Available online at https://asric.africa/engineering-sciences
ASRIC Journal on Engineering Sciences Vol.4(1)(2023) 198-208

Real-Time detection Using Deep Learning Techniques for autonomous Vehicle

Ghita Ikmel[1]*, EL Amrani EL Idrissi Najiba[1]

Signals Systems & Components Laboratory (LSSC), Faculty of Sciences and Technology of Fez, Fez, Morocco,

*Corresponding author: ghita.ikmel@usmba.ac.ma

Abstract
Autonomous driving heavily relies on object detection as a primary task. Ensuring both high accuracy and real-time detection is crucial for the object detection algorithm of autonomous vehicles. Consequently, the research in this area has garnered significant interest, becoming a prominent and trending topic in recent years. This study aims to find an algorithm that strikes a balance between speed and accuracy in object detection. In order to accomplish this, the latest object detection algorithms, including YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, YOLOv5x, YOLOv7, YOLOv7-tiny, YOLOv7-X, and Faster R-CNN, are compared using the Self-Driving Car dataset. Through comprehensive experiments, the algorithms' performance is evaluated based on their inference time, precision, recall, and mean accuracy (mAP) values. The results demonstrate that Faster R-CNN stands out with a precision of 0.930, showcasing its excellence in accuracy. On the other hand, YOLOv5n boasts the fastest inference time, clocking in at a mere 7.5 ms. In summary, this research delves into the crucial aspect of object detection in autonomous driving, and its findings provide valuable insights into the trade-offs between speed and accuracy for different state-of-the-art algorithms, such as YOLO and Faster R-CNN, utilizing deep learning techniques.
Keywords: autonomous driving, deep learning, object detection, You Only Look Once (YOLO), convolutional neural network.

Introduction
Object detection, a crucial branch of artificial intelligence and computer vision, aims to develop machine learning models capable of recognizing specific items within images. This concept underlies various practical applications, such as face and image recognition, making it a fundamental area of study (P. Viola and M. Jones, 2021), (J. Luo et al., 2001). To train a machine learning model effectively, a substantial amount of labeled data is required, consisting of images containing the target object(s) that the model is intended to identify, even when encountering previously unseen and unique items (I.Laptev, 2009). One significant application of object detection lies in street-level object identification, extensively employed in surveillance cameras, drones, and self-driving vehicles. The ability to identify objects in real-time helps prevent accidents and collisions, enabling law enforcement to monitor traffic law compliance and road safety violations. This application's widespread relevance makes it an ideal focus for research endeavors (K. Kang et al., 2018). Numerous techniques and architectures have been explored to develop machine learning models, including convolutional neural networks (CNNs), support vector machines (SVMs), and feature selection methods (V. Kurilová et al., 2021), (H. Zhang et al., 2013). Each algorithm and design offer distinct characteristics and performance outcomes, making their selection critical based on specific use cases.

In recent years, several architectures and algorithms have gained popularity for object detection tasks. Among them are You Only Look Once (YOLO), Single Shot Multibox Detectors (SSD), and Region Based Convolutional Neural Networks (RCNN) and its faster variant Faster-RCNN (J. Redmon et al., 2016), (W. Liu et al., 2016), (R. Girshick, 2015), (S. Ren, 2016).While many studies have investigated the performance of object detection algorithms under various scenarios, some novel methods, such as YOLOv7, have received limited attention. Comparisons between YOLOv7 and its predecessors or other object detection algorithms, particularly concerning multiple object detection on roads, remain scarce. Existing research has contrasted different road-specific object detection methods. For instance, a study by S. Choyal and A. K. Singh compared Faster-RCNN and MobileNet SSD on traffic signs, highlighting Faster-RCNN's superior accuracy but longer training time (S. Choyal et A. K. Singh, 2020). Another study evaluated various detection algorithms, comprising MobileNetv2 SSD FPN-lite 320x320, YOLOv3, YOLOv4, YOLOv5s, and YOLOv5l, where YOLOv5l demonstrated better precision, while MobileNetv2 FPN-lite exhibited faster inference time (M. G. Naftali, J. S. Sulistyawan, and K. Julian, 2022). Additionally, Faster RCNN outperformed SSD, YOLOv5n, YOLOv3-tiny, and YOLOv4-tiny in another comparative study (Bie et al., 2023). Although several studies have explored YOLO object detection, YOLOv3, Faster-RCNN, and SSD MobileNet remain the primary focus in vehicular cases. Yet, few studies have specifically compared the YOLOv7 algorithm with its predecessors in the context of detecting multiple objects on roads. In this paper, our objective is to address this gap through a comprehensive study. Section 2 provides background information on various object detection methods. Section 3 covers the dataset used, pre-processing techniques, and the selection of methods and tools. In Section 4, we present the experimental results and discussions, culminating in a conclusion that summarizes the findings and highlights the significance of comparing YOLOv7 with its predecessors in a multi-object detection scenario on roads.

