Who are we?

Pixel Pilot is a new startup company that creates self-driving systems for cars in Singapore.

As an in-house data scientist, I am presenting our object detection model to our engineers and shareholders.

PIXEL PILOT

Evokes High-Resolution Object Detection

By: Michael King Sutanto

Agenda

1. Persona & Problem Statement
2. Data Collection, Cleaning, & EDA
3. Models
4. Conclusion
5. Limitations & Recommendations

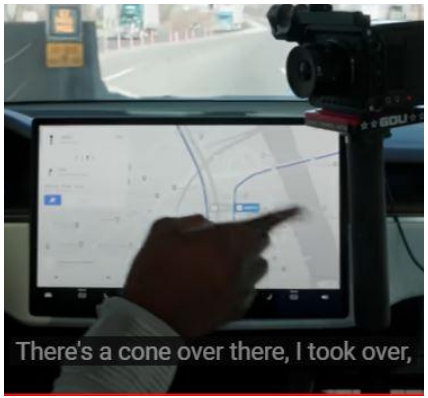
PERSONA & PROBLEM STATEMENT

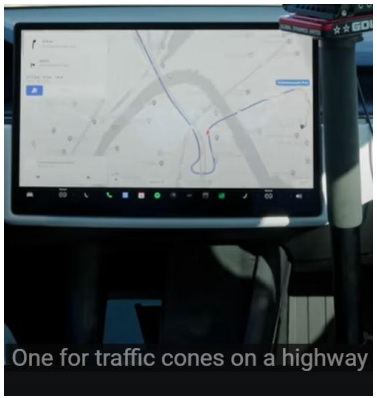
Meet Jason

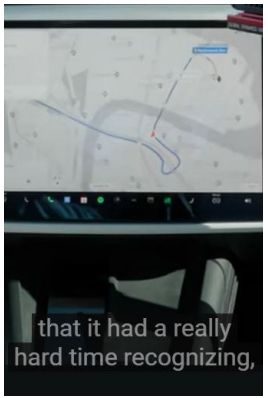
Jason is a 40-year-old business owner who commutes to work by **self-driving car** daily.

However he's often **dissatisfied** with its performance, particularly due to its handling of road obstacles like traffic cones, which forces him to frequently disengage the autopilot.

This recurring issue causes him **stress**. Jason believes that given the high cost of his vehicle, **it should possess better object detection capabilities**.

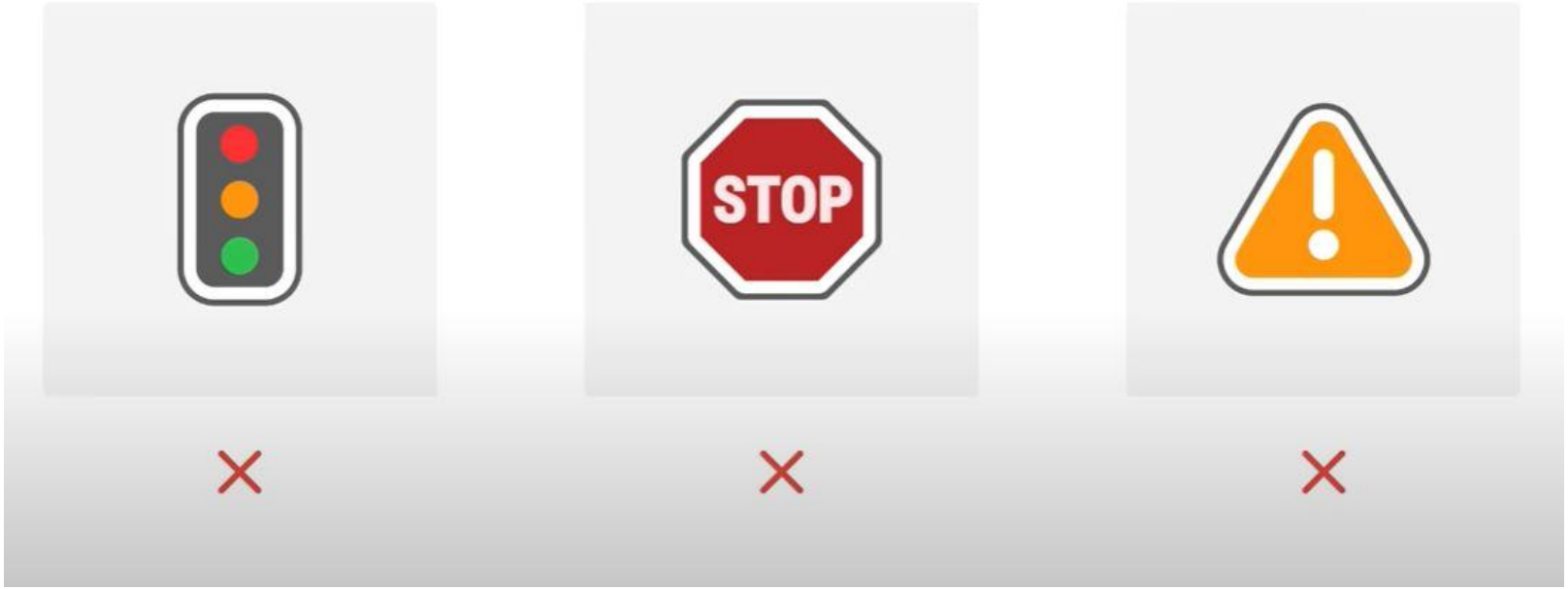






source

Traffic-Aware Cruise Control (TACC) as of 22 March 2024



“TACC responds to most objects but cannot recognize everything. TACC does not attempt to slow for traffic lights, stop signs or other traffic controls.”

source

Meet Janet

Janet, a 35-year-old marketing executive, drives a 2017 Honda Civic and is keen to **retrofit it with self-driving technology** to improve her daily commute.

She looks for a **reliable conversion kit** that integrates seamlessly with her **existing vehicle**, enhancing safety and efficiency without needing a new car.

Janet's challenges include ensuring that the kit can accurately **detect Singapore local vehicles, traffic signs and objects.**

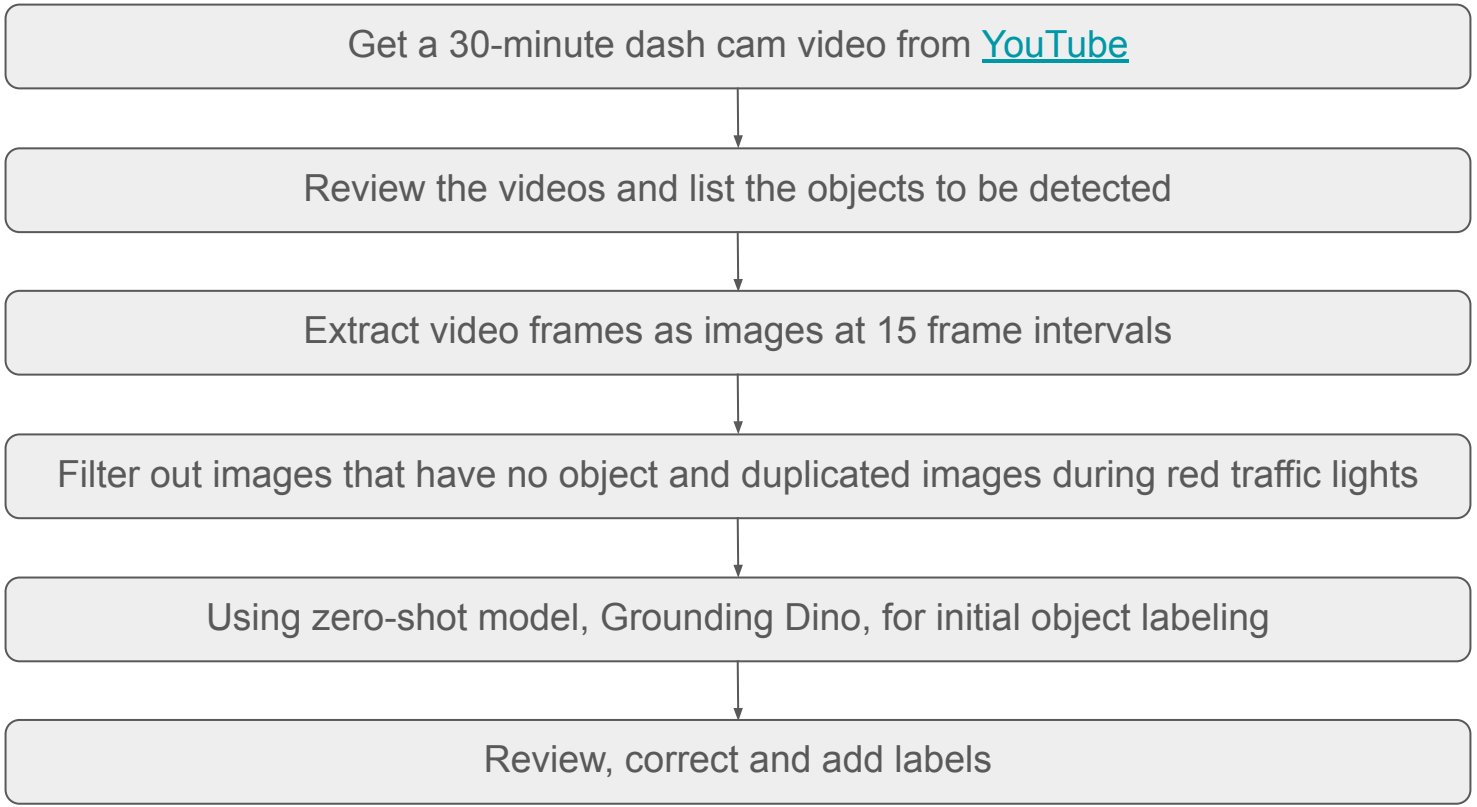
Her goal is to make commuting less stressful and more productive.

Problem Statement

How might we develop and integrate an advanced **object detection model** that can accurately identify **Singapore's local vehicles, traffic signs, and traffic-related objects** such as traffic cones, into existing car systems, enabling any vehicle to be equipped with reliable self-driving features?

