VEHICLE MOVEMENT ANALYSIS AND INSIGHT GENERATION IN A COLLEGE CAMPUS USING EDGE AI

Intel® Unnati Industrial Training Program 2024

Project and Report by: HEMANTH KUMAR A. R.

Contents

1. Introduction

1.1. Problem Statement:

1.1.1. Title:

“Vehicle Movement Analysis and Insight Generation in a College Campus using Edge AI”.

1.1.2. Description:

The increasing number of vehicles on college campuses has led to significant challenges in managing parking and ensuring smooth vehicle movement. Traditional methods of monitoring and managing vehicle flow are insufficient to handle the dynamic and complex nature of modern campus environments.

This results in congestion, unauthorized parking, safety risks, and inefficient use of parking resources. There is a need for a comprehensive solution that leverages Edge AI technology to analyse vehicle movement in real-time, generate actionable insights, and enhance the efficiency and safety of parking management on college campuses.

1.2. Background:

With the proliferation of personal vehicles among students, faculty, and staff, college campuses are facing growing challenges in managing vehicle movement and parking. According to recent surveys, vehicle ownership among college-goers has been increasing by approximately 10% annually. This surge has put immense pressure on existing parking infrastructure, which was often designed decades ago and is not equipped to handle the current volume of traffic. As a result, many campuses experience frequent congestion at entry and exit points, unauthorized parking, and safety hazards.

Traditional parking management systems, which rely on manual monitoring and fixed CCTV surveillance, are increasingly proving inadequate. These systems lack the capability to provide real-time data and insights needed to dynamically manage vehicle movement and parking spaces. This leads to inefficient utilization of available parking resources, extended search times for parking spaces, and increased emissions from idling vehicles.

Edge AI technology presents a promising solution to these challenges. By deploying AI-powered cameras and sensors at strategic points around the campus, it is possible to monitor vehicle movement in real-time and generate actionable insights. Edge AI systems can process

data locally, providing immediate feedback and reducing the need for expensive data transmission to centralized servers. This enables quicker response times and more efficient management of parking resources.

In a pilot study conducted at a major urban college campus, the implementation of Edge AI technology resulted in a 30% reduction in traffic congestion during peak hours and a 20% increase in the efficient use of parking spaces. The system was able to detect and alert authorities about unauthorized parking in real-time, enhancing overall safety and security on campus.

1.3. Objectives

Real-Time Vehicle Monitoring:

- Deploy Edge AI-powered cameras and sensors across the campus to continuously monitor vehicle movement and parking space availability in real-time.

Traffic Flow Optimization:

- Utilize AI insights to design and implement strategies for optimizing traffic flow, reducing congestion at entry and exit points, and minimizing vehicle idling times.

Unauthorized Parking Detection:

- Develop and deploy algorithms to detect unauthorized parking and immediately alert campus security, thereby enhancing enforcement and compliance.

User-Friendly Interface:

- Create a user-friendly web interface for students, faculty, and staff to receive real-time information about parking availability, traffic conditions, and route optimization within the campus.

Actionable Insights Generation:

- Collect and analyse data to generate actionable insights for campus administrators, helping them make informed decisions about infrastructure improvements, policy changes, and resource allocation.

Integration with Campus Systems:

- Ensure seamless integration of the Edge AI system with existing campus security, parking management, and administration systems for a cohesive operational framework.

Scalability and Flexibility:

- Design the Edge AI system to be scalable and flexible, allowing for easy expansion and adaptation to future changes in vehicle volume and campus infrastructure.

By leveraging Edge AI technology for vehicle movement analysis and insight generation, college campuses can address current challenges in parking management and vehicle flow, creating a safer, more efficient, and environmentally friendly campus environment.

2. Dataset

2.1. Collection of the dataset

From,

Hemanth Kumar A. R. (USN – 1SG21EC045),
6th semester, "A" section,
Electronics and Communication department,
Sapthagiri College of Engineering.

Date – 19 June 2024

To,
The Principal,
The Head of Department,
Department of Electronics and Communication,
Sapthagiri College of Engineering.

Subject: Permission to capture license plate images of vehicles entering the college campus as a part of data collection for the problem statement required by Intel Unnati Training 2024.

Respected Sir,

I am Hemanth Kumar A. R. (USN – 1SG21EC045), student of 6th semester "A" section in Electronics and Communication branch, selected candidate for Intel Unnati Training 2024.

The problem statement I am working is **"Vehicle Movement Analysis and Insight Generation in a College Campus using Edge AI"**. As recommended and prescribed method by Intel is to capture license plate images of vehicles entering and leaving college campus for authenticity and to maintain accuracy in vehicle movement analysis and insight generation (reference to Article - Example Solve for Vehicle Movement Analysis and Insight [Page: 1 - 2]).

I plan to take up this operation on 20th June 2024. Please allow me to conduct the necessary action as this is an essential and preliminary step in successfully completing my training from Intel Unnati Training 2024.

Please find the necessary documents attached below.

forwarded for kind permission
as it is a project useful if in collaboration with Intel
19/6/2024

Thank you,
Hemanth Kumar A. R.

Permission taken from the college management to collect images of the vehicles along with the license photo of the students, faculties and any vehicles entering and exiting the college campus for a period of 1 day.

The entire images or the dataset used in this project is solely collected by me instead of relying on online images dataset as this increases the accuracy and to simulate and generate insights live from the college campus.

2.2. Description and key features

The images were taken from the iPhone SE 2020 with 12 MP rear facing wide camera – 28mm with focal length of 1.8 f, which captured each image at a resolution of 4032 X 3024 with average size of 4.2 Mb

The original raw image look like this:

But due to high resolution and bigger size, which proved difficult to train the Object detection algorithm, so the images were scaled down to a resolution of 800 X 600, with average size of each file being just 400 Kb.

This definitely reduced training time which will be discussed later in the report.

The down scaled image which is used throughout the project for training, identification, and for detection looks like the following image as though the difference is barely observable:

Features of the dataset include:

- High clarity images of the vehicle along with License plate.
- Consists of images of live vehicles entering / exiting the college campus.
- The images contain the date and time as the timestamp at bottom left of the image, which is used later to record time of each vehicle.
- It contains images of both 2-wheeler as well as 4-wheeler.
- License Plate images are taken with various angles under different lighting conditions which really tests the model efficiency.
- Contains images of license plate with various shapes like rectangle and square.
- The font of the license plate varies in terms of colour, size, spacing and font itself.
- The images contain license plate of varying colour like White board, Green board and Yellow board.
- Few images contain the license plate with incomplete registration number details, which directly reflects on the performance of the trained model.
- Images are named in the chronological order.