State of The Art

Over the last few years, object detection has witnessed significant advancements, driven by the rapid progress in deep learning and computer vision. Researchers and practitioners alike have been actively exploring various state-of-the-art approaches to improve the accuracy and efficiency of object detection algorithms for autonomous vehicles and related applications. One of the most notable advancements in this domain is the emergence of deep learning-based methods, particularly convolutional neural networks (CNNs), which have shown remarkable success in handling complex visual recognition tasks. CNNs have revolutionized object detection by enabling end-to-end learning, where the model learns to automatically extract meaningful features from raw image data and make accurate predictions. Among the prominent object detection algorithms, You Only Look Once (YOLO) and Single Shot Multibox Detectors (SSD) stand out as real-time detection solutions that strike a balance between speed and accuracy. YOLO's single-pass architecture processes the entire image at once, offering impressive inference speeds. SSD, on the other hand, adopts a multi-scale approach to predict bounding boxes, efficiently detecting objects of varying sizes in a single shot. Region based Convolutional Neural Networks (RCNN) and its variant Faster R-CNN represent another significant milestone in object detection. These approaches utilize region proposal techniques to identify potential object regions, followed by object classification and bounding box regression. Although they excel in accuracy, their computational demands have led to the exploration of lighter architectures. The introduction of YOLOv7 showcases the ongoing efforts to enhance object detection performance even further. While specific details about YOLOv7 are not widely covered in the existing literature, its introduction suggests the possibility of advancements beyond its predecessors, such as YOLOv5. The research community's growing interest in YOLOv7 underscores its potential as a promising candidate for real-time object detection applications, particularly in autonomous driving scenarios. Despite the considerable progress in object detection algorithms, there remains a need for comprehensive comparative studies to understand the strengths and weaknesses of each approach concerning specific use cases. Addressing this gap in the literature, the current research aims to compare YOLOv7 with its predecessors and other popular object detection algorithms, particularly in the context of multiple object detection on roads. By evaluating precision, inference time, and other key performance metrics on the Self-Driving Car dataset, this study seeks to shed light on YOLOv7's applicability and potential advantages in real-world scenarios. As object detection technology continues to evolve, the findings of this research are expected to contribute significantly to the development of more robust and efficient autonomous driving systems. Leveraging the latest state-of-the-art techniques, researchers and developers can strive to improve road safety, enable smarter transportation, and pave the way for a future where self-driving vehicles become a widespread reality.

Backgrounds

This section present and explore the architecture of the different object detection models : Faster R-CNN, YOLO and its versions (YOLOv5 and YOLOv7).

Faster R-CNN

Faster R-CNN is a pioneering object detection algorithm that has significantly improved both accuracy and speed in the field (Uijlings et al., 2023), (K. E. A. van de Sande et al., 2011). One of the key factors contributing to this improvement is the selective search technique, which generates approximately 2000 region proposals or Regions of Interest (RoIs). However, this technique requires substantial time to analyze each region suggestion in the image and map them onto the feature maps, which can lead to a bottleneck in the object detection architecture. To address this limitation, Shaoqing Ren and his team proposed a groundbreaking solution in Faster R-CNN by introducing a Region Proposal Network (RPN) [15]. This RPN replaces the traditional selective search technique for generating region proposals and instead integrates region proposal extraction into Deep Convolutional Neural Networks (DCNNs).

By doing so, the RPN grants the detection network access to the convolutional features of the entire image, significantly speeding up the process. A noteworthy advantage of using the RPN is its implementation on the Graphics Processing Unit (GPU), resulting in near-costless operations. This is made possible because the region proposals share the convolutional feature maps, enabling the RPN to perform efficiently and further accelerating the object detection process. In summary, Faster R-CNN's success can be attributed, in part, to the adoption of the RPN, which replaces the conventional selective search technique with a more streamlined and GPU-accelerated approach. This enhancement not only boosts the speed of region proposal generation but also empowers the detection network with access to comprehensive convolutional features, leading to impressive gains in both accuracy and real-time performance. As a result, Faster R-CNN has become a pivotal milestone in the evolution of object detection algorithms and continues to influence the development of cutting-edge solutions within the realm of computer vision.

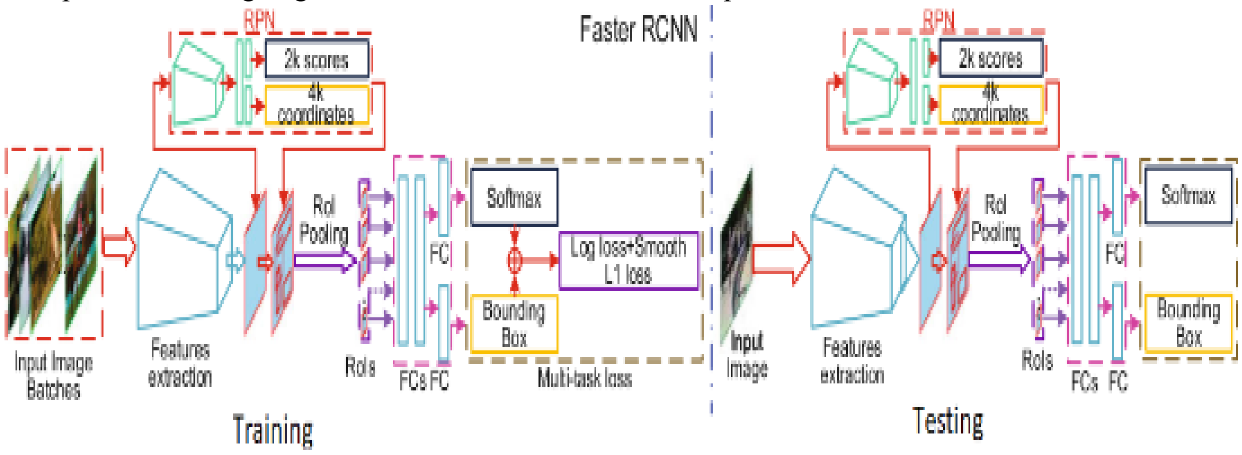


Figure 1: The details of the classical Faster RCNN (the training and testing process).***Yolo***

The Region Proposal Network (RPN) plays a vital role in YOLO, enabling the reduction of region proposals from around 2000 to just 300. Despite this improvement, the challenge of overlaps between region proposals persists. Overlapping regions can lead to repeated processing during object identification, hindering efforts to break the speed barrier. In 2015, J. Redmon et al. introduced an innovative solution to address these challenges with a single end-to-end neural network called YOLO (You Only Look Once) (J. Redmon et al., 2016). YOLO revolutionized object detection by directly generating bounding box regression and class probabilities from the entire image in one pass. The image is divided into a grid of size S×S, and each individual cell in the grid is responsible for class classification, regression, and object center detection. Within each grid cell, YOLO predicts bounding boxes B, confidence scores, and class probabilities C, resulting in a tensor of size S×S×(5B+C) that encodes the entire image. This efficient approach significantly reduces the computational complexity compared to traditional region-based methods. The YOLO architecture consists of 24 convolutional layers and 2 fully connected layers, processing input images of size 448×448. This streamlined design allows YOLO to achieve impressive real-time performance while maintaining competitive accuracy. In summary, YOLO's unique approach to object detection leverages a grid-based system that handles class classification and regression within individual cells, drastically reducing the number of region proposals and enabling real-time processing. By adopting an end-to-end neural network architecture, YOLO has made substantial contributions to the advancement of object detection technology, opening new possibilities for fast and accurate applications in fields such as autonomous driving, surveillance, and robotics. Figure 2 depicts the architecture of YOLO, showcasing its efficiency and effectiveness in object detection tasks.

