KTH ROYAL INSTITUTE
OF TECHNOLOGY

DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
FIRST CYCLE, 15 CREDITS

Investigation regarding the Performance of YOLOv8 in Pedestrian Detection

ADAM SJÖBERG, JONATAN HYBERG

Stockholm, Sweden 2023

Investigation regarding the Performance of YOLOv8 in Pedestrian Detection

ADAM SJÖBERG, JONATAN HYBERG

Degree Project in Computer Science and Engineering, First Cycle 15 credits
Date: June 10, 2023
Supervisor: Erik Fransén
Examiner: Pawel Herman
Swedish title: Undersökning angående YOLOv8s prestanda att detektera fotgängare
School of Electrical Engineering and Computer Science

Abstract

Autonomous cars have become a trending topic as cars become better and better at driving autonomously. One of the big changes that have allowed autonomous cars to progress is the improvements in machine learning. Machine learning has made autonomous cars able to detect and react to obstacles on the road in real time. Like in all machine learning, there exists no solution that works better than all others, each solution has different strengths and weaknesses. That is why this study has tried to find the strengths and weaknesses of the object detector You Only Look Once v8 (YOLOv8) in autonomous cars. YOLOv8 was tested for how fast and accurately it could detect pedestrians in traffic in normal daylight images and light-augmented images.

The trained YOLOv8 model was able to learn to detect pedestrians at high accuracy on daylight images, with the model achieving a mean Average Precision 50 (mAP50) of 0.874 with a Frames per second (FPS) of 67. Finally, the model struggled especially when the images got darker which means that the YOLOv8 in the current stage might not be good as the main detector for autonomous cars, as the detector loses accuracy at night. More tests with other datasets are needed to find all strengths and weaknesses of YOLOv8.

Keywords

Autonomous Cars, Machine Learning, YOLOv8, Pedestrian Detection

Sammanfattning

Autonoma bilar har blivit ett trendigt ämne då bilar blir bättre och bättre på att köra självständigt. En av de stora förändringarna som har gjort det möjligt för autonoma bilar att utvecklas är framstegen inom maskininlärning. Maskininlärning har gjort att autonoma bilar kan upptäcka och reagera på hinder på vägen i realtid. Som i all maskininlärning finns det ingen lösning som fungerar bättre än alla andra, varje lösning har olika styrkor och svagheter. Det är därför den här studien har försökt hitta styrkorna och svagheterna hos objektdetektorn You Only Look Once v8 (YOLOv8) i autonoma bilar. YOLOv8 testades för hur snabbt och precist den kunde upptäcka fotgängare i bilder av trafiken i dagsljus och bilder där ljuset har förändrat.

Den tränade YOLOv8-modellen kunde lära sig att upptäcka fotgängare med hög noggrannhet på bilder i dagsljus, där modellen uppnådde en genomsnittlig medelprecision 50 (mAP50) på 0,874 med en antal bilder per sekund (FPS) på 67. Modellen hade särskilt svårt när bilderna blev mörkare vilket gör att YOLOv8 i det aktuella stadiet kanske inte är tillräckligt bra som huvuddetektor för autonoma bilar, eftersom detektorn tappar noggrannhet på mörkare bilder. Fler tester med andra datauppsättningar behövs för att hitta alla styrkor och svagheter med YOLOv8.

Contents

1 Introduction

In the last couple of years, self-driving cars have become a trending topic both in society and within car manufacturers. Companies such as Google, Tesla, and several others are all trying to create autonomous vehicles. The current revolution in self-driving is in some parts due to the current progress in machine learning (ML). More specifically it is better object identification algorithms that have allowed cars to recognize threats in traffic and then respond accurately in real-time[1].

Currently, there are a lot of different object detection systems that exist. Some examples of these detection systems are: You only look once (YOLO), Region-based Convolutional Neural Networks, and Histogram of oriented gradients[2]. These detection systems have different strengths and weaknesses, depending on what they prioritize. Some systems prioritize the speed of detection, while others prioritize the accuracy of the object predictions. These two factors are often in a balancing act if the detection system becomes faster the accuracy often drops. Thus to solve an object detection system like self-driving cars, one must weigh these systems against each other and pick the one that fits the problem best.

To compare these detection systems, one needs data from previous experiments. That is why this report has presented the results of an experiment with the object detection system YOLO. The experiment tests how good the latest version of YOLO from Ultralytics is, specifically the model YOLOv8[3]. The two main criteria that the experiment has focused on are the precision and speed of the model, which also leads to the goal of the experiment. The goal is to see if the YOLOv8 object detection algorithm is fast and precise enough to be used in self-driving cars. The report will also discuss the advantages and disadvantages of the YOLOv8 object detection algorithm.

1.1 Problem

The research questions that have been examined are:
How well does YOLOv8 perform at detecting humans in traffic images, in terms of speed and accuracy?
How does the brightness of the images affect the precision of the YOLO v8 model?

