National Institute
of Technology
Calicut - CVLA
Research Group

11th June 2024

PRESENTED BY

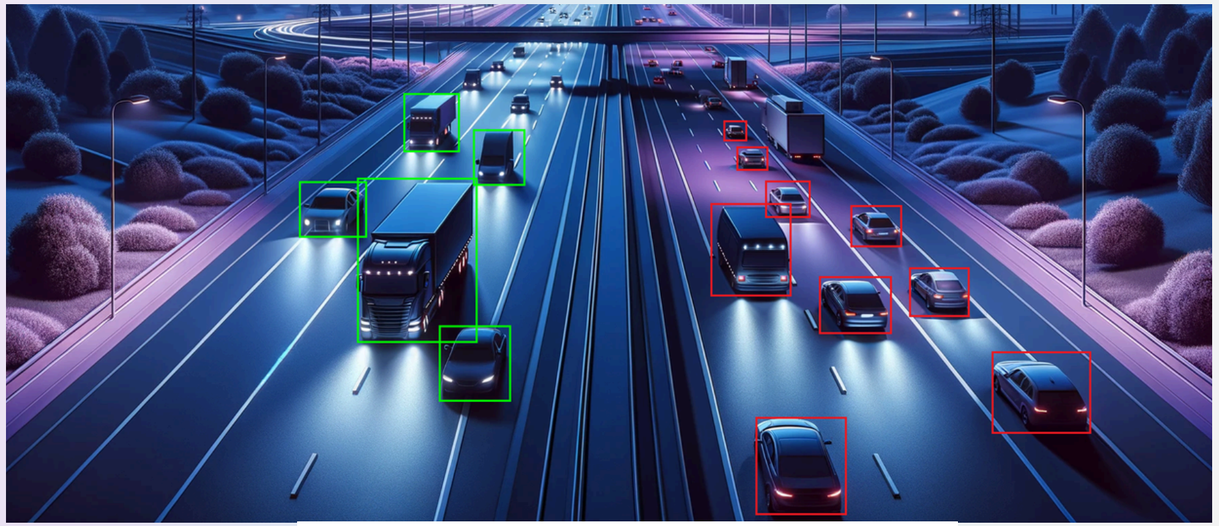
Ayushi K
Rakshitha K
Hamees M

Research Review Presentation

Dynamic Traffic Controller System

Main Objective - Task 2

- Estimate vehicle density and Queue Length of the oncoming lane
- Detect the presence of emergency vehicles (if any)
- Accurately predict the above quantities in real-time using machine learning algorithms.
- Real-time object detection and Vehicle density calculation



Work Done

- Research paper analysis on object detection
- Data Preprocessing
- Custom Dataset creation
- Working on model training for traffic queue length calculation
- Working on model training for Emergency Vehicle detection

Literature Survey on Object Detection Models

Sl. No	Title of the paper	Year of publication	Problem Statement	ML Technique used	Final Outcome
1	Machine learning driven intelligent and self adaptive system for traffic management in smart cities.	2022	Traffic congestion is becoming a serious problem with the large number of vehicles on the roads. In the traditional traffic control system, the timing of the green light is adjusted regardless of the average traffic rate at the junction. In order to handle road traffic issues, an intelligent traffic management solution is required. This article represents a self adaptive real-time traffic light control algorithm based on the traffic flow. We present a machine learning approach coupled with image processing to manage the traffic clearance at the signal junction.	1. OpenCV is used for counting the vehicles from the input image. Here in this work, the OpenCV.cdn module is used to count the number of vehicles. 2. YOLOv3 is a state-of-the-art model for real-time object detection. YOLOv3 predicts 4 coordinates for each bounding box around an object. Training is performed with a sum of squared error loss.	1. The YOLO object detection model was trained on 42 authentic traffic images of two main junctions of the city Jabalpur. The mean time taken for detecting the vehicles per image is 1.36s. The time taken in object detection mainly depends on the incorporated processing hardware. 2. For correct predictions, it is necessary to take an image that shows vehicles as discreet as possible, not overlapped by any other object. 3. It can be inferred that the proposed intelligent traffic management system based on the single image processing is self adaptive, highly accurate, fast and has the potential to be implemented in the traffic clearance at the junctions.

2	Dynamic Traffic Control System with Reinforcement Learning Technique	2020	Traditional reinforcement learning is hard to apply because of two key difficulties: (1)how to represent condition; and (2)how to model the correlation among condition and choice. To address these two difficulties, investigations have applied deep reinforcement learning techniques, for example, Deep Q-learning (DQN), for traffic light control problems. Late deep reinforcement learning approaches gained promising ground for the traffic light control problem. Our methodology expands this profession by making important element extraction algorithms for giving better performance of classification.	1. Phase-gated model learning: The operator will take the state,which is the representation of condition, as model info. The earth typically incorporates the current traffic light phase and traffic conditions. 2. A highlight extraction algorithm to show signs of improvement performance. 3. Object Recognition utilising Speeded-Up Robust Features (SURF) is made out of three stages - feature extraction, feature depiction, and feature coordinating.	1. Peak hour vs Non-peak hour: On the given day, there is more traffic on WE bearing than SN for more often than not, during which a perfect traffic light control strategy is relied upon to provide longer time for WE guidance. And during top hours (around 7:00, 9:30 and 18:00), the policies gained from our technique give longer time for green light on WE than non-top hours. In the early morning, the vehicle appearance rates on SN are larger than the rates on WE, and our strategy consequently gives longer time to SN. 2.Weekday versus Weekend: The policy gives fewer green lights on WE (more green light on SN) during weekend daytime than it gives on weekdays. This is on the grounds that there is more traffic on SN than on WE during weekend daytime.

