Traffic Detection for Advanced Driver Assistance System

Author

Hamza Nadeem

Regn. Number

319884

Supervisor

Dr. Kashif Javed

MS ROBOTICS & INTELLIGENT MACHINE ENGINEERING

DEPARTMENT OF ROBOTICS & AI

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

AUGUST 2022

Traffic Detection for Advanced Driver Assistance System

Author

HAMZA NADEEM

Regn Number

319884

A thesis submitted in partial fulfillment of the requirements for the degree of

MS ROBOTICS & INTELLIGENT MACHINE ENGINEERING

Thesis Supervisor:

DR. KASHIF JAVED

Thesis Supervisor's Signature: _____

MS ROBOTICS AND INTELLIGENT MACHINE ENGINEERING

DEPARTMENT OF ROBOTICS & AI

SCHOOL OF MECHANICAL & MANUFACTURING ENGINEERING

NATIONAL UNIVERSITY OF SCIENCES AND TECHNOLOGY

ISLAMABAD

AUGUST 2022

Thesis Acceptance Certificate

It is certified that the final copy of the MS Thesis written by Hamza Nadeem (Registration No. 319884), of the Department of Robotics & AI (SMME) has been vetted by the undersigned, found complete in all respects as per NUST statutes / regulations, is free from plagiarism, errors and mistakes and is accepted as partial fulfilment for the award of MS Degree. It is further certified that necessary amendments as pointed out by GEC members of the scholar have also been incorporated in this dissertation.

Signature: _____

Date: _____

Dr. Kashif Javed (Supervisor)

Signature HOD: _____

Date: _____

Signature Principal: _____

Date: _____

FORM TH-4

National University of Sciences & Technology

MASTER THESIS WORK

We hereby recommend that the dissertation prepared under our supervision by **Hamza Nadeem** having **Regn. No. 319884** titled **"Traffic Detection for Advanced Driver Assistance System"**, be accepted in partial fulfillment of the requirements for the award of MS Robotics & Intelligent Machine Engineering degree.

Examination Committee Members

1. Dr. Hassan Sajid Signature:_____

2. Dr. Karam Dad Signature:_____

3. Dr. M. Usman Bhutta Signature:_____

Supervisor: Dr. Kashif Javed Signature:_____

Co-Supervisor: Dr. M. Jawad Khan Signature:_____

_____ _____

Head of Department Date

COUNTERSINGED

_____

Date:_____ Dean/Principal

Declaration

I certify that this research work titled "*Traffic Detection for Advanced Driver Assistance System"* is my own work. The work has not been presented elsewhere for assessment. The material that has been used from other sources has been properly acknowledged / referred to.

Signature of Student

Hamza Nadeem

319884

Plagiarism Certificate (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached.

Signature of Student

Hamza Nadeem

319884

Signature of Supervisor

Dr. Kashif Javed

Copyright Statement

Acknowledgements

I am truly thankful to Allah the Almighty for all the blessings throughout my life, especially in my educational career. Allah has always guided me towards the path which was best for me in all aspects. Pursuing my master's degree and completing the course work, and research work would not be possible without the guidance of Almighty Allah.

I would love to express my heartily thanks to thesis supervisor Dr. Kashif Javed and co-supervisor Dr. M. Jawad Khan for trusting in me, by providing me with the opportunity to work with them. Dr. Jawad was not only my co-supervisor, but he also taught me 04 subjects during my Masters' coursework which I am proficient in, thanks to him. His guidance, support, and availability at every part of my research phase made me complete this work. I would like to thank him for guiding me in all the events other than my research work which has helped me in grooming my technical skills and being motivated. His efforts and dedication throughout my research work were the key elements for this accomplishment.

I would like to acknowledge my committee members Dr. Hasan Sajid, Dr. Karam Dad, and Dr. M. Usman Bhutta for their support and encouragement throughout my course work and research work.

I am very thankful to my parents for supporting and loving me throughout my life and always motivating me by providing a practical example of hard work, honesty, dedication and commitment to every phase of life.

Dedication

Dedicated to my exceptional parents and adored siblings whose tremendous support and cooperation led me to this wonderful accomplishment.

Abstract

The Advanced Driver Assistance System (ADAS) is not a new phenomenon. To minimize road accidents and other related issues, the current vehicles can be improved for a better driving experience through an automated system that assists the driver. Some of the basic elements that such ADAS systems utilize include, but are not limited to, sensing the environment, traffic signs, pedestrians, and other vehicles. The need for traffic to be detected and recognized up to a certain degree of accuracy arises due to our objective i.e., to ensure that the car and the passengers in it are safe. Traditional Image Processing techniques have previously been used which are way slower. Recently, CNN**s** have been deployed heavily in Traffic detection and identification. However, CNNs do require a huge number of input images to work efficiently, and no such traffic recognition dataset exists in Pakistan. In this research, we deployed a YOLOv7 based architecture trained on a self-collected and manually annotated Pakistani Traffic Type and Sign Recognition Dataset (PTSD) to detect and classify the types of traffic. The Deep Learning model was trained and tested to produce a mean average precision (mAP) of 87.20%. These results are state-of-the-art and strong enough for implementation as real-world models. The model was further tuned to help improve the model's working, and then tested in real-world scenarios. The final model was used to develop an ADAS Unit—which works on a priority-based decision system, providing specified instructions for the detected conditions.

