Task 2 - Vehicle Detection Report

• Introduction

Traffic signal control systems play a vital role in managing vehicular traffic at intersections, aiming to enhance road safety and efficiency. These systems use a combination of traffic lights, sensors, and timing algorithms to regulate the flow of vehicles and pedestrians. However, in many countries, including ours, traffic signal control systems face several challenges such as congestion during peak hours, lack of synchronization between signals, and inadequate detection of vehicles, especially at night or in adverse weather conditions. These issues often lead to increased travel times, fuel consumption, and pollution, highlighting the need for more advanced and adaptive traffic management solutions.

To address these challenges, several technical solutions have been devised. One approach involves implementing adaptive traffic signal control systems that adjust signal timings based on real-time traffic conditions. These systems use various sensors, such as inductive loops, cameras, and radar, to gather data on vehicle presence and flow. Another solution is the use of synchronized signal timing, or "green wave," which ensures a smoother flow of traffic along main corridors by coordinating the signals to minimize stops. Additionally, some cities have implemented intelligent transportation systems (ITS) that integrate traffic management with other modes of transportation, providing a holistic approach to traffic control. The major steps in these techniques typically include data collection, analysis, and real-time adjustments to signal timings.

Vehicle detection systems are crucial in modern traffic signal control, providing the data necessary for adaptive and intelligent traffic management. These systems utilize machine learning algorithms and computer vision techniques to accurately detect and classify vehicles in real-time. For instance, convolutional neural networks (CNNs) can process images from traffic cameras to identify different types of vehicles and assess traffic density. This information can then be fed into adaptive signal control algorithms to dynamically adjust signal timings, reducing congestion and improving traffic flow. The process is a proposed solution for the given task in the below block diagram. It takes images captured by the lane CCTV as the input and produces the necessary quantities as the required output.

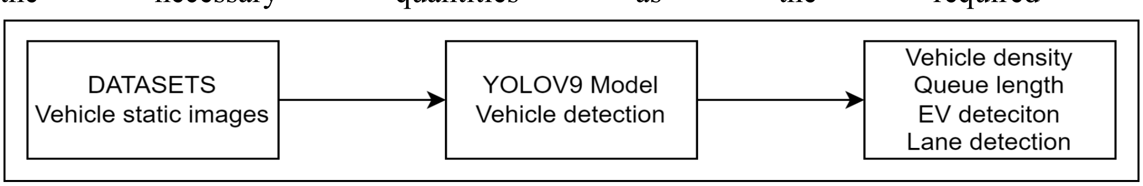


Fig. 2.1: Block diagram for Vehicle Detection using YoloV9.

- **Literature Survey**

Sl. No	Title of the paper	Year of publication	Problem Statement	ML Technique used	Final Outcome
1	Vehicle detection and counting of a vehicle using openCV	2021	The trouble of getting the initial background 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 observation 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.
2	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	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	1. The YOLO object detection model was trained on 42 authentic traffic images of two main junctions of the city of Jabalpur. The mean time taken for detecting the vehicles per image is 1.36s. The time taken in object detection mainly depends on the

			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.	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.	incorporated processing hardware. 2. For correct predictions, it is necessary to take an image that shows vehicles as discreetly 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 traffic clearance at the junctions.
3	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	1. Phase-gated model learning: The operator will take the state, which is the representation of the 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 in	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 a longer time for WE guidance. And during top hours (around 7:00, 9:30 and 18:00), the policies gained from our technique give a longer time for the green light

			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.	performance. 3. Object Recognition utilizing Speeded-Up Robust Features (SURF) is made out of three stages - feature extraction, feature depiction, and feature coordinating.	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 a longer time to SN. 2. **Weekday versus Weekend:** The policy gives fewer green lights on WE (more green lights 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.
4	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 analyzing a camera picture with the help of computer vision. With a vision-based adaptive traffic light system, the device that	1. The detection is done by the Darknet object detection framework. Darknet is one of the open-source object detection frameworks that have YOLOv3. 2. The ROI masking was done through OpenCV.	1. We could run detection at a 2,4 fps average when using the YOLOV3 dataset, and around 15.5 fps average when using the Tiny version of the 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

			is needed is only a camera and a 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.		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
5	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, the 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	Three machine learning algorithms were trained and tested upon selection of the best MLA to predict short-term traffic flow 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 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

			models and random forests and to predict vehicle intensity.		algorithm to predict short-term traffic flow.
6	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 they work 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):**	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. The 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 a number of vehicles estimated by decision trees, deep learning, linear regression, LSTM and random forest method - belongs to the

				The LSTM deep learning algorithm is known as a recurrent neural network introduced to eliminate the disadvantages of the RNN architecture.	study using the LSTM method.
7	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 signalized 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 reduced congestion, fewer 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.

- ## **Proposed Methodology/Solution**

The proposed vehicle detection system employs the YOLOv8.2 model from Ultralytics for robust and efficient vehicle detection. YOLOv8.2's advanced architecture ensures high accuracy and real-time processing, crucial for traffic monitoring applications. The system processes video feeds to detect vehicles, calculate vehicle density, and measure queue length in various lanes. Additionally, the model is trained to identify emergency vehicles, enabling the system to prioritize and manage their presence in traffic. This comprehensive approach provides valuable insights for traffic management, enhancing road safety and optimizing traffic flow by dynamically adjusting signals and lanes based on real-time vehicle data and emergency vehicle detection.

