National College of Ireland

Enhancing Object Detection in Autonomous Cars: A Fusion of YOLO and Cascade R-CNN

MSc Research Project
Data Analytics

Kalyani Deshpande
Student ID: X21215961

School of Computing
National College of Ireland

Supervisor: Mr. Aaloka Anant

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Kalyani Deshpande
Student ID:	X21215961
Programme:	Data Analytics
Year:	2023
Module:	MSc Research Project
Supervisor:	Mr. Aaloka Anant
Submission Due Date:	31/01/2024
Project Title:	Enhancing Object Detection in Autonomous Cars: A Fusion of YOLO and Cascade R-CNN
Word Count:	5238
Page Count:	17

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	Kalyani Deshpande
Date:	29th January 2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

ENHANCING OBJECT DETECTION IN AUTONOMOUS CARS: A FUSION OF YOLO AND CASCADE R-CNN

Kalyani Deshpande
X21215961

31st January 2024

Abstract

This study presents a novel hybrid implementation of an object detection system based on the YoloV4 and Cascade RCNN models. The research aims to understand the performance dynamics of these models in terms of precision, recall, Average Precision (AP), and mean Average Precision (mAP). During the comparison test, the YoloV4 model did pretty well, showing that it could handle high recall situations but had trouble keeping its precision steady. Its AP and mAP were both 0.16. It was easier for the Cascade RCNN Standalone model to keep its precision across a wider range of thresholds, as shown by its AP and mAP scores of 0.625. It was the YoloV4-Cascade RCNN hybrid model that did the best, with the highest scores (AP and mAP of 0.79) and a great balance between accuracy and recall. Combining different object detection methods to improve overall detection accuracy works, as shown by this hybrid model's excellent performance. Understanding the study's results is important for making progress in real-world applications like self-driving cars.

1 Introduction

The advent of increased processor powers equipped with the improvement in imaging technology has seen a sharp rise in research towards building cars that do not require human drivers to drive. These self-driving or autonomous cars are considered to be a game-changing technology. These autonomous vehicles provide an important aspect of road transportation in that they can avoid the errors that humans tend to make while driving. These errors can cause both financial and human losses Faisal et al. (2019). Several companies, including Waymo, Baidu, Cruise, etc., are working towards building these autonomous cars. Tesla, on the other hand, introduced the Tesla Autopilot technology that makes use of the Light Detection and Ranging (LiDAR), cameras and other hardware to provide semi-autonomy in driving, mainly in navigation and lane changingIngle and Phute (2016). On the other hand, fully autonomous vehicles are being designed for their use in multitudes of businesses ranging from freight transport and taxi services (Faisal et al., 2019). This technology is a composite of three stages of operations that involve object detection, action prediction and acting. The object detection is considered to be the most important aspect in autonomous driving Hnewa and Radha (2021).

The motivation for undertaking lies in the limitations associated with the current object detection techniques. Current models catering to object detection in autonomous vehicles, exemplified by standard iterations of YOLO (You Only Look Once) and R-CNN (Region-Based Convolutional Neural Networks), have undeniably achieved noteworthy progress in the real-time identification and classification of objects. However, these models are not exempt from limitations. For instance, YOLO, despite its acclaimed speed, occasionally compromises accuracy, especially when confronted with small or overlapping objects. Conversely, R-CNN variants, renowned for their precision, often lag in processing speed, a critical consideration for real-time decision-making in autonomous driving scenarios. The specific shortcomings of YOLO and R-CNN models, and these limitations manifest in real-world scenarios, potentially compromising safety. These limitations manifest tangibly in practical scenarios as accidents or near-misses arising from failure to detect and respond to road objects. Inaccurate or delayed object detection instances translate into real-world consequences, posing risks to passenger safety and disrupting efficient traffic flow.

This study aims to develop a hybrid model based on the YoloV4 and RCNN models to use both technologies' strengths in developing an object detection algorithm. This approach provides a comprehensive and effective solution that is well-suited to the complex and dynamic environment in which autonomous vehicles operate. The intention of this hybrid model is to address the inherent trade-offs between speed and accuracy. This hybrid model capitalises on the fast detection capabilities of YOLOv4 and the accuracy of Cascade R-CNN. Integrating these models facilitates the creation of a system that expeditiously identifies potential objects using YOLOv4 and subsequently refines these detections with higher accuracy through the Cascade R-CNN framework.

In order to fulfil the aim of the research, the following research question has been put forth:

"To what extent does the integration of YOLO and cascade R-CNN increase mAP (mean Average Precision) in detecting objects for a self-driving car?"

The rest of the report is arranged in 5 chapters. Chapter 2 discusses the literature review of the state-of-the-art systems, Chapter 3 discusses the methods used to fulfil the objective of the study, Chapter 4 discusses the system architecture, and Chapter 5 discusses the implementation of the system. The system is evaluated and discussed thoroughly in Chapter 6 and Chapter 7 concludes the study with key takeaways and provides future directions for further research in the field.

2 Related Work

This section of the report analyses past literature in the field of object detection in autonomous vehicles. The section provides overall understanding of the different methodologies, deep learning models and their performance in object detection for autonomous vehicles.

Zhou et al. (2021) augment the Faster-RCNN object recognition method for automated driving, integrating spatial attention, deformable convolution, and an enhanced feature pyramid structure. These integrations aim to improve object recognition by addressing false and missing discoveries. The method employs a ResNet-50 backbone for enhanced feature extraction, particularly benefiting the detection of tiny objects. The side-aware boundary localization further enhances frame regression in the process.