Key Words: advanced driver assistance system, traffic type recognition, deep learning, object detection, YOLOv7

Table of Contents

List of Figures

List of Tables

List of Acronyms

1.	CNN	Convolutional Neural Network
2.	ReLU	Rectified Linear Unit
3.	FC Layer	Fully Connected Layer
4.	ANN	Artificial Neural Network
5.	ML	Machine Learning
6.	DL	Deep Learning
7.	AI	Artificial Intelligence
8.	CV	Computer Vision
9.	ADAS	Advanced Driver Assistance System
10.	PTSD	Pakistani Traffic Type and Sign Recognition Dataset
11.	YOLO	You Only Look Once
12.	CVAT	Computer Vision Annotation Tool

CHAPTER 1: INTRODUCTION

The world has moved towards Industry 4.0—Artificial Intelligence—while Pakistan is still playing catch-up with the rest of the world. According to an article published by The News International on November 21st, 2021, the past decade has seen 104,105 road accidents, which have caused 55,141 deaths and left 126,144 injured. A total of 120,501 vehicles were involved in these accidents causing huge material loss as well as the loss of human life. Different causes of these accidents include not abiding by traffic rules, over-speeding, driver negligence, and blind spots. Therefore, measures must be taken to minimize road accidents. To accomplish this, there is a need to make the cars smarter and the driver more able to use the information to make better decisions on the road [1].

Initially, these traffic types were detected using conventional image processing systems which were both slower and less accurate. These systems worked based on visual features such as colors, and shapes with algorithms such as Color Segmentation used widely [2], [3]. Oher notable algorithms include Scale Invariant Feature Transform, Speeded-up Robust Features, and Binary Robust Invariant Scalable Keypoints among others [4], [5], [6].

More recently learning-based algorithms have replaced them and have successfully been implemented on traffic type and traffic-sign recognition problems such as the use of CNNs of German Dataset – GTSRB [7]. CNNs require a huge no. of images to work efficiently, and there is an absence of any maintained dataset containing traffic-sign images from Pakistan. Furthermore, Pakistani traffic signs differ from other signs around the world hence an indigenous dataset is required. There is a need to gather a diverse set of images from across Pakistan, in different lighting conditions and using various cameras and imaging modes. Labeling the acquired data to accurately detect and classify traffic-sign images will turn it into an excellent benchmark for future research.

1.1 Problem Statement

Self-driving cars are the next step in the evolution of the automobile industry. Although they were meant to be a sign of luxury, they carry a lot more benefits. These range from environmental impacts to better traffic system which in turn brings a net positive change in the society as a whole. These self-driving cars require a certain number of elements to work properly including, but not limited to, traffic signs, nearby vehicles, and pedestrians i.e., traffic types. The detection of traffic types is especially difficult in countries like Pakistan where

standard datasets are not available.

1.2 Proposed Solution

To cater for arrival of self-driving cars in Pakistan a computer vision system needs to be made which can detect and identify types of traffic among other things. Our system is a deep learning-based model and needs thousands of images to train properly. Firstly, the data was collected in form of videos, using different cameras and in different lighting conditions. The video keyframes were then extracted from the collected videos and subsequently annotated against the set classes. These images (keyframes) were compiled into a single dataset—Pakistani Traffic Type & Sign Recognition Dataset (PTSD). Secondly, the aforementioned self-collected dataset was used for training a Deep Learning based model for the identification of types of traffic. The model was cross-validated and regularized to help improve the model's working, and then tested in real-world scenarios and tweaked according to requirements. Finally, the final model was used to develop an Advanced Driver Assistance System—ADAS—which works on a priority-based decision system, dependent on the different traffic types on the road, providing specified instructions for the detected conditions at that time (traffic in front, overtaking vehicle, etc.).

1.3 Expected Outcome

The aim of this project was multifold and pertinent to real-world problems faced on the roads in Pakistan. These include, but are not limited to, real-time monitoring of traffic around the drivers' vehicle using an ADAS, which can later be evolved and geared towards Self-driving Cars and Smart City initiatives. This research was intended to, firstly, provide a massive dataset for the training of other models related to traffic and traffic signs. Secondly, the research focused on a trained model for the detection of traffic types.

1.4 Methodology

The research was conducted in 3 distinct phases. Firstly, the data was collected in form of videos. The different cameras used for the videography include smartphone cameras and a dashcam, all mounted on the car windshield. The videos were taken in different lighting conditions to avoid low variance in the evaluation model. From these videos, the keyframes were extracted and subsequently annotated against the set classes. Secondly, the self-collected

video frame dataset was used for training a Deep Learning-based model, YOLOv7, for the detection of traffic types. The model was cross-validated and regularized to help improve the model's working, and then tested in real-world scenarios and tweaked according to requirements. Finally, the final model was used to develop an Advanced Driver Assistance System—ADAS—which works on a priority-based decision system, dependent on the different traffic types on the road, providing specified instructions for the detected conditions at that time (traffic in front, overtaking vehicle, etc.).

1.5 Thesis Overview

The thesis is further divided into the following chapters; firstly, the current literature present on the topic is reviewed in detail, in the **Literature Review** chapter, to extract the shortcomings and research gaps in relevant state-of-the-art solutions. Afterwards, based on an in-depth analysis of these issues, the process used to reach the solution has been described along with the simulation setup in the **Methodology** chapter, which also explains the process of data collection and all the pre-processing which has gone in to make the data ready for the detection models. Next is the **Results** chapter which discusses the output of the models, its training, validation, and testing results along with other performance metrics. The **Discussion** chapter revolves around the novelty, improvements over the state-of-the-art, and future work possible in the research area. Finally, the **Conclusion** chapter rounds up the article's key achievements and outcomes of the conducted research.

CHAPTER 2: LITERATURE REVIEW

2.1 Related Work

The idea of a system that assists the driver to possible blind spots and missed cues has been discussed as an easier way to automation and autonomy. Some of the basic elements of such Driver Assistance Systems included sensing the environment for different obstacles, traffic-signs, pedestrians, and other traffic [8], [9], [10], [11], [12].

Traditional Image Processing techniques were previously used [13], [14], [15], [16], [17], [18], [19]. However, these methods are way slower compared to the current state-of-the-art practices, and that is cause for hindrance. Considering the application of our project, the accuracy and precision of the ADAS is paramount to the safety of the user, the vehicle, and the traffic on the road.

