

OBJECT DETECTION FOR ENSURING CAR SEATBELT BUCKLING USING THE YOLOV7 ALGORITHM

Lwando Nkuzo

Dissertation submitted in fulfilment of the requirements for the degree

MASTER OF ENGINEERING IN ELECTRICAL ENGINEERING

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment, and Information Technology

at the

Central University of Technology, Free State

Supervisor: Dr M Sibiya
Co-supervisor: Prof ED Markus

Bloemfontein

2023

DECLARATION

I, Lwando Nkuzo, with ID number _____, certify that this is my original work and that I am aware of the definition and consequences of plagiarism as defined by the university guidelines. As a result, I attest that this dissertation has been submitted to the originality check software, and I can confirm that it falls within the acceptable limits of originality. I also assert that the provided data and content of this dissertation are authentic and based on research principles as well as my engineering skills, knowledge, and understanding of the topic. In addition, I certify that this work or any portion thereof has never been submitted to any institution to obtain another qualification.

Student number: _____

Signature:

Date: August 2023

LIST OF PUBLICATIONS

1. L. Nkuzo, M. Sibiya and E. Markus, "Computer Vision-based Applications in Modern Cars for Safety purposes: A Systematic Literature Review," 2023 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa, 2023, pp. 1-9, doi: 10.1109/ICTAS56421.2023.10082722.

Link: <https://ieeexplore.ieee.org/document/10082722>

Status: Published

2. L. Nkuzo, M. Sibiya, and E. D. Markus, "A Comprehensive Analysis of Real-Time Car Safety Belt Detection Using the YOLOv7 Algorithm," Algorithms, vol. 16, no. 9, p. 400, Aug. 2023, doi: 10.3390/a16090400.

Link: <https://www.mdpi.com/1999-4893/16/9/400>

Status: Published

ABSTRACT

Background: An estimated 3287 people die in accidents every day, and about 1.35 million people die in automobile accidents every year, and nearly half of all individuals involved in these accidents were not wearing their seat belts at the time of the crash. Car drivers who disregard safety regulations are the biggest contributors to most accidents, which may sometimes lead to death or severe injuries. This statistic exists even when research has shown that buckling a seat belt reduces fatalities and severe injuries by approximately 45% and 50%. Lack of safety precautions occurs despite the law in place and the numerous safety sensors and warning indicators embedded in modern cars (Advanced Driver Assistance System (ADAS) technology).

Objectives: There is currently not a lot of research devoted to enforcing the buckling of a seat belt. Drivers and car occupants easily cheat the available seat belt technology for most cars, as they sometimes choose to buckle their car seat belts behind their backs to mute the seat belt alarm. Consequently, this dissertation presents an object detection model to monitor the buckling of a car seat belt between two (2) classes: buckled and unbuckled. The You Only Look version 7 (YOLOv7) object detection model was adopted because of its accuracy, robustness, and real-time capabilities. The main objectives of the study were to:

1. Develop and train an object detection model for a car seat belt using a graphics processing unit (GPU).
2. Deploy the training weights from a GPU platform to a Jetson Nano board for inferencing.
3. Test the deployed model using the NVIDIA DEEP STREAM object detection applications.

Design: The data was collected by recording a video of a driver and vehicle when driving in different light conditions with different apparel. Some of the data was taken from friends, and some was downloaded online. The videos were converted to images and processed so that they could be ready for annotation, training, testing, and validation of the model. A model's training was performed on a core i5 Windows computer processor (10th generation) with a GTX 1650 Nvidia GPU. The trained model weights were converted to lighter weights called tensorrt (this

was done to increase the speed and optimise the deployed model) to accommodate the Nvidia Jetson Nano platform. An inference was implemented on the Jetson Nano board in a moving vehicle to examine the real-time capabilities of the model.

Results: The present study achieved a mean average precision (mAP) of 99.6% when the threshold was set at 0.50. Furthermore, the model demonstrated an F1-score of 98%, along with precision rates of 99.7% and 99.5% for the buckled and unbuckled classes, respectively.

Conclusion: This dissertation demonstrated YOLO's potential for car seat belt buckling detection. The results showed high accuracy even under challenging conditions like light variations, partial occlusion, blurry images, and occupants wearing seat belt-like clothing. The study also showed that GPU model training saves time. The model's successful deployment on a resource-efficient platform like the Jetson Nano development board allowed for its integration into modern car systems.

Keywords: seat belt, buckled, unbuckled, computer vision, object detection, deep learning, YOLO, YOLOv7 model, cars, convolutional neural network, Nvidia Jetson Nano, camera, GPU

DEDICATION

To my late mother, Neliswa Nkuzo, sister, Thabisa Nkuzo, and brother, Thembelani Nkuzo, may your souls continue to rest in peace. You will forever be remembered for your unwavering contribution to my life.

ACKNOWLEDGEMENTS

I would like to thank the almighty God for giving me the fortitude to persevere and surrounding me with people who helped me during difficult times.

I would also like to express my gratitude to my supervisors, Dr M Sibiya and Prof. ED Markus, for their insightful guidance and for instilling optimism and hope when none existed.

In addition, I would like to thank my wife and family for their unwavering support and patience throughout my studies.

The Faculty and Department of Electrical, Electronic, and Computer Engineering at Central University of Technology (CUT) are to be commended for entrusting me with this research project.

Contents page

DECLARATION	I
LIST OF PUBLICATIONS	II
ABSTRACT	III
DEDICATION	V
ACKNOWLEDGEMENTS	VI
CONTENTS PAGE.....	VII
LIST OF FIGURES.....	X
LIST OF TABLES	XIII
LIST OF ABBREVIATIONS AND ACRONYMS.....	XIV
CHAPTER 1: INTRODUCTION	1
1.1 Background.....	1
1.2 Computer Vision and Object Detection.....	4
1.2.1 Computer Vision.....	4
1.2.2 Object Detection	6
1.3 Problem Statement.....	8
1.4 Aim and Objectives	9
1.5 Research Methodology and Design.....	9
1.6 Delimitation	10
1.7 Knowledge Contribution	10
1.8 Dissertation Structure	10
CHAPTER 2: LITERATURE REVIEW	12
2.1 Background.....	12
2.2 Object Detection models	15
2.2.1 Traditional object detection	16
2.2.2 Deep learning-based object detection	16
2.2.3 Pros and Cons of deep learning and traditional object detection.....	23

2.3 Types of object detection.....	24
2.3.1 Two-stage object detection	25
2.3.2 Single-stage Object Detectors.....	31
2.4 Performance Metrics.....	35
2.4.1 Confusion matrix and Intersection of Union (IOU).....	35
2.4.2 Recall, Average Precision, and Mean Average Precision.....	37
2.5 Optimisation	40
2.6 Overfitting and Underfitting.....	46
2.7 Related Studies	47
2.8 The summary of safety belt detection literature	52
CHAPTER 3: MATERIALS AND METHODOLOGY	55
3.1 Introduction	55
3.2 Data Collection	56
3.3 Data preprocessing and preparation	58
3.3.1 Data Annotation	58
3.3.2 Data Splitting	61
3.4 Network Design.....	62
3.4.1 Model Selection	62
3.4.2 Configuration of Hyperparameters	66
3.5 Network Training	68
3.6 Network Evaluation.....	69
3.7 Deployment	69
3.8 Summary of the Chapter.....	70
CHAPTER 4: RESULTS AND DISCUSSION.....	71
4.1 Performance Evaluation Results.....	71
4.2 Analysis of the Trained YOLOv7 Model Based on Experiments	75

4.3 Evaluating statistical observations and resilience of safety belt buckle detection in varied experimental scenarios	80
4.4 Discussion.....	81
4.5 Conclusion	84
CHAPTER 5: SUMMARY AND CONCLUSION	86
5.1 Evaluating Objectives and Key Research Contributions.....	86
5.2 Recommendations and Suggestions for Future Research.....	87
5.3 Concluding Remarks	88
REFERENCES.....	89

LIST OF FIGURES

Fig 1.1: Components of a Seat Belt.....	2
Fig 1.2: Example of a Seat Belt pillar loop.	3
Fig 1.3: Illustration of human vision and the computer vision system [8].....	5
Fig 1.4: Computer vision image interpretation process.	6
Fig 1.5: Example of a water bottle object detection.....	7
Fig 1.6: Example of detecting multiple objects.....	7
Fig 1.7: Visualisation of the presented problem.	8
Fig 1.8: Method and design process followed.....	9
Fig 2.1: A pictorial illustration of an advanced driver assistance car with infused sensors and applications or features [12].	13
Fig 2.2: An example of a traditional object detection work process (a) and a deep learning-based object detection work process (b).....	15
Fig 2.3: Deep neural network (DNN).....	17
Fig 2.4: Neural Network (NN), known as the Perceptron.....	18
Fig 2.5: SLP architecture.....	19
Fig 2.6: MLP architecture.	19
Fig 2.7: RNN architecture.	19
Fig 2.8: RBM architecture.....	20
Fig 2.9: DBN architecture.	20
Fig 2.10: CNN general architecture.	21
Fig 2.11: One stage and two stage object detection block diagrams.....	24
Fig 2.12: Object detection types.....	25
Fig 2.13: The two-stage object detection process.	26
Fig 2.14: R-CNN Model.....	26
Fig 2.15: Fast R-CNN model.	28
Fig 2.16: Faster R-CNN model.	30
Fig 2.17: Mask R-CNN framework.....	31
Fig 2.18: Single-stage object detection architecture.	32
Fig 2.19: Summary of YOLO variant history.	32
Fig 2.20: Confusion matrix used to explain performance key indicators for object detection models.	35
Fig 2.21: A pictorial illustration of an IOU concept.	37

Fig 2.22: IOU example in traffic sign detection.....	37
Fig 2.23: Optimisation flow diagram.	41
Fig 2.24: Example of a Feedforward Neural Network.....	42
Fig 2.25: Example of backpropagation of weights.	44
Fig 2.26: An example of the behaviour of each model [20].	47
Fig 3.1: Flow diagram of the proposed study.....	55
Fig 3.2: The sample of the collected dataset with different apparels, different lighting conditions, blurry images due to vibrations, and occluded seat belt pillar loop.	57
Fig 3.3: The illustration of the annotated dataset for the proposed study.	58
Fig 3.4: The mathematical representation of the YOLO format bounding boxes.....	59
Fig 3.5: The illustration of the annotated dataset in terms of coordinates.	60
Fig 3.6: An illustration of how the annotated dataset is saved after the text files have been created for each annotated image.	60
Fig 3.7: Illustration of the dataset splitting process.	61
Fig 3.8: Graphical illustration of the YOLOv7 performance when compared to other YOLO variants.	63
Fig 3.9: The concept of a grid cell used by YOLO models.	64
Fig 3.10: The YOLO history.	65
Fig 3.11: The architecture of the YOLOv7 architecture [35].	66
Fig 3.12: The steps used to deploy the YOLOv7 model into a Jetson Nano development board.	70
Fig 4.1: The precision and the recall graphs.	72
Fig 4.2: Precision-recall curve (PR) and F1-score curve.	72
Fig 4.3: The confusion matrix of the study.	73
Fig 4.4: The losses of the proposed study	74
Fig 4.5: The setup of all the experiments conducted inside the vehicle.....	75
Fig 4.6: A, B, C, and D depict the safety belt detection status when the model was tested in occluded and visible pillar loop conditions.....	77
Fig 4.7: (A) shows the detection status of a safety belt when the hand and hair partially obscured the webbing, and the pillar loop was only 50% visible. B shows the safety belt detection status with the hand blocking the webbing while buckled. C shows safety belt detection with the pillar loop partially visible.....	78
Fig 4.8: The safety belt detection status of A and B was represented when occupants were wearing clothing that was of a similar colour to the webbing.	78

Fig 4.9: This illustrates the detection status of safety belts in both well-lit conditions (A) and low-light conditions (B).	79
Fig 4.10: The visual distortion observed in the scene is a result of the vehicular vibrations induced by traversing a speed hump at a velocity of 40 kilometres per hour.	80
Fig 4.11: Comparative analysis of a safety belt buckle detection performance.....	81

LIST OF TABLES

Table 2.1: The levels of autonomy as described by the SAEs.	14
Table 2.2: The summary of the common activation functions.....	18
Table 2.3: The summary of the deep learning models and their architectures.....	19
Table 2.4: The summary of the three (3) mainly used pooling techniques.....	22
Table 2.5: The summary of the state-of-the-art CNN architectures.....	23
Table 2.6: Pros and Cons of Traditional and deep learning object detection methods.	24
Table 2.7: The summary of the YOLO variants.....	33
Table 2.8: The summary of the literature related to the safety belt object detection.	53
Table 3.1: Data collection size.	58
Table 3.2: Tuned hyperparameters of the study.	67
Table 4.1: Result comparison with related studies	75

LIST OF ABBREVIATIONS AND ACRONYMS

AdaBoost	Adaptive Boosting
ADAS	Advanced Driver Assistance System
ADS	Automated Driving System
ADV	Automated Driving Vehicle
AI	Artificial Intelligence
ANN	Artificial Neural Network
AUC	Area Under the Curve
CNN	Convolutional Neural Network
CO ₂	Carbon dioxide
CV	Computer Vision
DL	Deep Learning
DNN	Deep Neural Network
FC	Fully Connected
FN	False Negative
FP	False Positive
FPN	Forward Propagation Network
GPS	Global Positioning System
GPU	Graphic Processing Unit
HOGS	Histogram of Oriented Gradients
HT	Hough Transform
Hz	Hertz
IEEE	Institute of Electrical and Electronic Engineers
IR	Infrared
LED	Light Emitting Diode
Lidar	Light detection and ranging
LWD	Lane Warning Departure
mAP	Mean Average Precision
ML	Machine Learning
MMR	Mixed Method Research
NN	Neural Network
Radar	Radio Detection and Ranging

R-CNNs	Region-based Convolutional Neural Networks
R-FCNs	Region-based Fully Convolutional Networks
RF	Radio Frequency
ROI	Region Of Interest
RPN	Region Proposal Network
SA	South Africa
SAE	Society of Automotive Engineers
SIFT	Scale Invariant Feature Transform
SPP	Spatial Pyramid Pooling
SSD	Single Shot Detector
SURF	Speed-Up Robust Feature
SVM	Support Vector Machine
TN	True Negatives
TP	True Positive
WHO	World Health Organization
YOLO	You Only Look Once

CHAPTER 1: INTRODUCTION

1.1 BACKGROUND

According to Leland et al. [1], the annual mortality rate resulting from automobile accidents stands at approximately 1.35 million individuals. Furthermore, it has been observed that nearly 50% of individuals involved in these accidents were not wearing their seat belts at the moment of impact [1]. Despite the availability of research evidence indicating that the act of fastening a seat belt significantly decreases the likelihood of fatalities and severe injuries by approximately 45% to 50%, respectively, the aforementioned statistic continues to persist [1]. The act of fastening a seat belt in South Africa (SA) is mandated as a safety measure by the Road Traffic Department. According to Regulation 213 of the National Road Traffic Act of South Africa, 93 of 1996 [2], it is stipulated that no adult is permitted to occupy a seat in a motor vehicle being operated on a public road if the vehicle is equipped with a seat belt unless the individual in question is wearing said seat belt. In contemporary automobiles, seat belt buckle sensors are commonly integrated (refer to Fig 1.1 for a visual representation of seat belt components) as a means to alert drivers about the necessity of fastening their seat belts [3]. However, the occurrence of injuries and fatalities resulting from motor vehicle collisions continues to pose a significant obstacle to the healthcare system in South Africa [4]. Despite the existence of laws and the incorporation of various safety sensors and warning indicators in modern vehicles, there continues to be a failure to implement adequate safety precautions.

There is a paucity of research examining the enforcement of seat belt utilisation. Several research studies have directed their attention towards conventional computer vision (CV) algorithms, which have been found to exhibit instability and inaccuracy when deployed in diverse environmental conditions [5]. This study presents a computer vision-based model as a solution for detecting both unbuckled and buckled seat belts, taking into consideration the need for precise and consistent algorithms to ensure accuracy and stability. The presented model categorises the buckling of a seat belt into two distinct cases.

1. The belt is unbuckled, even when it is fastened behind.
2. The seat belt has been buckled.

The working details of a Seat belt:

As per the World Health Organization (WHO) [6], a car seat belt is a specifically designed strap that serves the purpose of securely restraining an individual within a vehicle, thereby mitigating the risk of forward movement in the event of a collision. The individuals situated within a vehicle in motion possess a velocity equivalent to that of the vehicle itself. If the occupants fail to fasten their seat belts, there is a heightened risk of being forcefully expelled from the vehicle through the windows, resulting in significant bodily harm or fatality in the event of a collision [6]. Nevertheless, when an individual is properly restrained with a seat belt, they will apply a forward force onto the seat belt, which in turn will exert an equal and opposite force on the occupant. Consequently, this force will effectively secure the occupant in their seat [6]. There are certain seat belt components which makes this process possible. The components of a seat belt include a pillar loop, webbing, a latch plate, a buckle, and a retractor, as depicted in Fig 1.1.

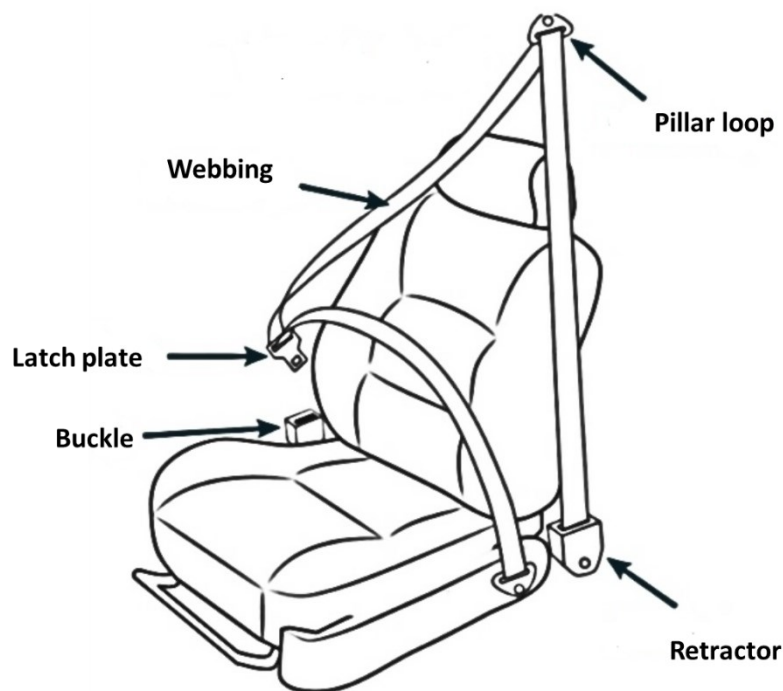


Fig 1.1: Components of a Seat Belt.

Pillar loop: The pillar loop is a component that is securely attached to the region above the driver's or passenger's shoulder. Its purpose is to facilitate the extension or retraction of the seat belt webbing, ensuring that it can be adjusted appropriately to achieve optimal fit and

comfort. Fig 1.2 depicts an illustrative representation of the functionality of a pillar loop during the process of occupants fastening their seat belts within a vehicle.



Fig 1.2: Example of a Seat Belt pillar loop.

Webbing: webbing, commonly known as the "belt" among individuals, functions as a means of restraint for vehicle occupants during collisions.

The latch plate: capable of vertical movement along the restraint's webbing and securely engages with the corresponding buckle upon insertion.

The buckle: functions as the component that receives and secures the latch plate of a restraint. When securing the seat belt, the latch plate swiftly engages with the buckle of the system, where it becomes securely locked until manually disengaged.

The retractor: is accountable for the inward and outward dispersion of webbing during typical operational circumstances. The retractor mechanism is additionally accountable for securing the webbing of a seat belt in a stationary state when subjected to a sudden outward force, as exemplified by a collision.

1.2 COMPUTER VISION AND OBJECT DETECTION

1.2.1 Computer Vision

CV refers to the process of automatically extracting information from images [7]. Images can encompass both static visual representations and dynamic visual representations in the form of videos. The field of computer vision emulates human perception and understanding through the manipulation of data using statistical methods and occasionally geometric analyses in order to address practical challenges in various domains. The human sensory system encompasses various modalities, including tactile perception (sense of touch.), olfaction (sense of smell.), vision, gustation (sense of tasting.), vestibular perception (body's ability to perceive movement in relation to gravity, such as when using an elevator.), proprioception (awareness of body position and self-movement, such as the ability to clap hands with eyes closed.), and auditory perception (ability to receive and understand the difference between sounds.). CV-based systems are widely utilised across various industries, including automation, automotive, medicine, robotics, and others, for the purpose of interpreting real-time data acquired from sensors. It is evident that both CV systems and humans consist of two fundamental elements: sensory apparatus and interpreting apparatus.

Human vision sensor apparatus: The eye functions as a sensory apparatus employed for the purpose of capturing visual representations of diverse surroundings.

Computer vision sensing devices: A type of technology that enables computers to perceive and interpret visual information from the surrounding environment. These devices utilise various techniques and algorithms to analyse images. A camera is a device that is closely associated with the human eye. However, there exist other sensing devices, such as light detection and ranging (Lidar), radio detection and ranging (Radar), CT scans, X-rays, and so on. Occasionally, a confluence of these mechanisms can be employed to achieve a given objective. In the automotive sector, lidars, radars, and cameras are employed to facilitate the implementation of advanced driver assistance systems (ADAS). These systems are integrated into contemporary vehicles, including automated driving systems (ADS), which are utilised in autonomous vehicles.

Human vision-interpreting apparatus: The brain serves as the interpreting apparatus responsible for processing and comprehending visual information obtained from the sensory apparatus.

Computer vision-interpreting apparatus: Employs a series of stages to acquire knowledge and interpret the visual data contained within an image. For CV systems to effectively interpret the information contained within an image, a series of distinct steps must be undertaken. The steps are as follows:

- Step 1: Acquiring the image data from the sensory apparatus.
- Step 2: Performing image processing and conditioning operations.
- Step 3: Extracting relevant features from the processed image.
- Step 4: Utilising a model to interpret the image. The selection of a model is contingent upon the specific application.

Fig 1.3 depicts a pictorial illustration of the functioning of a human vision system and computer vision system as explained earlier. Fig 1.4 demonstrates a summary of the interpretation device for computer vision systems.

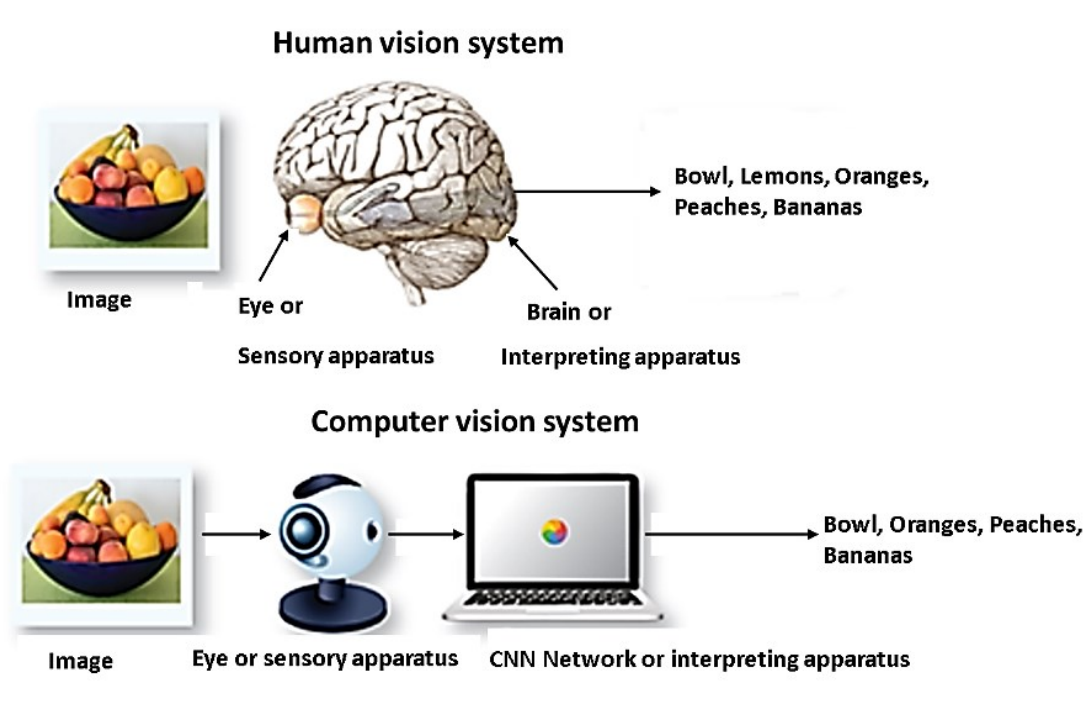


Fig 1.3: Illustration of human vision and the computer vision system [8].

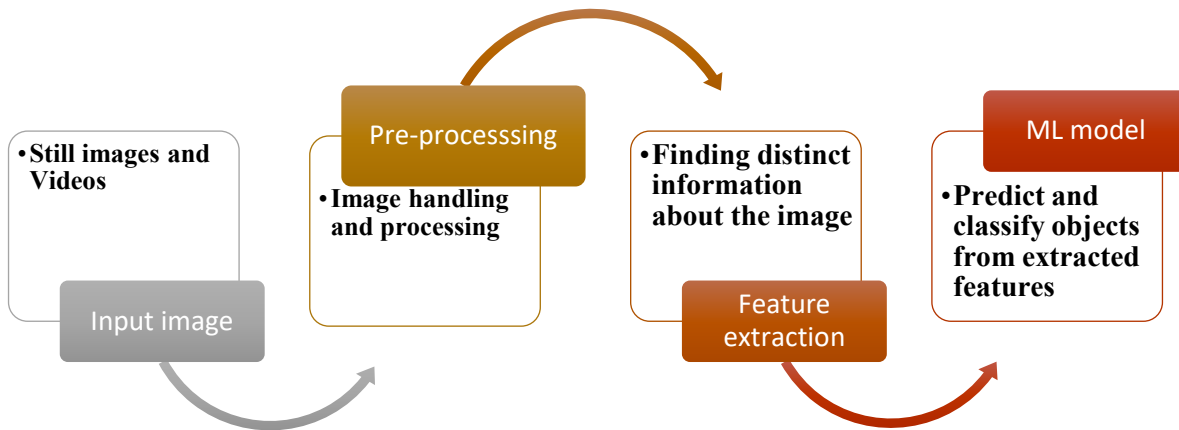


Fig 1.4: Computer vision image interpretation process.

Fig 1.4 depicts an illustrative representation of the feature extraction process in the domain of CV. This domain is bifurcated into two principal methodologies for feature extraction: the traditional approach and the deep learning paradigm [9], the latter of which encompasses convolutional neural network (CNN) models. Within the context of CNNs, the feature extraction process is automated through the mechanism of convolution. In contrast, traditional CV methods entail a manual, artisanal approach for feature extraction, which is subsequently employed as input for classifiers like support vector machines (SVM) and adaptive boosting (AdaBoost).

Among the spectrum of traditional CV techniques, several prominent methods warrant mention, including the histogram of oriented gradients (HOG), Haar cascades, the scale-invariant feature transform (SIFT), and the speeded-up robust features (SURF). These methodologies are characterised by their manual configuration, engendering effective means of discerning salient features from visual data. This preliminary process substantially contributes to the foundation of subsequent tasks involving classification and recognition.

1.2.2 Object Detection

Object detection is a multidisciplinary technology related to CV and image processing that detects and defines objects such as people, vehicles, and animals from images and videos [8]. The basic principle of an object detection model is that it must be able to localise and identify the object. Object detection models can detect multiple objects in an image if they are trained to do so. For instance, in Fig 1.5, only a water bottle is detected, and Fig 1.6 shows multiple

objects, such as the water bottle, can, and milk being detected. Those models were trained to detect the items that they are detecting. Bounding boxes are used to localise and identify objects detected by a model.



Fig 1.5: Example of a water bottle object detection.

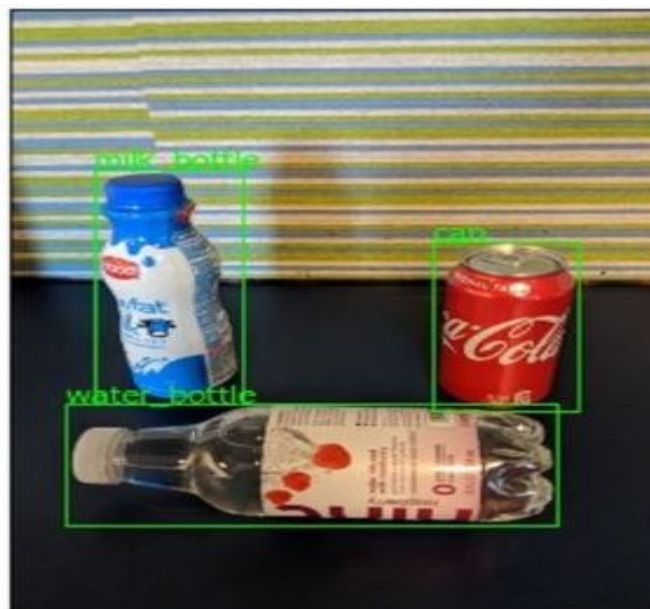


Fig 1.6: Example of detecting multiple objects.