1.2 Scope

There are multiple ways to test the object detection system, this report limits the test to only still images of traffic. This will affect the final results compared to using videos, because each frame in a video is similar to the previous frame, while each photo can differ drastically from the others. Additionally, the speed of an object detection model is affected by the hardware used, this effect will not be measured as this experiment will only use one computer for every experiment.

2 Background

2.1 Glossary

Hyperparameter: A hyperparameter is a parameter that fine-tunes the neural network model.
Transfer learning: Transfer learning in machine learning where a pre-trained model adapts the new task. This is useful when there is a limited amount of training data.
Overfitting: Overfitting is when the model fits exactly to the training data, so the model cannot predict on new data
Underfitting: Underfitting is when the model does not learn from the training data.
Generalization: The model's ability to handle new datasets and new environments.
Epoch: One Epoch is a single pass-through of the algorithm on the training data, during training.
Frames Per Second (FPS): FPS is how many images the model can process per second.

2.2 Object detection

Object detection is a computer technology that given an image can recognize and label all objects to a set of predetermined classes [4]. These classes are determined by the creator of the detector and can be for example "Human", "Car", or "Traffic lights". There are in general two different detection techniques, artificial neural network (ANN) and non-neural network. One of the currently most common ANN detection techniques is convolutional neural networks (CNN).

A CNN detector is created by making a model algorithm of the system. Then the model is trained on a set of images to learn to recognize the objects and lastly evaluating the model on a set of images that contain objects the system is designed to detect. These sets of images are called datasets and they are split into three smaller sets: training set, validation set, and test set [5]. The training set is the images the model has been trained on. The validation set is the set of images in

which the trained model is tested to see that the model works as intended. Finally, the test set is used to test the finished model on new images.

The basic idea of how a detection system detects an object is built up of two parts. The first part is drawing bounding boxes (x,y coordinate, and height and width) around all objects on the image. The second part is predicting what class of object can be inside of this bounding box, this part is done by sending the bounding boxes to a CNN model. Each predicted class has a confidence score, which is how certain the CNN model is that the class is true. If the confidence score is too low then the prediction is removed. The bounding boxes plus the classes together become the system's predictions on the starting image[6].

Figure 2.1: Example of a prediction of this paper's YOLOv8 detector

2.3 Evaluation

To decide if a predicted bounding box is correct the standard is to use the metric Intersect over Union(IoU) [7]. IoU is defined as the overlap between the predicted bounding box and the ground-truth box that can be seen in Figure 2.2. The ground-truth box is the correct area in an image where the object is, which is defined as equation 1.

$$IoU = \frac{area\ that\ overlaps}{area\ over\ union} \tag{1}$$

The score is equal to one if the two boxes overlap exactly and zero if there is no overlap. In this report, a prediction is true if it has an $IoU \geq 0.5$ and that is called True positive. If $IoU < 0.5$ it is called a false positive.

Figure 2.2: Example of ground truth and the predicted bounding box. The green box is the ground truth and the red is the predicted bounding box

Two important metrics in object detection are precision and recall[8]. Precision is defined in equation 2.

$$Precision = \frac{True\ positive}{True\ positive + False\ positive} \tag{2}$$

This is the ratio of correct predictions to the total predictions and can be seen as how often the model is correct when predicting a class. The recall is defined in equation 3

$$Recall = \frac{true\ positives}{true\ positives + false\ negatives} \tag{3}$$

It is the ratio of correct predictions to the total amount of that class, so it is the percentage of objects that the detector found. False negatives are the objects that the detector does not find.

Precision and recall are often impractical when comparing different models because they do not give the whole picture. That is why the popular metric to measure the accuracy of object detection systems is the mean average precision (mAP). mAP is defined as the area under a detector's precision-recall curve. This report uses the standard COCO mean Average Precision 50 (mAP50)[8]. In mAP50 a prediction is correct if the $IoU \geq 0.5$ and if the prediction is the correct class. In mAP50 the area under the precision-recall curve is interpolated and uses 101 points along the curve to interpolate.

2.4 YOLO

The normal object detection systems are too slow to use in real-time detection because the system is split into two programs, which slows down the detection[6]. That is why YOLO was developed, which is an object detection system that can be run in real-time. The reason for this is that YOLO is a one-stage detection system which means that it only needs to be run once and can draw bounding boxes and predict at the same time.

The YOLO model works by first splitting the image into a grid with C grid cells [6]. Then every grid cell predicts B number of bounding boxes around all the features the cell thinks is an object. The model gives each bounding box a confidence score, which is the probability that the box contains an object. At the same time, the grid cell also predicts a multiclass probability for which classes can be in the cell. When both the bounding boxes and class probabilities are calculated, then they are added together, leaving a final bounding box and object class for that box. These last boxes become the model's prediction.