In several recent studies the traffic is detected and identified up to a certain degree of accuracy [20], [21], [22], [23]. CNNs have been deployed heavily in Traffic detection and identification. CNNs require a huge number of images to work efficiently, but Pakistan lacks any such local dataset pertaining to Traffic types.

Such datasets are purposefully built and curated and regularly used to inspire new learning-based model by pitting them against each other in competitions [7]. This has proved fruitful as these competitions have resulted in a great deal of literature and state-of-the-art computer vision models being developed all of which seem to agree on the assumption that more data mean better performance.

To avoid any potential accidents involving traffic, such as forward collision and vehicle overtaking, work has been done for assistance with safe lane change operations using symmetry verification to detect lanes [17], [24], [25]. Research has also been conducted on giving priority to certain traffic types [20], [21], [22], [23], . This helps to reduce the damage caused due to road accidents, or in cases of [21] and [22] to help ambulances and firetrucks avoid traffic-jams with help of smart city surveillance systems.

2.1.1 Traffic Type Recognition

Identification of traffic type is an important part of self-driving cars and assisted driving systems. Considering these applications, it is extremely important that the traffic is detected and identified up to a certain degree of accuracy to ensure that the vehicle and the passengers in it are safe. Research on traffic detection and recognition has been carried through different

methods, using both traditional image processing based, and learning based algorithms.

A paper titled “Pedestrian, bike, motorcycle, and vehicle classification via deep learning Deep Belief Network and small training set” [21] presented a different and interesting approach to traffic classification. A Deep Belief Network (DBN) based model was used for the classification of four categories, namely: pedestrian, bike, motorcycle, and vehicle. The proposed model achieved a high accuracy classification rate of 89.53%, especially considering that the model was trained using only 1000 images. However, that also raises a red flag since a small dataset can cause the model to have sampling bias and under-perform on outliers.

Another paper titled “Lightweight PVIDNet: A Priority Vehicles Detection Network Model Based on Deep Learning for Intelligent Traffic Lights” [22] introduced an algorithm for vehicle detection based on YOLOv3, integrated with an intelligent traffic light. The use of YOLOv3 as base model provided a lightweight design with low execution time. The proposed network was also used for traffic control after being trained on the Brazilian Traffic Code. However, this model worked on a single image basis and not on continuous video frames.

A recent paper titled “A deep-learning-based computer vision solution for construction vehicle detection” [23] published in 2020 proposed an improved version of the single shot detector MobileNet. Nearly all MobileNet based architectures use the ‘depthwise separable convolutions’ which basically make use of the two operations: depthwise convolution and pointwise convolution. An mAP value of 0.912 was achieved but, the model was only trained and tested for detection of construction vehicles.

2.2 Convolutional Neural Networks

According to Li Deng and Dong Yu [26], deep learning is a class of machine learning algorithms that uses raw input to extract features by utilising several layers. The features extracted are of a higher level, incorporating greater detail into the model. Deep learning was introduced to machine learning by Rina Dechter in 1986 in [27]. Since then, various developments have taken place over time such as the daw of neural networks working in both supervised and unsupervised conditions. Deep learning or convolutional neural networks (CNN) are part of the unsupervised realm of machine learning.

Among the pioneers of the development were Y. Lecun, L. Bottou, Y. Bengio and P. Haffner who developed the LeNet [5]. It was a 7-level convolutional network used to classify handwritten digits on cheques. Its major constraint was the high computational requirement. Later in 2012 Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton developed the CNN,

AlexNet [28] that won the annual ImageNet Large Scale Visual Recognition Challenge (ILSVRC) held by ImageNet. The network had an architecture akin to LeNet but was deeper and consisted of more filters. Filters included 11x11, 5x5, 3x3, convolutions, max pooling, dropout, data augmentation, ReLU activations and SGD with momentum.

Then in 2013, the ILSVRC was won by ZFNet, also a CNN, developed by Zeiler, Matthew D., and Rob Fergus [29] achieving a top-5 error rate of 14.8 percent, better than last year's AlexNet which had a top-5 error rate of 15.3 percent. This was achieved by primarily fine-tuning and tweaking the hyper-parameters of the AlexNet architecture. In 2014, the competition was won by GoogleNet, codenamed in the journal Inception V1 [30]. It made the first big leap after the AlexNet in terms of a top-5 error rate of 6.67 percent. It is based on the LeNet architecture and used batch normalisation, image distortions ad RMSprop. This novelty is dubbed the *Inception Module.* This worked on reducing the number of parameters, using a 22-layer deep CNN to reduce the parameter from 60 million of AlexNet to 4 million. The runner-up to the GoogleNet was the VGGNet developed by Simonyan, Karen and Zisserman, Andrew [31] of the Oxford Robotics Institute.

The VGGNet consisted of 16 convolutional layers of 3x3 convolutions with more filters than the AlexNet. Its uniform structure makes it a go-to for various applications as a baseline feature extractor. Then in 2015, ResNet took the ILSVRC crown, formally called the Residual Neural Network (RNN) [32]. RNN introduces skip connections, also called gated units allow this neural network to use 152 layers while retaining a computational complexity less than VGGNet. It achieved a top-5 error rate of 3.57 percent.