Advancements include three cascade detectors collaborating to minimize IOU threshold mismatches, contributing to improved overall object detection accuracy. The proposed method employs Soft-NMS for determining optimal bounding boxes, refining them effectively compared to other methods. Empirical findings on COCO2017 and BDD100k datasets demonstrate a practical 7.7% and 4.1% accuracy improvement for detecting tiny and obstructed objects, showcasing efficacy in real-world automated driving scenarios. Dhayighode et al. (2022) emphasize the importance of accurate object detection in autonomous vehicles and discuss challenges addressed by computer vision. Efforts to enhance model efficiency for Automatic Driving Systems (ADSs) involve strategies contributing to realizing the full potential of ADSs. The introduction of a weighted bidirectional feature pyramid network (BiFPN) facilitates effective multi-scale feature composition, offering advantages over alternative techniques. An integrated scaling strategy optimizes the overall model performance, resulting in a 4×-9× reduction in model size and 2×–4× faster GPU processing, enhancing the efficiency of AV object recognition.

Li et al. (2022) presents an enhanced Faster R-CNN technique for precise traffic sign identification. AutoAugment technology, attention-guided context feature pyramid network (ACFPN), and a ResNet50-D feature extractors are integrated to improve traffic sign identification. The ACFPN minimizes the loss of contextual information, contributing to overall improvement. On the CCTSDB dataset, a mean average accuracy of 99.5% and 29.8 frames per second highlights superior performance compared to standard approaches, with implications for real-world applications. Carranza-García et al. (2021) introduce an improved 2D object detector based on Faster R-CNN for driverless cars. Evolutionary algorithms for anchor optimization and a perspective-aware methodology address challenges in anchor production and performance decline in minority classes. The proposed module, integrating spatial data of potential areas, enhances accuracy and addresses related challenges. An ensemble approach shows a 9.69% mAP improvement, effectively enhancing mean Average Precision and overall model performance. Fang et al. (n.d.) tackle challenges in segmenting and detecting targets in autonomous driving scenarios, replacing ResNet with the ResNeXt network in Mask R-CNN. Bottom-up path augmentation in the Feature Pyramid Network (FPN) efficiently contributes to feature fusion, offering advantages over alternative methods. The use of the "CIoU loss" reduces errors and accelerates model convergence, demonstrating efficacy with a 62.62% mAP for target recognition on the CityScapes dataset.

Hu et al. (2020) categorize 3D object recognition techniques in autonomous driving into lidar-based, stereo-image-based, and monocular image-based approaches. Their proposed technique combines cascading geometric constraints with monocular pictures, contributing to robust detection. Monocular images are utilized for 3D object recognition, overcoming associated challenges. Shi et al. (2022) introduce Sparse R-CNN 3D (SRCN3D), a two-stage fully-sparse detector for tracking and recognizing moving objects in autonomous driving scenarios. SRCN3D addresses computational efficiency challenges through sparse queries and attention mechanisms, with a special sparse feature sampling module contributing to effective box refining and overall efficiency.

Mahmoud and Nasser (2021) focus on real-time accuracy in object identification for autonomous vehicles, utilizing a dual architecture combining a highly accurate multiclass CNN with YOLOv3. The modified Feature Pyramid Network (FPN) and Region-Based Convolutional Neural Networks (Faster R-CNN) enhance microscopic item recognition, with the proposed Sniffer Faster R-CNN (SFR-CNN) camera-LiDAR sensor fusion architecture addressing challenges related to the regional proposal network (RPN). Cai et al.

(2021) propose the YOLOV4-5D one-stage object detection system for improved accuracy and genuine real-time operation in autonomous vehicles. Modifications to the

CSPDarknet53_dcn(P) backbone network enhance accuracy, and the inclusion of deformable convolution contributes to this improvement. Five scale detection layers and the PAN ++ feature fusion module collectively address small object detection problems.

Islam and Karimoddini (2022) introduce a fusion framework combining semantic segmentation networks with asymmetric inferences from object detectors to improve pedestrian identification. This framework addresses challenges associated with detecting unexpected abnormalities and barriers, concurrently improving efficiency through reduced runtime costs. Thorough assessments demonstrate the effectiveness and resilience of the fusion architecture, outperforming previous approaches. Carranza-García et al. (2021) assess the effectiveness of 2D object identification systems for autonomous vehicles, comparing two-stage detectors (Faster R-CNN) and one-stage detectors (RetinaNet, FCOS, YOLOv3). Findings consider performance factors, especially in identifying minority classes.

Carranza-García et al. (2022) presents a camera and LiDAR data fusion architecture for object identification in autonomous driving. Integration of an effective LiDAR sparse-to-dense completion network addresses LiDAR data sparsity, bringing benefits to overall object identification models. Dai (2019) introduces HybridNet, a two-stage cascade object identification system for vehicle detection in autonomous driving. Leveraging regression techniques, HybridNet achieves fast and accurate vehicle detection, offering specific advantages over other methods. Lee et al. (2021) contribute to intelligent transportation systems with a focus on object identification methods for monocular cameras in automated driving systems. "You Only Look Once (YOLO)" V2 and "Faster R-CNN" models serve the goal of improving safety, each providing unique advantages. Juyal et al. (2021)suggests a method for anonymous activity detection in nearby vehicles, emphasizing "deep learning" techniques, particularly "You Only Look Once (YOLO)," for real-time irregularity detection. Challenges associated with detecting unexpected abnormalities and barriers are addressed.

Jia et al. (2023)built upon "YOLOv5" to create a quick and precise object detector for autonomous driving. Structural "re-parameterization (Rep)" contributes to increased precision and speed, with specific enhancements improving model recognition for tiny cars and pedestrians. Peng et al. (2022) acknowledge challenges in accurate environment perception in autonomous vehicles with a single sensor. Multi-sensor fusion strikes a balance between AV cost and detection accuracy, with various fusion methodologies classified based on image and point-cloud fusion.