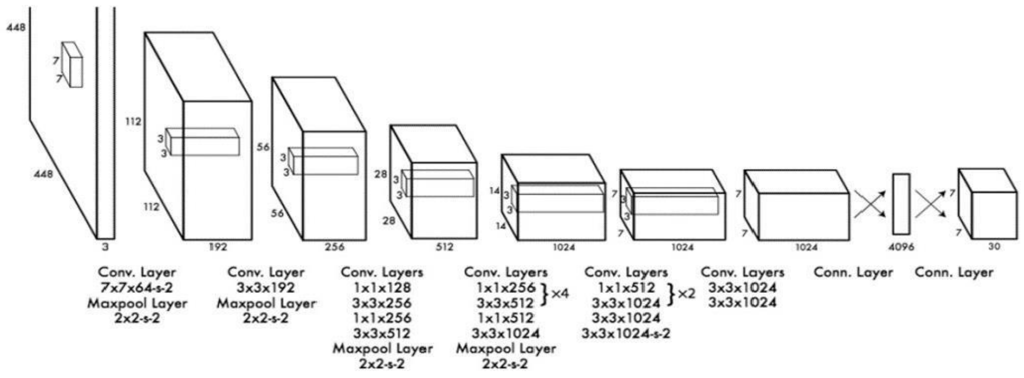


Figure 2: Architecture of YOLO model.

Yolov5

Building upon the YOLO detection architecture, YOLOv5 (M. Horvat, L. Jelečević, and G. Gledec, 2022) introduces various optimization techniques utilizing convolutional neural networks. Notably, it incorporates the automatic learning of bounding box anchors. YOLO was the first object detector to create an end-to-end differentiable network, connecting class labels with the bounding box prediction algorithm (J. Nelson and J. S. JUN, 2022). The YOLOv5 network consists of three primary components: the backbone, the neck, and the output (X. Hao, L. Bo, and Z. Fei, 2022). Data preparation operations, including adaptive image filling and elevation of mosaic data, are performed at the input point. YOLOv5 features adaptive computation of the anchor frame, allowing it to automatically adjust the anchor frame's starting size based on different datasets, enhancing adaptability. The backbone employs a convolutional neural network to gather and refine image features at multiple granularities. It utilizes repeated convolution and pooling, incorporating techniques like the partial cross network (CSP) and spatial pyramid pooling (SPP) (K. He, X. Zhang, S. Ren, and J. Sun, 2015). The SPP framework performs feature extraction of various scales on the same feature map, aiding in detection accuracy. On the other hand, the BottleneckCSP design reduces computational burden and accelerates inference. The neck neural network serves as a layer array for combining and transmitting picture information to the prediction stage. It contains both the Feature Pyramid Network (FPN) and the Path Aggregation Network (PAN) pyramidal feature structures. The FPN architecture conveys high-level semantic features from upper to lower feature maps, whereas the PAN structure facilitates the transmission of resilient localization features from lower to higher feature maps(L. Liu, 2016). Finally, the head output acts as the last detection step, making predictions for targets of various sizes on the feature maps. In summary, YOLOv5 represents a significant advancement in object detection algorithms, employing convolutional neural network optimizations and efficient feature extraction techniques. With its end-to-end differentiable network and adaptive anchor frame computation, YOLOv5 strikes a balance between accuracy and speed, making it a powerful tool for real-time object detection tasks in diverse datasets and applications.

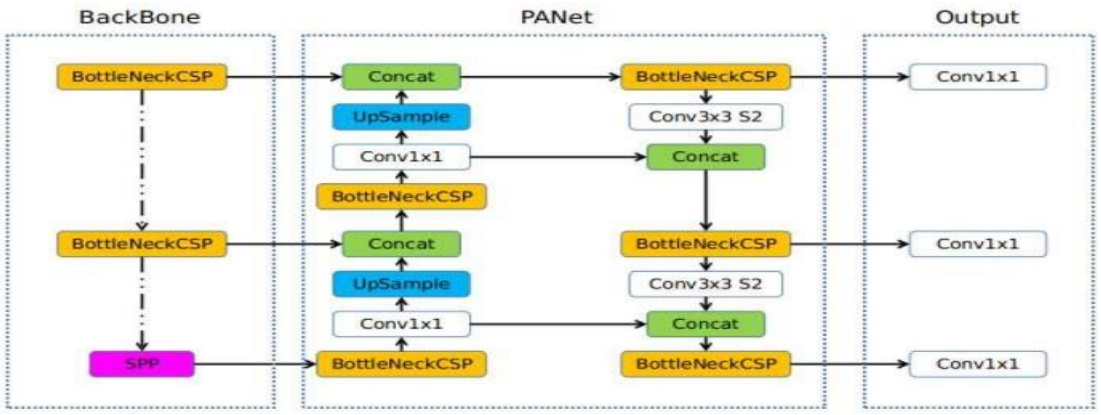


Figure. 3: Architecture of YOLOv5.