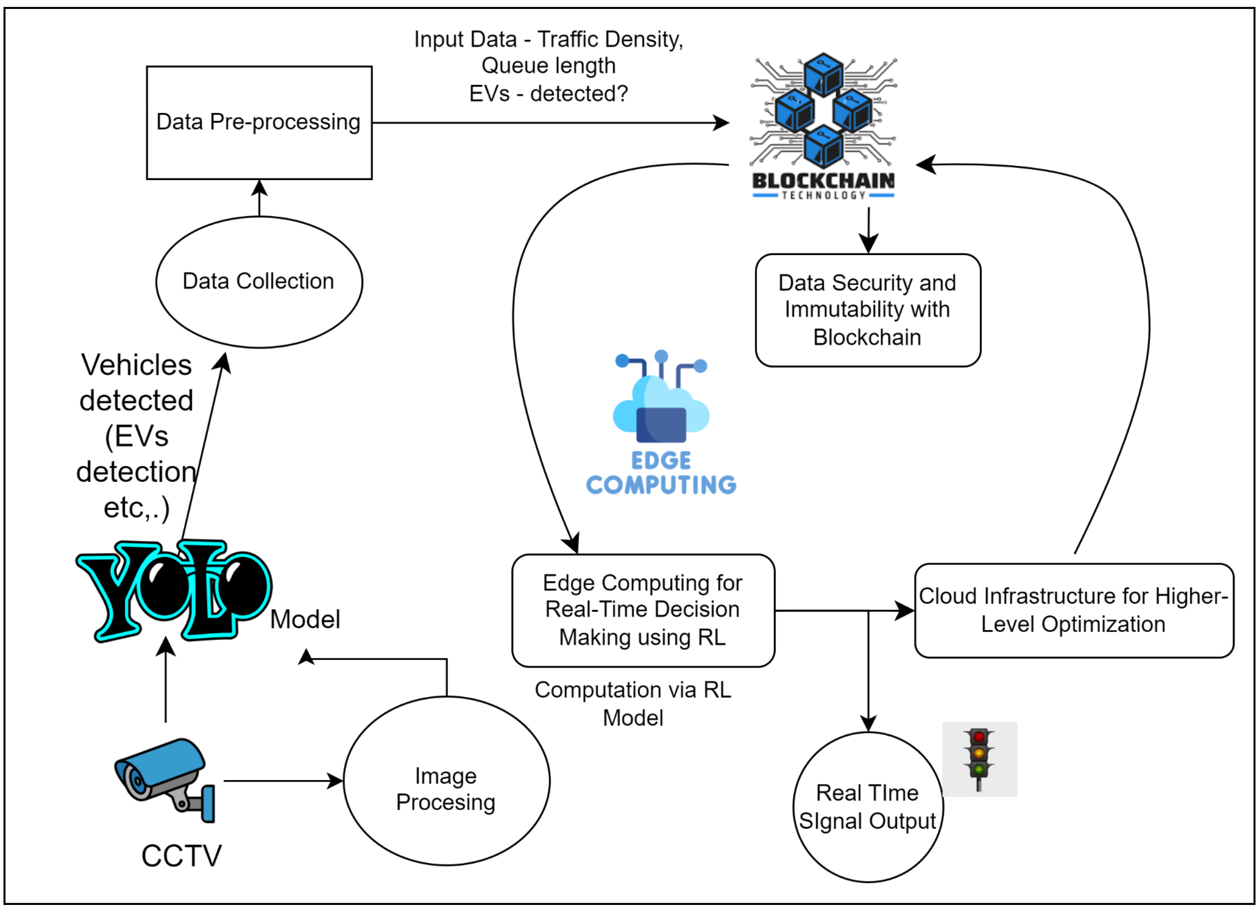


Fig. 2.2: Block diagram for the dynamic traffic signal controller using machine learning techniques and computer vision.

In this project, we develop a dynamic traffic signal controller using advanced image processing and machine learning techniques. RoboFlow is employed for image preprocessing, annotation, and augmentation, ensuring high-quality input data. We utilise the YOLOv9 model for real-time analysis of vehicle density, queue length calculation, and emergency vehicle detection. This approach enables adaptive signal timing, improving traffic flow and reducing congestion. By accurately identifying and responding to varying traffic conditions, the system enhances overall traffic management and safety, particularly for emergency vehicles needing prioritised passage. The detailed process is represented in the form of a block diagram below.

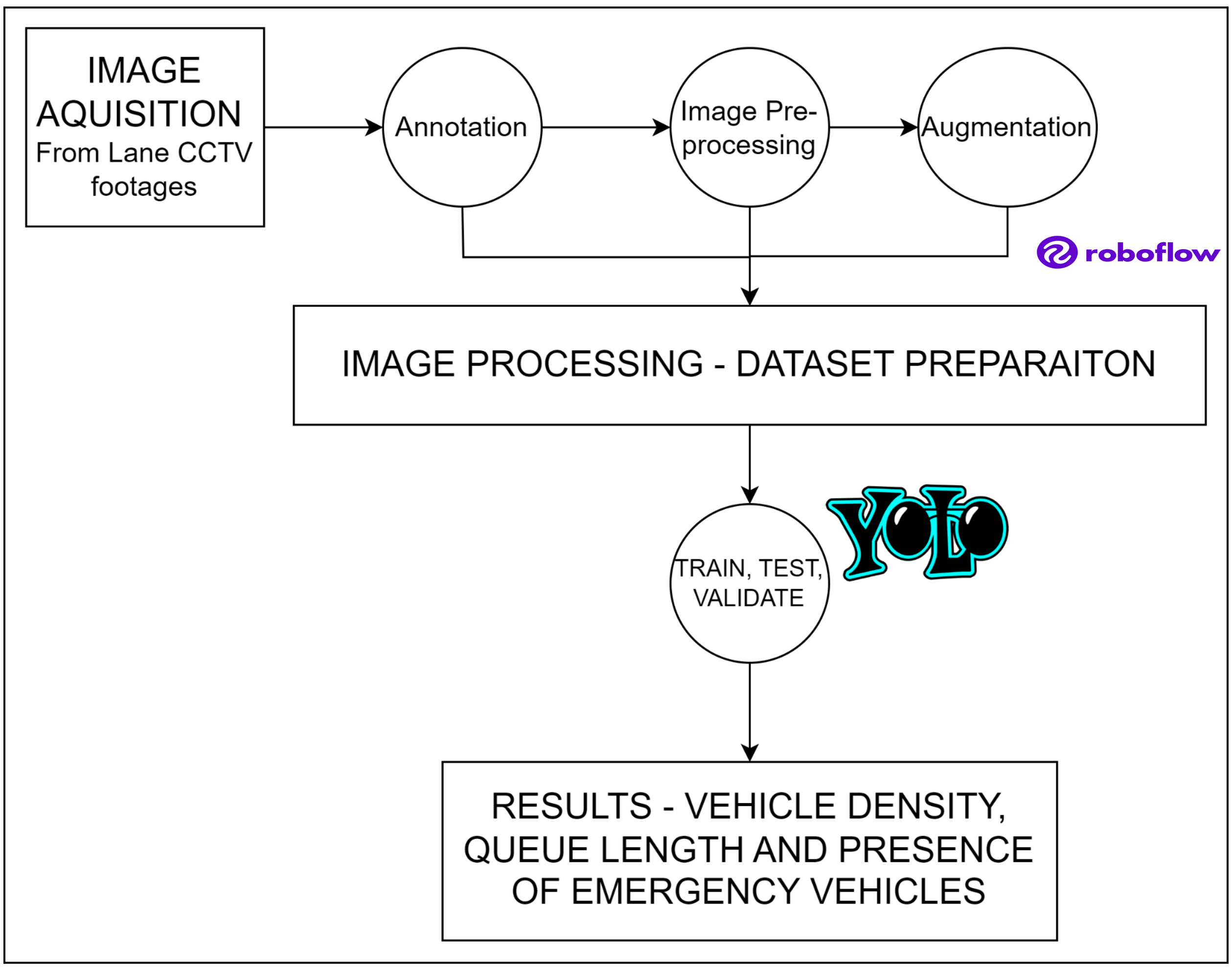


Fig. 2.3: Block diagram for vehicle detection using YOLOV9 and image processing using Roboflow.

- ## Novelty of the solution

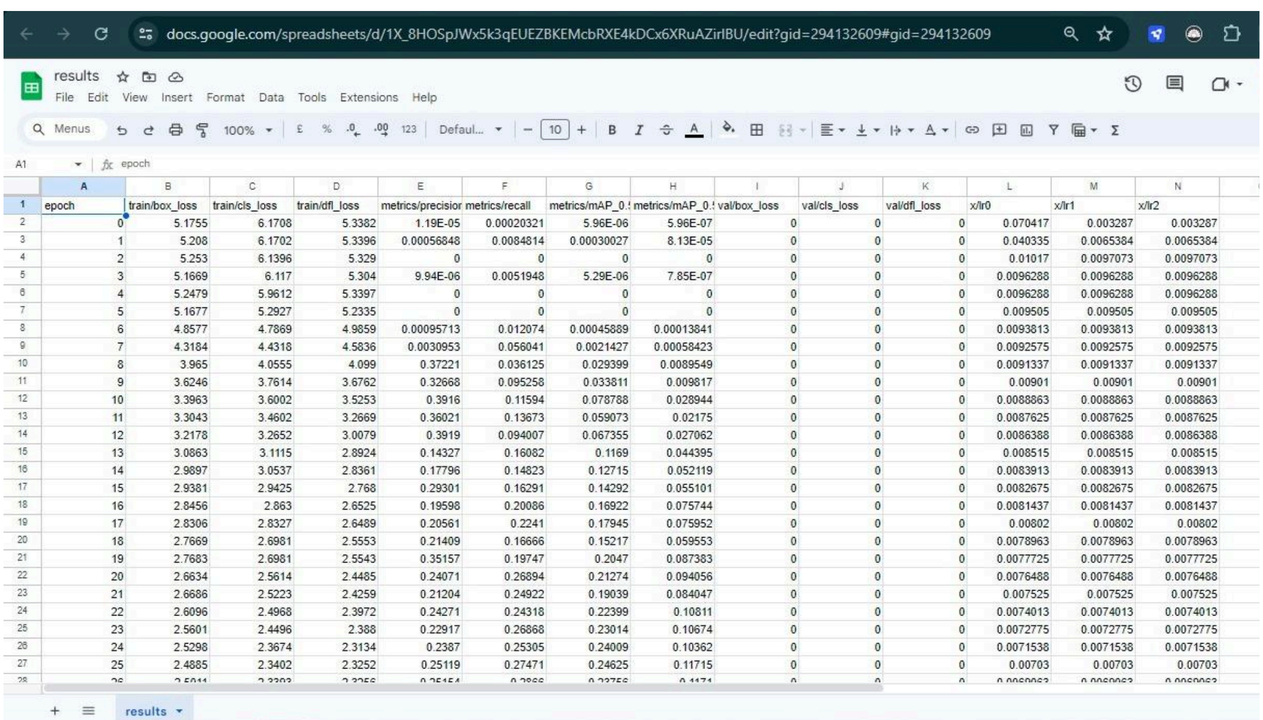
Our proposed dynamic traffic signal controller introduces several novel elements compared to existing machine learning solutions. While traditional systems rely on static timers or simplistic vehicle counting methods, our approach leverages advanced image preprocessing, annotation, and augmentation via RoboFlow, ensuring high-quality and diverse training data. The integration of the YOLOv9 model, renowned for its superior accuracy and speed in object detection, enhances the system's ability to analyze vehicle density, and queue length, and detect emergency vehicles in real time.

