Object Detection in Self-Driving Cars

By Raffaele 2047688, Waddah 2049298, and Srinjan 2053796

Abstract:

This report presents a study of fine-tuning Faster R-CNN and YOLO object detection models on the Udacity dataset for autonomous driving applications, focusing on the fine-tuning process, performance metrics, and practical implications of each model's results.

Introduction:

Urban environments characterized by their complex and busy layout present a significant challenge for autonomous vehicles. The ability of self-driving cars to make smart and informed decisions about their environment depends heavily on object detection. The emergence of robust models like Faster R-CNN and YOLO has been key to the advancement of this field. Yet, their effectiveness is contingent upon the quality and diversity of the datasets they are trained on. The Udacity self-driving car dataset offers an opportunity to fine-tune these models beyond standard benchmarks. This report compares Faster R-CNN and YOLO by fine-tuning them on the Udacity self-driving car dataset, hoping to create a more robust and effective object detection model for the future.

Related Work:

Our exploration of object detection in autonomous vehicles is based on a series of recent academic contributions. We first got the idea when reading a 2021 Hindawi paper about a real-time YOLO-based detector optimized for self-driving cars[1]. On top of this, we used a comprehensive survey from ResearchGate to deepen our understanding of how deep learning leads object detection in this field[2]. Other relevant papers include a 2023 MDPI Electronics article, which proposes a hybrid model combining YOLO's speed with Faster R-CNN's precision to try and achieve higher accuracy[3]. Practical applications and training methodologies are detailed in a Roboflow Blog post, which guided us on how to train a TensorFlow2 model for object detection[6].

Proposed Method:

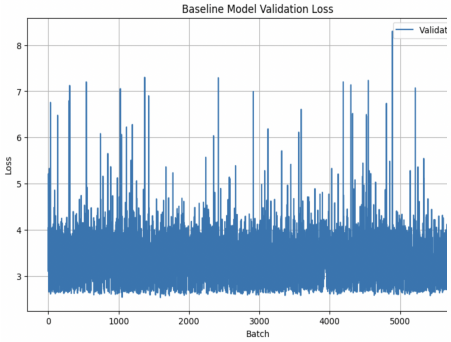
Our report is a comparison between a relatively old and proven object detection model and a new and exciting one. We took both these models, Faster R-CNN and YOLO, and fine-tuned them on the Udacity Self-Driving dataset, something neither of them had seen before. After taking a baseline result for Faster R-CNN we fine-tuned both the models on Udacity and compared the results, allowing us to catch a glimpse of how these models respond to fine-tuning on a new and complex dataset, then we pass the output of the detection model to Yolov7 segmentation model to investigate the combination.

Dataset and Benchmark:

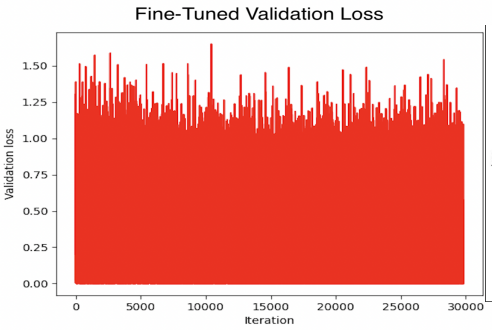
We used the Udacity self-driving car dataset[8], this dataset was perfect for our needs as it is both lightweight and complex, allowing us to run it on our computers while still providing a challenging environment for our models. Our benchmark was the baseline Faster R-CNN model that had been trained on the COCO image dataset, this model is one of the breakthroughs for

object detection in the late 2010s and provided us with a great comparison as we started fine-tuning.

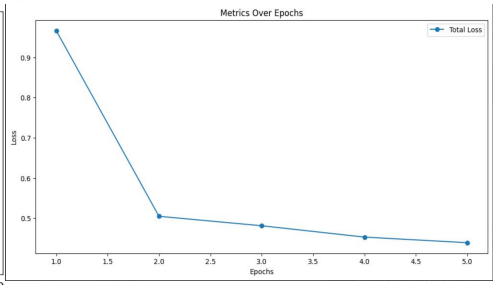
Experimental Results:
The faster R-CNN baseline had an average IoU of 0.07, an average precision of 0.47 and an average recall of 0.37. When fine-tuned the average IoU and recall improved slightly (0.1 and 0.48 respectively) while the average precision drastically improved, hitting 0.78. YOLO on the other hand, it hits about 0.84 for precision and 0.45 for recall, while also achieving a mAP score of 0.5. Here we can see the baseline, fine-tuned R-CNN and fine-tuned YOLO validation losses. These show how YOLO was able to decrease its loss over time while Faster R-CNN was able to improve from baseline but not consistently improve.



