

Pedestrian Detection

Suraj Kunthu, Dominic Scerbo, Abigail Snyder

DS6050

Final Project

Summary

- Over 7,500 pedestrians were hit and killed by drivers in 2022
- The goal of this project was to develop an efficient and accurate pedestrian detection model using deep learning techniques
- The hope is that such a model could be applied to a full pedestrian system with wide application—from additional safety warnings in current-model vehicles, to the future development of autonomous vehicles, and even surveillance systems—with the goal of improving pedestrian safety and making a significant reduction in the number of pedestrian deaths in the U.S.
- Specifically, the project focused on training an object detection model on the EuroCity Persons and Caltech Pedestrian datasets.
- The methods used in this project were You Only Look Once (YOLO) and Detection with Transformers (DETR) via open-source pretrained frameworks Ultralytics/YOLOv8 and Meta/DETR

Historical & Current Approaches

- Historically, pedestrian detection primarily relied on hand-crafted features and traditional machine learning algorithms.
- Currently, the U.S. Department of Transportation's Exploratory Advanced Research (EAR) Program is engaged in a project that attempts to integrate two- and three-dimensional technology into pedestrian detection.



Key Deep Learning Architectures

- Convolutional Neural Networks (CNNs)
- Single Shot Multibox Detector (SSD)
- You Only Look Once (YOLO)
- Vision Transformers (ViT)



Challenges & Innovations

- Occlusion Handling
 - Dealing with partial occlusions remains a challenge in pedestrian detection.
 - Multi-stage models and advanced post-processing techniques have been proposed to improve occlusion handling.
- Anomaly Detection
 - Detecting abnormal pedestrian behavior, such as jaywalking, has gained attention for enhancing safety.
 - Deep learning-based anomaly detection methods can identify unusual patterns in pedestrian movement.
- Real-time Processing
 - Achieving real-time processing and strict accuracy requirements for pedestrian detection in video streams is essential for applications like autonomous driving.
 - Efficient architectures like YOLO have made real-time detection feasible



Datasets (Caltech Pedestrian and EuroCity)

TABLE 1
Comparison of person detection benchmarks in vehicle context

	Caltech [2]	KITTI [3]	CityPersons [4]	TDC [5]	EuroCity Persons
# countries	1	1	3	1	12
# cities	1	1	27	1	31
# seasons	1	1	3	1	4
# images (day / night)	249884 / -	14999 / -	5000 / -	14674 / -	40217 / 7118
# pedestrians (day / night)	289395 ^a / -	~9400 ^b / -	31514 / -	8919 / -	183004 / 35309
# riders (day / night)	- / -	~3300 ^b / -	3502 / -	23442 / -	18216 / 1564
# ignore regions (day / night)	57226 ^a / -	~22600 ^b / -	13172 / -	- / -	75673 / 20032
# orientations (day / night)	- / -	~12700 ^b / -	- / -	- / -	176879 / 34393
resolution	640 × 480	1240 × 376	2048 × 1024	2048 × 1024	1920 × 1024
weather	dry	dry	dry	dry	dry, wet
train-val-test split (%)	50-0-50	50-0-50	60-10-30	71-8-21	60-10-30

Caltech Pedestrian

- Tasks:

- Convert video to images
 - ~10 hours of 640x480 30Hz video
 - ~250,000 frames (in 137 approximately minute long segments)
- Convert labels format to respective formats (YOLO custom and MS COCO)

- Challenges:

- Lack of biomes diversity
 - Season
 - Time of Day
 - Region
 - Weather
- Lack of pedestrian variation



EuroCity Persons

- **Tasks:**
 - Convert labels format to respective formats (YOLO custom and MS COCO)
 - Select label ontology
 - Sample data from categories appropriately
- **Challenges:**
 - Computation time due to large volume of data

Aggregated Label Classes

Object Class	# objects (day)	# objects (night)	# objects (sum)
Pedestrian	183004	35309	218313
Rider	18216	1564	19780

Rider Label Classes

Object Class	# objects (day)	# objects (night)	# objects (sum)
Bicycle	9666	614	10280
Motorbike	2196	229	2425
Scooter	5748	683	6431
Tricycle	94	5	99
Wheelchair	125	4	129
Buggy	322	26	348
Co-Rider	671	137	808

All Label Classes

Object Class	# objects (day)	# objects (night)	# objects (sum)
Pedestrian	73449	19857	93306
Rider+Vehicle	2224	175	2399
Bicycle	21446	1779	23225
Motorbike	4817	318	5135
Scooter	12438	748	13186
Tricycle	89	10	99
Buggy	1141	109	1250
Wheelchair	16	4	20

Data Preprocessing

Major Task: Generate YOLO TXT Schema with appropriate bound box format from raw data format

Minor Task: Convert YOLO TXT Schema to MS COCO JSON Schema

MS COCO JSON Schema

"categories":

"id": int,

"name": "person",

"supercategory": "pedestrian", ...

"image":

"id": int,

"width": 640,

"height": 640,

"file_name": "train/img-name.jpg", ...