Object detection models are categorised into two types. The first type is the one that recognises the regions of the object to detect it. Examples of the latter mentioned model are region-based

convolutional neural networks (R-CNNs) and faster R-CNNs. The second category uses anchor boxes that are spread all over the image while taking regression to the ground truth bounding box of the object. The loss function is used to determine the box to be used as a bounding box for the object to be detected. Known models of the latter-mentioned type of object detection are Single Shot Multi-Box Detectors (SSD) and YOLO.

1.3 PROBLEM STATEMENT

Most cars are equipped with safety seat belt warning systems that alert the drivers and front passengers whenever they have not buckled their seat belts. The alarming sound of the warning system stops immediately after buckling both the driver and the passenger. However, this method has shortcomings. Some drivers and passengers get irritated by the safety belt's alarming sound and therefore choose to buckle the belt behind their backs; refer to Fig 1.7 for a pictorial illustration of this phenomenon. The current seat belt warning systems assume that the car occupants have buckled if the seat belt's latch plate connects with the buckle, which is sometimes not the case. This method of safety belt detection is susceptible to manipulation by vehicle occupants to silence the warning signals.

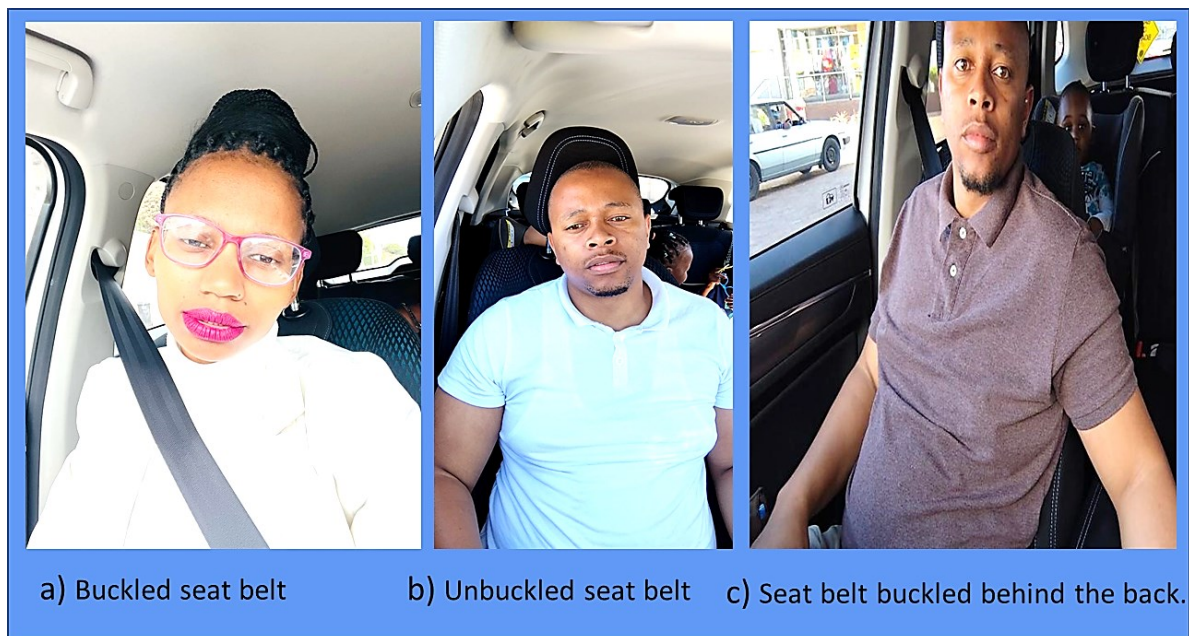


Fig 1.7: Visualisation of the presented problem.

1.4 AIM AND OBJECTIVES

A CV-based model was developed to predict the buckling of a car seat belt for drivers and front passengers, with the intention of deploying it into a Jetson Nano. In the case of an unbuckled driver and a front-seat passenger, the model would notify the user to buckle by emitting a beeping sound from a buzzer. These objectives were achieved by implementing the following:

1. Developing and training an object detection model for a car seat belt using a graphics processing unit (GPU).
2. Deploying the training weights from a GPU platform to a Jetson Nano board for inferencing.
3. Testing the deployed model using the NVIDIA DEEP STREAM object detection applications.

1.5 RESEARCH METHODOLOGY AND DESIGN

The study followed a design and experimental procedure to achieve its objectives. The study employed a mixed method research (MMR) design and experimental procedure to achieve its objectives, arranged as illustrated in Fig 1.8.

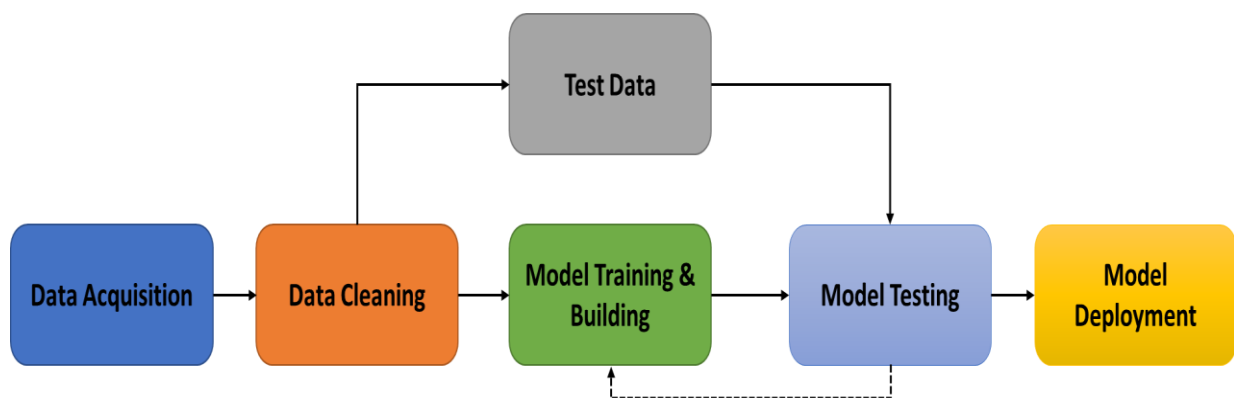


Fig 1.8: Method and design process followed.

Data was acquired from the sensing device (camera) and processed using a cleaning and testing approach, and the model was built and trained from the cleaned data to extract the seat belt features. The built model was tested and deployed to the hardware system. Fig 1.8 shows the chronological order of the research method and design process. A detailed explanation of the methodology is given in Chapter 3.

1.6 DELIMITATION

This research was limited to detecting the seat belt usage of the driver and the front seat passenger. A single camera mounted on the windshield of a vehicle was used to detect the safety belt status. The detection of a seat belt was designed to be possible during the day.

1.7 KNOWLEDGE CONTRIBUTION

This study was focused on developing a CV-based model to detect the buckling of a car seat belt. The model was based on a YOLOv7 algorithm. The model was trained and deployed into a Jetson Nano platform to predict different scenarios of bypassing a seat belt. A beeping sound from a buzzer was used to indicate an unbuckled occupant.

1.8 DISSERTATION STRUCTURE

The structure of the dissertation is as follows:

Chapter 1 introduces the study's context, delineates the research problem, elucidates the overarching aims and objectives, and outlines the general research methodology employed in this research.

In Chapter 2, an extensive literature review is conducted, encompassing a comprehensive analysis of challenges related to seat belt detection and a thorough examination of various object detection approaches explored by previous researchers. This review is substantiated by a detailed exploration of relevant literature sources.

Chapter 3 expounds upon the comprehensive design concept methodology that underpins the proposed study, providing a holistic overview of the study's architectural framework and methodology.

Chapter 4 is dedicated to presenting the study's results and conducting an in-depth discussion of the model employed. The focus was to align the research objectives through rigorous experimental procedures and meticulous analysis of the obtained results, following the research methodology.

In Chapter 5, a comprehensive evaluation of the study is undertaken, involving an analysis of the attainment of the stipulated objectives. The analysis was to determine whether the study

had successfully met its objectives. Subsequently, future recommendations were articulated, and the chapter concluded with a summary conclusion for the study.

CHAPTER 2: LITERATURE REVIEW

2.1 BACKGROUND

Yan [10] posits that there is a projected rise in road accidents due to the exponential growth in the number of individuals who possess private vehicles. According to a study conducted by Kohli and Chadha [11], it is estimated that 3287 individuals perish in accidents on a daily basis. Furthermore, Leland et al. [1] reported that the annual death toll amounts to approximately 1.35 million individuals. Many of these accidents can be attributed to human errors, such as distractions, negligence, fatigue, misjudgement, and speeding [11, 12]. According to Nkuzo, Sibiya, and Markus [13], the primary cause of many accidents resulting in fatalities or serious injuries is attributed to car drivers who demonstrate a lack of adherence to safety regulations. Contemporary automobiles are outfitted with safety cautionary indicators, such as seat belt monitoring systems, which serve the purpose of alerting drivers and passengers to fasten their safety restraints [13]. Roadways are equipped with traffic signs that serve the purpose of notifying drivers regarding various aspects such as speed limits, the prohibition of driving under the influence of alcohol, monitoring driver fatigue, and so forth. The road traffic department has implemented regulations aimed at safeguarding the well-being of both passengers and drivers. Nevertheless, there is a persistent occurrence of inadequate implementation of safety measures, despite the existence of legislation and the incorporation of various safety sensors and warning indicators in contemporary automobiles.

The automotive industry and the academic community have placed significant emphasis on identifying strategies to mitigate traffic accidents and enhance overall road safety [10, 11, 14]. One of the solutions implemented was the integration of ADAS technology into the majority of contemporary automobiles. This technology facilitates the enhancement of safety measures, the enforcement of legal regulations, and the provision of comfort [13]. The ADAS technology incorporates a variety of sensors and transducers, including cameras, radar sensors, lidar sensors, ultrasonics, global positioning sensors (GPSs), and microphones, in order to detect, respond to, and provide information about environmental conditions [12]. The core of ADAS technology is the CV algorithms utilised for various object detection functionalities, including parking assistance, lane departure warning (LWD), pedestrian detection, fatigue detection, traffic sign detection, and others [12]. Fig 2.1 illustrates a visual representation of the various sensors and applications that are seamlessly incorporated into an ADAS vehicle.

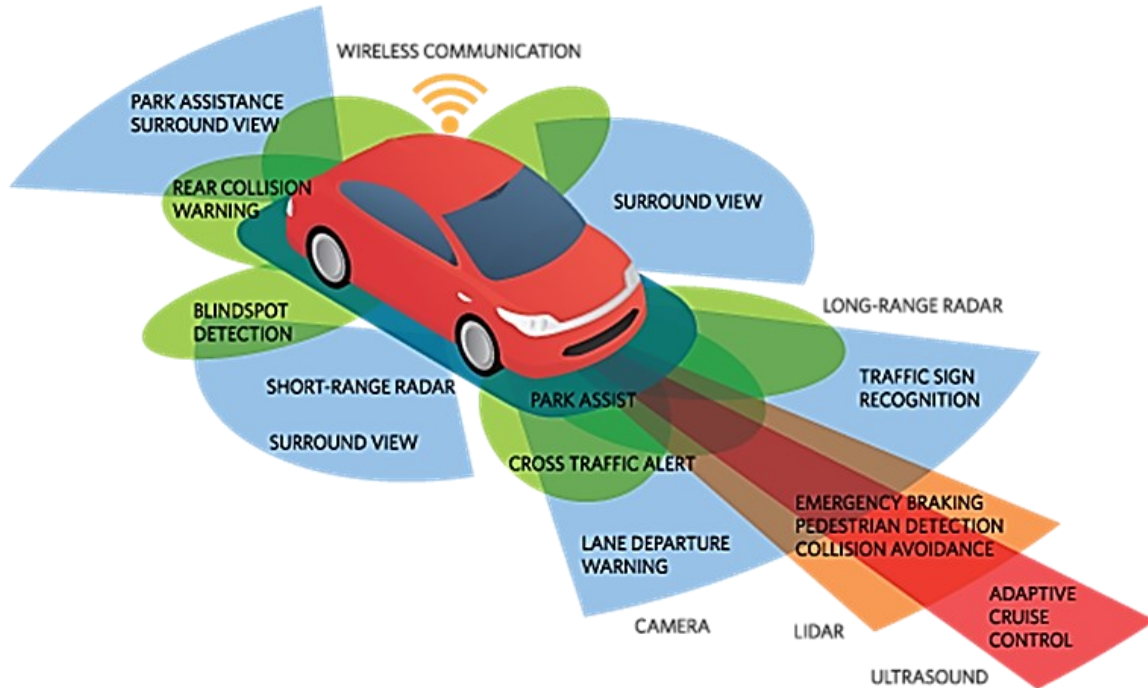


Fig 2.1: A pictorial illustration of an advanced driver assistance car with infused sensors and applications or features [12].

The current research endeavours in the field of ADAS are focused on the development of a fully automated vehicle known as the Automated Driving Vehicle (ADV). One notable distinction between ADVs and ADAS cars lies in the level of driver involvement. ADVs are designed to operate without the need for a human driver, whereas ADAS cars rely on the presence of a driver to oversee and participate in certain automated driving functions [12]. In 2014, a roadmap towards achieving complete automation of self-driving vehicles was established by the Society of Automotive Engineers (SAEs) [15]. Table 2.1 presents an overview of the levels of vehicle autonomy as defined by the Society of Automotive Engineers (SAEs) in 2014.

Table 2.1: The levels of autonomy as described by the SAEs.

Level of automation	Automation description	The human driver monitors the road
Level 0	No Automation: all the driving tasks are performed by a human being.	
Level 1	Driver Assistance: there are few automated tasks, but most of the tasks are still performed by the driver.	
Level 2	Partial Automation: many tasks are performed by the car, but the driver is still heavily involved when driving.	The automated driving system monitors the road
Level 3	Conditional Automation: the car can perform the most tasks, but the driver is still required to monitor the driving and perform a few tasks.	
Level 4	High Automation: the car can perform all the driving tasks but under specific conditions such as areas with low-speed driving.	
Level 5	Full automation: the car performs all the driving tasks with no human interference.	

Based on the information presented in Table 2.1, it is apparent that the current state of technology in the automotive industry ranges from level 0 to level 3. This is primarily due to the prevailing requirement for drivers to actively engage in the driving task and maintain continuous vigilance throughout the process. Nevertheless, a limited number of level 4 autonomous vehicles have been created. According to Yeong et al. [15], Waymo (a subsidiary of Google), Cruise, Uber, and Tesla are among the prominent brands that assert to have successfully engineered level 4 autonomous vehicles.

Denton [12] posits that the implementation of an ADV yields several benefits, including a significant reduction in fatalities by 90%, a decrease in carbon dioxide (CO₂) emissions, an enhancement in vehicle energy efficiency, a decrease in transportation costs, an improvement in transportation accessibility, a reduction in land utilisation for roads, and a diminished requirement for parking spaces. According to Agarwal, Chiang, and Sharma [16], it is also argued that ADVs have the potential to shape the future of our society and our way of life. However, there is a shared consensus among both authors that the realisation of such

technology will require a significant amount of time. The accuracy requirement and the cost of constructing and sustaining infrastructure to accommodate such technology are identified as significant contributing factors [12, 16].

2.2 OBJECT DETECTION MODELS

This literature review aims to provide a comprehensive analysis of the object detection literature in the field of computer vision, employing a descriptive and analytical approach. According to Zou et al. [17], object detection has gone through two (2) historical periods in the past two (2) decades: (a) there was a traditional object detection period before the year 2014 and (b) a deep learning-based object detection period that started after the year 2014. Fig 2.2 reveals the work process for both traditional and deep learning-based approaches.

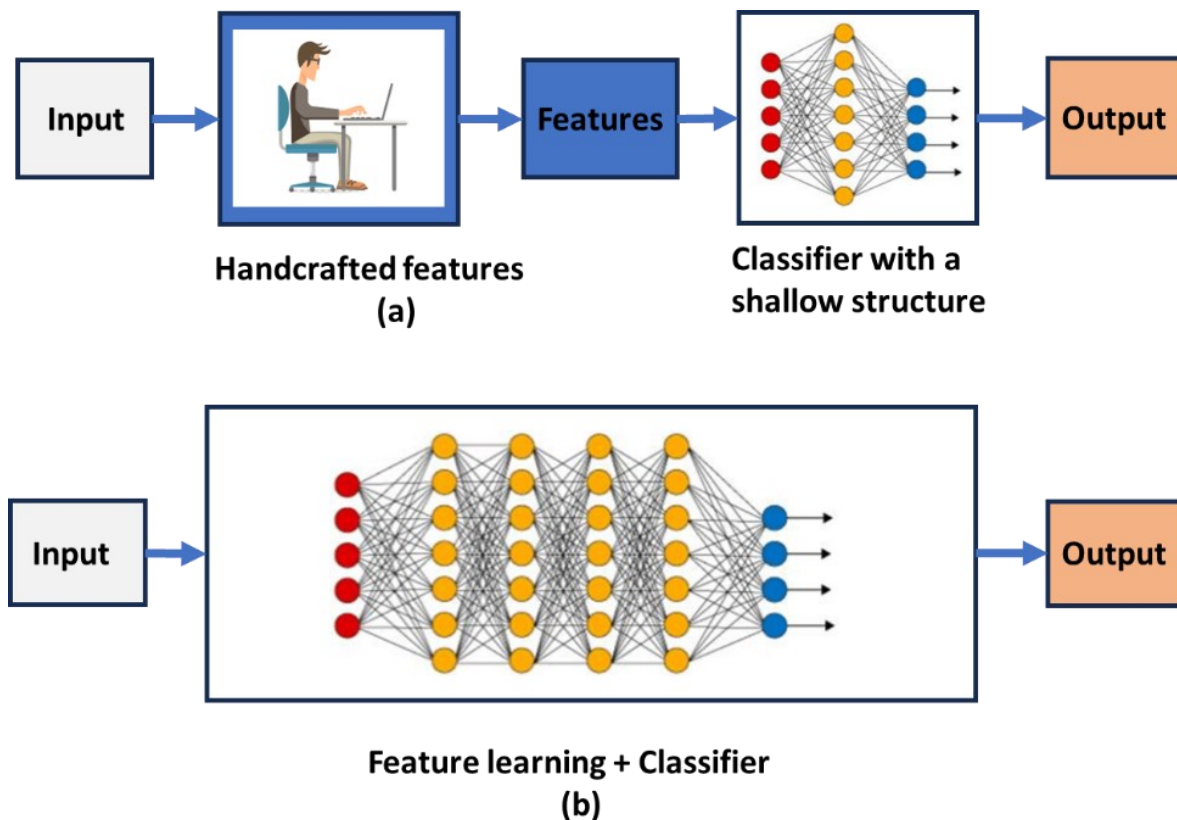


Fig 2.2: An example of a traditional object detection work process (a) and a deep learning-based object detection work process (b).

The emergence of deep neural networks (DNNs) has exerted a significant impact on the domain of CV in recent times, leading to a substantial advancement in the capacity to accurately identify and classify objects. The recent surge in advancements can be attributed to enhanced

computational capabilities and the growing availability of data for training neural networks [18]. However, it is worth noting that conventional object detection algorithms are occasionally employed in combination with deep learning-based object detection algorithms.

The advent of DNNs has wielded substantial impact within the realm of CV in contemporary times, propelling a significant advancement in the capacity to accurately discern objects. An increase in computing power and an increase in the amount of data available for training neural networks have both contributed to this surge in progress [18]. Nonetheless, traditional object detection algorithms are sometimes used in conjunction with DL-based object detection algorithms.

2.2.1 Traditional object detection

According to Zou et al. [17], traditional object detection models include Viola-Jones detectors, HOGs, and deformable part-based models (DPMs). However, Lee [19] submitted that traditional object detection in computer vision models includes models such as edge detection, Hough transforms (HT), scale-invariant feature transforms (SIFT), binary robust independent elementary features (BRIEF), and speeded-up robust features (SURF). The author also submitted that traditional object detection models have been around for more than two decades. Zou et al. [17] submitted that the models have been evolving over the years. The authors further submitted that traditional object detection methods had challenges in speed and accuracy and would struggle in variable environments, whereas deep learning-based object detection methods are faster and more accurate than the old models.

2.2.2 Deep learning-based object detection

DL is a specialised area within the field of machine learning that is dedicated to the advancement of algorithms capable of comprehending and acquiring both high-level and low-level abstractions from data. This is a capability that conventional machine learning algorithms lack [18, 20]. DL models often consist of layers of nonlinear data-processing units called neurons. These models typically incorporate multiple layers that are responsible for processing different levels of data abstraction. DNNs are characterised by the presence of numerous hidden layers. These layers are termed "hidden" due to the fact that the inputs and outputs of the neurons within them may not be directly observable, except for the understanding that they serve as the output of the preceding layer. Fig 2.3 presents a graphical illustration of the DNN layers.

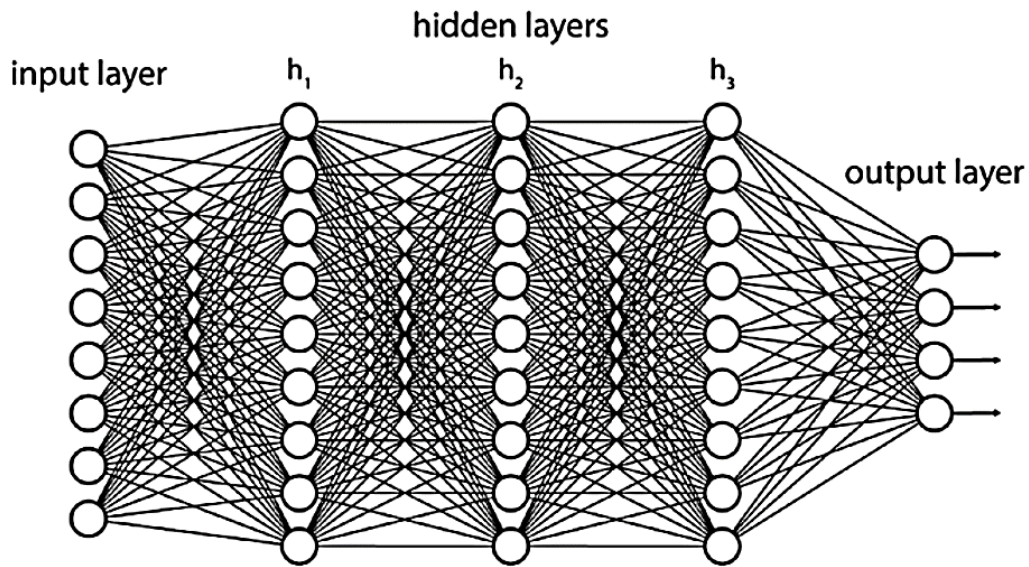


Fig 2.3: Deep neural network (DNN).

In a neural network (NN), the transmission of information occurs between layers through a network of interconnected channels. These channels, referred to as weighted channels, are assigned values that reflect their significance. Each neuron in the network possesses a unique bias value. This bias, in combination with the weighted sum of inputs received by the neuron, is then subjected to an activation function. The outcome of this function determines whether the neuron becomes activated. Activated neurons subsequently transmit information to the subsequent layers, continuing until the penultimate layer. The output layer, which represents the final layer of an artificial neural network, generates program outputs. For a mathematical representation and expression of the functioning of a single neuron (perceptron), refer to Fig 2.4 and equation (2.1).

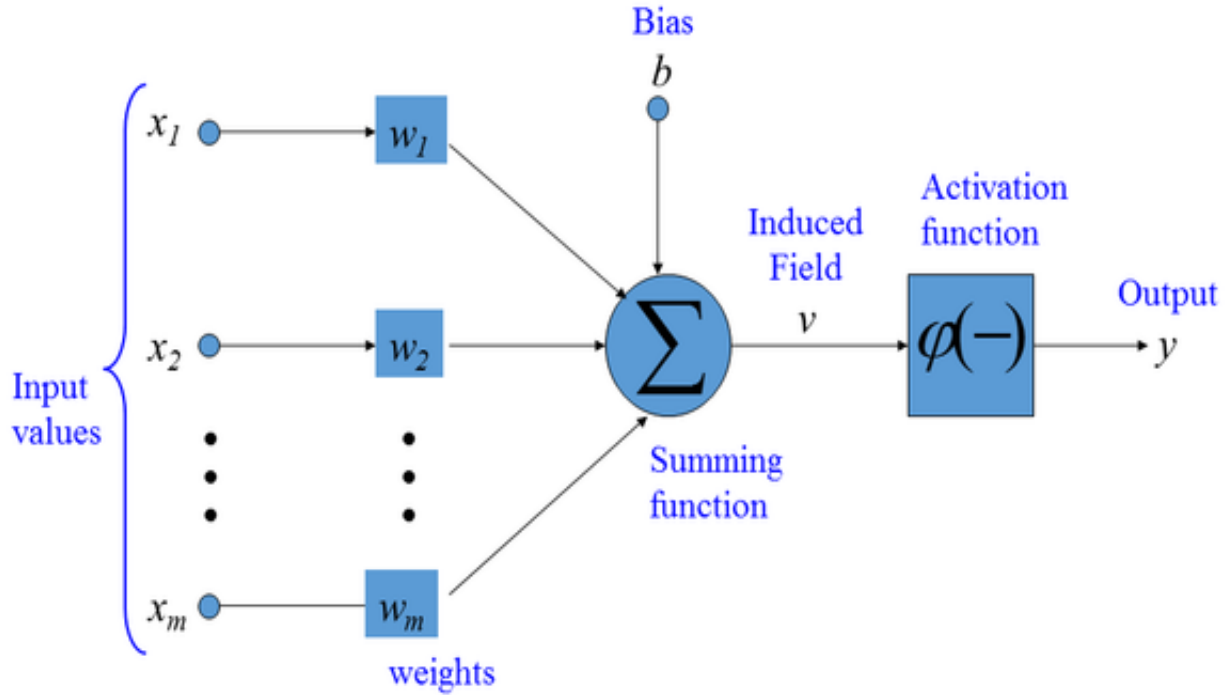


Fig 2.4: Neural Network (NN), known as the Perceptron.

$$\text{Output} = y = f(w_1 \times x_1 + w_2 \times x_2 + \dots w_m \times x_m + b) \quad (2.1)$$

The neurons within a neural network are occasionally denoted as nodes. Learning in a neural network is facilitated by the utilisation of an activation function, which enables the network to exhibit non-linear behaviour, thereby enabling it to effectively process intricate computational tasks. Table 2.2 illustrates the prevalent forms of activation functions along with their corresponding mathematical representations.

Table 2.2: The summary of the common activation functions

Activation function	Mathematical expression	
Sigmoid	$\delta(x) = \frac{1}{(1 + e^{-x})}$	(2.2)
Tanh	$tahn(x) = 2\delta(2x) - 1$	(2.3)
Softplus	$f(x) = \log(1 + e^x)$	(2.4)
Rectified Linear Unit (ReLU)	$f(x) = \max(0, x)$	(2.5)
Leaky ReLU	$f(x) = \max(0.1x, x)$	(2.6)
Mish	$f(x)xtanh(softplus(x))$	(2.7)
Sigmoid-Weighted Linear Unit (SiLU)	$f(x) = \frac{x}{(1 + e^{-x})}$	(2.8)

Table 2.3 illustrates the different commonly used models for deep learning-based techniques and their corresponding architectures.

Table 2.3: The summary of the deep learning models and their architectures.

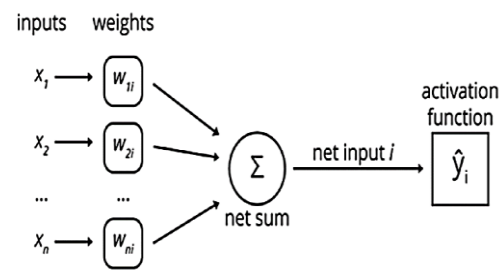
Model name	Architecture
Single Layer Perceptron (SLP)	

Fig 2.5: SLP architecture.

Multiple Layer Perceptron (MLP)

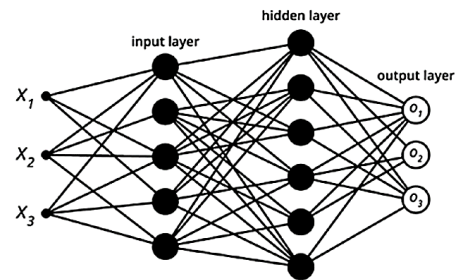


Fig 2.6: MLP architecture.

CNNs

See Fig 2.10.

Recurrent Neural Networks (RNN)

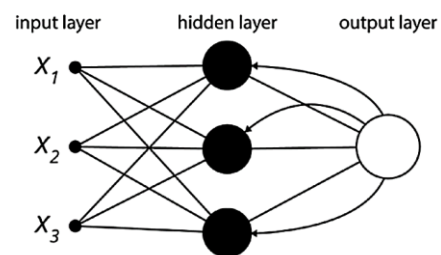


Fig 2.7: RNN architecture.