Faster Rcnn validation loss



fined tuned model VL



Yolo Validation loss

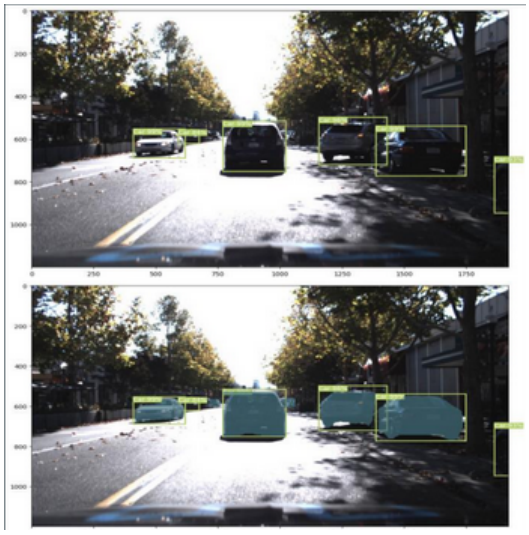
Qualitative testing and inference:

Faster RCNN

Yolo7

Combining segmentation with detection:

We passed the output of the detection models to the Yolo v7 segmentation model, and visualized many images that showed matching segmentation of the detected objects



Conclusions and Future Work:

Our project highlighted the differences between Faster R-CNN and YOLO in learning efficiency, with YOLO showing greater responsiveness and improvement. The architecture of YOLO, processing the image holistically, demonstrated an enhanced ability to recognize smaller and consistently located objects like street lights and pedestrians. In contrast, the region proposal mechanism of Faster R-CNN may have limited its capability in such cases as it takes the objects out of the context of the entire image. Moreover, YOLO has localization errors built into the loss function, which proved advantageous in the case of detecting hard-to-see streetlights. Also, Faster R-CNN seemed to suffer from the dataset imbalance more than YOLO, as for any classes that appeared too infrequently, the model would stop guessing them altogether, For future work, addressing the dataset imbalance is crucial. Additionally, exploring hybrid models or custom modifications that combine the strengths of both architectures might yield more comprehensive detection capabilities, especially in diverse and challenging urban environments. Reparameterization techniques can be applied to reduce the complexity of trainable Bag of Features (BoF) modules, resulting in a streamlined model optimized for faster deployment and inference.

Contributions:

Raffaele: First trying with Efficient det and waymo, then Faster RCNN fine tuning and testing, and writing final presentation and report
Waddah: fine tuning Yolo 7 and segmentation inference, preparing the first presentation and final report, researching waymo dataset format conversion.
Srinjan: helped prepare the first presentation, researching references.

References:

1. "A Real-Time Object Detector for Autonomous Vehicles." Hindawi, 2021. Available at: https://doi.org/10.1155/2021/9218137.
2. "Deep Learning for Object Detection and Scene Perception in Self-Driving Cars: Survey, Challenges, and Open Issues." ResearchGate, 2021. Available at: https://www.researchgate.net/publication/349526591.
3. "A Hybrid Approach to Object Detection in Self-Driving Cars." MDPI Electronics, vol. 12, no. 13, 2023. Available at: https://www.mdpi.com/2079-9292/12/13/2768.
4. Shashankag14. "Fine-Tune Object Detector." GitHub repository. Available at: https://github.com/shashankag14/fine-tune-object-detector.
5. "YOLOv3: An Incremental Improvement." arXiv, 2015. Available at: https://arxiv.org/abs/1506.01497.
6. "Train a TensorFlow2 Object Detection Model." Roboflow Blog. Available at: https://blog.roboflow.com/train-a-tensorflow2-object-detection-model/.
7. Raz4. "Waymo Object Detection." GitHub repository. Available at: https://github.com/raz4/waymo-object-detection/tree/master.
8. https://public.roboflow.com/object-detection/self-driving-car