Zhao et al. (2018) present "CFENet," an enhanced one-stage object detector focusing on detecting small objects and sustaining high detection speed for autonomous driving applications. The "Comprehensive Feature Enhancement (CFE)" module contributes to improved performance, outperforming other techniques like "SSD" and "RefineDet." Chen et al. (2021) thoroughly examine popular object identification architectures and feature extractors for autonomous driving. Evaluation and comparison of "Faster R-CNN," "R-FCN," 'SSD," "ResNet50," "ResNet101," "MobileNet_V1," "MobileNet_V2," and

"Inception_ResNet_V2" consider accuracy, speed, and memory consumption. Liu et al. (2023) introduce "BiGA-YOLO," a lightweight network derived from "YOLOv5" for object detection in autonomous driving. "Coordinate Attention," "Ghost-Hardswish Conv" module and the "BiFPN" structure collectively contribute to improved performance in

detecting objects of different sizes in dynamic situations. Yang et al. (2020) present a system for real-time object recognition and range in autonomous driving, focusing on lane detection using "RGB-D" pictures. The system utilizes two networks for accurate data analysis, with the "multi-GPU" synchronization technique playing a role in improving speed and accuracy.

3 Methodology

This section of the report details the methods used in the fulfilment of the study objective of developing a robust and accurate system of object detection for autonomous vehicles as a data mining approach. A research methodology can be implemented via two techniques viz. Cross-Industry Standard Process for Data Mining (CRISP-DM) and Knowledge Discovery in Databases (KDD). While the CRISP-DM is an industry standard and involves steps that involve deploying the developed system in real-world applications, the KDD methodology is more research-based and does not involve the deployment of the system. Hence, the KDD methodology is implemented in the study as it involves studying the effectiveness of the system in object detection and contributing to the existing knowledge of object detection in autonomous vehicles. Figure 1 below shows the steps involved in the KDD methodology.

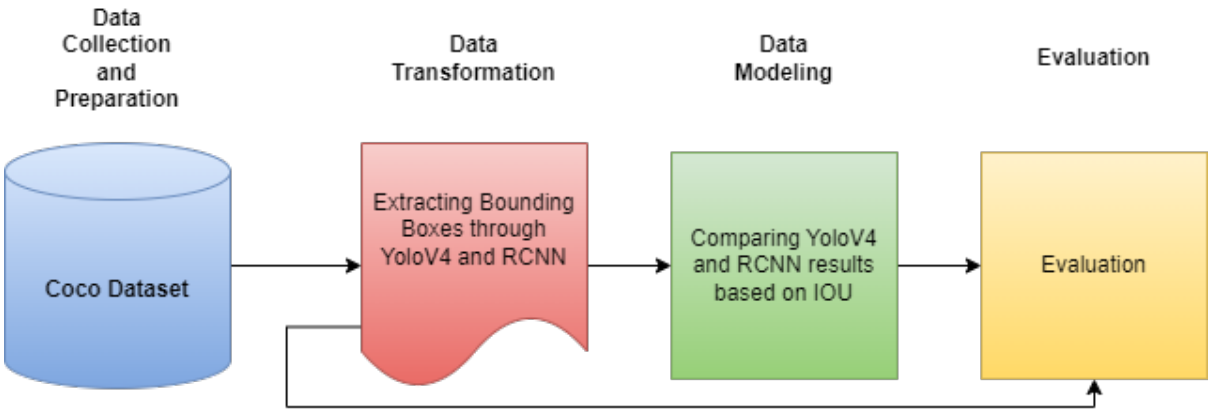

Figure 1: Modified KDD methodology

The figure above shows that the KDD methodology is modified for the study as the data processing part of the methodology is not needed as the dataset is already processed.

3.1 Data Collection

The study makes use of the Common Object in Context (COCO) dataset Lin et al. (2014). The COCO dataset is a collection of 80 common everyday objects ranging from small objects like needles, spoons, knife to large objects like cars and aeroplanes including humans. There are thousands of instances for each category making it a vast dataset. The dataset is not directly used in the study, but the models implemented are pre-trained on the dataset. Some sample images from the dataset are shown in Figure 2 below.

Figure 2: Sample images from the COCO dataset

3.2 Modeling

In the modelling part, the following detectors that are pre-trained on the COCO dataset are used to obtain the bounding boxes housing the object of interest in a query image. The accuracy of the detection is calculated using a metric known as Intersection over Union (IoU). It involves comparing the ground truth bounding box to the predicted bounding box. The IoU for a detector is obtained as a ratio given below:

Figure 3 below shows the IoU over an image using object detectors.

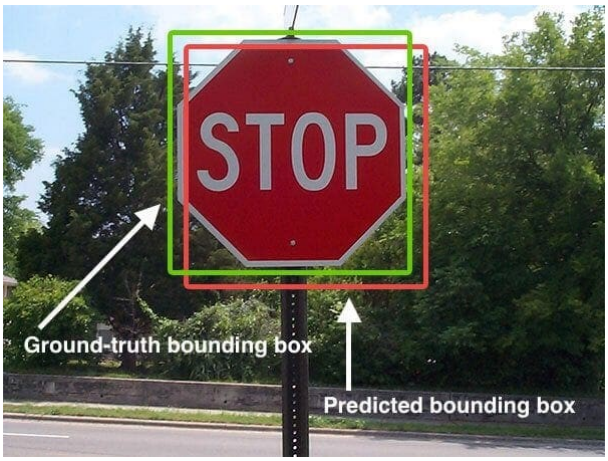

Figure 3: IOU in an image

(Source: www.pyimagesearch.com)

This metric is used to evaluate the performance of the detector at work. Figure 4 below shows the significance of different IoUs.