Restricted Boltzmann Machines (RBMs)

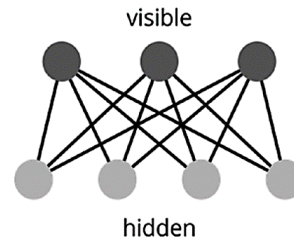


Fig 2.8: RBM architecture.

Deep Belief Networks (DBNs)

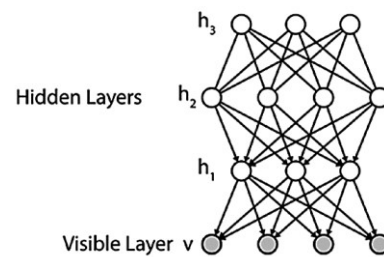


Fig 2.9: DBN architecture.

CNNs:

CNNs have emerged as a prominent DL methodology, characterised by the training of multiple layers in a robust manner [18]. Guo et al. [18] have demonstrated the high efficacy and widespread utilisation of CNNs in diverse computer vision applications. In general, a CNN comprises three primary layers: **convolutional layers**, **pooling layers**, and **fully connected layers**. Each layer serves a distinct purpose [18]. Fig 2.10 illustrates a comprehensive CNN structure, specifically designed for image classification, showcasing the architecture of each layer. The training process of a network consists of two distinct stages: a forward stage and a backward stage. The primary goal of the forward stage is to present the input image using the present parameters (weights and bias) in every layer. Next, the prediction output is utilised to calculate the loss cost by comparing it with the ground truth labels. Next, utilising the loss cost, the backward stage calculates the gradients of each parameter using the chain rule. The parameters are updated using the gradients and are subsequently prepared for the subsequent forward computation. Once a satisfactory number of iterations of the forward and backward stages have been completed, the process of network learning can be terminated.

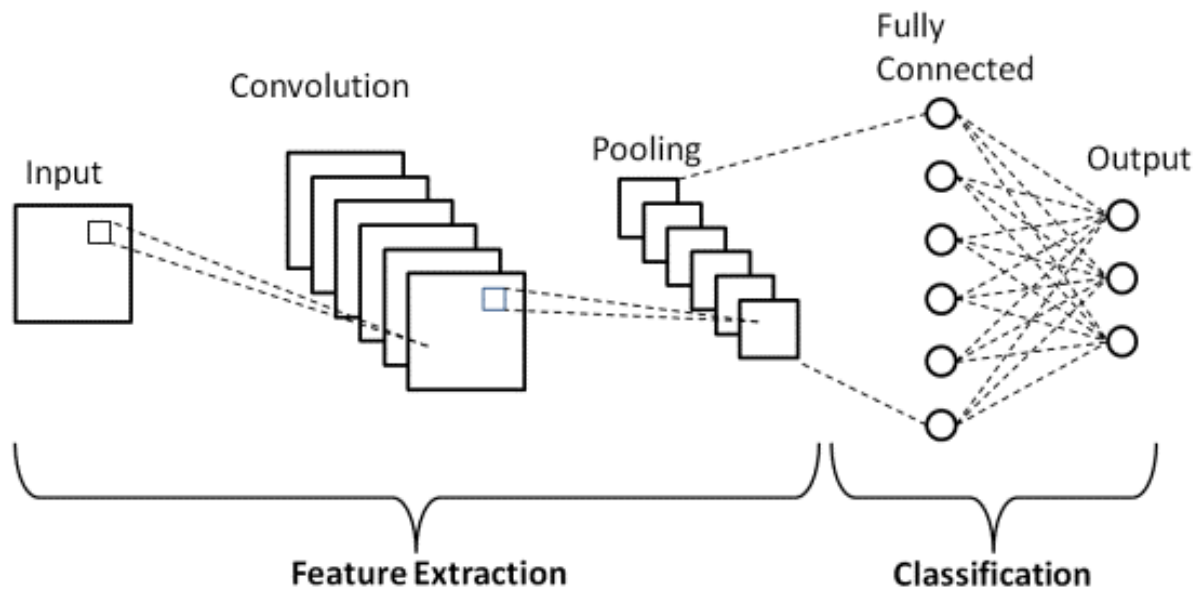


Fig 2.10: CNN general architecture.

Fig 2.10 reveals the general architecture of a CNN model. It has a feature extraction and classification stage. The feature extraction stage consists of the input, convolution, and pooling stages, and the classification stage consists of the fully connected and output stages.

Input: the CNN model obtains data or information, which may manifest as images, from the sensing device, which is typically a camera. The aforementioned data is subsequently inputted into the convolution layer for further processing.

Convolutional layer: The convolutional layer serves as the initial layer responsible for extracting diverse features from the input images. Within this stratum, the process of convolution, a mathematical operation, is executed between the input image and a filter possessing a specific size denoted as $M \times M$. By sliding the filter across the image, the dot product between the filter and the corresponding regions of the input image—whose dimensions are the filter's ($M \times M$)—is calculated. The resulting output is referred to as the feature map, which provides us with valuable information regarding the image, including the identification of corners and edges. Subsequently, the aforementioned feature map is inputted into subsequent layers in order to acquire additional features from the input image. The convolutional layer in a CNN transfers the output to the subsequent layer after performing the

convolution operation on the input. Convolutional layers in CNNs confer significant advantages by preserving the spatial correlation among pixels.

Pooling layers: The main objective of pooling layers is to reduce the dimensions of the convolved feature map in order to minimise computational expenses. This process involves reducing the interconnections among layers and conducting independent operations on individual feature maps. Various types of pooling operations exist, contingent upon the employed methodology. The provided statement encapsulates the characteristics produced by a convolutional layer.

In the process of max pooling, the feature map is subjected to the selection of the largest element. The process of average pooling involves computing the mean value of the elements within a predetermined region of an image. Sum pooling is a computational technique used to determine the cumulative sum of the elements within a predefined section. The three mainly used pooling techniques are shown in Table 2.4.

Table 2.4: The summary of the three (3) mainly used pooling techniques.

Pooling	Description
Stochastic pooling	Equivalent to standard max pooling, with the addition of multiple replicas of the input image, each with minor local distortions. The incorporation of stochastic elements mitigates the problem of overfitting.
Def-pooling	Occasionally employed in the context of addressing deformation, a significant challenge in the field of computer vision.
Spatial pyramid pooling (SPP)	The SPP method can extract fixed-length representations from images or regions of arbitrary size and shape. This provides a flexible solution for effectively managing scale, size, and aspect ratio variations. In addition, the SPP technique can be seamlessly integrated into any CNN

architecture, thereby improving the network's overall performance.

Fully connected and output layers: The fully connected (FC) layer is comprised of neurons, weights, and biases and serves the purpose of connecting neurons from two distinct layers. Typically, these layers are positioned prior to the output layer and constitute the final layers within a CNN architecture. The input image from the preceding layers is flattened and subsequently passed to the FC layer. The flattened vector subsequently proceeds through several FC layers, which typically involve mathematical operations. At this juncture, the process of classification commences. The rationale behind connecting two layers is that the performance of a neural network can be enhanced by employing two fully connected layers instead of a single connected layer. The utilisation of layers in CNNs serves to decrease the level of human supervision required. Table 2.5 provides a summary of several contemporary CNN architectures that are considered to be at the forefront of the field.

Table 2.5: The summary of the state-of-the-art CNN architectures.

Author(s)	Types of CNN architecture
LeCun et al. [21]	LeNet-5
Krizhevsky et al. [22]	AlexNet
Fergus and Zeilar [23]	ZFNet
Szegedy et al. [24]	GoogleNet/Inception
Simonyan and Zisserman. [25]	VGGNet
He et al. [26]	ResNet

2.2.3 Pros and Cons of deep learning and traditional object detection

The advantages and disadvantages of traditional and deep learning-based techniques are listed in Table 2.6.

Table 2.6: Pros and Cons of Traditional and deep learning object detection methods.

Traditional object detection		Deep learning-based object detection	
Pros	Cons	Pros	Cons
	Computational expensive.	Computational cheap	Requires a lot of training data.
They are well-researched topics.	Can be inaccurate	They are accurate	Training time can be high.
	Heavily affected by environmental situations	Less affected by environmental situations.	Requires the GPU to lessen the training time.
They do not require annotated images	Struggle to learn complex features	Can learn complex features.	Laborious manual annotation process.
	They are slow	They are faster	

2.3 TYPES OF OBJECT DETECTION

Object detection is divided into two (2) categories: two-stage and single-stage object detection as shown in Fig 2.11. The fundamental discrepancy between these methodologies lies in the integration of a region proposal network (RPN) or object proposal within the two-stage detector. Single-stage detectors, exemplified by YOLO and SSD, have no RPN. The block diagrams of the object detection types are as shown in Fig 2.12.

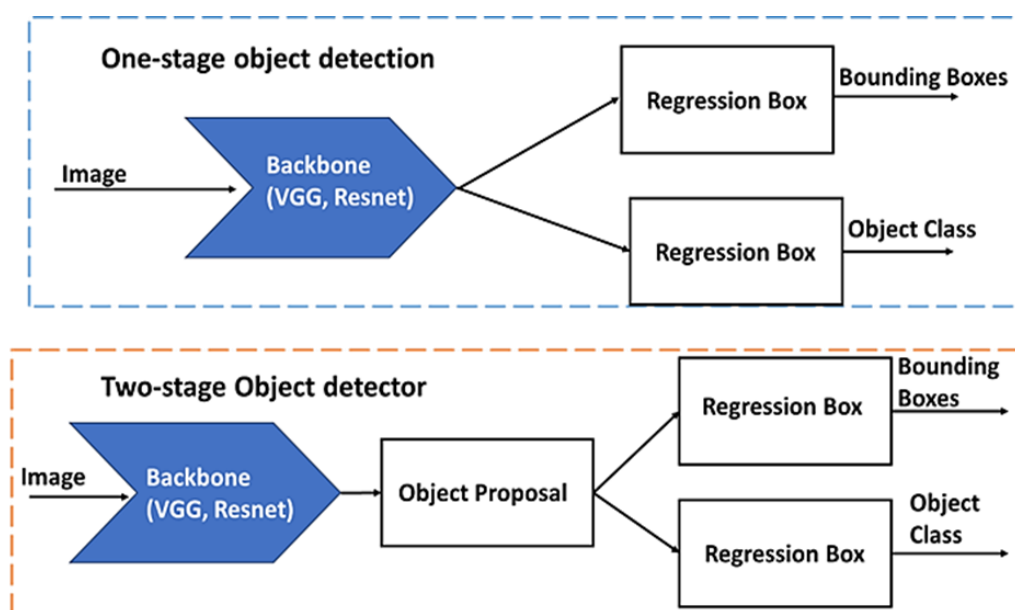


Fig 2.11: One-stage and two-stage object detection block diagrams.

Two-stage object detection models are R-CNN, Fast R-CNN, Faster R-CNN, R-FCN, and Mask R-CNN. Single-stage object detection models are YOLO and SSD, as shown in Fig 2.12.

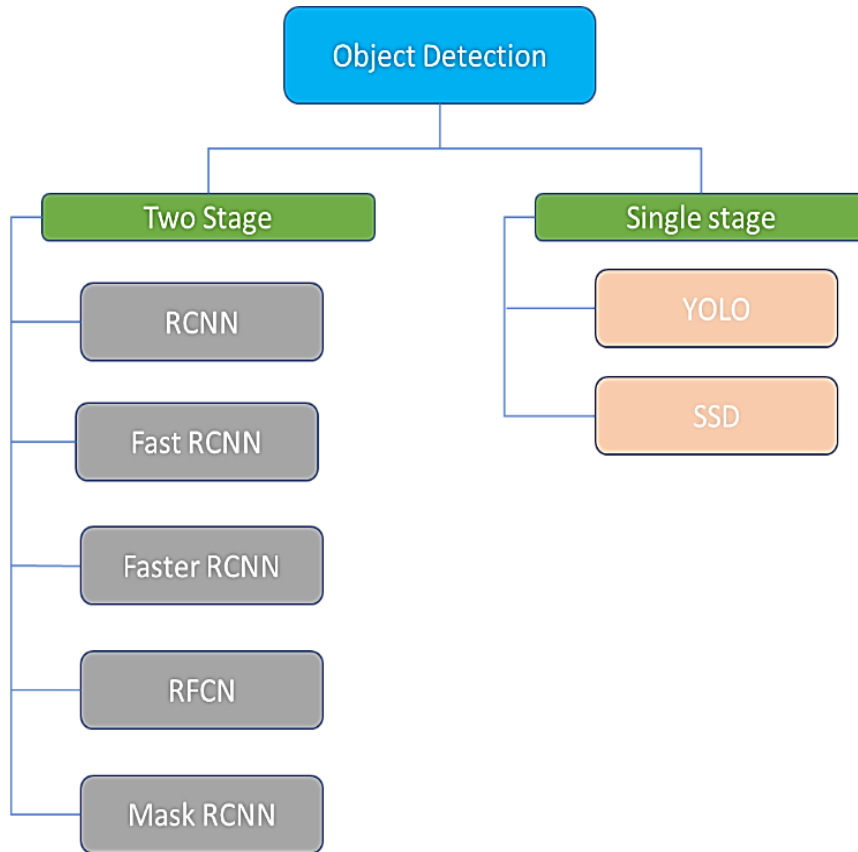


Fig 2.12: Object detection types.

2.3.1 Two-stage object detection

The division between the two stages of a two-stage detector can be delineated by the inclusion of a region of interest (ROI) pooling layer. The initial phase of the system is referred to as the RPN, which is responsible for generating predictions of potential bounding boxes. During the second stage, the process of extracting features is carried out through the utilisation of ROI pooling operations for every candidate box. This extraction is performed to facilitate the tasks of classification and bounding box regression. Fig 2.13 presents the illustration of a two-stage object detection process.

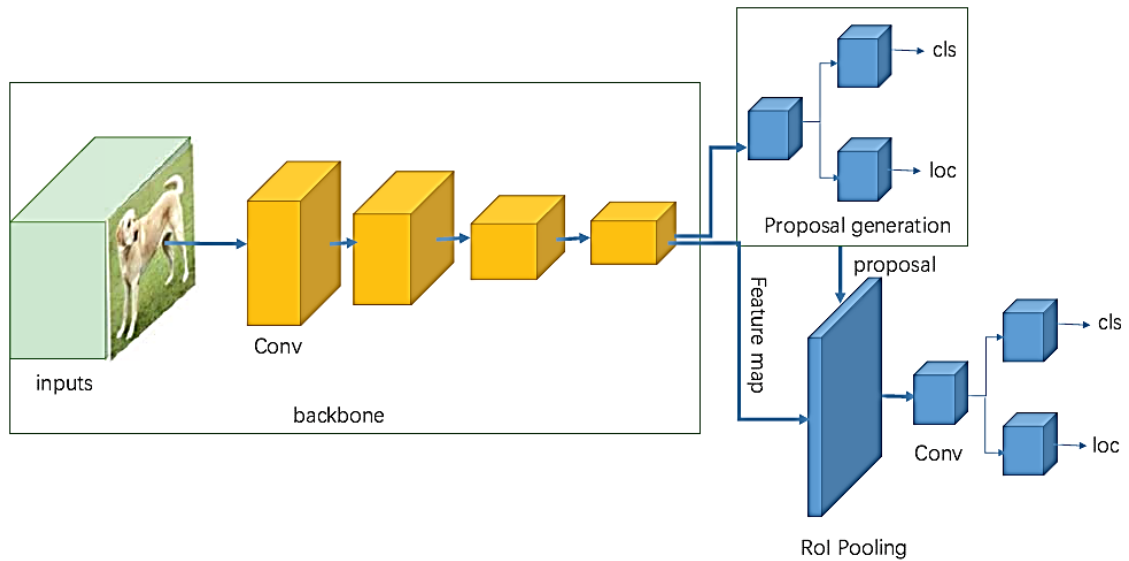


Fig 2.13: The two-stage object detection process.

R-CNN:

It is an object detection technique that uses a region-based CNN approach to implement selective search with NNs. It is an early object detection algorithm that produces good results in a large dataset for object detection and localisation. In the formative work of [27], to obtain 2,000 region proposals per image, it was recommended to use an external object proposal, also known as a region proposal generator. From these region recommendations, CNN is used for feature retrieval and SVM for identification purposes. Fig 2.14 shows an example of an R-CNN model.

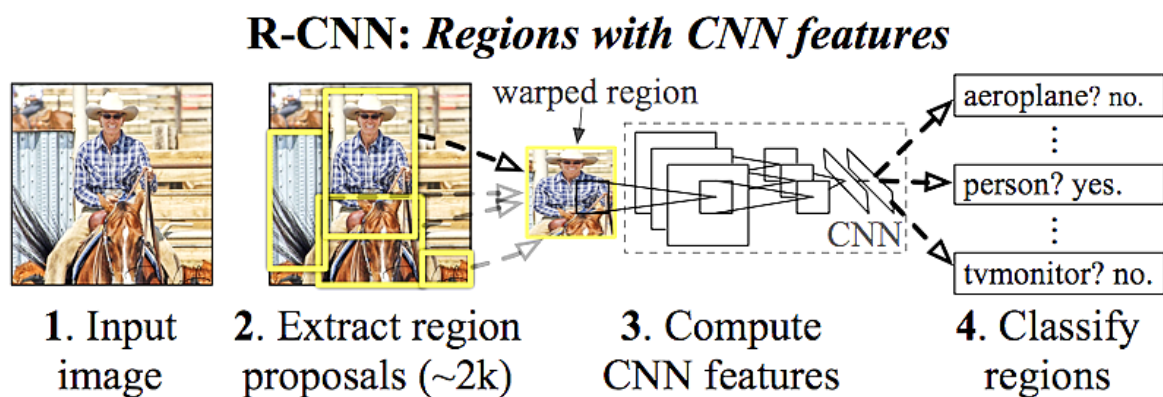


Fig 2.14: R-CNN Model.

R-CNN model consists of the following components:

1. The input to the model consists of an image.
2. **The model proceeds to extract region proposals** by employing the selective search method. The iterative process of the selective search method involves grouping regions based on shared features with the intention of identifying regions that potentially contain objects. Subsequently, a subset of approximately 2,000 regional proposals is chosen for subsequent analysis and processing.
3. **The process of extracting features from the region proposals** that have been extracted. Subsequently, a CNN model that has been pre-trained, such as AlexNet or ResNet is employed to extract the features from approximately 2,000 region proposals. In order to make each region proposal work with the R-CNN's input requirements, the image sizes within each region must be changed to fit CNN's needs. This is done by using dilation and wrapping techniques.
4. **The classification module** employs an SVM to categorise the identified objects using the extracted features from the previous stage. The SVM classifier exhibits a notable propensity for achieving a high recall rate, yet it tends to exhibit relatively lower precision. Hence, it is not suitable for independent use as an object detector; however, it can be effectively employed as the initial component in a detection pipeline.

The localisation module is responsible for generating predictions of the bounding box coordinates that will be used to outline the localised object. This process yields four parameters, namely (x, y, w, and h). The parameters (x, y) represent the coordinates of the origin, while (w, h) correspond to the width and height of the box, respectively. The R-CNN methodology offers the benefit of being capable of utilising any CNN for the purpose of feature extraction. CNNs can be made better by adding more complex convolutional architectures, which could lead to an increase in the mean average precision.

Fast R-CNN:

In 2015, Girshick [27] proposed an enhanced architecture called Fast R-CNN with the aim of enhancing the existing R-CNN model. The Fast R-CNN approach addressed the issues related to speed and memory by implementing modifications to the architecture and eliminating the computationally expensive selective search algorithm. The fundamental R-CNN model has undergone significant advancements at a rapid pace. This section discusses the modifications implemented in R-CNN to enhance the fast R-CNN model. In addition to employing CNN,

Fast CNN exhibits a slight divergence in the functioning of its region proposal mechanism. In the Fast-RCNN framework, the initial step involves applying the CNN feature extractor to the complete input image, followed by the subsequent proposal of regions. In contrast to utilising 2,000 convolutional neural networks (ConvNets) across 2,000 overlapping regions, a singular ConvNet is employed to process the entirety of the image. The ConvNet is accountable for both the feature extraction and categorisation aspects of the process.

Components of Fast R-CNN:

The fast R-CNN architecture is depicted in Fig 2.15. A series of region proposals are sent to the network along with an input image. The red box on the input image is a visual example of a region proposal for this image. Instead of applying selective search to the generation of numerous region proposals for each image, Fast R-CNN accepts them as inputs and does not generate them.

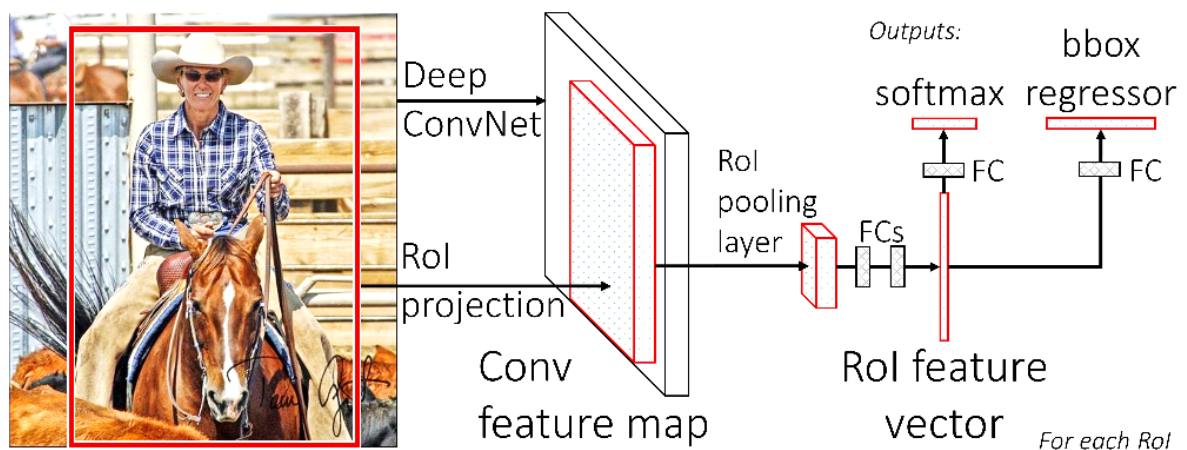


Fig 2.15: Fast R-CNN model.

The ConvNet utilises a combination of convolution and max-pooling layers to perform feature extraction on the entire image. A feature map refers to a compilation of extracted features. The ROI pooling layer is a component commonly used in CNN for object detection and instance segmentation tasks. It is designed to extract fixed information. The provided depiction serves as a simplified representation of a SPP layer, featuring a sole pyramidal layer. The process involves extracting feature vectors from a convolutional feature map within a predetermined window size, which are then passed as input to FC layers. A max-pooling layer is employed to

produce a reduced feature map with a constant spatial extent $H \times W$ (height $\times$ width) for the features of a valid ROI.

The output layers are a crucial component in neural networks. They are responsible for producing the final predictions or outputs of the system, which consists of two output layers that operate in parallel. The first output layer employs a fully connected layer to classify the image, which is then followed by a softmax classifier. On the other hand, the second output layer utilises multi-task training to simultaneously train the classifier and perform irregular regression. The outputs of the classification process are the size of the bounding box and the location coordinates of the identified object. The conventional approach of utilising the bag of visual words methodology resulted in the development of SPP. A spatial pyramid is constructed above the ROI. The initial tier of the pyramid encompasses the ROI itself. At the second level, the region is partitioned into four cells using a two-by-two grid. The geographical area is partitioned into sixteen individual units arranged in a four-by-four grid configuration at the tertiary level. Every individual cell undergoes the process of average pooling. Consequently, when the final convolutional layer comprises 256 maps, the pooling operation within each cell yields a vector with a length of 256. The feature vectors of all the cells are combined and provided as input to the fully convolutional layer. In conclusion, a fixed-length feature map is generated for input images of diverse resolutions.

Faster R-CNN

Shaoqing Ren et al. [28] introduced their proposed model as an expansion of Fast R-CNN, which is the third iteration in the R-CNN family. A ConvNet generates a convolutional feature map from the input image, similar to the approach employed by Fast R-CNN. The region proposal network is utilised to identify ROI within the feature map of the region proposal. Subsequently, the image in question is categorised within the designated region, and a prediction is made for the bounding box.

The main components of Faster R-CNN are:

1. **RPN:** When a fully convolutional network uses convolution, it is called an RPN. The RPN is designed to undergo comprehensive training, resulting in the generation of field resolutions of superior quality. These resolutions are subsequently employed by the R-CNN for the purpose of detection. The output of the RPN is the predicted object bounding box and the object score for each pixel.

2. **Detector:** A pre-trained Faster R-CNN model is used as an object detector. Fig 2.16 illustrates a faster R-CNN model.

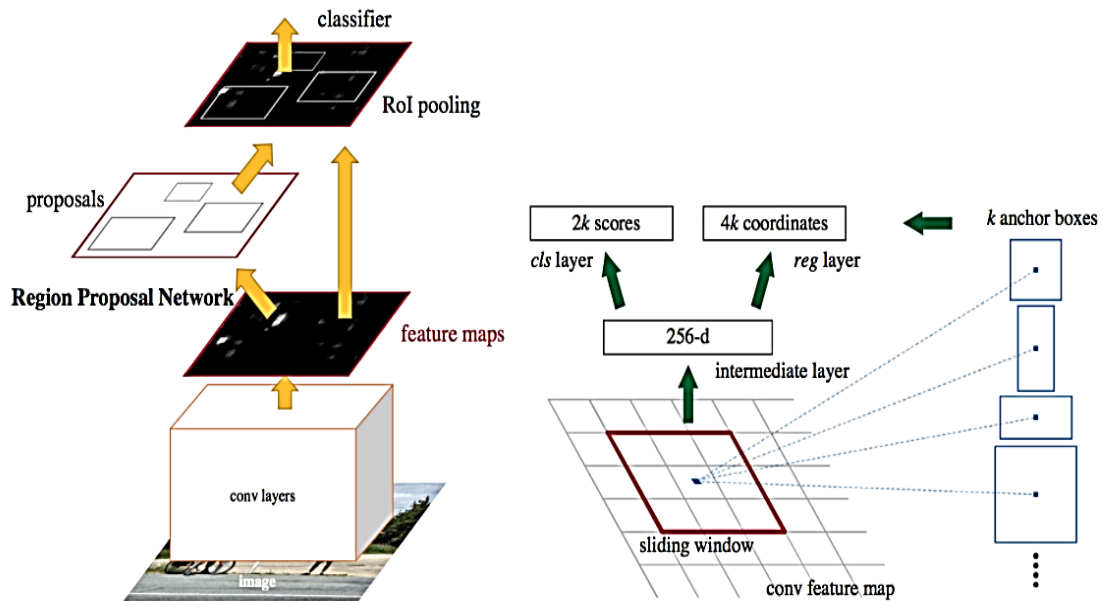


Fig 2.16: Faster R-CNN model.

As depicted in Fig 2.16, the faster R-CNN algorithm utilises a ConvNet to process the entire image and produce feature maps. These feature maps are generated by employing a pre-trained CNN model. The feature maps obtained are subsequently transmitted in conjunction with the RPN and ROI pooling layers. Subsequently, the generated output is transmitted to the two fully connected layers. The object classifier classifies the object based on its anticipated class, while the bounding box regression model estimates the coordinates for localising the object.

R-FCN:

Region-based Fully Convolutional Networks (R-FCN) possess an architecture that closely resembles that of a R-CNN but solely utilises CNNs. This methodology utilises a region-based detector that is entirely convolutional and exposes its computation across the entire image. Consequently, it eliminates the need to compute subnetworks in each region, thereby avoiding repetitive computations for every image. Scholars have proposed the utilisation of score maps with a positive-sensitive nature as a potential solution to address the issues of translation invariance in image classification and translation variance in object detection. Therefore, the R-FCN methodology adopts a sequential two-step procedure involving region proposal and

classification. In this process, the candidate regions are acquired using a fully convolutional RPN.

Mask R-CNN:

He et al. [29] introduced a novel object detection methodology known as Mask R-CNN. The authors assert that the proposed framework is characterised by its conceptual simplicity, adaptability, and broad applicability in the context of object instance segmentation. Their proposed methodology was aimed at identifying and localising objects within an image while concurrently producing a segmentation mask for each individual instance. This approach is an extension of a Faster R-CNN, wherein an additional branch is incorporated to concurrently predict an object mask alongside the existing branch responsible for bounding box recognition. Fig 2.17 illustrates the Mask R-CNN framework.