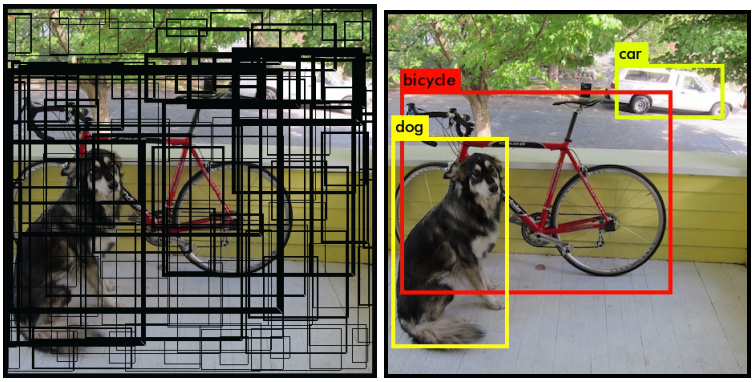

Figure 2.3: Calculating and finding bounding boxes via confidence score[9].

2.5 Previous studies

A previous study by Akshatha et al. experimented with detecting humans in aerial thermal images using ML[10]. The detection system they used was a regional Convolutional Neural Network (faster RCNN), which is a two-stage detection system. That means that the system first tries to find all regions with objects, then a separate system tries to identify what object is in those regions. The study used two datasets to test their model. The first dataset was Thermal Pedestrian Database from Ohio State University which according to the Akshatha et al. is the standard for testing thermal computer vision models. The second dataset is Thermal Pedestrian Database from Aalborg Universitets, which according to Akshatha et al. is a harder dataset compared to the first one. In Akshatha et al.'s experiment their faster RCNN model was able to get an mAP score of 1 with 8 FPS on the Ohio database. On the second dataset, the model was able to get an mAP score of 0.545 with 8 FPS.

Another previous study by Hsu and Lin researched YOLO for pedestrian detection[11]. The study managed to create a version of YOLO which was able to achieve an mAP of 0.916 at detecting pedestrians in the dataset VOC 2012 comp4, which is a high accuracy in pedestrian detection.

Further research that Lan et al. researched was with the model YOLO-R which is a new structure for of YOLO model[12]. The study also experimented with finding pedestrians and they manage to get a detection speed of 25 FPS. Lan et al. model had a precision was 0.986 and a recall of 0.912.

3 Materials

3.1 Roboflow

This report used the website Roboflow to store and process the dataset used in the experiments. Through Roboflow a dataset of traffic images was found that was used to create the dataset for the experiment. Additionally, Roboflow was employed to augment the images by changing their brightness. When the datasets were complete then we exported the final datasets in Roboflow's YOLOv8 format.

3.2 Original Dataset

The images that were used in the experiment were selected from the dataset "Object detection and localization Image Dataset" (original dataset) [13]. The dataset is a set of 3813 images of traffic situations in inner cities in North America. Furthermore, all the images were taken in daylight, but there they were taken during different seasons and weather conditions. For example, there were images taken in sunshine, snowstorms, and rain. Images in the database were labeled and used 11 different classes to label objects in the images. Three unlabeled example images can be seen in Figure 3.1.

There were a few objects in the original dataset that were incorrectly labeled for example, a car was labeled as a pedestrian in one of the images. The bounding boxes of these images were manually edited in Roboflow so that the boxes were correct.

Figure 3.1: Three unlabeled example images from the original dataset, top left is a sunny image, top right is a snowy image, and the bottom image is a rainy image[13].

3.3 Creating Dataset

For the experiment, a smaller dataset was created from the images in the original dataset. To create the smaller dataset 1011 images were selected from the original dataset. The images were selected so that around half of the images contained at least a pedestrian. For the experiment, the number of object classes in this paper's database was lowered from 11 to 1 class, namely pedestrians. The reason for this was that the experiment was to detect humans in traffic and the other classes were outside of the scope of the study. Out of the 1011 images in this dataset 637 images contained pedestrians and 374 images with no pedestrians.

This paper's dataset of 1011 images was split into three subsets, the first one is the training set containing 381 images, the second is the validation set with 331 images, and the last is the test set with 299 images. In Table 3.1 the ratios of pedestrian to non-pedestrian images in each dataset can be seen.

Table 3.1: The split between pedestrians and non-pedestrian pictures in the dataset

Dataset	Pedestrians	Non-pedestrians	Total
Train	379	2	381
Validation	189	142	331
Test	69	230	299

For the ratio of pedestrian to non-pedestrian images between the 3 datasets the experiment followed 2 standards. The first standard was that the training set should have 70% of the images, 20% for validation, and 10% for testing[14], and the second was that the training set should have between 0-10% negative samples[15]. A negative sample is an image without any class, and in this report that is a non-pedestrian image.

These two standards used conflict because if the training set has 0-10% non-pedestrian images and is 70% of the total dataset then the training set will contain all pedestrian images, which would invalidate the testing. That is why the training is limited to only 379 pedestrian images and 2 non-pedestrian images. The validation was split so that it had around 30% of all pedestrian images to see if the training was successful, if it had lower pedestrian images the results varied too much between runs. The test set had around 10% of the pedestrian images to match the first standard. The ratio of non-pedestrian images in the validation set and test set gave better results in validating the training and better testing results. The final dataset that was used can be found on Roboflow[16].