Yolov7

The development of the YOLOv7 architecture (C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, 2022) was influenced by the YOLOv4, Scaled YOLOv4, and YOLO-R designs. At the core of YOLOv7 lies the Extended Efficient Layer Aggregation Network (E-ELAN), a main computing block that adopts the Expand, Shuffle, Merge cardinality strategy. This E-ELAN architecture not only enhances the model's learning efficiency but also preserves the original gradient path, contributing to improved performance. Model scaling is a fundamental concept in YOLOv7, allowing for the expansion of the model's depth, picture resolution, and width. Unlike traditional concatenation-based architectures (e.g., ResNet or PlainNet), where scaling individual aspects separately might yield suboptimal results, YOLOv7 enables composite model scaling. By considering all scaling aspects together, the original model's characteristics are retained, ensuring an ideal structure. The process of scaling composite models involves adjusting depth factors in computational blocks and simultaneously modifying the output channels to maintain consistency. Additionally, the width factor is scaled on the transition layers with the same degree of change. In YOLOv7, a novel reparameterized convolutional architecture called RepConvN is employed, building on the concept of RepConv, which demonstrates excellent performance in VGG architectures. However, when directly applied to ResNet or DenseNet, accuracy suffers significantly. To address this, RepConvN is designed without an identity connection, effectively preventing an identity connection when a parameterized convolution replaces a convolutional layer with residue or concatenation. The YOLOv7 architecture is not limited to a single head. Instead, it employs multiple heads to support different aspects of the model. The auxiliary head is used to aid training in the intermediate layers, while the main head is responsible for producing the final output. In summary, YOLOv7 represents an advanced and innovative object detection architecture, building upon the successes of its predecessors while introducing novel components like E-ELAN and RepConvN. By incorporating composite model scaling and employing multiple heads, YOLOv7 offers flexibility and efficiency in handling a variety of tasks and datasets. These advancements contribute to YOLOv7's potential as a state-of-the-art solution for real-time object detection and its relevance in advancing the field of computer vision.

Materials and methods

This section describes the software and data used in our experiments. We will also talk about the six metrics that are used to evaluate the algorithms.

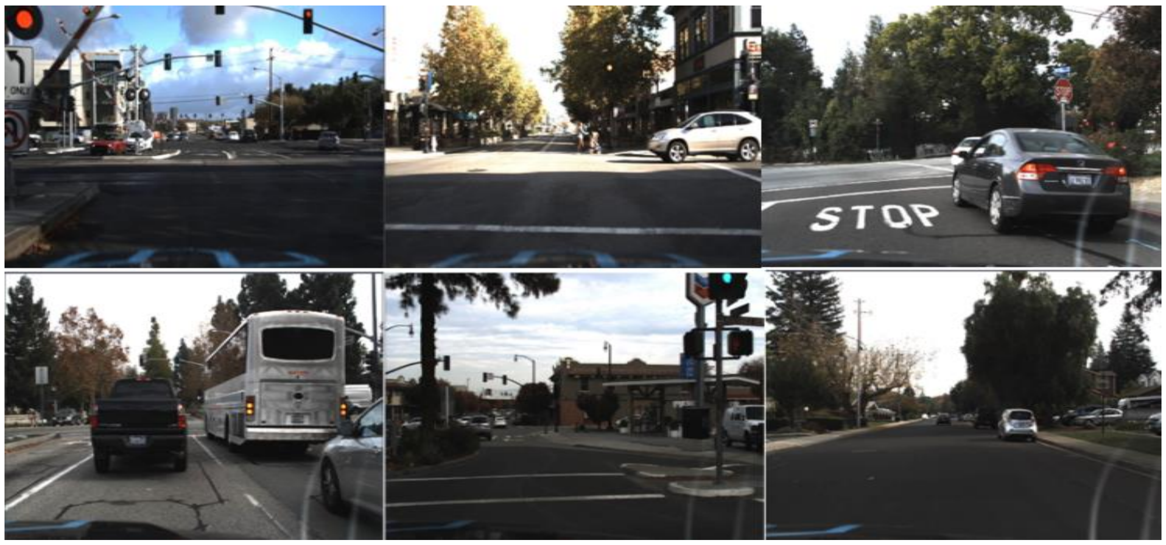
Dataset Description

The self-driving cars dataset (Version Roboflow), used in this research, contains 5950 images of roads, in JPG color format and with a resolution of 1920x1200 pixels. The images were taken in different environments and weather conditions to ensure that the algorithms' genericity was tested. There are eleven objets classes (pedestrian, bike, truck, car, trafficLight, trafficLight-Green, trafficLight-Red, trafficLight-Yellow, trafficLight-GreenLeft, , trafficLight-RedLeft, , trafficLight-YellowLeft). This dataset is described in the Roboflow public dataset list and can be downloaded or transferred from this list. The goal of the platform known as Roboflow is to provide users with faster and more accurate computer vision models. It allows them to collect and analyze data faster. For this research, the “self_driving_cars” dataset was transferred to the new Roboflow project, where modifications were made.

This image dataset is divided into three parts:
A training set used to adapt or train a learning model.
A validation set is used to impartially assess how well the model fits the training dataset. It also adjusts the model's hyperparameters to ensure that the training model is performing well.
A test set used for the evaluation of the final model.

The training set is composed of 4700 images of labelled objects, the test set contains 677 images. Figure 4 shows a portion of the used data set.