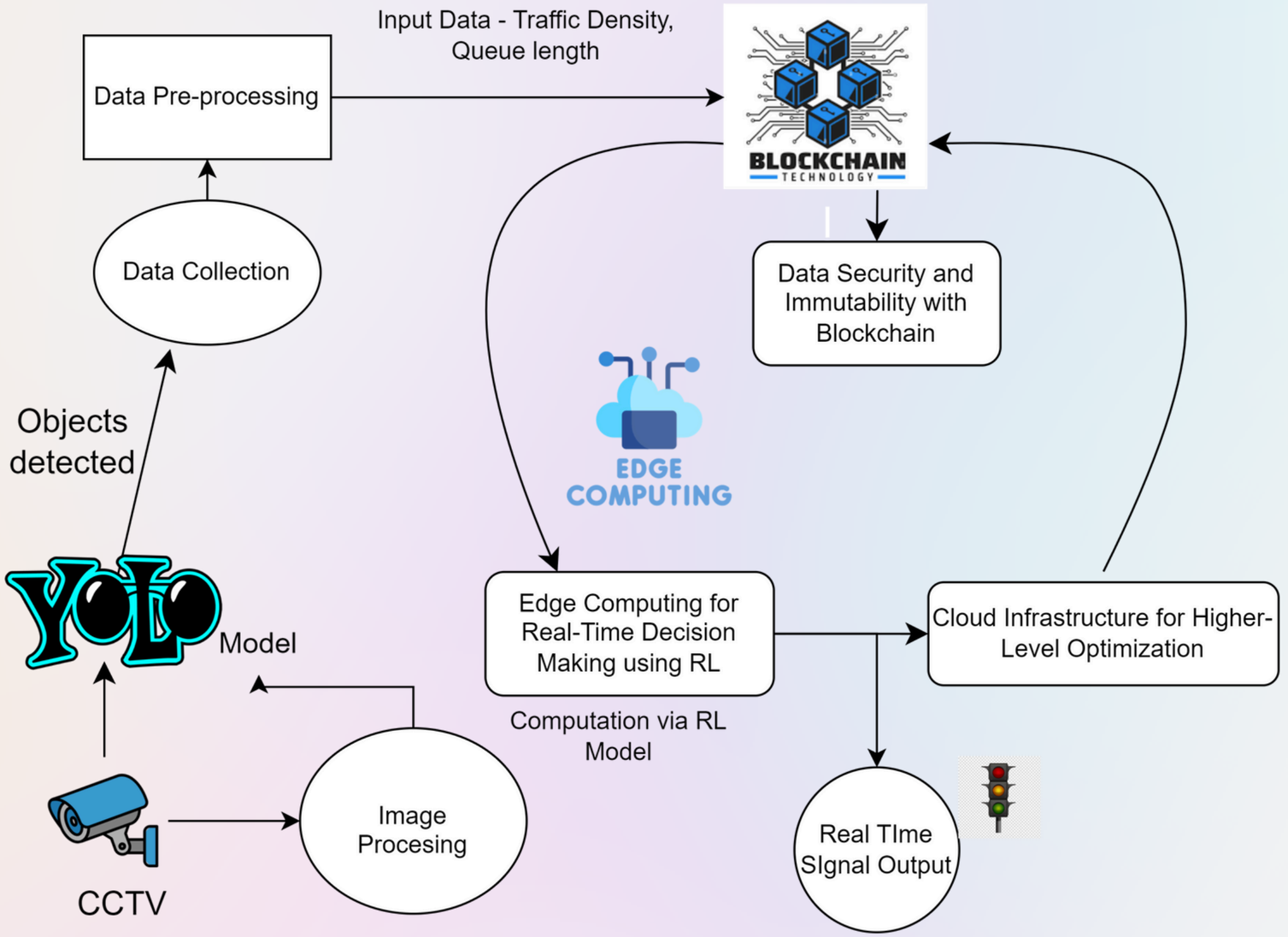
3	Vehicle detection and counting of a vehicle using OpenCV	2021	The trouble of getting the initial background, there is the mistake of continuous background update and the trouble of controlling the update speed in moving vehicle location of traffic video. With the expanding number of streets and traffic everywhere, traffic observing and control utilizing current advancements has become a necessity. Vehicle detection is the key task in this area and counting of vehicles plays an important role.	1. Adaptive background subtraction, binarization, and morphological activities are used to detect a moving vehicle. 2. Blob tracking to coordinate with vehicles in the current frame and those in the past outline.	1. The exactness of the proposed vehicle counting technique changed from 95-99%, based on the video input. 2. Once the vehicle passes into the virtual detection zone, it will be detected and counted as per sequential order.
4	Smart Traffic Control System using YOLO	2019	Traffic congestion is becoming a serious problem with a large number of cars on the roads. Vehicle queue length waiting to be processed at the intersection is rising sharply with the increase of the traffic flow, and the traditional traffic lights cannot efficiently schedule it. A real time traffic light control algorithm based on the traffic flow is proposed in this paper. In fact, we use computer vision and machine learning to have the characteristics of the competing traffic flows at the signalised road intersection	1. YOLO is a clever convolutional neural network (CNN) for doing object detection in real-time. 2. A deep neural network (DNN) is an artificial neural network (ANN) with multiple layers between the input and output layers. The DNN finds the correct mathematical manipulation to turn the input into the output. 3. A convolutional neural network consists of an input and an output layer, as well as multiple hidden layers.	The new system facilitates the movement of cars in intersections, resulting in reducing congestion, less CO2 emissions, etc. The richness that video data provides highlights the importance of advancing the state-of-the-art in object detection, classification and tracking for real-time applications.

5	**Adaptive Traffic Light Based on Yolo-Darknet Object Detection**	**2019**	**Nowadays roads and streets are getting overcrowded, especially in bigger cities. Hence the main goal of our project is to build a traffic monitoring system that is able to detect the movement of cars and to track and count the different vehicles by analysing a camera picture with the help of computer vision. With a vision based adaptive traffic light system, the device that is needed is only a camera and computer. This means this system will cost less because most intersection or traffic lights already have some CCTV, we just need to calibrate the camera and install the proprietary software for a self-regulating traffic light.**	**1.The detection is done by the Darknet object detection framework. Darknet is one of open source object detection framework that has YOLOv3.** **2. The ROI masking was done through OpenCV.**	**1. We could run detection at 2,4 fps average when using YOLOV3 dataset, and around 15.5 fps average when using Tiny version of YOLO detector. This result shows that the YOLO detector is lightweight enough and very possible to run better on a better high-performance system.** **2. The ROI successfully managed the system to only detect the part of the road that we want to control. OpenCV helps the system only detect a lane that we want to control from two or more lane roads. Therefore, it also helps to reduce system resources because detection only occurs in certain sections, not in the whole scene**

