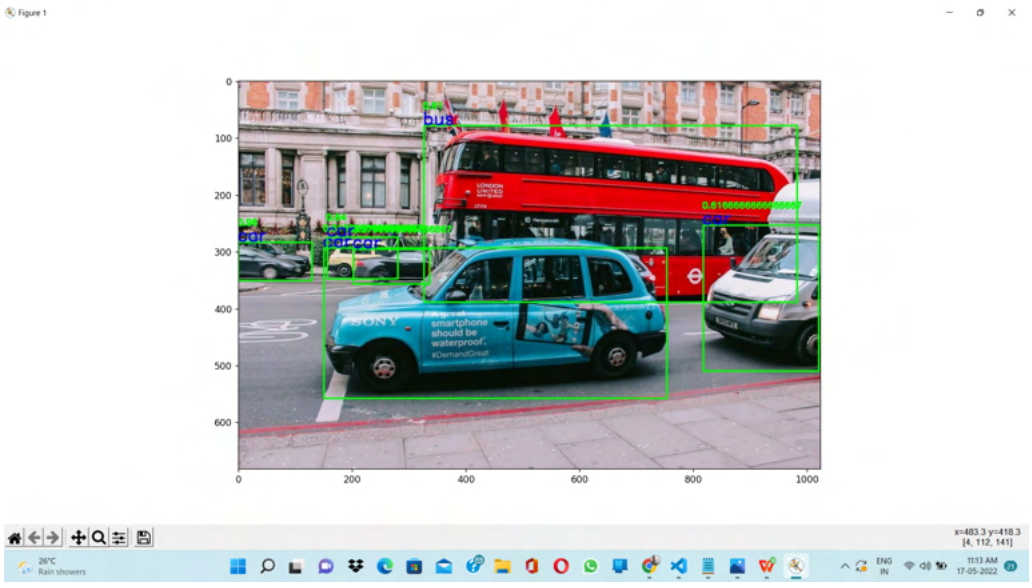


Figure 3 : Single Shot Detector

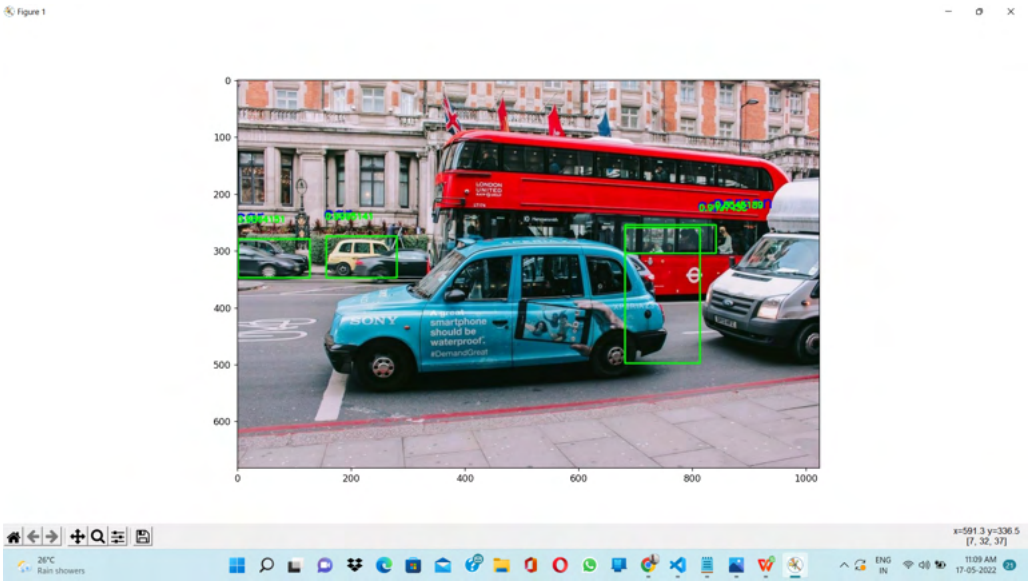


Figure 4 : RetinaNet