Figure 4: Images' sample of the dataset



Preprocessing of the images

Preprocessing of images is a crucial step in ensuring consistent and accurate predictions and learning in object detection experiments. In this study, Roboflow was employed to perform two essential preprocessing steps. First, the "Auto-Orient" option was enabled, which ensures that the images are displayed in a manner consistent with how they are stored on disk. This step is essential to avoid any discrepancies in image orientation during the preprocessing stage. Next, the images were resized to a uniform size of 640x640 pixels. This standardization of image dimensions ensures that all images in the dataset have the same resolution, which is essential for training and evaluating the object detection models consistently. As part of the preprocessing pipeline, Roboflow automatically adjusted the annotations to correspond accurately with the resized images. Annotations play a crucial role in object detection, as they define the ground-truth bounding boxes and class labels for the objects of interest. The automatic adjustment of annotations helps maintain the integrity of the dataset and facilitates seamless training of object detection models. Furthermore, to ensure compatibility with specific object detection models like Faster R-CNN and when working with frameworks like Detectron2, the preprocessed data was exported to a specific annotation format. In this case, the COCO JSON format was utilized. COCO is a well-structured JSON dataset format widely used by major tech companies such as Microsoft, Google, and Facebook. It provides a standardized way to store annotations, making it easier to work with various object detection tools and frameworks. In summary, the preprocessing steps conducted using Roboflow, including image resizing, auto-orienting, and automatic annotation adjustment, ensure consistency and compatibility for subsequent object detection tasks. The use of the COCO JSON format streamlines data management and facilitates seamless integration with popular object detection models and frameworks, making the entire experimentation process more efficient and effective.

Training environment and details

Due to the computational limitations of our local computer, we opted to work in Google's Colab, a free and powerful environment tailored for deep learning and machine learning applications. Colab offers access to GPUs and TPUs without any cost, making it an ideal choice for resource-intensive tasks. The Colab code environment comes pre-installed with popular deep learning libraries such as PyTorch, TensorFlow, OpenCV, and Keras, ensuring developers have the necessary tools at their disposal to build and execute sophisticated models seamlessly. While working in Colab, there is a maximum connection time of about 12 hours to a virtual machine, which varies based on the GPU type and available resources. Despite these limitations, the resources provided by the free environment were sufficient to successfully run our experiments. Table 1 presents the training durations of various object detection models. Notably, the Faster R-CNN model demonstrated the shortest training time, completing in only two hours. On the other hand, the YOLOv7-X model exhibited the longest training duration, spanning almost seventeen hours. These variations in training times are expected, as different models have distinct architectures and computational requirements. By leveraging Colab's GPU and TPU capabilities, we were able to overcome the computational constraints of our local machine and conduct extensive experiments, successfully training complex object detection models without incurring any additional costs. The accessibility and flexibility of Colab significantly facilitated our research, allowing us to explore and analyze various algorithms and architectures effectively.

Table I. Duration of Training of Models In Hours on The Experimental Dataset

Models	**Version**	**Duration[h]**	**Params[M]**
Yolov7	Yolov7	14.577	36.9
	Yolov7-X	17.074	71.3
	Yolov7-tiny	11.009	6.2
Yolov5	Yolov5n	3.229	1.9
	Yolov5s	3.965	7.2
	Yolov5m	5.727	21.2
	Yolov5l	6.866	46.5
	Yolov5x	5.111	86.7
Faster R-CNN		2.080	105

Performance Metrics

In the presented research, several assessment metrics were employed to analyze the obtained experimental outcomes. These metrics encompass] precision, recall, F1 score, mean accuracy (mAP), weight distribution, and inference time for the purpose of comparison.

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$\text{F1-Score}=2\times\frac{P\times R}{P+R} \quad (3)$$

$$\text{mAP}=\frac{\sum_{q=1}^{Q} AveP(q)}{Q} \quad (4)$$

Equation (1) calculates the proportion of True Positives among all samples that were classified as positive.

As indicated in equation (2), to compute the recall rate, you can easily obtain it by dividing the count of true positive samples (correctly classified as positive) by the total number of positive samples.

With a value between 0 and 1, equation (3) describes the balance between accuracy and recall. A higher F1 score indicates a more balanced combination of precision and memory.

Equation (4) is used to calculate mAP, which is the average precision across all classes. An object detection algorithm's overall performance may be evaluated using the measure known as mAP.

The inference time reflects how fast the algorithm can predict in real time. For a real-time application, the shorter the forecast time, the better.

Results and Discussion

Table 2 shows the tests results for the investigated object detection algorithms.

Table II: Evaluation Metrics of Various Object Detection Algorithms

Models	**Precision (%)**	**Recall (%)**	**F1 (%)**	**mAP (%)**	**Weights (MB)**	**Inference time (ms)**
Yolov7	67.5	58.9	62.90	58.9	74.9	12
Yolov7-tiny	78.2	53	63.17	59.4	12.3	8.8
Yolov7-X	78.4	68.2	72.94	70.7	142.2	21.5
Yolov5n	75.3	57.1	64.94	60.9	3.8	7.5

Yolov5s	80.7	60.8	69.35	68.1	14.4	8
Yolov5m	79.2	68.5	73.46	72	42.2	13.4
Yolov5l	82.7	69.6	75.58	74.8	92.9	15.2
Yolov5x	83.9	68.2	75.23	74.3	173.1	26
Faster R-CNN	93	98	95.43	98.19	401.34	194.5

Table 2 provides a comprehensive summary of the performance of Faster R-CNN, YOLOv5n, YOLOv5s, YOLOv5m, YOLOv5l, and YOLOv5x models. All models were trained for 200 epochs, and the results indicate promising performance across the board. The YOLOv5n model demonstrated a mean Average Precision (mAP) of 0.609, with accuracy and recall values of 0.753 and 0.571, respectively. Notably, this model was the lightest and fastest, weighing only 3.8 MB, making it a practical choice for applications where resource constraints are a concern. The YOLOv5s model showed improved performance, with an mAP of 0.681, accuracy of 0.807, and recall of 0.608. Compared to YOLOv5n, this model achieved better results across the board. Moving to the YOLOv5m model, we observed further improvements in detection performance. It outperformed both YOLOv5n and YOLOv5s models, boasting an mAP of 0.72, accuracy of 0.792, and recall of 0.685. Both YOLOv5l and YOLOv5x models exhibited high mAP values of 0.748 and 0.743, respectively. The results clearly indicate that increasing the number of parameters significantly influences the detection performance of the security system. As the number of parameters increased from YOLOv5n to YOLOv5x, the model's performance improved. It is worth noting that as the number of parameters increased, the detection speed of the model was impacted.