6	A Sound-based Machine Learning to Predict Traffic Vehicle Density	2021	Traffic flow mismanagement is a significant challenge in all countries, especially in crowded cities. An alternative solution is to utilise smart technologies to predict traffic flow. In this study, frequency spectrum describing traffic sound characteristics is used as an indicator to predict the next five-minute vehicle density. Sound frequency and vehicle intensity are collected during a thirteen-hour data gathering. The collected sound intensity and frequency are then used to learn three machine-learning models and random forest and to predict vehicle intensity.	Three machine learning algorithms were trained and tested upon selection of the best MLA to predict short-term traffic flow are Support Vector Machine (SVM), Artificial Neural Network (ANN), and Random Forest (RF). The performance of the three machine-learning algorithms in prediction was evaluated using RMSE.	1. RF yielded the least RMSE value of 10.67, by SVM with an RMSE value. Lastly, ANN yielded the highest RMSE value. Consequently, the machine learning algorithm that yielded the least RMSE for sound frequency is Random Forest. 2. In the cross-validation process, Random Forest has a consistent performance where in it yielded the lowest RMSE among the three algorithms, which is the same as the result in the first learning phase. Hence, it can be validated that Random Forest (RF) is the best machine-learning algorithm to predict short-term traffic flow.

7	Traffic Density Estimation using Machine Learning Methods	2021	The Intelligent Transportation System (AUS) is expressed as a system that provides users with better information and safer, more coordinated, and smarter use of transportation networks with different transportation modes and traffic management. One of the most important components of AUS models is the determination of traffic density which is a difficult problem as it affects other interconnected intersections and varies in time. In this study, the long term short memory network (LSTM) model, one of the deep learning methods, is proposed to estimate the traffic density of a certain region using open data of Istanbul Metropolitan Municipality.	1. Linear regression method: A multiple linear regression model was used because the number of independent variables in the traffic dataset was more than one. 2. Decision tree: Regression trees were used because it works with constantly changing data. 3. Random forest: the algorithm is an ensemble learning algorithm consisting of the output of multiple decision trees. Each node branches by choosing the best among the randomly selected variables in the node. 4. Deep learning: An artificial neural network is formed by the combination of neural nodes. 5. Long short-term memory (LSTM): The LSTM deep learning algorithm is known as a recurrent neural network introduced to eliminate the disadvantages of the RNN architecture.	1. Experimental evaluations of linear regression, decision trees, random forest, classical deep learning, and LSTM methods were made using real data from IMM. RMSE and MAE values were used as evaluation criteria to examine the performance of the methods. 2. The LSTM method made predictions with lower error rates compared to other methods. Standard deviation value of the LSTM method gave better results compared to other methods. This shows that the LSTM method works more stable. 3. The graph with the highest overlap between the predicted value and the actual value of number of vehicles estimated by decision trees, deep learning, linear regression, LSTM and random forest method - belongs to the study using the LSTM method.

