15.773 Hands-on Deep Learning
March 11, 2024

Pedestrian and Wheelchair Dash-Cam Detection using Computer Vision

Anthony Khaiat, Jeremy Michael, Ethan Fahimi, Jad Makki

Contents

1 Problem Description

1.1 Background

Dash-cam object detection has emerged as a critical area of research in computer vision (CV), particularly motivated for enhancing road safety and lowering traffic-related accidents. Despite advancements in automotive safety features, roadside safety remains a global concern. According to the World Health Organization (WHO), road traffic injuries caused an estimated 1.35 million deaths worldwide in recent years, with pedestrians constituting a substantial portion of these fatalities. In urban environments, where vehicular and pedestrian paths frequently intersect, the risk of accidents is particularly large. A CV approach to dashcam object detection promises to significantly mitigate these risks by improving accuracy and speed recognition in real-time, thereby enabling advanced driver-assistance systems and autonomous vehicles to make safer navigational decisions. This project aims to develop a robust deep learning model that can efficiently detect objects such as cars, trucks, pedestrians, etc in diverse and challenging scenarios, such as varying lighting conditions, or crowded scenes.

Additionally, in an effort to make autonomous driving object detection models more robust and equitable, a model specifically designed to be capable of detecting wheelchair-bound pedestrians could be an improvement over current models. This would not only enhance the inclusivity of safety measures but also address a critical gap in the current pedestrian detection systems. Wheelchair-bound pedestrians are often at a greater risk in traffic environments due to their lower profile and different movement patterns compared to other pedestrians. Traditional models may not always effectively recognize these differences, leading to potential safety hazards. By incorporating data specifically featuring wheelchair users in roadside and traffic related scenarios, the proposed model aims to improve detection rates and reduce the likelihood of accidents involving this vulnerable group.

The development of this model involves leveraging advanced deep learning techniques and datasets enriched with wheelchair-bound pedestrian imagery to train algorithms to identify these individuals accurately. It also entails rigorous testing in various scenarios to ensure the model's effectiveness across different environmental conditions and urban settings. Overall, by aiming to create a more inclusive and equitable dash-cam object detection system, this project not only addresses a significant safety concern but also contributes to the broader discourse on responsible AI development and the role of technology in fostering a safer, more inclusive society.

1.2 Data

The first dataset utilized in this project is taken from Kaggle and is tailored for self-driving car applications. This dataset consists of high quality images taken from a vehicle's dash-cam, with examples shown in figures 1.1(a) and 1.1(b).

(a) Sunset Dash-cam Image

(b) Midday Dash-cam Image

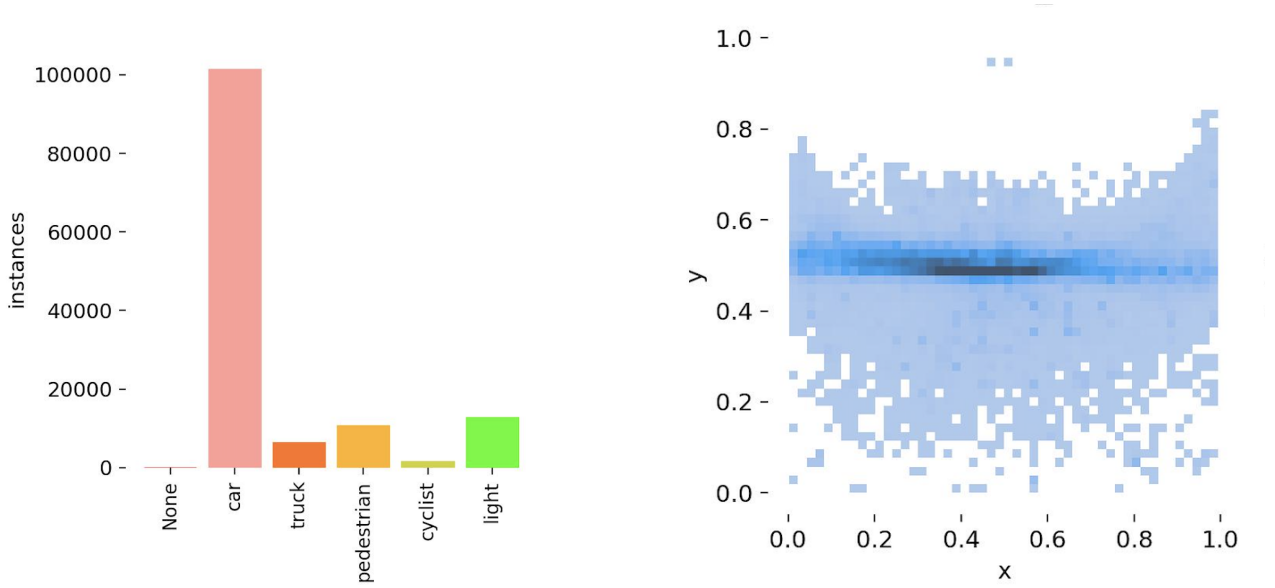
The dataset is designed to facilitate the development of object detection algorithms, which are critical for the perception systems of autonomous vehicles. This data contains roughly 100,000 images with annotations of pedestrians, cars, trucks, traffic lights, and bikers. In addition to the class labels for each item found in the image, the dataset includes the object's respective bounding box coordinates.