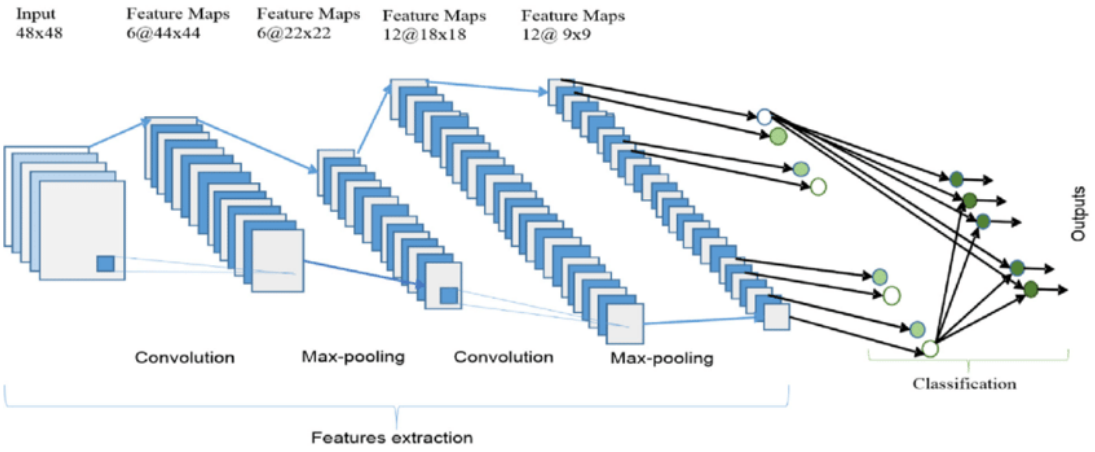

Figure 1: Convolutional Neural Network [33]

Convolutional Layers are responsible for the convolution of the Input Image and the filter to extract the required features and generate a feature map according to the filter size.

Filter size is determined by the size of the Input Image. The filter consists of two parts the filter size *F* and the total amount of filters *K*. The input of the convolutional Layer would be the Input Image dimensions (*W(i) * H(i) * D(i)*) and the output (*W(o) * H(o) * D(o)*) where *D(o)* is equal to the total amount of filters *K* and *W(o)* and *H(o)* can be calculated by the following equation [34].

$$\frac{((W(i),\ H(i)) - F) + 2p}{Stride + 1}$$

Where,
W(i), H(i) is the Input Size of the square image
F is the Filter Size
p is the Padding

Parameters for each convolution layer are calculated to get the overall trainable and non- trainable parameters in the model and to calculate the complete memory consumption of the network. If we have an input of (*W(i) * H(i) * D(i)*) and a convolution filter *(W(f) * H(f) * D(f))* where *W(i)*, *H(i)* and *D(i)* are the Width, Height and Dimension of the input to the convolutional layer and *W(f)*, *H(f)* and *D(f)* are the width, height and total number of feature maps in a convolution filter. Thus, the parameters can be calculated by using the following formula:

$$(W(f) * H(f) * D(i) + 1) * D(f)$$

Pooling Layers are used to reduce the total number of parameters which will be used further in the network, and it also reduces the overall computational cost [34]. Most commonly used pooling techniques include Average Pooling and Max Pooling.

Dropout Layers were made to avoid overfitting or underfitting of the model on the given dataset. It chooses the number of nodes which will be used in the training process. These Layers are commonly used after fully connected layers which are prone to overfitting [34].

Activation function helps in providing the non-linear relation between the class of image and Image Data. They determine which neuron should be fired or not depending upon the relevancy of the neuron towards the required output [34]. Various activation functions are being used which include tanh, sigmoid, ReLU, Leaky ReLU etc.

Optimization techniques are used to calculate the weights for your model. They update the weights in the learning process until you reach your desired output. Various optimization techniques are used which include SGD, SGD with momentum, NAG, Adagrad, RMSprop and Adam [34].

Flatten Layers are responsible for converting the data into a one-dimensional vector so

it could be fed to Fully Connected Layers where classification will be completed.

Fully Connected Layers are the feed-forward neural networks. The first FC layer collects the data from the last convolution Layer after getting flattened into a one-dimensional vector to compute the classification and the Last FC Layer provides the final probabilities calculated for each label.

The Accuracy is calculated by using the f1 Score which has two metrics Precision and Recall [68]. Precision describes the number of true class predictions which truly belongs to the true class whereas recall defines the number of true class predictions completed out of all the true samples in the complete dataset.

Formulae for each are given below,

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$f1\ Score = 2 . \frac{Precision\ .\ Recall}{Precission\ +\ Recall}$$

Where *TP* is True Positive, *FP* is False Positive, *FN* is False Negative

CHAPTER 3: METHODOLOGY

3.1 Data Collection

Deep Learning-based learning algorithms require massive amounts of data to be able to generalize. This is due to their inherent properties of modeling around the available training data. This is an area in which we as a country lack and there is a severe shortage of available data. To rectify this issue this research included a data collection phase where videos were collected from across Pakistan.

The videos collected were 30 in number and deliberately collected with varying properties such as framerate (FPS), brightness, exposure, and lighting settings. These properties are known to affect the results of any learning-based model ultimately. The framerate is responsible for the number of frames being extracted from each second and will affect the number of images in total.

The total runtime of these 30 videos amounted to 05 hours, 42 minutes, and 01 seconds. Of these 30 videos, 23 were collected from across a few cities in Pakistan including, but not limited to, Quetta, Karachi, Lahore, Islamabad, and Rawalpindi, and totaled 02 hours, 35 minutes, and 58 seconds of video footage. Further videos were fetched from various open-source video-sharing platforms with a total of 03 hours, 06 minutes, and 03 seconds.

3.2 Preprocessing

The next part after data collection is getting that ready for training and it starts with extracting individual frames from the video footage. The video framerate dictates the number of extracted frames from each second of that video. Considering various framerates of each video and the total runtime all the videos equate to approximately 0.56 million frames/images. Even though it was stated above that the number of training images is generally directly proportional to model performance, in this specific case a lot of the extracted frames had little or no visual change and would only prove expensive processing-wise. This is because the spatial features present in these adjacent frames are usually very similar and will not add any benefit.