4 Methods

4.1 Tools

4.1.1 Ultralytics

This project uses Python with the package Ultralytics to train and test the detection models, the package is distributed by the company Ultralytics[17]. Furthermore, Ultralytics offers pre-trained YOLOv8 models with the Python package, and all of the models are trained on the 2017 COCO dataset[3]. This report used the detection model yolov8s.pt as the base model for all experiments, which is the second-smallest detection model that Ultralytics offers. This model was chosen because it is smaller, which makes the model faster to train.

4.1.2 Google colab

To run the experiments we used Google's colab built hosted runtime. During training and testing the GPU used was Nvidia's Tesla T4.

4.2 Training model

This report aims to discuss the advantages and disadvantages of the YOLOv8 object detection system in a self-driving car scenario. To achieve this a YOLOv8 model named the trained base model (TBM) was trained, based on the base model (BM) yolov8s.pt [17]. The TBM was trained using the transfer learning technique, with the training set as input. The training was done using Ultralytic's YOLOv8 python package's training mode. The TBM was trained using the hyperparameters: learning rate (Lr) of 0.01, image size(imgsz) of 1080, and 20 epochs. All the other parameters were left as default values.

The hyperparameters for training were chosen by training different models and picking the parameters that gave the best results in the validation test. The validation test was done using Ultralytic's validation mode with the validation set as the input. For validations, we used the hyperparameters: object confidence threshold for detection (conf) of 0.5, imgsz of 1080, and IoU of 0.5. Conf was chosen to be 0.5 because during training it gave the model the best mAP50 score.

YOLOv8's training function uses the three loss functions to evaluate how well the model is learning. In the training results section, the results of two of these will be presented, namely Box loss and Classification loss. Box loss is defined as the error between the predicted bounding box to the ground truth bounding box. Classification loss is the error between the predicted class and the true class of an object. A problem is that Ultralytics does not give out the exact loss functions used in their package.

4.3 Experiments

4.3.1 Evaluation

In this report, a prediction is true if it has an $IoU \geq 0.5$. Furthermore, to measure the accuracy of the models the metrics used were precision, recall, and mAP50. Definitions for these metrics can be found in the chapter evaluation in the background 2. The speed of the model was calculated as the average inference time per image, across all images in the test set. It is important to note that the experiments were only run once on both models.

The experiments were performed with the Ultralytics package function called validation mode. The hyperparameters used for in the tests were, conf of 0.5, imgsz of 1080, and IoU of 0.5. These parameters were consistent with the parameters used in the validation test.

4.3.2 Execution of experiments

The research question for this report is split into two questions. To answer the two questions two experiments were performed with the TBM and BM. The first experiment is designed to answer the first question “How fast and precise is the YOLO v8 to detect humans in traffic?”. To answer the first question, the first experiment tested the models on the traffic images in the test set. The experiment measured the models’ accuracy and speed on the images that the model had never seen before. The goal of the experiment is to see the statistics of the two models’ speed and accuracy.

The second experiment was designed to answer the second question “How does the brightness of the images affect the precision of the YOLO v8 model?”. The experiment does this by testing the model on the same images as in the first experiment however, now the images had light augmentations. The test images first had their light lowered to between 0-50%, this set is called “50% Darker”. Then the light was increased to between 0-50%, this set is called “50% Lighter”. These two sets aimed to simulate that the images were taken in the nighttime and bright sunlight.

The light augmentations for experiment two were done with Roboflow’s light augmentation tool. The tool works by either brightening or darkening every image in the dataset randomly between the percentage 0 and the chosen upper limit, which was 50% in these experiments. There was no way to remove the randomness from the Roboflow tool. Examples of how the dark and light augmentations look can be seen in the figures 4.1 and 4.2.

Figure 4.1: Example of one of the darker images, the left image is the original image and the right is 40% darker.

Figure 4.2: Example of one of the lighter images, the left image is the original image and the right is 41% lighter.

5 Results

In this section, we present the results of the training and the accuracy and speed of the BM and TBM in the two experiments. In the first experiment the images are from the test set and not augmented. In the second experiment, the images are from the same set as the first experiment, but now they are augmented by increasing or decreasing the brightness of the images.