Figure 4: Evaluating IoU

(Source: www.pyimagesearch.com)

3.2.1 You Only Look Once, Version 4 (YoloV4) Detector

YoloV4 is a cutting-edge object detection model that is known for having a great balance of speed and accuracy Mahasin and Dewi (2022). It can be used in real-time applications and is much better than its predecessors. YoloV4 is built around a convolutional neural network (CNN), which can process an entire image in a single pass and predict both the bounding boxes and the class probabilities. This one-pass detection method, which is unique to the YOLO series, makes it possible to find things very quickly. YoloV4's architecture is made up of CSPDarknet53 as the main part for feature extraction, PANet and Spatial Pyramid Pooling for better feature integration in the neck, and an anchor-based detection head Liu et al. (2018). It has many optimizations, such as Mish activation, Weighted-Residual-Connections (WRC), and Cross-Stage-Partial-connections (CSP). These improvements not only make the model more accurate at detecting things, but they also keep it relatively light and flexible so it can be used in a wide range of settings.

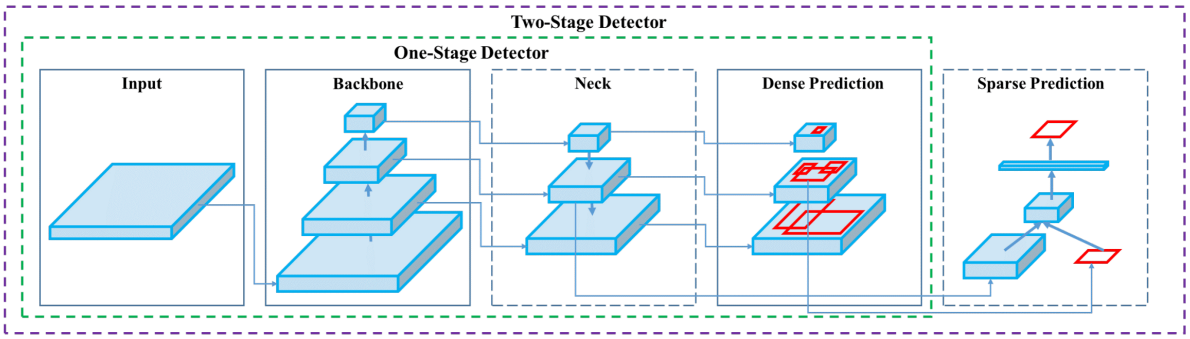

Figure 5: YoloV4 Architecture

(Source: www.ultralytics.com)

YoloV4 is better than many of its predecessors and modern competitors when it comes to training and running operations efficiently. It's made to be trainable on standard hardware, like a single GPU, so that more people can use its advanced object detection features. During training, advanced data enhancement methods such as Mosaic and CutMix are used to make the model better at adapting to different object sizes and aspect ratios Liu et al. (2021). Because of this, YoloV4 works really well in many real-time detection situations, from surveillance systems to self-driving cars and even finding anomalies in factories.

3.2.2 Cascade Region-based Convolutional Neural Network (RCNN) detector

Cascade R-CNN is a novel approach to object detection that improves on the standard Region-based Convolutional Neural Network (R-CNN) model by adding a new multi-stage architecture. This design is made to improve the detection process by fixing some of the problems that come with regular R-CNNs. Object detection is usually done in just one step in standard R-CNN models, which means that recall and precision are often not as good as they could be. Cascade R-CNN comes up with a creative way to get around this problem by using a series of detectors, each trained with higher Intersection

over Union (IoU) thresholds. This method works in a certain order so that each stage improves on the predictions made by the previous stage. This makes the bounding boxes more accurate over time while keeping the recall rates high.

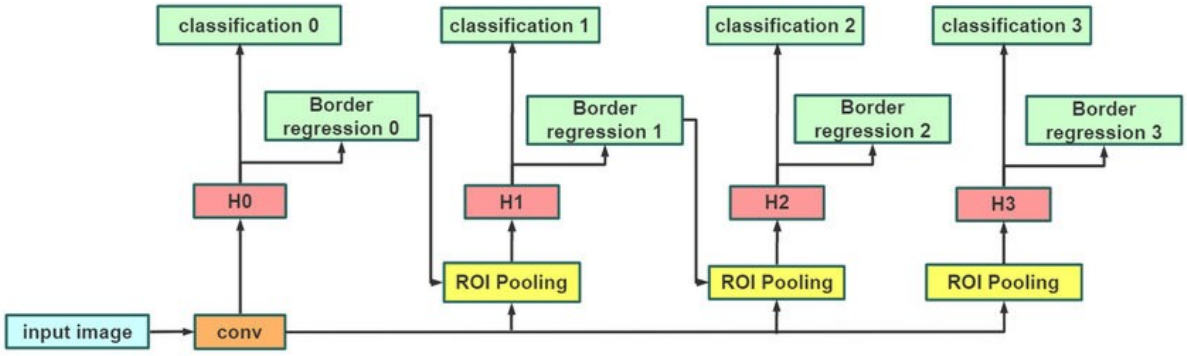

Figure 6: Framework of Cascade RCNN

(Source: www.researchgate.com)

Cascade R- CNN's architecture is unique because it can change the IoU thresholds in a way that makes the objects it finds more and more in line with the real world as it goes through the stages. The model can handle a wide range of object sizes and shapes well thanks to this progression. This makes it strong and useful for many detection tasks. Cascade R-CNN is very complex, but it manages to find a good balance between accuracy and computational efficiency. It is used a lot in situations where pinpointing the location of things is very important, like in surveillance systems, medical image analysis, and self-driving cars. Cascade R-CNN, like many other advanced object detection models, can be hard on computers because it needs a lot of resources for training and inference, which could be a problem for some uses.

