Deep Learning for Pedestrian Detection

Abigail Snyder
School of Data Science
The University of Virginia
syc6vs@virginia.edu

Suraj Kunthu
School of Data Science
The University of Virginia
sk9km@virginia.edu

Dominic Scerbo
School of Data Science
The University of Virginia
ybt7qf@virginia.edu

***Abstract*—Pedestrian safety is an issue of critical proportions in the U.S. Recent estimates from the Governor's Highway Safety Association (GHSA) report "that drivers struck and killed at least 7,508 people walking in 2022 – the highest number since 1981 and an average of 20 deaths every day [1]." The aim of this project is to develop an efficient and accurate pedestrian detection model using deep learning techniques with the hope that such a model could be applied to a full pedestrian detection 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 focuses on training an object detection model on the EuroCity Persons (ECP) and CalTech Pedestrian data sets.**

Group-1 Pedestrian Detection Repository

I. Literature Review

A. Introduction

Pedestrian detection is a critical task for improved pedestrian safety. Applications of an effective computer vision pedestrian detection system have applications ranging from autonomous vehicles to surveillance systems. Over the years, significant progress has been made in developing accurate pedestrian detection systems, with deep learning techniques playing a pivotal role. This literature review provides an overview of key research contributions and trends in the field of deep learning-based pedestrian detection systems.

B. Traditional Approaches vs. Deep Learning

Historically, pedestrian detection primarily relied on handcrafted features and traditional machine learning algorithms. These methods often struggled with variations in pose, scale, lighting, and occlusions [2]. The emergence of deep learning has revolutionized this field by enabling the automatic learning of discriminative features from raw data [3]. 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 [4].

C. Key Deep Learning Architectures

CNNs have been the foundation for many pedestrian detection systems [3]. Early models, such as the R-CNN and Fast R-CNN, employed region-based CNNs for object localization. Modern CNN-based architectures, including Faster R-CNN, SSD, and YOLO, have achieved real-time detection while maintaining high accuracy [2] [5].

SSD is a popular choice for pedestrian detection due to its speed and accuracy. It combines multiple convolutional layers to predict bounding boxes and class scores at various scales [6]. SSD-based models have demonstrated strong performance in detecting pedestrians under various conditions. Researchers found that by integrating an additional classifier at each stage of a multi-stage cascaded classifier within the deep learning model, an enhanced optimization strategy is achieved [6]. This modification effectively mitigates overfitting and substantially reduces the error rate during testing across multiple datasets [6].

YOLO is known for it's real-time object detection capabilities, optimized through directly predicting the object center and bounding box through anchor placement on the feature map [2]. YOLOv3 and YOLOv4 variants have improved precision and recall in pedestrian detection tasks. Its single-stage architecture processes the entire image at once, making it efficient for real-time applications, where speed is of the essence in improving pedestrian safety [2]. The open-source community has continued to improve the YOLO method. The latest version, YOLOv8, introduces several new features and improvements. These improvement include a lighter weight architecture to improve speed and efficiency, An updated backbone network based on EfficientNet, which improves the model's ability to capture high-level features, a new feature fusion module that integrates features from multiple scales, and enhanced data augmentation techniques [15].

Transformer models have established themselves as the standard modeling method in the field of Natural Language Processing (NLP). Emerging from this trend, ViTs have proven to be a compelling alternative to CNNs. ViT models leverage the concept of "Attention" typically in conjunction with CNN architectures. This component quantifies how different parts of data interact, aiding a network in learning structure and patterns in the input data and boosting the robustness of vision networks [8]. Furthermore, studies have shown ViT-based methodologies for pedestrian detection have effectively lowered log-average miss rates while simultaneously achieving faster processing speeds [7].

D. Challenges and Innovations

1) Occlusion Handling

- Dealing with partial occlusions remains a challenge in pedestrian detection. [2] [6]
- Multi-stage models and advanced post-processing techniques have been proposed to improve occlusion handling. [3] [4]

2) 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. [4]
3) Real-time Processing
   - Achieving real-time processing and strict accuracy requirements for pedestrian detection in video streams is essential for applications like autonomous driving. [2]
   - Efficient architectures like YOLO have made real-time detection feasible.

E. Datasets

Datasets like Caltech Pedestrian Dataset, ECP Dataset, Waymo Open Dataset, KITTI, INRIA Person Dataset, and CityPersons are widely used in advancing pedestrian detection research [2] [6]. These datasets are specifically tailored to pedestrian detection tasks containing a variety of environments and scenarios one would be exposed to while driving a motored vehicle. The datasets also spans a variety of modalities to include video, images, and 3-dimensional sensor data. For the scope of this review, the Caltech Pedestrian Dataset and ECP Dataset will be further discussed due to ease of use and availability. Furthermore, a more generic dataset, Microsoft Common Object in Context (MS COCO), which has also aided in the development of pedestrian detection models will also be discussed [10].

COCO is one of the most popular public datasets for computer vision. It is a very large dataset that contains 80 categories of common objects with over 1.5 million object instances. Its versatility, multi-purpose scene variation, and a variety of annotation formats have enabled it to become a common benchmark for computer vision performance among various types of tasks, such as object detection, full scene segmentation, key point detection as well as natural language descriptions of images [11]. Today, the top-performing model on this dataset for object detection is Co-DETR, which is a transformer-based detection model, with an average precision of 66.0 [12]. For the intent of this project, models trained on the COCO dataset can help in a comparative analysis and test and evaluation of model architectures and performance. A model showcasing desired performance and speed can be leveraged and then fine-tuned on datasets tailored specifically to Pedestrian detection to achieve the desired level of accuracy. Link: MS COCO

The Caltech Pedestrian Dataset contains around 10 hours of 640x480 30Hz video captured from a vehicle in urban traffic [13]. It comprises approximately 250,000 frames in 137 segments, totaling 350,000 bounding boxes and 2300 distinct pedestrians. [13] Even though the dataset is contained in a video format tools, such as Python, can be used to convert this data into batches of annotated images. Not only can this dataset on its own support a transfer learning process but it can also be used to supplement other datasets to reduce biases such as out-of-distribution scenes. Link: Caltech Pedestrians