5.1 Training

5.1.1 Loss

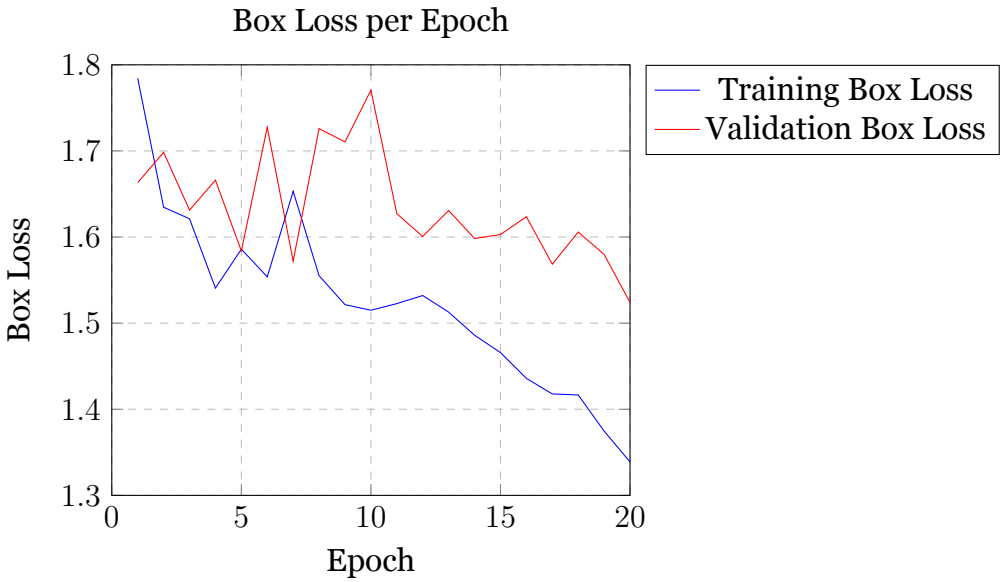

Figure 5.1: The box loss of the training model on the training set and validation set over the 20 training epochs.

In Figure 5.1 we can see that Training box loss is decreasing and does not plateau in the 20 epochs of training. The final value of training box loss at epoch 20 stops at 1.34. The Validation box loss increases in the first 10 epochs, only in the last 10 epochs does the function slowly decrease. The final value of the validation box loss value is 1.52.

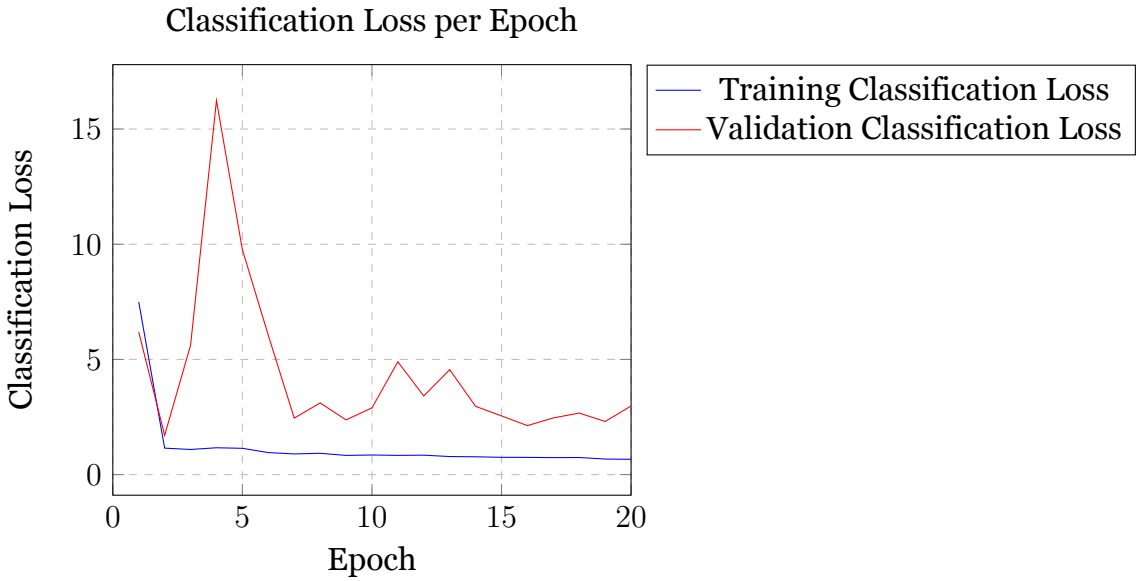

Figure 5.2: The classification loss of the training model on the training set and validation set over the 20 training epochs

Figure 5.2 shows that the model's classification loss on the training set reaches its limit after only one epoch. The reason for this is that dataset only has one class to predict, so the model only predicts that class. The classification loss on the validation set spikes at the fourth epoch and after that settles down and plateaus. A possible reason behind this spike can be that the BM knows many classes and has not yet unlearned them for this dataset.

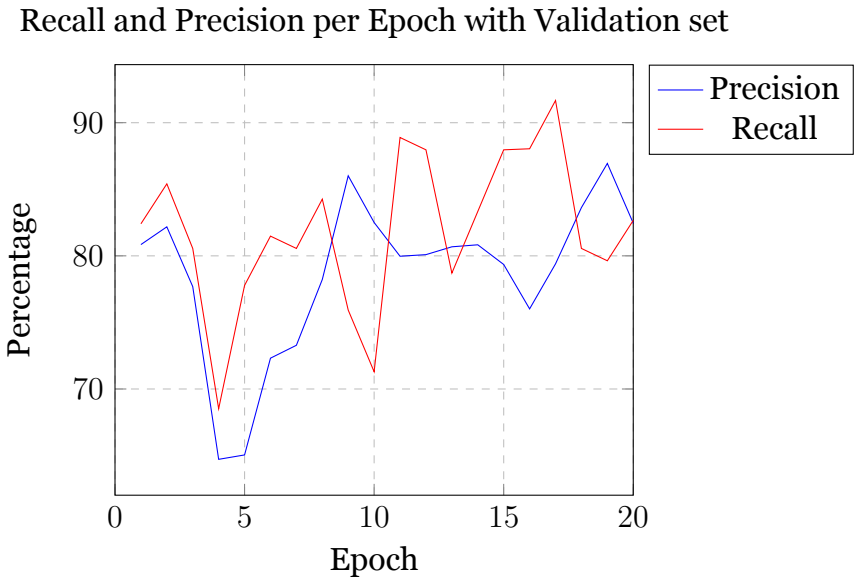

Figure 5.3: A diagram that presents the Precision per Epoch and the Recall per Epoch with Validation set