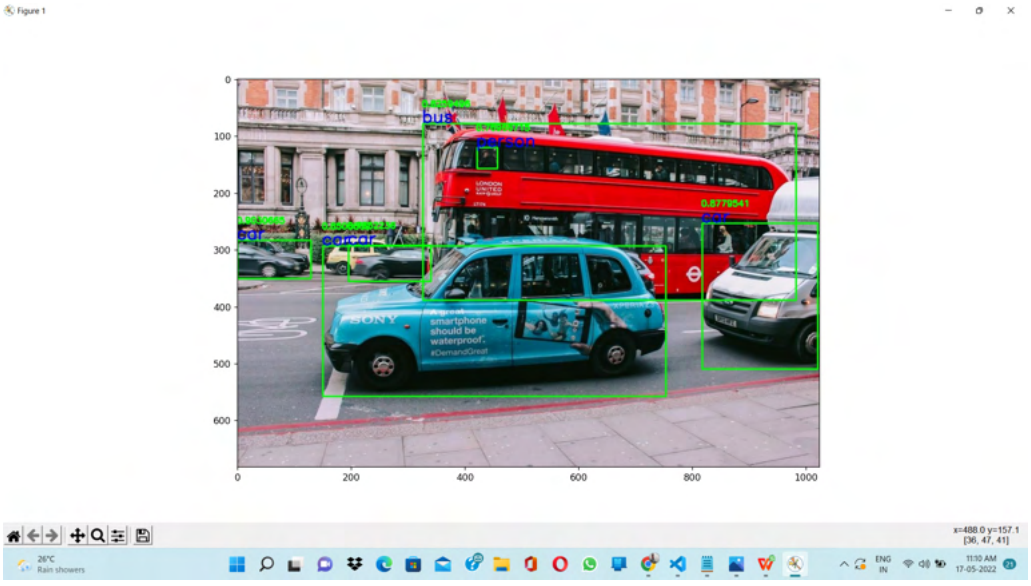


Figure 5 : YoloV3

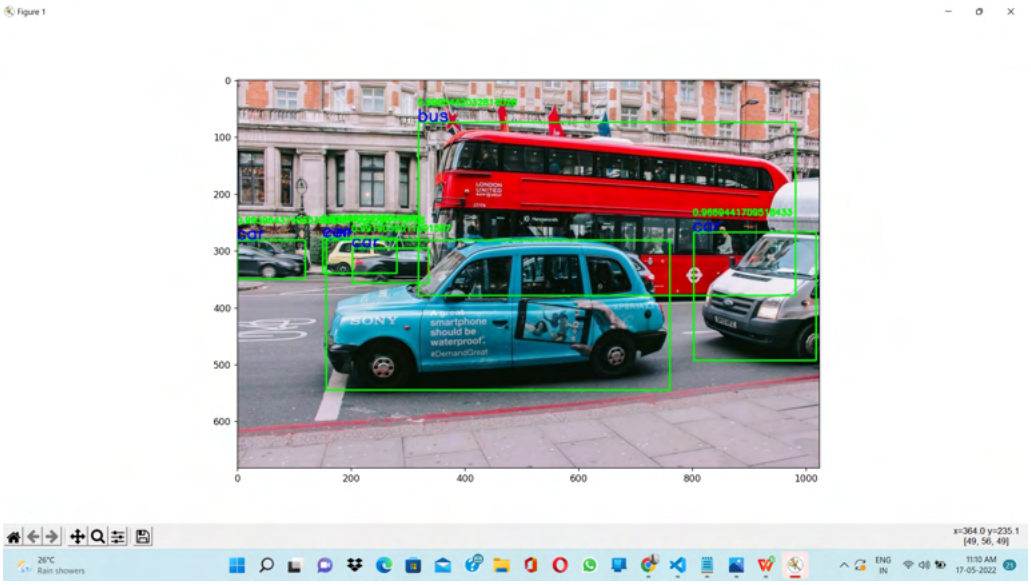


Figure 6 : Unanimous