To cater to this, ‘key-frames’ were extracted. This resulted in 109,463 final number of frames. Furthermore, the input dimensions of all the images/frames being passed on to any learning-based algorithms need to be constant. In the case of this research, this resolution was fixed at 640 × 380. Another reason for setting the resolution to this specific value was that the

videos also differed in aspect ratios and resolutions and some of these were very high. Higher resolution, while carrying better spatial features, also mean longer training time and use of precious computational resources. After a certain point, it becomes important to look at the cost-benefit analysis of the input resolution. It is generally observed that after a certain resolution, any increase will return a very negligible increase in model performance, but it will take a significantly longer time to train.

3.3 Data Annotation

After preprocessing the video to set requirements, the frames need to be annotated for the presence of relevant objects. This is important as the annotations are the labels that are passed to the deep learning model in a supervised learning scenario. The annotations are done using the Computer Vision Annotation Tool—CVAT—from Intel. It can output the annotations in various formats depending on the type of model being trained.

The types of objects to be detected were divided into four main classes. The four types of traffic being considered in this research are pedestrians, bikes, LTVs, and HTVs.

Figure 2: Annotation Examples

3.4 Flow Diagram

The overall flow of the data and all the individual steps are shown in the flow diagram in Fig. 2 below. The process starts with key preprocessing as detailed in section 3.2 above, it includes the key frame extraction, resizing and train/test split steps. The train/test split is done to distribute the data into two parts, one used for training the data and the other used to test the performance of the model by emulating real-world conditions where the model will encounter unseen traffic signs and types. The preprocessed data is then annotated and then passed on to the proposed Convolutional Neural Network—CNN.

The proposed CNN will carry out three steps in general—using varying techniques based on type of CNN being used—extracting relevant features, detecting objects, and

classifying them. The CNNs being used for the task of this research is YOLOv7 and the detailed architecture has been further detailed in the next section. Consequently, predictions are made, and further decision are taken based on them.

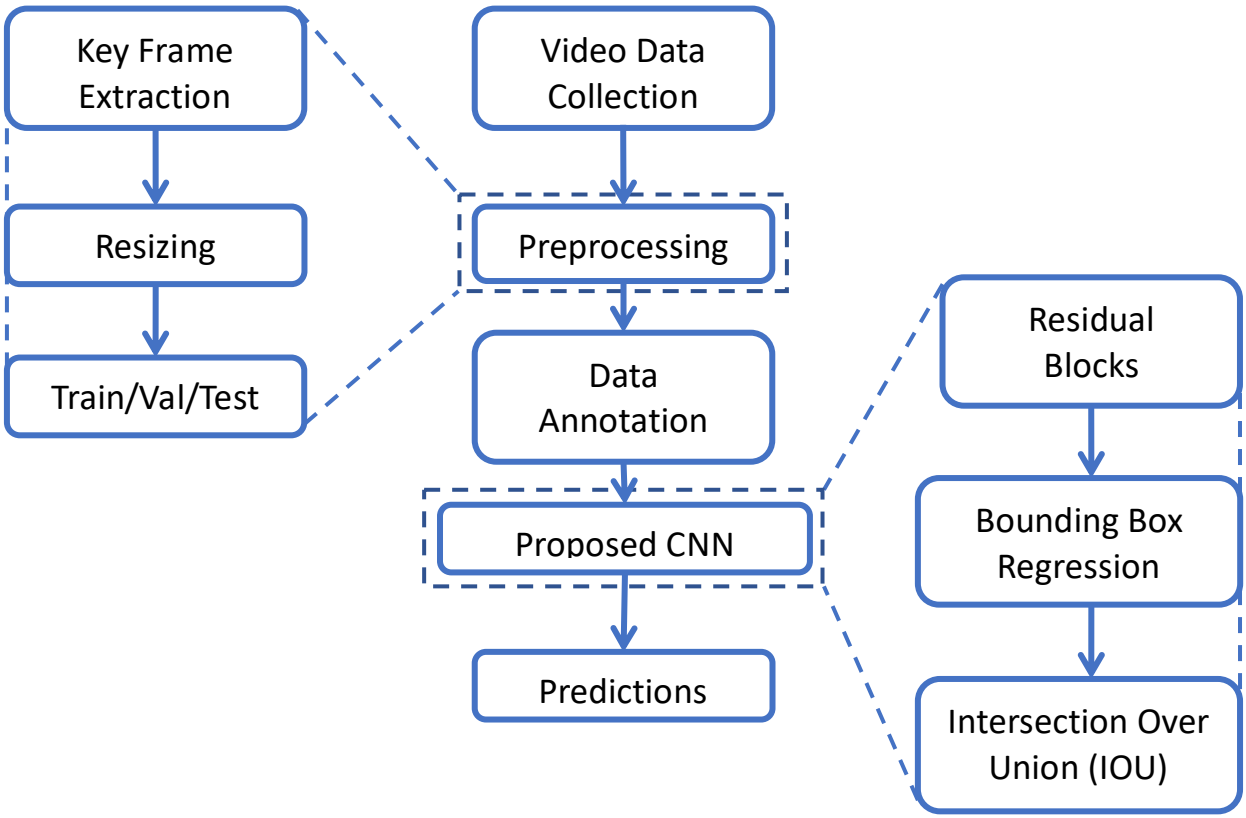

Figure 3: Flow Diagram

3.5 Experimentation Setup

The training process is performed on an Nvidia Tesla P100 GPU provided by the Google Collaboratory. The training has been performed for 1000 epochs/iterations to obtain precise and stable results. The training time is 26:27:02 on the aforementioned GPU.

3.6 Model Architecture

The first version of the YOLO object detector was introduced in 2015 in the paper titled "You Only Look Once: Unified, Real-Time Object Detection" [35]—YOLOv1. Since then, multiple versions, and flavors, of the base model have been released, up to the 7th version—which was presented in the paper titled "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object" [36]. In our research, YOLOv7 architecture is used for the detection and identification of traffic types. It makes the predictions for the bounding boxes

more accurately than its predecessors, and at similar inference speeds.