DATA COLLECTION, CLEANING, & EDA

Data Collection



EDA - 25 Classes


Person


Bicycle


Motorcycle


Car

Bus


Truck

Directional


Turn Left


Keep Left


Split-way


Crosswalk


Zebra Cross


No Left


No Right


No Entry

Stop

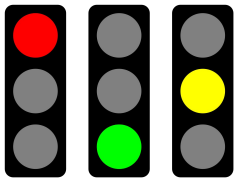
Max Height 4.5m

Max Speed 50 km/hr

Max Speed 60 km/hr

Max Speed 70 km/hr

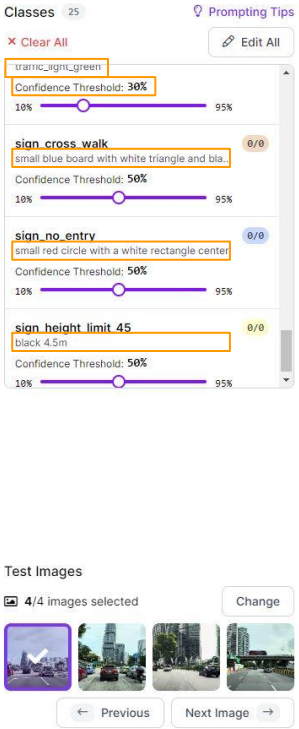
Temp Work-zone

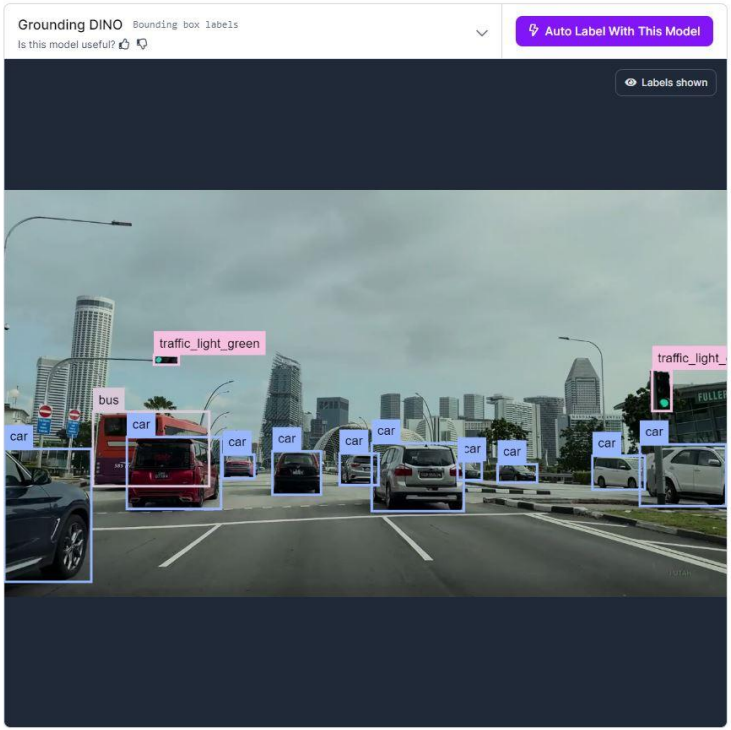

Traffic Light, Red, Green, Amber


Traffic Cone

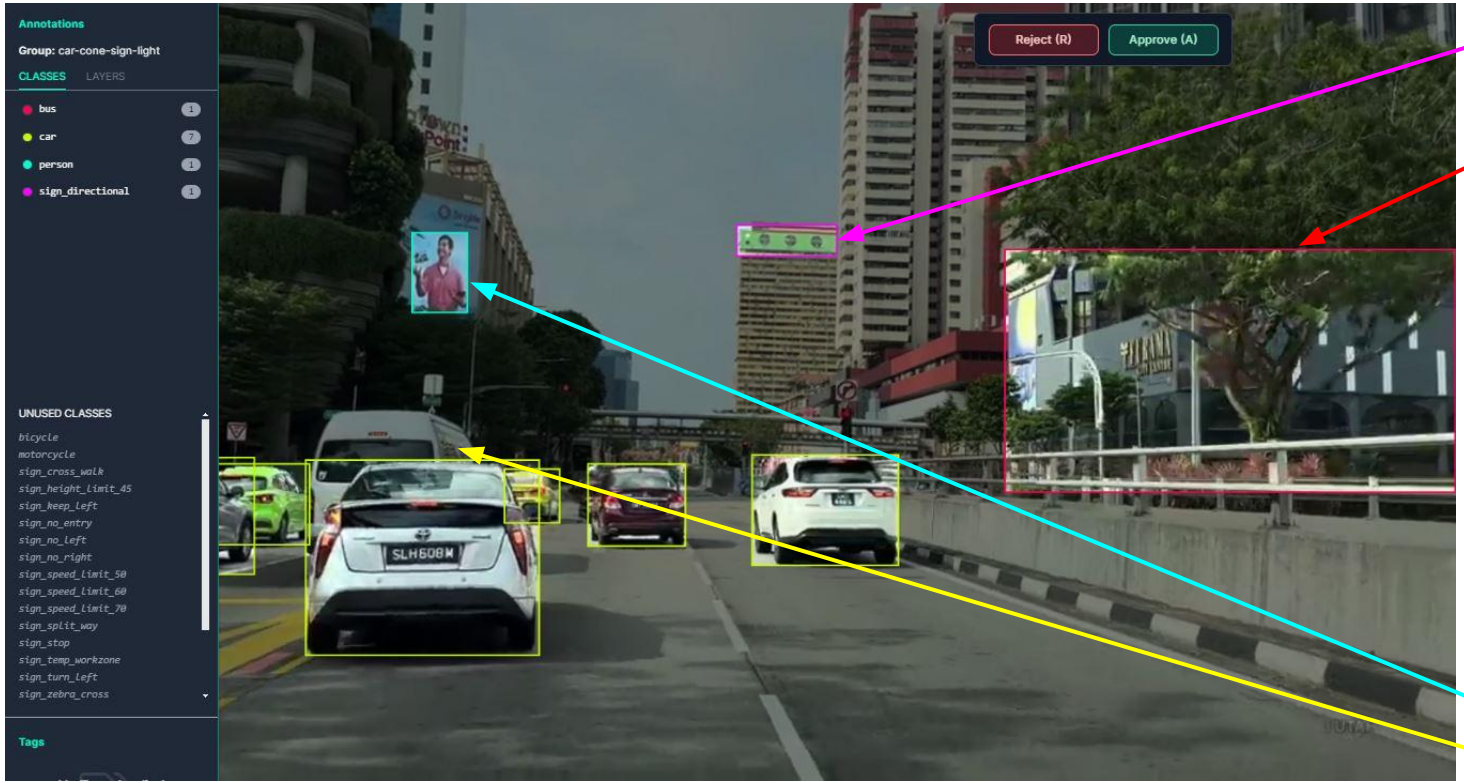
Labeling Using Grounding Dino Model in Roboflow

- Use **text prompts** to **detect objects**.
- Use **colors, shapes and size**. For example instead of **directional sign board**, we use **green rectangle board**
- Able to adjust **confidence threshold.**

