Vehicles detection and counting using YOLO architecture

David Altamirano
davidalejandro.altamiranocoello@studenti.unipd.it
Stefano Meza
stefanorafael.mezaanzules@studenti.unipd.it

Abstract

Object detection is a highly demanding task in computer vision due to its relevance in today's world where high amounts of information are generated daily. As a consequence, many algorithms have been develop to filter visual media. YOLO (You only look once) is a powerful architecture capable of detecting, classifying and segmenting instances in photos and videos. This work is focused on testing the detection capacity of the last YOLO version using the smallest architecture and applying it on vehicle detection and counting. Several approaches were defined, creating a new dataset with many classes (approach A), merging existing datasets for achieving higher accuracy (approach B) and using trained models for performing detection and counting (approach C). Approach B had better results in the confusion matrix than Approach A, due to the less classes that were taken into account. Approach C significantly achieved the higher accuracy, being able to count all objects. Additionally, it is shown the versatility of this model in measuring traffic in different environments where human control can be inefficient.

1. Introduction

Object tracking [8] is a computer vision task that performs identification, monitoring and labeling of a given object through space and time. Essentially, it is based on tracking the position of every individual object in an image through consecutive frames of a video [1]. In the present work, we track vehicles in a highway which is extremely important in the context of autonomous vehicles due to enable the vehicle to recognize and track other vehicles, pedestrians and obstacles. This is vital for making informed decisions and ensuring safe navigation.

In order to perform vehicles counting, firstly we have to detect the objects and be able to recognize different kind of vehicles in a highway as cars, buses, trucks or motorcycles. Nowadays, there exist many algorithms for object detection based on different techniques and approaches. In this work we have played with YOLO [6] (You Only Look Once) algorithm for detecting the objects and then we use the trained model for counting vehicles in videos of highways [12]. Additionally, we compare our tracking result with a tracking done using a pre-trained model provided by YOLO as well. In Section 2 we discuss related works used for object detection. Section 3 describes the used dataset for training YOLO model. Section 4 discuss the approach used for tracking the vehicles. Section 5 shows the perform experiments and results that we got. Finally, section **??** summarizes our results in a precise way.

2. Related Work

For object detection we have used YOLOv8 [5] which is the lastest version of YOLO algorithms. The architecture consist of a backbone and head. The backbone is a pre-trained Convolutional Neural Network (CNN) that extracts feature maps from an input image. Then, these features are merged using path aggregation blocks. The head classify objects and predicting bounding boxes. In general, YOLOv8 was designed to be fast, accurate, and easy to use. It comes in five variants - nano(n), small(s), medium(m), large(l), and extra large(x) - depending on the number of parameters.

Besides, there exist 7 other versions of Yolo that have been improved along the time [4]. The first version was proposed in 2016 by Redmond [9] and consists on a single-shot detection (SSD) model that improved upon the standard R-CNN detection mechanism with faster and better generalization performance. SSD [7] is another approach to object recognition which consist on a single deep neural network which discretize the output space of bounding boxes into a set of default boxes over different aspect ratios and scales per feature map location.

Another important YOLO version is 5 [2]. The most remarkable difference is that YOLOv5 is an anchor-based model, whereas the YOLOv8 is an anchor-free model. Optimizing the procedure of selecting anchor boxes.

3. Dataset

The structural backbone of the present work is deeply detailed in the next section, but it is necessary to do sporadic mention about it for a better understanding of the present section. The first idea consisted on training Yolov8 model, for that we have built up our own dataset with the help of Roboflow. The dataset passes through a process of augmentation and consists of 2592 images (.jpg) with their respective bounding boxes (.txt) for 8 different classes ("SUV": 0, "big truck": 1, "bus": 2, "car": 3, "motorcycle": 4, "pickup": 5, "truck": 6, "van": 7).

Secondly, in order to improve the results, we decided to reduce the classes and increase the images of each one. In this case, the dataset has only 4 classes ("cars": 0, "buses": 1, "trucks": 2, "motorbikes": 3) and consists on 1975 images with their respective bounding boxes. For collecting them, different public repositories were used and combined [11, 13, 10, 3]. In the two previous cases, processed data was divided in: testing (10%), training (70%), validation (20%).