YOLO is a regression-based algorithm but, instead of selecting important features of an image, it predicts bounding boxes and labels/classes for the image, and the most significant element of this algorithm is that it does all this in one run—pertaining to its name "You Only Look Once". Ultimately, the aim is to predict the object class and specify the object's location through a bounding box.

The base model of YOLO, on which every version is based, consists of three main modules:

- Residual Block: The first module takes the input image and divides it into various grid cells, typically, 19x19. Each grid cell is then responsible for detecting objects that may appear within them, based on the location of the object's center.
- Bounding Box Regression: The second module uses Single Bounding Box Regression on any and all detected objects. This provides the probability of an object appearing in the bounding box—the outline or boundary that highlights the location of the object in the input image. For each bounding box, there are four attributes to predict: center (x, y), width, height, and class.

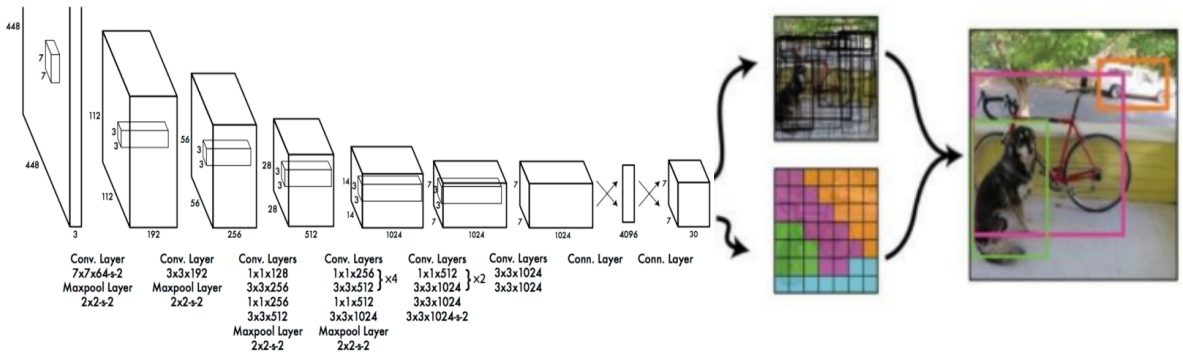

Figure 4: Model Architecture

- Intersection Over Union (IOU): The third module, using the concept of IOU—a description of how the bounding boxes overlap—provides an output box in which the objects are perfectly surrounded. Every grid cell is tasked with the prediction of the bounding boxes and their probability, or confidence score. If the prediction for the bounding box is the same as the real bounding box, the IOU value equals 1, and consequently, any predicted bounding box that is not equal to the real bounding box is eliminated.

3.7 Advanced Driver Assistance System

The ADAS has been designed as an active system to assist in a wide array of issues while driving. Some of issues include, but are not limited to:

- Lane-keeping
- Over-taking vehicles
- Jay-walking pedestrians
- Potential blind spots
- Negligent/distracted drivers

The ADAS Unit consists of two modules. The first module divides each video frame into a grid of 3 x 2 cells. For every predicted bounding box centers (bx ,by), their location is then identified in the grid, along with the predicted labels for traffic type. Using the labels and bounding box locations, the second module provides specified instructions for the detected conditions at that time (traffic in front, overtaking vehicle, etc.).

Given in table 1 below, in decreasing order of priority, is the list of instructions provided as assistance to the driver by the ADAS according to different situations detected.

Table 1: ADAS Instructions for Different Situations

#	**ADAS Instructions**
1	No traffic detected, assistance not required
2	Pedestrian right in front of you, stop immediately
3	Bike right in front of you, slow down immediately
4	LTV right in front of you, slow down2 immediately
5	HTV right in front of you, slow down immediately
6	Pedestrian in front of you, slow down immediately
7	Bicycle in front of you, slow down slightly
8	LTV in front of you, stay cautious
9	HTV in front of you, slow down slightly
10	Pedestrian near your left side, watch out
11	Pedestrian near your right side, watch out
12	Bike overtaking from your left side, stay cautious
13	Bike overtaking from your right side, stay cautious

14	LTV overtaking from your left side, stay cautious
15	LTV overtaking from your right side, stay cautious
16	HTV overtaking from your left side, stay cautious
17	HTV overtaking from your right side, stay cautious
18	Pedestrian on your far-left side, watch out
19	Pedestrian on your far-right side, watch out
20	Bike on your far-left side, be careful while overtaking
21	Bike on your far-right side, be careful while overtaking
22	LTV on your far-left side, be careful while overtaking
23	LTV on your far-right side, be careful while overtaking
24	HTV on your far-left side, be careful while overtaking
25	HTV on your far-right side, be careful while overtaking

CHAPTER 4: RESULTS & DISCUSSION

4.1 Performance Metrics

It is pertinent to mention discuss the performance metric being used to characterize the predictions of the model. The primary metrics are True Positive—TP, False Positive—FP, True Negative—TN, and False Negative—FN. These are further explained in table 2 below:

Table 2: Performance Metrics

True Positive	It is when a model **makes** a prediction and correctly identifies the object
False Positive	It is when a model **makes** a prediction even though no object was present
True Negative	It is when a model **does not make** a prediction when there is no object
False Negative	It is when a model **does not make** a prediction even though an object was present

Another aspect which needs to be considered when an object detection model is being used is Intersection over Union—IoU. IoU is a measure of how much the predicted bounding box overlaps the original—or ground truth—bounding box in the case of a prediction being made, i.e., it is a ratio of the area of overlap and the total area covered by the original and predicted bounding boxes as given by the equation below

IoU = Intersection of bounding boxes' areas/Union of bounding boxes' areas

This metric is used along with a threshold to classify bounding boxes in accordance with one of the primary metrics. So as, if a bounding box is below the required threshold, it is classified as a false positive because it made a prediction, but that prediction did not have enough quality in it to be called a correct prediction or a true positive. The IOU threshold can be varied depending on various situations and applications as well as the size of the object under observation, but the default or generally accepted value is set at 0.5.