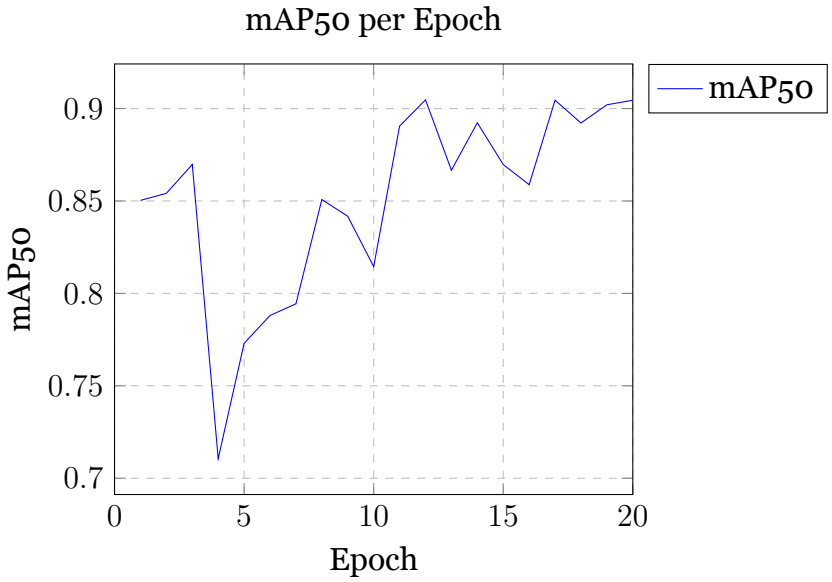

Figure 5.4: Mean Average Precision (mAP) with IoU=0.50 with the training model over 20 epochs

Both the figures 5.3 and 5.4 show that the model is slowly improving over the training. However, the model's metrics drop at the fourth epoch reaching the model's lowest score over the whole training. This drop matches with the spike in classification loss in figure 5.2, which could be the reason why the model gets worse at epoch four. The final mAP50 of the model on the validation set after 20 epochs is 0.905.

5.2 First Experiment:

Table 5.1: Accuracy of BM and TBM on the test set with no augmentations.

Model	Precision	Recall	mAP50
Base model	0.79	0.74	0.80
Trained model	0.832	0.87	0.874

From Table 5.1 we can see that our TBM is better in every metric compared to the BM. The TBM has a mAP50 of 0.874 while the BM has 0.80.

Table 5.2: Speed of BM and TBM on the test set with no augmentations

Model	Average Inference per image (ms)	FPS
Base model	14.9	67.1
Trained model	15	66.7

From Table 5.2 we can see that the TBM is marginally slower than the BM. The TBM's average inference per image is 14.9 ms while the TBM has 15 ms, which is a 0.1 ms difference. This leads to a 1.6 FPS difference between the models.

5.3 Second experiment

5.3.1 50% Darker

Table 5.3: Accuracy of BM and TBM on the test set "50% Darker."

Model	Precision	Recall	mAP50
Base model	0.77	0.54	0.65
Trained model	0.784	0.537	0.658

From Table 5.3 we can see that lowering the brightness of the images worsens both models in all metrics compared to results in Table 1. The biggest losses can be seen in both models' recalls. The base model goes from a recall of 0.74 to 0.54, and the trained model goes from 0.87 to 0.537. The base model loss in precision is negatable, going from 0.79 to 0.77. The trained model loss in precision is larger, going from 0.832 to 0.784.

5.3.2 50% Lighter

Table 5.4: Accuracy of BM and TBM on the test set "50% Lighter"

Model	Precision	Recall	mAP50
Base model	0.82	0.74	0.81
Trained model	0.911	0.667	0.789

Table 5.4 shows that increasing the light of the images has a different effect on both models. For the base model increasing the brightness, has a negligible positive on all metrics compared to Table 5.1 (Table 5.1 mAP50 of 0.8 and in Table 5.4 0.81). On the trained model increasing brightness had a big effect on the model's precision, going from 0.832 to 0.911. Increasing brightness also lowers the TBM's recall from 0.87 to 0.667.

The 50% Lighter experiment is the first time the base model has a higher mAP50 compared to the trained model. The BM has a mAP50 of 0.81 while the trained has 0.789, which is a 2.7% difference.