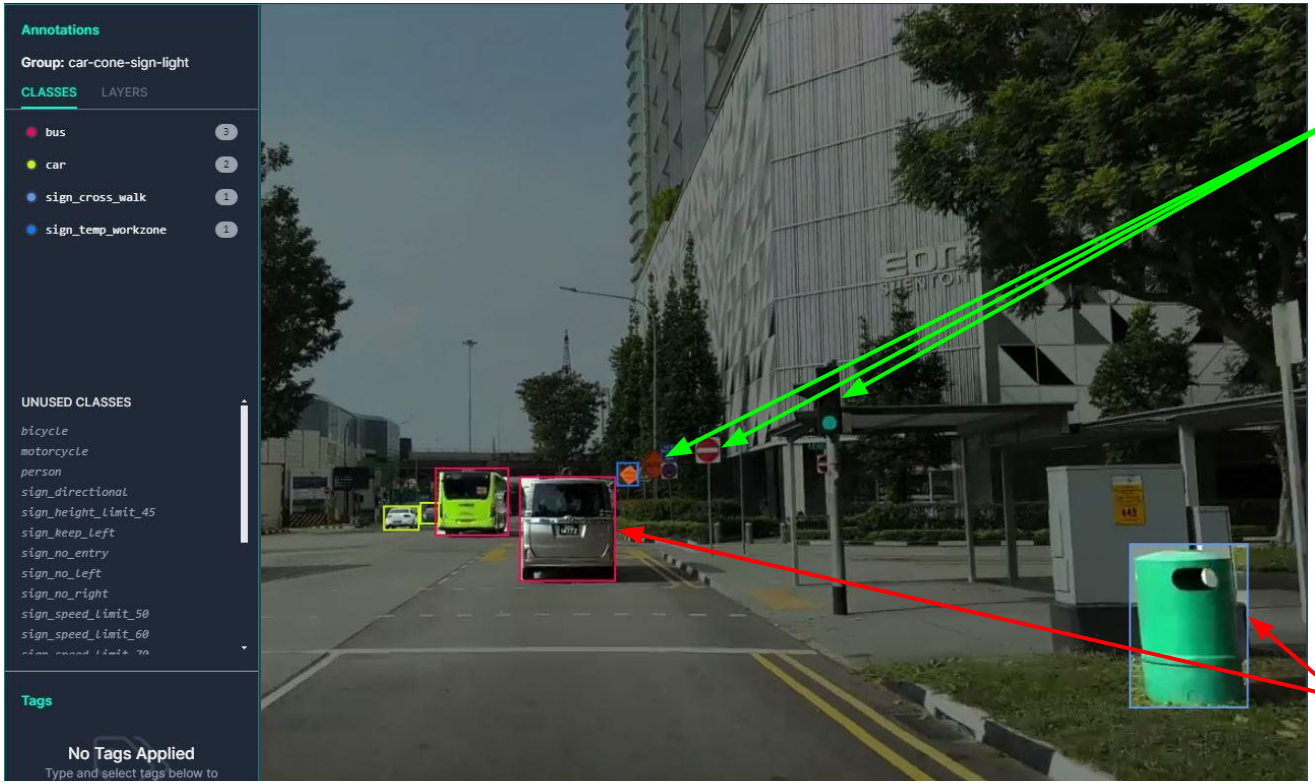




Grounding Dino - Bad Labels



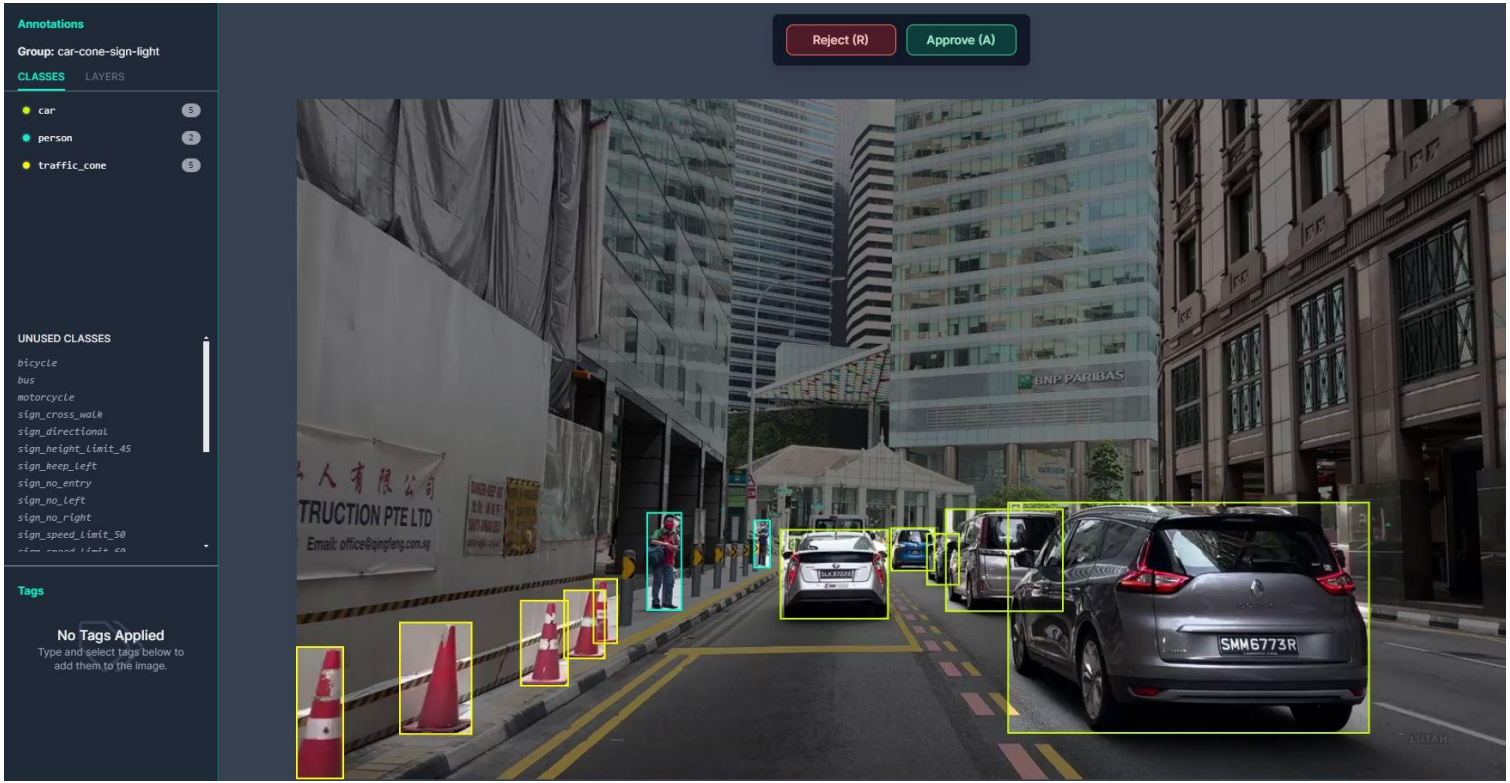
Grounding Dino - Bad Labels



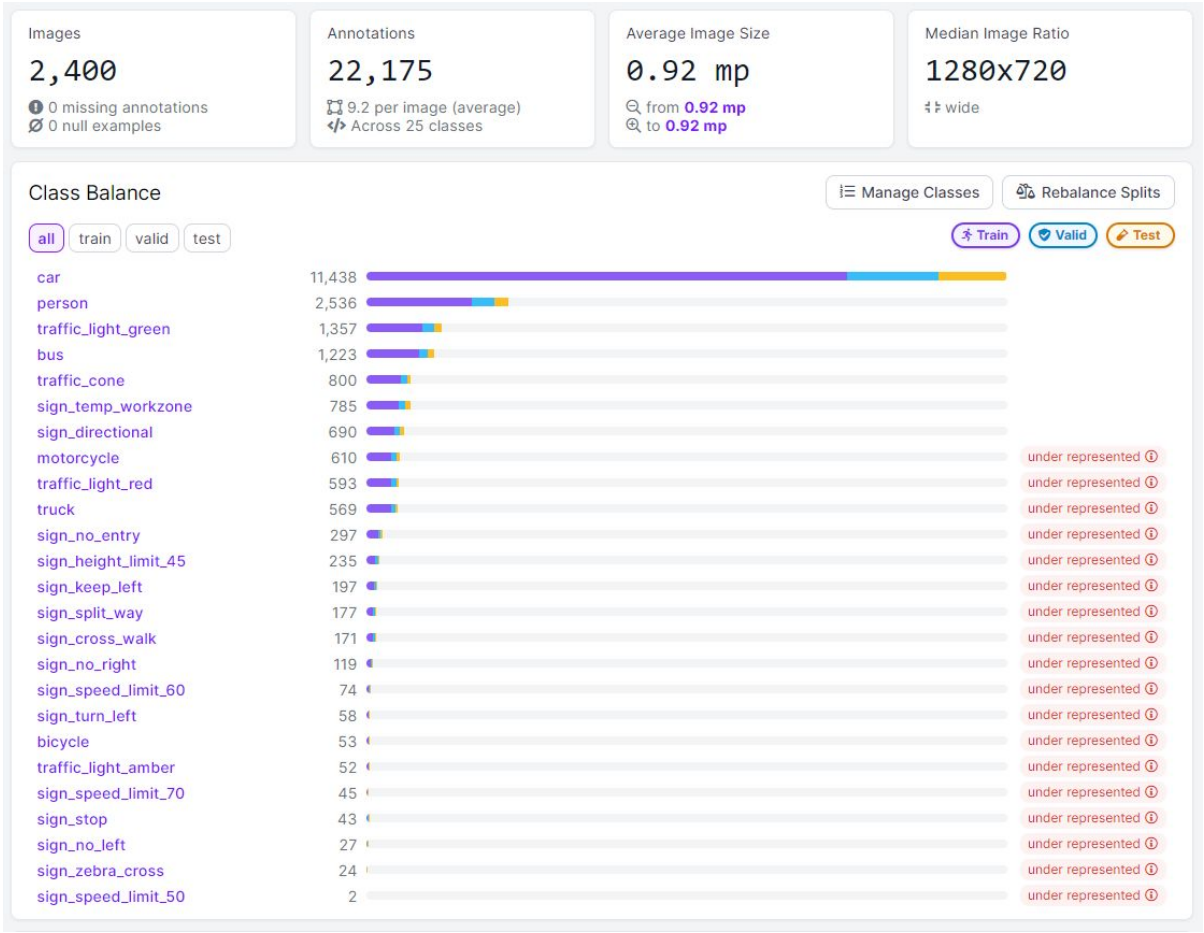
No label for traffic sign and traffic light

Wrong labels

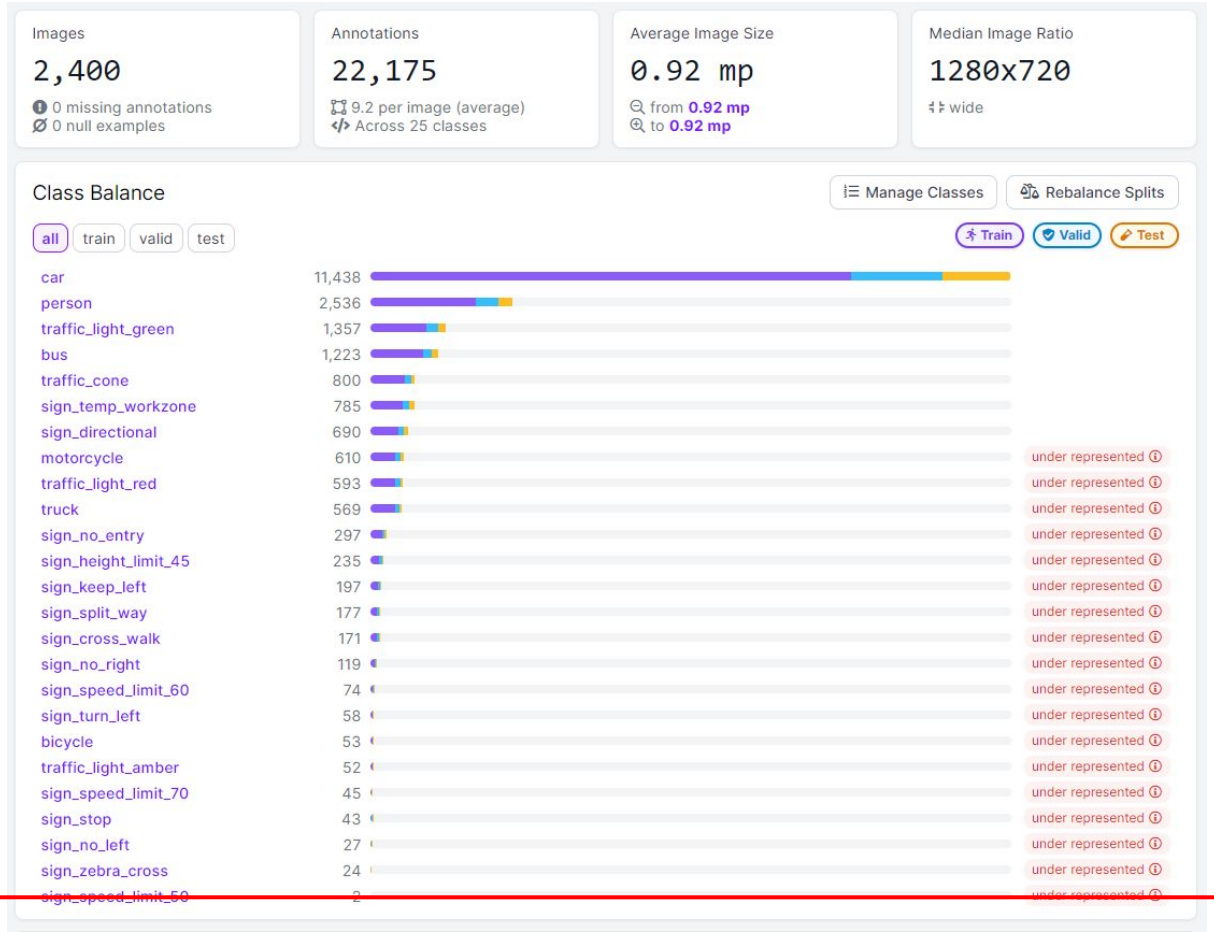
Grounding Dino - Good Labels



EDA - Initial Class Distribution



EDA - Initial Class Distribution



Removed

Adding Data for Minority Class

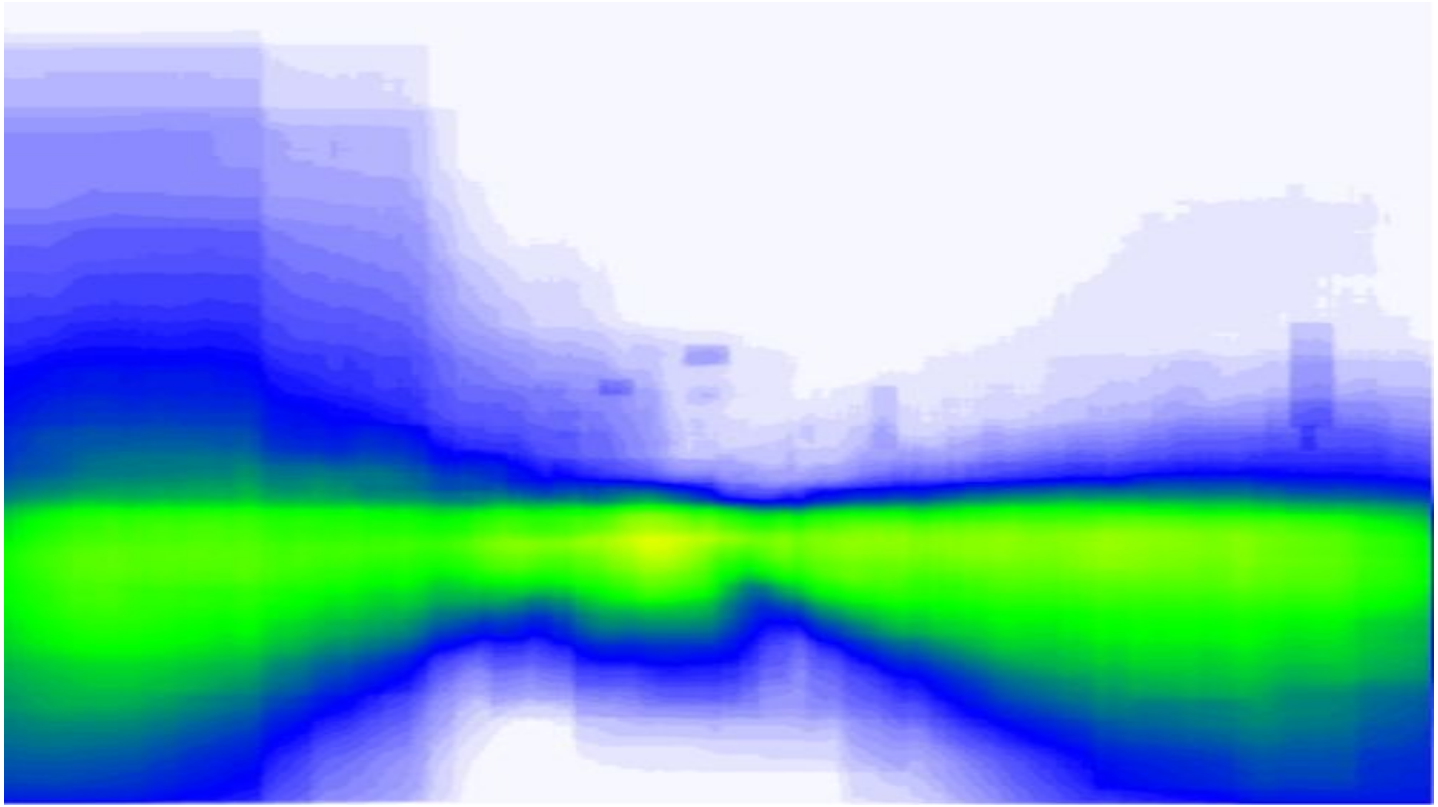
Use handheld phone camera and replicate the framing typically seen in dash cam footage

Importance of the Position of the Classes

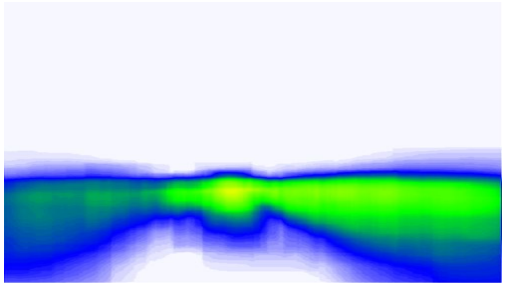
Models like YOLO often consider the entire scene (**contextual understanding**), which helps the system **differentiate between objects** based on their **position** and **size**.