The ECP dataset offers extensive, diverse, and precise annotations for pedestrians, cyclists, and other riders in urban traffic scenes [14]. It features images from 31 European cities, with 238,200 person instances labeled in 47,300 images, including over 211,200 person orientation annotations, making it almost ten times larger than other benchmark datasets [14]. Another added benefit to this data set is that it not only spans a wide variety of scenes/environments, but also includes images during the day and night. Thus, this dataset will help this model generalize well to different environments and times of day. Since this dataset strives in volume and veracity it can be used for not only fine-tuning a pretrained model but potentially training a model from scratch. This dataset is believed to be well-suited for the task of pedestrian detection and can help achieve the desired levels of accuracy. Note the link to the dataset below requires access approval. Our team has been granted access and downloaded the data into the group-1/datasets/eurocity folder within the ds6050 rivanna project. Link: Eurocity Persons

F. Conclusion

Deep learning has significantly improved the accuracy and efficiency of pedestrian detection systems. Architectures like CNNs, SSD, ViT, and YOLO have become standard choices for researchers and practitioners. Despite the progress made, challenges like occlusion handling and real-time processing continue to motivate further research in this field. The development of robust and efficient pedestrian detection systems is crucial for enhancing pedestrian safety and enabling various applications in computer vision.

II. Project Proposal: Deep Learning for Pedestrian Detection

A. Project Overview

Pedestrian detection is a crucial component of various computer vision applications, including autonomous vehicles, surveillance systems, and pedestrian safety. This project aims to develop an efficient and accurate pedestrian detection system using deep learning techniques. An effective pedestrian detection system has potential application in current-model vehicles for improvements in pedestrian detection and safety, as well as in the ever-developing field of autonomous vehicles. In this project, we will harness the power of transfer learning by utilizing a pretrained model on the COCO dataset to enhance our performance on the EuroCity and/or Caltech dataset. We will also consider supplementing this model as needed using the secondary dataset depending on validation accuracy.

B. Project Objectives

The goal of this project is to create an efficient and accurate pedestrian detection model using deep learning techniques. We will use the Caltech and Eurocity datasets to train the model and test performance.

C. Data Collection and Preparation:

- Gather a diverse and representative dataset of images containing pedestrians and non-pedestrian objects (EuroCity and/or Caltech). Convert video-based dataset (Caltech) into images if needed to supplement the foundational dataset.
- Evaluate label ontology and refine for use case as needed.
- Review existing annotations and refine/create labels with bounding boxes around pedestrians as needed.
- Convert the annotations to the appropriate format for the model training process.
- Partition the data into training and testing data sets.

D. Model Selection and Architecture:

- Research and choose appropriate deep-learning architectures for pedestrian detection. This research will evaluate the models in terms of inference speed as well as accuracy on benchmark datasets, such as COCO.
- Fine-tune or adapt the selected model for our specific use case.

E. Training and Evaluation:

- Train the chosen model using the annotated data set, optimizing for accuracy. It will be important, in this context, to minimize false negatives, as the consequences of not accurately detecting a pedestrian are high. False positives, in this context, generally refer to person-like objects that are not pedestrians. In the end, a strong model will both minimize false negatives and be able to differentiate between pedestrians and person-like objects.
- Measure and evaluate the inference time of the model to optimize for efficiency.
- Evaluate the model's performance on a separate test data set.
- Leverage Machine Learning Operations (MLOps) software, such as Weights and Biases and/or Tensorboard, during training and testing processes to conduct large-scale experimentation, monitor performance in real-time, and share results more effectively with the team.
- Leverage hyperparameter tuning strategies to maximize model performance as needed.
- Tweak model architecture as needed for improved performance.

F. Performance Bench-marking:

- Compare the developed deep learning-based solution with existing pedestrian detection methods.
- Measure performance metrics, including precision, recall, and F1-score.

G. Sample Experiments:

- Inference Test: Test pedestrian detection accuracy and speed using pre-trained model with COCO weights on the Caltech and/or EuroCity Pedestrian datasets. This can include but is not limited to YOLOv8 and DETR.
- Transfer Learning: The model showcasing the best performance in terms of accuracy and speed will be used for further transfer learning to minimize false negatives.
- Hyperparameter Tuning: Hyperparameter Tuning will be used to help increase accuracy and minimize the false negative rate as much as possible for this use case.

H. Deliverables

- Annotated dataset of pedestrian images.
- Trained deep learning model for pedestrian detection.
- Evaluation report detailing model performance.
- Documentation for the model and system.
- Project Presentation

III. Methods

In this project, we employed advanced methodologies, specifically the YOLOv8 and DETR models—versions of the YOLO and ViT models, respectively. Through our initial exploration and experimentation with these models, it became evident that the choice between them could be tailored to specific scenarios, taking into account the computing environment and data availability.

As highlighted in our literature review, YOLOv8 demonstrates a remarkable capability for near-real-time inference coupled with a high degree of accuracy. This characteristic renders it particularly well-suited for pedestrian detection across diverse devices, accommodating a spectrum of computational resources. In contrast, DETR, although proven to enhance accuracy on benchmark COCO datasets, falls short in terms of suitability for near-real-time inference, especially when constrained by limited compute resources to power the system. However, the transformer architecture, renowned for its remarkable ability to capture complex patterns through self-attention mechanisms, has been particularly effective in enhancing accuracy during the transfer learning process, especially when faced with limited data. Therefore, we will compare and contrast the abilities of these models on the Caltech and Eurocity datasets which present contrasting properties in terms of diversity and quantity.