Furthermore, the primary metrics combine to form secondary metrics, which are Precision and Recall and are given by the equations given in section 2.2.

Precision is a measure of accuracy of the model's predictions, i.e., the number—or percentage—of correct predictions made by the model with respect to the total number of predictions made. While recall is the measure of how well the model is predicting the presence of objects, i.e., the number—or percentage—of objects detected with respect to all the objects present.

This is then succeeded by the tertiary metric called Average Precision which is generally defined as the area under the precision-recall curve. This can be calculated by simple integration as per equation below. The main metric being used to characterize the findings of this research is the mean Average Precision (mAP). mAP is the cumulative mean of the Average Precision across all the classes of the object being predicted and is given by the equation below.

$$p_{interp}(r) = \max_{\tilde{r} \geq r} p(\tilde{r})$$

$$mAP = \frac{1}{n} \sum_{k=1}^{k=n} AP_k$$

$$AP_k = \textit{the AP of class } k$$

$$n = \textit{the number of classes}$$

4.2 Results

Traffic type images were trained on two different state-of-the-art networks and model architecture of the YOLO family. Both of the selected networks have shown remarkable results and groundbreaking mAP numbers on the COCO Dataset over the year. As we can see in table 3 and table 4 below the YOLOv7 model architecture produces the higher values for all the relevant benchmarks. Here the "P" means Precision, "R" means Recall, "mAP@0.5" means mAP over IOU threshold 0.5, and "mAP@0.5:.95" means mAP over different IOU thresholds, from 0.5 to 0.95 with a jump of 0.05.

Table 3: Performance Metrics for the Trained Models

Architecture	**P**	**R**	**mAP@.5**	**mAP@.5:.95**
YOLOv5	0.877	0.623	0.746	0.430
YOLOv7	0.876	0.730	0.872	0.579

Table 4: Performance Metrics for the Top Performing Model

YOLOv7				
Class	**P**	**R**	**mAP@.5**	**mAP@.5:.95**
all	0.876	0.730	0.872	0.579
pedestrian	0.925	0.691	0.841	0.548
bike	0.930	0.784	0.899	0.564
HTV	0.818	0.500	0.834	0.452
LTV	0.959	0.674	0.873	0.560

Figure 5: ADAS Instructions Examples

4.2.1 Graphical Results

The graphical results of several other common and useful metric for the best performing model are given below.

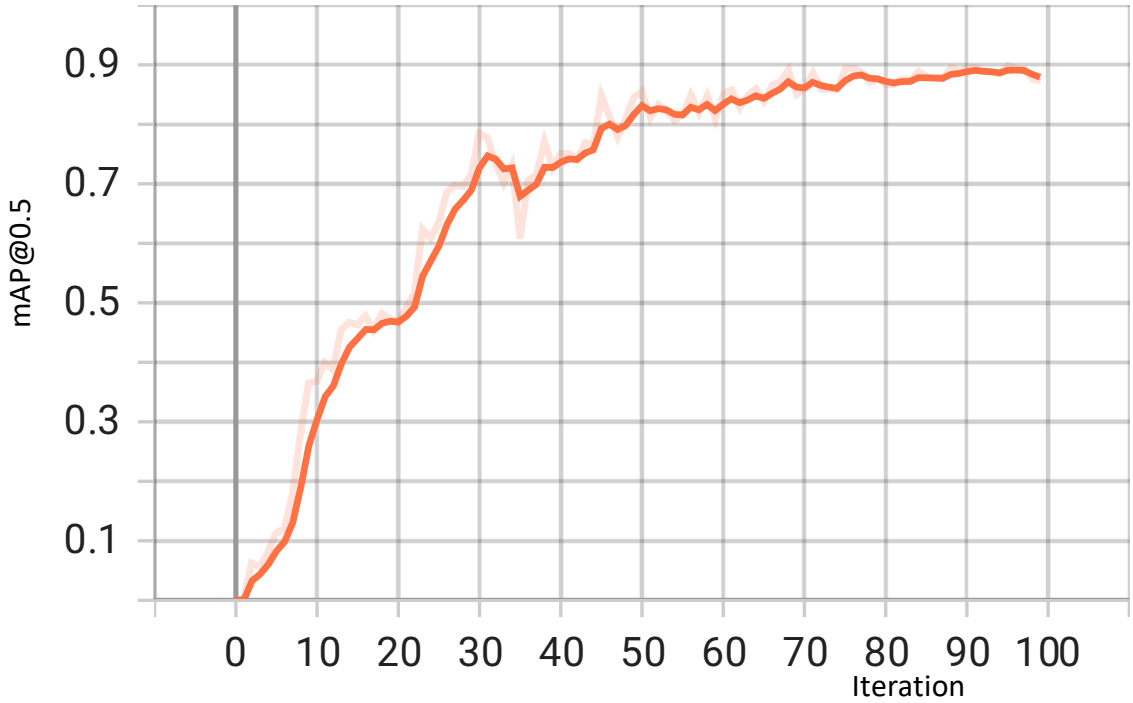

Figure 6: mAP@0.5 for YOLOv7

The mAP is cumulative mean of the Average Precision across all the classes of the object being predicted. As visible from the graph in Fig. 4 above, the mAP over IOU threshold 0.5 improves consistently and ultimately flattens out after about 70 iterations with a final value of 87.20%.