2.3. Technologies used

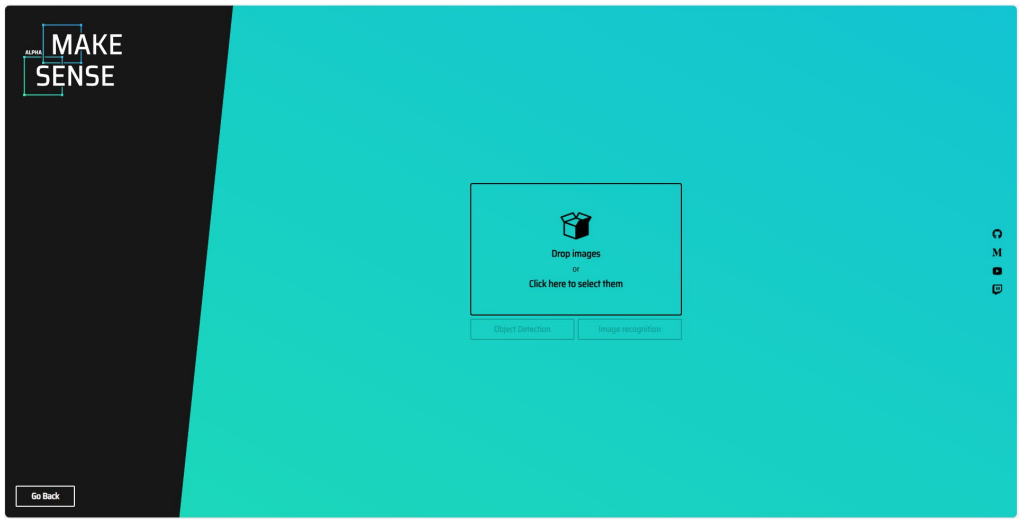
- Python
- OpenCV
- NVIDIA CUDA Graphics Engine
- Pandas
- Numpy
- Yolo
- Pytesseract
- PyQt5
- Pyexcel-ods3
- Flask
- HTML
- Plotly
- Pillow
- Google Colab
- Git
- PyCharm

3. Methodology

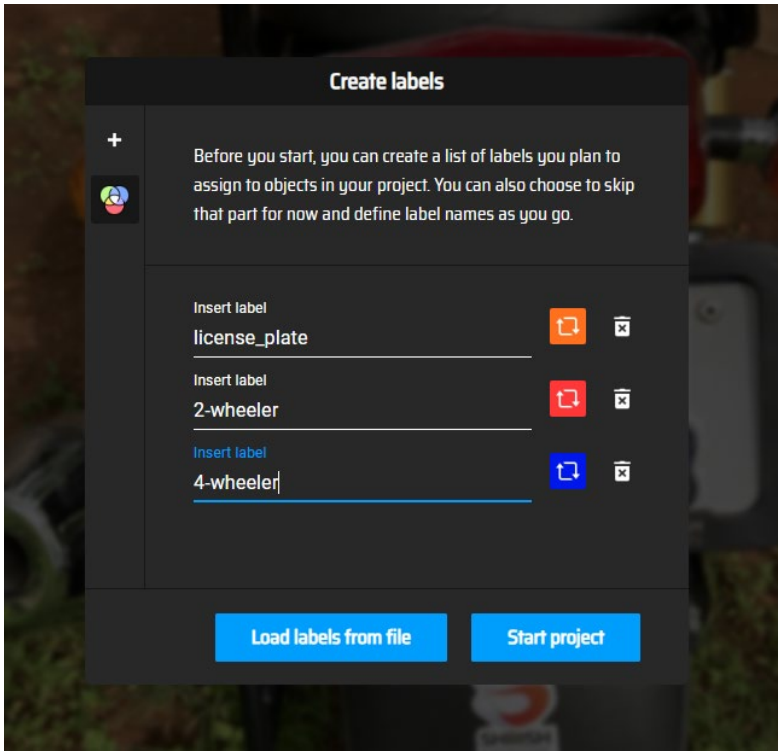
3.1. Annotation of Images

To annotate the images collected, I have made use of the website: https://www.makesense.ai/ which is easy to upload the images and annotate images online, without a need to install an additional software in the system locally.

Step 1: Upload the image.

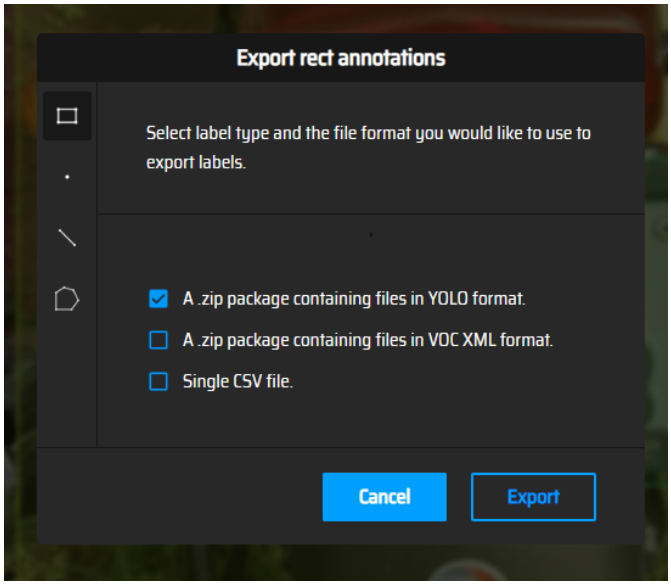


Step 2: Select the labels.



Step 3: Annotate the bounding box.

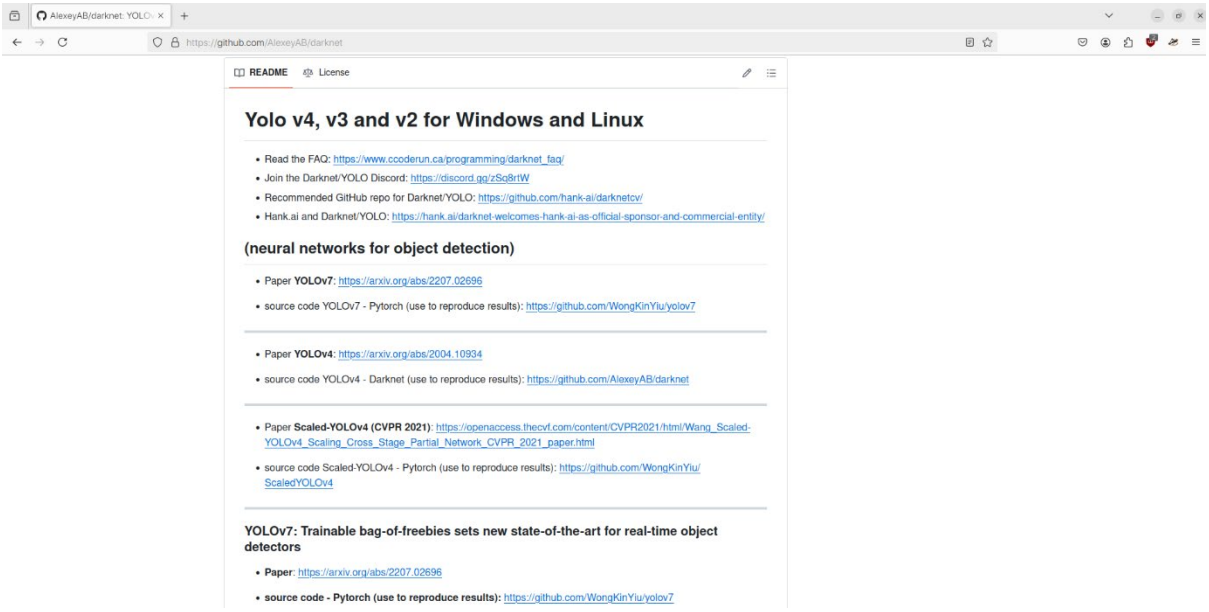
Step 4: Export the annotated image in the YOLO format.