IV. Motivation

This research is motivated by the critical issue of pedestrian safety in the United States, as underscored by the alarming increase in pedestrian fatalities. The Governor's Highway Safety Association reported a staggering 7,508 pedestrian deaths in 2022, the highest since 1981. The overarching goal of this project is to develop a robust pedestrian detection model using advanced deep learning techniques. Such a model holds immense potential for widespread applications, ranging from safety warnings in existing vehicles to the future integration of autonomous vehicles and surveillance systems. The emphasis

is on leveraging efficient and accurate object detection models, specifically YOLOv8 and DETR, trained on the ECP and Caltech Pedestrian datasets. The project acknowledges the diverse needs of users and aims to provide a versatile solution, allowing the choice of models based on specific scenarios, considering factors such as computing environment and data availability. Through a comparative analysis of these state-of-the-art models using the Caltech and Eurocity datasets, the research endeavors to present contemporary frameworks with distinctive advantages and limitations. This exploration offers a nuanced understanding, allowing further research to be tailored to particular constraints and restraints to enhance pedestrian safety.

V. Experiments

A. Preprocessing

To train a model on the Caltech and ECP datasets a significant amount of preprocessing needed to be done. Each model uses a custom annotation format and directory structure that we need to abide by for the training to occur.

The format for YOLOv8 is as follows:

```
/datasetname
    /images
        /train
            -image-id.png
        /val
            -image-id.png
    /labels
        /train
            -image-id.txt
        /val
            -image-id.txt
    datasetname.yaml
```

Each label text file contains the name as its respective image in the adjacent folder. With the text file, each row contains a class ID for the label followed by the pixel coordinates of the associated bounding box. For images with no labels, a text file does not exist and it will be used in a negative / background class. The dataset.yaml contains the paths to the dataset, images, and labels, as well as an enumerated list of class categories for the IDs specified in the label files. The bounding box information for the labels includes normalized pixel values indicating the x and y for the center of the bounding box followed by the width and height of the image. The benefit of this format allows the user to resize images as needed without needing to modify the bounding box information for the labels.

The format for DETR uses the COCO JSON format and the following directory structure.

```
/datasetname
    /images
        /train2017
            -image-id.png
        /val2017
            -image-id.png
    /annotations
        -instances-train2017.json
        -instances-val2017.json
```

The COCO formatted JSON consists of a list of enumerated categories within the annotation files, and a list of images with respective names, IDs, and image dimensions. Lastly, it contains a list of enumerated annotations that are associated with an image by its ID and contains the class category ID, and for this use cases bounding box information. The bounding box in this case uses the true x and y pixel coordinates of the top left corner of the box followed by the width and height.

The Caltech dataset consisted of annotated video files that needed to be converted into separate image files with respective annotations for each frame. Various Python image video and image processing tools were used to make this conversion and begin the transformation to the YOLOv8 format. The bounding box and labels we also converted from pixel coordinates to the format required for the YOLOv8 model. Once the data was successfully converted to the custom YOLOv8 format open-source Python custom code was written to automate the transformation to the respective COCO format.

The ECP dataset consisted of actual images with its own custom label format. Similar utilities to acquire an enumerated list of labels categories and functions to convert the bounding boxes information to the YOLOv8 format were used. Since the dataset was nearly 100G in size, multi-processing was utilized to accelerate the conversion. With the labels in the YOLOv8 format, the same utility to convert the YOLOv8 labels to the COCO JSON was used.

A major deficiency discovered in the preprocessing stage was that both datasets were not free of error. Several files contained partial, incorrect, or missing annotations. This is an issue that can lead to an increase in false positives and false negatives. Situations in which the bounding box does not accurately label the target class can lead to false positives, while the absence of labels can result in false negatives.

B. Dataset Analysis

After completing the preprocessing of the data we were able to obtain class and biome distributions for each dataset. In figure 1, a comparative table showcases the differences between the Caltech Pedestrian dataset and the ECP dataset.

The table highlights the substantial advantages of the ECP dataset over the Caltech Pedestrian dataset. Notably, the ECP dataset boasts approximately 18 times more images and approximately 40 times more identified persons. Moreover, it encompasses diverse elements such as images from various countries and cities, capturing different times of day and featuring two distinct weather conditions.

Given these significant differences, it is evident that the ECP dataset is better suited for pedestrian detection tasks and is likely to yield higher accuracy levels when compared to the Caltech Pedestrian dataset.

In Figure 2, the distribution of identified objects within the images is depicted. Notably, the ECP dataset offers a

	CalTech	Eurocity Persons
# images	2,498	45,157
# persons	3,534	138,620
Countries	1	12
Cities	1	31
Day/Night	day	day, night
Weather	dry	dry, wet

Fig. 1. Comparison of CalTech Pedestrian Dataset to ECP

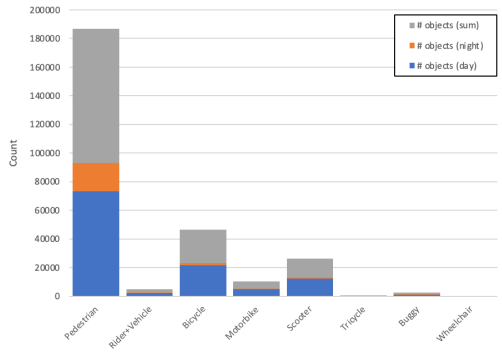

Fig. 2. Distribution of objects in ECP dataset

substantial number of data objects, prominently labeled as "Pedestrian." This abundance of pre-defined pedestrian objects is a significant asset for pedestrian detection tasks. It is crucial to acknowledge the absence of adequate representation for certain classes, a factor that could lead to increased errors in these minority classes, consequently exerting a detrimental influence on the overall precision and recall of the model.

However, it is worth noting that the majority of these identified objects are present in daytime images. This concentration of daylight images may pose challenges for detecting pedestrians in adverse visibility conditions or during nighttime. Additionally, the dataset reveals that bicycles constitute the most prominent among the identified vehicles. This observation might be attributed to the fact that individuals on bicycles are more distinguishable than those in other types of vehicles.