We supplement this dataset by web-scraping 122 images of "wheelchair crossing road" using Google Images and the Python Selenium library. The images in this external dataset were manually filtered to remove any inapplicable images (such as logos and poor quality images). We then labelled bounding boxes manually around each wheelchair in our images to feed to the object detection model. Figure 1.1 shows an example of one of the web-scraped images.

Furthermore, visualizations of the class distribution within our dataset can be found below, in Figure 1.2, which reveals that cars are the majority class of objects observed in the dataset. Furthermore, we can

Figure 1.1: Wheelchair Dataset Example

also observe from 1.2 (b) that a majority of the objects occur within the driver's peripheral view, and thus should be more easily detectable by the model.



(a) Distribution of Labeled Categories in Dashcam Dataset (b) Distribution of Bounding Box Locations in Dashcam Dataset

Figure 1.2: Visualization of Dataset

2 Methods

2.1 Pre-processing

The pre-processing stage begins with querying and reshaping the dataset to align with the specific objectives of enhancing pedestrian detection accuracy. The images are resized to a uniform dimension of 650×650 pixels, an essential step for the images to fit the neural network's input requirements.

Next, each image is split into three channels, representing the red, green, and blue (RGB) spectrums, capturing the full range of visual information necessary for effective learning. The dataset is then divided into a training set, comprising 80% of the data, and a test set, containing the remaining 20% for evaluation. This distribution is crucial for assessing the model's ability to generalize on unseen data.

2.2 Baseline Model

In order to establish a baseline performance metric for this task, we utilize the YOLOv8-nano model architecture, while disregarding the pre-trained weights of the model. As such, we train the model from scratch

to specifically tailor it to dash-cam footage. The YOLOv8 model employs a rigorous architecture, consisting of a backbone, which employs 53 convolutional layers and utilizes cross-stage partial connections to motivate the network to learn unique features of objects prior to the merge of the convolutional layers, a neck, which incorporates high level semantic features with low-level spatial information to improve detection accuracy for small objects, and a head, which extracts features by merging feature maps from different stages of the backbone and employs a self-attention mechanism that allows the model to focus more on different features depending on their relevance to the designated task. The use of the YOLOv8-nano architecture is also desirable due to its computational efficiency, as the nano model is optimized for speed, as the architecture's backbone prevents excess redundancy between features learned among different layers. Based on prior performance, this model is a strong candidate for a baseline model in this task. Applying this model directly to our dataset allows us to evaluate further enhancements. This model was trained for 15 epochs.

2.3 Transfer Learning

In addition to the baseline model, which is only trained on the dash-cam footage, we evaluate the effect of using pre-trained weights for the model, and fine-tuning these weights instead of starting from scratch. We hypothesize that fine-tuning the pre-trained weights will improve the model performance since it will be able to leverage a larger, more diverse set of data from pre-training, which will allow the model to learn more complex features of objects, and thus be more capable of identifying them at different angles, including when the objects are further away, and thus smaller, in the dash-cam image. For this task, we utilize both the YOLOv8-nano model, which omits convolutional layers in order to prioritize speed, and the YOLOv8-small model, which is slightly more computationally expensive, but utilizes a more complex architecture.

2.4 Wheelchair Model

Addressing the vital need for detecting wheelchair-bound pedestrians, we fine-tune the best performing transfer-learning model on the dataset with manually labeled wheelchair user images. Incorporating this data will enhance the YOLOv8 model's ability to identify wheelchairs in various environments, and will thus help serve as a more inclusive object detection algorithm for autonomous vehicles.