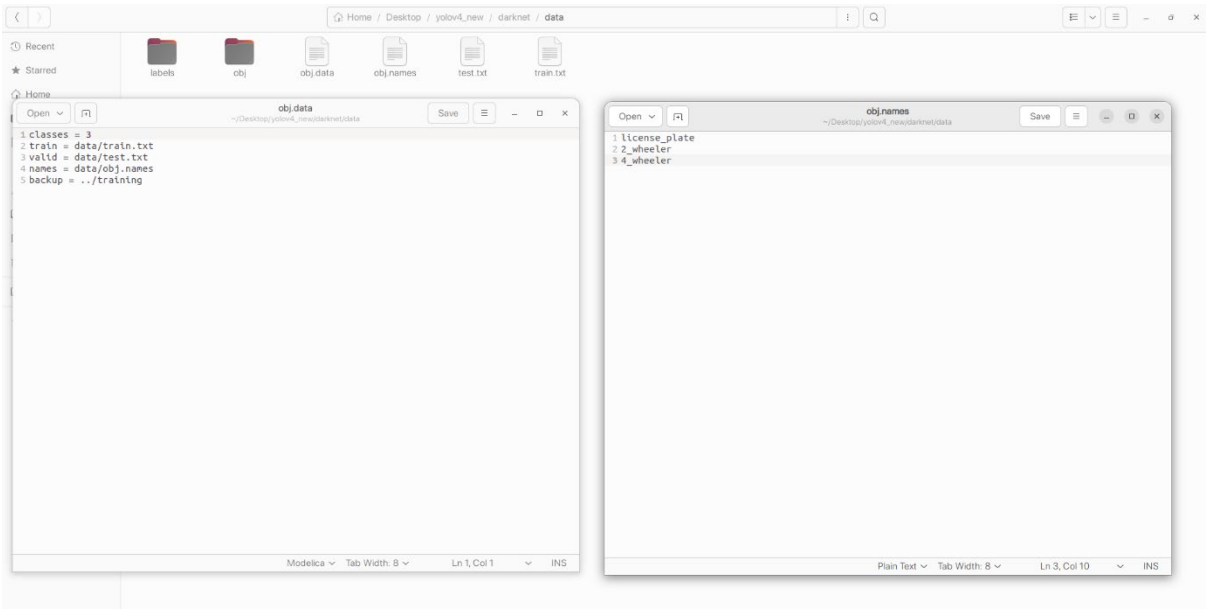
Step 5: Continue the process for all the images.

3.2. Training of the YOLOv4 deep learning algorithm model

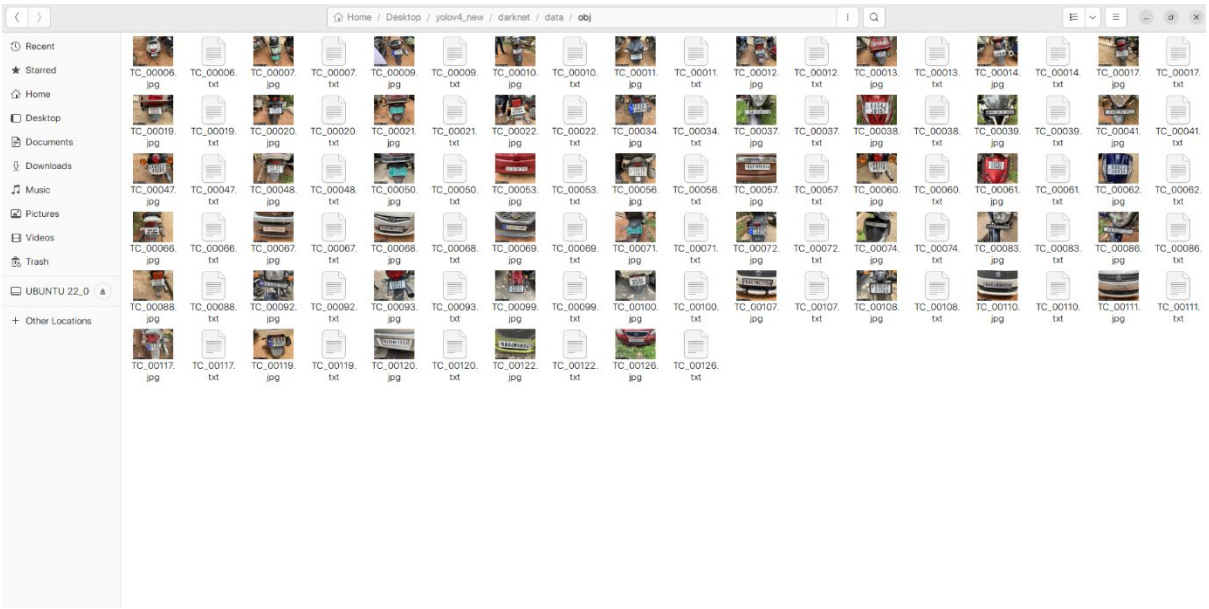
Step 1: Cloning the Yolov4 model from https://github.com/AlexeyAB/darknet



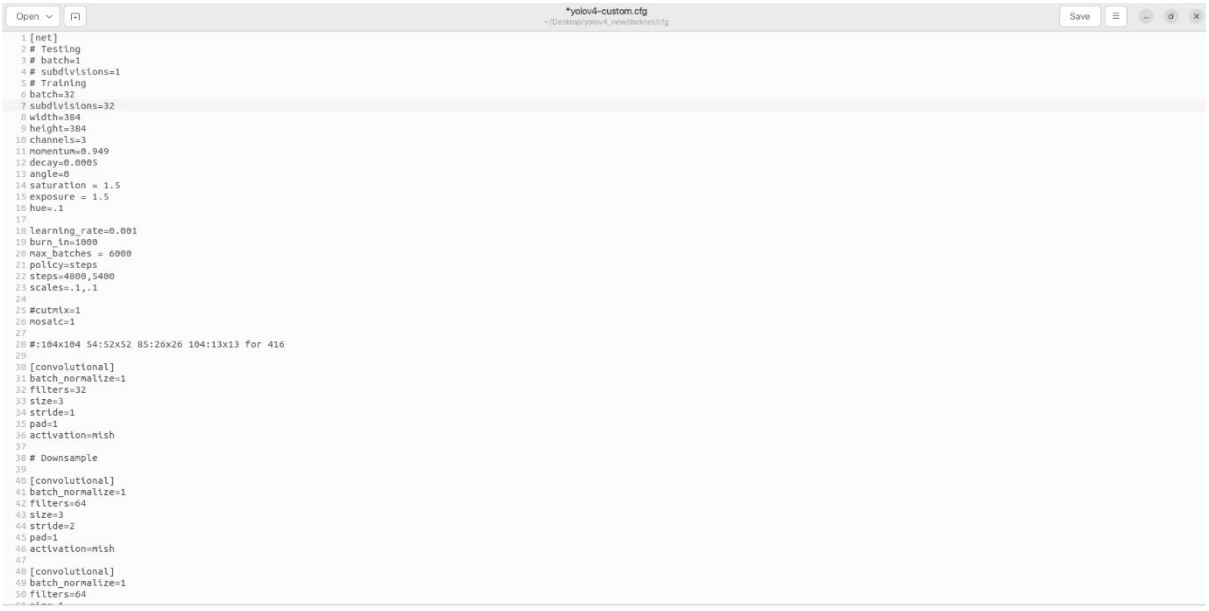
Step 2: Include obj.data and obj.txt file which contains information of number of classes and labels of each image in the data folder of darknet directory under yolo



Step 3: Include annotated files with the same name as the images and place them both under the obj directory under darknet directory.