Finally, the most widely used dataset for the present work was COCO (Common Objects in Context) which is a well-studied dataset and actually it is the base for pretrained Yolo models.

4. Method

Methodology may be resumed in three approaches: Approach A is based on creating a new data set from scratch; adding all classes we want to detect and performing training; Approach B focuses on merging existing datasets, with a reduced number of classes for obtaining a higher detection accuracy and performing training from zero; and finally, Approach C, is essentially using a trained model and performing detection.

4.1. Approach A

As it was mentioned previously in Section 3, this dataset was created from scratch using public repositories and adding bounding boxes for each class. Consisting of 8 classes, it was tried to specify the class of the vehicle as much as possible. Once completed, model was trained and detection was performed.

4.2. Approach B

For achieving a higher accuracy, classes were shortened from 8 to 4 (cars, buses, motorcycles, trucks) and different available datasets were merged in order to increase dataset size. Next, the model was trained and some metrics were obtained.

4.3. Approach C

Finally, for this step we focused explicitly on the detection and counting stage. Detection was done using "yolov8n.pt" which is a trained model based on COCO dataset. COCO dataset is a public repository with more than 100 classes and well-known in object detection field.

4.4. Detection and Counting

For performing detection and counting of vehicles, using the previously obtained models, we decided to test them in two ways: first in images and then in videos. For images, it was straightforward due to the availables functions already implemented. Essentially, given an input image and a trained model, it recognizes the classes inside the images, showing the bounding boxes, accuracy and labels. Therefore a simple counting can be done.

For videos, the following pipeline was implemented:

- First, split a given video into many frames.
- For each frame, object detection was performed and bounding boxes and labels were retrieved.
- For counting vehicles, a horizontal line was drawn and objects passing through it were stored in a list.
- It is possible to distinguish direction of the vehicle by considering the sign of the distance between the line and the object

As these steps were performed during each frame of the video, it is possible to have double counting in many objects, therefore a tracking function was implemented. The tracker function stores the id of the objects that already were counted. Finally, counted objects and their labels are shown in the screen.

4.5. Computational Calculations

Training models from scratch can be computationally expensive and may cause failures in the training process. As an alternative, model training stage was divided into small training periods depending on the size of dataset and classes numbers. Basically, we selected a short number of epochs and then, the computed parameters, were used as the initial parameters for the next training. This led into a more stable learning.

5. Experiments

The results of our experiments are going to be presented according to the previous section. In Approach A, the confusion matrix and the evolution of the metrics indicate the performance of the model training using 8 classes. From this confusion matrix (Fig. 1) we may appreciate that SUV,