Proposed Methodolgy

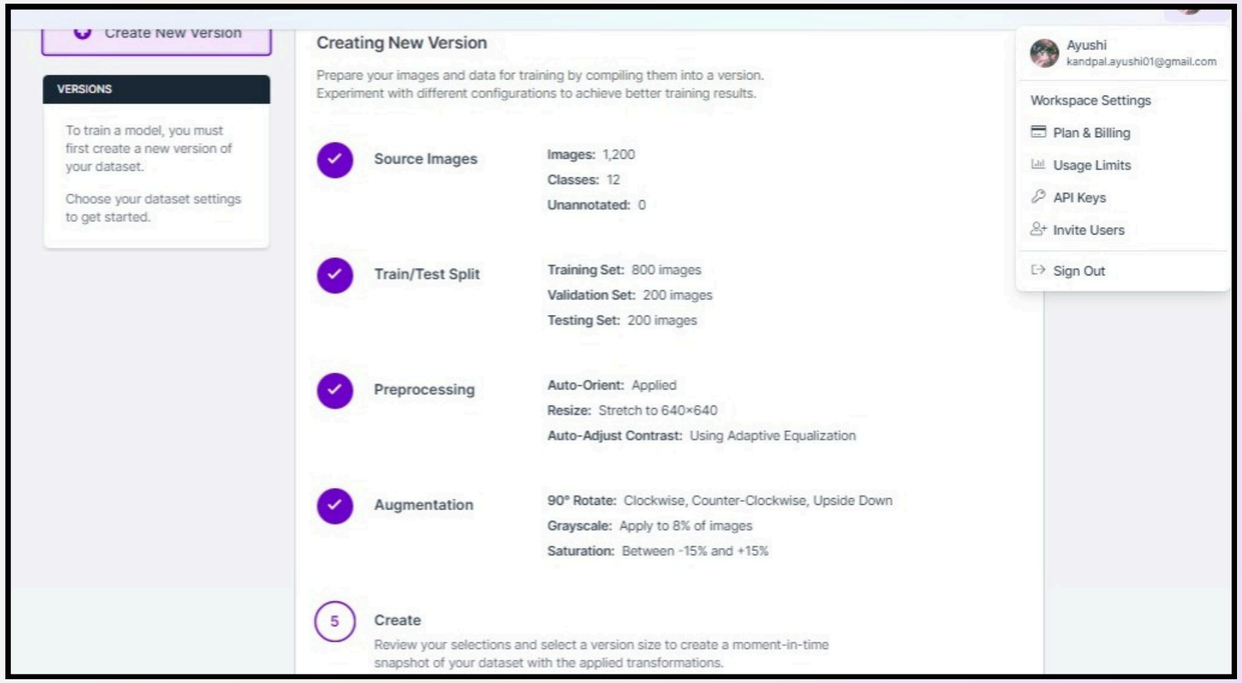


Work on model training for traffic queue length calculation

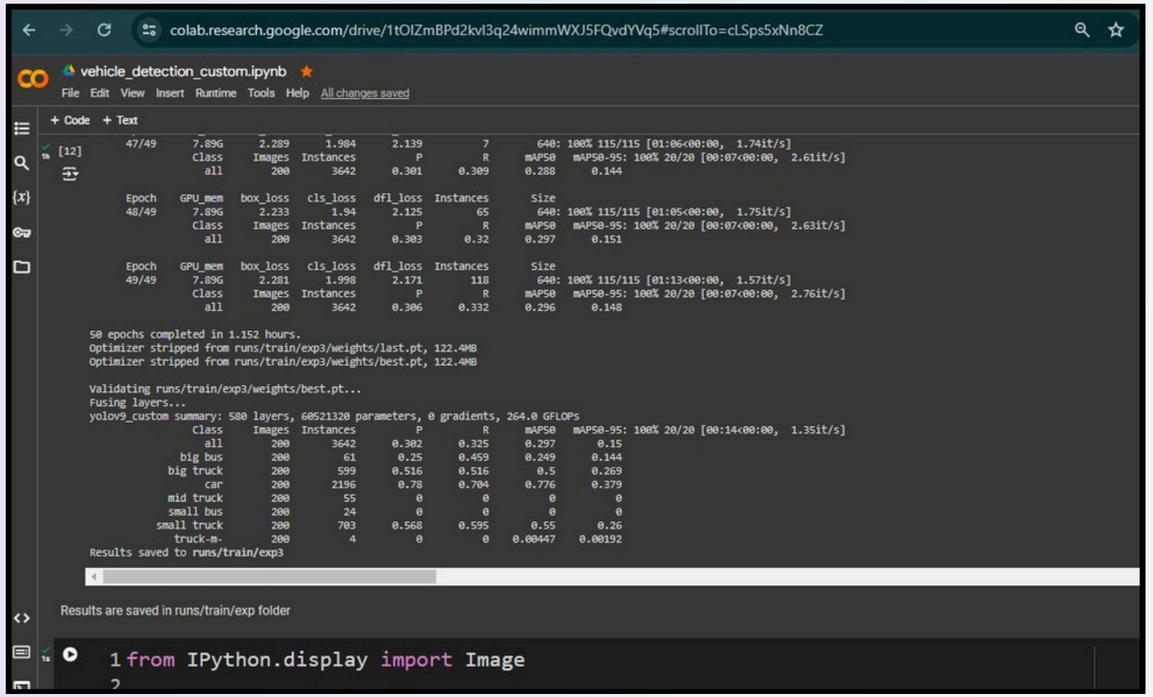
Steps performed

1. Dataset Preparation
2. Data Preprocessing
3. Vehicle detection model preparation using yolov9

Work on model training for vehicle detection

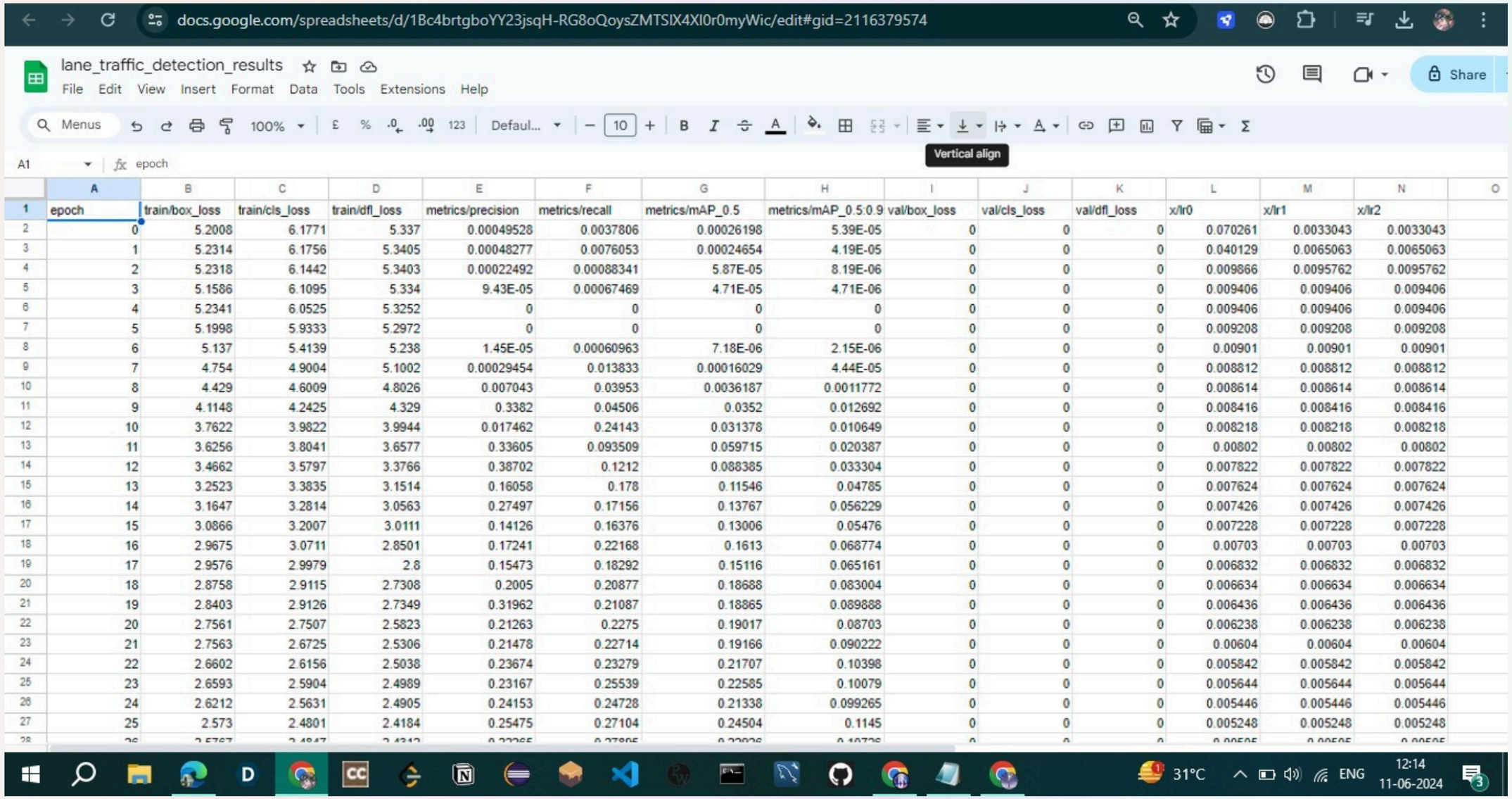


Dataset preparation and preprocessing

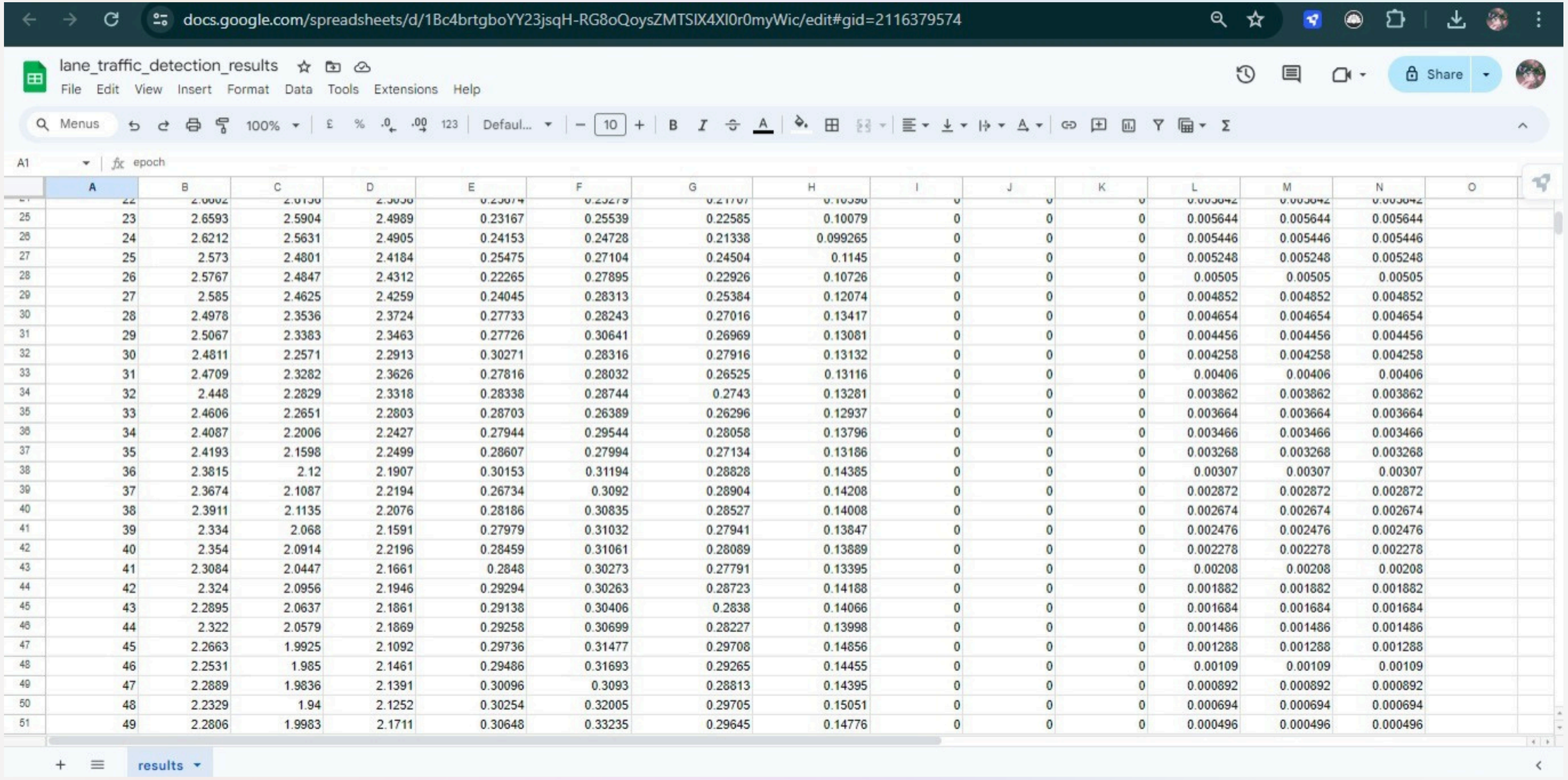


Model Training

Epochs Result

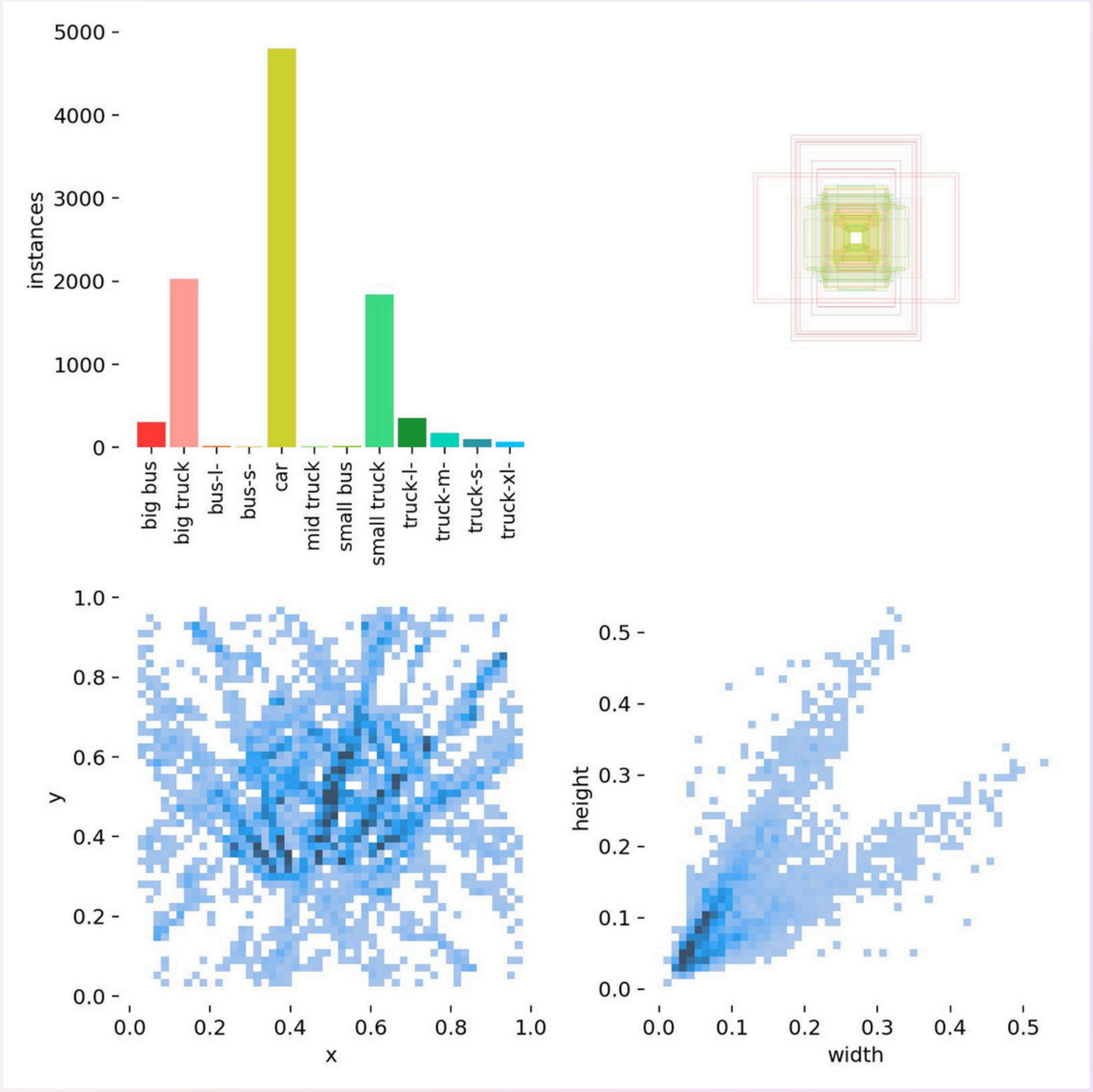


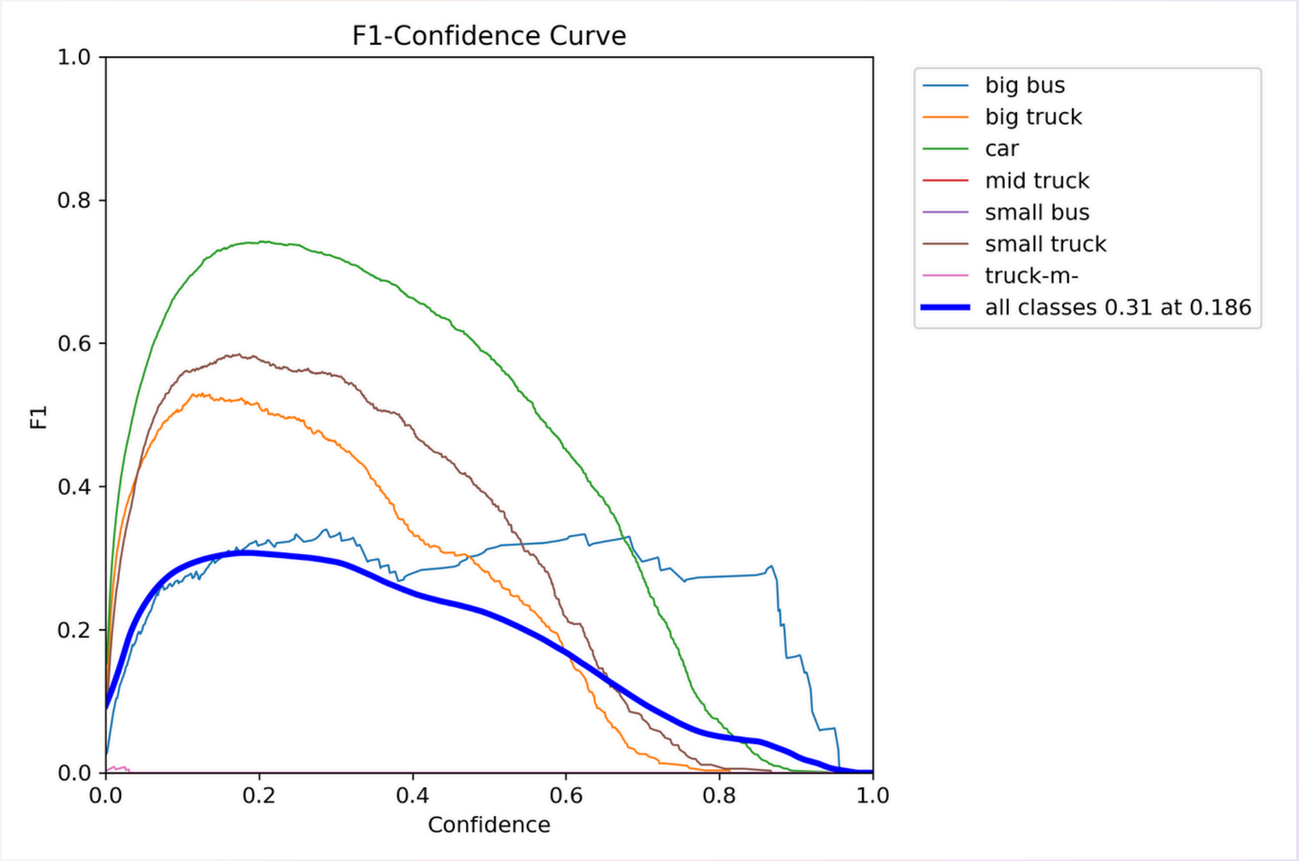
epoch	train/box_loss	train/cls_loss	train/dfl_loss	metrics/precision	metrics/recall	metrics/mAP_0.5	metrics/mAP_0.5:0.9	val/box_loss	val/cls_loss	val/dfl_loss	x/lr0	x/lr1	x/lr2