Step 4: Edit the config file

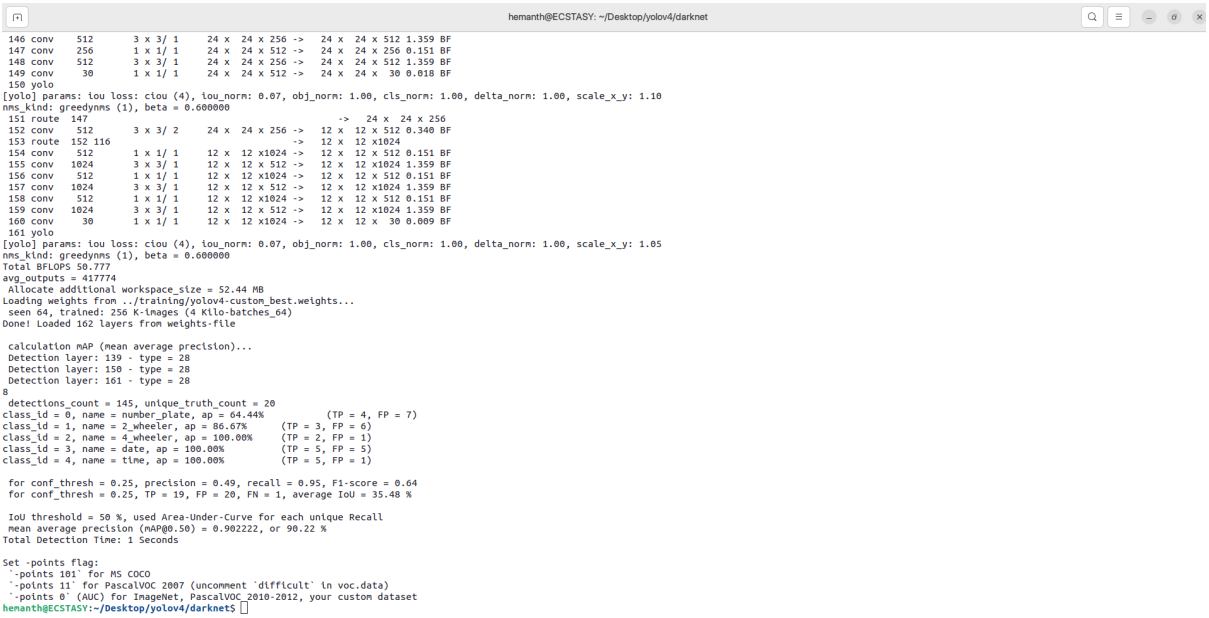


Here, we give batch as 32 which the model takes 32 images at a stretch and runs the algorithm

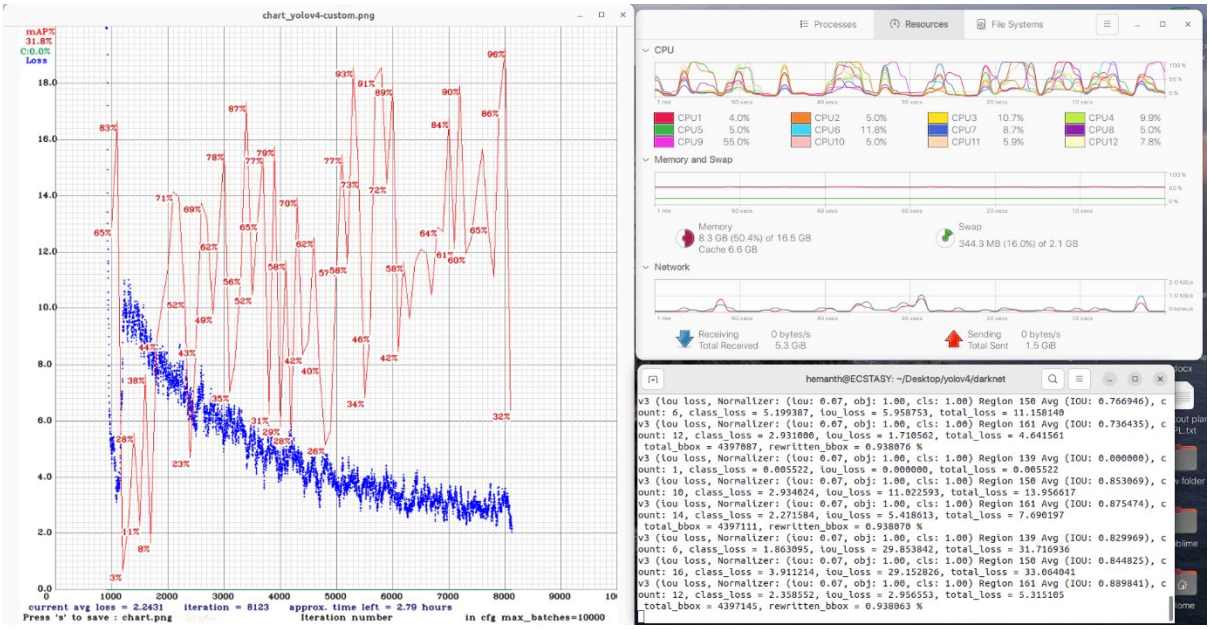
Next is subdivisions as 32, this will divide each image into 32 different grid, and the training of the model takes place.

Also, max_batches is taken as 6000, so the model, goes for 6000 of total iterations along with other few parameters.

Step 5: Train the model (it takes around 10 hours to finish)