3 Results

3.1 Comparison of Models on Dashcam Dataset

We analyze the performance of the above models on a holdout set, consisting of roughly 20,000 images. The results are summarized in Table 3.1 below:

Model	Acc % (Car)	Acc % (Truck)	Acc % (Pedestrian)	Acc % (Cyclist)	Acc % (Lights)
Baseline	72	47	23	13	48
TL (nano)	77	53	43	45	63
TL (small)	82	69	54	56	69

Table 3.1: Model Performances on Dash-Cam Holdout Set

3.1.1 General Observations

Looking at the table, all three models demonstrate an impressive capacity for detecting cars in comparison to the other objects present in the data. If we examine further, this behavior is a result of class imbalance as cars are present in almost all of the images in our dataset, as can be observed from Figure 1.2. Even without pre-training, the YOLOv8-nano model is capable of learning the features of cars more easily than features of other objects. Furthermore, we observe that pedestrians and cyclists are particularly hard to identify, which is rather concerning as they are the most vulnerable when on the road. The lack of accuracy for these classes can also be explained by the frequency in which they appear in the dataset, as they are the two least frequent classes in the training data. This is understandable, as dash-cam footage can encompass a variety of locations where there are no pedestrians or cyclists, such as highways or rural areas. Cyclists and pedestrians are more frequent in big cities, where streets are typically crowded, and the majority of the images in our dataset come from suburban areas or the freeway, where pedestrians and bikers are less likely to share the road with vehicles. As such, these models, particularly the baseline model, are unable to learn with high confidence the features of pedestrians or cyclists.

3.1.2 Interpretation of Results

Another notable observation upon looking at Table 3.1 is that as we increase the complexity of the YOLOv8 model, the model's capacity for object detection in all classes improves. Going from the baseline model, which simply uses the raw YOLOv8-Nano architecture trained on the dash-cam data, to the fine-tuned YOLOv8-Nano model which was previously trained on the COCO dataset, we added another layer of complexity to the model as the pre-trained model already had the capacity to learn complex features of objects, whereas

the baseline model had to learn this from scratch. With this increased complexity, we can observe, on average, a 16% increase in class accuracy going from the baseline model to the pre-trained model. This is especially driven by improvements in the model's ability to detect pedestrians and cyclists, which the baseline model really struggles to detect, since it has no prior basis for knowing how to identify these object classes. Introducing pre-trained weights improves performance on pedestrian detection by 20%, and detection of cyclists by 32%. Although the overall performance on these classes is nowhere near state of the art, the sizable improvements highlight that using pre-trained models provide a notable head-start for the model's ability to learn complex features of certain objects, and in a task such as object detection from dash-cam footage, this becomes increasingly important as public safety is of concern. In addition to the large improvements in detecting cyclists and pedestrians, the pre-trained and fine-tuned model also demonstrate impressive improvement in detecting traffic lights, with a 15% increase in accuracy, and good improvements in detecting cars and trucks.

Adding another layer of complexity, the YOLOv8 small model expands upon the nano model by adding more layers and parameters. As such, the expectation is that this model is capable of learning more complex features of objects, improving upon the accuracy on the nano model, which is optimized for computational speed over accuracy. The results support this expectation, and we can see impressive improvements across all classes, with an average class improvement of 9.8% from the nano model. The primary classes in which we see improvement are with detecting trucks, where we see a 16% improvement compared to the nano model. This improvement is driven by the addition of the extra parameters in the small model, which allows the model to learn more given the same data, and can thus distinguish trucks from cars more easily, which improves model performance on both of these classes. Furthermore, we again observe impressive improvements for detecting pedestrians and cyclists, with 11% improvement in accuracy for both classes. This is important to note, as it proves that even though the models are trained on the same data, both during pre-training and fine-tuning, the added layers of the YOLOv8 small model allow it to learn more complex associations - such as how a pedestrian or car might look when it is far away - than the nano model has the capacity for, and therefore leads to increased improvements even when there is little data for a particular object. We show further accuracy metrics, such as F1-Confidence curve, Precision-Confidence Curve, Precision-Recall, and Recall-Cinfidence Curve, in section 5.

3.1.3 Visualization of Results

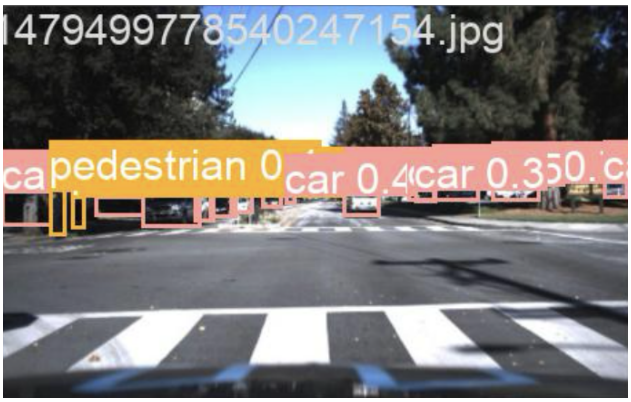
In this section, we will provide a demonstration on the output of these models based on fixed dash-cam images, and analyze the outputs directly. These outputs are depicted below, in figure 5, where we can observe each model's object assignments on the same image in comparison with the true value. From these

visualizations, we can observe that the baseline model is unable to detect either of the pedestrians in the image, and is only capable of detecting the cars, although the bounding boxes assigned to the cars are slightly off. As such, the baseline model tends to assign multiple labels of a car to the same car. The pedestrians in this image are small and obscured, as they are far away and in a dark environment, and because the baseline model tunes the raw weights of the YOLOv8 nano-model with no pre-training, the lack of context for understanding the features which identify a pedestrian explains the model's poor performance.