3.2.3 Hybrid Model based upon YoloV4 and Cascade RCNN

The hybrid model based on the two detectors is developed through the strategy given below: It integrates an IoU-based scoring mechanism to assess the overlap between bounding boxes generated by YOLOv4 and Cascade R-CNN. In the event of identical IoU values but divergent confidence scores, the box with the higher confidence score is determined, and this conflict resolution strategy is employed. This role does a predetermined confidence score threshold play in this determination, and it is integrated into the evaluation process. To contend with computational complexity, the model is implemented within a GPU-accelerated environment. This strategic choice was made, and it facilitate real-time processing capabilities, especially when managing the increased computational load introduced by Cascade R-CNN. So, this implementation choice impacts the overall efficiency and effectiveness of the object detection model.

3.3 Evaluation Metrics

The object detection in the study is evaluated using the mAP score. mAP score is a widely used metric, which stands for "mean Average Precision," to judge how well object

detection models work, especially when they have to find more than one object in an image. It gives a full picture of how accurate a model is by looking at both its precision and recall across different types of objects and different levels of confidence in detection.

mAP score is based on Average Precision (AP). AP finds the mean value of the precision and recall values over the range [0, 1]. The model's detection outputs are used to make a precision-recall curve for each class, which is then used to find the area under the curve. Precision is the percentage of true positives found out of all the positives the model found, and recall is the percentage of true positives found out of all the real positives in the data. In object detection tasks, a finding is usually thought to be a true positive if its Intersection over Union (IoU) with a ground truth bounding box is greater than a certain value, which is usually 0.5.

Then, to get mAP, the AP values for all classes in the dataset are averaged. The mAP score is very useful because it gives a single number that measures both how reliable the model is (through accuracy) and how well it can find all relevant objects (through recall). In object detection, this balance is very important because missing an important object or labelling a non-important object wrongly can be expensive in autonomous vehicles.

4 Design Specification

This section of the study describes the architecture of the system implemented. The system architecture employed in the study is shown in Figure 7 below.

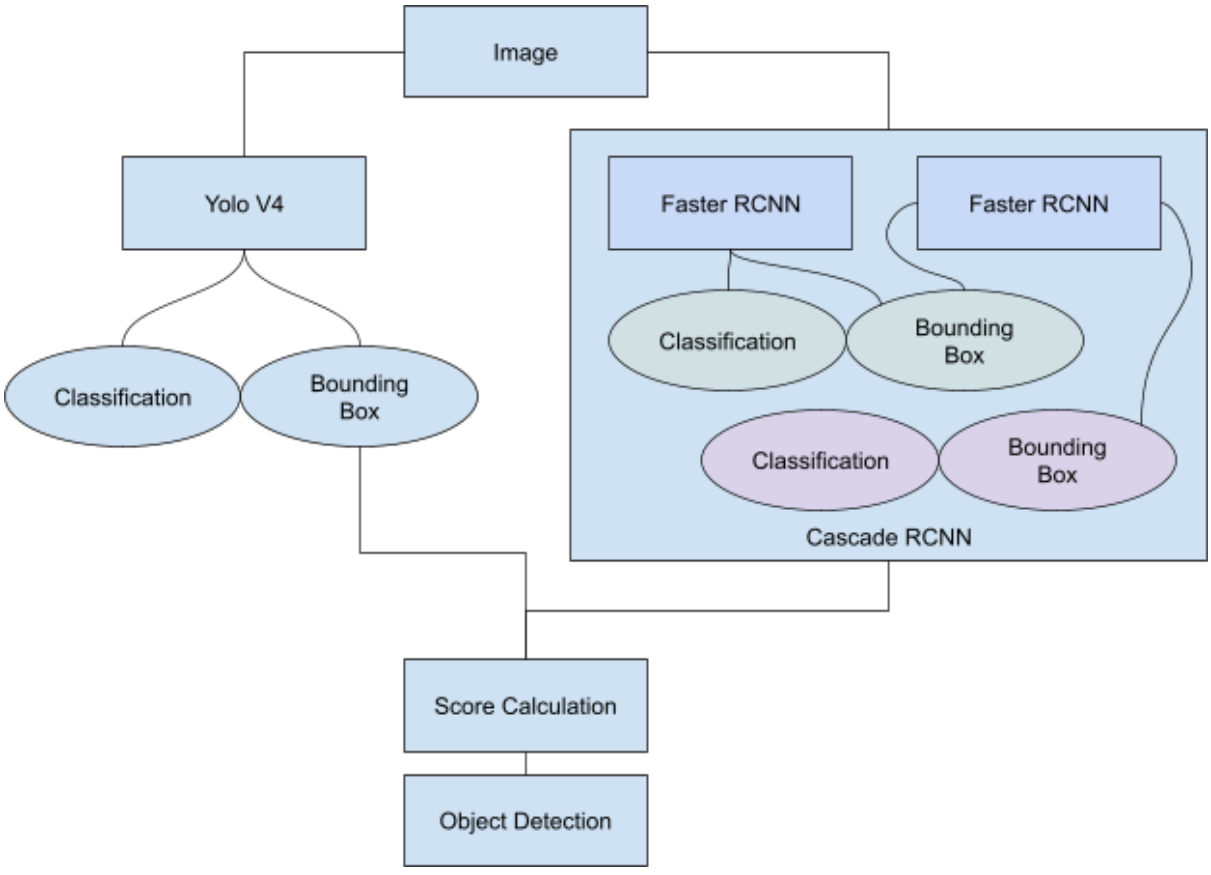

Figure 7: Architecture of the system

The process flow of the system can be understood from the architecture above. The