For example traffic lights are often at the top, and traffic cones are often at the bottom.

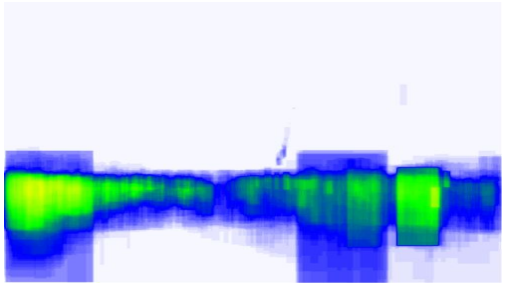
EDA - Class Count and Position Heatmap



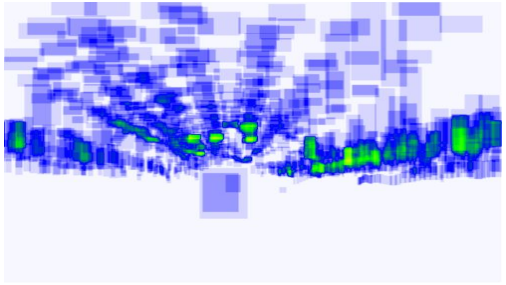
EDA - Top 6 Class Count and Position Heatmap



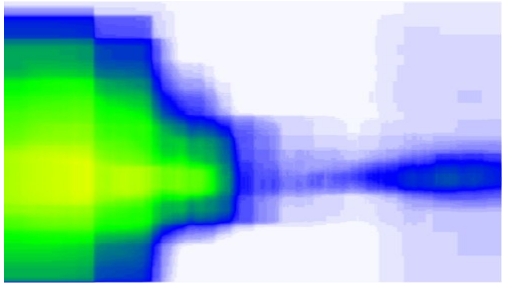
Car



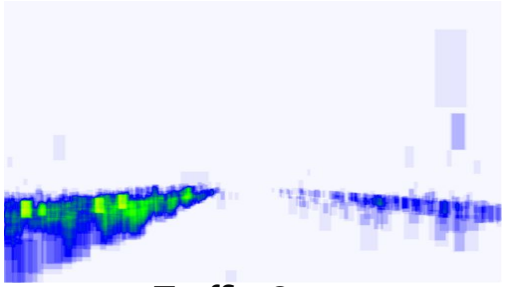
Person



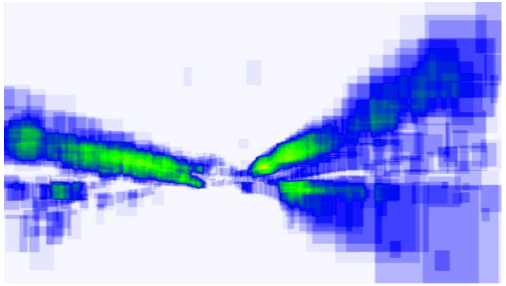
Traffic Light Green



Bus

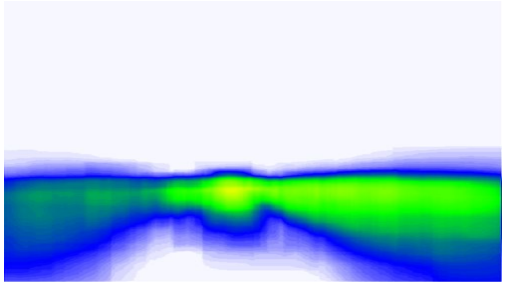


Traffic Cone

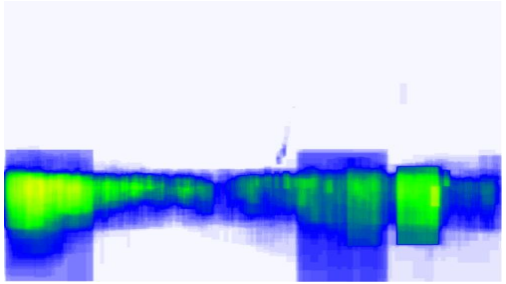


Temp Work-Zone Sign

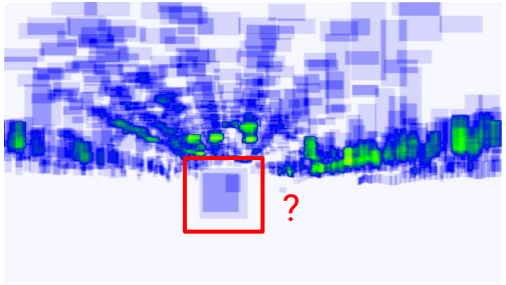
EDA - Top 6 Class Count and Position Heatmap