(a) Labelled Dash-cam Image - True Value

(b) Baseline Model Assignments

(c) YOLOv8 Nano Fine-Tuned Model Assignments



(d) YOLOv8 Small Fine-Tuned Model Assignments

Figure 3.1: Comparison of Model Outputs to True Value

In stark comparison to the baseline model, both YOLOv8 models are able to detect the two pedestrians in the image, with relatively accurate bounding boxes. They also prove capable of detecting the cars in the image, revealing that utilization of the pre-trained weights ultimately proves beneficial for this task as the models can differentiate relatively accurately between the pedestrians and vehicles for this example. The spatial accuracy of the bounding boxes is also impressive, which is also attributable to the fact that these models have more information about how to discern between different objects due to the fact that they are pre-trained, and thus demonstrate important improvements over the baseline model.

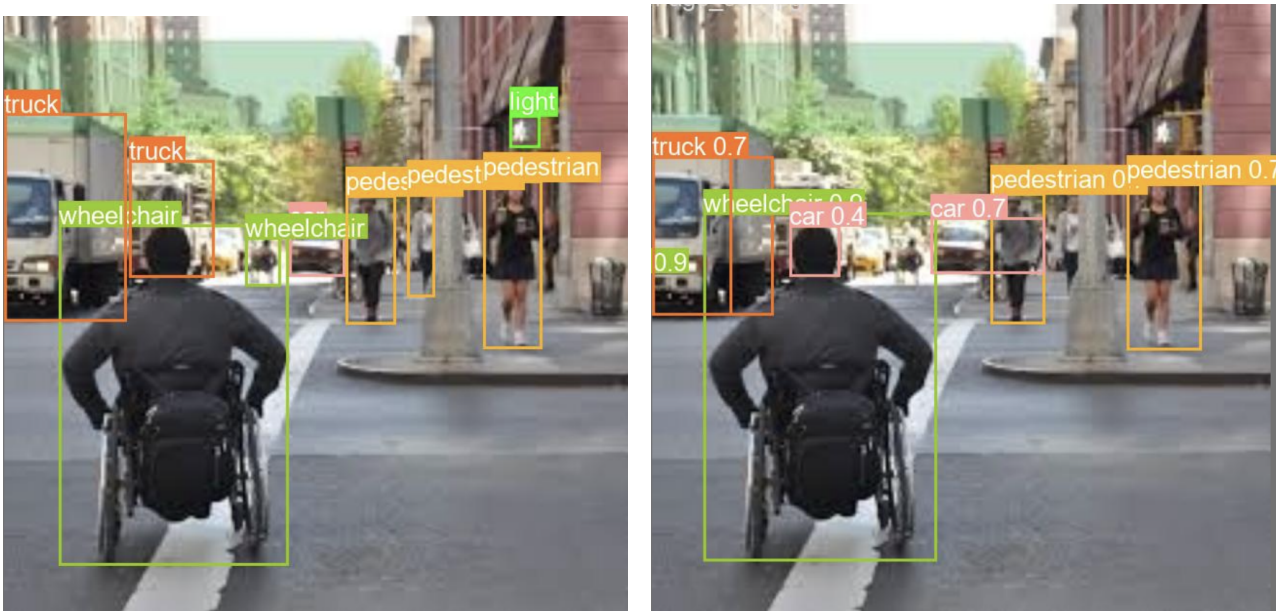
3.2 Wheelchair Model

In order to make these object detection models more inclusive and capable of discerning between different categories of pedestrians, we further fine-tune the YOLOv8-small model on the web-scraped wheelchair dataset, and evaluate the results of such a model in table 3.2 below:

Acc (Wheelchair) %	Acc % (Car)	Acc % (Truck)	Acc % (Pedestrian)	Acc % (Cyclist)	Acc % (Lights)
89	82	50	33	N/A	80

Table 3.2: Fine-Tuned YOLOv8-small Performance on Wheelchair Data

We note that, in the absence of fine-tuning, the YOLOv8 model is unable to detect wheelchairs at all in an image, and thus, despite having a small dataset, the model proves to be capable of both classifying wheelchair objects while also discerning between wheelchairs and other objects, such as cars. This again speaks to the importance of using the inductive bias from pre-trained models, as it allows the model to learn new associations more easily when it has this baseline. We also note that in our wheelchair dataset, there are no cyclists, as such there are no predictions of this object class on this data. An example of this model in action is shown Figure 3.2 for reference:



(a) Labelled Image

(b) YOLOv8-small Object Assignments

Figure 3.2: Comparison of Labelled Inputs and Model Outputs for Wheelchair Detection

Looking at the predictions from our fine-tuned model, the model is able to detect the wheelchair nearest to the image; however, it misses the wheelchair user who is further down the street. This can be attributed to the fact that the model does not have enough data on wheelchairs to be able to learn to detect them when they

are in the distance within the image, and thus appear smaller. Future work would involve obtaining access to more data such that the model can learn to make these associations, and be capable of detecting wheelchair-users even when they are further away in the frame. We observe that the model is still able to discern between wheelchair users and other objects, such as vehicles and non-wheelchair pedestrians. Overall, considering the fact that the YOLOv8-small model did not previously have the capacity for wheelchair detection and that the wheelchair only contained 122 images, this performance is impressive and promises to be a valuable addition to real-time object-detection algorithms given a larger dataset. We show other accuracy metrics, such as F1-Confidence curve, Precision-Confidence Curve, Precision-Recall, and Recall-Confidence Curve, in section 5.

4 Lessons Learned

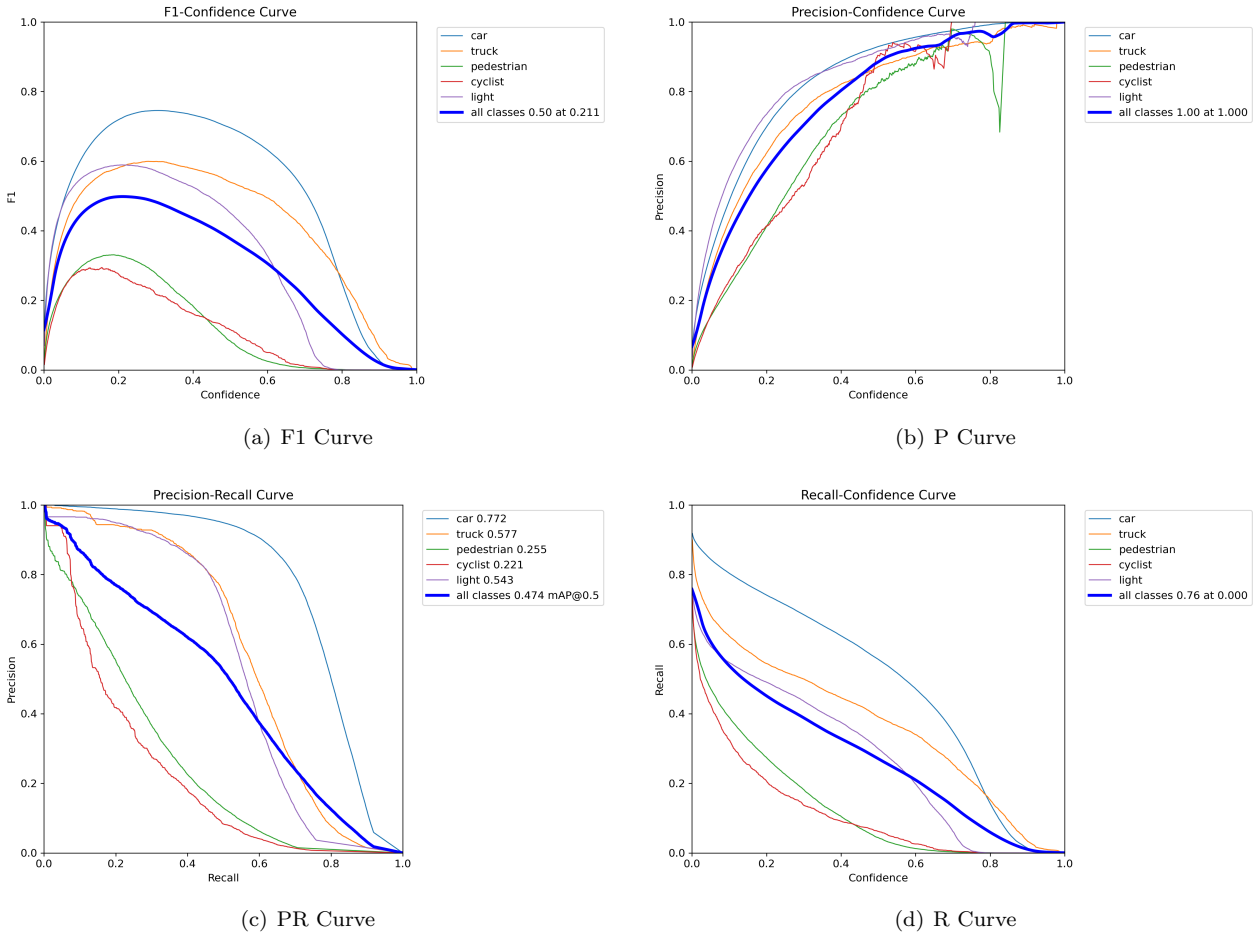
4.1 Conclusions

In this project, we began with a simpler model that utilizes the YOLOv8-nano architecture only trained on the existing dataset, this discarded the pre-trained weights of the model. We then incorporated these pre-trained weights from the COCO dataset, and subsequently introduced additional layers to the model, making it wider and deeper. Through evaluating these models against each other, this project underscores the importance of utilizing the inductive biases inherent in state-of-art models via transfer learning. Allowing the models to use their prior knowledge ultimately drove an increase in performance, which speaks to the