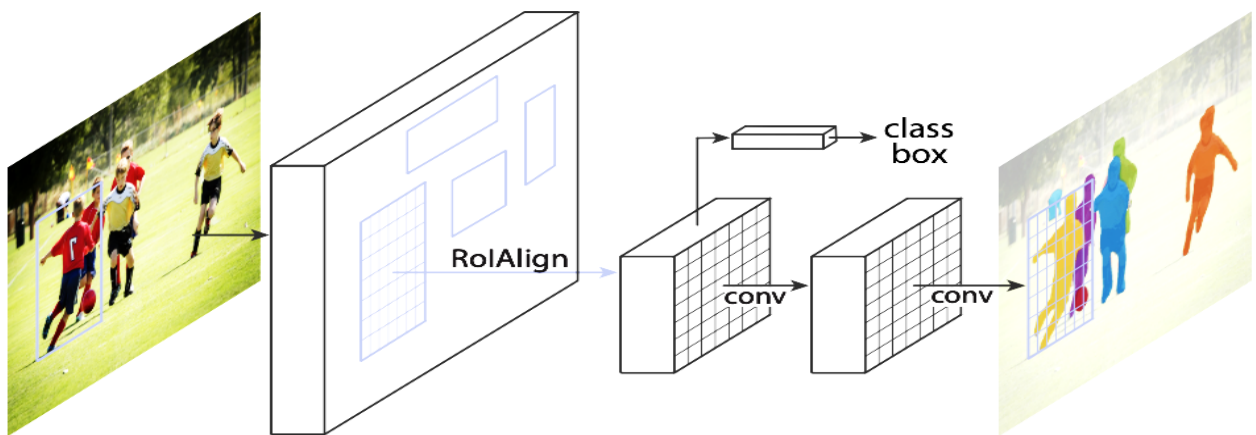


Fig 2.17: Mask R-CNN framework.

2.3.2 Single-stage Object Detectors

A one-stage detector is a type of object detection model that directly predicts bounding boxes in a single step without the need for region proposals. The utilisation of a grid box and anchors are employed to localise the specific region of detection within the image and restrict the shape of the object. The single-stage detection process is depicted in Fig 2.18.

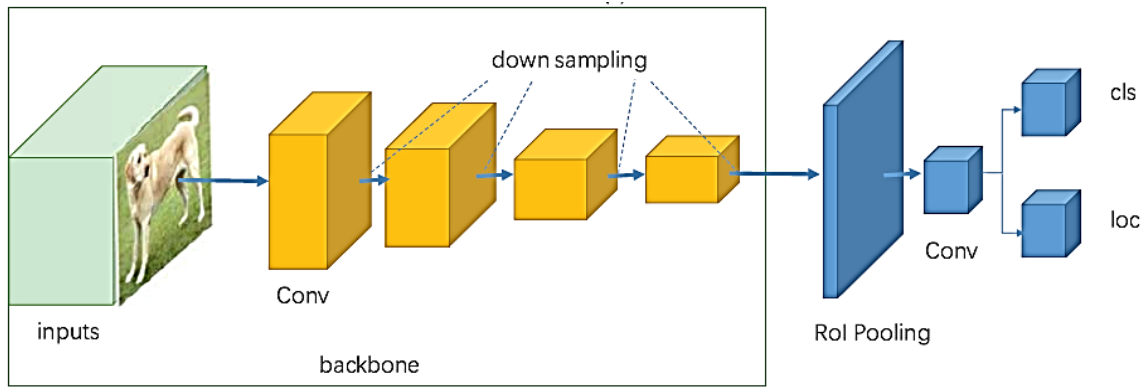


Fig 2.18: Single-stage object detection architecture.

You Only Look Once (YOLO):

YOLO is an open-source object detection algorithm that provides real-time detection via neural networks. In contrast to conventional neural network algorithms, which use both forward and backward propagation, YOLO employs only forward propagation, making it faster and easier to compute. Different researchers have used YOLO for applications such as car registration detection, traffic sign detection, pose estimation, horse detection, face mask detection, and more. The YOLO family is also a collection of different object detection frameworks that have evolved and improved over the years. Fig 2.19 illustrates the various versions of YOLO models that were developed by different researchers and developers.

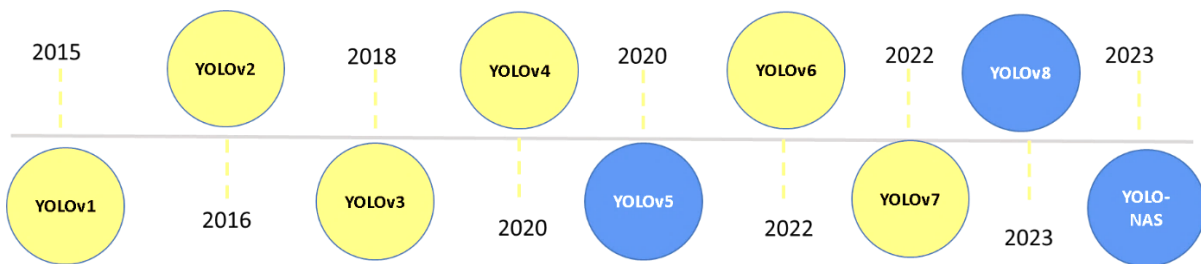


Fig 2.19: Summary of YOLO variant history.

In Fig 2.19, YOLO is written as YOLOv1 for YOLO version 1, YOLOv2 for YOLO version 2, up to YOLOv8. YOLOv5, YOLOv8, and YOLO-NAS are not official versions of YOLO as they have no official publication. Table 2.7 shows the summary of the YOLO variants.

Table 2.7: The summary of the YOLO variants.

Version	Architecture	Advantages	Disadvantages
YOLOv1 Redmond et al. [30]	Googlenet	Groups together different object techniques into a single object detection neural network.	Large localisation errors and low recall.
YOLOv2 Redmond and Farhadi [31]	Darknet-19	Can detect over 9000 objects at multiple scales.	Struggles to detect small objects.
YOLOv3 Redmond and Farhadi [32]	Darknet-53	Can detect small objects, which were problematic in YOLOv2.	Suffers in producing accurate results for medium and large-sized objects.
YOLOv4 Bochkovskiy et al. [33]	Backbone is CSP Darknet-53 and Neck is SPP and PAN	Can detect objects of various sizes faster and more accurately than previous versions and can take any input without resizing or reshaping it.	It has low recall and more localisation errors than Faster R-CNN.
YOLOv5 No published paper	CSPNet and CSPDark-53	Fast, versatile, accurate, efficient, and easy to implement than previous versions.	Requires large dataset for training, is not compatible with previous versions.
YOLOv6 Meituan [34]	EfficientNet-L2	The utilisation of a modified version of the EfficientNet architecture, specifically EfficientNet-L2, is employed. The architecture employed in this model exhibits greater	Slower in inference than yolov5

efficiency compared to EfficientDet utilised in YOLO v5, as consists of reduced parameter count and enhanced computational efficiency.

YOLOv7 Wang et al. [35]	E-ELAN	Introduces a loss function called focal loss to improve the detection of small objects and has a higher resolution than the previous versions. This variant also has high speed.	This version can be called computational intense when running on CPU.
YOLOv8 No published paper	No insight as there is no published paper.	Promises to make training easier on CPU and GPU using API. Caters to a wide range of object segmentation tasks.	Not sufficient information to deduce as this variant is new.
YOLO-NAS No published paper	No insight as there is no published paper.	Not sufficient information to deduce as this variant is new.	Not sufficient information to deduce as this variant is new.

Single Short Multi Box Detector (SSD):

The SSD consolidates all computational tasks into a single feed-forward CNN in order to directly estimate the offsets of bounding boxes and the scores of object categories. Consequently, the necessity of generating bounding box proposals and subsequently resampling features or pixels is obviated. The proposed approach employs a predetermined set of standardised boxes, referred to as anchor boxes or anchors, which are thoughtfully selected by the developers. These boxes are chosen based on the anticipated sizes of the objects to be detected, and this selection process is performed in advance. The purpose of these boxes is to provide an approximation of the output space of bounding boxes across various aspects and scale ratios at each location on the map. In a precise manner, the SSD has the capability to generate predictions regarding the offsets and anchor shapes at each cell of the feature map.

These predictions encompass category scores, which indicate the presence of object classes within each anchor. Moreover, SSD integrates its estimations and predictions derived from the feature maps to effectively handle objects of varying sizes. The initial layers of the SSD model are constructed following a conventional architecture commonly employed for the classification of high-resolution images. Thereafter, an extra architectural part is added to the network in order to make feature maps that can be scaled across multiple levels. This makes it easier to use these maps in the detection process. The architecture is made up of convolutional feature layers, which are meant to reduce the size of the feature maps and make it possible to predict detection at different scales.

2.4 PERFORMANCE METRICS

In the development stages of an object detection model, researchers or developers analyse their models to understand their performance. In object detection and computer vision, the widely used performance metrics are confusion metrics, intersection of union (IOU), recall, average precision (AP), mean average precision (mAP), losses, and F1-score.

2.4.1 Confusion matrix and Intersection of Union (IOU)

A common representation for the summary of predictions made by a trained model is the confusion matrix. The table provides data on the accuracy of predictions for each class or label. The assessment of accuracy is achieved through the calculation of statistical metrics such as true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), as demonstrated in Fig 2.20.

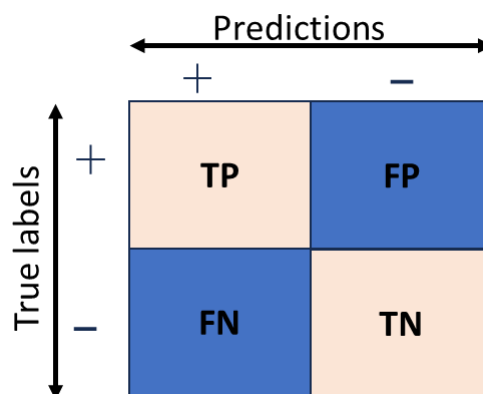


Fig 2.20: Confusion matrix used to explain performance key indicators for object detection models.

According to Sibiya and Sumbwanyambe [20], confusion matrix terms can be defined as follows:

TP is when a model correctly predicts the positive classes. For example, in the case of the proposed study, a positive class prediction is when a driver or a vehicle occupant is buckled, and a negative class prediction is when a driver or a vehicle occupant is unbuckled. So, if the model predicts that a passenger or a driver is buckled and is indeed buckled, then that is known as the true positive prediction.

TN is when a model correctly predicts the negative classes. If we take the proposed study as an example, then that would mean that the model must predict a driver's or passenger's safety belt as unbuckled when it is really unbuckled.

FP is when a model incorrectly predicts a positive class.

FN is when a model incorrectly predicts a negative class.

According to Padilla et al. [36], the concept of intersection of union (IOU) relates to the determination of accurate and inaccurate detections. The IOU metric quantifies the extent of overlap between the predicted bounding box and the ground-truth bounding box relative to their combined area. Mathematically, this can be expressed as follows:

$$IOU = \frac{\text{area of overlap } (B_p \cap B_{gt})}{\text{area of union } (B_p \cup B_{gt})} \quad (2.9)$$

Fig 2.21 and F2.22 show a concept illustration and a concept implementation of the IOU.

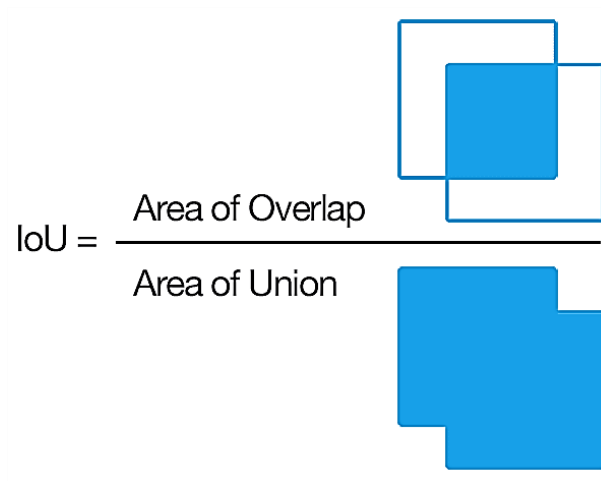


Fig 2.21: A pictorial illustration of an IOU concept.

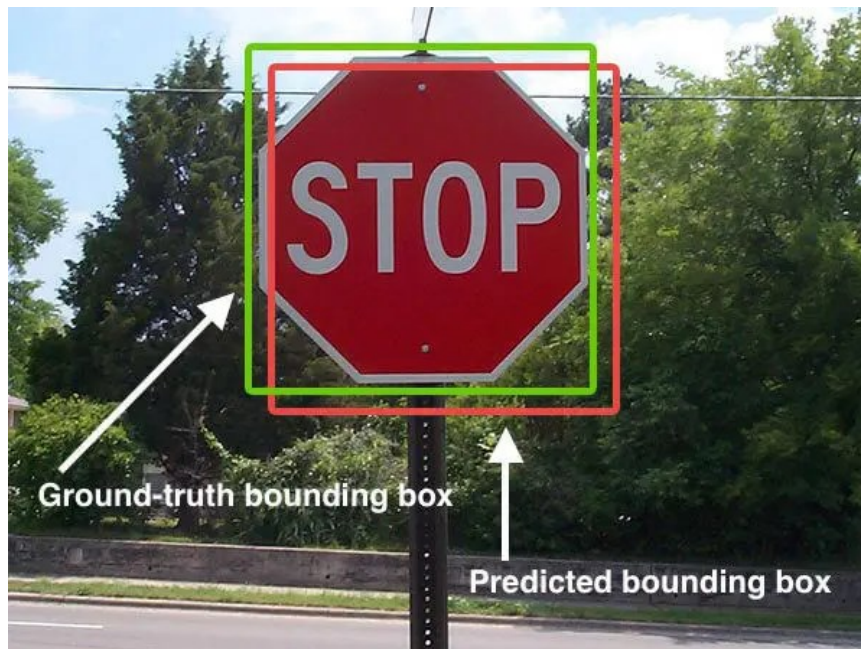


Fig 2.22: IOU example in traffic sign detection.

2.4.2 Recall, Average Precision, and Mean Average Precision

Recall is the ability of a model to find and detect all the relevant classes within the dataset. Mathematically, this can be defined as the number of TP divided by TP plus FN; see (2.10).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \quad (2.10)$$

Precision is the ability of a model to identify relevant objects. Mathematically, precision can be defined as the number of TP divided by the number of TP plus FP, as shown by (2.11).

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \quad (2.11)$$

The precision-recall curve can be interpreted as a balancing act between precision and recall across various confidence values assigned to the bounding boxes produced by a detector. If the confidence level of a detector is such that its FP rate is low, it can be inferred that the precision of the detector will be high. Nevertheless, in this particular scenario, there is a possibility of overlooking several positive instances, resulting in a significant number of FN and consequently leading to a low recall rate. On the other hand, if one chooses to include a greater number of positive instances, the recall metric will experience an increase. However, it is important to note that this decision may also result in an increase in FPs, consequently leading to a decrease in precision. Nonetheless, an effective object detector must be capable of detecting all objects that are considered ground truth (FN = 0, indicating high recall) while simultaneously ensuring that only relevant objects are identified (FP = 0, indicating high precision). Hence, the efficacy of a specific object detector can be deemed satisfactory if its precision remains consistently high while its recall increases. This implies that even when the confidence threshold fluctuates, both precision and recall will maintain a high level. Therefore, a substantial area under the curve (AUC) generally suggests the simultaneous presence of elevated precision and recall. Regrettably, in real-world scenarios, the precision-recall plot frequently exhibits a zigzag pattern, which presents difficulties in accurately assessing its AUC. The elimination of the zigzag behaviour in the precision-recall curve is achieved through a preprocessing step before estimating the AUC. There are two primary methodologies for achieving this objective: 11-point interpolation and all-point interpolation. The precision-recall curve in the 11-point interpolation method is characterised by calculating the average of the maximum precision values at 11 recall levels that are equally spaced, ranging from 0 to 1. This calculation is represented by (2.12).

$$\text{AP}_{11} = \frac{1}{R_L} \sum_{R \in \{0, 0.1, \dots, 0.9, 1\}} P_{\text{interp}}(R), \quad (2.12)$$

Where:

$R_L = 11$ which is the number of recall levels.

And:

$$P_{\text{interp}}(R) = \max_P(\bar{R}) \max_{\bar{R}: \bar{R} \geq R} P(\bar{R}) \quad (2.13)$$

The current definition of AP involves utilising the precision $P(R)$ observed at various recall levels R . However, an alternative approach to calculating AP involves considering the maximum precision value $P_{\text{interp}}(R)$ where the corresponding recall value exceeds R . In the context of all-point interpolation, it is possible to perform interpolation through all available points rather than limiting the interpolation to only 11 equally spaced points.

$$AP_{\text{all}} = \sum_n (R_{n+1} - R_n) P_{\text{interp}}(R_{n+1}), \quad (2.14)$$

Where:

$$P_{\text{interp}}(R_{n+1}) = \max_P(\bar{R}) \max_{\bar{R}: \bar{R} \geq R_{n+1}} P(\bar{R}) \quad (2.15)$$

In this scenario, interpolating precision values at various levels determines the AP rather than relying solely on precision measurements at a small number of data points. Specifically, the maximum precision is selected when the corresponding recall value is equal to or greater than R_{n+1} .

The mAP is a quantitative measure employed to assess the precision of object detection models across all categories within a given dataset. The mathematical representation of this can be expressed as:

$$mAP = \frac{1}{N} \sum_{i=1}^N AP_i \quad (2.16)$$

Where AP_i is the number of AP and N is the total number of the evaluated classes.

F1-score It is the harmonic means of precision and recall. Mathematically F1-score can be expressed as:

$$F1 - \text{score} = 2 \times \left(\frac{\text{Precision} \times \text{Recall}}{\text{Recall} + \text{Precision}} \right) \quad (2.17)$$

2.5 OPTIMISATION

Optimisation plays a vital role in achieving the best possible results. When a model is not optimised, it might suffer from either overfitting or underfitting issues [20]. Sibiya and Sumbwanyambe [20] suggest that these problems can be tackled by adjusting hyperparameters. Hyperparameters are settings given to a model by the developer. There are many parameters that can be fine-tuned to optimise a model. Here are some parameters that can be adjusted to optimise a model:

Hyper-parameters related to the neural network structure:

1. The sequence of layers involves convolution, pooling, and dropout.
2. The choice of activation function.
3. The quantity of hidden neurons.
4. The arrangement of pooling and convolutional layers.

Hyper-parameters related to training models:

1. Learning rate.
2. Number of epochs and batch size.

Various other methods for optimising machine learning and deep learning involve regularisation techniques. Presented below are some of the optimisation methods employed:

1. L1 and L2 regularisation
2. Elastic Net Regularisation
3. Dropout
4. Batch Normalisation
5. Early Stopping
6. Data Augmentation
7. Learning Rate Scheduling
8. Weight Decay
9. Gradient Clipping
10. Transfer Learning

11. Ensemble Methods

These techniques contribute to enhancing the performance and generalisation of machine learning and deep learning models. Many other techniques can be employed to optimise a model. Fig 2.23 shows the optimisation flow diagram that can be followed to optimise a model.

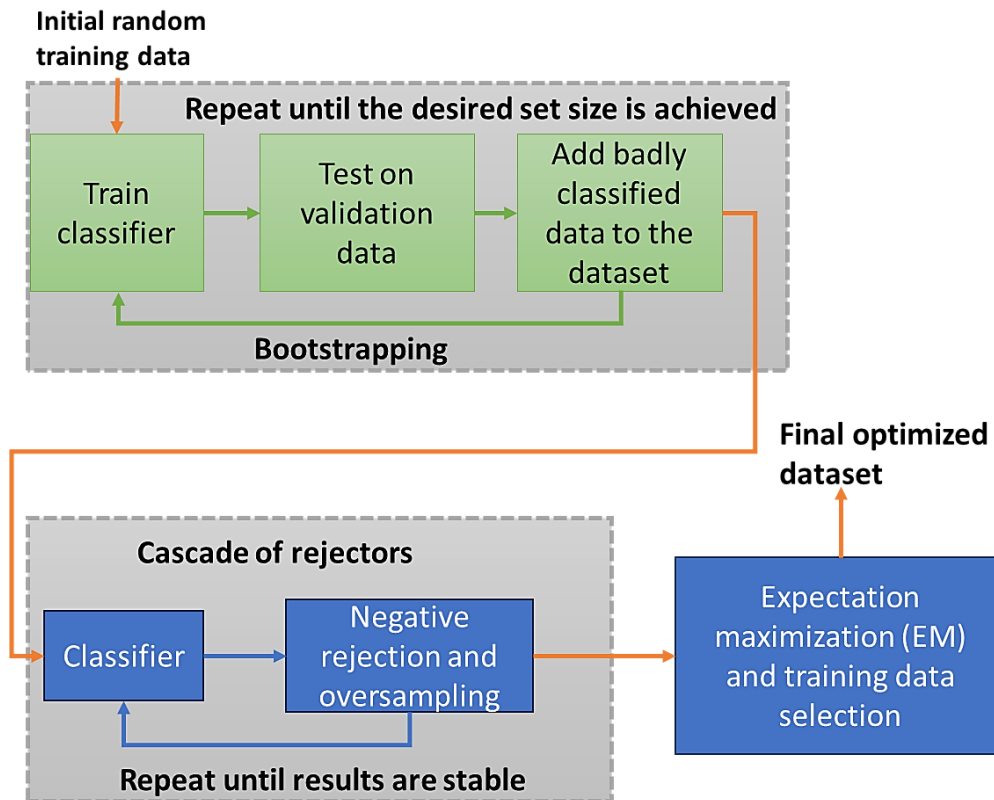


Fig 2.23: Optimisation flow diagram.

Understanding the mathematical concepts for both forward and back-propagation algorithms is an essential method for improving the neural networks used in the training process. It helps the network learn better. Here, we demonstrate how a neural network with two hidden layers learns through backpropagation.

Forward Propagation:

The forward propagation computes the predicted output of the neural network based on the input data and the current weights. Fig 2.24 shows an example of a feedforward neural network.

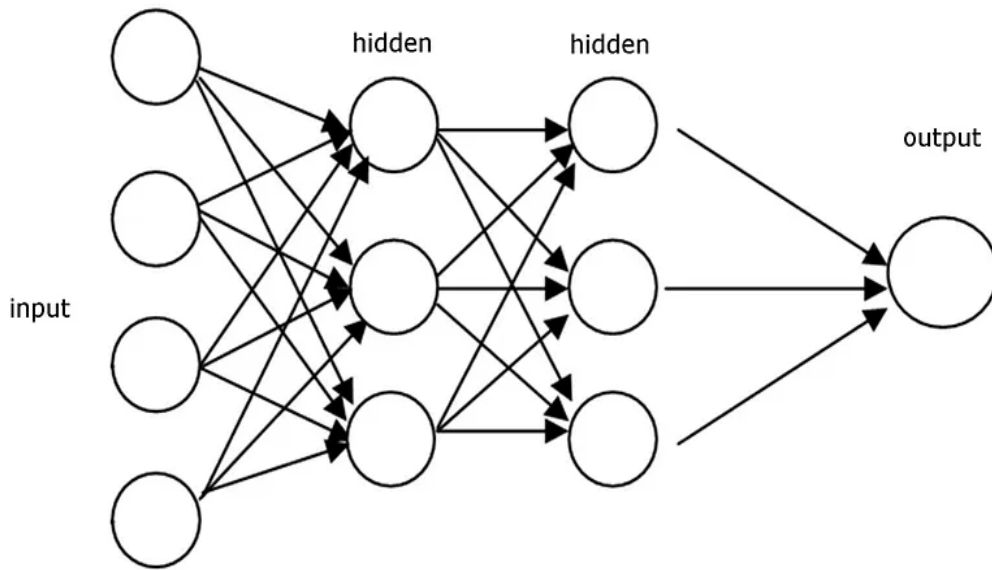


Fig 2.24: Example of a Feedforward Neural Network.

For each training example i , the forward propagation is as follows:

a) Input layer:

The activations of the input layer are just the input features, denoted as by x_i .

b) First hidden layer:

The activations of the first hidden layer (without the bias term) are given by:

$$Z_{1i} = W_1 \times x_i + b_1 \quad (2.18)$$

And

$$a_{1i} = \text{sigmoid}(Z_{1i}) \quad (2.19)$$

c) Second hidden layer:

The activations of the second hidden layer (without the bias term) are given by:

$$Z_{2i} = W_2 \times a_{1i} + b_2 \quad (2.20)$$

And

$$a_{2i} = \text{sigmoid}(Z_{2i}) \quad (2.21)$$

d) Output layer:

The activations of the output layer (without the bias term) are given by:

$$Z_{3i} = W_3 \times a_{2i} + b_3 \quad (2.22)$$

And

$$a_{3i} = \text{sigmoid}(Z_{3i}) \quad (2.23)$$

Loss Calculation:

The loss (error) for the i-th training example is calculated using the MSE loss function:

$$\text{Loss}_i = \frac{(a_{3i} - y_i)^2}{2} \quad (2.24)$$

where y_i is the actual output for the i-th training example.

Back-propagation:

Backpropagation is applied to adjust the weights of the neural network by moving in the opposite direction, starting from the output layer and going back to the input layer. The objective is to decrease the loss function by modifying the weights. Fig 2.25 illustrates the structural arrangement for the backpropagation of a neural network.

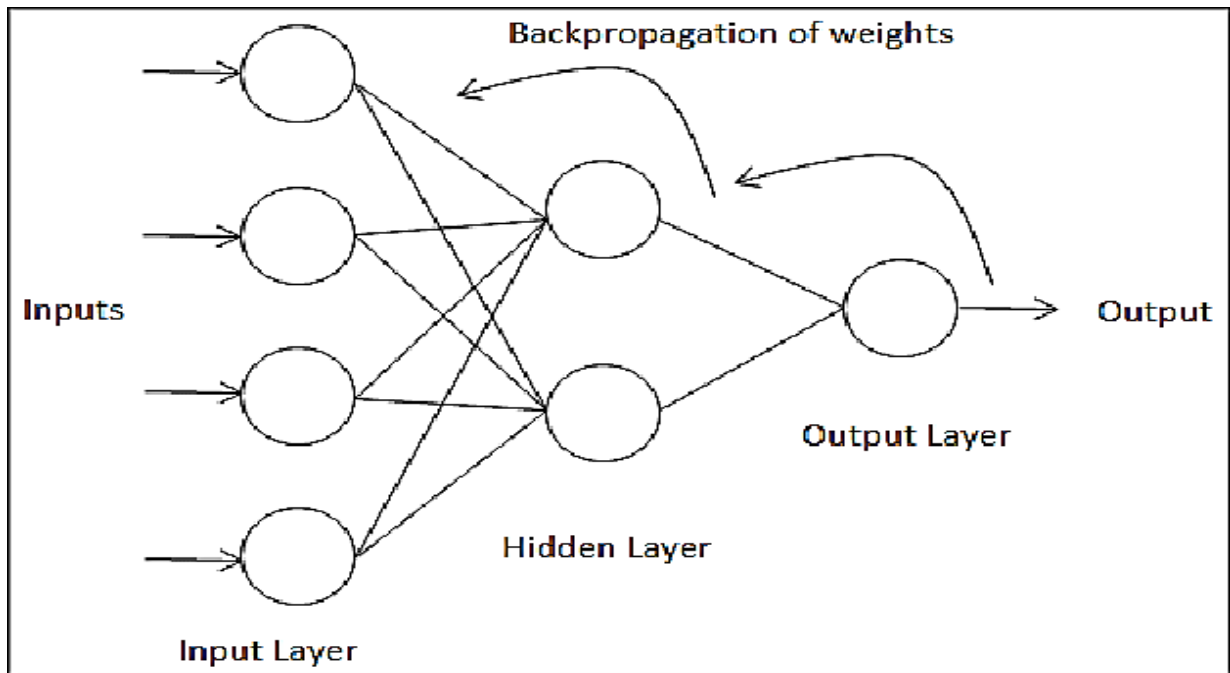


Fig 2.25: Example of backpropagation of weights.

a) Output layer:

The derivative of the loss with respect to the activations of the output layer is:

$$\frac{dL}{da_{3i}} = (a_{3i} - y_1) \quad (2.25)$$

The derivative of the output layer activations with respect to the output layer inputs is:

$$\frac{da_{3i}}{dz_{3i}} = \text{sigmoid}'(Z_{3i}) = a_{3i} \times (1 - a_{3i}) \quad (2.26)$$

The delta (error) of the output layer is given by:

$$\Delta_{3i} = \frac{dL}{da_{3i}} \times \frac{da_{3i}}{dz_{3i}} \quad (2.27)$$

b) Second hidden layer:

The derivative of the output layer inputs with respect to the activations of the second hidden layer is:

$$\frac{dz_{3i}}{da_{2i}} = W_3 \quad (2.28)$$

The derivative of the second hidden layer activations with respect to the second hidden layer inputs is:

$$\frac{da_{2i}}{dz_{2i}} = \text{sigmoid}'(Z_{2i}) = a_{2i} \times (1 - a_{2i}) \quad (2.29)$$

The delta (error) of the second hidden layer is given by:

$$\Delta_{2i} = \frac{dz_{3i}}{da_{2i}} \times \frac{da_{2i}}{dz_{2i}} \times \Delta_{3i} \quad (2.30)$$

c) First hidden layer:

The derivative of the second hidden layer inputs with respect to the activations of the first hidden layer is:

$$\frac{dz_{2i}}{da_{1i}} = W_2 \quad (2.31)$$

The derivative of the first hidden layer activations with respect to the first hidden layer inputs is:

$$\frac{da_{1i}}{dz_{1i}} = \text{sigmoid}'(Z_{1i}) = a_{1i} \times (1 - a_{1i}) \quad (2.32)$$

The delta (error) of the first hidden layer is given by:

$$\Delta_{1i} = \frac{dz_{2i}}{da_{1i}} \times \frac{da_{1i}}{dz_{1i}} \times \Delta_{2i} \quad (2.33)$$

Weight Update:

Once we have the deltas for each layer, we can update the weights to minimise the loss function. The weight updates for each layer are as follows:

a) Output layer weights:

$$W_3 = W_3 - \text{learning_rate} \times \Delta_{3i} \times a_{2i} \quad (2.34)$$

$$b_3 = b_3 - \text{learning_rate} \times \Delta_{3i} \quad (2.35)$$

b) Second hidden layer weights:

$$W_2 = W_2 - \text{learning_rate} \times \Delta_{2i} \times a_{1i} \quad (2.36)$$

$$b_2 = b_2 - \text{learning_rate} \times \Delta_{2i} \quad (2.37)$$

c) First hidden layer weights:

$$W_1 = W_1 - \text{learning_rate} \times \Delta_{1i} \times x_i \quad (2.38)$$

$$b_1 = b_1 - \text{learning_rate} \times \Delta_{1i} \quad (2.39)$$

Repeat:

Repeat steps 1 to 4 for all training examples in the dataset for a certain number of epochs or until the loss converges to a satisfactory level. This is the basic mathematical model of how a 2-hidden-layer neural network learns using backpropagation. In practice, there are various optimisations and techniques (e.g., mini-batch training, momentum, weight decay, etc.) used to improve training efficiency and avoid overfitting, but the fundamental principles remain the same.