Step 6: Viewing initial, performance vs error rate graph

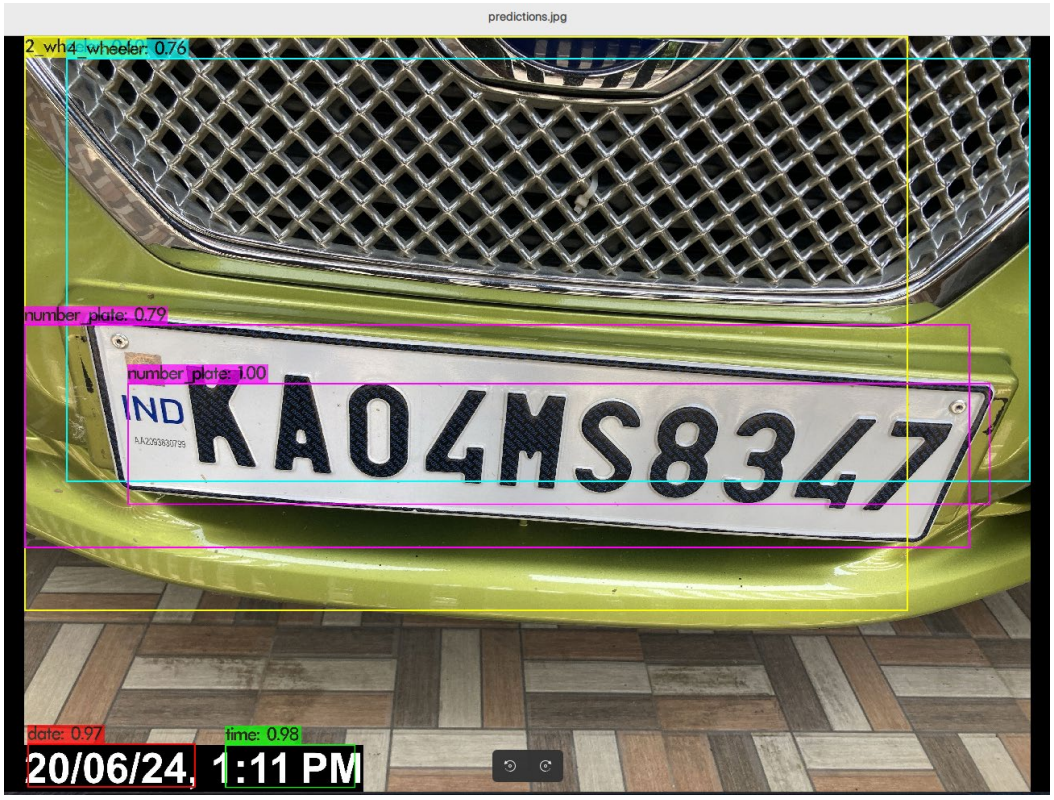


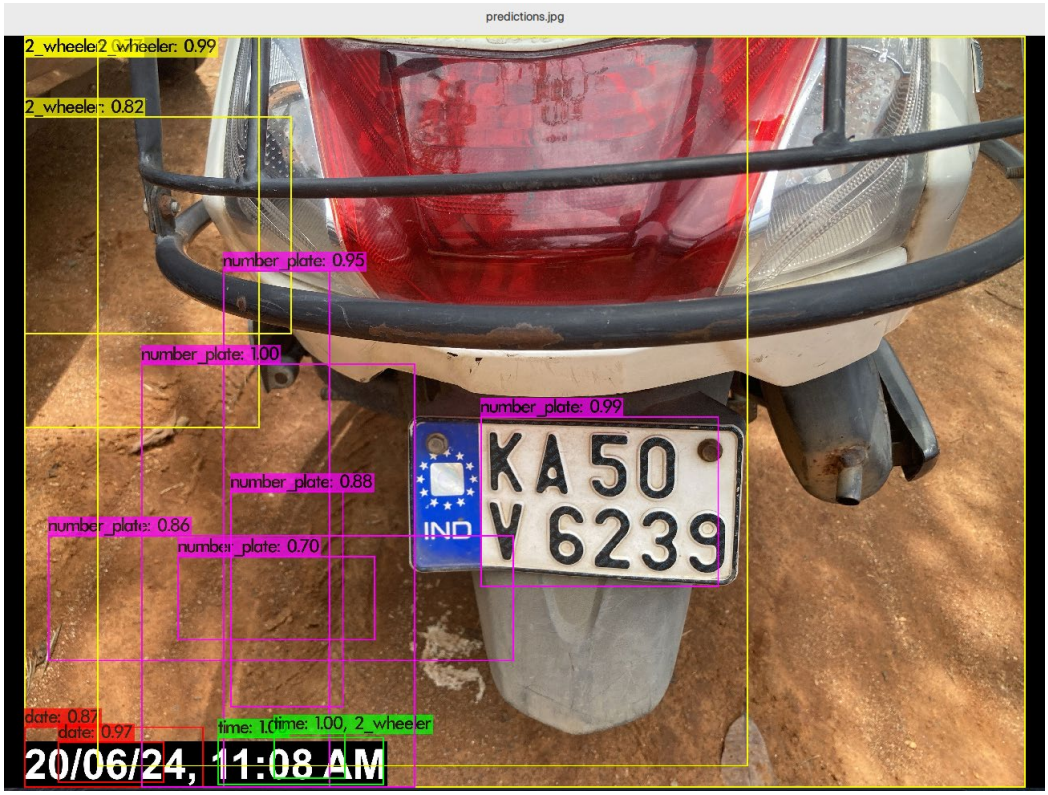
Here, the model performed poorly with lot of fluctuating parameters which are undesirable characteristics and they reduce the model efficiency.

As the number of classes increased, like around 5 for first run, the prediction rate decreased.

In the next run, the classes are dropped to important 3, which are license_plate, 2-wheeler, 4-wheeler.

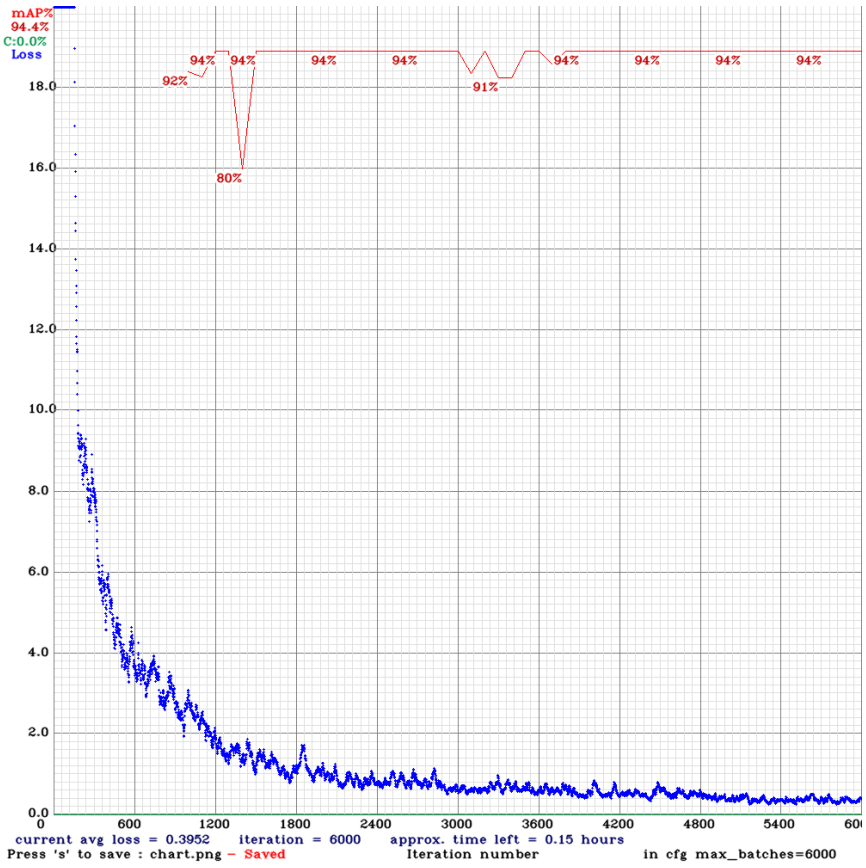
Step 7: View initial predictions





The model performed poorly with multiple false predictions and cropping of the important data.

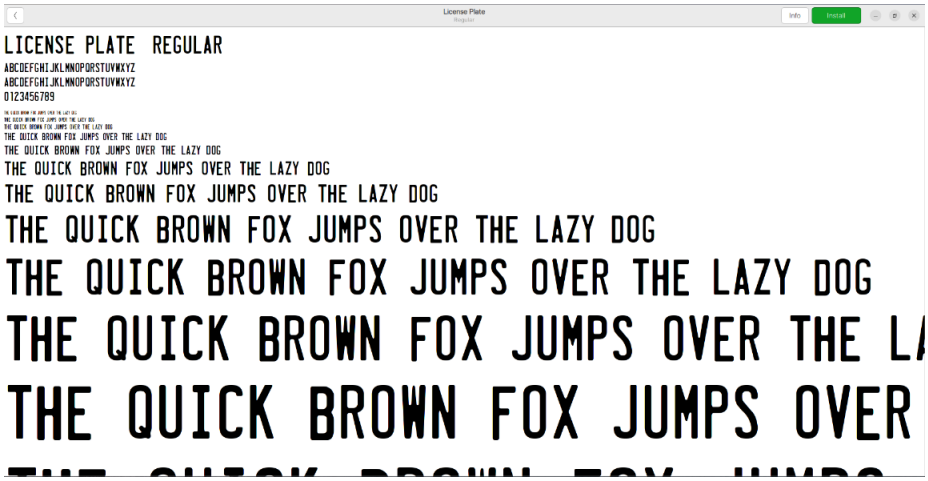
Step 8: Re-run the model adjusting the model classes