The element of novelty lies in our comprehensive use of state-of-the-art techniques to create an adaptive traffic control system. By incorporating real-time data analysis and machine learning, our solution dynamically adjusts traffic signals based on current conditions, rather than pre-set intervals. This results in optimized traffic flow, reduced congestion, and improved response times for emergency vehicles, setting our methodology apart from existing, less adaptive ML solutions.

- ## Results

Task 2.1: Vehicle detection and counting

After training the model on the custom dataset for 75 epochs to detect vehicles, the following are the key results:



epoch	train/box_loss	train/cls_loss	train/dfl_loss	metrics/precisio	metrics/recall	metrics/mAP_0.	metrics/mAP_0.	val/box_loss	val/cls_loss	val/dfl_loss	x/lr0	x/lr1	x/lr2
0	5.1755	6.1708	5.3382	1.19E-05	0.00020321	5.96E-06	5.96E-07	0	0	0	0.070417	0.003287	0.003287
1	5.208	6.1702	5.3396	0.00056848	0.0084814	0.00030027	8.13E-05	0	0	0	0.040335	0.0065384	0.0065384
2	5.253	6.1396	5.329	0	0	0	0	0	0	0	0.01017	0.0097073	0.0097073
3	5.1669	6.117	5.304	9.94E-06	0.0051948	5.29E-06	7.85E-07	0	0	0	0.0096288	0.0096288	0.0096288
4	5.2479	5.9612	5.3397	0	0	0	0	0	0	0	0.0096288	0.0096288	0.0096288
5	5.1677	5.2927	5.2335	0	0	0	0	0	0	0	0.009505	0.009505	0.009505
6	4.8577	4.7869	4.9859	0.00095713	0.012074	0.00045889	0.00013841	0	0	0	0.0093813	0.0093813	0.0093813
7	4.3184	4.4318	4.5836	0.0030953	0.056041	0.0021427	0.00058423	0	0	0	0.0092575	0.0092575	0.0092575
8	3.965	4.0555	4.099	0.37221	0.036125	0.029399	0.0089549	0	0	0	0.0091337	0.0091337	0.0091337
9	3.6246	3.7614	3.6762	0.32668	0.095258	0.033811	0.009817	0	0	0	0.00901	0.00901	0.00901
10	3.3963	3.6002	3.5253	0.3916	0.11594	0.078788	0.028944	0	0	0	0.0088863	0.0088863	0.0088863
11	3.3043	3.4602	3.2669	0.36021	0.13673	0.059073	0.02175	0	0	0	0.0087625	0.0087625	0.0087625
12	3.2178	3.2652	3.0079	0.3919	0.094007	0.067355	0.027062	0	0	0	0.0086388	0.0086388	0.0086388
13	3.0863	3.1115	2.8924	0.14327	0.16082	0.1169	0.044395	0	0	0	0.008515	0.008515	0.008515
14	2.9897	3.0537	2.8361	0.17796	0.14823	0.12715	0.052119	0	0	0	0.0083913	0.0083913	0.0083913
15	2.9381	2.9425	2.768	0.29301	0.16291	0.14292	0.055101	0	0	0	0.0082675	0.0082675	0.0082675
16	2.8456	2.863	2.6525	0.19598	0.20086	0.16922	0.075744	0	0	0	0.0081437	0.0081437	0.0081437
17	2.8306	2.8327	2.6489	0.20561	0.2241	0.17945	0.075952	0	0	0	0.00802	0.00802	0.00802
18	2.7669	2.6981	2.5553	0.21409	0.16666	0.15217	0.059553	0	0	0	0.0078963	0.0078963	0.0078963
19	2.7683	2.6981	2.5543	0.35157	0.19747	0.2047	0.087383	0	0	0	0.0077725	0.0077725	0.0077725
20	2.6634	2.5614	2.4485	0.24071	0.26894	0.21274	0.094056	0	0	0	0.0076488	0.0076488	0.0076488
21	2.6686	2.5223	2.4259	0.21204	0.24922	0.19039	0.084047	0	0	0	0.007525	0.007525	0.007525
22	2.6096	2.4968	2.3972	0.24271	0.24318	0.22399	0.10811	0	0	0	0.0074013	0.0074013	0.0074013
23	2.5601	2.4496	2.388	0.22917	0.26868	0.23014	0.10674	0	0	0	0.0072775	0.0072775	0.0072775
24	2.5298	2.3674	2.3134	0.2387	0.25305	0.24009	0.10362	0	0	0	0.0071538	0.0071538	0.0071538
25	2.4885	2.3402	2.3252	0.25119	0.27471	0.24625	0.11715	0	0	0	0.00703	0.00703	0.00703

docs.google.com/spreadsheets/d/1X_8HOSpJWx5k3qEUEZBKEMcbRXE4kDCx6XRuAZirlBU/edit?gid=294132609#gid=294132609