2.6 OVERFITTING AND UNDERFITTING

Underfitting model: This happens when a line fitted by a machine learning or deep learning model approximates the data poorly, failing to achieve a more accurate representation of the data.

Overfitting model: This situation arises when a model demonstrates a low error rate on the training data, yet it fails to generalise effectively to the broader dataset that encompasses the main area of interest.

Appropriate model: The suitable model neither overfits nor underfits; it effectively generalises to the training data. Fig 2.26 reveals the behaviour of each of the models mentioned above.

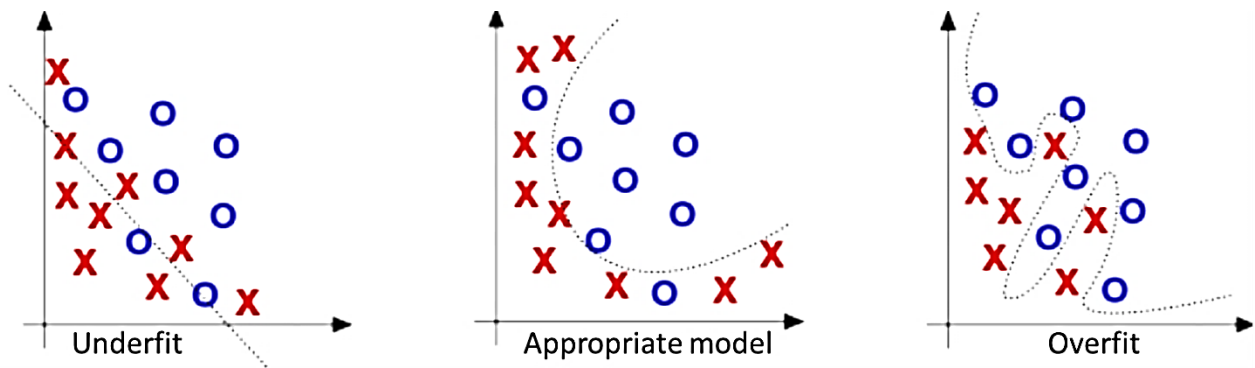


Fig 2.26: An example of the behaviour of each model [20].

2.7 RELATED STUDIES

This study endeavours to employ computer vision for object detection to ensure the proper fastening of a car seat belt, thereby preventing any potential injuries and deaths caused by unbuckling. According to Chen et al. [3], most automobiles are equipped with seat belt warning systems that incorporate sensors within the seat belt buckle to prompt drivers to fasten their seat belts. According to Chen et al. [3], occupants can easily deceive such systems by fastening their seat belts behind their backs. In the existing body of research, little attention has been given to investigating potential solutions for this issue. Most of the research has concentrated on electronics and conventional algorithms of computer vision, which are unreliable and imprecise in diverse settings [5].

Gupta et al. [37] proposed a study on the utilisation of RF (radio frequency)-based methodology to detect speed and seat belt systems. The study employed a pressure sensor, an infrared (IR) sensor, a seat belt latch, and a microcontroller to ascertain the usage of seat belts by the driver and the front passenger. According to their proposal, if the signal received from the seatbelt latch is not detected, an error signal will be transmitted to a microcontroller, which will subsequently immobilise the car's engine. Accurate pressure measurement can be achieved when occupants are seated in their designated seats, irrespective of whether they are wearing seat belts. Mazzae et al. [38] proposed a system that utilises an RFID-based latch plate sensor,

seatbelt webbing pay-out sensor, D-ring angle sensor, and occupancy detection sensor to detect instances of seat belt misuse. The study employed 34 distinct data sets of individuals with varying body sizes to assess the incidence of seat belt misuse. One of the key observations was that the seat belt system functioned optimally when utilised by individuals within the system's specified range, specifically those belonging to the 5th percentile of females and the 95th percentile of males. Leland et al. [1] proposed a methodology for human-centric activity recognition to detect whether a user has fastened their seatbelt. The proposed approach involves recognising physical activities using the wrist smartwatch sensor. The data acquired from the sensor was subjected to a sampling rate of 25 Hz to facilitate processing, as reported by [1]. The scholars posited that a significant constraint of their investigation was the prerequisite for the subjects to wear a smartwatch, which was deemed outdated.

Continued endeavours have been made towards the detection of seat belts through computer vision techniques, as evidenced by the work of Chun et al. [39]. Most current techniques rely on conventional algorithms, including edge detection and HT, as noted by [3]. Lee [19] posits that conventional algorithms have a historical origin of 10–20 years and are distinguished by their ability to extract human-engineered features (such as edges, corners, and colour) that are considered pertinent in visual tasks. According to Hosameldeen [40], vehicle factors such as the driver's clothing and steering wheel can have an impact on the performance of edge detection and HT techniques. The existence of numerous vehicle types and background environments can result in a multitude of potential edges, leading to the generation of false positives in these methods, as noted by [3]. The techniques are commonly known as conventional image recognition methods. However, they often encounter issues related to inadequate feature extraction and low robustness when dealing with intricate application scenarios, as noted by [41]. As per Hosameldeen's [40] findings, most algorithms rely on edge detection as a fundamental technique. The initial step in this process involves the detection of the horizontal and vertical edges of the vehicle [40]. Subsequently, non-vehicle edges are removed, and an estimation of the symmetry of vertical edges is conducted. The research methodology employed in this study relies on edge detection, which has been found to exhibit inaccuracies when applied in varying environmental contexts [3].

Significant advancements have been made in the realm of computer vision in terms of object detection. Chen et al. [3] proposed a seat belt detection system that utilises CNNs and SVMs to address the impact of various factors, such as illumination from edge detection, vehicle body,

and driver's clothing, on the seatbelt detection process. The methodology employed was found to solely identify the fastening of the driver's seatbelt while neglecting to detect the fastening of the passenger's seatbelt. Naik [42] has put forth a proposal for detecting driver's seatbelt usage through the utilisation of CNNs. Their methodology involved detecting the driver's seat belt rather than that of the front-seat passenger. Both studies employed an external vehicle detection approach utilising traffic surveillance camera in their research. Previous research has demonstrated a keen interest in alternative investigative techniques, such as YOLO, which have been recognised for their speed and precision. The development of swift and precise algorithms for detecting objects has the potential to facilitate the operation of autonomous vehicles without the need for dedicated sensors, provide assistive technologies with the ability to communicate real-time environmental data to human users, and unleash the possibilities of versatile and reactive robotic systems [31].

The use of a safety belt is of utmost importance in the prevention of severe injuries and significantly decreasing the likelihood of fatalities [42–45]. According to a study by Khohli and Chadha, accidents claim the lives of 3287 people on average every day [11]. According to Leland et al. [1], it has been suggested that the projected annual fatality count arising from motor vehicle accidents is approximately 1.35 million. Despite the implementation of safety belt monitoring systems in contemporary vehicles, which function as reminders for individuals to fasten their seat belts, this phenomenon continues to endure. In SA, a distinct division of law enforcement is dedicated to the enforcement of safety belt usage and the promotion of comprehensive safety protocols. Nevertheless, a considerable proportion of individuals who are travelling in vehicles opt not to fasten their seatbelts. The topic of safety belt detection holds considerable importance due to its ability to protect human lives.

Other than the use of traffic officers on highways [44] and reliance on safety belt warning systems that are simple for vehicle occupants to bypass, there are currently few efficient ways to ensure and monitor the proper securing of safety belts. The current methodology is characterised by its arduous and inefficient nature, primarily due to the restricted number of officers assigned to do inspection duties [44]. Furthermore, an important challenge emerges from the capacity of individuals in vehicles to manipulate the safety belt system by either securing the belt behind their backs or employing a device referred to as a "seatbelt warning stopper" to circumvent the system [43]. The current safety belt warning systems exhibit a restricted level of effectiveness. Occasionally, drivers perceive traffic officers from a

significant distance and subsequently choose to fasten their seat belts at that specific moment in order to ensure their safety. As a result, there has been a growing interest in the automated detection and enforcement of safety belt regulations [45]. Academic researchers have been working to improve computer vision algorithms to meet the demand of traffic safety organisations around the world for automated solutions for safety belt detection that don't just use onboard sensors.

Scholars have looked at the issue of safety belt compliance using two different approaches. One method entails the deployment of surveillance cameras along highways in order to observe and record compliance with safety belt usage [1, 18, 11]. However, it is important to take into account that the installed cameras must consider the existence of two modes of computer vision models, namely single-stage and two-stage. Consequently, it is recommended to utilise stage 1 computer vision models for real-time detection [35, 46]. Additional researchers, including Hosameldeen [40], Chen et al. [3], and Tianshu et al. [47], have also explored the implementation of surveillance cameras along highways. The second methodology involves the installation of cameras inside vehicles to continuously observe the practice of securing safety belts [5, 43, 48]. Both proposed solutions face several challenges. The deployment of safety belt monitoring systems on highways offers a viable solution to the challenge of insufficient law enforcement personnel. However, it does not adequately tackle the fundamental issue that drivers may choose to adhere to safety belt regulations selectively in regions where they are cognizant of the existence of surveillance cameras. There is potential for the onboard monitoring system to efficiently enforce the use of a car safety belt. However, the process of onboard monitoring faces ongoing difficulties related to varying levels of illumination, the occurrence of blurred images due to uneven road conditions, and the obstruction caused by objects inside the vehicle.

As mentioned earlier, the current design of the safety belt monitoring mechanism is vulnerable to manipulation. The safety belt mechanism consists of several constituent elements, such as the webbing, latch plate, buckle, pillar loop, and retractor, as depicted in Fig 1.1 in Chapter 1. The buckle component is equipped with a metallic sensor that has been specifically designed to detect the presence of the latch plate. Furthermore, the seat is equipped with a pressure sensor that is responsible for quantifying the magnitude of the force exerted by the occupant's weight when they are in a seated position. In the event that the driver, the front passenger, or both occupants fail to utilise seat belts and their weight is detected on the seat, an alarm is triggered.

The alarm persists until a buckle is securely fastened and it detects the presence of a latch plate. However, this specific methodology for detecting safety belts has limitations, as it may not accurately differentiate between a properly fastened belt and a latch plate that has been inserted behind the person's back. Additionally, there exists an alternative product referred to as the "seatbelt warning stopper," which operates as a metallic apparatus inserted into the buckle to imitate the presence of a latch plate [43]. The use of vehicle occupant weight as a metric for determining seat occupancy is susceptible to inaccuracies, as there may be instances where a seat belt alarm is triggered due to the presence of a pet or a significant object placed on the seat.

This study introduces a novel methodology to tackle a significant deficiency in existing safety belt systems by incorporating cutting-edge computer vision technology. Through the utilisation of sophisticated image analysis techniques, we have successfully created an innovative YOLOv7 model that proficiently identifies the precise condition of car seat belt fastening. This model effectively addresses the difficulties presented by deceptive practices, including the manipulation of the belt by routing it behind the occupant's back or the use of aftermarket devices to generate misleading buckle signals. In contrast to traditional systems that are vulnerable to manipulation, the proposed solution utilises real-time visual data to achieve a dependable and tamper-resistant method for detecting the status of seat belts. The continuous monitoring capability of the model is indicated by an audible alert, which serves to discourage unsafe practices and effectively reduces the risk of potential fatalities by encouraging the genuine usage of seat belts. This study represents a notable progression in the field of automotive safety through the utilisation of computer vision technology. By leveraging the accuracy and quick response of computer vision, this research aims to enhance the effectiveness of seat belt enforcement, thereby creating a safer environment for both drivers and passengers.

As emphasised, this study employs computer vision object detection techniques. The field of computer vision in object detection can be classified into two main methodologies: single-stage object detection and two-stage object detection [46], as mentioned earlier. Single-stage detectors, exemplified by YOLOv7, demonstrate superior computational efficiency in comparison to two-stage detectors due to their exclusion of the RPN. Based on the research conducted by Wang et al. [35], it has been observed that YOLOv7 exhibits enhanced performance in comparison to existing object detection models with regards to both speed and

accuracy. Based on the findings of the authors, YOLOv7 exhibits a range of performance speeds, spanning from 5 to 160 frames per second (FPSs). Additionally, it attains the utmost level of precision, as indicated by an AP value of 56.8%. YOLOv7 is positioned as the foremost real-time object detection system in terms of accuracy.

2.8 THE SUMMARY OF SAFETY BELT DETECTION LITERATURE

Extensive scholarly research has been devoted to tackling the issue of safety belt detection, with researchers examining potential solutions utilising image-based techniques both within and outside the confines of the vehicle [43]. Multiple researchers [18, 44, 47] have said that external methods for detecting safety belt use could be used instead of only relying on traffic officers to keep an eye on safety belt use on highways. This is because there are not enough people to do this kind of work. Nevertheless, this method exhibits a deficiency in its ability to ensure compliance with safety belt regulations, as individuals inside vehicles have the option to fasten their seat belts solely in regions where external surveillance cameras are present.

The majority of studies primarily concentrate on the detection of the windshield and subsequently identifying the safety belt [3, 40]. Furthermore, the scope of safety belt status detection in numerous studies is restricted to the identification of a single individual's safety belt, specifically either the front passenger or the driver. For example, Elihos et al. [44] introduced a system aimed at detecting the use of the safety belt by the front-seat passenger. In a similar vein, Tianshu et al. [47], Guo et al. [18], and Hossein and Fathi [45] have each contributed to the development of systems specifically designed to detect the use of the safety belt by the driver. Regrettably, these studies fail to consider the safety of additional occupants within the vehicle and instead prioritise the preservation of a single individual.

The implementation of an in-vehicle safety belt detection system is regarded as a potentially effective solution to the aforementioned concerns. According to Jaworek-Korjakowska, Kostuch, and Skruch [48], the concept of in-vehicle detection of safety belts is considered a novel approach that has the potential to improve the functionality of modern cars. However, this approach presents a few obstacles, including the need to address variations in lighting conditions during vehicle movement, the possibility of confusion between the safety belt and the colour of passengers' attire, image blurring caused by vibrations and uneven road surfaces, and obstruction caused by objects such as hands, hair, and clothing [43, 48]. Furthermore, the absence of a publicly accessible dataset presents an additional obstacle.

The issue of safety belt detection remains unresolved, both in terms of internal and external monitoring, due to the intricate nature of these challenges [48]. Nevertheless, a multitude of potential remedies have been suggested in order to tackle these concerns. According to the findings of Kashevinik et al. [5] and Guo et al. [18], numerous suggested approaches for safety belt detection heavily depend on edge detection algorithms, which may exhibit instability when exposed to diverse environmental conditions. Chen et al. [3] agree with this observation and highlight that certain studies employ Hugh transform (HT) algorithms in conjunction with edge detection techniques. The aforementioned techniques can be classified as traditional algorithms, wherein features are manually extracted. Zou et al. [17] have identified several challenges that traditional object detection algorithms encounter, including issues pertaining to speed, accuracy, and environmental variations. Additionally, Luo, Lu, and Yue [49] have noted that conventional algorithms are insufficient when dealing with sizable datasets. On the other hand, deep learning-based methodologies present an alternative strategy. Using these techniques makes it possible to automate the process of extracting features. This makes it possible to offer a solution for safety belt detection that is both faster and more accurate than traditional methods.

The results of multiple studies, which are shown in Table 2.8, demonstrate that deep learning-based algorithms are more accurate than traditional methods. This backs up the submissions of Zou et al. [17]. Deep learning techniques outperform traditional algorithms in terms of accuracy. Nevertheless, it is noteworthy to acknowledge that Hu [43] asserts to have attained a precision exceeding 95% through the utilisation of his shape modelling algorithm, signifying a commendable accomplishment. The algorithm YOLOv5, as demonstrated by Hosein and Fathi [45] in their study conducted in 2023, exhibited the most superior accuracy among the deep learning-based algorithms.

Table 2.8: The summary of the literature related to the safety belt object detection.

Author(s)	Method	Results
Chen et al. [3]	CNN and SVM	Average detection rate: 92.1%
Guo et al. [18]	Edge detection algorithms with HSV colour.	Accuracy@81%
Chun et al. [39]	Using CNN.	Sensitivity@63.51%

		Precision@63.58% F1-score@63.55%.
Hosameldeen [40]	CNN with Alex Net approach.	True positive rate@92% True negative rate@90% True rate@91%
Bhupal Naik et al. [42]	Using SVM and CNN	CNN accuracy@91.4% SVM accuracy@87.17%
Hu [43]	Shape modelling	Accuracy@97.5% (seatbelt ON) Accuracy@100% (seatbelt OFF) Accuracy@98.11% (seatbelt under the arm)
Elihos et al. [44]	SSD, CNN, and Finite Volume (FV) algorithms	Accuracy@91.9% Precision@94.5%
Hosseini and Fathi [45]	Using YOLOv5.	Accuracy@99.7%.
Jaworek- Korjakowska, Kostuch, and Skruch [48]	Adjusted CNN architecture and model interpretability.	Accuracy@79.87% Recall@95.92% Specificity@90.32%
Jha et al. [50]	Segmentation and Synthetic images.	Segmentation F1-score@51% Synthetic F1-score@67%
Kapdi et al. [51]	Using deep learning.	Accuracy@89.1%

CHAPTER 3: MATERIALS AND METHODOLOGY

3.1 INTRODUCTION

The study's design methodology is organised into three primary stages: data collection, data preparation or preprocessing, training, and the model prediction stage, which are extensively discussed in the experimental section. This section also provides an overview of the historical background and operational principles of the YOLOv7 algorithm, which was chosen as the preferred model for this study. The final segment of this section delves into the deployment procedure of the proposed study. The schematic representation of the logical design methodology for the proposed study is depicted in Fig 3.1.

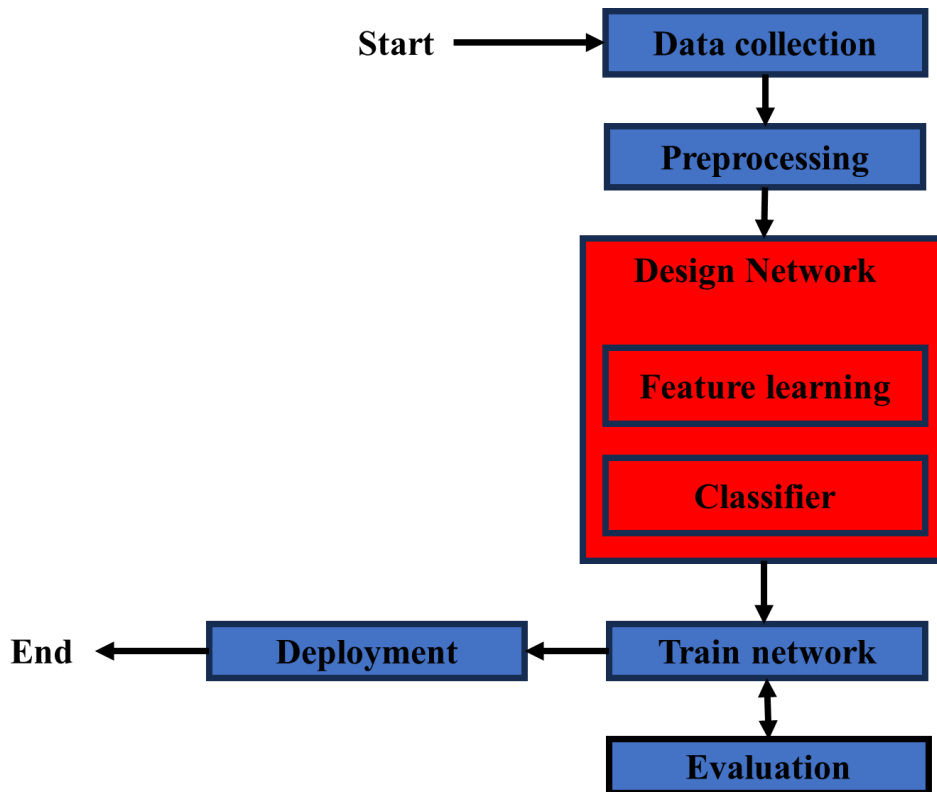


Fig 3.1: Flow diagram of the proposed study.

Chapter one establishes the overarching objectives of the study. Within this section, the aim is to respond to these objectives by expounding upon the chosen design methodology and substantiating key decisions, such as the selection of the object detection method. As a result, the user requirements for this study are articulated as follows:

1. To develop an onboard real-time YOLOv7 model for detecting the safety belt status of vehicle occupants and drivers.

2. To train the model using a GPU.
3. To deploy the trained weights on an Nvidia Jetson Nano development board.
4. To detect the safety belt buckle status in partially occluded conditions, in different lighting conditions, in blurry scenes caused by vibration, and in clothing with similar colours as the safety belt webbing.

3.2 DATA COLLECTION

The main challenge in this study is the insufficient availability of an appropriate dataset, a problem that Chun et al. [39] also acknowledge. In order to tackle this matter, it is proposed to acquire real-time video recordings from the interior of a vehicle in motion. The rationale behind the choice to capture data within a mobile vehicle was to acquire authentic real-world data that encompasses a range of light intensities and the presence of blurred images resulting from the vibrations inherent to vehicular motion. The data collection procedure was conducted during the summer months of November and December 2022 in Cape Town, South Africa. During this time frame, a wide variety of clothing styles were acquired in order to replicate real-world situations. The acquisition of certain datasets was facilitated through the utilisation of crowdsourcing methods, wherein video contributions were actively sought from various social media platforms. All individuals who contributed their videos provided their consent. Furthermore, we acquired supplementary datasets by procuring arbitrary images from the internet in order to enhance the variety of the gathered data. The video dataset was subsequently transformed into images using the VLC media player software. The provided dataset exhibits several scenarios that were examined in the study. Included in the range of scenarios are instances of indistinct images, well-defined images, images with high levels of light intensity, images featuring clothing resembling the safety belt, and images in which the pillar loop was obstructed. The anticipated study was designed to differentiate between the fastened and unfastened conditions of a safety belt in every circumstance. The dataset was augmented through the utilisation of noise addition and brightness contrast techniques. The augmentation approaches yielded advantages such as an augmented dataset size and enhanced model resilience in dynamic in-vehicle environments. Fig 3.2 shows the sample of the collected dataset.



Fig 3.2: The sample of the collected dataset with different apparels, different lighting conditions, blurry images due to vibrations, and occluded seat belt pillar loop.

Video to image conversion process:

After collecting the video dataset, a series of videos had to be converted into images that were later going to be annotated. The VLC media player software program was utilized for conversion. The following procedure outlines the steps taken for this conversion:

- Open VLC on your laptop and select "Convert/Save." under "Media" in the top menu.
- Click "Add" to select the video file to convert.
- Click "Convert/Save" to continue.
- Select the "Destination" folder for the converted images in the "Convert" window.
- Choose a format (e.g., JPEG) and name the converted images.
- Finally, click "Start" to convert.

VLC media player will begin converting the video into a series of individual images and save them in the specified destination folder. In the case of the proposed study, the images were saved under a folder specified as “Seat_belt”. Table 3.1 shows the collected dataset.

Table 3.1: Data collection size.

Dataset source	Number of datasets
Brightness and contrast adjustment augmentation	240
Noise augmentation	120
Original dataset	880
TOTAL	1240

3.3 DATA PREPROCESSING AND PREPARATION

3.3.1 Data Annotation

The annotation process was conducted utilising the labellmg tool, which led to the creation of a label file with buckled and unbuckled classes. The dataset was annotated using the YOLO format, which involved the utilisation of rectangular bounding boxes. The annotation procedure involved the identification of the ROI as the pillar loop of the safety belt webbing. In cases where the pillar loop was not discernible, the corresponding images were labelled as unbuckled. Conversely, in instances where the pillar loop was perceptible, the images were labelled as buckled. Fig 3.3 shows the sample of the annotated dataset to illustrate the annotation process employed in this study.

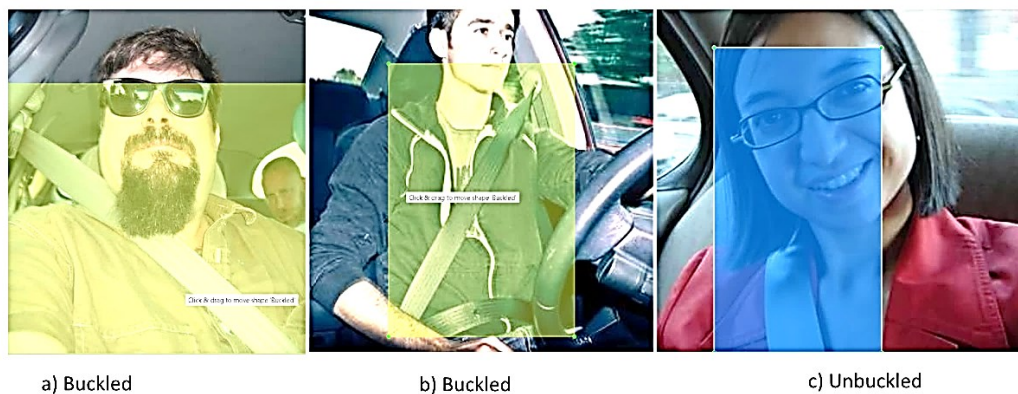


Fig 3.3: The illustration of the annotated dataset for the proposed study.

The annotated images visually depicted the bounding rectangles and were denoted in text form so that they can be retrieved by Pytorch YOLOv7 as the software only reads text files. The initial number in the line column text denoted the object's category or class, represented by an

integer, while the subsequent four numbers indicated the coordinates of the bounding boxes. The first two numerical values represent the coordinates of the centre point of the enclosing rectangle with respect to the X and Y axes, respectively. Subsequently, the aforementioned values are divided by the dimensions of the image, specifically its height and width. The last two numerical values indicate the measurements of the bounding box, where one value represents the width and the other represents the height. The dimensions are subsequently divided by the corresponding width and height of the image. An example of this phenomenon is shown mathematically by (3.1) and (3.2) and illustrated in a graphical format in Fig 3.4.

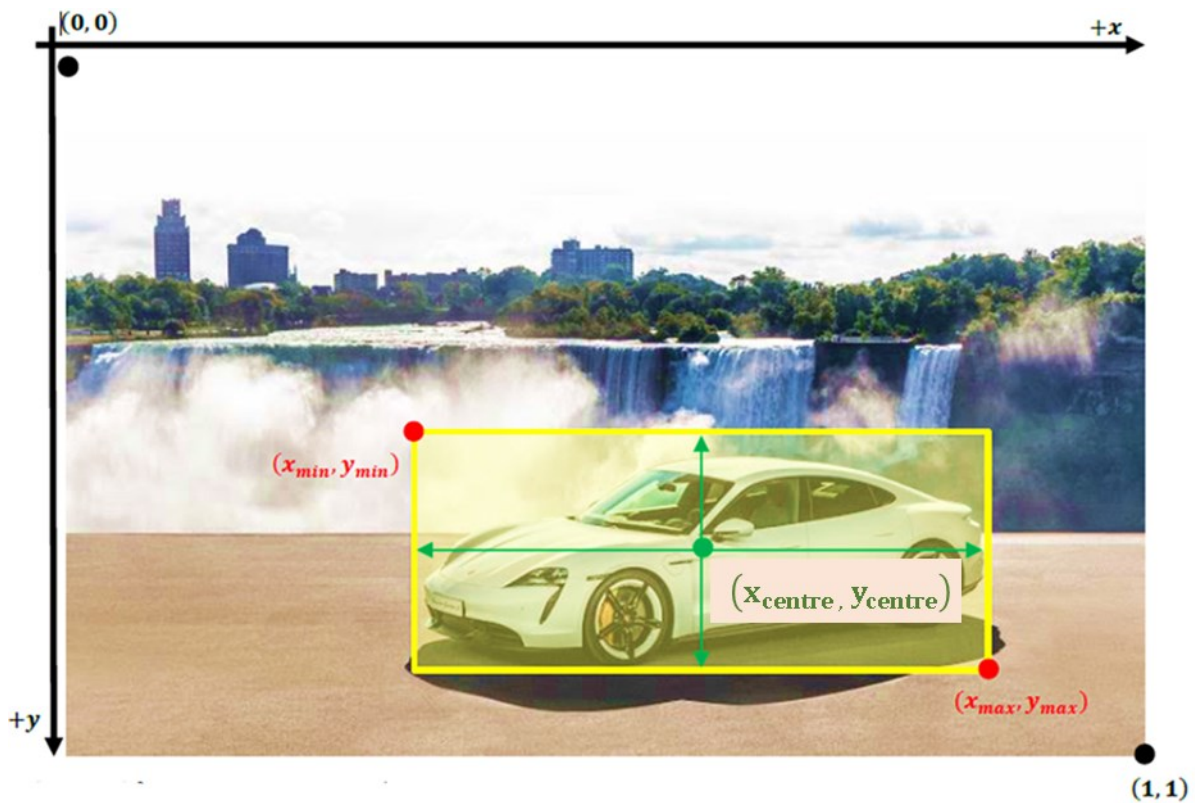


Fig 3.4: The mathematical representation of the YOLO format bounding boxes.