importance of fine-tuning models, which is both easier and less resource intensive than programming these models from scratch. Furthermore, incorporating the YOLOv8-small model, which introduces additional layers to the nano-model, indicates that the trade-off between computational efficiency and accuracy is an important consideration with respect to object detection tasks, as we were able to improve accuracy at an increased computational burden. In real-time, these models need to have the right balance between efficiency and accuracy, especially when considering public safety implications. Moreover, while adjustments made to this model to recognize wheelchairs showed encouraging outcomes, the modest size of the specifically curated wheelchair dataset meant such results were anticipated. Expanding this dataset would significantly enhance the model's performance, contributing to the development of more reliable and intrinsically safer autonomous driving technologies.

4.2 Future Work

In the future, we look to obtain a better quality images on wheelchairs in urban areas along with more images. This can be done in two ways: scraping websites such as Google Images and Pexels for more images, then filtering those images with either a CV model or human labelers, or obtaining the rights to use a dash cam company's footage data. Additionally, we would like to create a system that works from a single video input, ideally live video for use in actual autonomous driving scenarios. While this project serves as a good baseline for improving autonomous vehicle safety mechanisms, it is imperative that future work consider the implications of this model with respect to live footage, where vehicles are constantly moving and the object detection algorithm is in constant use.

5 Appendix



(a) F1 Curve

(b) P Curve

(c) PR Curve

(d) R Curve

Figure 5.1: Performance Metrics of YOLOv8 Nano Model with No Pre-training

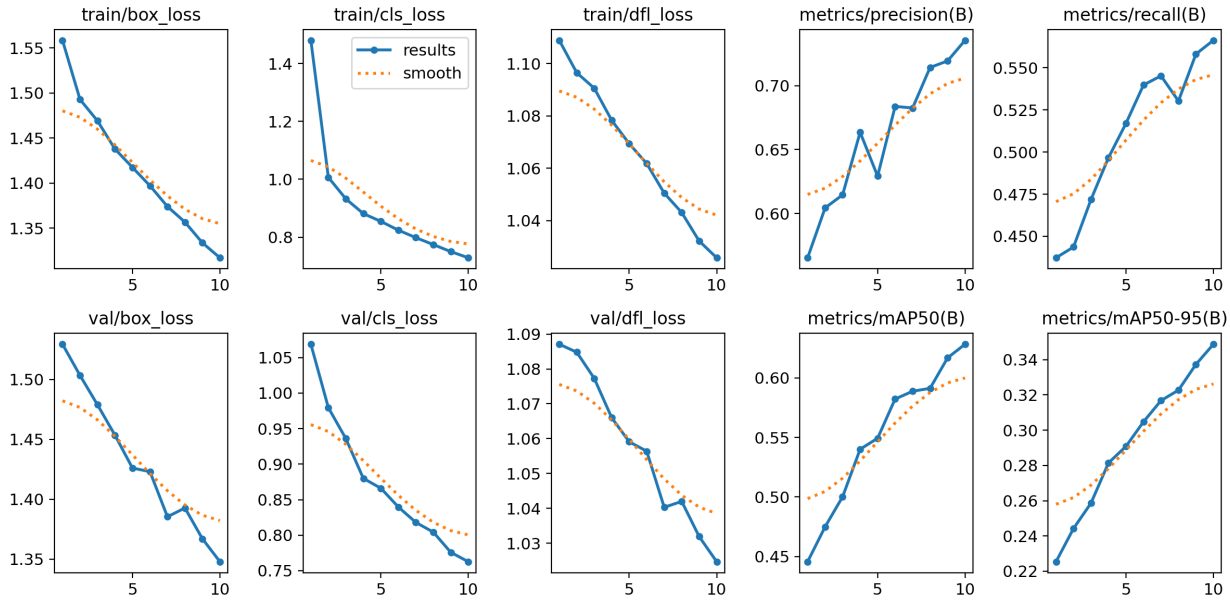
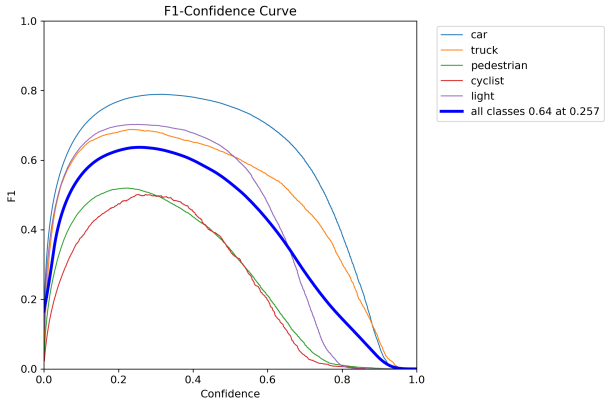
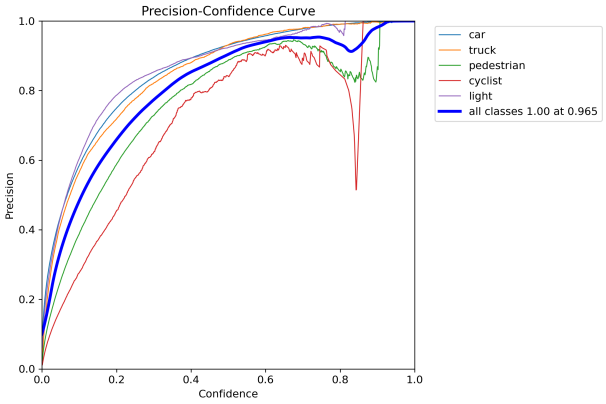