Faster R-CNN box is a component of the Cascaded R-CNN algorithm. It does two tasks: classification, which determines the type of object in a suggested region, and bounding box processing, which precisely locates the object. The relationship between these two is based on the model's ability to anticipate both the item class and accurately adjust the bounding box coordinates at the same time. Cascade R-CNN enhances the Faster R-CNN methodology by incorporating many iterative stages, each of which contributes to the enhancement of object detection precision. By employing a series of sequential procedures to enhance the precision of the bounding box and classification, guarantees improved accuracy in the detection process at each stage. In the presented study, pretrained detectors are first used to obtain the bounding box and classification. The bounding box is necessary to identify the IoU and confidence of the detector. The bounding boxes obtained from the detectors are compared to the ground truths provided. Based on the IoUs obtained for the detectors, through a comparison with the threshold of 0.5, the classification as either 1 or 0 is obtained. Based on the classification, the mAP score is then evaluated. The implementation of the system is further detailed in the upcoming chapter.

5 Implementation

5.1 Environmental Setup

The environment to perform the study is created as a Jupyter environment in Google Colab and is programmed in Python programming language. Important libraries such as Open-CV (known as cv2 in Python), Scikit-learn, OpenMIM, mmdet, mmengine, and mmcv are used to implement the system. The Scikit-learn library is already available for use in Google Colab whereas the remaining libraries used are installed in line through '!pip install' operation. The only data given as input to the system is the image over which the object detection is to be performed.

5.2 Data Handling

The image given to the system is read using the cv2's imread() function. The class names for the object detections are also read into the environment using a function developed to read the contents of the file 'coco.names'.

5.3 Implementation of the YoloV4 detector

The YoloV4 model is loaded with pre-trained weights (yolov4.weights) and configuration (yolov4.cfg) files using the cv2 library's dnn module. A function *detect_objects*() is defined to perform object detection on images. It converts images into a blob, sets it as the input to the network, and gets the output from the specified output layers that are obtained from the detector configuration file and the weights. A function *get_boxes_yolo*() is defined that processes the network's output to extract bounding boxes, confidences, and class IDs. It checks for the confidence score in the detection and rejects those below a threshold of 0.5. The overlapping boxes from the detection are filtered out, keeping only those with the highest confidence scores through a method called non-maximum separation.

5.4 Implementation of the RCNN detector

The RCNN detector implementation starts with the model's initialisation with the configuration file and checkpoint file download. The checkpoint file contains the pre-trained weights. The configuration file is available in the mmdetection object detection toolbox that is cloned using the git clone option from the git repository. The $inference\_detector()$ function from mmdet library is used to perform object detection on the same query image that takes the model and image path as inputs. The bounding boxes and prediction scores are obtained for the objects detected in the images by the detector. A threshold of 0.5 is applied on the scores to detect the class labels associated with detected objects from the image.

5.5 Implementation of the YoloV4 and RCNN hybrid

A custom function *compare_bounding_boxes*() is defined to compare the bounding boxes from YoloV4 and Cascade RCNN. It uses the IoU metric to find matches and keep the best bounding boxes based on a threshold. The function accepts yolo and rcnn bounding boxes as inputs, along with the confidence scores (yolo confidences and rcnn confidences). Additionally, an iou threshold parameter is set; this parameter's value is 0.5 by default. If two models' bounding boxes score the same in the Intersection over Union (IoU) test, this threshold is used to decide if the two models represent the same thing. It measures how much two bounding boxes overlap; a higher IoU means more overlap. Bounding boxes from YOLO and Cascade R-CNN are iterated over in the function using nested loops. IoU is found for each pair using the *calculate_iou*() function. Two models have found the same object in an image if their IoU values are higher than the set threshold. The IoU scores and confidence scores from both models are then added to the matched boxes list along with these pairs. In addition, the function labels the pair with the highest IoU score as the best match and keeps track of it. This is especially helpful for figuring out which detection is the most accurate if both models detect the same object. Lastly, the function gives back two outputs: a list of all the matched bounding boxes that met the IoU threshold; and the single best match with the highest IoU score. Comparing and analysing how well the YOLO and Cascade R-CNN models work on the same set of images can be made easier with this output. The relative strengths and weaknesses of each model in detecting different objects can be found by looking at which model gives higher confidence scores or better IoU in matched detections.

6 Evaluation and Discussion

The results obtained for the implementation of the study are discussed in this section. Table 1 below shows the results obtained from the experimentation.

Model	Precision	Recall	AP value	mAP value
YoloV4	[0.33 0 1]	[1 0 0]	0.16	0.16
Cascade RCNN	[0.5 0.5 1]	[1 0.5 0]	0.625	0.625
YoloV4-Cascade RCNN Hybrid	[0.67 0.5 1 1]	[1 0.5 0.5 0]	0.79	0.79