In Figure 3, all the images originate from European cities, in line with the dataset's title, "Eurocity Persons." Notably, Roma and Barcelona stand out as the cities with the highest distribution of associated images. This observation is consistent with the higher prevalence of cyclists in Europe compared to the United States, and it can be inferred that Roma and Barcelona are cities with significant cyclist populations.

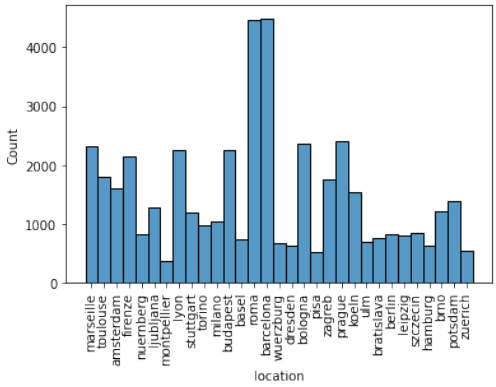

Fig. 3. Distribution of Locations in ECP dataset

C. Modeling and Analysis

The experiments used are documented in the table below, showing each model, weights, epochs, batch size, image size, and other parameters for training I.

Based on those experiments we achieved a number of high-level outcomes, with performance varying by model II.

However, after further experimentation with trained instances across dataset and on unseen test datasets we found the mAP to not be indicative of the performance of the model. First and foremost, the biggest discrepancy comes from the calculation across all object classes. For example, III showcases how the model how poor performance on select classes with few instance counts where the mAP was 0 while our primary target class (Pedestrian) had a mAP of 0.459 which is much higher. This case also appears to be true for the DETR model as well but a precision calculation was not obtained. It is estimated that the mAP is close to the large object mAP value of about 0.4. To perform a true comparative analysis of these models experimentation using a single pedestrian class would need to be done which will also need to include the analysis of data labels to determine how to remove or aggregate them without decreasing the model's performance on the Pedestrian class. Alternatively, the Caltech data could be fused to supplement the ECP data and perform a comparison between the fused dataset.

We also found that the model trained on the Caltech dataset did not transfer well to the ECP data, but the model trained on the ECP data did perform well on the Caltech data. This is primarily due to the simplicity of the Caltech dataset which contains little occlusion, less diverse weather conditions, and little variability in depth of field/position/angle for pedestrians in the dataset. Therefore, to truly evaluate the error of the models in a less relative manner we utilized test data that none of the models had seen before. This experimentation indicated different results than what was gleaned from the mAP from each model. As a result, we found that overall, 1) YOLOv8 using Caltech data had the most amount of false negatives 2) YOLOv8 using the ECP data had very few false negatives with the least amount of false positives 4) DETR using Caltech data had fewer false negative than the YOLOv8 model and

Architecture	YOLO	YOLO	DETR	DETR
Weights	yolov8n	yolov8n	detr-r50	detr-r50
Dataset	Caltech	ECP	Caltech	ECP
Epochs	50	50	10	10
Batch Size	64	32	32	16
Image Size	640	1024	640	1024
Early Stopping	Yes	Yes	No	No
Optimizer	AdamW(lr=0.002, momentum=0.9)	AdamW(lr=0.002, momentum=0.9)	AdamW(lr=0.0001, momentum=0.9)	AdamW(lr=0.0001, momentum=0
Device	NVIDIA a100 40Gx1	NVIDIA a100 40Gx1	NVIDIA a100 40Gx1	NVIDIA a100 40Gx1

TABLE I
SUMMARY OF EXPERIMENTS AND MODEL TRAINING

Model	YOLO (Caltech)	YOLO (ECP)	DETR (Caltech)	DETR (ECP)
mean Average Precision (0.50:0.95)	0.411	0.133	0.468	0.055
Pedestrian mAP	0.411	0.459	0.468	0.4
Training Time (HH:MM:SS)	00:39:44	01:58:31	01:11:46	03:25:15

TABLE II
RESULTS FROM EACH MODEL

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

TABLE III
SUMMARY OF MODEL METRICS

4) DETR using ECP data had very false negatives, but more false positives than the YOLOv8 model. Since our unseen test data was not labeled precise metrics of performance were not obtained; however, it is recommended further experimentation should be carried out using a labeled test dataset.

D. Workflow Timing

In addition to the modeling and analysis we also conducted timing experiments to validate if the models would be able to perform an end-to-end inference workflow in near-real time. Following the execution of the inference experiments for the YOLOv8 models, 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 4.

The DETR models, on the other hand, exhibit 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 5.

On average we observed a mean inference time of roughly 14000ms, 2000ms, and 850ms for the benchmark pretrained, Caltech fine-tuned, and ECP fine-tuned DETR models respectively. This was a much larger latency than the respective YOLOv8 models with inference times of 96ms, 200ms, and 250ms. Thus, we found that the YOLOv8 model is much more practical for near-real-time inference as opposed to the DETR model unless sufficient vRAM is provided on the inference server.