results

File Edit View Insert Format Data Tools Extensions Help

A1 epoch

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
28	26	2.5011	2.3393	2.3256	0.25154	0.2866	0.23756	0.1171	0	0	0	0.0069063	0.0069063	0.0069063
29	27	2.489	2.2685	2.2716	0.27221	0.27794	0.25474	0.11918	0	0	0	0.0067825	0.0067825	0.0067825
30	28	2.4158	2.1998	2.2525	0.29055	0.27071	0.25503	0.1185	0	0	0	0.0066587	0.0066587	0.0066587
31	29	2.4243	2.1955	2.2435	0.26707	0.27792	0.23994	0.11551	0	0	0	0.006535	0.006535	0.006535
32	30	2.4103	2.1874	2.2289	0.28954	0.28221	0.27106	0.12894	0	0	0	0.0064112	0.0064112	0.0064112
33	31	2.386	2.1723	2.2344	0.24844	0.29478	0.24106	0.11629	0	0	0	0.0062875	0.0062875	0.0062875
34	32	2.3585	2.1448	2.2291	0.26725	0.27755	0.26364	0.12988	0	0	0	0.0061637	0.0061637	0.0061637
35	33	2.3908	2.1193	2.1854	0.26445	0.30078	0.26338	0.13123	0	0	0	0.00604	0.00604	0.00604
36	34	2.3364	2.0702	2.1719	0.27846	0.2893	0.27856	0.13877	0	0	0	0.0059163	0.0059163	0.0059163
37	35	2.3281	2.0313	2.1469	0.28242	0.29324	0.28426	0.13793	0	0	0	0.0057925	0.0057925	0.0057925
38	36	2.3062	1.9639	2.0926	0.30837	0.2803	0.28919	0.14475	0	0	0	0.0056688	0.0056688	0.0056688
39	37	2.2973	1.9935	2.1403	0.2823	0.3082	0.2867	0.14316	0	0	0	0.005545	0.005545	0.005545
40	38	2.2795	1.9562	2.0993	0.27792	0.31288	0.27763	0.13421	0	0	0	0.0054212	0.0054212	0.0054212
41	39	2.2894	1.9506	2.1143	0.29091	0.31728	0.28749	0.1409	0	0	0	0.0052975	0.0052975	0.0052975
42	40	2.279	1.9507	2.09	0.28913	0.31369	0.28877	0.14133	0	0	0	0.0051737	0.0051737	0.0051737
43	41	2.2519	1.9267	2.0773	0.28445	0.30197	0.26967	0.13096	0	0	0	0.00505	0.00505	0.00505
44	42	2.2122	1.9235	2.099	0.28395	0.32458	0.2836	0.13995	0	0	0	0.0049263	0.0049263	0.0049263
45	43	2.221	1.8909	2.0725	0.28945	0.31342	0.28526	0.14418	0	0	0	0.0048025	0.0048025	0.0048025
46	44	2.2258	1.9151	2.0716	0.30091	0.30675	0.27218	0.13309	0	0	0	0.0046788	0.0046788	0.0046788
47	45	2.1974	1.8195	2.0192	0.29872	0.31907	0.29392	0.15061	0	0	0	0.004555	0.004555	0.004555
48	46	2.1707	1.8268	2.0084	0.30927	0.33119	0.29912	0.15032	0	0	0	0.0044313	0.0044313	0.0044313
49	47	2.1799	1.8249	2.0729	0.30982	0.31526	0.29056	0.14874	0	0	0	0.0043075	0.0043075	0.0043075
50	48	2.1735	1.8146	2.0303	0.31404	0.31456	0.30635	0.1588	0	0	0	0.0041837	0.0041837	0.0041837
51	49	2.1763	1.823	2.044	0.2899	0.34489	0.29501	0.1543	0	0	0	0.00406	0.00406	0.00406
52	50	2.1649	1.8049	2.0219	0.30302	0.33843	0.30495	0.1596	0	0	0	0.0039362	0.0039362	0.0039362
53	51	2.2011	1.8421	2.0723	0.29731	0.31369	0.29435	0.1552	0	0	0	0.0038125	0.0038125	0.0038125
54	52	2.1457	1.7721	2.0171	0.31633	0.33296	0.3097	0.16189	0	0	0	0.0036888	0.0036888	0.0036888

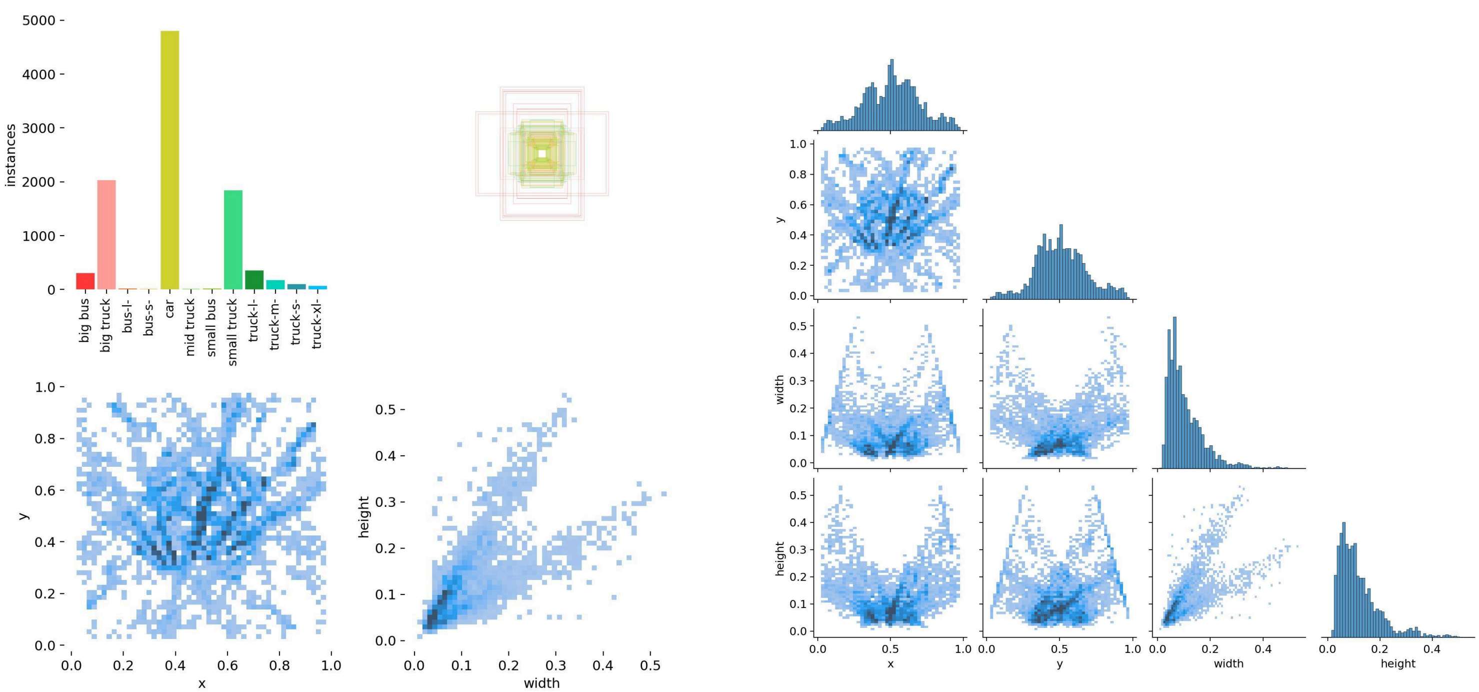
docs.google.com/spreadsheets/d/1X_8HOSpJWx5k3qEUEZBKEMcbRXE4kDCx6XRuAZirlBU/edit?gid=294132609#gid=294132609