Table 1: Comparison of model performances

The presented results give a full analysis of three object detection models: YoloV4 Standalone, Cascade RCNN Standalone, and a hybrid of YoloV4 and Cascade RCNN. These models are evaluated on their accuracy, recall, Average Precision (AP) value, and mean Average Precision (mAP) value. These are all important metrics for judging how well object detection systems work. When looking at the YoloV4 Standalone model, the precision and recall values change a lot when the thresholds are changed. The precision starts at 0.33 and the recall is 1. This means that when the model tries to find as many relevant objects as possible, it gets a lot of false positives, which makes the precision lower. The precision changes as the recall goes down, dropping to 0 before reaching a perfect score at the lowest recall. The AP and mAP values for YoloV4 are both 0.16, which means that it does about average at finding the right balance between accuracy and recall. The Cascade-RCNN Standalone model, on the other hand, performs better and more consistently. For higher recall levels, the precision values stay the same at 0.5, which means that the ability to detect is balanced. As with YoloV4, though, the highest level of accuracy at the end comes at the cost of no recall at all. The AP and mAP values for Cascade RCNN are both much higher at 0.625, which shows that it is better at finding objects than YoloV4 Standalone. The best results are obtained for the YoloV4-Cascade RCNN hybrid model. It starts with a higher initial precision of 0.67 while keeping total recall, which means it can find more true positives and fewer false positives. This balance stays the same even as recall goes down, and the model stays very accurate the whole time. The hybrid model has the best overall performance, as shown by its AP and mAP scores of 0.79, which are the highest of the three models. This suggests that using YoloV4 and Cascade RCNN together makes the best of both models, making a more accurate and dependable system for object detection.

Figure 8 below shows query image 1 given to the system.

Figure 8: Query Image 1

The performance of the model on the query image above is shown in Figure 9 below.

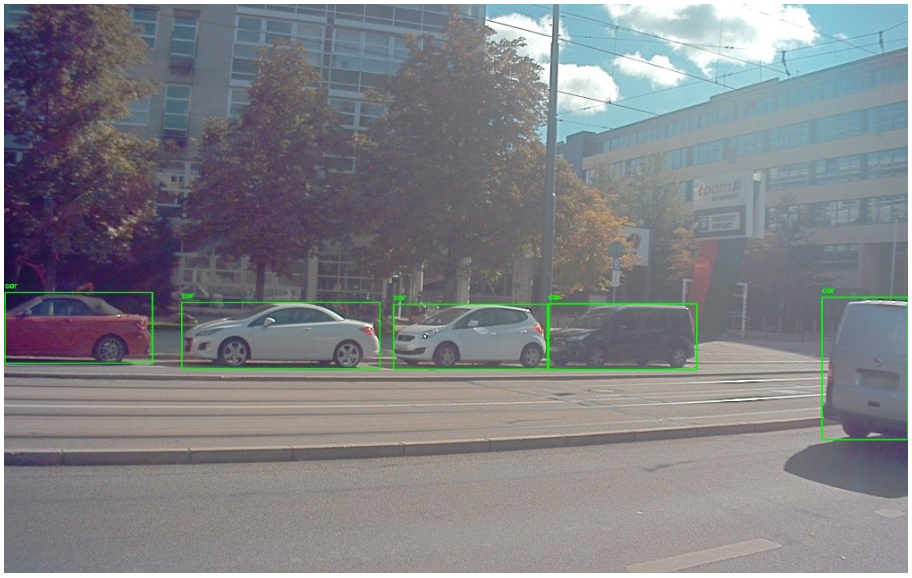

Figure 9: Performance of the implemented detector

From the figure above, it can be seen that the model has been able to perfectly detect all the cars in the image. Figure 10 below shows the second query image given to the model.

Figure 10: Query Image 2

The performance of the model on the second query image is shown in Figure 11 below.

Figure 11: Performance of the model on query image 2

From the figure above, it can be seen that the model performed decent detection on the scene provided. The image consists of a large number of objects such as cars, bicycles, persons etc. The model correctly detected the majority of them but with some exceptions. A person crossing the road missed by the detector highlighting the difficulty that exists in developing an object detection system.

7 Conclusion and Future Work

The study that compares YoloV4, Cascade RCNN, and a model that combines YoloV4 and Cascade RCNN is a big step forward in the field of object detection. The study's results are very important because they show how different models work in different situations and how their strengths can be combined to make them work better. This study only improves the accuracy as the speed of detection will be slower compared to standalone the Cascade RCNN as well as the YoloV4. This is because the object detection in the study is being performed as a combination of results from both models. However, a significant improvement is obtained for the model implemented as shown by the mAP and precision, recall values. The model created through the combination is also expected to be fast as two fast models are used in the combination in the study. Yolo is itself a fast algorithm and Faster RCNN also shows significant speed improvement as compared to other RCNN models. So, the speed aspect of the model is considered by incorporating these models. When it comes to object detection, precision and recall are important metrics that are often linked to each other. The YoloV4 model is known for being fast and efficient. It worked well when there was a high recall, but it struggled with precision fluctuating. This makes it good for situations where finding all possible objects is very important, but not so good when finding things accurately is very important. Cascade RCNN, on the other hand, was very stable in its accuracy across a range of thresholds, though it sometimes had lower recall. This trait is necessary when reducing false positives is important, even if it means missing some true positives. However, the YoloV4-Cascade RCNN hybrid model goes beyond these individual flaws and shows a great balance between accuracy and recall. In the real world, where the cost of false

positives and false negatives is high, this balance is very important. For example, in autonomous vehicle navigation, it can be very bad if the vehicle misses an obstacle (low recall) or thinks that a harmless object is an obstacle (low precision). In the same way, it is important for medical imaging to accurately identify pathological features without overdiagnosing them so that patients can get the best care. The way the study combined the best features of YoloV4 and Cascade RCNN shows how useful hybrid models can be in finding difficult objects. It shows that the future of object detection is not just making new models, but also putting together smart combinations of models that have already been used successfully. This plan could make object detection systems stronger, more flexible, and more effective, so they can be used in a wider range of situations.