The YOLO bounding box data is structured as [class, x-centre, y-centre, width, height]. It's important to highlight that these bounding box coordinates are normalised within the range of 0 to 1. To achieve this normalisation, x -centre and width are divided by the image width, whilst y -centre and height are divided by the image height. The calculation for determining the centre point (x -centre, y -centre) from the minimum point (x -min, y -min) is as follows:

$$x_{centre} = \frac{(x_{min} + \frac{width}{2})}{image\ width} \quad (3.1)$$

And:

$$y_{centre} = \frac{(y_{min} + \frac{width}{2})}{image\ height} \quad (3.2)$$

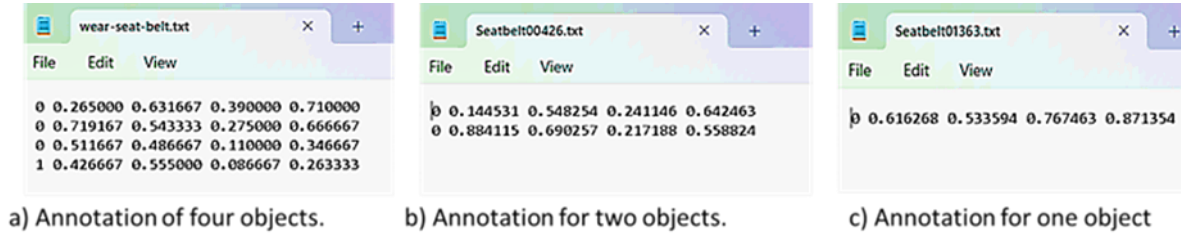


Fig 3.5: The illustration of the annotated dataset in terms of coordinates.

The annotated dataset coordinates are arranged in terms of rows and columns. Each row signifies the presence of an annotated object in an image; for example, one row denotes one annotated object, while two rows indicate the existence of two annotated objects, and so on. The object classes are assigned numerical labels, ranging from 0 to the highest-class number to be detected. In this study, two object classes were predicted: "buckled" and "unbuckled." Thus, the first column of the annotated coordinates contains either a 0 or a 1, representing the class label. The second and third columns denote the x and y centre values of the bounding box, while the fourth and fifth columns specify the width and height of the bounding box. After completing the annotation process, the outcome comprises both the original image and a corresponding text file containing the annotations, as depicted in Figure 3.6.

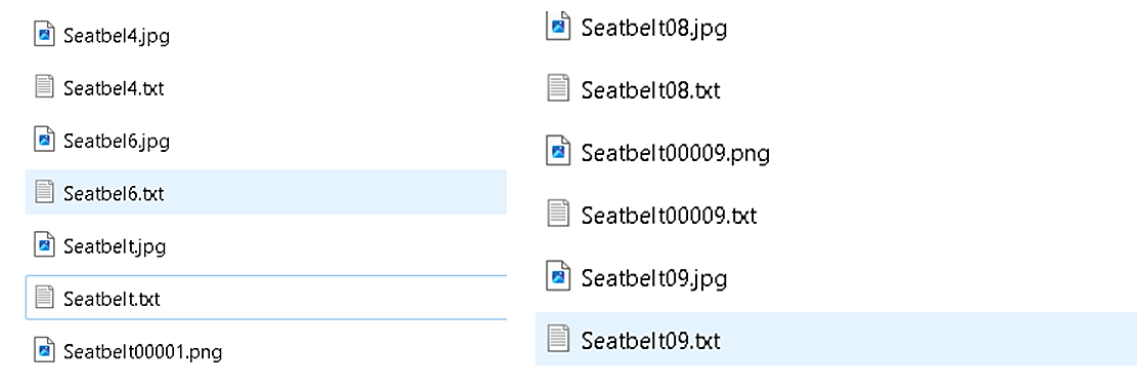


Fig 3.6: An illustration of how the annotated dataset is saved after the text files have been created for each annotated image.

3.3.2 Data Splitting

In order to impartially evaluate the performance of the neural network model, a dataset containing the same type of object as the trained object is essential. However, collecting another dataset with a bunch of images and labels is time-consuming, and sometimes impossible for an AI researcher. In this case, for example, not all researchers can go to the outdoor field and collect the dataset themselves to evaluate the model after the training is complete. So, they usually use the images directly from the given datasets. By dividing it proportionally, the common rate is 80-20, with 80% of the dataset used for training and 20% used for evaluation. Thus, model performance will be assessed on 20% of data that was not used during training, or in other words, this data is data which the model has never seen before. This ensures the review will be equitable. In the proposed study, the dataset was partitioned into three distinct subsets, namely the test, the train, and the validation dataset. The rationale behind the decision to include both a validation dataset and a testing dataset was to mitigate the potential issue of overfitting. The dataset was partitioned into three subsets: 80% for training and 10% each for validation and testing. Fig 3.7 shows the data splitting process from the stored images to the annotated text files which are stored under the labels folder.

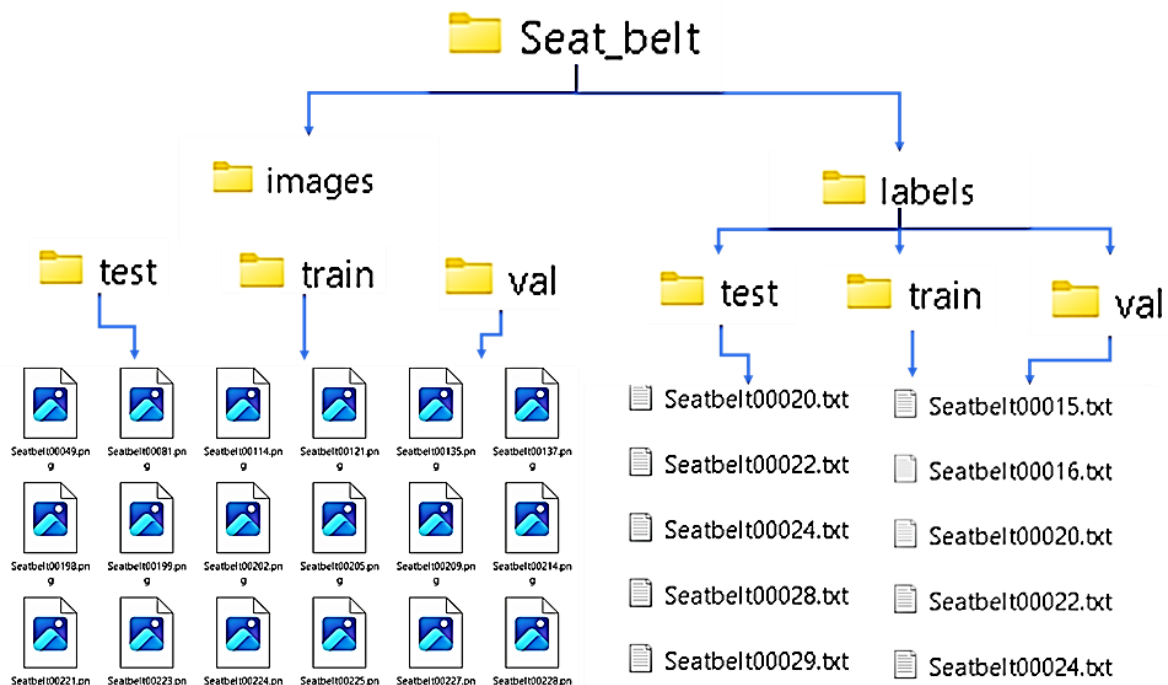


Fig 3.7: Illustration of the dataset splitting process.

The dataset images are stored within the "Seat_belt" directory. Within the "Seat_belt" directory, two sub-folders are created: one for images and another for labels. Both these sub-folders include folders for testing, training, and validation (val) data. The "images" sub-folders contain the dataset images, while the sub-folders within the "labels" directory hold the annotated dataset, represented as text files. It is important to note that the model is trained using text-formatted images, as the PyTorch platform exclusively processes data in this text-based format.

3.4 NETWORK DESIGN

3.4.1 Model Selection

Justification:

The case study outlined in this scholarly document employs the YOLOv7 algorithm [35]. The justification for this is that, despite the development of newer versions of the YOLO algorithm, YOLOv7 has emerged as the most recent iteration preferred by researchers in various fields, as indicated in the available scholarly literature. The YOLO algorithm has proven to be highly effective in automobile-related image tasks. This is primarily attributed to its remarkable inference speeds, which allow for near-real-time image classification. Additionally, the algorithm exhibits a notable capability to achieve high classification performance metrics. Undoubtedly, it is reasonable to affirm that, currently, this algorithm represents the only viable approach that achieves a seamless equilibrium between speed and performance in tasks pertaining to concurrent object detection.

YOLOv7 Performance:

The evaluation of YOLOv7's performance was conducted in reference to previous iterations of the YOLO model (YOLOv4 and YOLOv5), as well as YOLOR, which served as the baseline comparisons [35]. All models underwent training under identical settings. Notably, YOLOv7 exhibits the most favourable equilibrium between speed and accuracy when juxtaposed with contemporary object detection models [35]. In a general context, YOLOv7 outperforms all preceding object detection models in terms of both processing speed and detection precision, with a range spanning from 5 frames per second (FPS) to an impressive 160 FPS. The YOLOv7 algorithm attains the highest degree of accuracy among real-time object detection models, while concurrently sustaining a performance rate of 30 FPS or higher when deployed on a GPU V100 [35]. Fig 3.8 provides a visual representation, illustrating the comparative performance between the preceding YOLO version and the introduced YOLOv7.

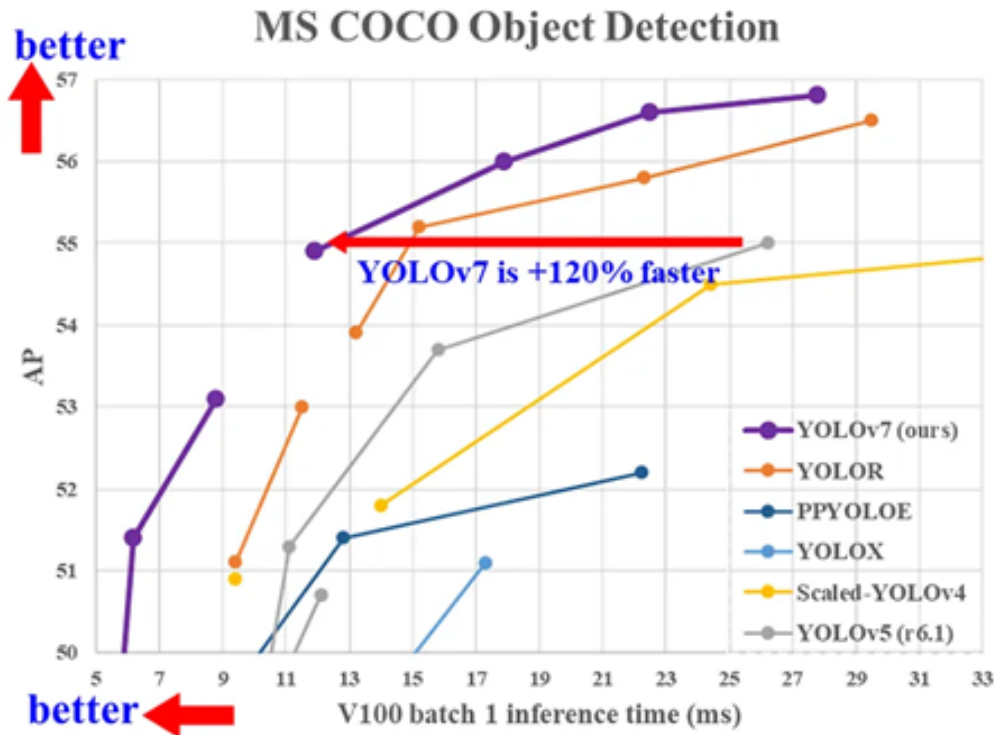


Fig 3.8: Graphical illustration of the YOLOv7 performance when compared to other YOLO variants.

In comparison to the Cascade-Mask R-CNN models that have exhibited superior performance, YOLOv7 attains a notable 2% increment in accuracy, coupled with a substantial enhancement in inference speed, achieving a remarkable 509% acceleration. This accomplishment is of particular significance, as it stands in contrast to R-CNN variants, such as Cascade-Mask R-CNN, renowned for their adoption of multi-step architectures that have historically yielded considerably superior detection accuracies when compared to their single-stage detector architecture counterparts.

The working of the YOLO model:

The YOLO algorithm, first proposed in 2015, employs a methodology that frames object detection as a regression problem. The process entails the segregation of bounding boxes and their associated class probabilities into distinct spatial entities [30]. The processes of classification and bounding box location calculation occur simultaneously. The algorithm begins by partitioning the image into a grid with dimensions $S \times S$ (7×7 is default). Afterwards, predictions are made for each cell in the grid regarding the positions of bounding boxes, the level of confidence in the presence of an object, and the probability of its classification. As a result, the algorithm can complete the entire process of object detection and classification in a

single pass through the image. Fig 3.9 demonstrates the process of grid cell implemented in the YOLO algorithm.

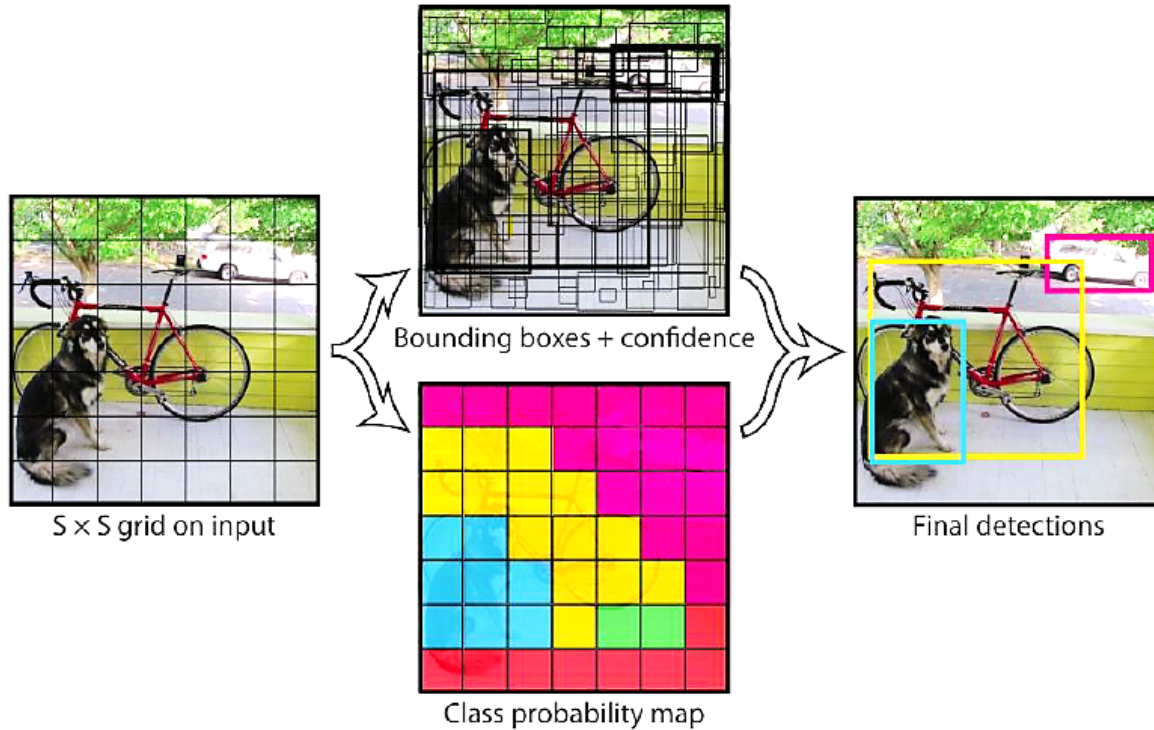


Fig 3.9: The concept of a grid cell used by YOLO models.

The mathematical expression for the total output can be represented as follows, considering that each grid cell predicts B bounding boxes, each described by five parameters:

$$\text{Output parameters} = S \times S \times (5 \times B + C) \quad (3.3)$$

History of YOLO:

The YOLO project is presently undergoing vigorous development, leading to the introduction of subsequent iterations that exhibit enhanced capabilities within recent years. The aforementioned iterations have exhibited enhancements in both the velocity and efficacy of the algorithm [52]. These improvements have been accomplished by making alterations to the structure of the algorithm or by integrating supplementary functionalities. A unique version identifier is assigned to each successive iteration, and currently, the dominant edition of YOLO is YOLO-NAS, which represents the ninth official release. Fig 3.10 shows the progression of the YOLO algorithm, tracing its origins back to its inception in 2015.

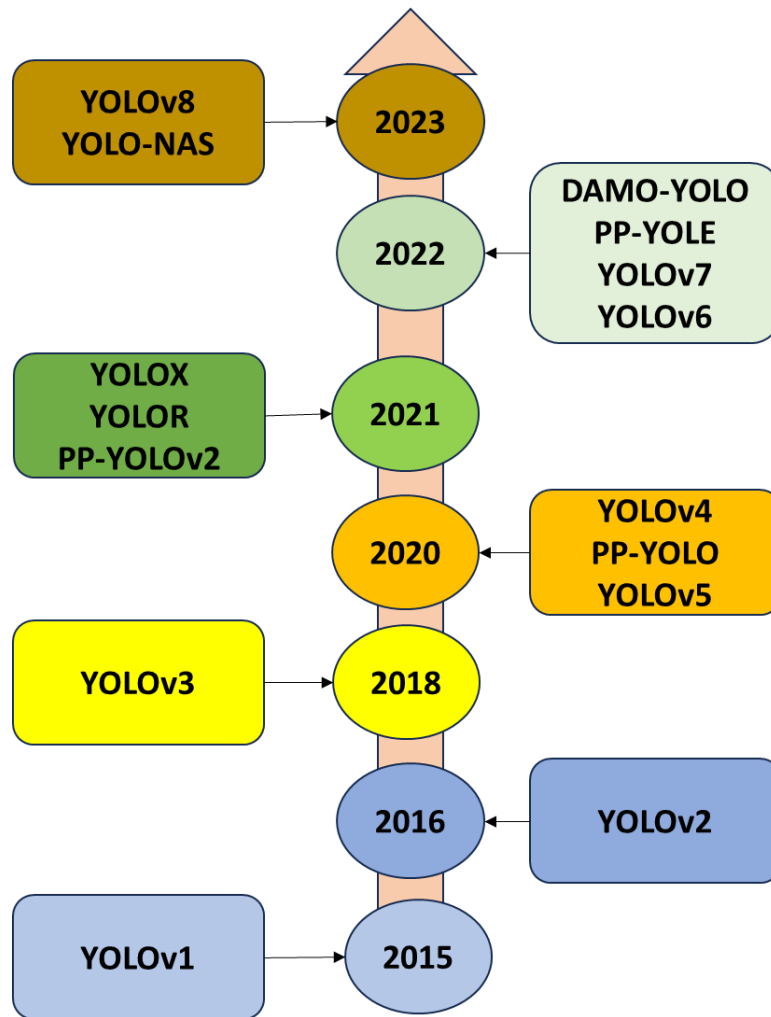


Fig 3.10: The YOLO history.

Architectural structure of the YOLOv7 model:

The YOLOv7 algorithm incorporates a range of techniques, including the extended efficient long-range attention network (E-ELAN), model scaling achieved through concatenation-based models [53], and convolution reparameterisation [35]. These techniques improve the model's ability to withstand challenges and increase its efficiency in accurately when detecting safety belts. The incorporation of diverse strategies in YOLOv7 attains a desirable balance between the efficiency of object detection and the accuracy of the outcomes. The detection methodology employed in YOLOv7 demonstrates similarities with YOLOv4 and YOLOv5, as observed in other network models within the YOLO series. The architectural representation of YOLOv7 can be found in the citation provided as a reference [35]. The YOLOv7 network consists of four distinct modules: the input module, the backbone module, the head module, and the prediction module, as depicted in Fig 3.11.

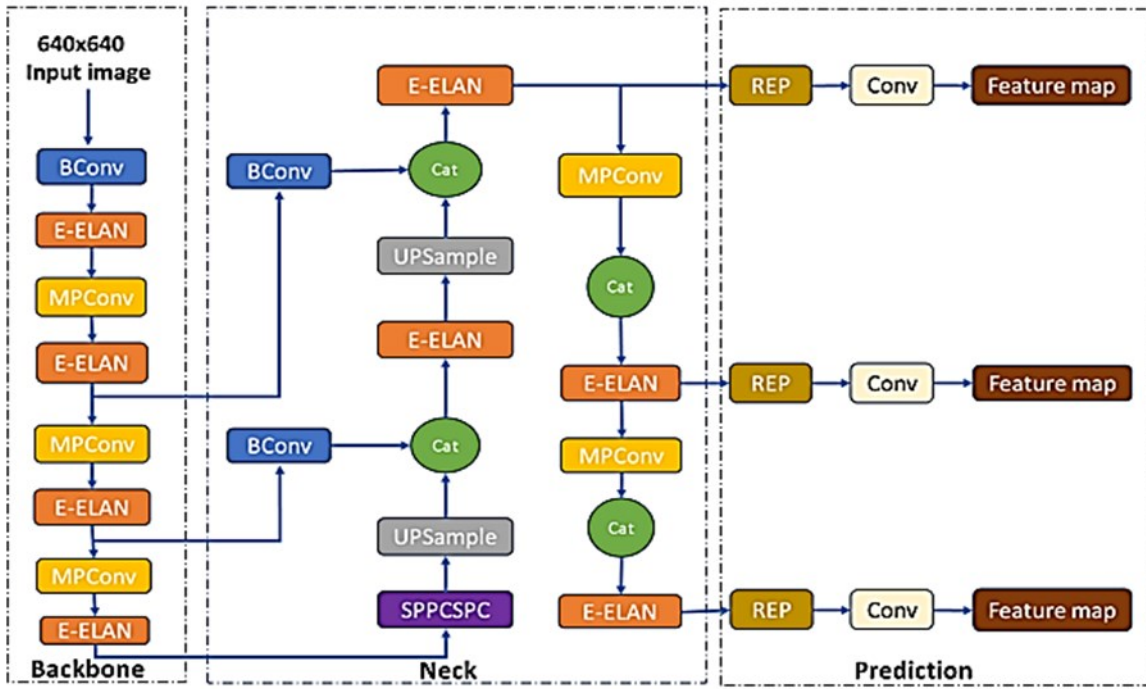


Fig 3.11: The architecture of the YOLOv7 architecture [35].

The backbone module consists of a sequence of Bconv [54] convolution layers, E-ELAN convolution layers, and max-pooling convolution (MPConv) [54] layers. The primary function of the input module in the YOLOv7 architecture is to resize the input image to a consistent pixel dimension, ensuring compliance with the specified input size criteria of the backbone network [55]. The BConv method uses a convolutional layer, a batch normalisation (BN) layer, and a LeakyReLU activation function [54] to capture image features at different scales in a quick and efficient way. This process facilitates the transfer of foundational knowledge from lower levels to higher levels, thereby promoting the effective integration of traits across various levels. The rep-vgg block (REP) structure [55] is used by the prediction module to change the image channel numbers for the P3, P4, and P5 features that come from the outputs of different scales. Subsequently, a 1×1 convolution is employed to generate predictions pertaining to the confidence, class, and anchor frame.

3.4.2 Configuration of Hyperparameters

To address potential underfitting and overfitting challenges as discussed in Chapter 2, it is essential to fine-tune hyperparameters. In the context of this study, the hyperparameters subjected to adjustment include image size (img size), batch size, the number of training epochs, IOU threshold, the data and configuration files (cfg), learning rate, weight decay, and

the optimizer configuration file. Table 3.2 provides a summary of the specific hyperparameter configurations employed in this investigation.

Table 3.2: Tuned hyperparameters of the study.

Parameter	Configuration
Image size	640x640
Batch size	4
Epochs	Initial setting was @300 and final setting due to manual early stopping technique was @30
Learning rate	0.01
Weight decay	0.0005
Optimizer	Adam

Image Size (img size): Image Dimensions: In this context, we specify the size of the input images. YOLOv7's default image size is set at 640×640, but to speed up training, it can be made smaller. After extensive experimentation, it was determined that an input size of 640×640 works best as it maintains a substantial amount of detail while enhancing the training speed.

Batch Size: The batch size is determined, and it has a significant impact on the training process. Processing a thousand images simultaneously through the neural network significantly increases the number of model parameters learned in a single epoch. Therefore, the dataset is typically divided into multiple batches, each containing 'n' images, and training occurs batch by batch. The results from each batch are stored in the system's RAM and combined once training for all batches is completed. It's important to note that larger batch sizes consume more memory due to the storage of learned weights from each batch. In this specific case, with a training set consisting of 1240 images and a batch size of 4, there were a total of 310 batches.

Training Epochs: The number of training epochs is defined. An epoch involves learning from all input images, effectively training on the entire dataset. Since the dataset is divided into multiple batches, one epoch covers the training of all batches. The number of epochs indicates how many times the model is trained on all inputs and adjusts its weights towards the correct labels. The choice of the number of epochs often relies on experience and intuition. In this particular case, the initial setting was 300 epochs, but it was observed that the model's performance was nearly perfect. Therefore, a manual early stopping technique was applied.

Data: This refers to the file path leading to the data.yaml file, which contains a summary of dataset information. The model evaluation process occurs immediately after each epoch. As a result, the model accesses the validation directory by utilizing the path specified in the data.yaml file for evaluation at that point.

Configuration File (cfg): This specifies the file path for the model configuration. It allows the train.py file to compile and construct the architecture as defined in the model yaml file for training input images.

Learning Rate: This parameter has a significant impact on the training speed of the model. A higher learning rate leads to faster training, but it should fall within the range of 0 to 1. An excessively large learning rate can result in premature convergence to a suboptimal model, while a very small learning rate can cause the training process to stagnate. In this study, the learning rate was maintained at its default setting of 0.1, and no issues were encountered during training.

Weight Decay: Weight decay is employed to mitigate overfitting and ensure that the model's weights are relatively modest. This is especially important when the weights are intended for deployment on a memory-efficient processor, such as a Jetson Nano.

Optimizer: When selecting an optimisation algorithm, the choice often revolves around Adam and SGD optimizers.

The YOLOv7 algorithm was chosen as the foundational model for this project due to its real-time processing capabilities and high accuracy. The methodology involved defining two distinct classes: 'buckled' and 'unbuckled,' and making corresponding adjustments to the configuration and coco-yaml files to accommodate these classes. The class names in the coco-yaml file were modified to 'buckled' and 'unbuckled.' Additionally, the coco-yaml file was updated to establish links to directories containing the training, validation, and testing datasets.

3.5 NETWORK TRAINING

For a model to be trained, preparation and preprocessing of dataset must be done. According to Poonkuntran et al. [9], the process of data preprocessing encompasses the elimination of

HTML tags and URL tags, the renaming of the data, and the resizing of the images. Given that a segment of the data was obtained from online sources, it encompassed HTML tags and URLs that necessitated removal prior to model training. The training and performance evaluation platform for the network model was implemented on a personal computer with the following configuration: The CPU utilised in the system was an Intel (R) i5-10300H, which possessed the ability to operate at a peak frequency of 2.5 GHz. In addition, the system was equipped with an NVIDIA GeForce GTX 1650 GPU and 16 gigabytes of random-access memory (RAM). The system was equipped with the Windows 11 operating system, which utilised a 64-bit architecture. In the context of deep learning tasks, the system employs Pytorch version 1.21, leveraging CUDA version 12.2 to enhance computational performance through GPU acceleration. The programming language utilised for software development was Python 3.9.16. The initial phase of the training procedure involved duplicating the YOLOv7 library sourced from the GitHub repository [35]. The initial training of the study was done utilising pretrained weights of the from the YOLOv7 GitHub repository. The benefit of using pretrained weights is that the model gets to achieve high accuracy with less biasing [9].