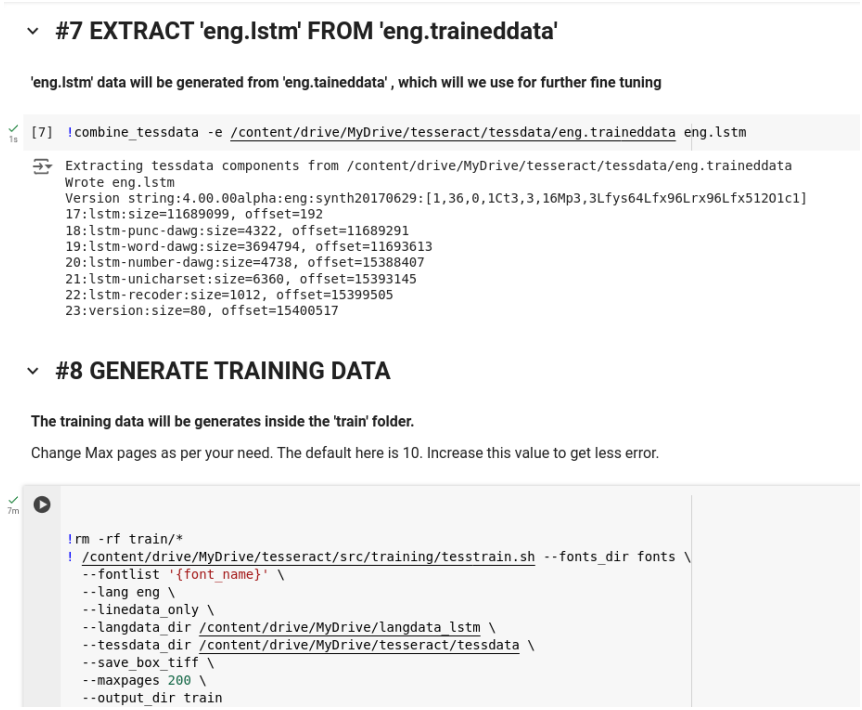


The performance chart looked healthy, with minimal fluctuations and high accuracy of around 94%.

The predictions too proved the point, as the prediction confidence of the objects where noticeably higher around 1 or 0.9 and no cropping of images.

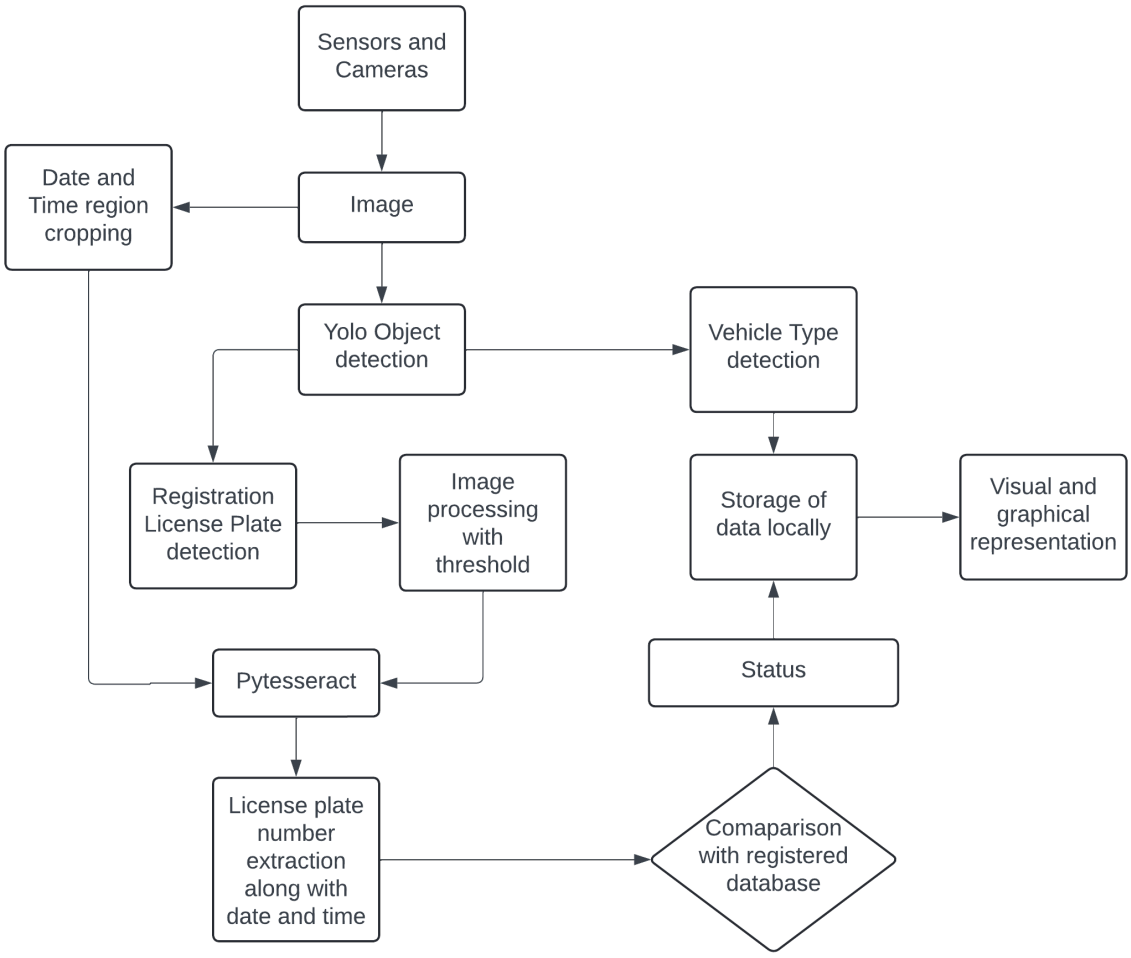
Step 9: Train the Pytesseract OCR model to recognize the custom font in the Indian HSRP (High Security Registration Plate) using the custom font I have downloaded online and trained in Google Colab here: https://colab.research.google.com/github/AniqueManiac/new-font-training-with-tesseract-in-google-colab/blob/main/TrainAnewFontWithTesseract.ipynb#scrollTo=Idoqy3pLnuWa





Step 10: Place all the config files and the weights generated by Yolo and Pytesseract into the main directory.

3.3. Architecture Diagram



3.4. Comparison with the database

The license plate number obtained in the process is checked with “registered_vehicles.ods” file which contains the list the registered license plate number.

This helps us to identify whether the vehicle belongs to the college and is registered or not.

It is also helpful during the times of high vehicle occupancy of the parking space, as these registered vehicles must be given a priority.

The output is returned as a status code – “Yes” or “No” which is stored along with the vehicle information.