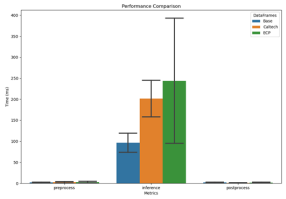

Fig. 4. YOLOv8 Workflow Timing

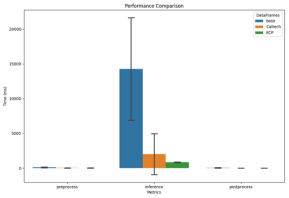

Fig. 5. DETR Workflow Timing

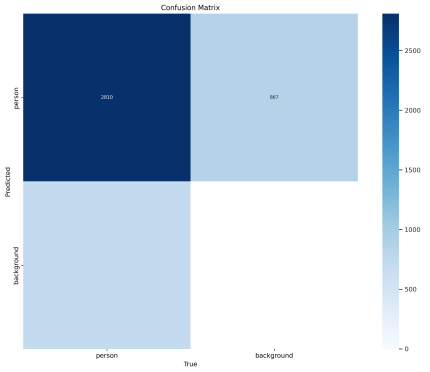

Fig. 6. Confusion Matrix for YOLOv8s on Caltech data

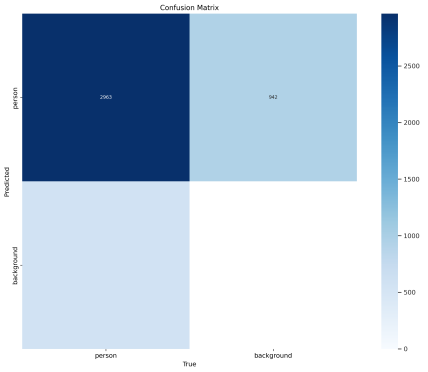

Fig. 8. Confusion Matrix for YOLOv8x on Caltech data

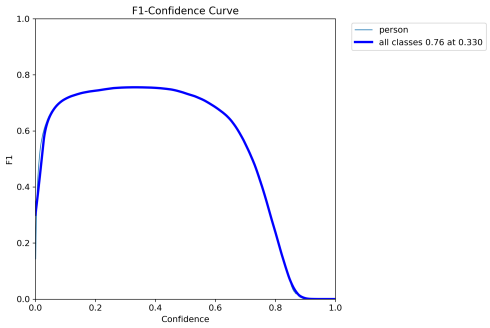

Fig. 7. F1 Curve for YOLOv8s on Caltech data

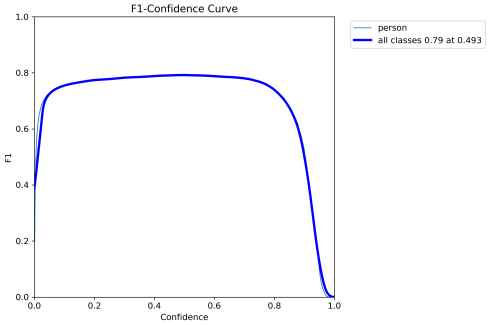

Fig. 9. F1 Curve for YOLOv8x on Caltech data

VI. Results

A. YOLOv8 on the Caltech Pedestrian Dataset

We started by training the YOLOv8s pretrained model on the Caltech Pedestrian dataset for 100 epochs. With no modifications, the model training stopped early, at 50 epochs, as no improvement was achieved after the first run. The first run of the model achieved a MAP-50-95 of 0.4384. The model incorrectly labeled 942 persons (false positives) and correctly identified 2963 persons (true positives) 6. Of the total persons in the data (3534), this is an 83.84% true positive rate and 26.66% false positive rate. The precision of this model was 0.743, with a 0.764 recall, and F1-confidence score of 0.76 at 0.330 7.

Before adjusting any hyperparameters on the YOLOv8s model, we also trained the YOLOv8x model on the same data. Similarly to the 8s model, the 8x version stopped after 50 epochs, having achieved no improvements after the first run, which had a MAP-50-95 of 0.4586, which is only a slight improvement on the simpler model. This model incorrectly labeled 867 persons and correctly labeled 2810, giving it a 79.51% true positive rate and 24.53% false positive rate 8. The precision was 0.7807 with a recall of 0.8031, and F1-confidence score of 0.79 at 0.493 9.

On the Caltech Pedestrian dataset, the larger YOLOv8x pretrained model seem to have slightly better performance.

B. YOLOv8 on the Eurocity Persons Dataset

As previously mentioned the Caltech dataset is less robust than the ECP data in terms of class occlusion, type of pedestrian, time of day, weather conditions, and even distance of each object. So, while this dataset does allow for rapid prototyping of models it does not serve as a well-balanced robust dataset for the task. Hence, the use of the ECP dataset. However, using this dataset, the model will have to learn much more information to achieve the levels of performance obtained on the Caltech data. This process was started by fine-tuning the YOLOv8 model on the ECP Dataset. For the training job, we utilized the yolov8s.pt base model with an image size of 640, a batch size of 32, and 50 epochs. The yolov8s model was bench-marked on the COCO dataset with a mAP of 44.9 containing 11.2 million parameters. A Rivanna Jupyter Lab instance with 16 cores, 64G of memory a single NVIDIA A100-SXM4-40GB GPU ran this job in 2h 1m 59s.

Unlike the training done on the Caltech data, this experiment did not stop early at 50 epochs, but continued to climb in performance. The first run of the model achieved a MAP-50-95 of 0.1103. The figures below provide an overall direction of the decrease in loss 10 as well as the increase in mAP 11.

However, these metrics are slightly misleading due to the balance of annotations in the dataset. As we previously mentioned, Pedestrian is the dominant class in the ECP dataset, however, there are other classes with fewer annotations that the model is incorrectly classifying. As a result, this is diminishing