0	5.2008	6.1771	5.337	0.00049528	0.0037806	0.00026198	5.39E-05	0	0	0	0.070261	0.0033043	0.0033043
1	5.2314	6.1756	5.3405	0.00048277	0.0076053	0.00024654	4.19E-05	0	0	0	0.040129	0.0065063	0.0065063
2	5.2318	6.1442	5.3403	0.00022492	0.00088341	5.87E-05	8.19E-06	0	0	0	0.009866	0.0095762	0.0095762
3	5.1586	6.1095	5.334	9.43E-05	0.00067469	4.71E-05	4.71E-06	0	0	0	0.009406	0.009406	0.009406
4	5.2341	6.0525	5.3252	0	0	0	0	0	0	0	0.009406	0.009406	0.009406
5	5.1998	5.9333	5.2972	0	0	0	0	0	0	0	0.009208	0.009208	0.009208
6	5.137	5.4139	5.238	1.45E-05	0.00060963	7.18E-06	2.15E-06	0	0	0	0.00901	0.00901	0.00901
7	4.754	4.9004	5.1002	0.00029454	0.013833	0.00016029	4.44E-05	0	0	0	0.008812	0.008812	0.008812
8	4.429	4.6009	4.8026	0.007043	0.03953	0.0036187	0.0011772	0	0	0	0.008614	0.008614	0.008614