3.6 NETWORK EVALUATION

During the training and testing stages, the effectiveness of the model was assessed by utilising various metrics, including precision, recall, confusion matrix, F1-score, recall, precision, losses (Box, Objectness, and Classification), and mAP. The previously mentioned measurements were employed as quantitative indicators to evaluate the effectiveness of the model. The utilisation of the confusion matrix is prevalent in the field, as it serves as a concise and informative tabular depiction of the predictions produced by a trained model. The classification framework consists of four fundamental components, namely TP, FP, TN, and FN.

3.7 DEPLOYMENT

The objective of the deployment was to optimise the execution of inference during the safety belt detection experiments conducted within the vehicle. In order to accomplish this goal, it was essential to convert the weights of the YOLOv7 model that had been trained into a format that is more lightweight, thereby enabling efficient processing on the Jetson Nano development board. The procedure encompassed three discrete stages, as depicted in Fig 3.12.

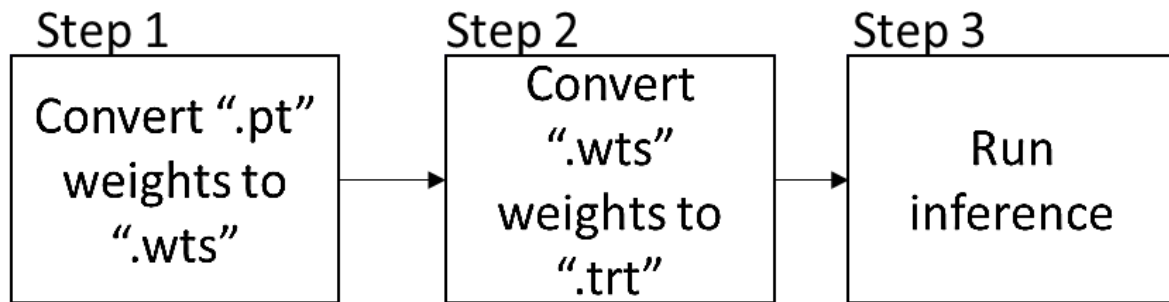


Fig 3.12: The steps used to deploy the YOLOv7 model into a Jetson Nano development board.

The YOLOv7 model underwent training on the PyTorch platform, as indicated in Stage 2 of the flow diagram. The weights corresponding to each training process were generated in the PyTorch format (.pt). Weights that have the ".pt" extension are generally not suitable for deployment on devices that have limited memory capacity, such as the Jetson Nano. In order to enhance compatibility with the Jetson Nano platform, we executed a conversion procedure to convert the weights from the ".pt" format to TensorRT weights in the ".trt" format. The purpose of this conversion was to attain reduced weights that could be efficiently handled by the Jetson Nano platform. To facilitate the conversion of weights to the Tensor RT format, it is imperative to initially transform the weights from the ".pt" format to the ".wts" format. Following this, there is a requirement to convert the ".wts" format into the ".trt" format.

3.8 SUMMARY OF THE CHAPTER

The chapter's focus encompassed several pivotal aspects, notably the data collection process, the selection of the object detection model (YOLOv7), the rationale behind its choice, an exploration of its functionality, architectural nuances, and historical context. Furthermore, attention was devoted to the annotation process, hyperparameter configuration, comprehensive insights into the training phase (including hardware and software selections), the evaluation protocol, and deployment procedures. Subsequently, the following chapter will shift its focus towards presenting and discussing the outcomes of the proposed study in greater depth.

CHAPTER 4: RESULTS AND DISCUSSION

This section presents the study's outcomes, engages in a discussion about them, and offers concluding remarks. The outcomes encompass an evaluation of the proposed model's performance, as well as the experimental results obtained when the model was put into use within an automobile.

4.1 PERFORMANCE EVALUATION RESULTS

This section presents a thorough examination of the performance metrics utilised in the model training process. The inclusion of illustrations and tables provides support for the presentation of these metrics. As a result, a sequence of experiments was undertaken to validate the diverse hypothetical theories, which were subsequently complemented by visual depictions.

Experimental results and analysis of the collected dataset:

The assessment of the detection performance of the proposed YOLOv7 object detection method was conducted by means of experimental evaluation using the collected dataset. The findings presented in Fig 4.1 and Fig 4.2 demonstrate a notable degree of effectiveness, particularly in the case of the 'bucked' category, which displays an AP of 99.7%. The group of individuals who did not fasten their seatbelts also demonstrated a notable degree of effectiveness, as indicated by a performance rate of 99.5%. The mAP of the model was found to be 99.6%. The graphical results for precision, recall, mAP, AP, and F1-score for both categories are depicted in Fig 4.1 and Fig 4.2.

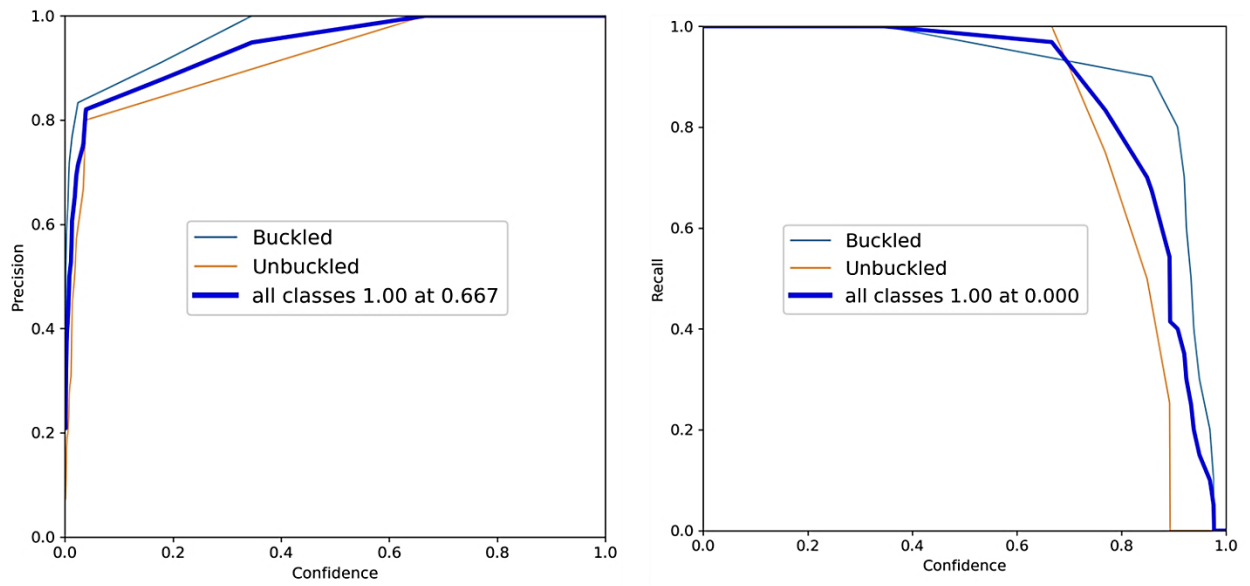


Fig 4.1: The precision and the recall graphs.

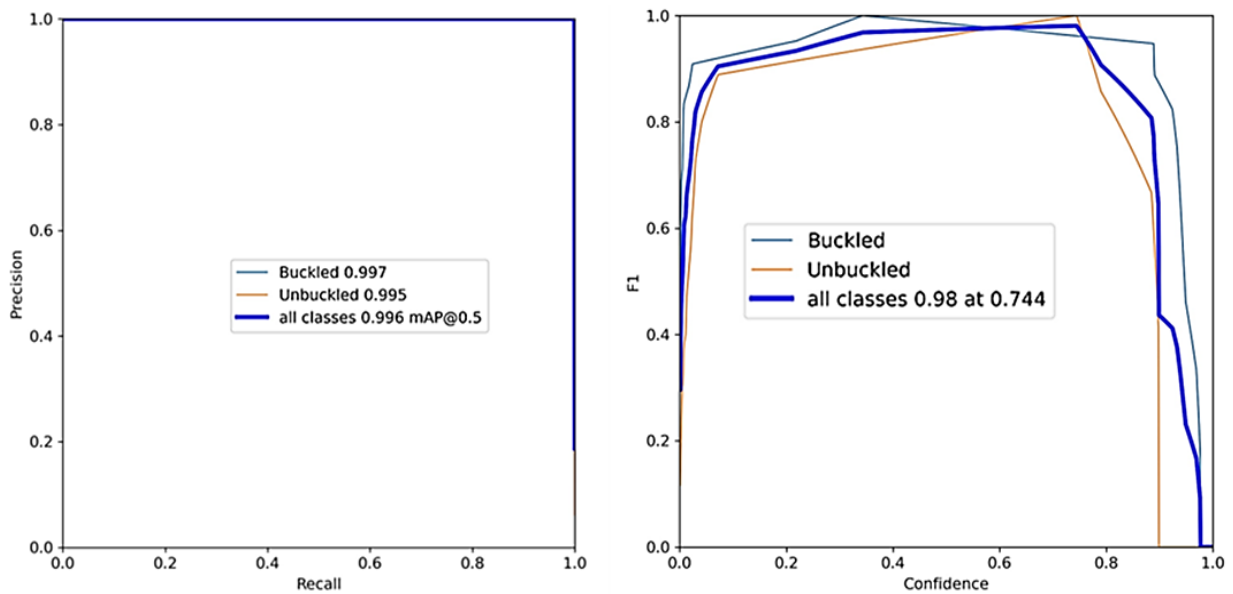


Fig 4.2: Precision-recall curve (PR) and F1-score curve.

The analysis of the outcomes of the proposed study was conducted using a confusion matrix. In the matrix, the representation of the predicted class is observed in each column, whereas the true proportions of each class in the dataset are depicted in each row, as illustrated in Fig 4.3. The examination of the results presented in Fig 4.3 demonstrates that the model demonstrates a high degree of precision in forecasting the "buckled" and "unbuckled" categories, achieving a perfect prediction rate of 100%.

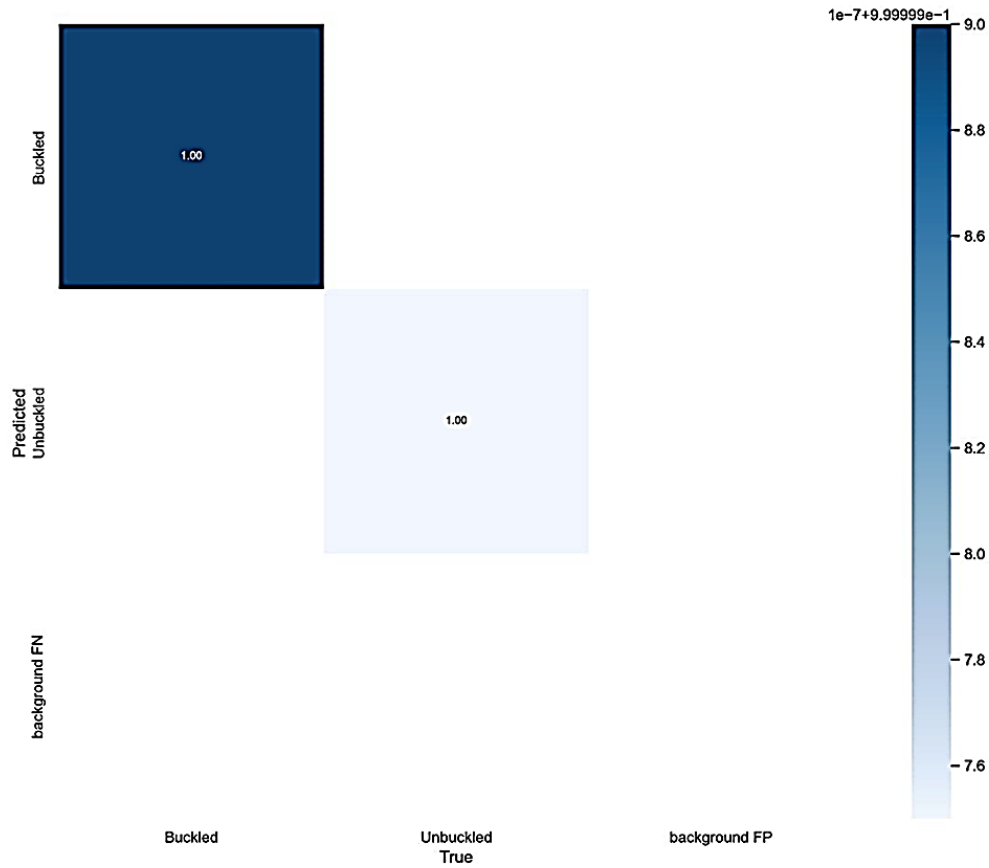


Fig 4.3: The confusion matrix of the study.

Furthermore, this research offers a graphical representation of the variations in the loss values, specifically focusing on the box loss, objectness loss, and classification loss. The YOLOv7 model adopts the Generalised Intersection Over Union (GIOU) loss as its selected loss function for bounding boxes. The calculation of the box loss involves taking the average of the GIOU loss function, as described in reference [56]. A decrease in the value of the loss function indicates an increase in the level of accuracy. The term "objectness loss" pertains to the average value of the loss function used for target detection. A decrease in this value signifies a greater degree of accuracy [56]. The term "classification loss" denotes the mean value of the classification loss metric. A lower value indicates a greater degree of accuracy, as depicted in Fig 4.4. As depicted in Fig 4.4, the loss values demonstrated a consistent downward trend, eventually converging to a steady state after 30 epochs. In order to mitigate the issue of overfitting, a manual early stopping technique was implemented at the 30th epoch in a random manner, resulting in satisfactory performance metrics for the model.

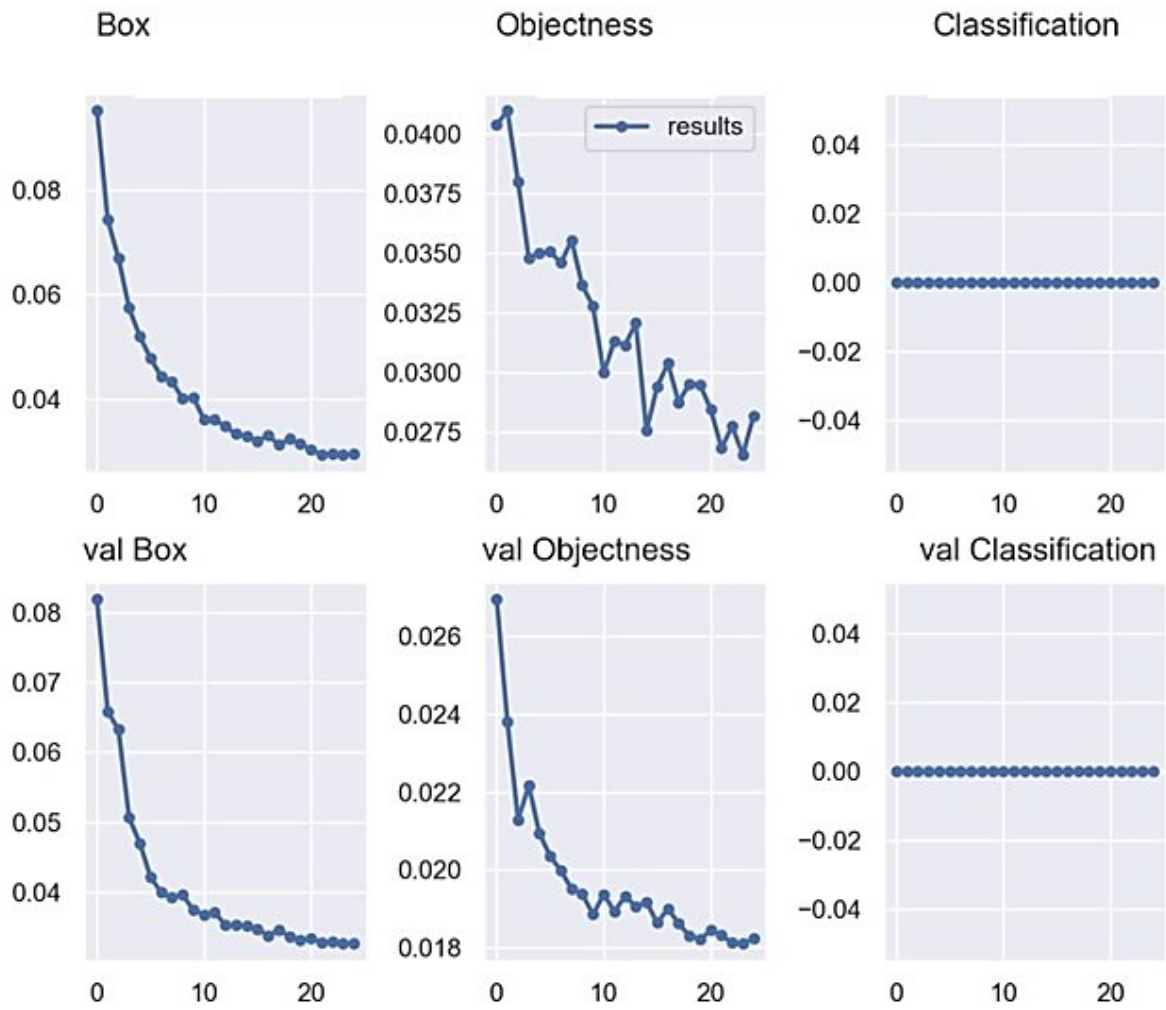


Fig 4.4: The losses of the proposed study

In order to provide additional evidence of the excellence of the proposed study, a comparative analysis was conducted on the outcomes of the relevant studies under review, as depicted in Table 4.1. The findings presented in Table 4.1 indicate that the performance of the proposed model output surpasses that of the object detection models reviewed. Nevertheless, according to the findings of Hossein and Fathi [45], they have reported achieving a classification accuracy of 99.7%, which is marginally higher by 0.1% compared to this study's mAP rate of 99.6%. Nevertheless, the proposed model showcases the tangible benefits of implementing in-cabin safety belt detection.

Table 4.1: Result comparison with related studies

Author(s)	Method	Results
Chen et al. [3]	CNN and SVM	Average Precision@92.1%
Guo et al. [18]	Edge detection and HSV	Accuracy@81%
Chun et al. [39]	CNN	Precision@63.58% Sensitivity@ 63.51% F1-score@63.55%
Hosameldeen [40]	CNN with Alex Net	True positive rate@92% True Negative rate@90% True rate@91%
Elihos et al. [44]	SSD, CNN, and FV algorithms	Accuracy@91.9% Precision@94.5%
Hossein and Fathi [45]	YOLOv5	Accuracy@99.7%
OURS	YOLOv7	mAP @99.6% F1-score@ 98%

4.2 ANALYSIS OF THE TRAINED YOLOV7 MODEL BASED ON EXPERIMENTS

This section consisted of a series of experimental tests conducted to validate various hypothetical scenarios. A thorough elucidation of the testing methodologies employed was provided for each of these experiments, followed by an in-depth examination of the test outcomes. The experimental setup for all the onboard experiments is depicted in Fig 4.5.

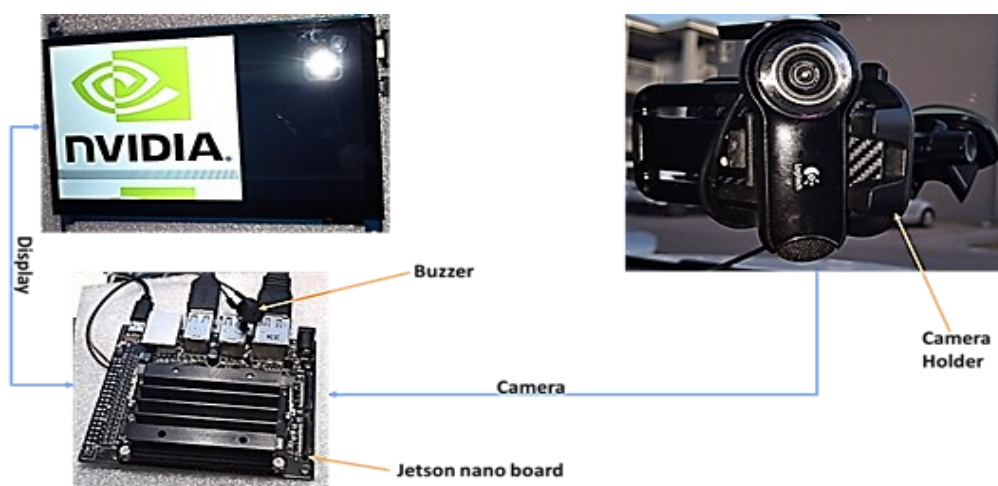


Fig 4.5: The setup of all the experiments conducted inside the vehicle.

The experimental configuration consisted of an Nvidia Jetson Nano board, a camera module, a display unit, a buzzer component, and a camera holder apparatus. The camera was employed as a means of acquiring data, while the Jetson Nano was utilised for processing the incoming frames in order to detect the status of a safety belt. The display was employed to ascertain the detection status and camera configuration. The Jetson Nano was interfaced with the buzzer, which would exclusively activate when the safety belt's status was identified as unbuckled. Finally, the camera holder was affixed to the windshield and employed to securely stabilise the camera, thereby mitigating the risk of accidental dislodgement during vehicular movement.

Experiment 1: Concept Testing

The installation of a camera on the windshield, along with necessary adjustments, is required in this study to ensure the visibility of the safety belt pillar loop. The purpose of the detection system is to classify the status of the safety belt as "unbuckled" when the pillar loop is not visible and conversely as "buckled" when the individual is securely fastened, and the pillar loop is visible. Moreover, in situations where an individual fastens the safety belt around their back, yet the pillar loop remains visible, it is imperative for the system to accurately recognise this condition as "unbuckled. As a result, an audible alarm is triggered whenever the system detects that the safety belt is not fastened. The decision to employ the pillar loop is justified by the potential occurrence of unanticipated situations in which individuals may be attired in garments featuring stripes that bear resemblance to a safety belt or when an individual is carrying a bag that possesses webbing that resembles a safety belt. In both scenarios, the model would erroneously identify the safety belt as fastened, even in cases where individuals may not be securely fastened if the pillar loop is not employed as the decisive criterion.

The efficacy of the pillar loop approach was confirmed through validation in four instances, as depicted in Fig 4.6.

1. When an individual inside a vehicle fastens their safety belt behind their back, resulting in the visibility of the safety belt's pillar loop.
2. When two occupants of a vehicle properly utilise their seat belts, resulting in the visibility of their pillar loops.
3. When two occupants of the vehicles were not wearing their seat belts and their seat belt anchor points were clearly visible.

4. The first scenario involves a detection status in which one vehicle occupant was properly buckled despite the presence of an invisible pillar. In contrast, the second scenario pertains to a detection status where the other occupant was also properly buckled, but with the aid of a visible pillar loop.



Fig 4.6: A, B, C, and D depict the safety belt detection status when the model was tested in occluded and visible pillar loop conditions.

Experiment 2: Occlusion

The present study successfully demonstrated the efficacy of the model in accurately detecting the status of safety belt buckling even when the pillar loop was partially obstructed. The camera underwent adjustments as outlined in Experiment 1. The experiment was carried out within the confines of a stationary vehicle. The experiment aimed to assess the model's capability to accurately identify the safety belt status under conditions of occlusion. Fig 4.7 illustrates multiple scenarios (A, B, and C) in which the model exhibited superior performance in detecting the status of safety belts under different occlusion conditions.

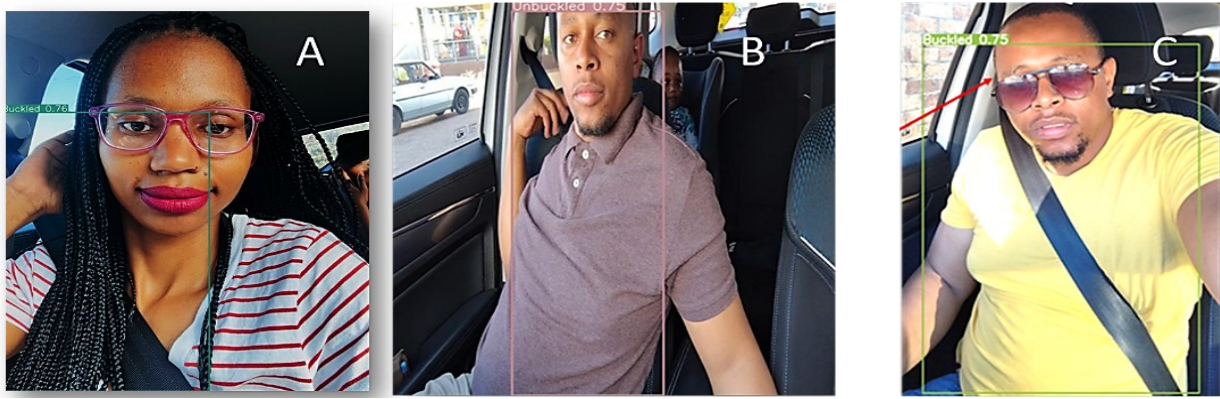


Fig 4.7: (A) shows the detection status of a safety belt when the hand and hair partially obscured the webbing, and the pillar loop was only 50% visible. B shows the safety belt detection status with the hand blocking the webbing while buckled. C shows safety belt detection with the pillar loop partially visible.

Experiment 3: Clothing

The objective of Experiment 3 was to determine the ability to detect safety belts when the occupants of the vehicle are wearing clothing that closely matches the colour of the safety belt webbing. The camera underwent the necessary adjustment procedures as described in Experiment 1. The research was carried out within the parameters of a moving vehicle. The primary aim of the experiment was to assess the effectiveness of a particular model in accurately identifying the presence or absence of a safety belt under predetermined conditions. Fig 4.8 depicts the empirical findings of the conducted experiment.

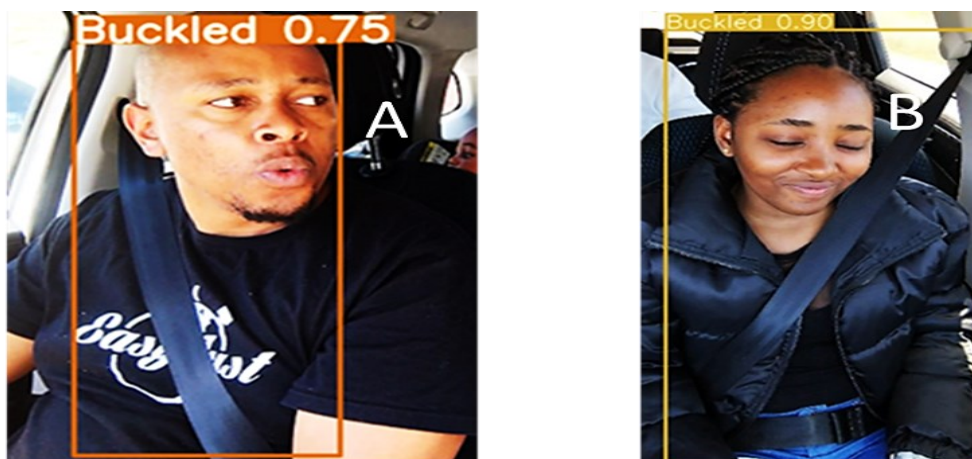


Fig 4.8: The safety belt detection status of A and B was represented when occupants were wearing clothing that was of a similar colour to the webbing.

Experiment 4: Lighting

The primary aim of this experiment was to evaluate the robustness of the proposed model in the face of different lighting conditions. The proposed model was tested in three distinct lighting conditions: clear lighting (A), dark lighting (B), and conditions characterised by high light intensity. To replicate various lighting scenarios, the vehicle was manoeuvred in shaded regions to emulate low lighting conditions, in areas exposed to direct sunlight to simulate high light intensity, and in areas with moderate sunlight to represent favourable lighting conditions for approaching vehicle occupants. Under conditions of high light intensity, no detections were observed. The lack of detection can be attributed to the distortion of safety belt features, which is caused by excessive exposure to sunlight on the camera. Fig 4.9 depicts the detection of safety belts under varying lighting conditions, namely clear lighting (A) and dark lighting (B).