While YOLOv5n was the fastest, YOLOv5x, with a substantial number of parameters, demonstrated a slower detection speed.

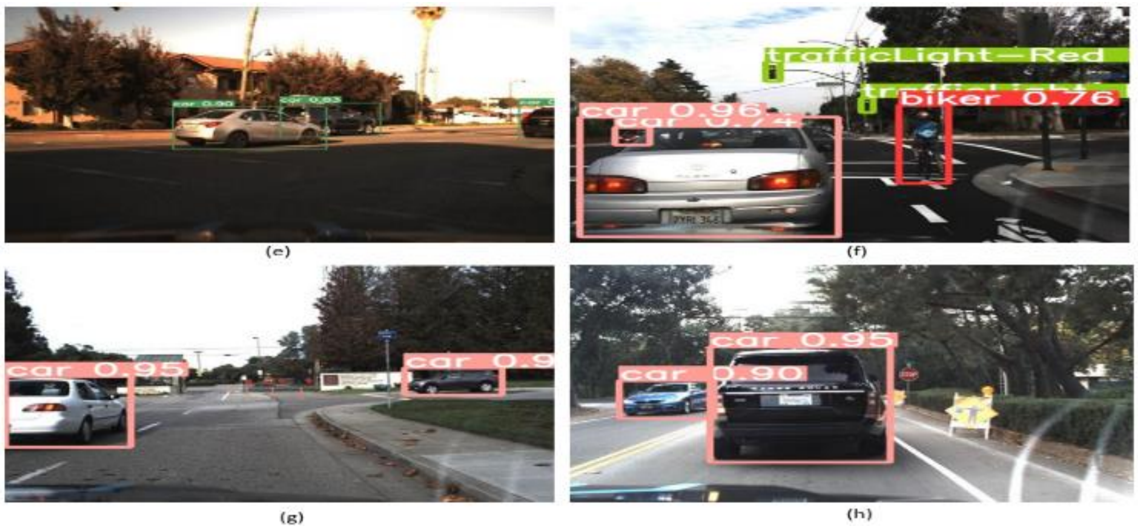


Figure. 5: Detection result illustrated on sample test image.

(e): Yolov7, (f): Yolov5s, (g): Yolov5m, (h): Yolov5n.

The evaluation of the YOLOv7 models on the dataset revealed that their performance was not satisfactory. Across the different versions of YOLOv7, there were no notable differences in the results. The smallest model, YOLOv7-tiny, achieved an mAP of 0.594, a precision of 0.782, and a recall of 0.53. These values indicate that the model struggled to achieve accurate and comprehensive object detection on the dataset. Similarly, the first version of YOLOv7, YOLOv7-X, showed limited improvement with mAPs of 0.707, a precision of 0.784, and a recall of 0.682. While there was a slight increase in performance compared to YOLOv7-tiny, the model's overall results were still not up to the desired level. The precision of 0.675 attained using YOLOv7 further highlights the challenges in achieving accurate object detection with these models. Based on the evaluation, it appears that the YOLOv7 models did not perform as well as anticipated on the dataset. The mAP values, precision, and recall scores for the different versions were relatively lower compared to the YOLOv5 models discussed earlier. It is important to note that model performance can be influenced by various factors, including model architecture, hyperparameter tuning, dataset characteristics, and training strategy. To improve the performance of YOLOv7 models, further investigation and experimentation might be required to identify potential areas for enhancement. Overall, the evaluation results indicate that YOLOv7 models might require additional refinements and optimizations to achieve better object detection performance on the given dataset. As with any deep learning model, fine-tuning and optimization are essential steps to maximize the model's capabilities and suitability for real-world applications.

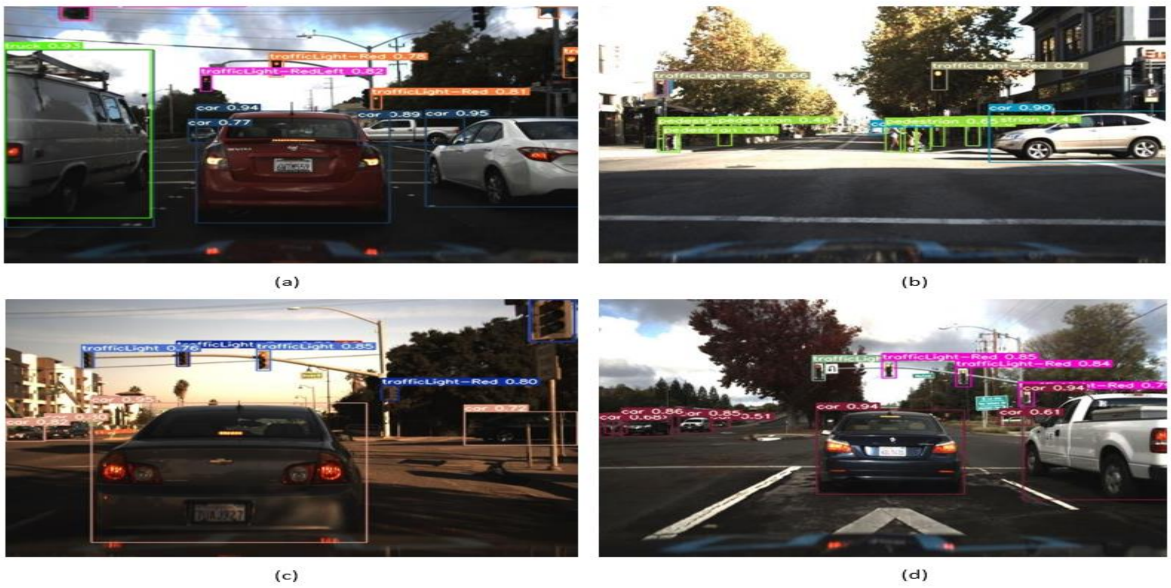


Figure. 6: Detection result illustrated on sample test image.
(a), (b): Yolov7-X, (c), (d): Yolov7-tiny .