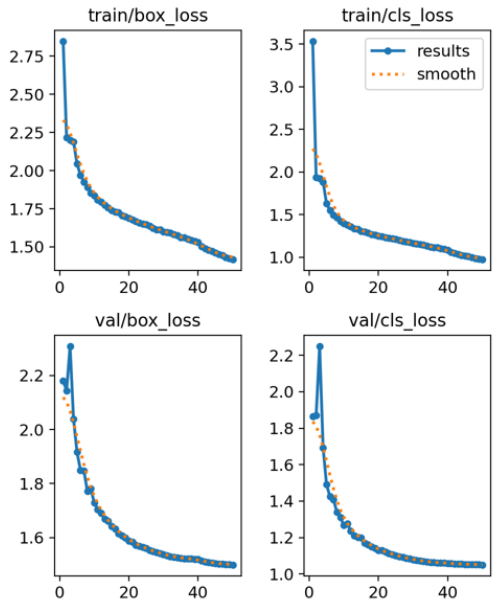

Fig. 10. YOLOv8 Loss on ECP

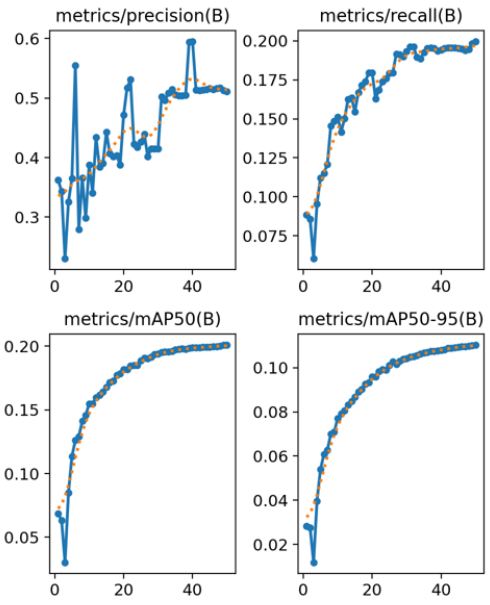

Fig. 11. YOLOv8 Metrics on ECP

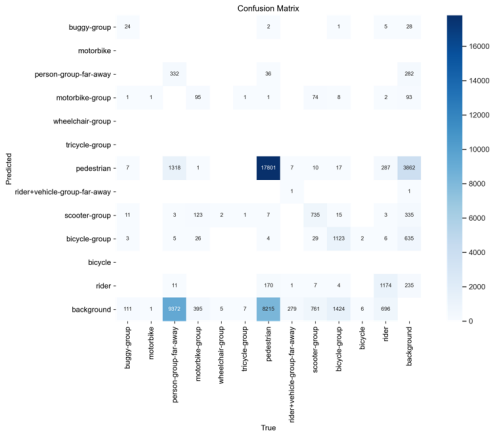

Fig. 12. YOLOv8 Confusion Matrix on ECP - V2

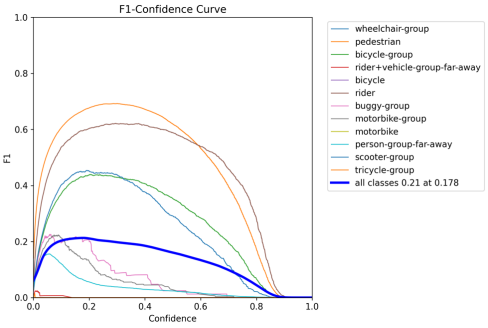

Fig. 13. YOLOv8 F1 Curve on ECP

the mAP which is calculated across all classes. The following confusion matrix shows this imbalance 12. However, the model does not predict the Pedestrian class significantly better or at the desired level of performance. As we can see the class is often confused for the background or a group of people. This indicates that the metrics could potentially be improved by performing some re-balancing of the labels by aggregating classes or even removing various class types with few annotations.

Lastly, the F1 13 and Precision-Recall 14 curves indicate the model could be significantly improved to correctly classify the labels with high confidence. Some classes have very poor performance, confirming the necessity to balance the annotations for this model. As for our dominant Pedestrian class, the precision drastically decreases as there is an increase in recall and confidence. For peak performance, this model would require more training data to help further increase the precision of the model.

As an important note on the evolution of this model, further analysis of the labels and image quality, it was discovered that there were slight errors in the bounding boxes during the conversion process. With those corrections in addition to the removal of a None type label classes and changing the image resizing function from 640 to 1024 we began to see improvement in the models' performance. Employing a larger image size necessitated a reduction in the batch size, from

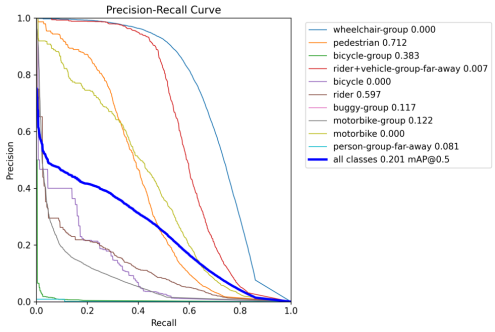

Fig. 14. YOLOv8 Precision-Recall Curve on ECP

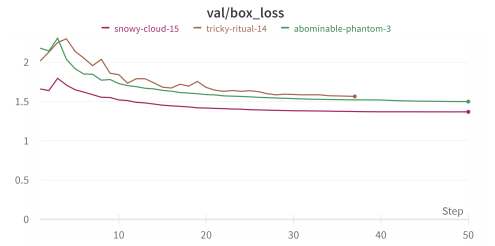

Fig. 15. YOLOv8 Loss Comparison on ECP

64 to 32, and consequently led to a marginal increase in the training duration. The images below illustrate how those modifications led to an overall improvement of the model's ability to detect (mAP) and localize (box loss) objects in the scene - as indicated by in Red - snowy-cloud-15 versus the prior model runs 15 16.

C. DETR on Caltech Dataset

We experimented with fine-tuning the ViT model DETR which contains 41,279,238 parameters (3.6 times larger than YOLOv8s). This model was trained on the same instance type with a single NVIDIA A100-SXM4-40GB GPUs. The training job was performed over 10 epochs with a batch size of 32 and 15 workers for the data loader. This job ran in 1h 53m 43s. The training time here shows the deeper transformer architecture is more computationally expensive than the YOLO method which took 48m 50s over 5 times as many epochs. This model does, however, achieve a mean average precision and recall (0.50:0.95) of 0.468 and 0.399 respectively. A further

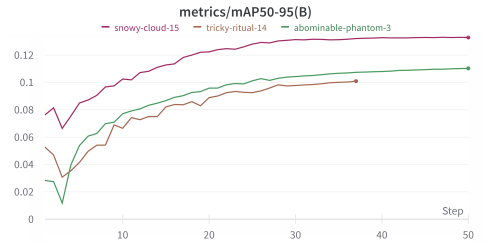

Fig. 16. YOLOv8 mAP Comparison on ECP

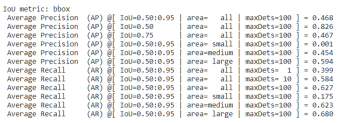

Fig. 17. DETR Caltech Performance Metrics

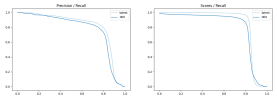

Fig. 18. DETR Precision / Recall Curves

breakdown of the metrics is figure 17, where the image shows how the model achieves much higher precision and recall on larger objects in the image.