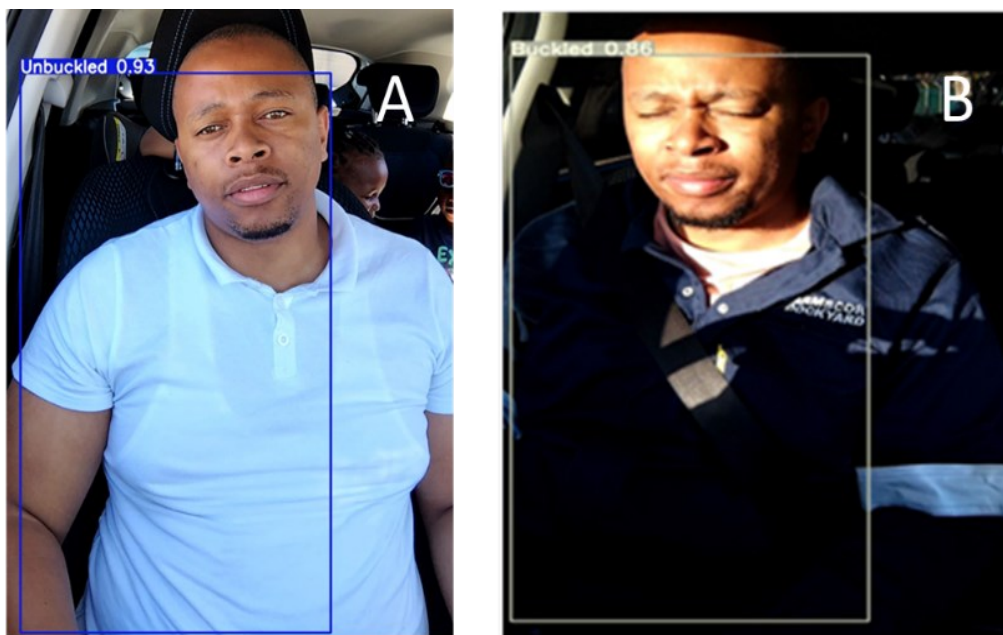


Fig 4.9: This illustrates the detection status of safety belts in both well-lit conditions (A) and low-light conditions (B).

Experiment 5: Blurry Scenes

The current investigation effectively showcased the capacity to identify the existence of a safety belt in images characterised by low visual resolution. This phenomenon was observed in instances where the camera underwent swift motion as a result of variables such as an uneven road surface or vibrations. The camera was securely attached following the procedures described in Experiment 1, and its stability was maintained by carefully adjusting the screws

of the holder. Subsequently, we undertook the task of traversing a speed bump at a velocity of 40 kilometres per hour, which led to an observation of visually distorted perception that is depicted in Fig 4.10.



Fig 4.10: The visual distortion observed in the scene is a result of the vehicular vibrations induced by traversing a speed hump at a velocity of 40 kilometres per hour.

4.3 EVALUATING STATISTICAL OBSERVATIONS AND RESILIENCE OF SAFETY BELT BUCKLE DETECTION IN VARIED EXPERIMENTAL SCENARIOS

The findings presented in this study are derived from a comprehensive set of 15 tests conducted for each of the five distinct experimental scenarios. The devised scenarios were intended to encompass a diverse array of circumstances that may impact the efficacy of safety belt buckle detection. The results of these tests are presented in Fig 4.11, which visually represents the accuracy and performance of the object detection model under various conditions. The adoption of this comprehensive methodology guarantees that the research outcomes are based on a strong and inclusive dataset, enabling us to derive significant observations regarding the efficacy of the model in practical situations.

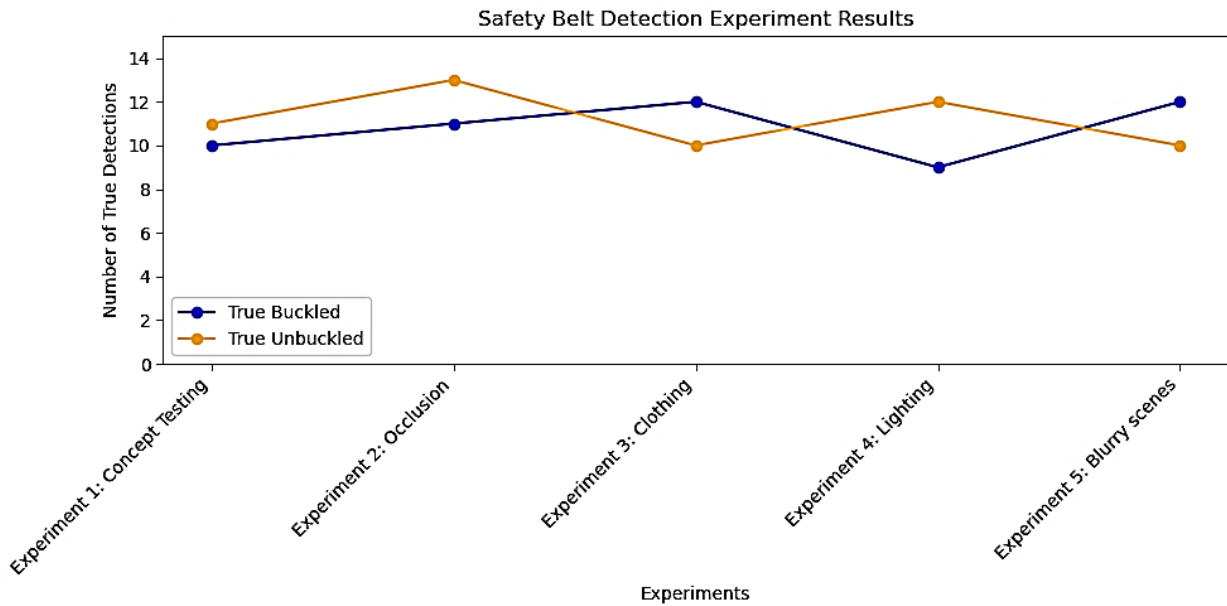


Fig 4.11: Comparative analysis of a safety belt buckle detection performance.

Fig 4.11 presents a line plot visualisation that offers a comprehensive depiction of the accuracy of the YOLO v7-based object detection model in identifying safety belt buckles during the experimental outcomes. The analysis emphasises the model's ability to accurately differentiate between buckled and unbuckled belts in various situations while also identifying areas for improvement. These observations provide valuable contributions to the progress of intelligent safety systems, fostering additional investigation and improvements in safety belt detection technology for safety-related purposes.

4.4 DISCUSSION

The main aim of this study was to evaluate the effectiveness of the YOLOv7 algorithm in detecting the presence of a safety belt. Based on the findings cited in reference [11], the failure to employ safety belts is responsible for a considerable number of fatalities, resulting in a daily death count of 3285 and an annual count of 1.35 million. The examination of safety belt fastening is a noteworthy field of study owing to the possibility of severe harm and loss of life that can arise from the action of unfastening [18, 42, 43]. The existing body of research has predominantly concentrated on conventional algorithms, which have been observed to display instability and inaccuracy when faced with dynamic environments [5]. Previous methodologies have entailed the deployment of surveillance cameras positioned along highways [18, 40, 44] in order to ascertain compliance with safety belt usage. The objective of this approach was to tackle the issue of a scarcity of traffic officers to carry out manual inspection tasks. However,

this methodology does not take into account the possibility of vehicle occupants conveniently securing their safety restraints in areas where the cameras have been deployed. Previous research [18, 45] has predominantly focused on examining the driver's compliance with safety belt usage while overlooking the safety of other occupants in the vehicle. Fig 4.6 depicts the effectiveness of the model in identifying multiple occupants in vehicles, thereby augmenting the safety of a greater number of individuals present within the vehicle.

The predominant focus of research in this domain has been devoted on seat belt fastening solutions using traditional algorithms, as indicated by the results of prior studies [3, 5, 18]. Zou et al. [17] have observed that traditional object detection algorithms encounter challenges in terms of their computational efficiency, precision, and ability to perform effectively in diverse environmental contexts. The model being discussed was derived from YOLOv7, which falls within the same category as other deep learning-based methodologies. The study conducted exhibited a significant degree of accuracy, as indicated by a mAP of 99.6%, an F1-score of 98%, and a confidence level of 100% for both buckled and unbuckled categories. The performance of the model was deemed satisfactory during evaluations that took place within the vehicle. These evaluations specifically focused on occluded pillar loop scenes, where the pillars obstructed the view, instances where occupants were wearing clothing that resembled the webbing of the safety belt, scenes with reduced clarity, and scenarios that involved various lighting conditions. Experiments 2 to 5 demonstrate the observed experimental results.

However, the study conducted exhibited certain limitations. The experiments conducted revealed that an excessive amount of sunlight causes the deformation of certain safety belt characteristics, thereby resulting in the failure to accurately detect the status of the safety belt. The accuracy of detecting safety belt status may be compromised in the proposed studies due to the reliance on the safety belt pillar loop as the main determinant, especially when using a single camera positioned at a fixed angle. As a result, the model has the potential to incorrectly classify individuals as not wearing seat belts, even in cases where they are correctly restrained. The training of the model did not encompass instances of occupants being incorrectly buckled, with the exception of individuals being buckled behind their backs. Consequently, this resulted in a discernible reduction in the model's confidence when determining the safety belt status of individuals who were improperly buckled. The evaluation of the model did not include nocturnal scenes due to the unavailability of a camera with night vision capabilities.

The field of computer vision has made notable progress in the detection of safety belts. Nonetheless, there are still various gaps and potential areas for further investigation that need to be acknowledged and explored. One of the main challenges lies in the ability of detection algorithms to maintain their effectiveness in various environmental conditions. While current models demonstrate proficiency in controlled environments, further research is necessary to accurately detect seat belts in diverse lighting conditions, camera perspectives, seat configurations, and occupant morphologies. The consideration of real-time performance is of utmost importance, especially in scenarios that involve dynamic elements. The need to achieve a balance between accuracy and real-time processing creates a demand for the creation of algorithms that are specifically designed to efficiently and precisely detect information in a timely manner. The central concern that arises is the quality and diversity of training data. To improve the generalisation of models, it is essential to have comprehensive datasets that cover a wide range of real-world scenarios. The complex challenge of identifying safety belts in situations involving multiple occupants or obstructions necessitates the development of creative and novel approaches. Future research endeavours should focus on developing strategies to address privacy concerns associated with the collection of passenger images while also ensuring accurate detection of seat belt usage. Moreover, it is crucial to give specialised attention to the adaptation of detection algorithms to non-traditional seating configurations, as observed in specific types of vehicles. The incorporation of safety belt detection into user interfaces, such as infotainment systems or driver assistance displays, is being recognised as a feasible approach to improving passenger awareness and adherence. The interpretability of deep learning poses challenges, especially in safety-critical applications, despite its effectiveness. Therefore, investigating methods to enhance the transparency and interpretability of deep learning models is a relevant area of research. In addition, it is of utmost importance to optimise detection algorithms for hardware with limited resources and edge computing installations in order to facilitate extensive implementation in vehicular contexts. Continuous monitoring, calibration, and maintenance strategies are imperative for ensuring the durability and precision of system performance. In conclusion, the synchronisation of safety belt detection technology with established industry standards and regulatory frameworks highlights the significance of cooperative endeavours among researchers, industry participants, and regulatory entities to establish standards and guidelines for the successful deployment of this technology. As the field of computer vision progresses, continuous research efforts are positioned to address these gaps and enhance the effectiveness of safety belt detection systems.

4.5 CONCLUSION

Current research examines the process of determining the safety belt status of a vehicle by utilising the YOLOv7 algorithm. The construction of the model involved the utilisation of a dataset comprising 1240 images. These images were obtained from various sources, including Cape Town, as well as through the implementation of crowdsourcing and random online image acquisition methods. The dataset was partitioned into training, testing, and validation sets using a Python script to automate the process. The partitioning was done with a ratio of 80:10:10. Throughout the training phase of the model, the outcomes of the current investigation demonstrated a superior degree of efficacy when compared to other pertinent studies that were examined. The present study achieved a mAP of 99.6% when the threshold was set at 0.50. Furthermore, the model demonstrated an F1-score of 98%, along with precision rates of 99.7% and 99.5% for the buckled and unbuckled classes, respectively. After a thorough evaluation, the model was successfully deployed on the Jetson Nano development board to facilitate onboard experiments. The conducted experiments demonstrated that the model functioned in alignment with the proposed design concept. This concept involved the utilisation of a pillar loop as the primary determinant for differentiating between the buckled and unbuckled states of a safety belt. Further observations were conducted pertaining to the model's capacity to identify diverse and complex scenarios. The identified scenarios encompassed the identification of scenes with reduced clarity, environments that were partially obstructed, situations where occupants were seated with their seatbelts secured behind them, and scenes characterised by inadequate lighting conditions. However, it was noted that the model demonstrated limitations in its ability to detect instances of missing safety belt features in situations characterised by high levels of light intensity.

In contrast to the existing safety belt monitoring system, which is susceptible to manipulation through techniques such as securing the safety belt behind the back or using a safety belt stopper to simulate the latch plate of a safety belt, the proposed model utilises a camera as the sensing mechanism. The presented model demonstrates a high level of proficiency in accurately discerning between the conditions of being buckled and unbuckled, even in scenarios where individuals deliberately secure their seat belts behind their backs. The study being discussed demonstrated a higher degree of accuracy when compared to the latest studies presented in the current research literature. However, there is still potential for additional improvement. To help the field move forward, future research should focus on improving how well models can find safety belts in situations with a lot of light. One potential strategy for

addressing the difficulties associated with lighting conditions is to investigate the latest versions of YOLO algorithms. These algorithms have exhibited higher levels of accuracy and robustness compared to other object detection algorithms used in similar research projects. The potential exists for integrating these algorithms with advanced camera technologies that demonstrate the ability to dynamically adjust and adapt to varying lighting conditions.

CHAPTER 5: SUMMARY AND CONCLUSION

In general, this chapter provides the framework of the entire dissertation, tying up the key results to reflect on the overall contribution of the YOLOv7 algorithm to the detection of a car seat belt buckling status. This chapter evaluates the research objectives and summarises the key research contributions made in this work. It further presents future research, recommendations, and concluding remarks.

5.1 EVALUATING OBJECTIVES AND KEY RESEARCH CONTRIBUTIONS

The primary objective of this study was to introduce and apply the YOLOv7 methodology for the purpose of detecting and predicting the status of car seat belt buckling. The evaluation of the research findings is conducted with a discerning analysis of each predefined research objective. Moreover, an in-depth review of each research objective serves to accentuate the distinctive domains in which this study has introduced innovative contributions.

Objective 1: To train and test the YOLOv7 model using the GPU.

The study yielded significant insights into the establishment of a YOLOv7 model designed to identify the fastening status of a car seat belt. The utilisation of a GPU notably expedited the training and testing process of the model.

Contribution(s):

1. Pioneered a structured approach for training the YOLOv7 model, leading to the creation of a discriminative model capable of distinguishing between buckled and unbuckled states of a car seat belt.

Objective 2: Deploy the training weights from a GPU platform to a Jetson Nano board for inferencing.

The study demonstrated an approach that used a GPU as an environment to train and a model to predict the buckling of a car seat belt. The trained model was subsequently deployed to a Jetson Nano environment for the practical implementation of the study. The buzzer was integrated into the model to remind the occupants to buckle their safety belts.

Contribution(s):

1. Conversion of weights to lighter weights that can be handled by the Jetson Nano board using NVIDIA STREAM rules and installation of the necessary software packages.

2. Incorporating the buzzer into the GPIO pins of the Jetson Nano by manipulating the "detect.py" code of YOLOv7.

Objective 3: Test the deployed model using the NVIDIA DEEP STREAM object detection applications.

The study presented a novel approach to utilising YOLOv7 to predict the seat belt detection status of a car. The YOLOv7 model was trained on real-world images that contained various seat belt scenarios, including illuminance variation, occlusion, apparel with a similar colour as the car seat belt, and blurry images. The purpose of this training was to enable the model to make predictions about the status of the car seat belt. The model was tested on separate testing data sets and deployed to the Jetson Nano development board.

Contribution(s):

1. Experiments were conducted to validate the workings of a model in the Jetson Nano environment.
2. The introduction of a buzzer was used to irritate the car occupants into buckling their seat belts.

5.2 RECOMMENDATIONS AND SUGGESTIONS FOR FUTURE RESEARCH

Based on the implications derived from the findings presented in this dissertation, it becomes apparent that future investigations should be conducted in accordance with the subsequent recommendations.

1. The outcomes of this study are limited solely to the YOLOv7 algorithm. Consequently, alternative deep learning algorithms, such as YOLOv8, YOLO-NAS, SSD, and YOLOv7 segmentation, have not been subject to comparative evaluation. Therefore, it would be of considerable interest for automotive engineers and researchers to explore alternative avenues of detection to comprehensively assess disparities and potential enhancements in performance.
2. The dataset employed for training the model in this study was collected exclusively during the morning and afternoon periods of the summer season. It is worth considering the augmentation of model training data by incorporating diverse datasets encompassing varying times of the day across different seasons, including morning,

mid-day, and evening conditions. This augmentation strategy is poised to expand the dataset's diversity, thereby fostering an elevation in model accuracy.

3. The current study relied on a single camera that had no night vision capabilities to discern the seat belt's status. To ameliorate occlusion-related challenges, the implementation of multiple IR cameras positioned at distinct angles holds promise. Such a multi-camera approach is anticipated to mitigate occlusion concerns and enhance the overall efficacy of seat belt status detection.

5.3 CONCLUDING REMARKS

This dissertation established YOLOv7 as a promising tool for effectively detecting car seat belt buckling. The results indicated high accuracy even under challenging conditions, such as variations in light, partial occlusion, blurry scenes, and when the occupants wore clothing similar to the seat belt material. Additionally, the study highlighted the considerable time-saving benefits of employing GPUs for model training. Furthermore, the successful deployment of the model on a resource-efficient platform, such as the Jetson Nano development board, paved the way for its potential integration into modern car systems.

REFERENCES

- [1] J. Leland, E. Stanfill, J. Cherian, and T. Hammond, 'Recognizing Seatbelt-Fastening Behavior with Wearable Technology and Machine Learning', in Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems, Yokohama Japan: ACM, 2021, pp. 1–6.
- [2] Arrive Alive (2014, July), 'South African Law on Wearing of Seatbelts,' Arrive Alive. [Online]. Available: <https://www.arrivealive.mobi/south-african-law-on-wearing-of-seatbelts>. [Accessed: February 20, 2022].
- [3] Y. Chen, G. Tao, H. Ren, X. Lin, and L. Zhang, 'Accurate seat belt detection in road surveillance images based on CNN and SVM', *Neurocomputing*, vol. 274, pp. 80–87, Jan. 2018.
- [4] D. J. Van Hoving, C. Hendrikse, R. J. Gerber, M. Sinclair, and L. A. Wallis, 'Injury severity in relation to seatbelt use in Cape Town, South Africa: A pilot study', *South African Medical Journal*, vol. 104, no. 7, pp. 488–492, 2014.
- [5] A. Kashevnik, A. Ali, I. Lashkov, and N. Shilov, 'Seat Belt Fastness Detection Based Image Analysis from Vehicle In-cabin Camera', in 2020 26th Conference of Open Innovations Association (FRUCT), Yaroslavl, Russia: IEEE, 2020, pp. 143–150.
- [6] World Health Organization (2017), 'Save lives: a road safety technical package, 2017.' [Online]. Available: <https://www.who.int/publications/i/item/save-lives-a-road-safety-technical-package>. [Accessed: May, 18, 2022]
- [7] J. E. Solem, 'Programming Computer Vision with Python: Tools and algorithms for analyzing images', O'Reilly Media Inc., 2012.
- [8] A. Vahab, M. S. Naik, P. G. Raikar, and S. Prasad, 'Applications of object detection system', *International Research Journal of Engineering and Technology (IRJET)*, vol. 6, no. 4, pp. 4186–4192, 2019.
- [9] S. Poonkuntran, B. Balusamy, and R. Kumar Dhanraj, 'Object Detection with Deep Learning Models: Principles and Applications', 1st ed. Boca Raton: Chapman and Hall/CRC, 2022.
- [10] C. Yan, 'Vision-based driver behaviour analysis', Ph.D. Dissertation, Dept of Elec Eng, University of Liverpool, United Kingdom, 2016.

- [11] P. Kohli and A. Chadha, 'Enabling Pedestrian Safety Using Computer Vision Techniques: A Case Study of the 2018 Uber Inc. Self-driving Car Crash', in *Advances in Information and Communication*, K. Arai and R. Bhatia, Eds., Cham: Springer International Publishing, 2020, pp. 261–279.
- [12] T. Denton, 'Automated Driving and Driver Assistance Systems', 1st ed. Routledge, 2019.
- [13] L. Nkuzo, M. Sibiya, and E. Markus, 'Computer Vision-based Applications in Modern Cars for safety purposes: A Systematic Literature Review', in 2023 Conference on Information Communications Technology and Society (ICTAS), Durban, South Africa: IEEE, 2023, pp. 1–9.
- [14] M. Heimberger, J. Horgan, C. Hughes, J. McDonald, and S. Yogamani, 'Computer vision in automated parking systems: Design, implementation and challenges', *Image and Vision Computing*, vol. 68, pp. 88–101, Dec. 2017.
- [15] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, 'Sensor and Sensor Fusion Technology in Autonomous Vehicles: A Review', *Sensors*, vol. 21, no. 6, p. 2140, Mar. 2021.
- [16] N. Agarwal, C.-W. Chiang, and A. Sharma, 'A study on computer vision techniques for self-driving cars', presented at the Frontier Computing: Theory, Technologies and Applications (FC 2018) 7, Springer, 2019, pp. 629–634.
- [17] Z. Zou, K. Chen, Z. Shi, Y. Guo and J. Ye, 'Object Detection in 20 Years: A Survey', in *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257-276, Mar. 2023, doi: 10.1109/JPROC.2023.3238524
- [18] H. Guo, H. Lin, S. Zhang, and S. Li, 'Image-based seat belt detection', in *Proceedings of 2011 IEEE International Conference on Vehicular Electronics and Safety*, Beijing, China: IEEE, 2011, pp. 161–164.
- [19] A. Lee, 'Comparing deep neural networks and traditional vision algorithms in mobile robotics', Swarthmore University, 2015.
- [20] M. Sibiya, 'Classification and Severity Prediction of Maize Leaf Diseases using Deep Learning CNN Approaches', Ph.D. Dissertation, Dept of Elec Eng, University of South Africa: Pretoria, South Africa, 2021.

- [21] Y. LeCun, 'LeNet-5, convolutional neural networks', Y. LeCun, 'LeNet-5, convolutional neural networks', *URL: <http://yann.lecun.com/exdb/lenet>*, vol. 20, no. 5, p. 14, Sep. 2015.
- [22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, 'Imagenet classification with deep convolutional neural networks', *Communications of the ACM*, vol. 60, issue 6, pp. 84-90, May 2017, doi: <https://doi.org/10.1145/3065386>.
- [23] M. D. Zeiler and R. Fergus, 'Stochastic pooling for regularization of deep convolutional neural networks', *arXiv preprint arXiv:1301.3557*, Jan 2013.
- [24] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, 'Going deeper with convolutions', In Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 1-9.
- [25] K. Simonyan and A. Zisserman, 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*, Sept. 2014.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, 'Deep residual learning for image recognition', presented at the Proceedings of the IEEE conference on computer vision and pattern recognition, 2016, pp. 770–778.
- [27] R. Girshick, J. Donahue, T. Darrell, and J. Malik, 'Region-based convolutional networks for accurate object detection and segmentation', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.
- [28] S. Ren, K. He, R. Girshick, X. Zhang and J. Sun, 'Object Detection Networks on Convolutional Feature Maps,' in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 7, pp. 1476-1481, Jul. 2017, doi:10.1109/TPAMI.2016.2601099
- [29] K. He, G. Gkioxari, P. Dollár, and R. Girshick, 'Mask R-CNN'. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2961-2969
- [30] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, 'You Only Look Once: Unified, Real-Time Object Detection', in 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA: IEEE, 2016, pp. 779–788.

- [31] J. Redmon and A. Farhadi, ‘YOLO9000: Better, Faster, Stronger’, Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 7263-7271
- [32] J. Redmon and A. Farhadi, ‘Yolov3: An incremental improvement’, *arXiv preprint arXiv:1804.02767*, Apr. 2018.
- [33] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ‘Yolov4: Optimal speed and accuracy of object detection’, *arXiv preprint arXiv:2004.10934*, Apr. 2020
- [34] C. Li, L. Li, H. Jiang, K. Weng, Y. Geng, L. Li, Z. Ke, Q. Li, M. Cheng, W. Nie, and Y. Li, ‘YOLOv6: A single-stage object detection framework for industrial applications’, *arXiv preprint arXiv:2209.02976*. Sep. 2022
- [35] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, ‘YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors’, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 7464-7475
- [36] R. Padilla, S. L. Netto, and E. A. B. Da Silva, ‘A Survey on Performance Metrics for Object-Detection Algorithms’, in 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), Niterói, Brazil: IEEE, 2020, pp. 237–242.
- [37] H. Gupta, A. Pundir, and O. Sharma, ‘RF Module Based-Speed Check and Seatbelt Detection System’, presented at the 2016 Second International Conference on Computational Intelligence & Communication Technology (CICT), 2016, pp. 504–509.
- [38] E. N. Mazzae, G. Baldwin, and A. T. Andrella, ‘Performance Assessment of Prototype Seat Belt Misuse Detection System’, Depart. of Transportation, National Highway Traffic Safety, United States, 2018.
- [39] S. Chun, N. Ghalehjegh, J. Choi, C. Schwarz, J. Gaspar, D. McGehee, and S. Baek, ‘A nimble architecture for driver and seat belt detection via convolutional neural networks’, In The IEEE International Conference on Computer Vision (ICCV), 2019, pp. 0-0

- [40] O. Hosameldeen, 'Deep learning-based car seatbelt classifier resilient to weather conditions', *IJET*, vol. 9, no. 1, pp. 229, Feb. 2020.
- [41] J. Yongquan, W. Tianshu, L. Jin, Z. Zhijia, and G. Chao, 'GPU acceleration design method for driver's seatbelt detection', in 2019 14th IEEE International Conference on Electronic Measurement & Instruments (ICEMI), Changsha, China: IEEE, 2019, pp. 949–953.
- [42] D. B. Naik, G. S. Lakshmi, V. R. Sajja, D. Venkatesulu, and J. N. Rao, 'Driver's seat belt detection using CNN', *Turkish Journal of Computer and Mathematics Education (TURCOMAT)*, vol. 12, no. 5, pp. 776–785, Apr. 2021.
- [43] F. Hu, 'Robust seatbelt detection and usage recognition for driver monitoring systems', *arXiv preprint arXiv:2203.00810*, Mar. 2022
- [44] A. Elihos, B. Alkan, B. Balci, and Y. Artan, 'Comparison of Image Classification and Object Detection for Passenger Seat Belt Violation Detection Using NIR & RGB Surveillance Camera Images', in 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Auckland, New Zealand: IEEE, 2018, pp. 1–6.
- [45] S. Hosseini and A. Fathi, 'Automatic detection of vehicle occupancy and driver's seat belt status using deep learning', *SIViP*, vol. 17, no. 2, pp. 491–499, Mar. 2023.
- [46] E. Daoud, N. Khalil, and M. Gaedke, 'Implementation of a One Stage Object Detection Solution to Detect Counterfeit Products Marked with a Quality Mark', *IADIS International Journal on Computer Science & Information Systems*, vol. 17, no. 1, Jan. 2022.
- [47] W. Tianshu, Z. Zhijia, L. Zhanning, L. Yunpeng, and W. Shixian, 'Detection and Implementation of Driver's Seatbelt Based on FPGA', *J. Phys.: Conf. Ser.*, vol. 1229, no. 1, pp. 012075, May 2019.
- [48] J. Jaworek-Korjakowska, A. Kostuch, and P. Skruch, 'SafeSO: Interpretable and Explainable Deep Learning Approach for Seat Occupancy Classification in Vehicle Interior', in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Nashville, TN, USA: IEEE, 2021, pp. 103–112.
- [49] J. Luo, J. Lu, and G. Yue, 'Seatbelt detection in road surveillance images based on improved dense residual network with two-level attention mechanism', *J. Electron. Imag.*, vol. 30, no. 03, Jun. 2021.

- [50] S. Jha, I. Brooks, and S. J. Ray, ‘Seatbelt Segmentation Using Synthetic Images’, 2023 IEEE Intelligent Vehicles Symposium (IV), 2023, pp. 1-6.
- [51] R. A. Kapdi, P. Khanpara, R. Modi, and M. Gupta, ‘Image-based Seat Belt Fastness Detection using Deep Learning’, *SCPE*, vol. 23, no. 4, pp. 441–455, Dec. 2022.
- [52] Y. Lai, R. Ma, Y. Chen, T. Wan, R. Jiao, and H. He, ‘A Pineapple Target Detection Method in a Field Environment Based on Improved YOLOv7’, *Applied Sciences*, vol. 13, no. 4, p. 2691, Feb. 2023.
- [53] J. Ajala, ‘Object Detection and Recognition Using YOLO: Detect and Recognize URL(s) in an Image Scene’, Msc Thesis, Dept. Computer Science and Information Technology, St. Cloud State University, USA, MN, 2021.
- [54] C. -Y. Wang, A. Bochkovskiy and H. Y. M. Liao, ‘Scaled-YOLOv4: Scaling Cross Stage Partial Network,’ 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 2021, pp. 13024-13033.
- [55] T. Jiang and J. Cheng, ‘Target Recognition Based on CNN with LeakyReLU and PReLU Activation Functions,’ 2019 International Conference on Sensing, Diagnostics, Prognostics, and Control (SDPC), Beijing, China, 2019, pp. 718-722.
- [56] X. Ding, X. Zhang, N. Ma, J. Han, G. Ding, and J. Sun, ‘Repvgg: Making vgg-style convnets great again’, presented at the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 13733–13742.