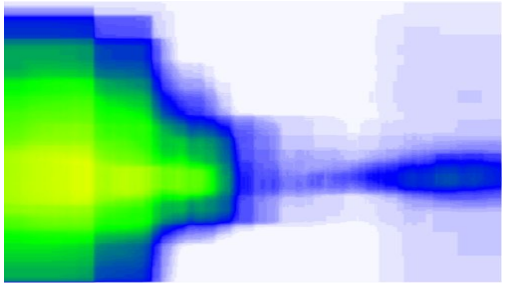
Car



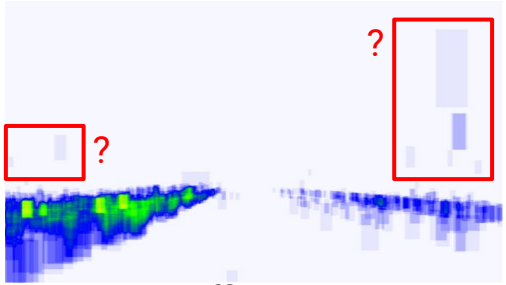
Person



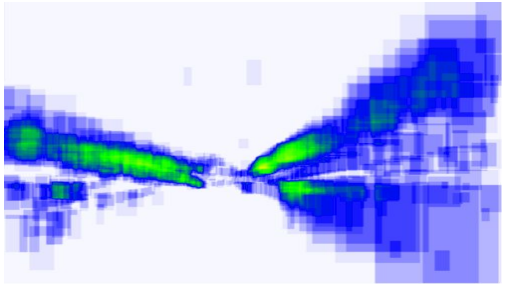
Traffic Light Green



Bus

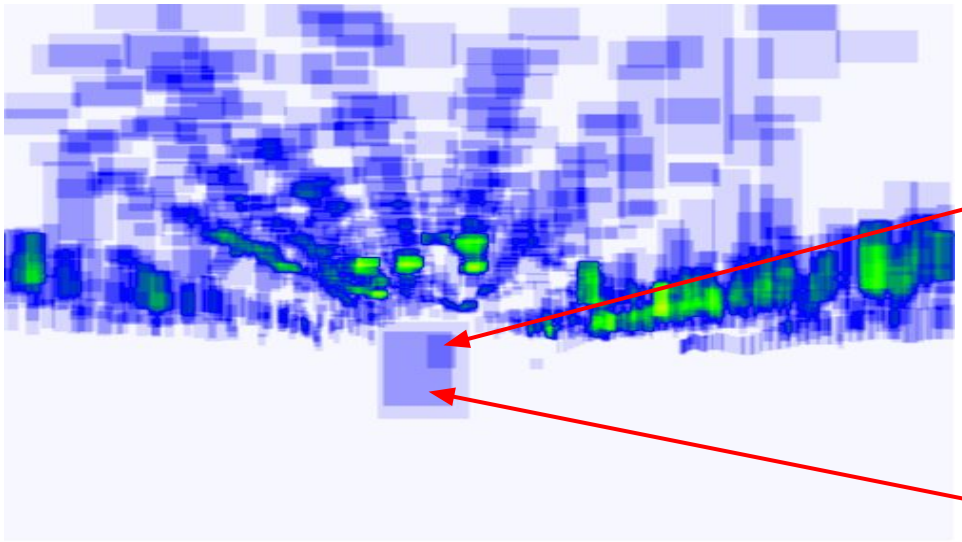


Traffic Cone



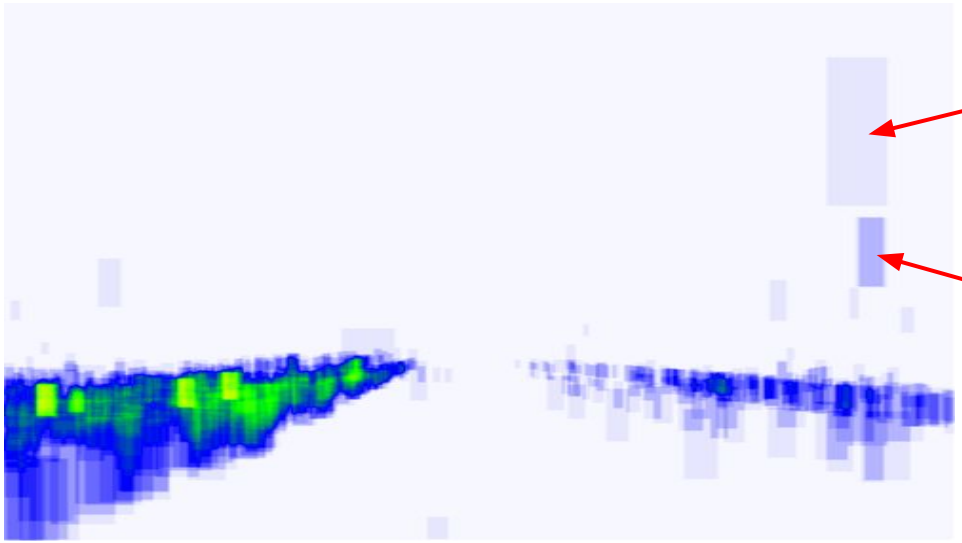
Temp Work-Zone Sign

Data Cleaning

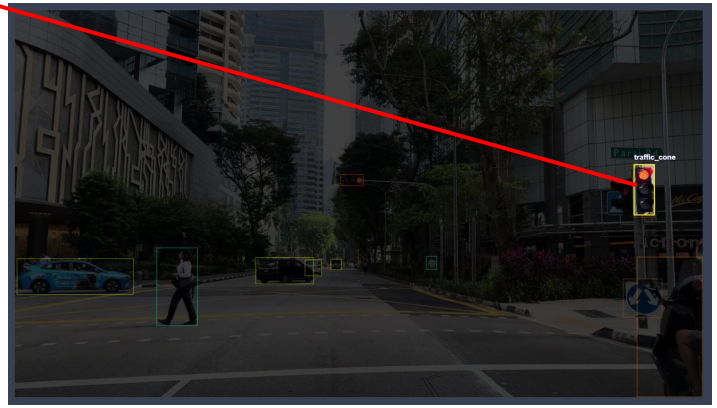


Traffic Light Green

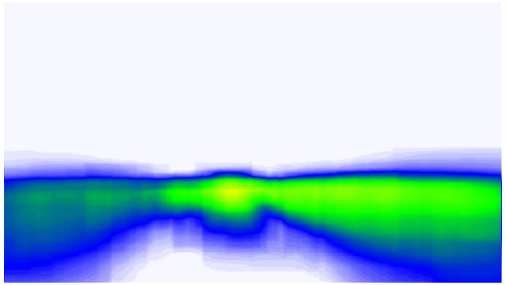
Data Cleaning



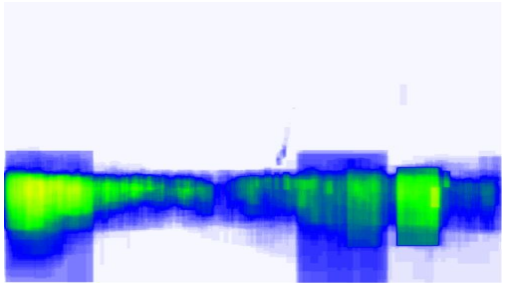
Traffic Cone



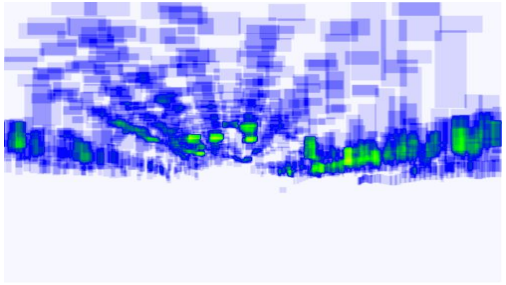
Dataset EDA. TOP 6 CLASSES HEATMAP



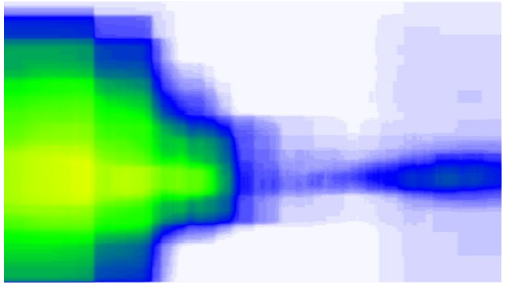
Car



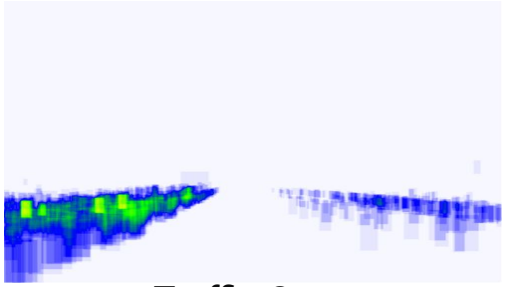
Person



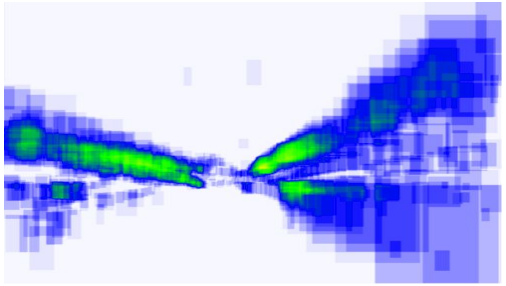
Traffic Light Green



Bus

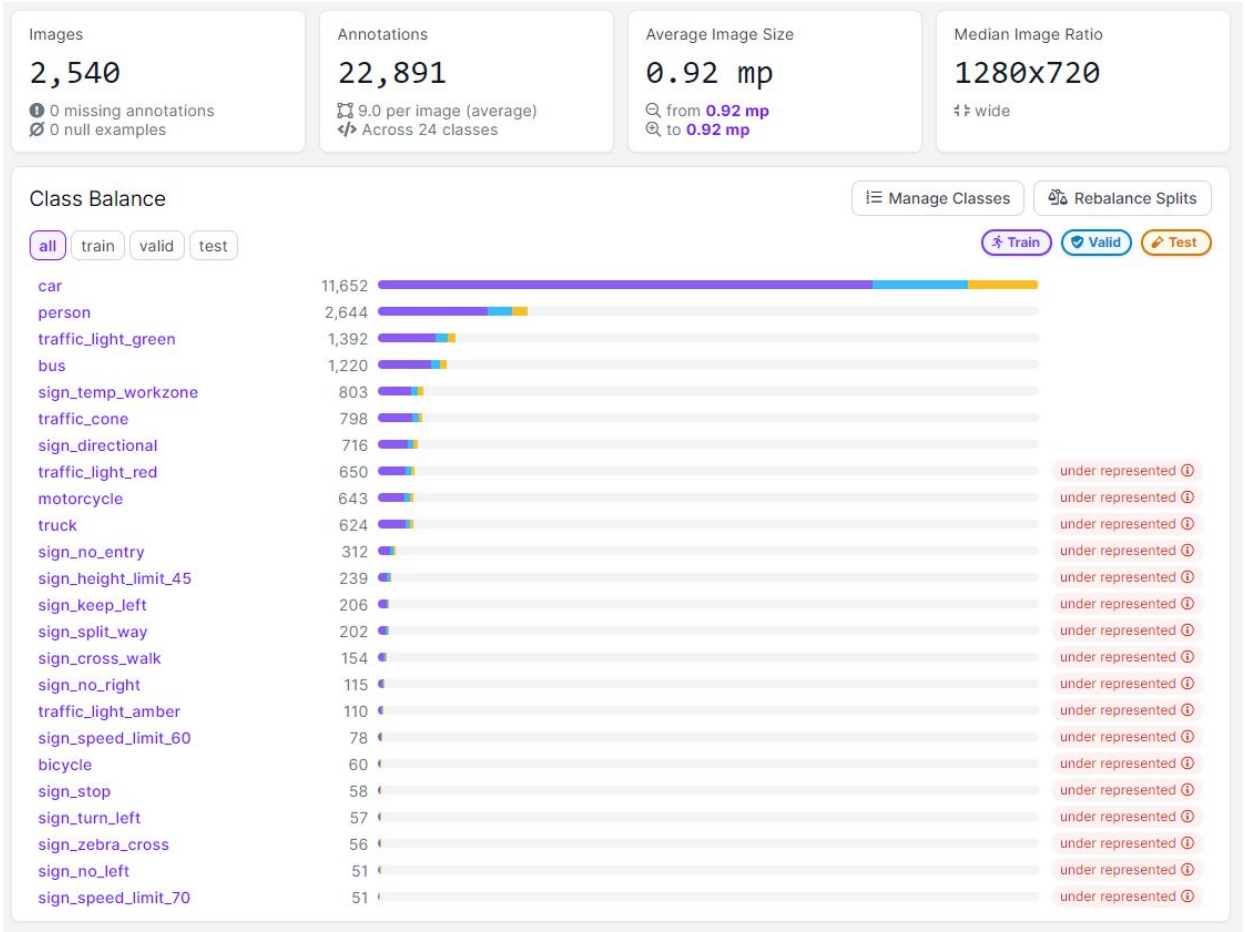


Traffic Cone

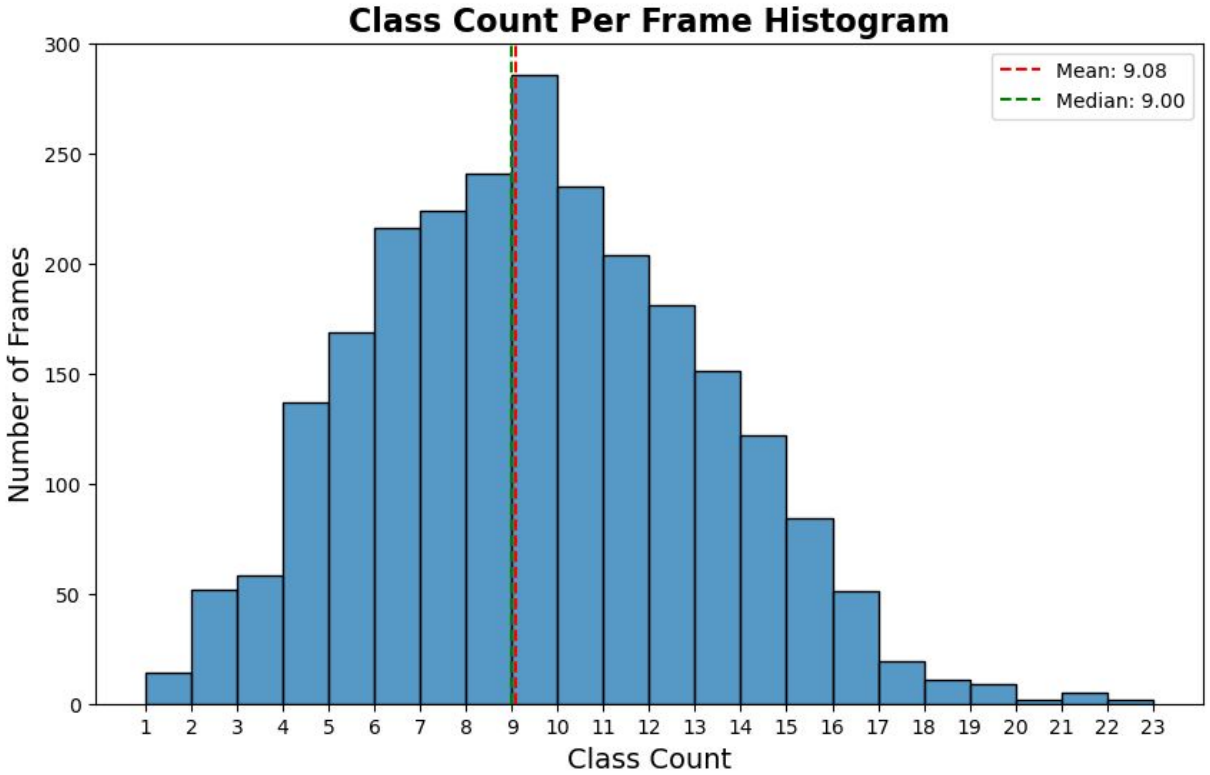


Temp Work-Zone Sign

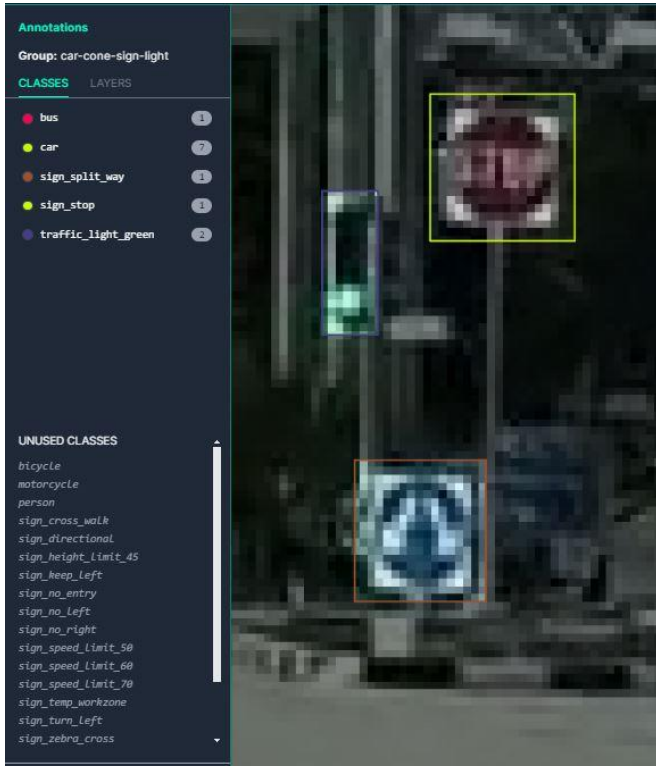
EDA - Final Class Distribution



Dataset EDA. Class Count Per Frame Histogram



EDA - Variance

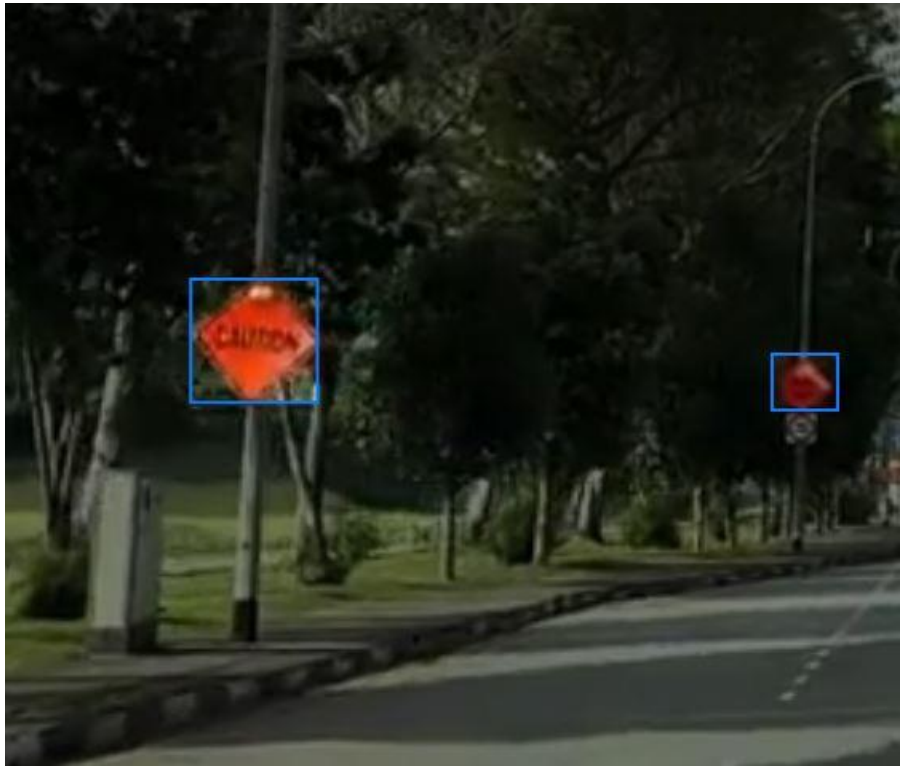


Example of traffic signs at various distances.