The results of the experiment indicate that Faster R-CNN is the most accurate object detection algorithm among those examined, achieving the highest F1 and mAP scores. However, this accuracy comes at the cost of the longest inference time, taking 194.5 ms, making it the slowest model compared to others. On the other hand, YOLOv7 performed the poorest in terms of accuracy, with a low mAP score of 0.589. The first fastest object detection method, YOLOv5n, showed a significant performance gap compared to the most accurate Faster R-CNN algorithm, performing 37.29% lower with an F1-Score of 0.6494 and a mAP of 0.609. These findings highlight the trade-offs between speed and accuracy in object detection algorithms. Faster R-CNN excels in accuracy but sacrifices inference time, while YOLOv5n achieves faster processing but falls short in accuracy when compared to Faster R-CNN. Choosing the most suitable object detection algorithm for a specific application requires considering the specific requirements and constraints of the task at hand. For applications where high accuracy is crucial, such as in safety-critical systems, Faster R-CNN might be the preferred choice, even with the longer inference time. On the other hand, for real-time applications that prioritize speed, YOLOv5n could be a more appropriate option, despite its lower accuracy compared to Faster R-CNN. Ultimately, the choice of the object detection algorithm should be made based on a careful evaluation of the application's needs, performance trade-offs, and available resources. The results of this experiment provide valuable insights into the strengths and weaknesses of different algorithms, aiding researchers and developers in making informed decisions for their specific use cases.

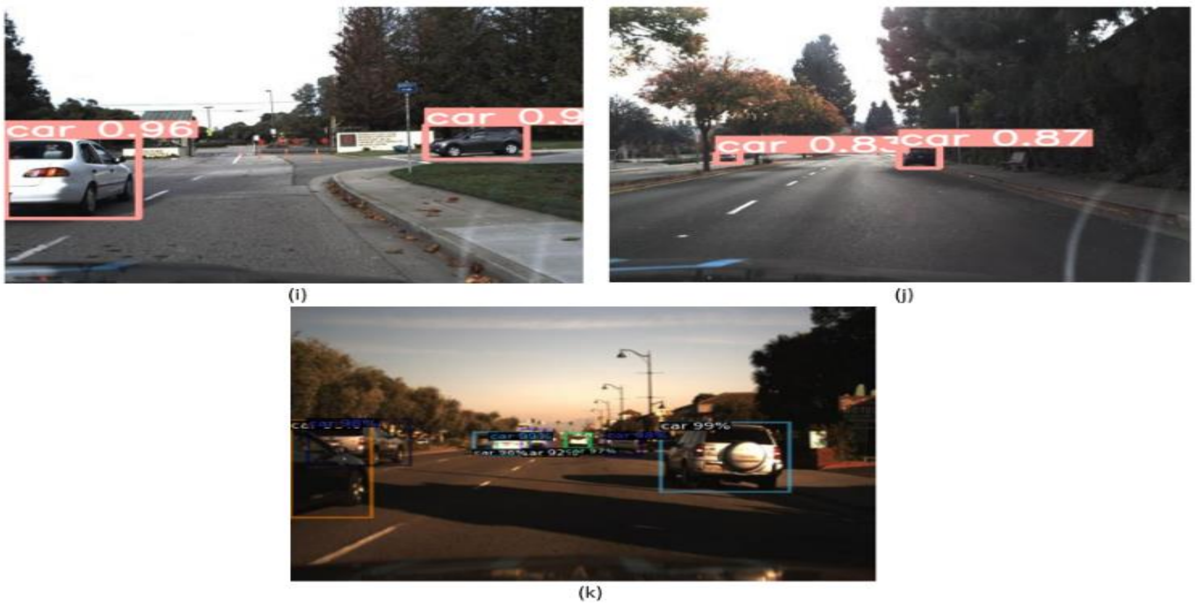


Figure 7: Detection result illustrated on sample test image.
(i): Yolov5x, (j): Yolov5l, (k): Faster R-CNN.

Conclusion

This research paper presents a comprehensive comparison of various object detection algorithms suitable for detecting objects on the street. Eleven algorithms were thoroughly trained and evaluated using the Roboflow Self-Driving Car dataset. The findings highlight that Faster R-CNN emerges as the most accurate algorithm among the tested models, while YOLOv5n stands out as one of the fastest in detecting objects at street level. Although the Faster R-CNN model achieves high accuracy by performing object detection in two steps, its speed is considerably slower, making it challenging to deploy in real-time applications. On the other hand, the YOLOv5 model proves to be the preferred choice for real-time applications due to its ability to deliver relatively accurate results within a reasonable processing time.

By emphasizing these key findings, we not only validate earlier research but also provide a nuanced perspective that combines accuracy and speed considerations, essential for decision-making in real-world applications. Our research acts as a bridge between prior works that have largely focused on individual aspects of object detection algorithms. Looking ahead, our commitment to advancing object detection models remains steadfast. We intend to build upon these findings by developing novel algorithms that strike a balance between accuracy and speed, further enhancing their suitability for street-level applications. Additionally, our plan involves expanding the dataset to encompass a broader range of street-level images, enhancing the robustness of our models. Through continuous exploration and evaluation, this research lays the foundation for improved object detection capabilities in street-level scenarios, contributing to the ongoing evolution of computer vision and autonomous systems.

In summary, our study not only reaffirms the importance of earlier research but also advances the field by providing a nuanced understanding of algorithm performance in the context of street-level object detection, offering practical insights for real-world applications in autonomous driving and surveillance.

References

[1] Assoc. Prof. Dr. S. Kumar Selvaperumal, 2018, Review of car make & model recognition systems, août 2018.