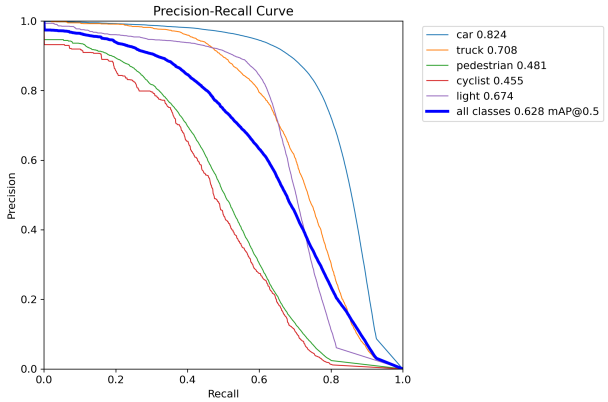
Figure 5.2: Results of YOLOv8 Nano Model Trained on COCO Dataset



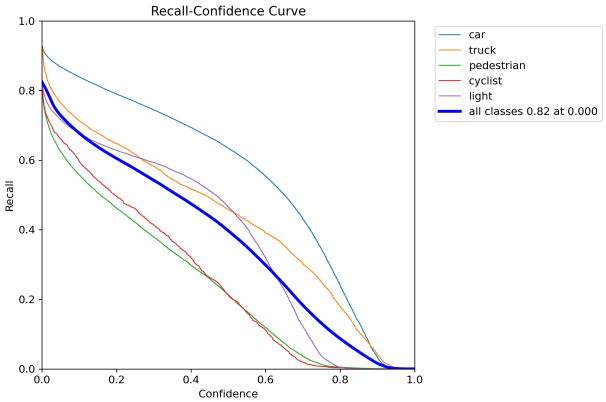
(a) F1 Curve



(b) P Curve



(c) PR Curve



(d) R Curve

Figure 5.3: Performance Metrics of YOLOv8 Nano Model Trained on COCO Dataset

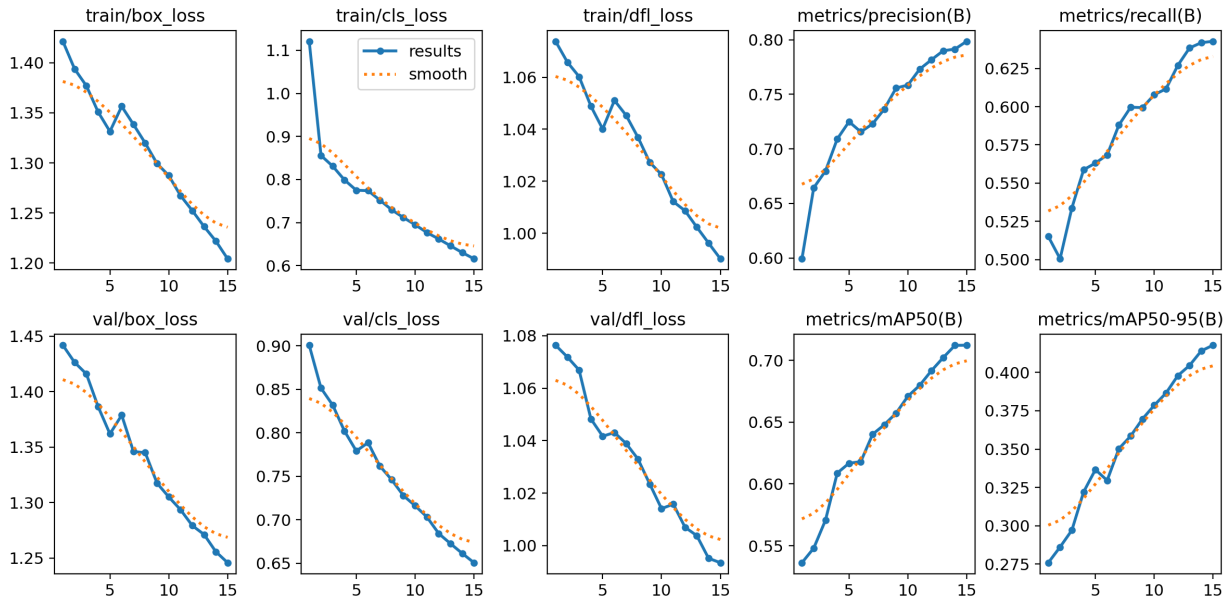
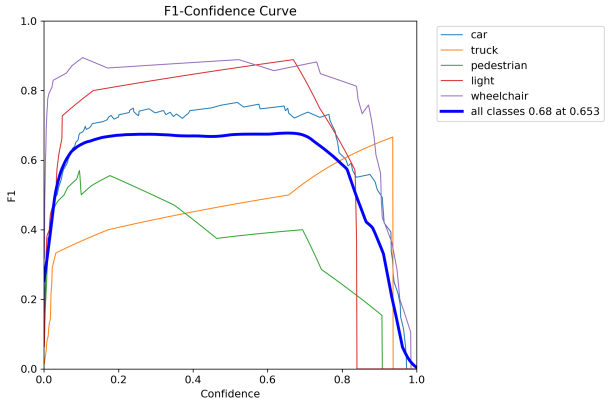