Images further away appearing more pixelated and those closer appearing sharper.

We can also see a slight difference in perspective.

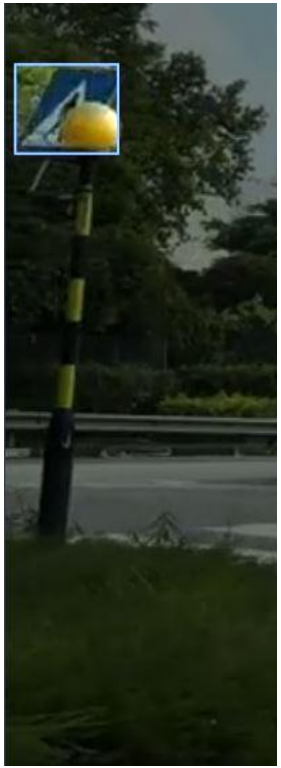
EDA - Variance

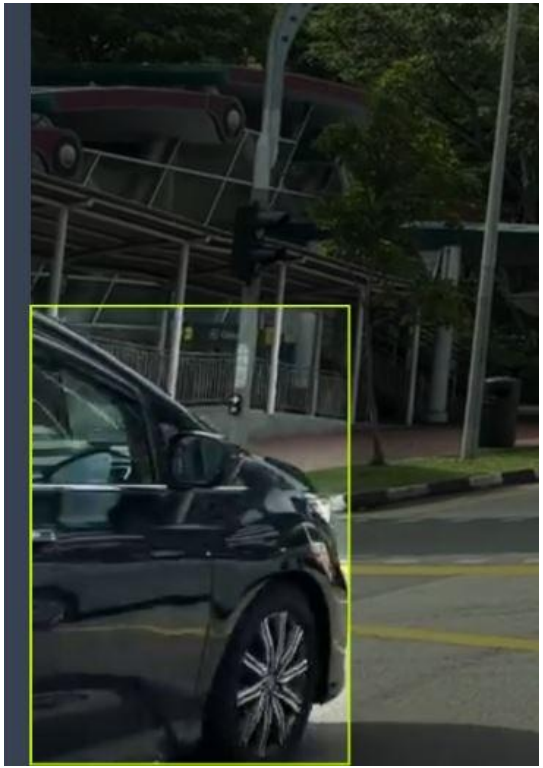


Example of traffic signs at various exposures and distances.

Left one is brighter and right one is darker.

EDA - Variance

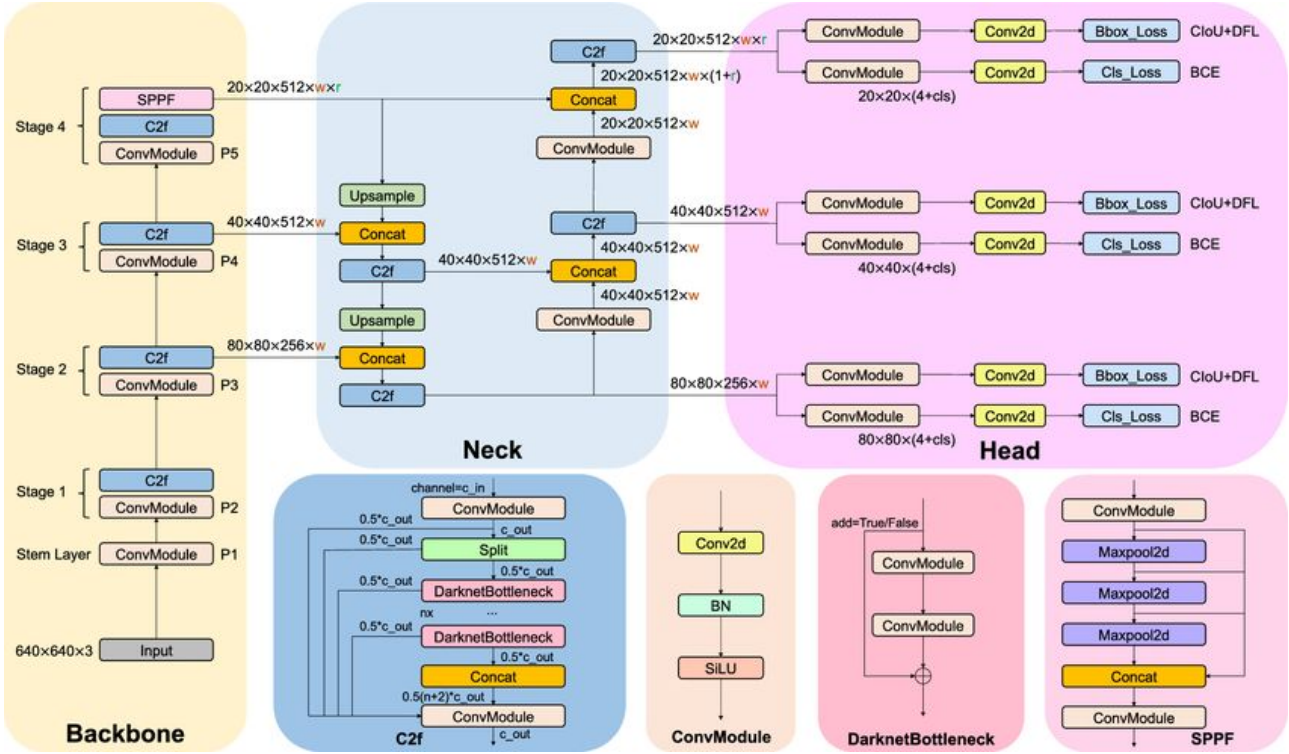




Example of covered or cropped objects.

Models

YOLOv8 Model



YOLOv8 Model

- YOLOv8 (You Only Look Once version 8) is an object detection model
- YOLOv8 Architecture:
    1. Backbone:
        - Extracts rich feature representations from the input image.
    2. Neck:
        - Aggregates and enhances features from different stages of the backbone.
    3. Head:
        - Classification Branch: Predicts the class probabilities for each object.
        - Localization Branch: Predicts the bounding box coordinates.

Why YOLOv8 Model?

1. Good accuracy while maintaining a small model size
2. Fast inference speed

Models Comparison

Hyperparameters

Model	Layers	Parameters	Batch Size	Image Size	Epochs	Optimizer	Momentum	Initial Learn Rate	Final Learn Rate	Weight Decay
Model 1	225	3.1 million	16	640	100	AdamW	0.937	0.000357	0.01	0
Model 2	225	3.1 million	16	736	350	AdamW	0.937	0.000357	0.01	0
Model 3	225	3.1 million	32	704	200	AdamW	0.95	0.001	0.001	0.0001
Model 4	225	3.1 million	32	704	200	SGD	0.95	0.01	0.01	0.0001
Model 5	295	25,9 million	32	736	150	SGD	0.95	0.01	0.01	0.0001

Metrics

Model	GPU	Price Per Hr	Run Time	Total Cost	Precision	Recall	mAP50	mAP50-95
Model 1	T4	$ 0.28	34.9 mins	$ 0.16	0.78	0.783	0.804	0.584
Model 2	T4	$ 0.28	112.5 mins	$ 0.53	0.826	0.849	0.868	0.645
Model 3	T4	$ 0.28	78.0 mins	$ 0.36	704	0.807	0.861	0.633
Model 4	T4	$ 0.28	76.6 mins	$ 0.36	0.871	0.783	0.857	0.639
Model 5	L4	$ 0.72	140.8 mins	$ 1.69	0.881	0.882	0.913	0.696

Models Comparison

Hyperparameters

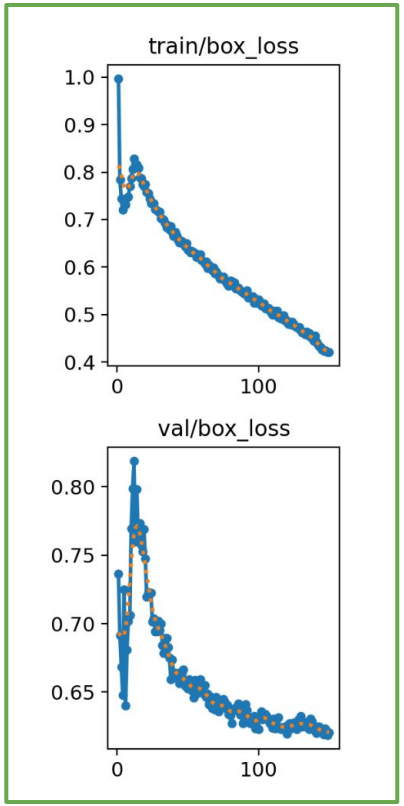
Model	Layers	Parameters	Batch Size	Image Size	Epochs	Optimizer	Momentum	Initial Learn Rate	Final Learn Rate	Weight Decay
Model 1	225	3.1 million	16	640	100	AdamW	0.937	0.000357	0.01	0
Model 2	225	3.1 million	16	736	350	AdamW	0.937	0.000357	0.01	0
Model 3	225	3.1 million	32	704	200	AdamW	0.95	0.001	0.001	0.0001
Model 4	225	3.1 million	32	704	200	SGD	0.95	0.01	0.01	0.0001
Model 5	**295**	**25,9 million**	**32**	**736**	**150**	**SGD**	**0.95**	**0.01**	**0.01**	**0.0001**

Metrics

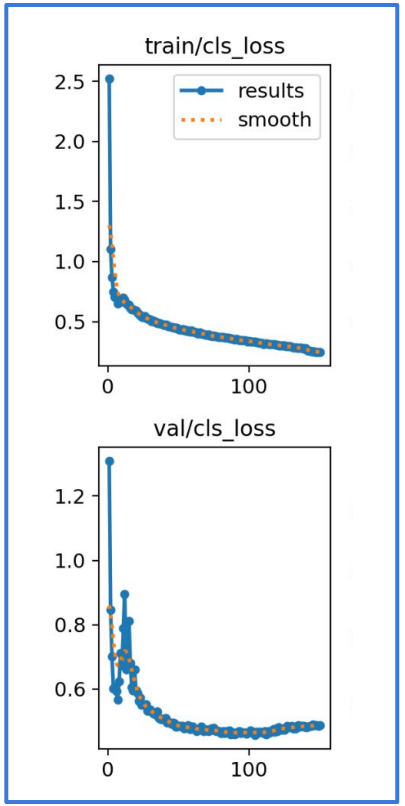
Model	GPU	Price Per Hr	Run Time	Total Cost	Precision	Recall	mAP50	mAP50-95
Model 1	T4	$ 0.28	34.9 mins	$ 0.16	0.78	0.783	0.804	0.584
Model 2	T4	$ 0.28	112.5 mins	$ 0.53	0.826	0.849	0.868	0.645
Model 3	T4	$ 0.28	78.0 mins	$ 0.36	704	0.807	0.861	0.633
Model 4	T4	$ 0.28	76.6 mins	$ 0.36	0.871	0.783	0.857	0.639
Model 5	L4	$ 0.72	140.8 mins	$ 1.69	**0.881**	**0.882**	**0.913**	**0.696**