[2] Bie et al, 2023, Real-time vehicle detection algorithm based, l.pdf.

[3] C.-Y. Wang, A. Bochkovskiy, et H.-Y. M. Liao, 2022, YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, DOI: http://arxiv.org/abs/2207.02696

[4] H. Zhang, X. Bai, J. Zhou, J. Cheng, et H. Zhao, 2013, Object Detection Via Structural Feature Selection and Shape Model, IEEE Trans. Image Process., vol. 22, no 12, p. 4984-4995, déc. 2013, doi: 10.1109/TIP.2013.2281406.

[5] I. Laptev, 2009, Improving object detection with boosted histograms », Image Vis. Comput., vol. 27, no 5, p. 535-544, avr. 2009, doi: 10.1016/j.imavis.2008.08.010.

[6] J. Luo et al., 2021, Real-World Image Datasets for Federated Learning, arXiv, 5 janvier 2021. doi: 10.48550/arXiv.1910.11089.

[7] J. Redmon, S. Divvala, R. Girshick, et A. Farhadi, 2016, You Only Look Once: Unified, Real-Time Object Detection, in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, juin 2016, p. 779-788. doi: 10.1109/CVPR.2016.91.

[8] J. Pont-Tuset, P. Arbelaez, J. T. Barron, F. Marques, et J. Malik, 2017, Multiscale Combinatorial Grouping for Image Segmentation and Object Proposal Generation, IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no 1, p. 128-140, janv. 2017, doi: 10.1109/TPAMI.2016.2537320.

[9] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, et A. W. M. Smeulders, 2013, Selective Search for Object Recognition, Int. J. Comput. Vis., vol. 104, no 2, p. 154-171, sept. 2013, doi: 10.1007/s11263-013-0620-5.

[10] J. Nelson, J. S. JUN, 2020, Min Read, YOLOv5 is Here, Roboflow Blog, 10 juin 2020. https://blog.roboflow.com/yolov5-is-here/.

[11] K. E. A. van de Sande, J. R. R. Uijlings, T. Gevers, et A. W. M. Smeulders, 2011, Segmentation as selective search for object recognition, in 2011 International Conference on Computer Vision, Barcelona, Spain, Nov. 2011, p. 1879-1886. doi: 10.1109/ICCV.2011.6126456.

[12] K. Kang et al., 2018, T-CNN: Tubelets with Convolutional Neural Networks for Object Detection from Videos, IEEE Trans. Circuits Syst. Video Technol., vol. 28, no 10, p. 2896-2907, oct. 2018, doi: 10.1109/TCSVT.2017.2736553.

[13] K. He, X. Zhang, S. Ren, et J. Sun, 2015, Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition », IEEE Trans. Pattern Anal. Mach. Intell., vol. 37, no 9, p. 1904-1916, sept. 2015, doi: 10.1109/TPAMI.2015.2389824.

[14]Kreienbühl et Koltun, 2014 , Geodesic Object Proposals.pdf, https://www.philkr.net/media/kraehenbuehl2014geodesic.pdf

[15] L. Liu, A. Finch, M. Utiyama, et E. Sumita, 2016, Agreement on Target-Bidirectional LSTMs for Sequence-to-Sequence Learning, Proc. AAAI Conf. Artif. Intell., vol. 30, no 1, Art. no 1, mars 2016, doi: 10.1609/aaai.v30i1.10327.

[16] M. G. Naftali, J. S. Sulistyawan, et K. Julian, 2022, Comparison of Object Detection Algorithms for Street-level Objects, arXiv, 24 août 2022, http://arxiv.org/abs/2208.11315

[17] M. Horvat, L. Jelečević, et G. Gledec, 2022, A comparative study of YOLOv5 models performance for image localization and classification.

[18] P. Viola et M. Jones, 2001, Rapid object detection using a boosted cascade of simple features , in Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, vol. 1, p. I-511-I-518. doi: 10.1109/CVPR.2001.990517.

[19] R. Girshick, 2015, Fast R-CNN. arXiv, 27 septembre 2015. Consulté le: 7 octobre 2022, http://arxiv.org/abs/1504.08083

[20] S. Ren, K. He, R. Girshick, et J. Sun, 2016, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks ». arXiv, 6 janvier 2016, http://arxiv.org/abs/1506.01497

[21] S. Choyal et A. K. Singh, 2020, An Acoustic based Roadside Symbols Detection and Identification using Faster RCNN and SSD, in 2020 International Conference on Emerging Trends in Communication, Control and Computing (ICONC3), Lakshmangarh, India, févr. 2020, p. 1-4. doi: 10.1109/ICONC345789.2020.9117222.

[22]Uijlings et al. - 2013 - Selective Search for Object Recognition.pdf, https://ivi.fnwi.uva.nl/isis/publications/2013/UijlingsIJCV2013/UijlingsIJCV2013.pdf

[23] V. Kurilová, J. Goga, M. Oravec, J. Pavlovičová, et S. Kajan, 20221, Support vector machine and deep-learning object detection for localisation of hard exudates, Sci. Rep., vol. 11, no 1, Art. no 1, août 2021, doi: 10.1038/s41598-021-95519-0.

[24] Version Roboflow, 2022, https://universe.roboflow.com/sebasa_p-hotmail-com/self_driving_cars/dataset/1

[25] W. Liu et al., 2016, SSD: Single Shot MultiBox Detector , in Computer Vision – ECCV 2016, vol. 9905, B. Leibe, J. Matas, N. Sebe, et M. Welling, Éd. Cham: Springer International Publishing, 2016, p. 21-37. doi: 10.1007/978-3-319-46448-0_2.

[26] X. Hao, L. Bo, et Z. Fei, 2022, Light-YOLOv5: A Lightweight Algorithm for Improved YOLOv5 in Complex Fire Scenarios », p. 10.