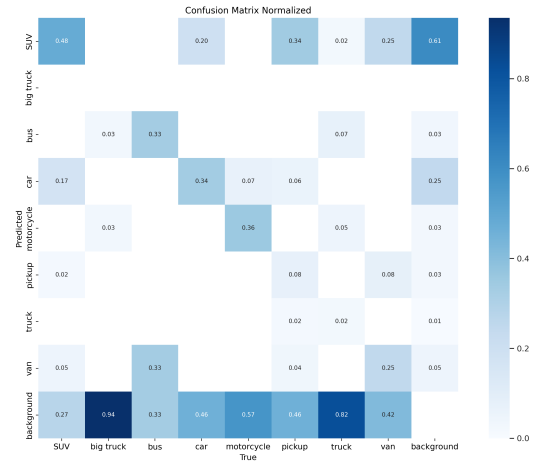

Figure 1. Confusion matrix related to Approach A where 8 classes were used

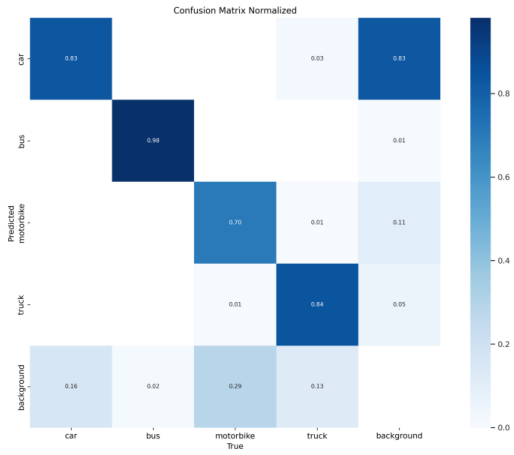

Figure 3. Confusion matrix related to Approach B where 4 classes were used

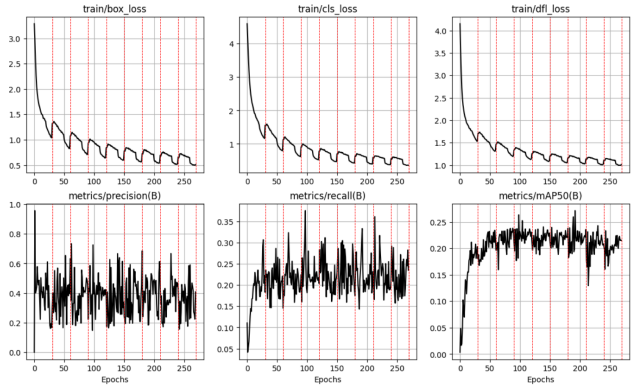

Figure 2. Metrics evolution during the training related to Approach A

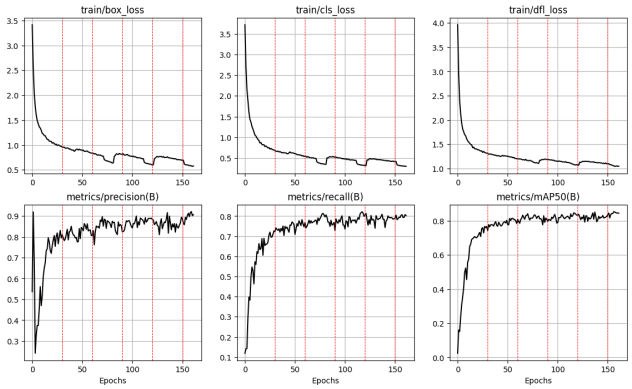

Figure 4. Metrics evolution during the training related to Approach B

car, bus and motorcycles show the higher accuracy. Nevertheless, it exists even more false negatives specifically when the object is a truck or big truck. Regarding to loss evolution, both, box loss and class loss, decreased during the training as we may expect while the metrics as precision or recall fluctuated in a stochastic way without being able to stabilize (Fig. 2. It is necessary to remark that the training was done up to 260 epochs but by parts because of the computational capacity. In any case, the results were not satisfactory at all. To improve the results, as explained above, we decided to reduce the number of classes and increase the number of images.

In Approach B, the confusion matrix (Fig. 3) shows a huge improvement where all classes have a remarkable performance having bus class as the highest result. At the same time, the metrics (Fig. 4) also showed a notable improvement trying to stabilize around 0.9 for precision and around 0.8 for recall and mAP50. In principle, this result seems like an acceptable one, but when it is used for detecting vehicles, it still has some errors as it may be appreciated in Fig. 5.B. At this point, we had tried to train the model for more epochs but we did not obtain significant improvements which led to Approach C.

For Approach C, confusion matrix nor metrics are available because the model is already trained. However, the accuracy can be observed in Fig. 5.C where more vehicles and classes are detected. For selecting the best model, we count the number of detected cars without taking into account the direction and the classes. Results can be seen in 1 where best performance was achieved by approach C (model trained with COCO). Surprisingly, approach A performed better than B which may depend on the training samples.

Once the approaches were performed and the C one was the winner, we decided to go beyond images and prove the model in videos counting how many cars travel in a highway. Four videos were used with different features in order to prove the performance of the model. The first video consists on a heavy traffic but mostly cars, the second video has less traffic but cars have higher speed, the third video is