Tuned Training Performance

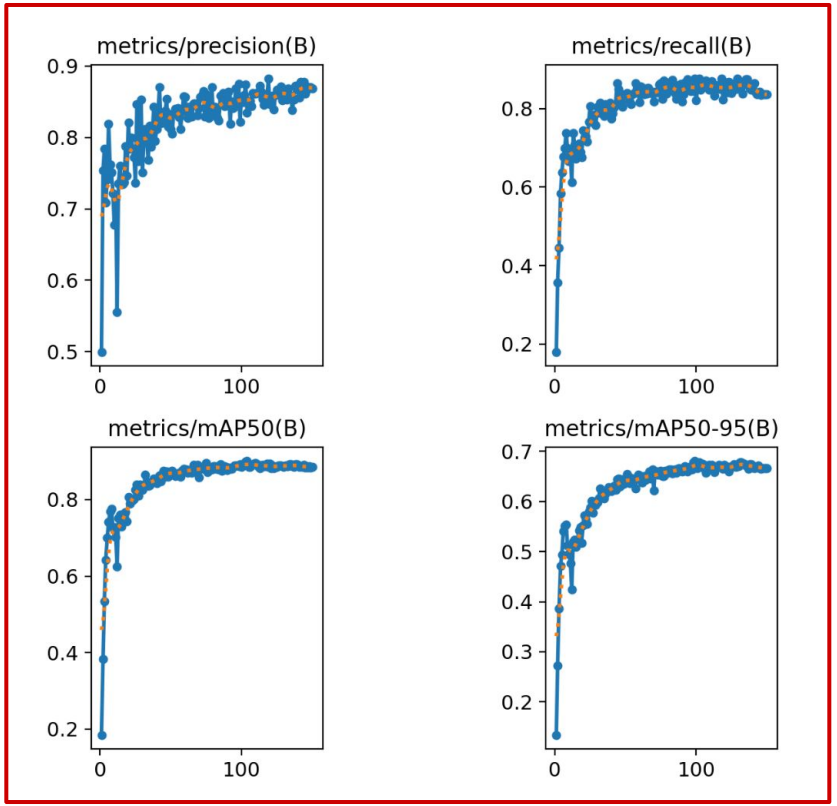
Error in bounding box prediction



Error in object class prediction



Metrics

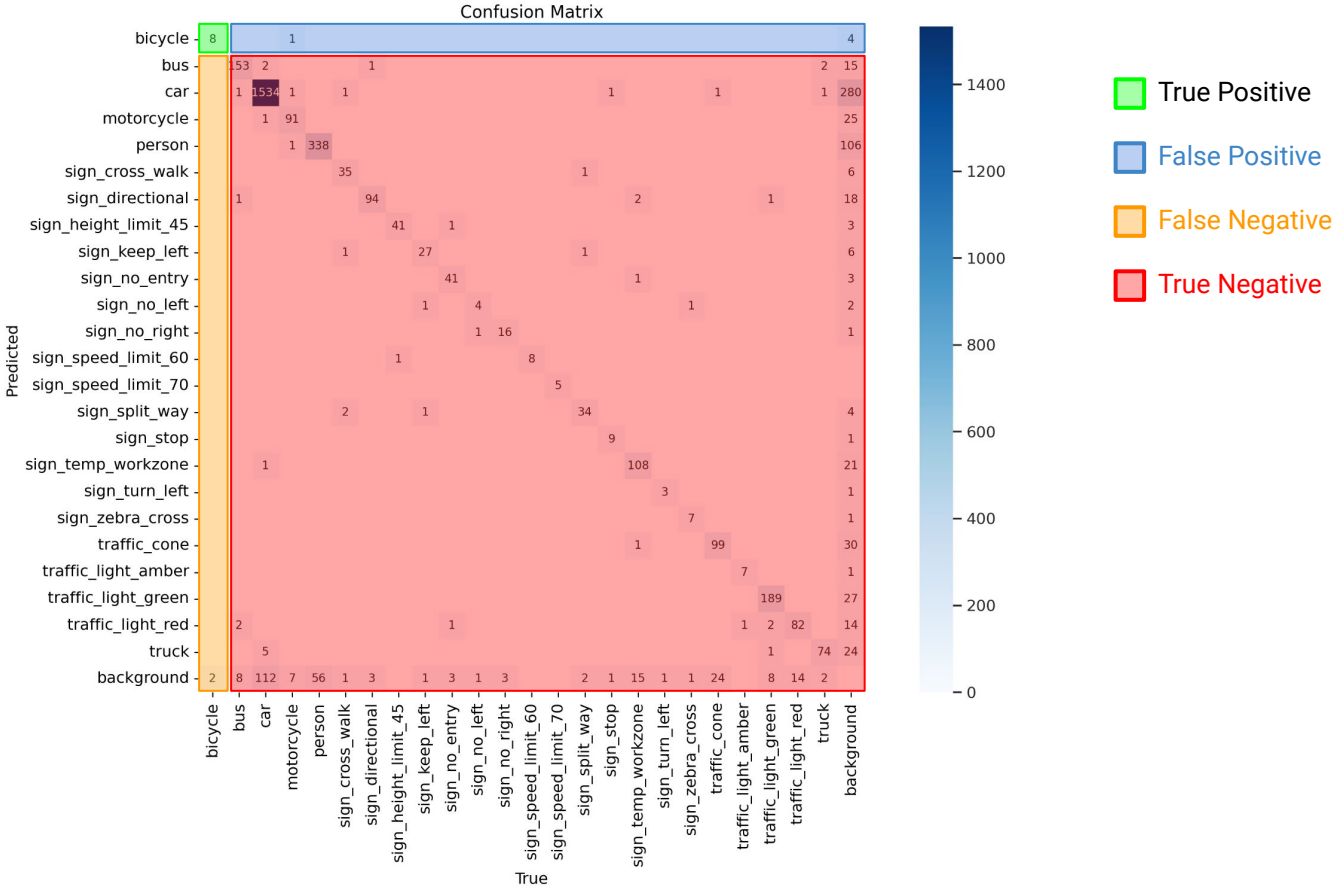


Tuned Validation Score

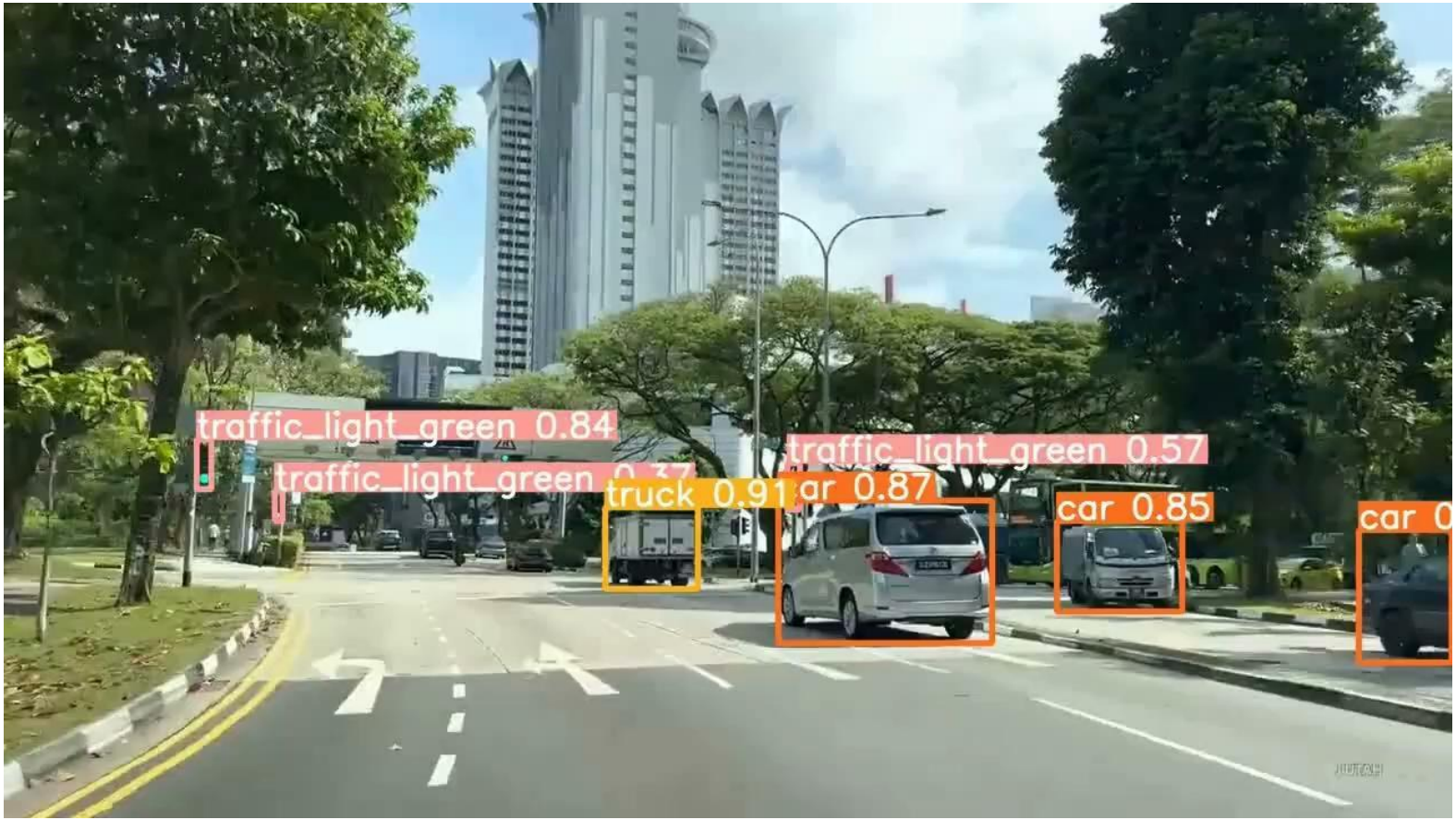
Class	Images	Instances	Box(P	R	mAP50	mAP50-95):
all	372	3316	0.822	0.876	0.892	0.687
bicycle	372	10	0.464	0.607	0.591	0.31
bus	372	165	0.888	0.91	0.945	0.879
car	372	1655	0.862	0.911	0.944	0.861
motorcycle	372	101	0.787	0.901	0.911	0.7
person	372	394	0.778	0.83	0.868	0.587
sign_cross_walk	372	40	0.841	0.875	0.88	0.677
sign_directional	372	98	0.856	0.959	0.948	0.81
sign_height_limit_45	372	42	0.927	0.976	0.973	0.8
sign_keep_left	372	30	0.794	0.9	0.867	0.657
sign_no_entry	372	46	0.908	0.855	0.958	0.608
sign_no_left	372	6	0.483	0.667	0.578	0.514
sign_no_right	372	19	0.853	0.842	0.831	0.642
sign_speed_limit_60	372	8	0.888	0.995	0.982	0.852
sign_speed_limit_70	372	5	0.953	1	0.995	0.868
sign_split_way	372	38	0.855	0.895	0.897	0.713
sign_stop	372	11	0.904	0.862	0.904	0.712
sign_temp_workzone	372	127	0.857	0.866	0.915	0.697
sign_turn_left	372	4	0.788	0.94	0.945	0.703
sign_zebra_cross	372	9	0.806	0.778	0.901	0.764
traffic_cone	372	124	0.793	0.774	0.851	0.497
traffic_light_amber	372	8	0.983	1	0.995	0.605
traffic_light_green	372	201	0.899	0.945	0.947	0.63
traffic_light_red	372	96	0.832	0.823	0.888	0.591
truck	372	79	0.725	0.924	0.89	0.8

Speed: 0.9ms preprocess, 13.4ms inference, 0.0ms loss, 20.5ms postprocess per image