"annotation":

...,

"area": float,

"iscrowd": <0/1>,

"Image_id": int,

"bbox": [x,y,width,height],

"category_id": int,

"id": int

YOLO TXT Schema w/ dataset.yaml

<img-id>.txt

<object-class><x><y><width><height>

dataset.yaml

path: / dataset/root/dir

train: images/train

val: images/val

test: images/val (optional)

Names:

<class-name>

Demos

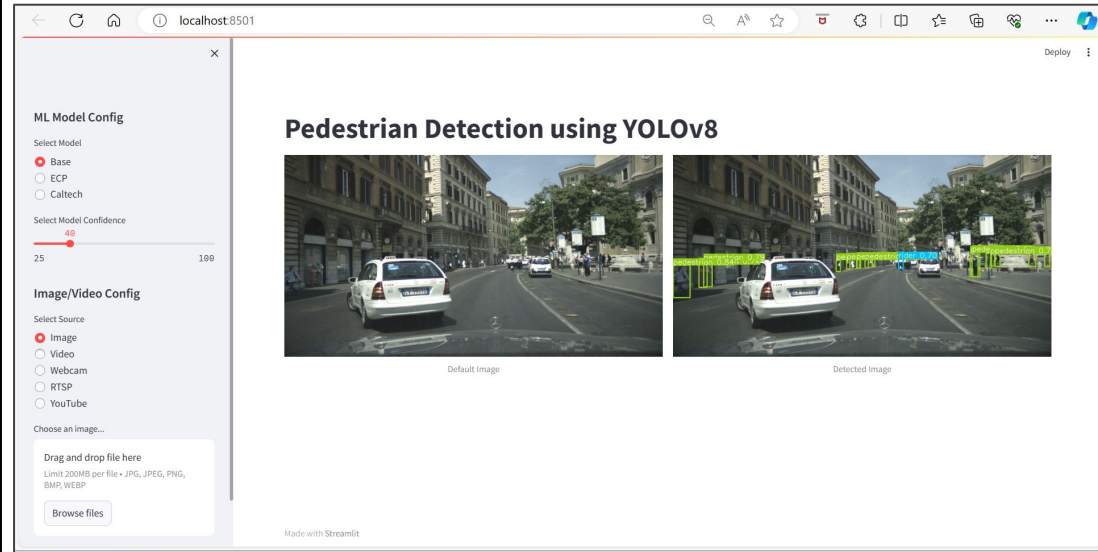
- Example 1) Pretrained YOLOv8n Model
- Example 2) Caltech Fine-Tuned YOLOv8n Model
- Example 3) ECP Fine-Tuned YOLOv8n Model
- Example 3) Pretrained detr-r50 Model
- Example 5) Caltech Fine-Tuned DETR Model
- Example 6) ECP Fine-Tuned DETR Model

*Bottom Line Up Front

- Fine-tuning YOLOv8n with the ECP data performed best overall (mAP: 0.59):
 - Reduced the presence of false negatives
 - Detects multiple classes of pedestrians other than Person, but at a low confidence
 - Improves detection of Pedestrians in diverse weather conditions, depth of field, and position/angle
 - Maintained sufficient inference time


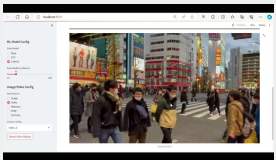

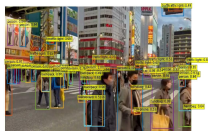


About the App

- Open-source Streamlit-based GitHub project
- Customizable confidence threshold
- Supports tracking / object detection via: Images, Video, Webcam, Real-Time Stream Pool, Youtube



**A GPU was NOT used to generate the demos*

Demos

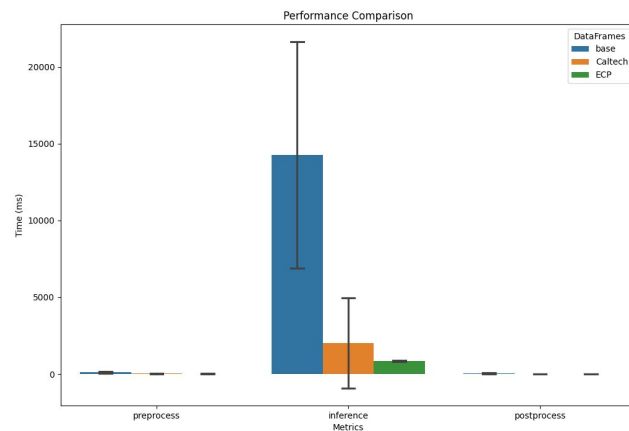
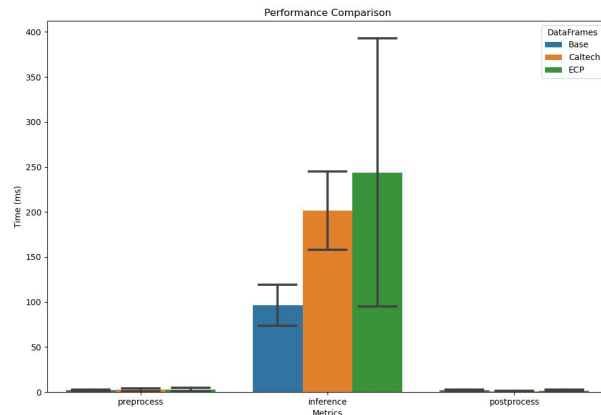
Ex: 1	Ex: 2	Ex: 3	Ex: 4	Ex: 5	Ex: 6
YOLOv8n	YOLOv8n	YOLOv8n	detr-r50	detr-r50	detr-r50
COCO	COCO + Caltech	COCO + ECP	COCO	COCO + Caltech	COCO + ECP
CPU only	CPU only	CPU only	NVIDIA a100 40G	NVIDIA a100 40G	NVIDIA a100 40G
-	mAP: 0.411	mAP: 0.459	-	mAP: 0.468	mAP: ~0.4
					

**This video was not included in the training or validation dataset*















Inference Performance Comparison

YOLOv8: Following the execution of inference experiments, it has been observed that the process of fine-tuning the model exhibits an augmentation in the inference time. However, it is noteworthy that this augmentation does not impede the model's capacity to execute inference tasks within applications, provided that the requisite hardware devices are employed.