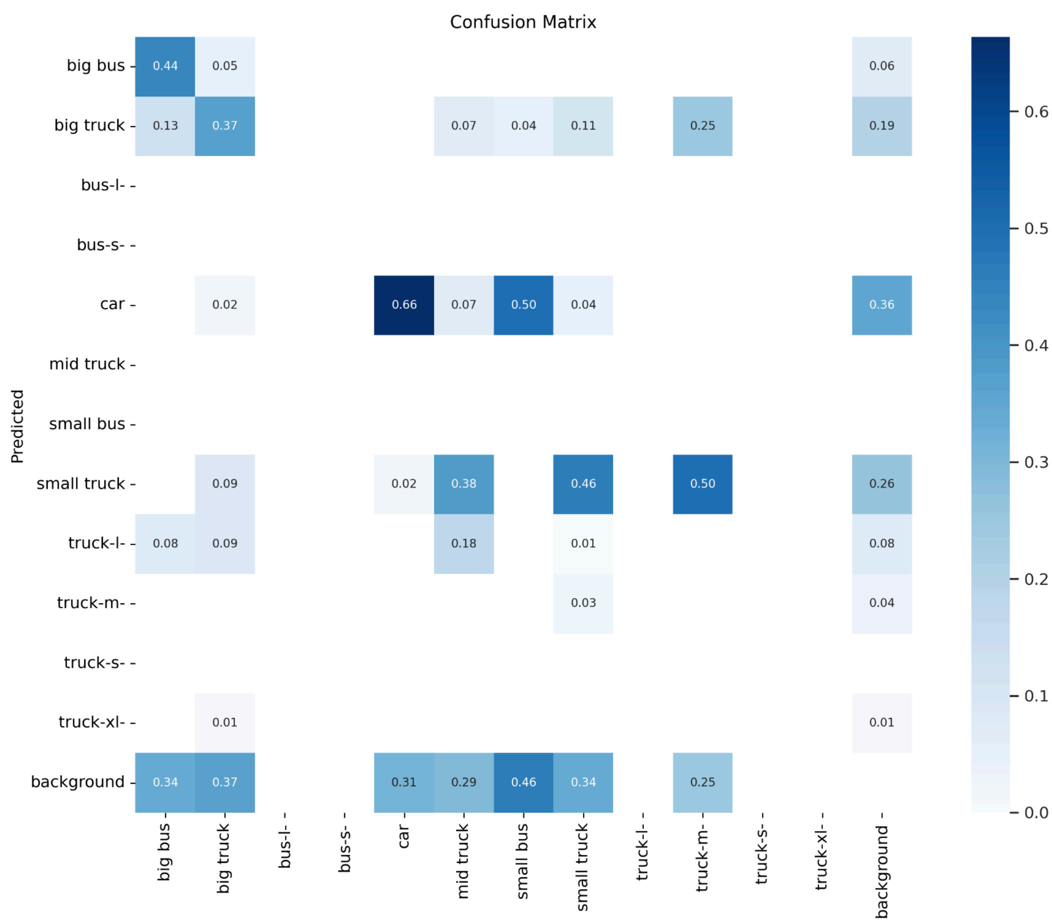
results

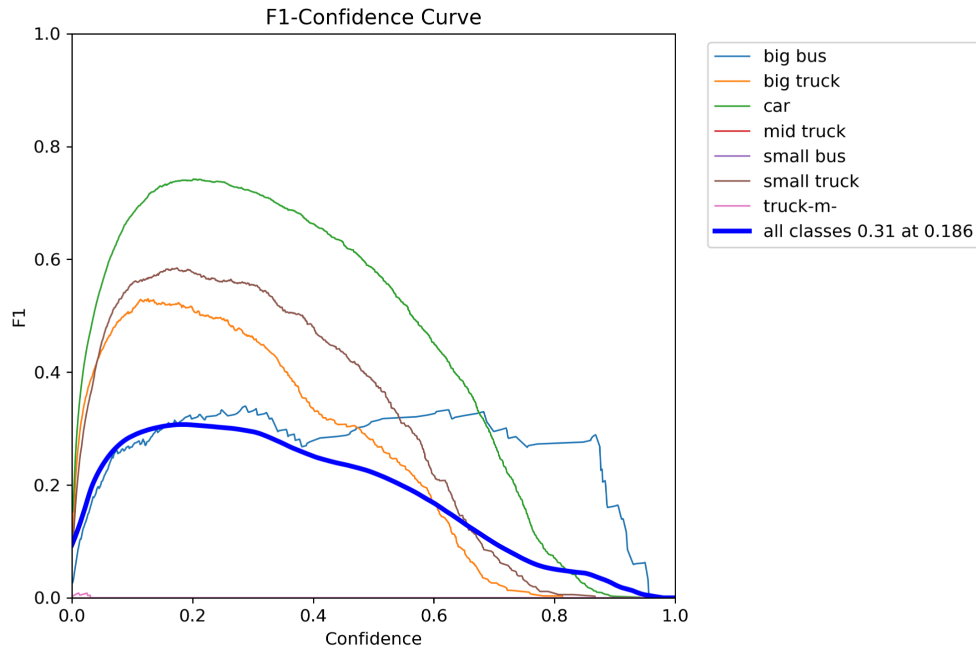
File Edit View Insert Format Data Tools Extensions Help

A1 epoch

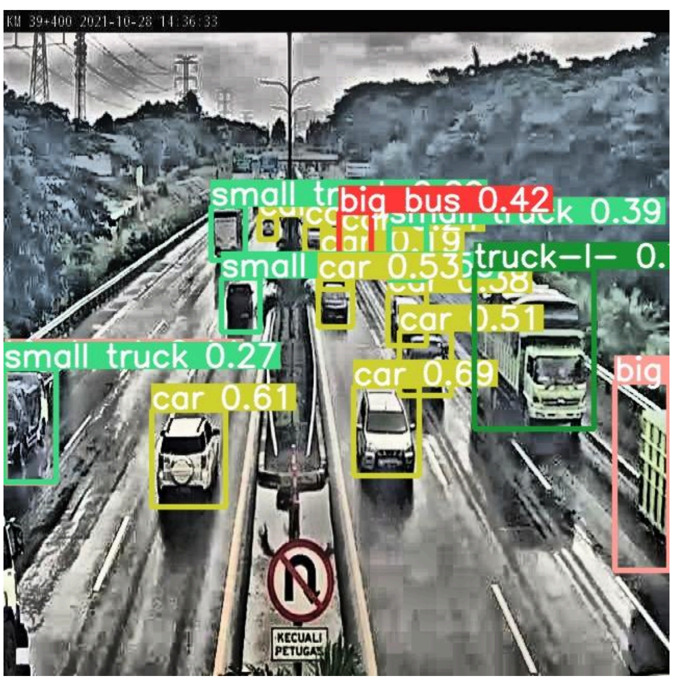
	A	B	C	D	E	F	G	H	I	J	K	L	M	N
51	49	2.1763	1.823	2.044	0.2899	0.34489	0.29501	0.1543	0	0	0	0.00406	0.00406	0.00406
52	50	2.1649	1.8049	2.0219	0.30302	0.33843	0.30495	0.1596	0	0	0	0.0039362	0.0039362	0.0039362
53	51	2.2011	1.8421	2.0723	0.29731	0.31369	0.29435	0.1552	0	0	0	0.0038125	0.0038125	0.0038125
54	52	2.1457	1.7721	2.0171	0.31633	0.33296	0.3097	0.16189	0	0	0	0.0036888	0.0036888	0.0036888
55	53	2.1265	1.7812	2.0357	0.44396	0.34808	0.30417	0.15931	0	0	0	0.003565	0.003565	0.003565
56	54	2.1041	1.7248	2.0193	0.29835	0.33069	0.2985	0.15216	0	0	0	0.0034413	0.0034413	0.0034413
57	55	2.1699	1.7783	2.0354	0.30413	0.31596	0.28966	0.14725	0	0	0	0.0033175	0.0033175	0.0033175
58	56	2.102	1.7404	1.9701	0.32409	0.3262	0.30063	0.15638	0	0	0	0.0031938	0.0031938	0.0031938
59	57	2.0749	1.7054	1.9688	0.45895	0.32186	0.30139	0.15615	0	0	0	0.00307	0.00307	0.00307
60	58	2.0995	1.7042	1.9658	0.45281	0.32715	0.30045	0.15537	0	0	0	0.0029462	0.0029462	0.0029462
61	59	2.0901	1.7206	1.9856	0.44209	0.334	0.30372	0.16001	0	0	0	0.0028225	0.0028225	0.0028225
62	60	2.0964	1.6911	1.969	0.47143	0.33703	0.30709	0.16055	0	0	0	0.0026987	0.0026987	0.0026987
63	61	2.0689	1.6606	1.9428	0.44002	0.32644	0.2853	0.14801	0	0	0	0.002575	0.002575	0.002575
64	62	2.0851	1.6873	1.9631	0.45014	0.33575	0.29032	0.15111	0	0	0	0.0024513	0.0024513	0.0024513
65	63	2.0522	1.6591	1.9668	0.47011	0.3198	0.30143	0.15777	0	0	0	0.0023275	0.0023275	0.0023275
66	64	2.0621	1.6401	1.9452	0.45243	0.33842	0.31171	0.16693	0	0	0	0.0022038	0.0022038	0.0022038
67	65	2.0607	1.6627	1.9987	0.46691	0.32771	0.30512	0.16068	0	0	0	0.00208	0.00208	0.00208
68	66	2.0167	1.5952	1.9494	0.47532	0.31646	0.30166	0.15744	0	0	0	0.0019563	0.0019563	0.0019563
69	67	2.0854	1.684	1.9814	0.45861	0.33106	0.31093	0.16597	0	0	0	0.0018325	0.0018325	0.0018325
70	68	2.0542	1.6437	1.9906	0.46555	0.33675	0.30935	0.16001	0	0	0	0.0017087	0.0017087	0.0017087
71	69	2.0399	1.6283	1.9365	0.47458	0.33362	0.3123	0.16517	0	0	0	0.001585	0.001585	0.001585
72	70	2.0404	1.6192	1.9531	0.47218	0.32935	0.31683	0.16804	0	0	0	0.0014612	0.0014612	0.0014612
73	71	2.0205	1.5995	1.9213	0.45037	0.33631	0.30696	0.16148	0	0	0	0.0013375	0.0013375	0.0013375
74	72	2.0519	1.6211	1.9458	0.45578	0.34291	0.31704	0.16871	0	0	0	0.0012138	0.0012138	0.0012138
75	73	2.0511	1.6519	1.9681	0.45719	0.32958	0.31154	0.16529	0	0	0	0.00109	0.00109	0.00109
76	74	2.0424	1.6186	1.9083	0.45216	0.3446	0.31415	0.16753	0	0	0	0.00096625	0.00096625	0.00096625
77	75	2.0194	1.5923	1.9211	0.46325	0.33835	0.30842	0.16464	0	0	0	0.0008425	0.0008425	0.0008425