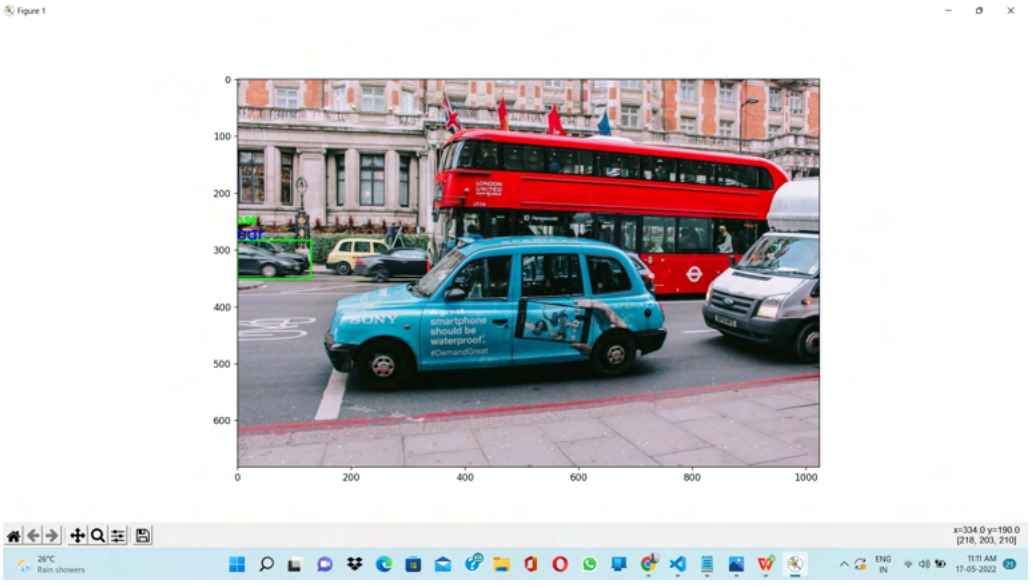


Figure 7 : Affrimative

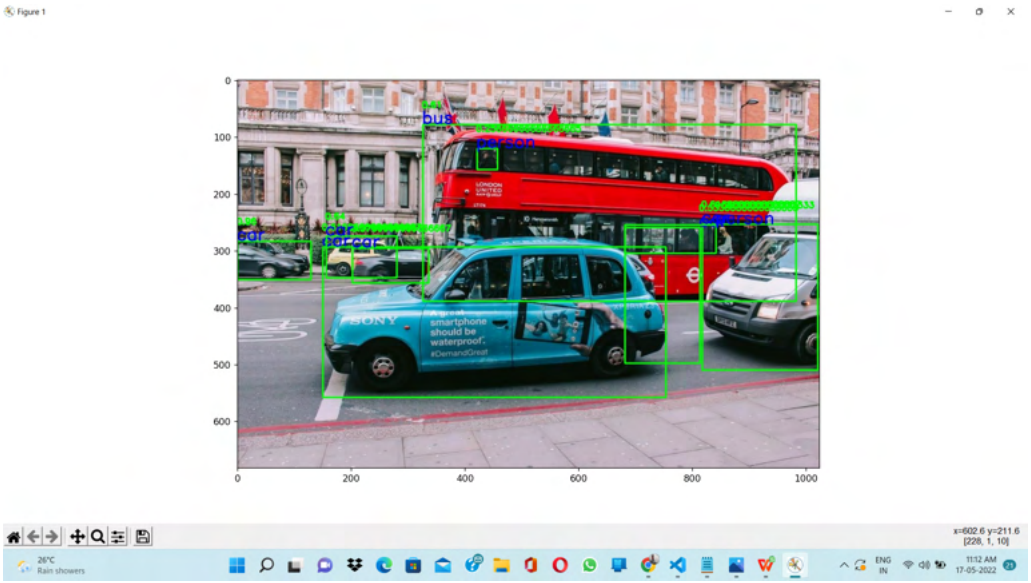


Figure 8 : Consensus