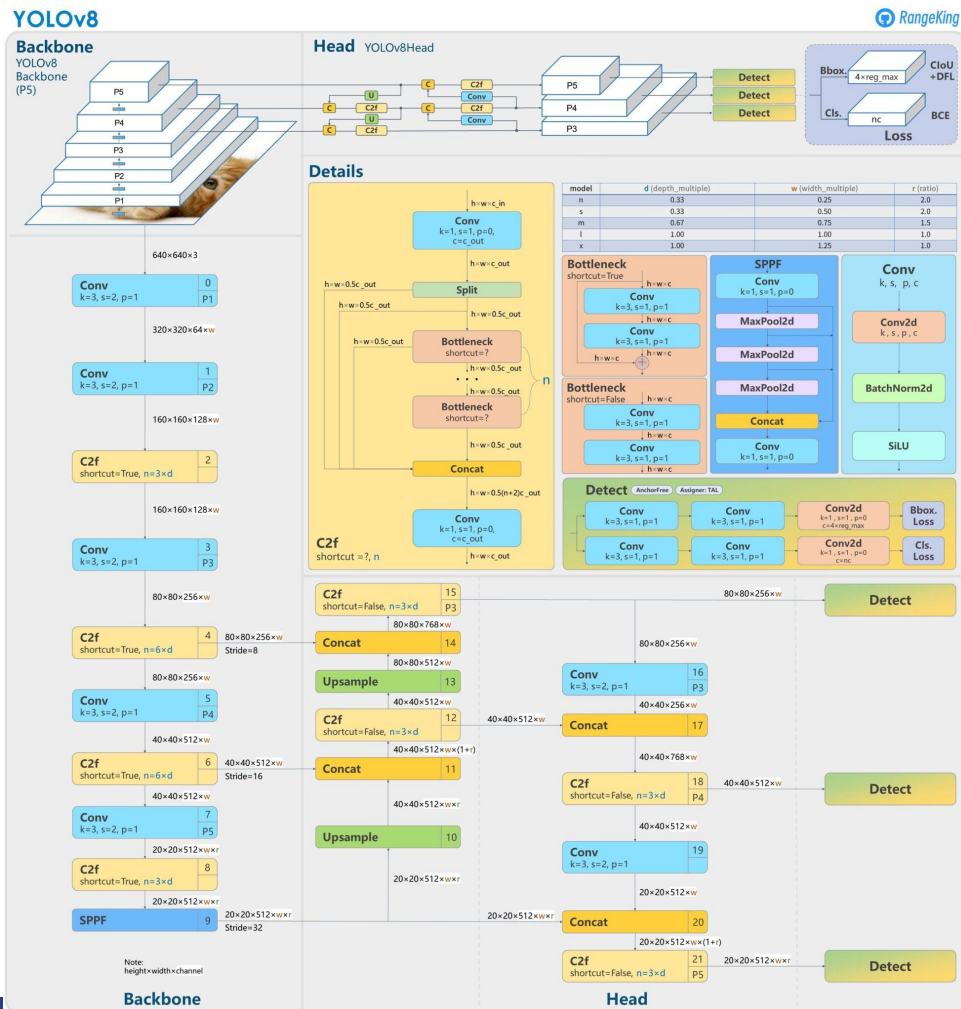
DETR: Following the execution of inference experiments, it has been observed that the process of fine-tuning the model exhibits a decrease in inference time compared to the base model, but a higher inference time compared to the YOLOv8 model framework while leveraging 1 NVIDIA a100.



Model Comparison Quick Sheet

	YOLO Caltech	YOLO ECP	DETR Caltech	DETR ECP
Training Cost	 Inexpensive	 Moderate	 Moderate	 Expensive
OS interoperability	 Windows/Linux		 Best suited w/ Linux	
Test Accuracy	 Highest FNR / Low FPR	 Low FNR / Low FPR	 High FNR / Low FPR	 Low FNR / High FPR
Data Quantity	 Requires large amounts of data		 Requires less amounts of data	
Speed	 Low latency w/ limited compute		 High latency w/o sufficient compute	

YOLOv8 *Nano* is the fastest and smallest, while YOLOv8 *Extra Large* (YOLOv8x) is the most accurate yet the slowest among them.