Tuned Confusion Matrix

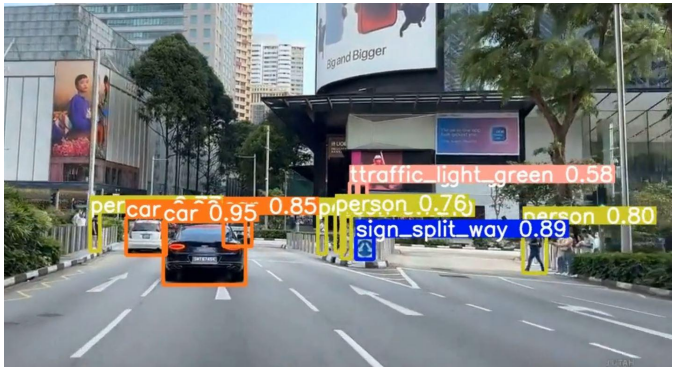


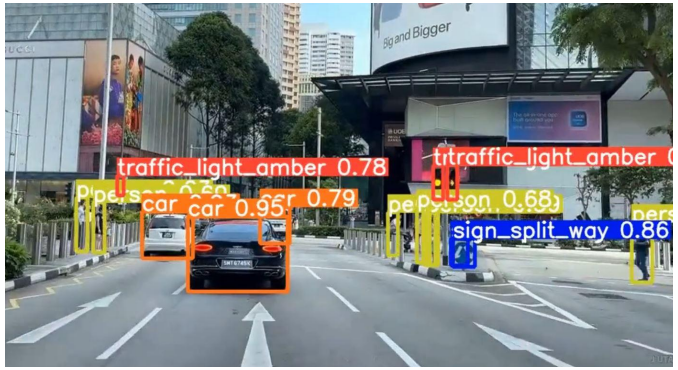
Demonstration of Pixel Pilot Kit Inference

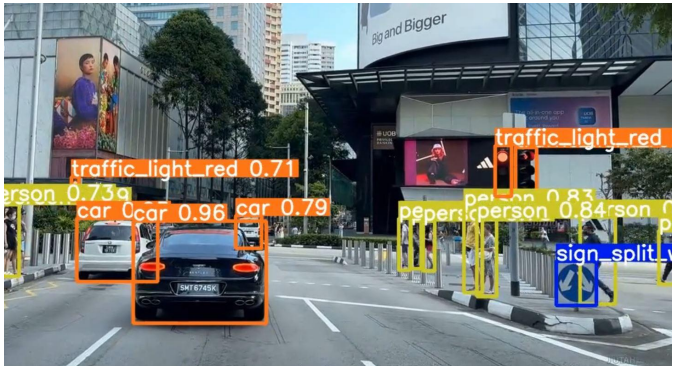


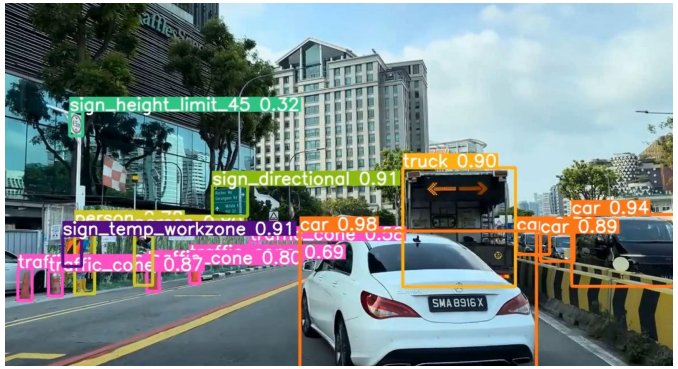
source

Demonstration of Pixel Pilot Kit Inference



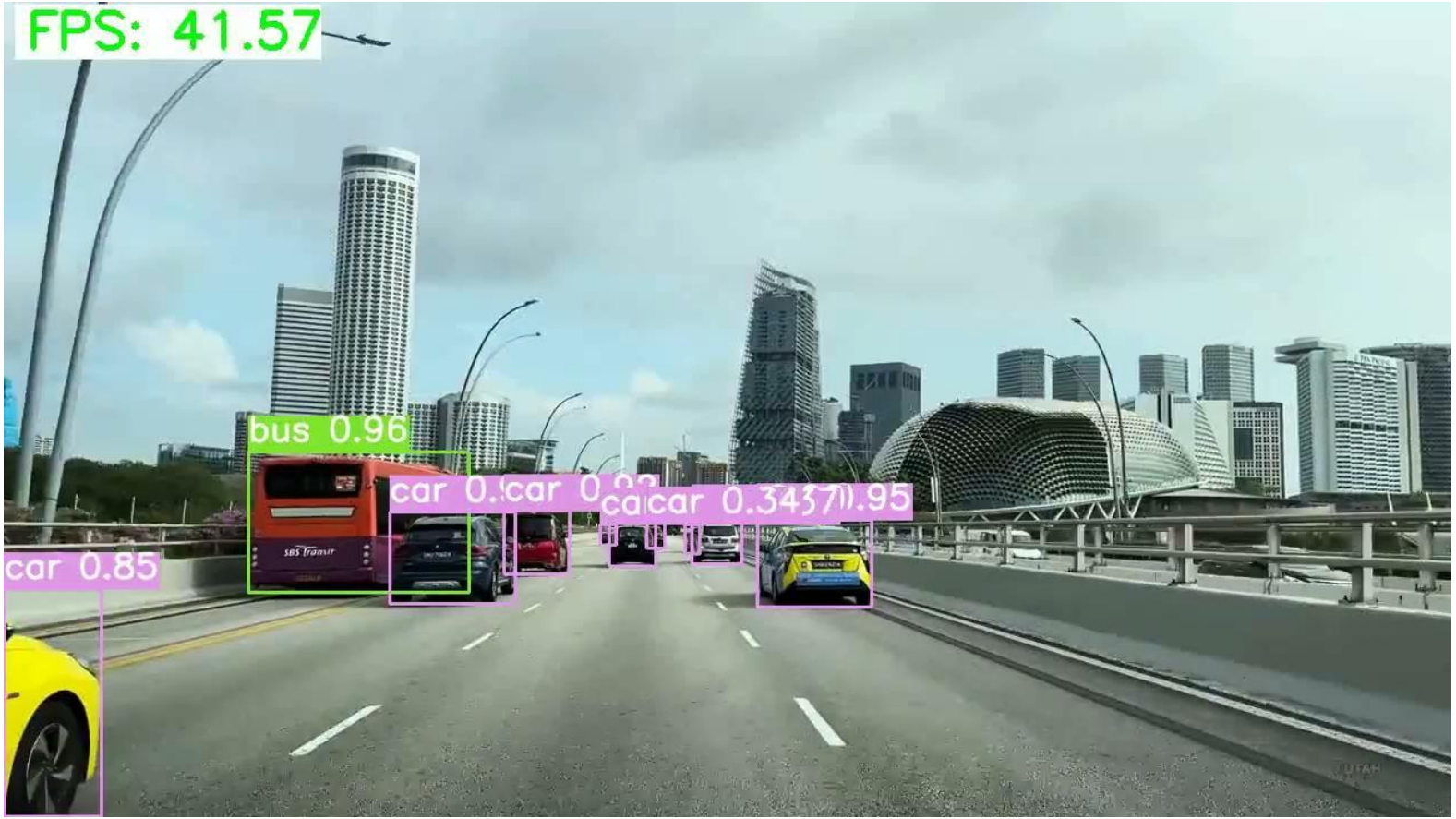






The model is able to detect changes in traffic lights, traffic cones, people, different types of vehicles, and road signs.

Demonstration of Pixel Pilot Kit Frame Rate



source

Conclusion

Problem Statement

How might we develop and integrate an advanced **object detection model** that can accurately identify **Singapore's local vehicles, traffic signs, and traffic-related objects** such as traffic cones, into existing car systems, enabling any vehicle to be equipped with reliable self-driving features?

Conclusion

To accurately identify various obstacles, such as pedestrians, other vehicles, road signs, and objects, to enhance safety and navigation in self-driving cars in Singapore, we can **leverage our trained model** that specializes in **detecting Singapore vehicles, road obstacles, traffic signs and lights**.

Conclusion

Instead buying new self driving car, Janet can now **enhance her car** by installing our **Pixel Pilot self-driving kit** for **SGD 6,000**.

Jason can now **replace** his original car's self-driving system with our **Pixel Pilot self-driving system**.

Limitations & Recommendations

Initial Limitations

Limitation

- Initially I tried to train YOLOv8 with just **new classes data**. I **froze all the layers** and **added an extra head** into pretrained YOLOv8 architecture. This involved **adjustments to configuration and engine files**. However, it was unsuccessful, the model forgot about previously learned classes and only detects new classes. **Update:** Solved: annex_model_exploration.ipynb
- **YOLOv8 zero-shot** is good, it can detect vehicles, traffic cones and traffic lights. However it has **difficulties differentiate traffic lights red, green, amber and also traffic signs.**

Limitations and Recommendations

Limitation

- **Imbalanced data** might cause the model to **miss** the **minority classes** or predict with much **lower confidence** score, leading to **lower recall and precision** for those classes.

Recommendation

- **Adjust loss weight** which can help to address the issue of imbalanced data.
- **Gather additional data** for **under-represented classes**.

Limitations and Recommendations

Limitation

- The existing dataset contains only **daytime data** and has l**ess than 20 traffic signs.**

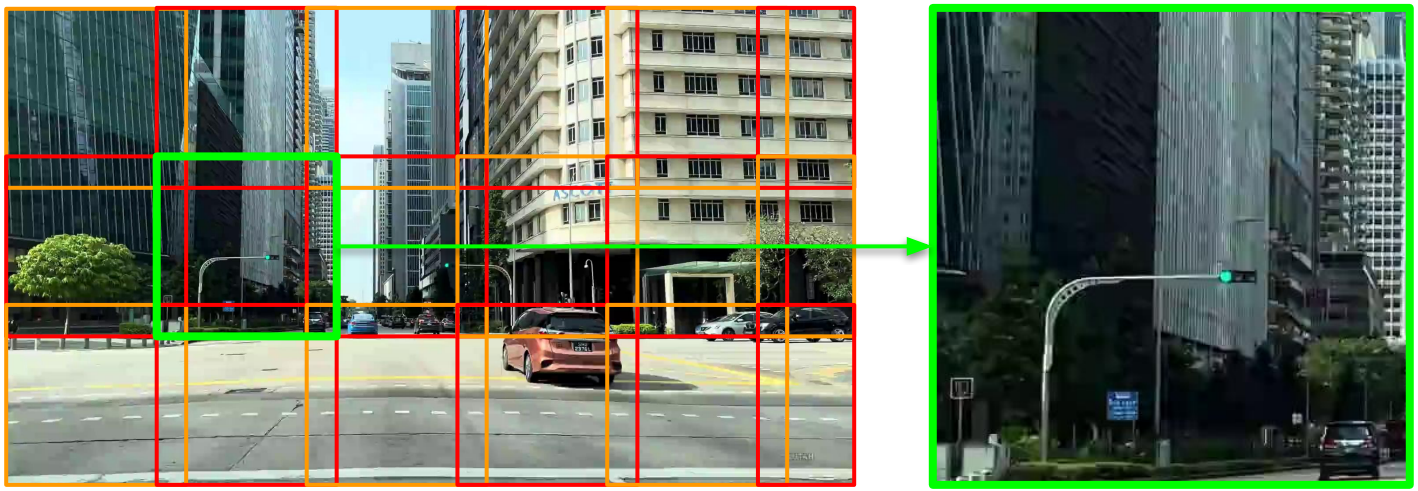
Recommendation

- **Expand the dataset**, there are other signs and objects that are not currently included.

  We also need to include **vehicle brake lights** so we can anticipate them.
- Collect **night time data** and use **data augmentation** to build a model that's **robust** across **different times** of day and **weather conditions**.

Other Recommendations

- Run **gridsearch** to get best parameters.
- Use **tiling windows** to **detect further or small objects**..



It’s slower but more effective at detecting small objects

- Use **TensorRT** to **improve inference time/ frame rate** by **4 to 5 times**.
- Use **YOLOv8 ByteTrack** to track objects and anticipate future movements.

Q & A