9	4.1148	4.2425	4.329	0.3382	0.04506	0.0352	0.012692	0	0	0	0.008416	0.008416	0.008416
10	3.7622	3.9822	3.9944	0.017462	0.24143	0.031378	0.010649	0	0	0	0.008218	0.008218	0.008218
11	3.6256	3.8041	3.6577	0.33605	0.093509	0.059715	0.020387	0	0	0	0.00802	0.00802	0.00802
12	3.4662	3.5797	3.3766	0.38702	0.1212	0.088385	0.033304	0	0	0	0.007822	0.007822	0.007822
13	3.2523	3.3835	3.1514	0.16058	0.178	0.11546	0.04785	0	0	0	0.007624	0.007624	0.007624
14	3.1647	3.2814	3.0563	0.27497	0.17156	0.13767	0.056229	0	0	0	0.007426	0.007426	0.007426
15	3.0866	3.2007	3.0111	0.14126	0.16376	0.13006	0.05476	0	0	0	0.007228	0.007228	0.007228
16	2.9675	3.0711	2.8501	0.17241	0.22168	0.1613	0.068774	0	0	0	0.00703	0.00703	0.00703
17	2.9576	2.9979	2.8	0.15473	0.18292	0.15116	0.065161	0	0	0	0.006832	0.006832	0.006832
18	2.8758	2.9115	2.7308	0.2005	0.20877	0.18688	0.083004	0	0	0	0.006634	0.006634	0.006634
19	2.8403	2.9126	2.7349	0.31962	0.21087	0.18865	0.089888	0	0	0	0.006436	0.006436	0.006436
20	2.7561	2.7507	2.5823	0.21263	0.2275	0.19017	0.08703	0	0	0	0.006238	0.006238	0.006238
21	2.7563	2.6725	2.5306	0.21478	0.22714	0.19166	0.090222	0	0	0	0.00604	0.00604	0.00604
22	2.6602	2.6156	2.5038	0.23674	0.23279	0.21707	0.10398	0	0	0	0.005842	0.005842	0.005842
23	2.6593	2.5904	2.4989	0.23167	0.25539	0.22585	0.10079	0	0	0	0.005644	0.005644	0.005644
24	2.6212	2.5631	2.4905	0.24153	0.24728	0.21338	0.099265	0	0	0	0.005446	0.005446	0.005446
25	2.573	2.4801	2.4184	0.25475	0.27104	0.24504	0.1145	0	0	0	0.005248	0.005248	0.005248