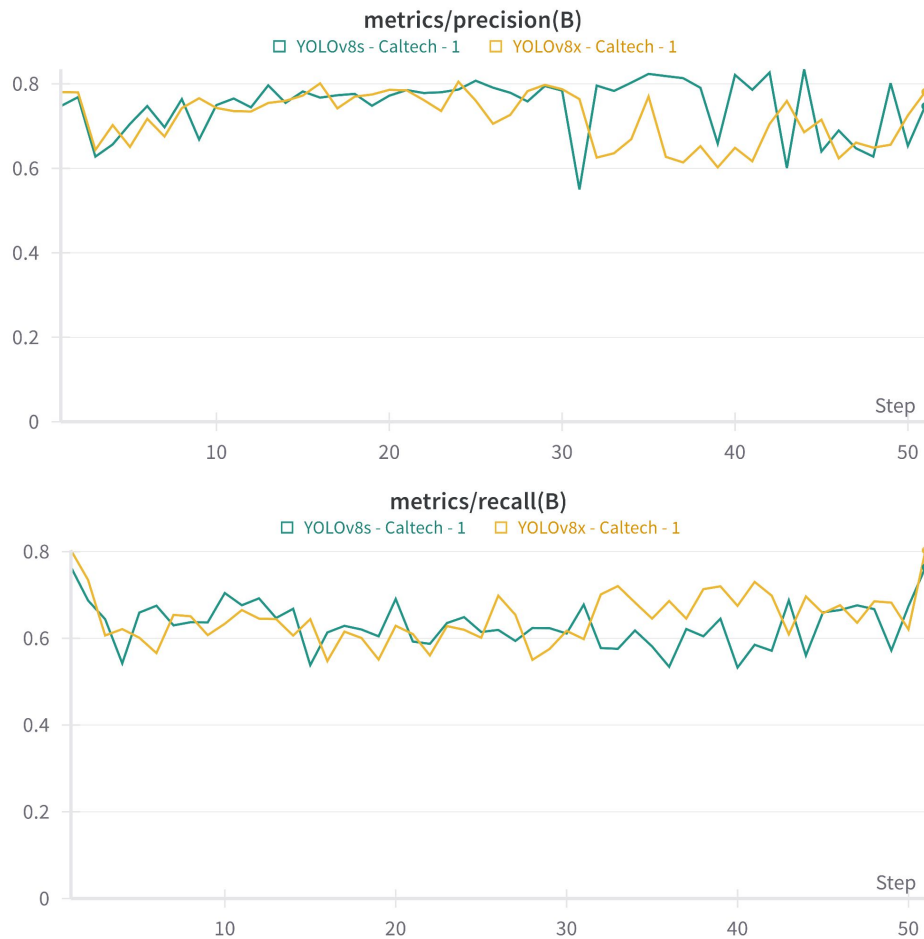


YOLOv8 Model:

Caltech Pedestrian Data

On the Caltech Pedestrian dataset, we trained both the YOLOv8x and YOLOv8s models. Over 50 epochs, on the training dataset, the more complex model (YOLOv8x) only showed slight advantages over the simpler model.

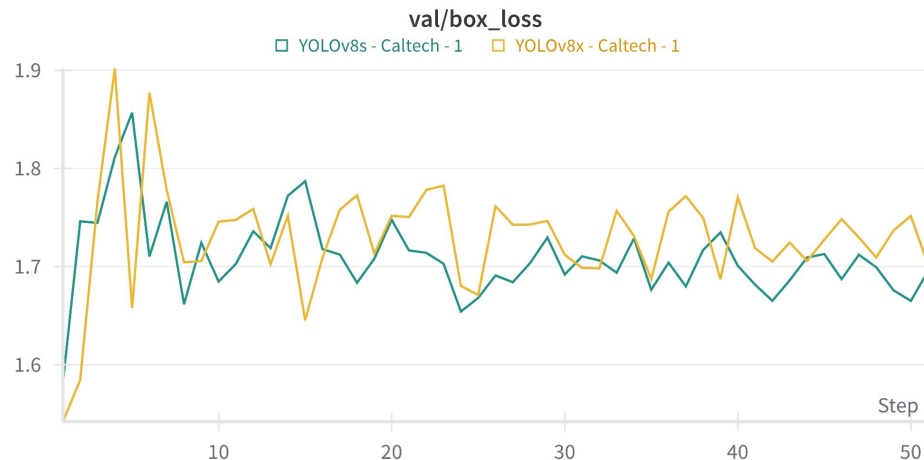
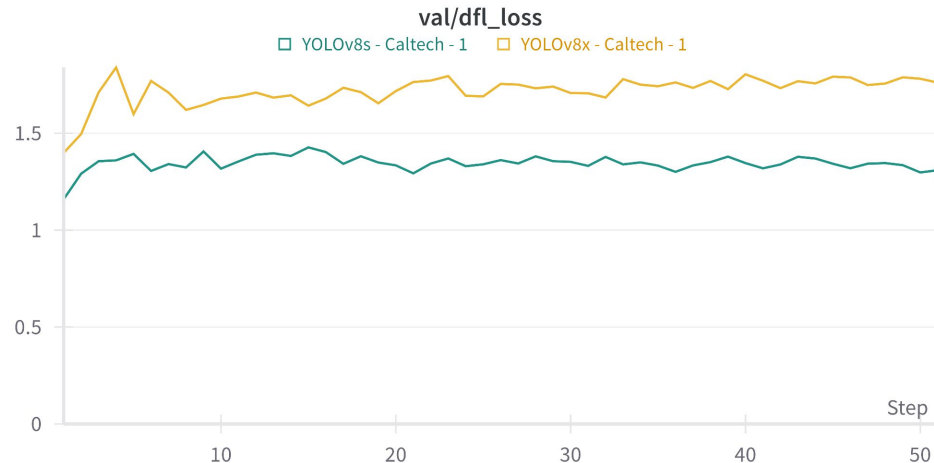
- **Recall: 0.8**
- **Precision: 0.78**



YOLOv8 Model:

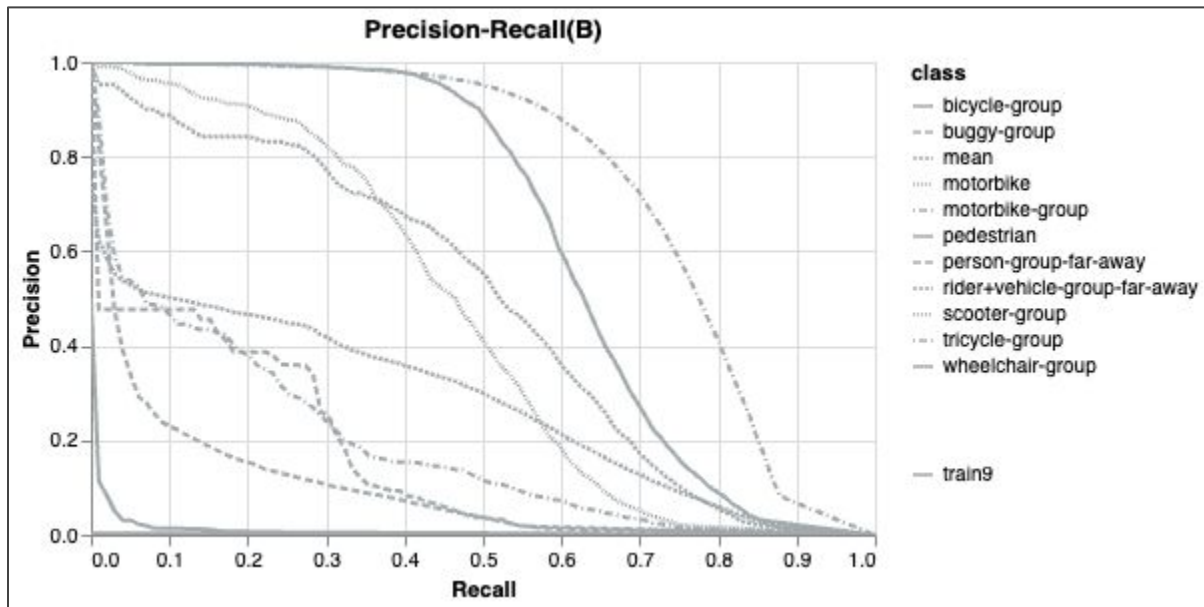
Caltech Pedestrian Data

On the validation set, the more complex model (YOLOv8x) had consistently higher distributional focal loss, though the box loss of the two models converged at approximately 1.7 towards the end of training.



YOLOv8 Model: Euro City Persons Data

Because of the more complex nature of the Euro City Persons dataset, with additional classes beyond “person” / “not-person,” training the model on this dataset gave more insight than was available with the Caltech Pedestrian dataset.



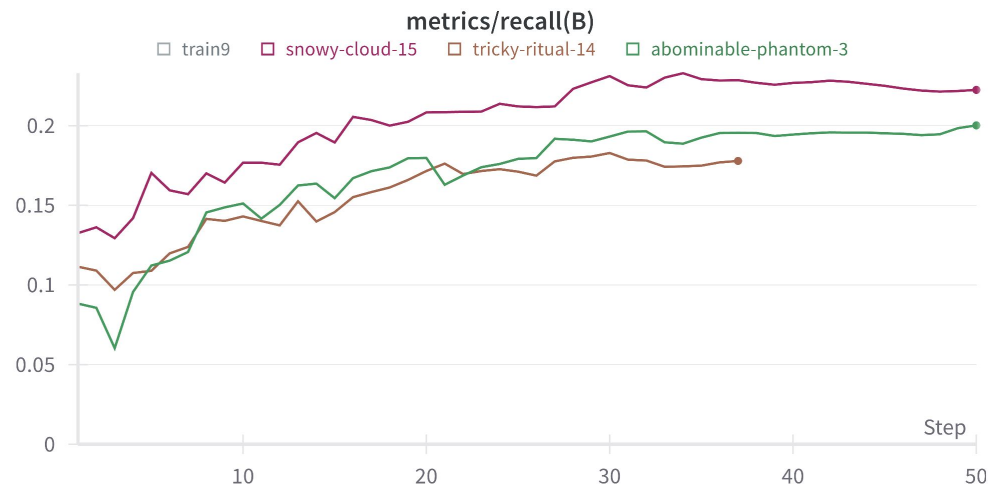
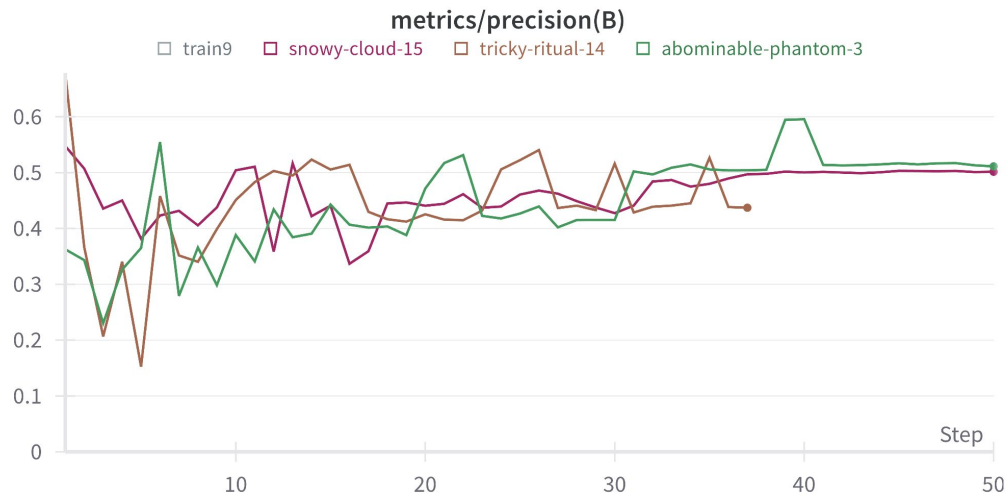
As this Precision-Recall curve indicates, over several iterations of training the YOLOv8 models on the ECP data, the precision and recall still vary greatly by class.