Based on the large amount of area under the curve for the precision-recall curve illustrated in figure 18. Thus, this model would allow us to decrease the confidence threshold of the model to allow for a reduction in the number of false negatives while still maintaining a high level of precision. However, the downside to this model is similar to that of the YOLOv8 model trained on the Caltech data. The issue resides in the model not being exposed to a more diverse Pedesitran class in terms of the biome backgrounds and field of view - ultimately revealed by a batch of test images that showcased the limited predictive power of the model in a practical use case.

D. DETR on ECP Dataset

The ECP dataset was also applied to the DETR model. In this scenario we also trained on the same instance type with a single NVIDIA A100-SXM4-40GB GPUs. The training job was performed over 10 epochs with a batch size of 16 and 15 workers for the data loader. This job ran in 3h 25m 15s. The training time here shows the deeper transformer architecture is more computationally expensive than the YOLO method which took 1h 58m 31s over 5 times as many epochs. This model achieved a mean average precision and recall (0.50:0.95) of 0.055 and 0.051 respectively. A further breakdown of the metrics is figure 19, where the image shows how the model achieves much higher precision and recall on larger objects in the image.

Contrary to the precision-recall curve illustrated below, the DETR model showcased great performance on our test dataset, figure 20. This is due to the same phenomenon that existed in the YOLOv8 model where the mAP of the rarer and more unequally represented object classes that the model performed poorly on diminished the overall mAP for the model. A more in-depth analysis and the ability to extract aggregated metrics

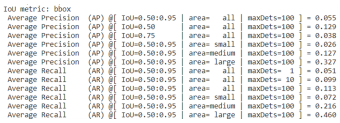

Fig. 19. DETR ECP Metrics

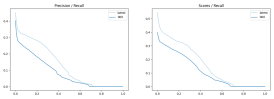

Fig. 20. DETR ECP Precision / Recall Curves

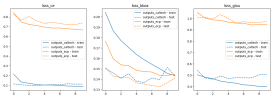

Fig. 21. DETR Loss Metrics

Fig. 22. Sample Pedestrian Detection using the YOLO model trained on the ECP dataset

from the DETR model could reveal that the true mAP on the Pedestrian class is much closer to that of the YOLOv8 model. From further testing, it was found that the model did however showcase more false positives than the respective YOLOv8 model. This metric gauges performance in a manner that doesn't compromise pedestrian safety but rather poses an inconvenience to end users who may be displeased if heightened noise levels negatively affect their system.

The supplementary charts below, featuring the loss_bbox metric, highlight the enhanced capability of the models to accurately localize objects within the scene, thanks to the utilization of the ECP dataset 21. This metric enables us to make direct model-to-model comparisons, even considering the distinct class structure for each dataset. Consequently, the utilization of the ECP data becomes evident in its positive impact on overall object localization for our specific use case. This improvement, in turn, facilitates more effective tracking of pedestrians across varying fields of view and weather conditions within the scene.

E. Limitations and Mitigations

1) Computing Resources: The computing resources available proved a significant limitation in testing other architectures, such as the transformer models, for application to pedestrian detection. Due to the size of the dataset as well as the complexity of such models, the time and resources necessary to train the models effectively were not an option for this specific project.

In addition, there were limitations due to the operating systems available, as the transformer model required Linux, which Azure's GPU did not support, while the Ultralytics YOLO tuning package conflicted with Rivanna's environments and operating system.

2) Ease of Use: In the end, our goal with this project was to create a functioning pedestrian detection system. With that in mind, the application of various models to user-friendly apps became a limitation. The transformer model, in particular, did not prove user-friendly when attempting to output results to a working app, whereas, the simpler YOLO model was more user-friendly.

3) Quality of Training Data: The challenges in object recognition and classification persist due to several factors. Firstly, mislabeled object types introduce confusion and inaccuracies in the learning process, hindering the model's ability to correctly identify and categorize objects. Additionally, the Caltech dataset exhibits a lack of diversity, limiting the model's exposure to various object instances and scenarios. This deficiency in representation can lead to biased or incomplete learning outcomes. Furthermore, the ECP dataset faces issues with a non-uniform distribution among classes, impacting the model's performance as it may be more adept at recognizing overrepresented classes while struggling with underrepresented ones. Addressing these challenges is crucial for enhancing the robustness and accuracy of object recognition models.

VII. Conclusion

At the conclusion of the project, we had a working app that used the trained weights from the ECP dataset and the YOLO model to effectively detect pedestrians in real-time 22. Though there is certainly room for continued improvement and wider application of this app and model, the goals of the project were met.

In conclusion, our research findings can be summarized as follows:

- Pretrained state-of-the-art (SOTA) models, such as YOLOv8 and DETR, exhibit commendable predictive capabilities for pedestrian detection. However, they fall short in handling nuanced factors such as varied weather conditions, different times of day, and diverse fields of view.
- The introduction of a dataset, like ECP, encompassing diverse weather conditions, depth of field, and varied positions/angles of the pedestrian class has proven effective in reducing the occurrence of false negatives. Notably, this improvement has been achieved while maintaining an acceptable level of inference latency, even without the use of accelerated computing devices.
- In scenarios where the data scientist lacks access to extensive ground truth labeled data, employing a Vision Transformer (ViT) architecture can be advantageous for enhancing model performance. This approach, however, requires the availability of sufficient computational resources or time for training.