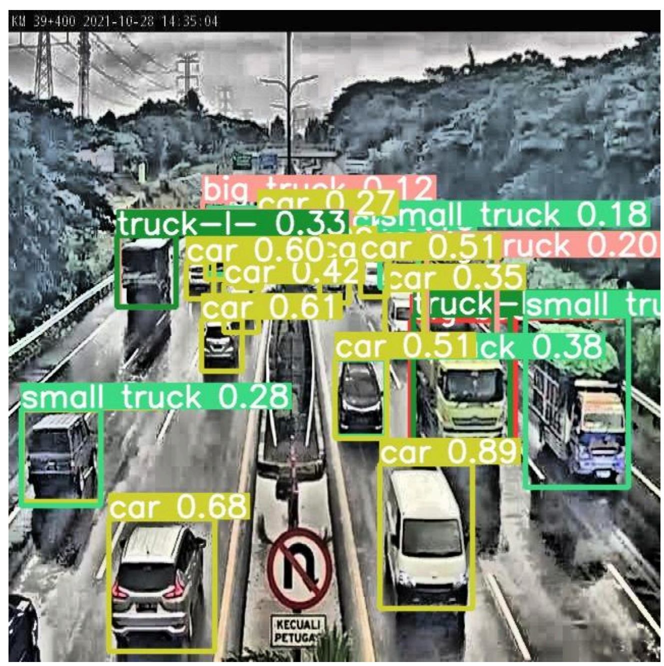

instances
big bus
big truck
bus-l-
bus-s-
car
mid truck
small bus
small truck
truck-l-
truck-m-
truck-s-
truck-xl-
x
y
width
height


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



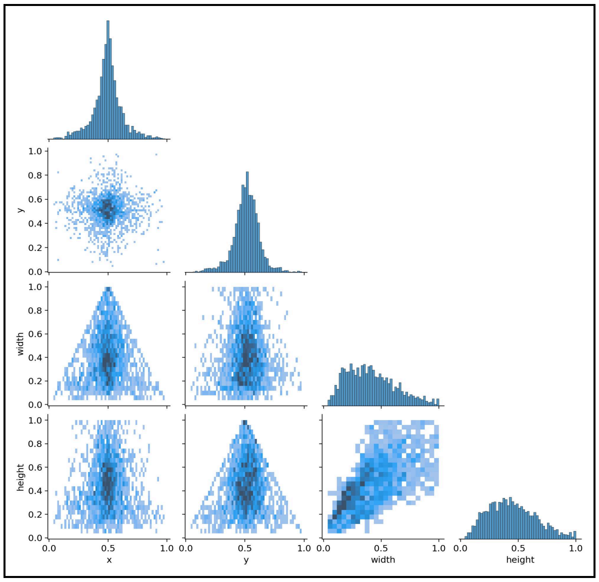
Result on a few of the test images:

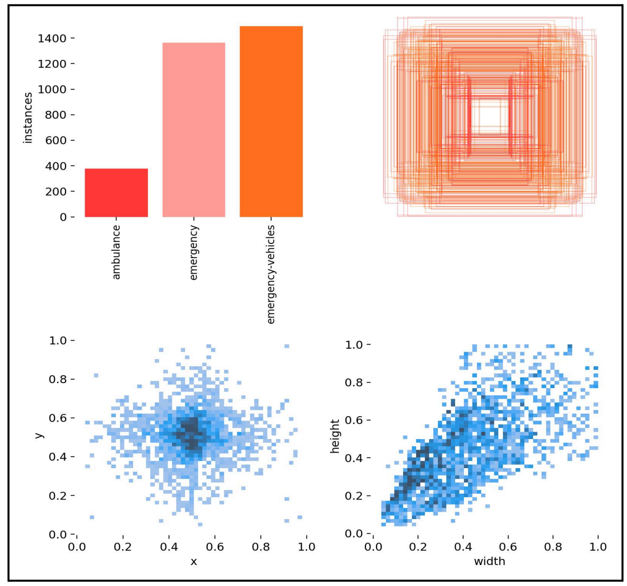




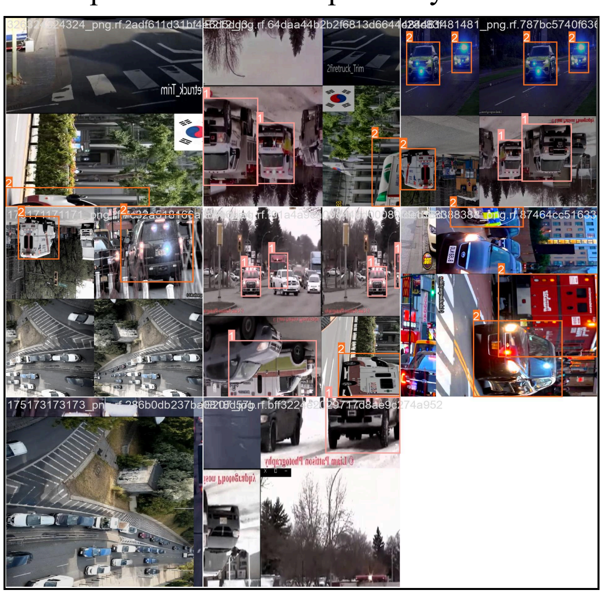
Task 2.2: Emergency Vehicle Detection

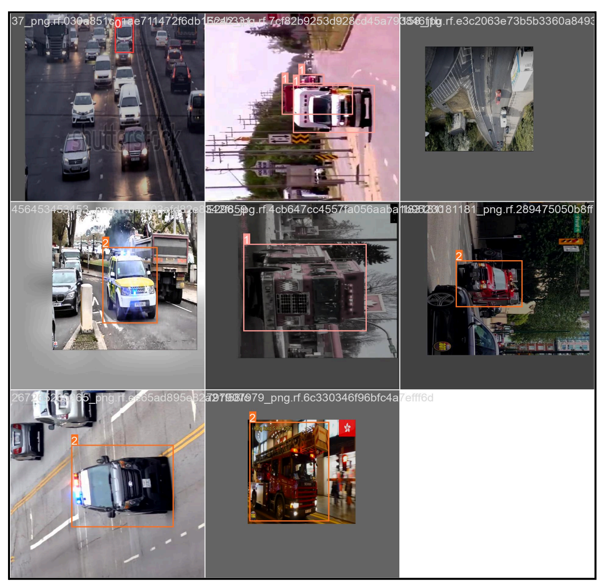
With a similar approach, we deduced a working model for emergency vehicles such as ambulances, fire trucks and police vans for lane prioritization. The process described in Fig. 2.3 was applied for the same which involved the extensive usage of RoboFlow for image processing and datasets and YoloV9 for training and model preparation. These models can be readily used in any Python inference and put into use. The following two images are the results of annotation and labelling of the datasets ran on Google Colab with GPU units.