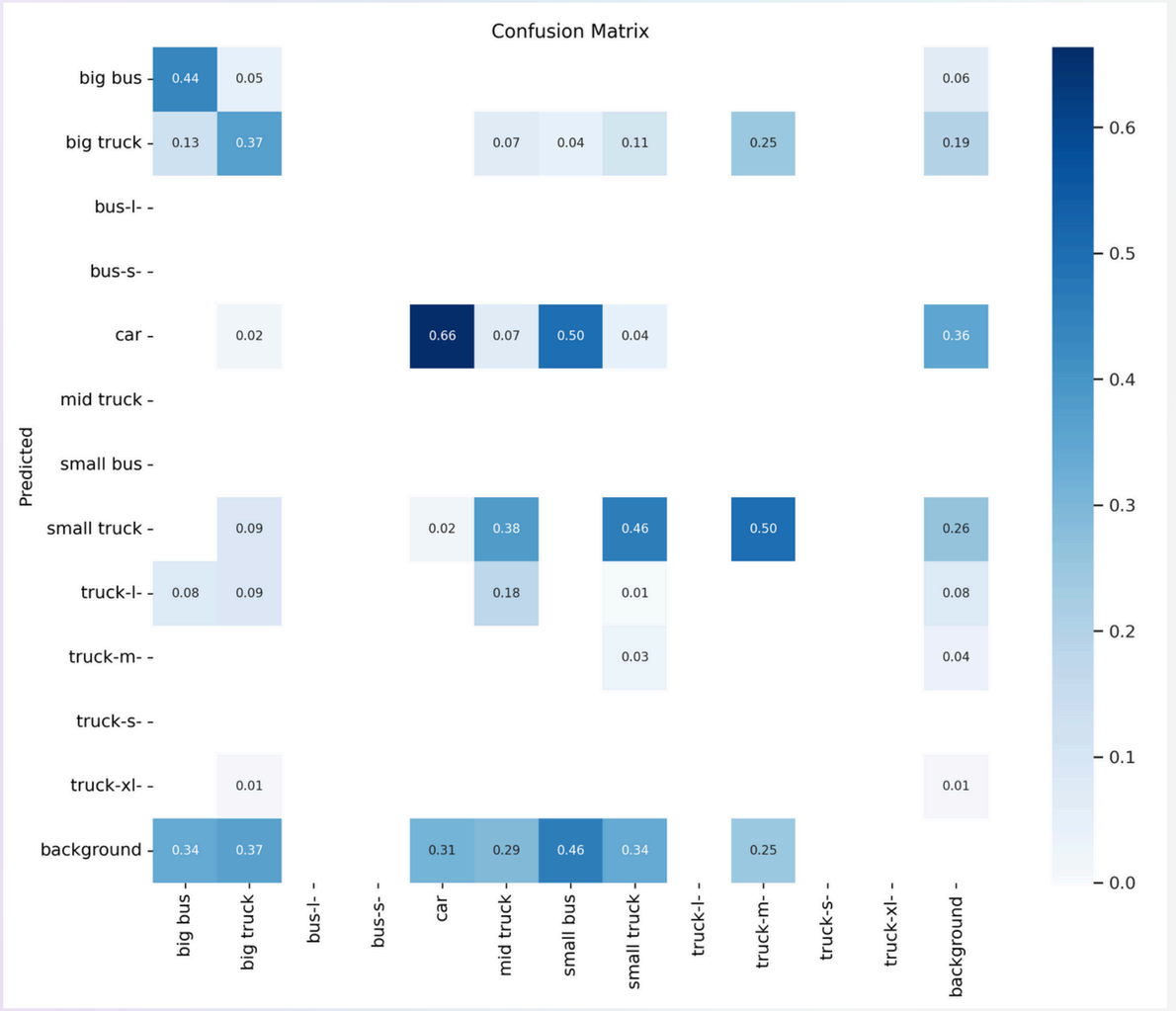


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
25	23	2.6593	2.5904	2.4989	0.23167	0.25539	0.22585	0.10079	0	0	0	0.005644	0.005644	0.005644	
26	24	2.6212	2.5631	2.4905	0.24153	0.24728	0.21338	0.099265	0	0	0	0.005446	0.005446	0.005446	
27	25	2.573	2.4801	2.4184	0.25475	0.27104	0.24504	0.1145	0	0	0	0.005248	0.005248	0.005248	
28	26	2.5767	2.4847	2.4312	0.22265	0.27895	0.22926	0.10726	0	0	0	0.00505	0.00505	0.00505	
29	27	2.585	2.4625	2.4259	0.24045	0.28313	0.25384	0.12074	0	0	0	0.004852	0.004852	0.004852	
30	28	2.4978	2.3536	2.3724	0.27733	0.28243	0.27016	0.13417	0	0	0	0.004654	0.004654	0.004654	
31	29	2.5067	2.3383	2.3463	0.27726	0.30641	0.26969	0.13081	0	0	0	0.004456	0.004456	0.004456	
32	30	2.4811	2.2571	2.2913	0.30271	0.28316	0.27916	0.13132	0	0	0	0.004258	0.004258	0.004258	
33	31	2.4709	2.3282	2.3626	0.27816	0.28032	0.26525	0.13116	0	0	0	0.00406	0.00406	0.00406	
34	32	2.448	2.2829	2.3318	0.28338	0.28744	0.2743	0.13281	0	0	0	0.003862	0.003862	0.003862	
35	33	2.4606	2.2651	2.2803	0.28703	0.26389	0.26296	0.12937	0	0	0	0.003664	0.003664	0.003664	
36	34	2.4087	2.2006	2.2427	0.27944	0.29544	0.28058	0.13796	0	0	0	0.003466	0.003466	0.003466	
37	35	2.4193	2.1598	2.2499	0.28607	0.27994	0.27134	0.13186	0	0	0	0.003268	0.003268	0.003268	
38	36	2.3815	2.12	2.1907	0.30153	0.31194	0.28828	0.14385	0	0	0	0.00307	0.00307	0.00307	
39	37	2.3674	2.1087	2.2194	0.26734	0.3092	0.28904	0.14208	0	0	0	0.002872	0.002872	0.002872	
40	38	2.3911	2.1135	2.2076	0.28186	0.30835	0.28527	0.14008	0	0	0	0.002674	0.002674	0.002674	
41	39	2.334	2.068	2.1591	0.27979	0.31032	0.27941	0.13847	0	0	0	0.002476	0.002476	0.002476	
42	40	2.354	2.0914	2.2196	0.28459	0.31061	0.28089	0.13889	0	0	0	0.002278	0.002278	0.002278	
43	41	2.3084	2.0447	2.1661	0.2848	0.30273	0.27791	0.13395	0	0	0	0.00208	0.00208	0.00208	
44	42	2.324	2.0956	2.1946	0.29294	0.30263	0.28723	0.14188	0	0	0	0.001882	0.001882	0.001882	
45	43	2.2895	2.0637	2.1861	0.29138	0.30406	0.2838	0.14066	0	0	0	0.001684	0.001684	0.001684	
46	44	2.322	2.0579	2.1869	0.29258	0.30699	0.28227	0.13998	0	0	0	0.001486	0.001486	0.001486	
47	45	2.2663	1.9925	2.1092	0.29736	0.31477	0.29708	0.14856	0	0	0	0.001288	0.001288	0.001288	
48	46	2.2531	1.985	2.1461	0.29486	0.31693	0.29265	0.14455	0	0	0	0.00109	0.00109	0.00109	
49	47	2.2889	1.9836	2.1391	0.30096	0.3093	0.28813	0.14395	0	0	0	0.000892	0.000892	0.000892	
50	48	2.2329	1.94	2.1252	0.30254	0.32005	0.29705	0.15051	0	0	0	0.000694	0.000694	0.000694	
51	49	2.2806	1.9983	2.1711	0.30648	0.33235	0.29645	0.14776	0	0	0	0.000496	0.000496	0.000496	

Training Result




F1-Confidence Curve
F1
Confidence
0.0
0.2
0.4
0.6
0.8
1.0
big bus
big truck
car
mid truck
small bus
small truck
truck-m-
all classes 0.31 at 0.186


Confusion Matrix
Predicted
big bus
big truck
bus-l-
bus-s-
car
mid truck
small bus
small truck
truck-l-
truck-m-
truck-s-
truck-xl-
background
0.44 0.05 0.06
0.13 0.37 0.07 0.04 0.11 0.25 0.19
0.02 0.66 0.07 0.50 0.04 0.36
0.09 0.02 0.38 0.46 0.50 0.26
0.08 0.09 0.18 0.01 0.08
0.03 0.04
0.01 0.01
0.34 0.37 0.31 0.29 0.46 0.34 0.25
0.6
0.5
0.4
0.3
0.2
0.1
0.0

Queue Lenght Calculation Plan

1. **Detect Vehicles in Each Image**: Use the trained YOLOv9 model to detect vehicles in each image. This will give the bounding boxes for each detected vehicle.
2. **Extract Centroids of Bounding Boxes**: For each detected vehicle, calculate the centroid of its bounding box.
3. **Sort Vehicles by Position**: Sort the detected vehicles based on their coordinates. This helps in determining the order of vehicles in the queue.
4. **Define Thresholds for Queue Detection:**
    - Vertical distance threshold: To determine if vehicles are in the same queue vertically.
    - Horizontal alignment threshold: To ensure vehicles are aligned horizontally within the same lane.

Next Steps in the project :

1. Work on reducing the no. of classes in the dataset
2. Modify the annotations and dataset labellings for improving the accuracy
3. Prepare Emergency Vehicle Detection Model
4. Queue Length and Traffic Density calculation

THANK YOU