A. Implications

This pedestrian detection model could have valuable applications in pedestrian safety throughout the state of Virginia, but particularly on college and university campuses. Because many campuses are more walkable than many suburban areas (and even city centers), they more closely resemble many of the scenarios within the ECP dataset as noted from the data analysis of the dataset. They also typically have a higher concentration of pedestrians, meaning the use of a pedestrian detection system in vehicles, such as campus buses or shuttles, could be especially useful in reducing accidents and even fatalities.

B. Further Development

When viewing the real-time pedestrian detection in the app, it is easy to see how an out-of-the-box state-of-the-art model is not effective, as it often misses pedestrians, especially those in the distance. Too, the difference in performance between a model trained on the Caltech Pedestrian data and a model trained on the more robust ECP data shows the necessity for training any pedestrian detection model on a dataset with a variety of domain applications and diverse scenarios and conditions.

Future endeavors should entail a more in-depth analysis of the ground truth data, specifically focusing on the evaluation of unevenly represented label classes within the model. Achieving a uniform representation of the ontology of ground truth data among different classes and across various biomes is essential for ongoing model training aimed at continuous performance improvement. Once a consistent level of precision is attained for each pedestrian class, the potential applications of this technology could extend to areas such as vehicles and walkway monitoring,

For specific applications to campus safety, universities could contribute by gathering data representative of their location, events, and population for future training.

VIII. Members' Contribution

- Dominic Scerbo:
  - Converted Caltech videos to images with respective labels.
  - Converted raw data annotation schemas to YOLOv8 and DETR label formats.
  - Trained YOLOv8 on ECP data.
  - Trained DETR on Caltech and ECP data.
  - Sourced and applied an open-source Streamlit application to demonstrate an inference workflow on test data with YOLOv8.
  - Created a function to demonstrate an inference workflow on test data with DETR.
- Suraj Kunthu:
  - Data Analysis of Caltech Pedestrian and ECP Datasets
  - Explored video output of pedestrian detection system
- Abigail Snyder:
  - Explored options for model and data sharing using Rivanna and other platforms
  - Trained YOLOv8 on Caltech data
  - Organized report and presentation

A. GitHub Repository

https://github.com/dscer/ds6050-group1-project

References

[1] Governor's Highway Safety Association. Pedestrian Traffic Fatalities by State: 2022 Preliminary Data, 2023, https://www.ghsa.org/resources/Pedestrians23

[2] Doulamis, Anastasios and Tian, Di and Han, Yi and Wang, Biyao and Guan, Tian and Wei, Wei. A Review of Intelligent Driving Pedestrian Detection Based on Deep Learning. Computational Intelligence and Neuroscience, 2021,https://doi.org/10.1155/2021/5410049.

[3] Xiau, Yanqui, Zhou, Kun, Cui, Guangzhen, Jia, Lianhui, Fang, Zhanpeng, Yang, Xianchao, and Xia, Quionpei. Deep learning for occluded and multi-scale pedestrian detection: A review. 2020, https://doi.org/10.1049/ipr2.12042.

[4] Real-time Pedestrian Detection: Layered Object Recognition System for Pedestrian Collision Sensing. U.S. Department of Transportation, Federal Highway Administration, 2010.

[5] Sahin, Emir. A Robust Pedestrian and Cyclist Detection Method Using Thermal Images; Psychological Ownership: A Study of Individuals and "Their" Autonomous Mobility. University of Virginia, School of Engineering and Applied Science, BS (Bachelor of Science), 2021, 2021, doi.org/10.18130/xdpm-h137.

[6] Zeng, Xingyu, Ouyang, Wanli, and Wang, Xiaogang. Multi-Stage Contextual Deep Learning for Pedestrian Detection. The Chinese University of Hong Kong, Department of Electronic Engineering, 2013, https://openaccess.thecvf.com/content_iccv_2013/papers/Ouyang_Joint Deep_Learning_2013_ICCpaper.pdf

[7] Yuan, Jing, Barmpoutis, Panagiotis, Stathaki, Tania. Effectiveness of Vision Transformer for Fast and Accurate Single-Stage Pedestrian Detection. Imperial College London, Department of Electric and Electronic Engineering, 2022. https://papers.nips.cc/paper_files/paper/2022/file/afb8caec018d3c8f6 ef8b81fa52386fe-Paper-Conference.pdf

[8] Boesch, Gaudenz. Vision Transformers (ViT) in Image Recognition – 2023 Guide. visio.ai, 2023. https://viso.ai/deep-learning/vision-transformer-vit/

[9] Wu, Yuxin, Kirillov, Alexander and Massa, Francisco, Lo, Wan-Yen, Girshick, Ross. Detectron2. Facebook Research, 2019. https://github.com/facebookresearch/detectron2

[10] Xu, C. et al. (2020) Fast vehicle and pedestrian detection using improved mask R-CNN, Mathematical Problems in Engineering. https://www.hindawi.com/journals/mpe/2020/5761414/ (Accessed: 30 September 2023).

[11] SuperAnnotate (2023) Introduction to the coco dataset, OpenCV. https://opencv.org/blog/2021/10/12/introduction-to-the-coco-dataset/ (Accessed: 30 September 2023).

[12] Papers with code - coco dataset (no date) COCO Dataset — Papers With Code. https://paperswithcode.com/dataset/coco (Accessed: 30 September 2023).

[13] Papers with code - caltech pedestrian dataset dataset (no date) Dataset — Papers With Code. https://paperswithcode.com/dataset/caltech-pedestrian-dataset (Accessed: 30 September 2023).

[14] Papers with code - eurocity persons dataset (no date) Dataset — Papers With Code. Available at: https://paperswithcode.com/dataset/eurocity-persons (Accessed: 30 September 2023).

[15] Mehra, A. (2023, July 11). Evolution of yolo object detection model from V5 to V8. Labellerr. https://www.labellerr.com/blog/evolution-of-yolo-object-detection-model-from-v5-to-v8/