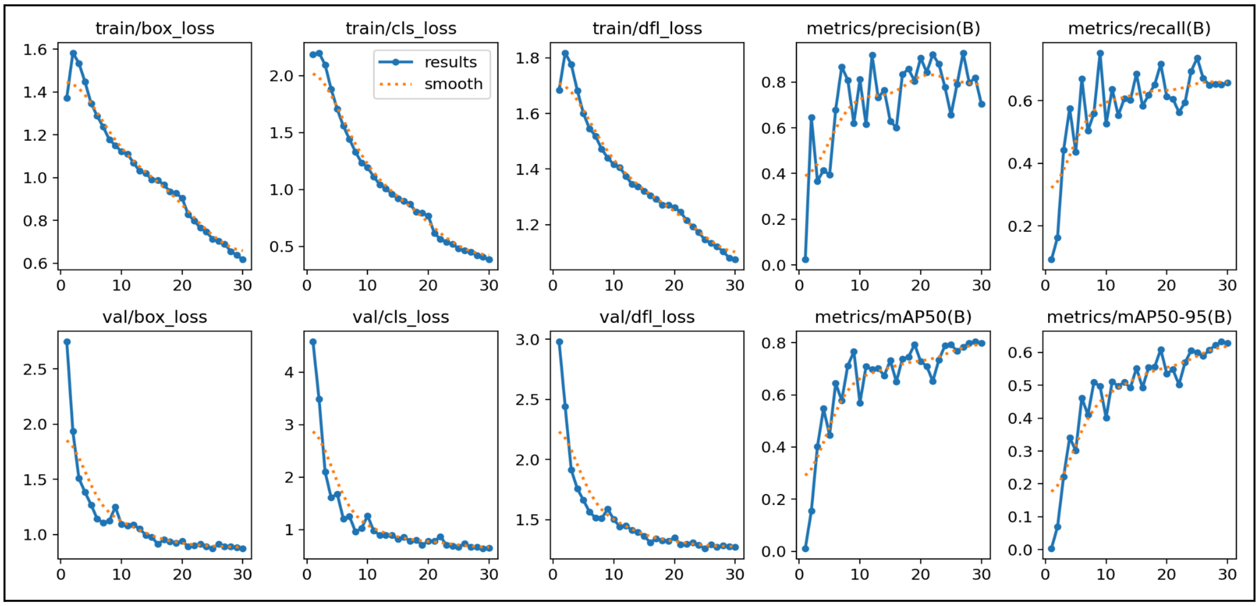
The above two figures represent the label and annotation distribution and instances/occurrences of that particular class respectively.

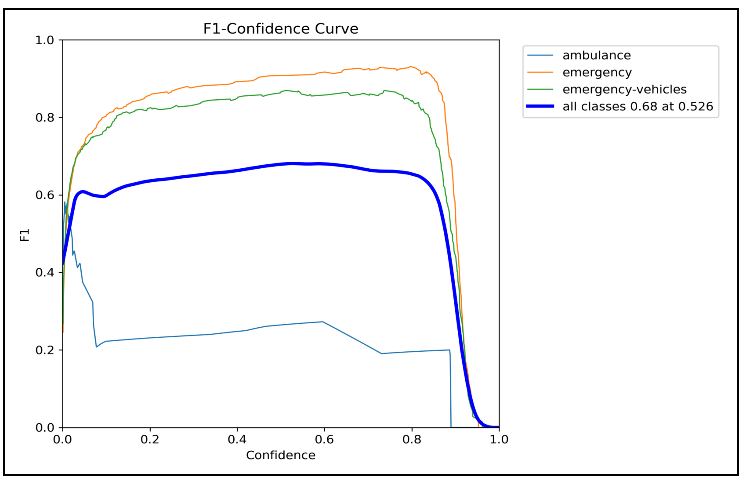


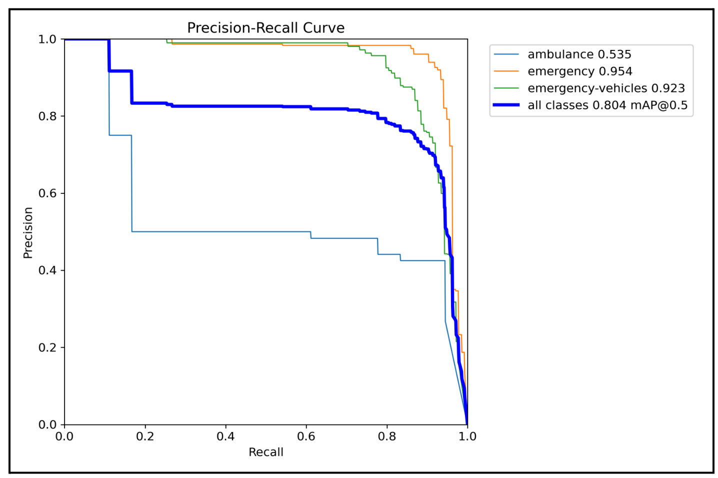


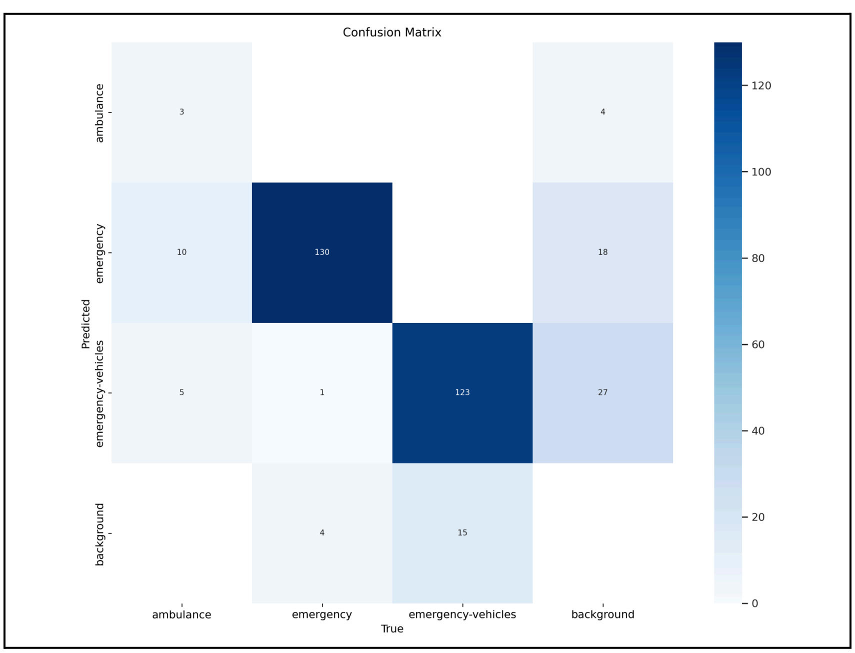
Two examples of the training batches used for training.

The below figure shows the results after training the model using YOLOV9.