3.5. Storage of the Information

All the relevant information about the vehicle is recorded at both entry and exit and is stored locally in the “vehicle.ods” file

The information it contains include:

- Date
- Vehicle Type
- License Plate Number
- Status of the Registration
- Entry Time
- Exit time
- Parking Space Occupied / Filled count
- Parking Space Available / Free count

4. Results

The intermediate images can be viewed as follows:

4.1. Object identification using Yolov4

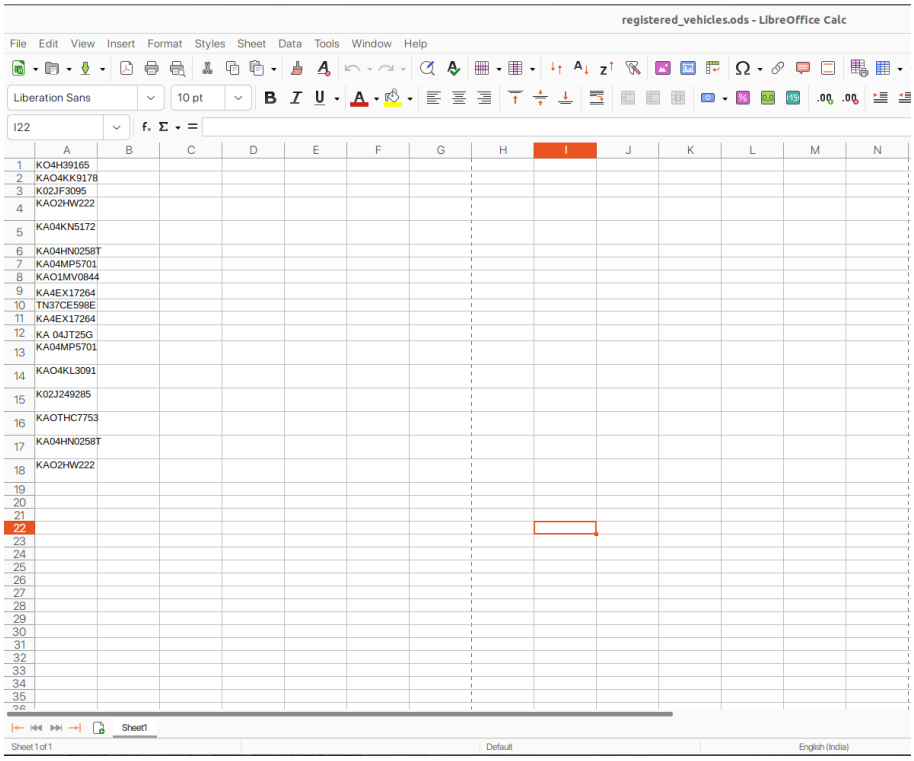
Here, the Yolov4 successfully identifies the vehicle type such as 2-wheeler or 4-wheeler along with the co-ordinates of the license plate.

4.2. Applying necessary filters to the image by OpenCV

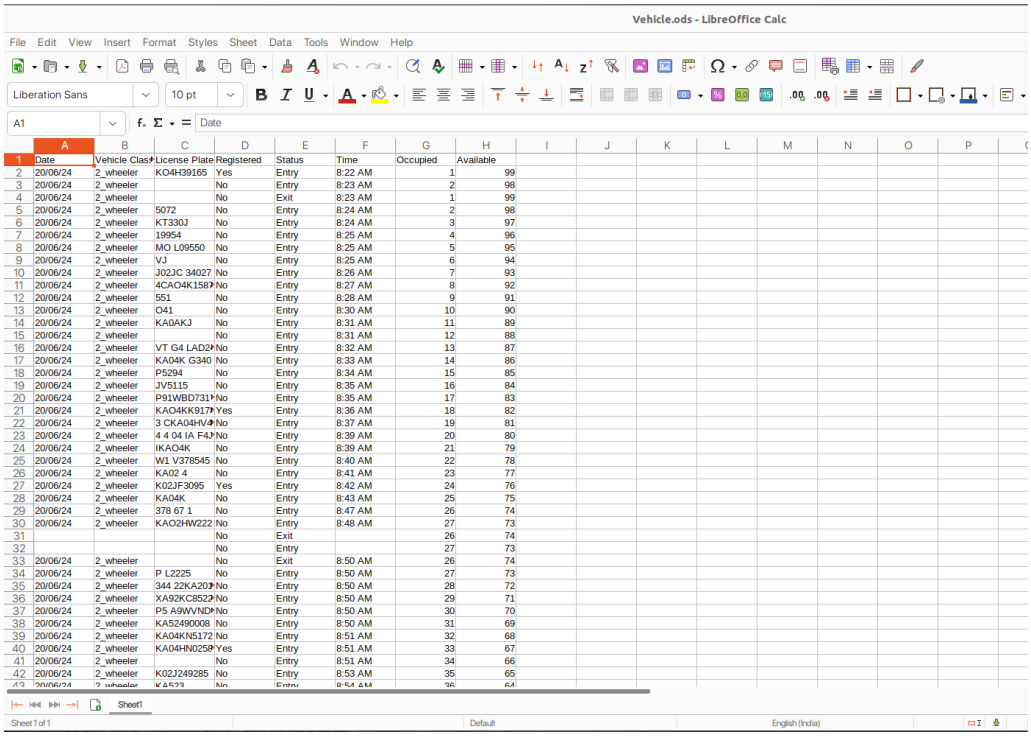
4.3. Crop the filtered image using the bounding box co-ordinates