YOLOv8 Model:

Euro City Persons Data

The recall and precision on the ECP dataset was significantly less than that on the Cal Tech dataset, due to the more complex and diversified nature of the images in the dataset and inability to detect class objects with less samples.

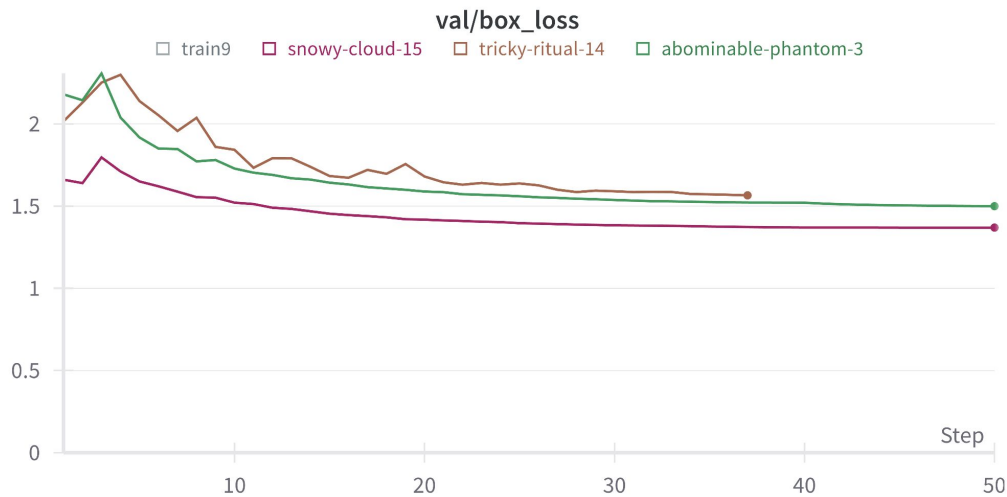
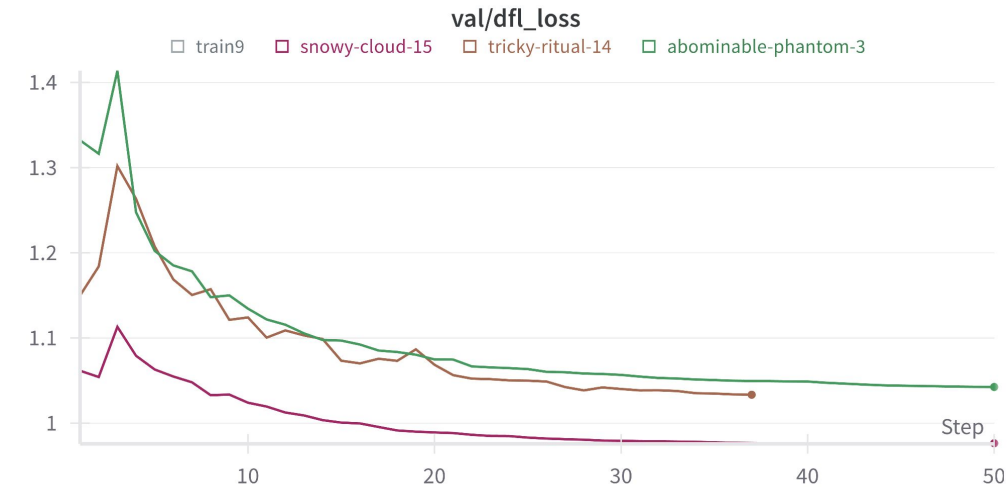
- **Recall: 0.2**
- **Precision: 0.5**



YOLOv8 Model:

Euro City Persons Data

Over the course of ~50 epochs, the models did show considerable improvement in **loss**, with most iterations converging near a box loss of 1.3 and a distributional loss of 0.97



YOLOv8 Model:

Euro City Persons Data

The model allows us to take a video or photo such as these and identify the various objects in it.

We were able to take this model and create an app for pedestrian detection.



YOLOv8 Model:

ECP mAP Breakdown

- Downside of datasets with multi-class labels (ECP) identified poor performance on rarer classes and diminished the mAP.
- Further annotation analysis should be conducted to remove/aggregate annotations in a way that does not decrease performance on the Pedestrian class.
- This will also impact DETR.

Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	5036	44769	0.501	0.223	0.23	0.133
buggy-group	5036	157	0.366	0.185	0.153	0.0749
motorbike	5036	2	1	0	0	0
person-group-far-away	5036	11041	0.413	0.0386	0.095	0.0363
motorbike-group	5036	640	0.369	0.206	0.184	0.0974
wheelchair-group	5036	7	0	0	0	0
tricycle-group	5036	9	1	0	0.00122	0.000905
pedestrian	5036	26236	0.711	0.705	0.75	0.459
rider+vehicle-group-far-away	5036	288	0.403	0.00347	0.00993	0.0041
scooter-group	5036	1616	0.54	0.508	0.481	0.272
bicycle-group	5036	2592	0.515	0.448	0.448	0.241
bicycle	5036	8	0	0	0	0
rider	5036	2173	0.699	0.577	0.635	0.412

DETR Model AP/AR Comparison

Caltech

IoU metric: bbox					
Average Precision	(AP) @[IoU=0.50:0.95	area= all	maxDets=100] = 0.468
Average Precision	(AP) @[IoU=0.50	area= all	maxDets=100] = 0.826
Average Precision	(AP) @[IoU=0.75	area= all	maxDets=100] = 0.467
Average Precision	(AP) @[IoU=0.50:0.95	area= small	maxDets=100] = 0.001
Average Precision	(AP) @[IoU=0.50:0.95	area=medium	maxDets=100] = 0.454
Average Precision	(AP) @[IoU=0.50:0.95	area= large	maxDets=100] = 0.594
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets= 1] = 0.399
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets= 10] = 0.584
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets=100] = 0.627
Average Recall	(AR) @[IoU=0.50:0.95	area= small	maxDets=100] = 0.175
Average Recall	(AR) @[IoU=0.50:0.95	area=medium	maxDets=100] = 0.623
Average Recall	(AR) @[IoU=0.50:0.95	area= large	maxDets=100] = 0.680

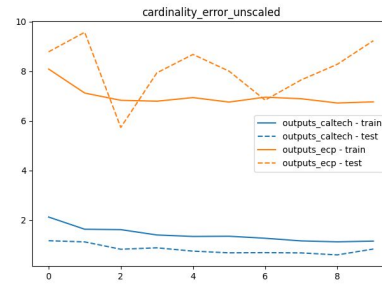
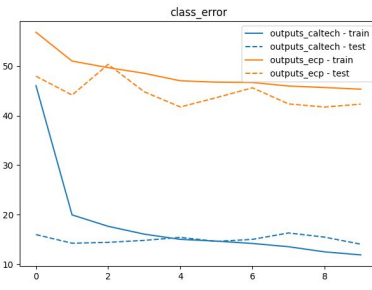
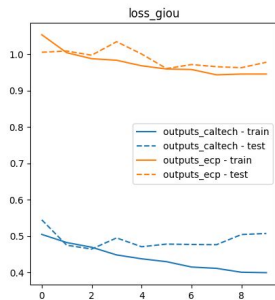
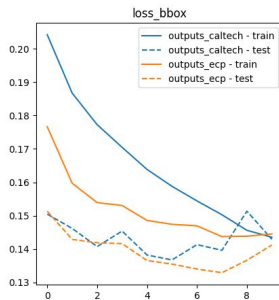
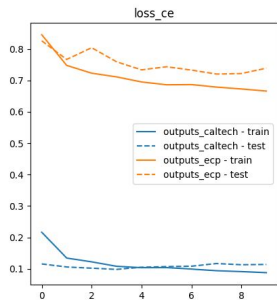
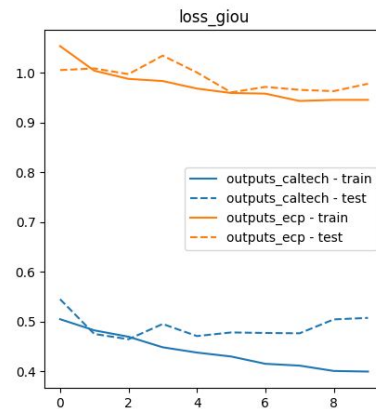
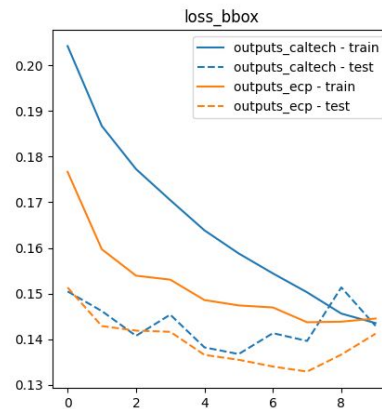
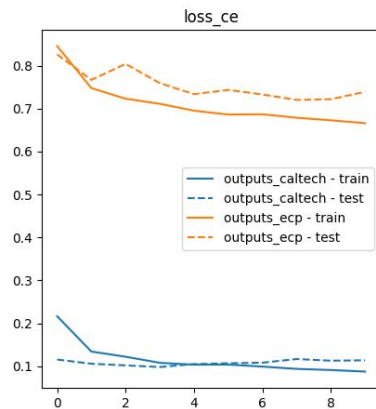
ECP

IoU metric: bbox					
Average Precision	(AP) @[IoU=0.50:0.95	area= all	maxDets=100] = 0.055
Average Precision	(AP) @[IoU=0.50	area= all	maxDets=100] = 0.129
Average Precision	(AP) @[IoU=0.75	area= all	maxDets=100] = 0.038
Average Precision	(AP) @[IoU=0.50:0.95	area= small	maxDets=100] = 0.026
Average Precision	(AP) @[IoU=0.50:0.95	area=medium	maxDets=100] = 0.127
Average Precision	(AP) @[IoU=0.50:0.95	area= large	maxDets=100] = 0.327
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets= 1] = 0.051
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets= 10] = 0.099
Average Recall	(AR) @[IoU=0.50:0.95	area= all	maxDets=100] = 0.113
Average Recall	(AR) @[IoU=0.50:0.95	area= small	maxDets=100] = 0.072
Average Recall	(AR) @[IoU=0.50:0.95	area=medium	maxDets=100] = 0.216
Average Recall	(AR) @[IoU=0.50:0.95	area= large	maxDets=100] = 0.460