Figure 5. Vehicle detection and counting in an image using: approach A, approach B and approach C

approach	cars detected	relative error
True	37	
A	12	0.66
B	10	0.72
C	37	0

Table 1. Counting vehicles error without taking into account labels or direction

again about heavy traffic but with more variety of vehicles, and the last video is an aerial video recorded in a perpendicular way with respect to land. The set up of the previous explanation is evidenced in Fig. 6. The first appreciation is that Yolov8n model is deficient for aerial shots perpendicular to the ground because this model is not trained to detect cars and actually it confuses cars with cell phones. For solving the problem, it is necessary to train the model with that specific dataset. With respect to the other videos, the model presents a good performance and it is able to count vehicles considering the direction and without considering it. Table 2 shows the result when the direction of vehicles is not taken into account and the most interesting and not so good result is about the video with fast cars because the model detects a little bit more than the existing ones. However, if we compare the results with Table 3 where the vehicles direction matter, a drastic decrease in the counted cars arises for fast vehicles which is closer to the real number. A possible hypothesis of the problem is related to the division of the frames because cars go faster than in the other videos. In addition to this, there is a problem that the number of cars in Table 2 and the sum of cars in Table 3 must be the same for each video, which does not happen. Nevertheless, the overall results of approach C are significantly and can be improve further by using more data to train the model, increasing the frames per second, and improving the Tracker

Figure 6. A frame of the used videos. A) is heavy traffic, B) is fast cars, C) is more variety, and D) is aerial view.

	cars	motorycles	buses	trucks
heavy	165	0	0	1
fast	234	0	0	18
mixed	82	7	3	17
aerial	0	0	0	0

Table 2. Counting of total vehicles using best model in different environments (heavy, fast, mixed, aerial).

	direction	cars	motorcycles	buses	trucks
heavy	in	24	0	0	0
	out	114	0	0	1
fast	in	80	0	0	6
	out	64	0	0	6
mixed	in	82	7	3	17
	out	31	0	1	7
aerial	in	0	0	0	0
	out	0	0	0	0

Table 3. Counting of total vehicles using best model in different environments (heavy, fast, mixed, aerial) distinguishing the direction of the vehicle.

function, which allow us to avoid double-counting.

6. Conclusion

YOLO is one of the best architectures for object detection. Its simpleness and versatility allows to perform different vision tasks such as detecting and labeling objects in videos or photos. Trained models are also very important for saving time in training, avoiding the necessity to create a dataset and providing accurate results. "Yolov8n.pt" pre-trained with COCO dataset has shown the possibility to measure traffic in different scenarios and taking into account the direction an the type of vehicle. Giving a remarkable accuracy with a simple pipeline that can be performed for first timers. Some improvements that can be performed in the future are: trying different YOLO models for specific set ups, creating better datasets that contains more impor-

tant features and improving the pipeline adding tasks which allow a better recognition and counting.

References

[1] Barga Deori and Dalton Thounaojam. A survey on moving object tracking in video. *International Journal on Information Theory*, 3:31–46, 07 2014.

[2] Yannan Hu, Mingming Kong, Mingsheng Zhou, and Zhanbo Sun. Recognition new energy vehicles based on improved yolov5. *Frontiers in Neurorobotics*, 17, 07 2023.

[3] HvA. school dataset. `https://universe.roboflow.com/hva/school-erqsf`, jan 2023. visited on 2024-01-28.

[4] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. A review of yolo algorithm developments. *Procedia Computer Science*, 199:1066–1073, 2022. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 2021): Developing Global Digital Economy after COVID-19.

[5] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics YOLO, Jan. 2023.

[6] Chengji Liu, Yufan Tao, Jiawei Liang, Kai Li, and Yihang Chen. Object detection based on yolo network. In *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pages 799–803, 2018.

[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. *SSD: Single Shot MultiBox Detector*, page 21–37. Springer International Publishing, 2016.

[8] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, and Tae-Kyun Kim. Multiple object tracking: A literature review. *Artificial Intelligence*, 293:103448, 2021.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection, 2016.

[10] scool. motorcycle dataset. `https://universe.roboflow.com/scool-rhhvh/motorcycle-xjypd`, sep 2023. visited on 2024-01-28.

[11] Mr. Simon. Bus dataset. `https://universe.roboflow.com/mr--simon/bus-rlkpi`, feb 2022. visited on 2024-01-28.

[12] Jing Tao, Hongbo Wang, Xinyu Zhang, Xiaoyu Li, and Huawei Yang. An object detection system based on yolo in traffic scene. In *2017 6th International Conference on Computer Science and Network Technology (ICCSNT)*, pages 315–319, 2017.

[13] Computer Vision. Cia-ii dataset. `https://universe.roboflow.com/computer-vision-iqni7/cia-ii`, jan 2024. visited on 2024-01-28.