4.4. Crop the region of date and time and pass it to the Pytesseract

4.5. Check the obtained license plate number in the “registered_vehicle.ods” file

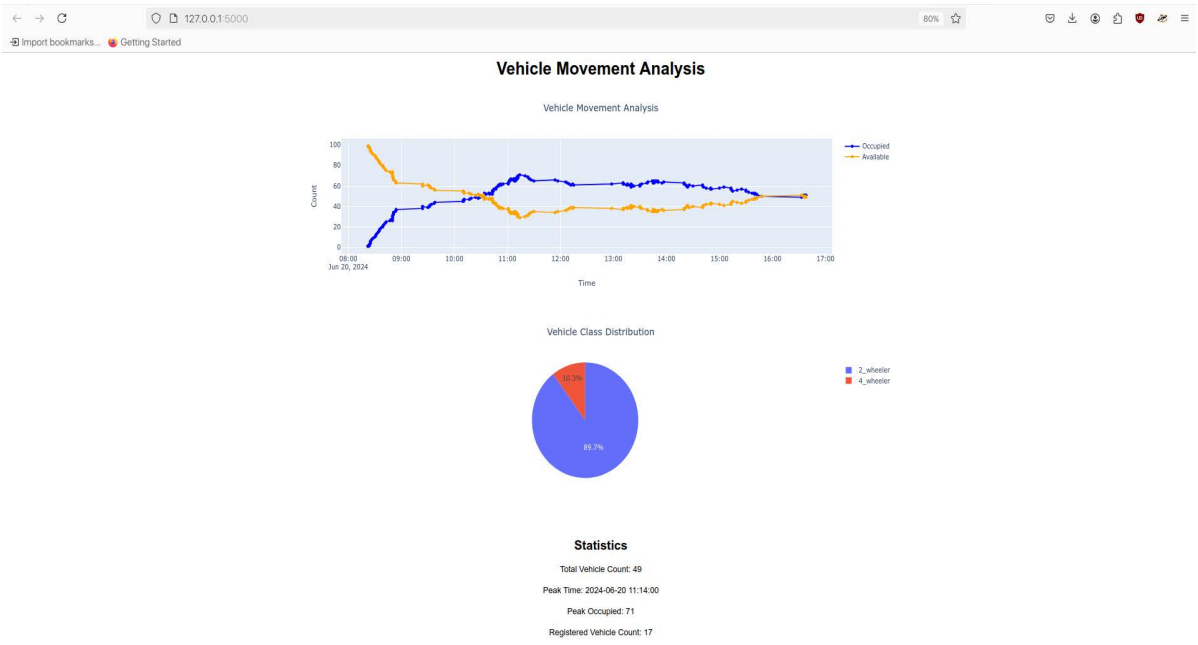


4.6. All the obtained information are stored locally in the “vehicle.ods” file



	A	B	C	D	E	F	G	H
1	Date	Vehicle Class	License Plate	Registered	Status	Time	Occupied	Available
2	20/06/24	2_wheeler	KO4H39165	Yes	Entry	8:22 AM	1	99
3	20/06/24	2_wheeler		No	Entry	8:23 AM	2	98
4	20/06/24	2_wheeler		No	Exit	8:23 AM	1	99
5	20/06/24	2_wheeler	5072	No	Entry	8:24 AM	2	98
6	20/06/24	2_wheeler	KT330J	No	Entry	8:24 AM	3	97
7	20/06/24	2_wheeler	19954	No	Entry	8:25 AM	4	96
8	20/06/24	2_wheeler	MO L09550	No	Entry	8:25 AM	5	95
9	20/06/24	2_wheeler	VJ	No	Entry	8:25 AM	6	94
10	20/06/24	2_wheeler	J02JC 34027	No	Entry	8:26 AM	7	93
11	20/06/24	2_wheeler	4CAO4K1587	No	Entry	8:27 AM	8	92
12	20/06/24	2_wheeler	551	No	Entry	8:28 AM	9	91
13	20/06/24	2_wheeler	O41	No	Entry	8:30 AM	10	90
14	20/06/24	2_wheeler	KA0AKJ	No	Entry	8:31 AM	11	89
15	20/06/24	2_wheeler		No	Entry	8:31 AM	12	88
16	20/06/24	2_wheeler	VT G4 LAD2	No	Entry	8:32 AM	13	87
17	20/06/24	2_wheeler	KA04K G340	No	Entry	8:33 AM	14	86
18	20/06/24	2_wheeler	P5294	No	Entry	8:34 AM	15	85
19	20/06/24	2_wheeler	JV5115	No	Entry	8:35 AM	16	84
20	20/06/24	2_wheeler	P91WBD731	No	Entry	8:35 AM	17	83
21	20/06/24	2_wheeler	KAO4KK917	Yes	Entry	8:36 AM	18	82
22	20/06/24	2_wheeler	3 CKA04HV3	No	Entry	8:37 AM	19	81
23	20/06/24	2_wheeler	4 4 04 IA F4J	No	Entry	8:39 AM	20	80
24	20/06/24	2_wheeler	IKAO4K	No	Entry	8:39 AM	21	79
25	20/06/24	2_wheeler	W1 V378545	No	Entry	8:40 AM	22	78
26	20/06/24	2_wheeler	KA02 4	No	Entry	8:41 AM	23	77
27	20/06/24	2_wheeler	K02JF3095	Yes	Entry	8:42 AM	24	76
28	20/06/24	2_wheeler	KA04K	No	Entry	8:43 AM	25	75
29	20/06/24	2_wheeler	378 67 1	No	Entry	8:47 AM	26	74
30	20/06/24	2_wheeler	KAO2HW222	No	Entry	8:48 AM	27	73
31				No	Exit		26	74
32				No	Entry		27	73
33	20/06/24	2_wheeler		No	Exit	8:50 AM	26	74
34	20/06/24	2_wheeler	P L2225	No	Entry	8:50 AM	27	73
35	20/06/24	2_wheeler	344 22KA203	No	Entry	8:50 AM	28	72
36	20/06/24	2_wheeler	XA92KC8522	No	Entry	8:50 AM	29	71
37	20/06/24	2_wheeler	P5 A9WVND	No	Entry	8:50 AM	30	70
38	20/06/24	2_wheeler	KA52490008	No	Entry	8:50 AM	31	69
39	20/06/24	2_wheeler	KA04KN5172	No	Entry	8:51 AM	32	68
40	20/06/24	2_wheeler	KA04HN0259	Yes	Entry	8:51 AM	33	67
41	20/06/24	2_wheeler		No	Entry	8:51 AM	34	66
42	20/06/24	2_wheeler	K02J249285	No	Entry	8:53 AM	35	65
43	20/06/24	2_wheeler	KA523	No	Entry	8:54 AM	36	64

4.7. Visual and graphical representation of the obtained data



4.8. Patterns

The graph provides a visual representation of the movement of vehicles throughout the day:

- The number of vehicles peak at 11:14 AM with the maximum number of vehicles entered were 71 at that time.
- The live count of the vehicle at the peak time was 49.
- The rate of inflow of vehicles is maximum between 8:30 AM to 9:00 AM.
- Maximum movement of vehicles happened around 1:00 PM to 3:00 PM.
- The number of registered vehicles entered during the particular day is 17.

5. Conclusion

5.1. Summary of the findings

- The model correctly detects the vehicles as 2-wheeler or 4-wheeler along with its license plate using Yolo object detection algorithm.
- Further the image is preprocessed and the license plate bounding boxes are passed to Tesseract, which identifies the characters of the number plate.
- Extraction of date and time (collectively as time-stamp) is also successfully achieved using Tesseract.
- However, the accuracy of the character recognition of license plate is not consistent and are prone to occasional error from the Tesseract even after custom training the license plate font.
- The reason would be, that the input images are photographed under different lighting conditions, different angles.
- The slant nature of the license plate with HSRP (High Security Registration Plate) font reduced the accuracy drastically which is reflected in the final local storage “vehicle.ods” file where few spots are either empty or erroneous data.
- The entry and exit time of the particular vehicle is accurately recorded.
- The vehicle license number is also compared and mapped accurately with the “registered_vehicles.ods” file.
- All of the above data are used to generate insights which are visually displayed in the webpage built using HTML and Flask.

5.2. Future Work

5.2.1. Immediate future work

- Increase the accuracy of the Tesseract in recognizing the characters of the number plate using stationary stand or taking the photo from a fixed particular angle.
- Identify the vehicle as privately owned, commercial or electric using the color of the license plate.
- Identify the state to which the vehicle belongs by reading the first few alphabets from the license plate.
- Calculate the number of minutes or hours the vehicle was parked inside.
- Identify the number of re-entries of the vehicle to calculate the frequency of the vehicle.
- Add support to capture vehicle and license plate from the live video feed.

5.2.2. Long-term future work

- Display the vehicle count and parking space available at the entrance so that the driver can make quick decisions and avoids wastage of time.
- Make the system as Priority based parking system which prioritizes and allows the entry of the registered vehicles compared to guest vehicles.
- During institution’s non-working day, the available parking space can be rented which saves parking space resources and adds extra income to the institute.
- Build a mobile application for the staff (faculty and students) of the institute for the easy monitoring of the parking space.

6. References

[1]. https://www.researchgate.net/publication/357880632_Priority-based_parking_system_for_university_campus_using_IOT_and_CVRP

[2]. https://www.sciencedirect.com/science/article/pii/S2405844021011531

[3]. https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/iet-its.2018.5207

[4]. https://www.mapsofindia.com/my-india/government/parking-problems-in-india-and-their-solutions

[5]. https://countercurrents.org/2021/07/problem-of-parking-in-urban-areas-and-their-possible-solutions/

[6]. https://www.klausindia.com/media/parking-issues-in-indian-metropolises-and-solutions-that-help/

[7]. https://github.com/AlexeyAB/darknet

[8]. https://github.com/tesseract-ocr/tesseract

[9]. https://towardsdatascience.com/yolo-v4-optimal-speed-accuracy-for-object-detection-79896ed47b50?gi=68a63e745afa

[10]. https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/

----------END---OF---REPORT----------