6 Discussion

The results show that the YOLOv8 model can be trained to detect humans in traffic situations. This can be seen in Table 5.1 where the TBM has a mAP50 increase of 9.3% compared to the BM. This increase in accuracy is also achieved without losing any speed of the model, with the two models differing in no meaningful way. The difference in average inference time is only 0.1 ms, which can be explained by the variance in testing speeds of algorithms. Even if the difference in speed is real, it would not matter in traffic situations as both models have over 60 FPS, meaning YOLOv8 can process images in real time.

From the training results it is probable that the TBM is not the best possible version of YOLOv8 for detecting pedestrians. The first reason is that the validation's box loss graph in Figure 5.1 never plateaus after around 20 epochs, which means that it never reaches its minimum loss. The second reason was that the other research papers [18] and [11], had YOLO models which were able to achieve higher mAP50 scores compared to TBM when detecting pedestrians. Hsu et. al was able to achieve a 91.6 mAP50 score with their YOLO model [11], which is significantly higher than TBM's mAP50 score of 0.874. If the experiment's training set was larger then it is possible that the YOLOv8 model could have achieved a higher mAP50 score, because Ultralytics recommends having at least 1500 images of a class for training and this experiment only had around 381 images[15].

In Table 5.3 we can see that both TBM's and BM's mAP50 scores decrease drastically, with TBM's score dropping 24.7% with darker images. The recall for both models dropped to around 0.5 which means they only were able to detect around half of all people in the images. This points to that the YOLOv8 model has difficulty detecting pedestrians in darker images. This contradicts the results shown in the study [18], their YOLO model was able to keep a high pedestrian detection rate in low-light images. One reason why TBM has trouble with darker images could be that the training set has no dark images for the detector to train on. Another reason could be that the darker images did not simulate nighttime in a realistic way. The results from the darker set could mean that YOLOv8 would not be a good choice as the main sensor for autonomous cars as they need to be able to

drive safely at all times of the day. This concludes that the YOLOv8 model needs further research with real night-time images to understand the reason behind the decrease.

The second experiment with 50% brighter images was the first that the BM got a better mAP50 score than TBM. The difference was not substantial only a 2.7% difference from TBM to BM. However, it shows that the TBM might be overfitted to the training set. Because the TBM lost its mAP50 score in this experiment while BM gained a little bit. If the TBM was able to generalize as well as BM, then both models should have behaved in a similar way.

Comparing the results from these experiments with the results from [10], shows that YOLOv8 is less accurate but is much faster compared to faster RCNN. This is in line with [6], Joseph Redmon et. al. showed that YOLO is magnitudes faster than faster RCNN but it has a lower mAP50. According to Joseph Redmon et. al., the reason why YOLO is faster is that it is a single-stage detector while faster RCNN is a two-stage detector. Depending on whether accuracy or speed is prioritized in a project YOLOv8 could be a good choice for object detection.

A problem with this study could have been that there are too many non-pedestrian images in our test set which could have affected the results of the experiments. The reason is that testing too many non-pedestrian images could have led to the experiments testing the models' ability to not give false positives, if the test set had more pedestrian images the experiment would test the models' ability to detect pedestrians.

In summary, these experiments can not possibly tell if YOLOv8 is a good model for use in autonomous cars, since the experiments do not test enough real-life traffic scenarios. However, we think that the model could be used in autonomous cars because the results show that YOLOv8 has a faster detection speed than humans at a high accuracy level. Humans have an average recognition reaction speed of 384 ms[19]. This means that the TBM is 25.6 times faster than humans, which could be life-saving in a traffic situation.

6.1 Future Work

The experiment with the “50% lighter” dataset showed that the precision of the base model and the trained model increased. This can show a promise that artificially increasing the brightness of images can lead to better detection rates for detection models. Further experiments with datasets with increased brightness would be interesting.

The images in the ”50% darker” dataset are artificially darkened and therefore the results can be misleading since it is not real night images. Further research should is needed with real night images to see if the results will differ.

The dataset for this paper is limited in the types of scenarios that are presented to the model. The pedestrians are all in normal traffic situations, walking on crosswalks or standing on the curb. There are no images where people are outside of the norm for example, images where people laying on the road or people who are in wheelchairs. This dataset limit could artificially increase the model’s mAP50 score compared to a more realistic database. An experiment with unique images could give more realistic results.

A problem identified during this study is that the standards for detection rates and accuracy of autonomous cars are not publicly available. This makes it hard to experiment with detection models.

7 Conclusions

In conclusion, the trained YOLOv8 model was able to get a mAP50 of 0.874 at an FPS of 66.7 when detecting pedestrians in traffic. The TBM has better accuracy than the BM of YOLOv8 by 9.3% while only being 0.1 ms slower than the original model BM. These results are not conclusive to say of YOLOv8 is accurate enough to be used in autonomous cars now, but the results show the YOLOv8 model is faster than some other object detection models and humans. The speed of the YOLOv8 detector could be the difference between life and death in a traffic situation, showing that YOLOv8 could be an important part of creating autonomous cars.