epoch	train/box_loss	train/cls_loss	train/dfl_loss	metrics/precisior	metrics/recall(B)	metrics/mAP50(I	metrics/mAP50-!	val/box_loss	val/cls_loss	val/dfl_loss	lr/pg0	lr/pg1	lr/pg2
1	1.373	2.186	1.6841	0.02398	0.09265	0.01123	0.00381	2.7482	4.5834	2.9795	0.00047508	0.00047508	0.00047508
2	1.5833	2.2018	1.8179	0.64469	0.1628	0.15509	0.06931	1.9358	3.4916	2.4401	0.00092001	0.00092001	0.00092001
3	1.5347	2.0966	1.7757	0.36721	0.44224	0.40196	0.2221	1.5074	2.1009	1.9123	0.0013335	0.0013335	0.0013335
4	1.4472	1.8827	1.6821	0.41434	0.57579	0.54703	0.3412	1.3829	1.6068	1.7575	0.0012875	0.0012875	0.0012875
5	1.3475	1.7064	1.5999	0.39385	0.43634	0.4455	0.30132	1.2685	1.6801	1.6614	0.0012404	0.0012404	0.0012404
6	1.2883	1.5585	1.5453	0.67783	0.66999	0.64408	0.46179	1.143	1.2049	1.5645	0.0011932	0.0011932	0.0011932
7	1.237	1.4428	1.5182	0.86748	0.50376	0.57881	0.4099	1.1054	1.2547	1.5142	0.0011461	0.0011461	0.0011461
8	1.1779	1.3313	1.4718	0.80832	0.55972	0.71163	0.50972	1.1241	0.9589	1.5105	0.0010989	0.0010989	0.0010989
9	1.1508	1.2357	1.4405	0.62024	0.75282	0.76534	0.49696	1.2483	1.0245	1.5885	0.0010517	0.0010517	0.0010517
10	1.1214	1.1948	1.416	0.81129	0.52598	0.56895	0.40124	1.0903	1.2597	1.5014	0.0010046	0.0010046	0.0010046
11	1.1114	1.114	1.4069	0.61483	0.63671	0.70825	0.51047	1.0796	0.979	1.441	0.00095743	0.00095743	0.00095743
12	1.0684	1.0429	1.3735	0.91751	0.55299	0.69785	0.49805	1.0881	0.89344	1.4507	0.00091027	0.00091027	0.00091027
13	1.0314	1.0097	1.3462	0.73187	0.60644	0.70137	0.50927	1.0506	0.8908	1.4114	0.00086312	0.00086312	0.00086312
14	1.0204	0.96027	1.3374	0.76449	0.60161	0.67393	0.49247	0.99441	0.88884	1.3962	0.00081596	0.00081596	0.00081596
15	0.99126	0.92156	1.3211	0.6291	0.68633	0.73137	0.55216	0.97471	0.81556	1.3635	0.0007688	0.0007688	0.0007688
16	0.98828	0.9011	1.3049	0.59994	0.58323	0.65014	0.49267	0.91189	0.85624	1.3103	0.00072164	0.00072164	0.00072164
17	0.96643	0.87211	1.2926	0.83477	0.61795	0.73843	0.55446	0.95218	0.77972	1.3414	0.00067449	0.00067449	0.00067449
18	0.93522	0.80159	1.2716	0.85696	0.6512	0.74507	0.55647	0.93605	0.7943	1.3271	0.00062733	0.00062733	0.00062733
19	0.92788	0.79413	1.2702	0.80459	0.71707	0.79357	0.60928	0.92197	0.70698	1.3231	0.00058017	0.00058017	0.00058017
20	0.90447	0.76814	1.2619	0.90555	0.61257	0.7284	0.53503	0.93851	0.77478	1.3502	0.00053302	0.00053302	0.00053302
21	0.82789	0.6154	1.2467	0.84282	0.60584	0.70986	0.54787	0.88949	0.77475	1.2945	0.00048586	0.00048586	0.00048586
22	0.79828	0.56437	1.2149	0.9207	0.56298	0.65194	0.50218	0.89697	0.86604	1.2986	0.0004387	0.0004387	0.0004387
23	0.76547	0.54049	1.1933	0.87872	0.59447	0.73399	0.56899	0.91254	0.70195	1.311	0.00039155	0.00039155	0.00039155
24	0.7467	0.51888	1.1723	0.77813	0.69333	0.78818	0.6058	0.88981	0.68232	1.287	0.00034439	0.00034439	0.00034439
25	0.71408	0.48252	1.146	0.65593	0.73639	0.7924	0.60023	0.87365	0.66994	1.2612	0.00029723	0.00029723	0.00029723
26	0.70399	0.46346	1.134	0.79103	0.67118	0.76768	0.58848	0.91261	0.73624	1.2932	0.00025007	0.00025007	0.00025007
27	0.69068	0.44967	1.122	0.92803	0.64987	0.78386	0.60792	0.88821	0.665	1.2702	0.00020292	0.00020292	0.00020292
28	0.65633	0.4227	1.1039	0.79722	0.65206	0.79888	0.62188	0.89	0.66888	1.2828	0.00015576	0.00015576	0.00015576
29	0.63804	0.40599	1.081	0.81901	0.65045	0.80398	0.63284	0.88256	0.64077	1.2755	0.0001086	0.0001086	0.0001086
30	0.61791	0.38534	1.0749	0.70391	0.65658	0.79891	0.62831	0.87397	0.64552	1.2707	6.14E-05	6.14E-05	6.14E-05

The above are the results after training the model. The epochs were set to 30 and the results are displayed in CSV format.

As we can see in the above picture, the model predicted the emergency vehicles with great accuracy. Also, we can see that the F1 score for all emergency vehicles in general is around 0.8, hence proving our model’s upper hand.

- ## Compared Methods

The majority of the existing models in India are pre-timed traffic signal systems. Since it is pre-timed, it means all the lanes get the same amount of green light time, irrespective of whether the lane is congested with traffic or fully free.

Our model suggests a smart traffic signal system, which can automatically determine which lane needs a higher green light time based on real-time conditions. In a junction, detecting which lane has a higher volume of traffic and green lighting that lane will reduce the overall traffic congestion significantly, compared to the existing signal systems used.

Comparing this model to other proposed models such as those mentioned in the literature survey (Vehicle detection and counting of a vehicle using openCV, Adaptive Traffic Light Based on Yolo-Darknet Object Detection, etc.) we see that many of them use older versions of YOLO. We have taken one of the latest versions of YOLO (v9), to utilize all the new features available. Yet another paper (Dynamic Traffic Control System with Reinforcement Learning Technique) uses Deep-Q Learning using phase-gated technique, and Speeded-Up Robust Features (SURF) for object detection. We have implemented OpenCV and YOLOv9 into our model. One of the key focuses in our model is the detection of Emergency Vehicles (EVs) and prioritizing lanes based on this information along with other data such as queue length and traffic density. In the papers mentioned above, detecting EVs is not a feature. Our model can benefit many people with this feature as vehicles like ambulances, police, etc can pass through the traffic quicker and avoid potential casualties/danger. Additionally, implementing blockchain technology into the system for extra security and protection of data is also done.

- ## Conclusions

The highest number of instances created was that of the cars, with around almost 5000. Following this are big and small trucks, with about 2000 instances each. Other vehicle types had significantly small numbers of instances.

In the confusion matrix, we can see that most of the cars are correctly predicted. Some of the smaller vehicles are wrongly predicted as cars. Trucks and buses also have some confusion. A large number of vehicles are getting predicted as background. This is so as many vehicle types have been underrepresented in the dataset.

By analyzing the F1 score, it can be concluded that the bus reached a maximum of 0.35 value which is the least of all the vehicles. The Rest of the vehicles had a good value of average precision and recall that made up the F1 score.

● References

[1] Pranav Shinde, Srinandini Yadav, Shivani Rudrake, Pravin Kumbhar - “Smart Traffic Control System using YOLO”, 2019

[2] Karthik Srivathsa D S, Dr Kamalraj R - “Vehicle Detection And Counting Of A Vehicle Using OpenCV”, 2021

[3] Hameed Khan, Kamal K. Kushwah, Muni Raj Maurya, Saurabh Singh, Prashant Jha, Sujeet K. Mahobia, Sanjay Soni, Subham Sahu, Kishor Kumar Sadasivuni - “Machine learning driven intelligent and self-adaptive system for traffic management in smart cities” - 2022

[4] V. Jyothirmai, N. K. Kameswara Rao - “Dynamic Traffic Control System with Reinforcement Learning Technique”, 2020

[5] Hersyanda Putra Adi ,Rd. Saifan Fachri Azharan - “Adaptive Traffic Light Based on Yolo-Darknet Object Detection”, 2019

[6] Geoferleen Flores, Eduardo Jr. Piedad, Anzeneth Figueroa, Romari Tumamak, Nesrah Jane Marie Berdon - “A Sound-based Machine Learning to Predict Track Vehicle Density” , 2021

[7] Sümeyye Aydin, Murat Taşyürek, Celal Öztürk - “Traffic Density Estimation using Machine Learning Methods” , 2021