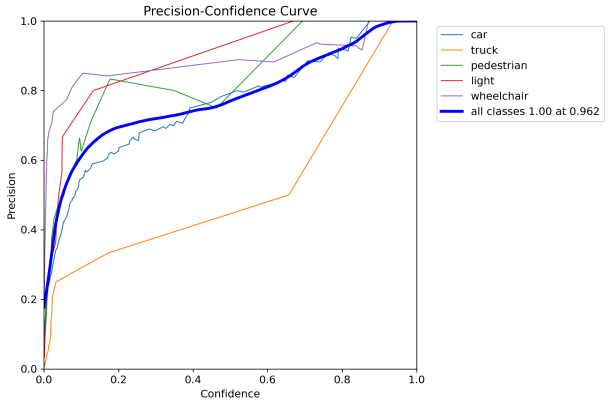


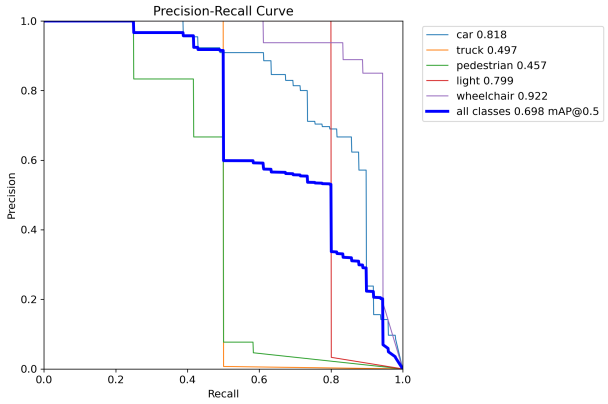
Figure 5.4: Results of YOLOv8 Small Model Trained on COCO Dataset



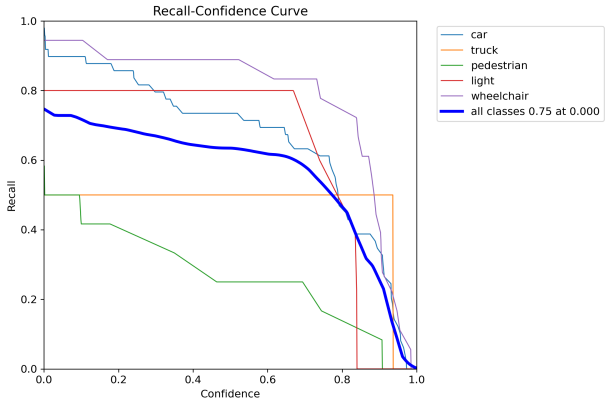
(a) F1 Curve



(b) P Curve



(c) PR Curve



(d) R Curve

Figure 5.5: Performance Metrics of YOLOv8 Small Model Trained on COCO Dataset with Fine-Tuning on Wheelchair Dataset