The TBM struggles to detect pedestrians in images with different light values than the original dataset. Especially in artificially augmented darkness since the model drastically decreased its capability to detect pedestrians. This shows that the model most likely is overfitted to the training set, because the BM does not lose the same percentage of accuracy when presented with new images. The reason for the overfitting may be that the training set is too small and homogeneous to train the model correctly. The YOLOv8 model needs further research before it can be used in autonomous cars.

References

[1] Gupta, Abhishek et al. “Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues”. In: *Array* 10 (2021), p. 100057. ISSN: 2590-0056. DOI: https://doi.org/10.1016/j.array.2021.100057. URL: https://www.sciencedirect.com/science/article/pii/S2590005621000059.

[2] Masita, Katleho L, Hasan, Ali N, and Shongwe, Thokozani. “Deep Learning in Object Detection: a Review”. In: *2020 International Conference on Artificial Intelligence, Big Data, Computing and Data Communication Systems (icABCD)*. Aug. 2020, pp. 1–11. DOI: 10.1109/icABCD49160.2020.9183866.

[3] Jocher, Glenn, Chaurasia, Ayush, and Qiu, Jing. *YOLO by Ultralytics*. Version 8.0.0. Jan. 2023. URL: https://github.com/ultralytics/ultralytics.

[4] MathWorks. *What Is Object Detection?* accessed 2023-06-08. URL: https://se.mathworks.com/solutions/image-video-processing/object-recognition.html.

[5] Henter, Gustav, Pokorny, Florian, and Smith, Kevin. *Generalization in Practice*. Accessed: 2023-05-14. URL: https://dd1420.notion.site/3-5-Generalization-in-Practice-56d81e02037a42ddbca2291f205d5d79#edeab4a39287463b96120ecfec261b8d.

[6] Redmon, Joseph et al. *You Only Look Once: Unified, Real-Time Object Detection*. 2016. arXiv: 1506.02640 [cs.CV].

[7] baeldung. *Intersection Over Union for Object Detection*. Last updated: May 5, 2023. URL: https://www.baeldung.com/cs/object-detection-intersection-vs-union.

[8] Hui, Jonathan. *mAP (mean Average Precision) for Object Detection*. Mar. 2018. URL: https://jonathan-hui.medium.com/map-mean-average-precision-for-object-detection-45c121a31173.

[9] Kumar, B Vinoth et al. “Detection and Content Retrieval of Object in an Image using YOLO”. In: *IOP Conference Series: Materials Science and Engineering* 590.1 (Oct. 2019), p. 012062. DOI: 10.1088/1757-899X/590/1/012062. URL: https://dx.doi.org/10.1088/1757-899X/590/1/012062.

[10] Akshatha, K. R. et al. “Human Detection in Aerial Thermal Images Using Faster R-CNN and SSD Algorithms”. In: *Electronics* 11.7 (2022). ISSN: 2079-9292. DOI: 10.3390/electronics11071151. URL: https://www.mdpi.com/2079-9292/11/7/1151.

[11] Hsu, Wei-Yen and Lin, Wen-Yen. “Adaptive Fusion of Multi-Scale YOLO for Pedestrian Detection”. In: *IEEE Access* PP (Aug. 2021), pp. 1–1. DOI: 10.1109/ACCESS.2021.3102600.

[12] Lan, Wenbo et al. “Pedestrian Detection Based on YOLO Network Model”. In: *2018 IEEE International Conference on Mechatronics and Automation (ICMA)*. 2018, pp. 1547–1551. DOI: 10.1109/ICMA.2018.8484698.

[13] roorkee, indian institute of technology. *Object detection and localization Dataset*. https://universe.roboflow.com/indian-institute-of-technology-roorkee-7immz/object-detection-and-localization. Open Source Dataset. visited on 2023-05-08. Mar. 2023. URL: https://universe.roboflow.com/indian-institute-of-technology-roorkee-7immz/object-detection-and-localization.

[14] Gillis, Alexander S. *What is data splitting and why is it important?* Apr. 2022. URL: https://www.techtarget.com/searchenterpriseai/definition/data-splitting.

[15] Jocher, Glenn. *Tips for Best Training Results*. Accessed: 2023-06-07. URL: https://docs.ultralytics.com/yolov5/tutorials/tips_for_best_training_results/.

[16] *HIT V3 Project*. URL: https://universe.roboflow.com/da150x-yolov8-pg/hit-v3.

[17] Ultralytics. *Home - Ultralytics YOLOv8 Docs — docs.ultralytics.com*. https://docs.ultralytics.com/. [Accessed 08-May-2023].

[18] Mao, Yitong. “A pedestrian detection algorithm for low light and dense crowd Based on improved YOLO algorithm”. In: *MATEC Web Conf.* 355 (2022), p. 03020. DOI: 10.1051/matecconf/202235503020. URL: ttps://doi.org/10.1051/matecconf/202235503020.

[19] Laming, D.R.J. *Information Theory of Choice-reaction Times*. Academic P., 1968. URL: https://books.google.se/books?id=%5C_cZWwAEACAAJ.

TRITA-EECS-EX-2023:282