- **Epochs:** 10
- **Model:** *detr-r50_no-class-head*
- **Batch Size:**
 - 32 (Caltech)
 - 16 (ECP)
- **Training Time:**
 - 1:11:46 (Caltech)
 - 3:25:15 (ECP)
- mAP has same phenomenon as YOLO due to poor distribution of rare classes

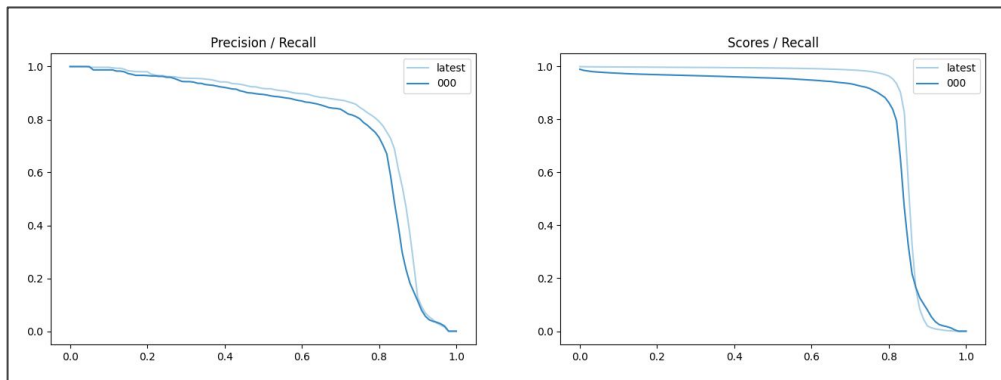
DETR Metrics Comparison

- Loss trends negatively with each epoch
- Class Error and GloU Losses may be due to diminished mAP



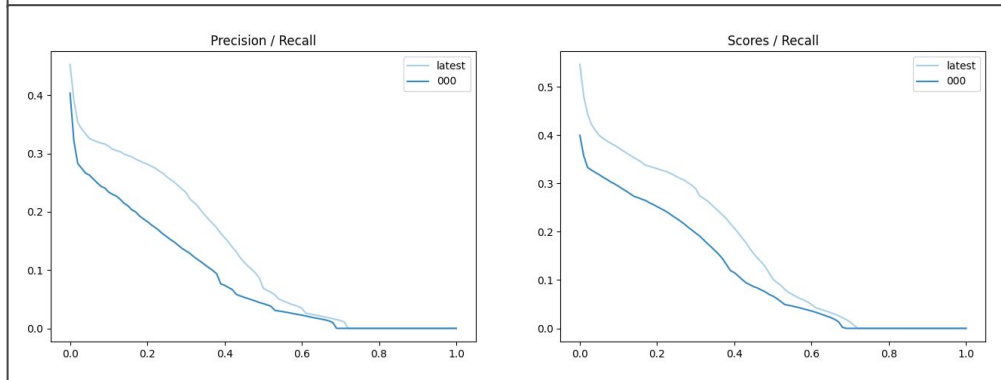
DETR Precision / Recall Comparison

Caltech



- Large area under curve → decrease confidence threshold
- Reduce FN, high precision
- Precision: 0.468
- Recall: 0.399
- Dataset lacks diversity

ECP



- Good performance despite charts (mAP phenomenon)
- Higher FP rate than YOLOv8 ECP
- Precision: 0.055
- Recall: 0.051

Limitations

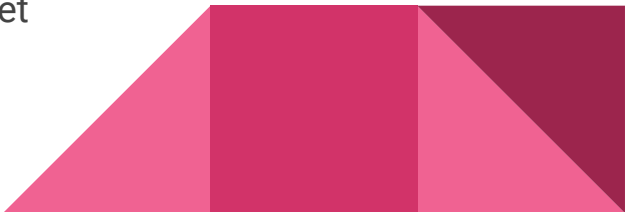
- **Computation Resources**

- The computing power necessary to further train and tune additional models was a limitation, as was the operating system. Rivanna and Azure each had dependency limitations which prevented various aspects of the different models and their architectures from being fully trainable and/or tuneable.

- **Ease of Use**

- The transformer model presented interoperability issues particularly with loading it on Windows to run the demo application.
- Additionally, when attempting to work with the transformer model, we encountered difficulties in developing any kind of app output.

- **Quality of Training Data**

- Mislabeled object types
 - Lack of diversity in the Caltech dataset
 - Lack of uniform distribution among classes for the ECP dataset
- 

Conclusion

- This pedestrian detection model could have valuable application in pedestrian safety throughout the state of Virginia, but particularly on college and university campuses. Ex: campus buses/shuttles
- The app demonstrated how an out-of-the-box model is not suitable, as well as the necessity for a dataset with a variety of domain applications and diverse scenarios and conditions.
- Future work should include continued training on more diverse data and equally represented object classes in the ontology, as well as development of the app for application in vehicles, etc. For specific application to campus safety, universities could contribute by gathering data representative of their location, events, and population for future model training.