Future Work To build on what this study found, more research can be focused on a few main areas: For the hybrid model to be used in real-time applications, especially on devices with limited processing power, it will need to be streamlined so that it works faster and more efficiently. Adding more difficult and varied datasets to the tests will help figure out how stable and usable the models are in a range of settings and with different kinds of objects. It is important to test and deploy these models in the real world in order to see how well they work in practice and to make them perfect for specific use cases. To conclude, the study not only compares how well object detection models work now, but it also shows how hybridization can be used to make them work better. The future of object detection lies in making models that are not only accurate and useful but also flexible and responsible, so they can adapt to the changing needs of different fields and everyday life.

References

Carranza-García, M., Galán-Sales, F. J., Luna-Romera, J. M. and Riquelme, J. C. (2022). Object detection using depth completion and camera-lidar fusion for autonomous driving, *Integrated Computer-Aided Engineering* **29**(3): 241–258.

Carranza-García, M., Lara-Benítez, P., García-Gutiérrez, J. and Riquelme, J. C. (2021). Enhancing object detection for autonomous driving by optimizing anchor generation and addressing class imbalance, *Neurocomputing* **449**: 229–244.

Dai, X. (2019). Hybridnet: A fast vehicle detection system for autonomous driving, *Signal Processing: Image Communication* **70**: 79–88.

Dhayighode, A. R., Subramanian, R. and Sunagar, P. (2022). Multi-scale fusion-based object detection network for advance driver assistance systems, *International Conference on Innovations in Computational Intelligence and Computer Vision*, Springer Nature Singapore, pp. 233–251.

Faisal, A., Kamruzzaman, M., Yigitcanlar, T. and Currie, G. (2019). Understanding autonomous vehicles, *Journal of transport and land use* **12**(1): 45–72.

Fang, S., Zhang, B. and Hu, J. (n.d.). Improved mask r-cnn multi-target detection and segmentation for autonomous driving in complex scenes, *Sensors* **23**(8).

Hnewa, M. and Radha, H. (2021). Multiscale domain adaptive yolo for cross-domain object detection, *2021 IEEE International Conference on Image Processing (ICIP)*, IEEE, pp. 3323–3327.

Hu, H., Zhao, T., Wang, Q., Gao, F. and He, L. (2020). R-cnn-based 3d object detection for autonomous driving, *CICTP 2020*, pp. 918–929.

Ingle, S. and Phute, M. (2016). Tesla autopilot: Semi autonomous driving, an uptick for future autonomy, *International Research Journal of Engineering and Technology* **3**(9): 369–372.

Islam, M. M. and Karimoddini, A. (2022). Pedestrian detection for autonomous cars: Inference fusion of deep neural networks, *IEEE Transactions on Intelligent Transportation Systems* **23**(12): 23358–23368.

Jia, X., Tong, Y., Qiao, H., Li, M., Tong, J. and Liang, B. (2023). Fast and accurate object detector for autonomous driving based on improved yolov5, *Scientific reports* **13**(1): 1–13.

Juyal, A., Sharma, S. and Matta, P. (2021). Deep learning methods for object detection in autonomous vehicles, *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, IEEE, pp. 751–755.

Lee, W., Kang, M. H., Song, J. and Hwang, K. (2021). The design of preventive automated driving systems based on convolutional neural network, *Electronics* **10**(14): 1737.

Li, X., Xie, Z., Deng, X., Wu, Y. and Pi, Y. (2022). Traffic sign detection based on improved faster r-cnn for autonomous driving, *The Journal of Supercomputing* pp. 1–21.

Lin, T., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P. and Zitnick, C. (2014). Microsoft coco: Common objects in context, *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer International Publishing, pp. 740–755.

Liu, J., Cai, Q., Zou, F., Zhu, Y., Liao, L. and Guo, F. (2023). Biga-yolo: A lightweight object detection network based on yolov5 for autonomous driving, *Electronics* **12**(12): 2745.

Liu, R., Chen, Y., Wang, J. and Guo, Z. (2021). Attentive mix: An efficient data augmentation method for object detection, *2021 7th International Conference on Computer and Communications (ICCC)*, IEEE, pp. 770–774.

Liu, S., Qi, L., Qin, H., Shi, J. and Jia, J. (2018). Path aggregation network for instance segmentation, *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8759–8768.

Mahasin, M. and Dewi, I. A. (2022). Comparison of cspdarknet53, cspresnext-50, and efficientnet-b0 backbones on yolo v4 as object detector, *International Journal of Engineering, Science and Information Technology* **2**(3): 64–72.

Mahmoud, M. M. and Nasser, A. R. (2021). Dual architecture deep learning-based object detection system for autonomous driving, *Iraqi J. Comput. Commun. Control Syst. Eng* **21**(2): 36–43.

Peng, Y., Qin, Y., Tang, X., Zhang, Z. and Deng, L. (2022). Survey on image and point-cloud fusion-based object detection in autonomous vehicles, *IEEE Transactions on Intelligent Transportation Systems* **23**(12): 22772–22789.

Shi, Y., Shen, J., Sun, Y., Wang, Y., Li, J., Sun, S. and Yang, D. (2022). Srcn3d: Sparse r-cnn 3d surround-view camera object detection and tracking for autonomous driving, *arXiv preprint arXiv:2206.14451* .

Yang, J., Wang, C., Wang, H. and Li, Q. (2020). An rgb-d-based real-time multiple object detection and ranging system for autonomous driving, *IEEE Sensors Journal* **20**(20): 11959–11966.

Zhao, Q., Sheng, T., Wang, Y., Ni, F. and Cai, L. (2018). Cfenet: An accurate and efficient single-shot object detector for autonomous driving, *arXiv preprint arXiv:1806.09790* .

Zhou, Y., Wen, S., Wang, D., Mu, J. and Richard, I. (2021). Object detection in autonomous driving scenarios based on an improved faster-rcnn, *Applied Sciences* **11**(24): 11630.