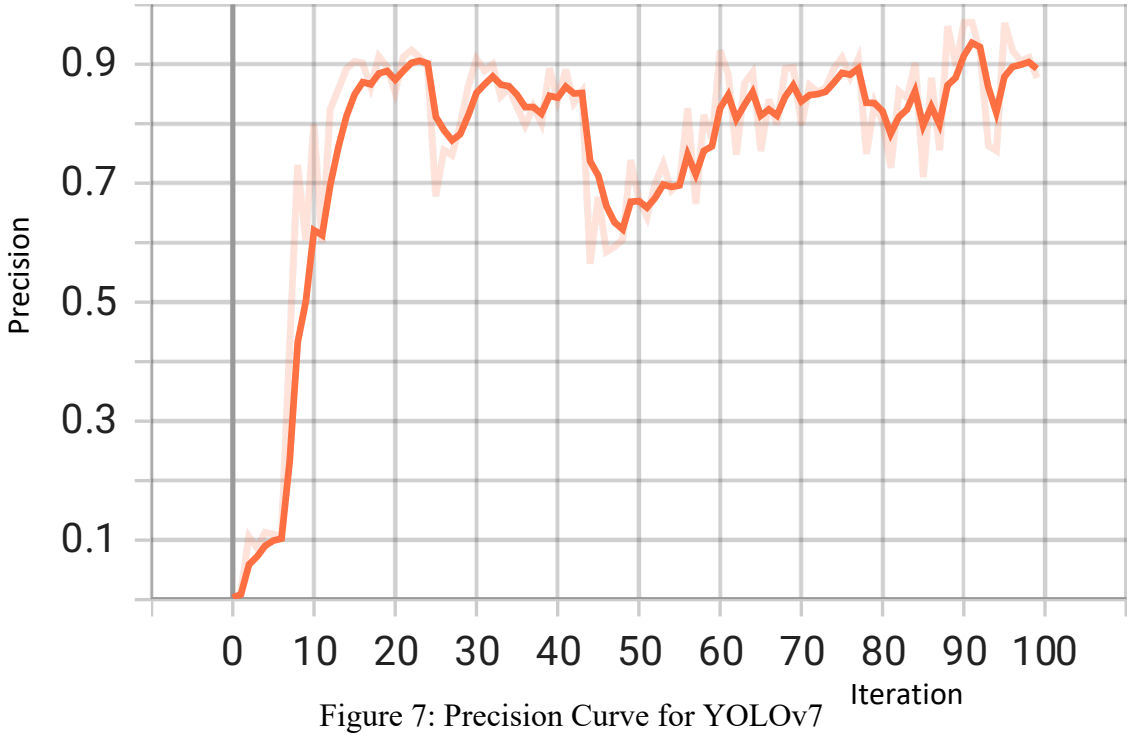

Figure 7: Precision Curve for YOLOv7

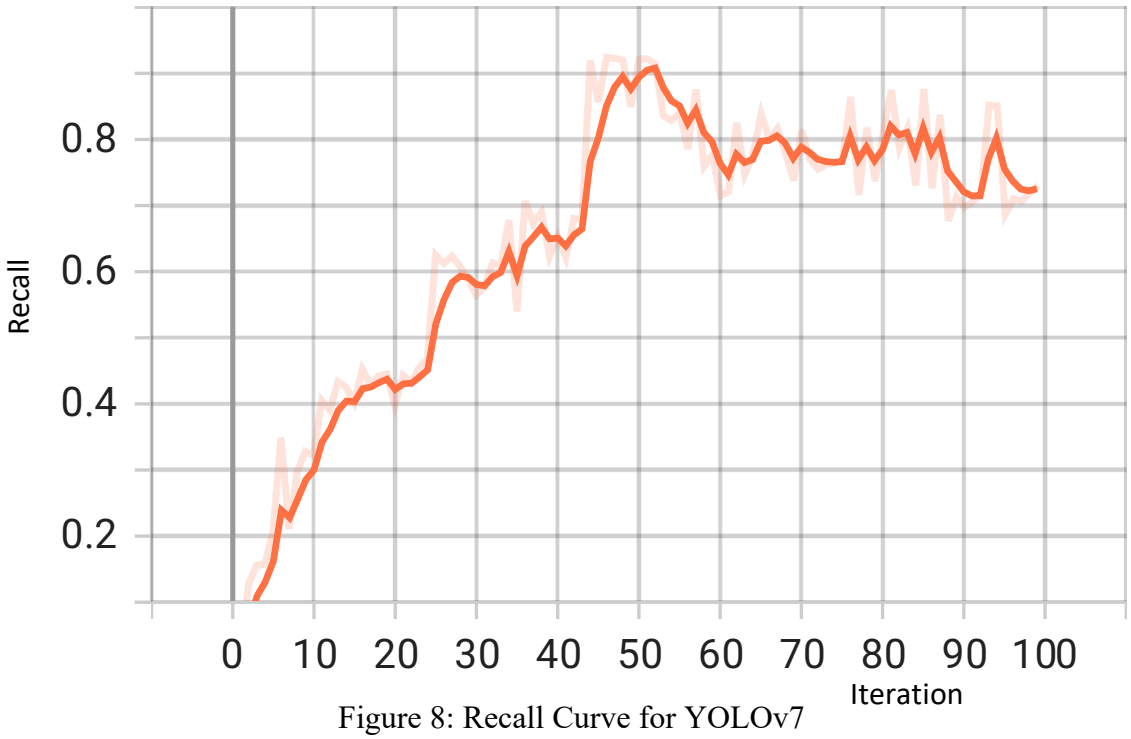

Figure 8: Recall Curve for YOLOv7

Precision is a measure of accuracy of the model's predictions, i.e., the number—or percentage—of correct predictions made by the model with respect to the total number of predictions made. While recall is the measure of how well the model is predicting the presence of objects, i.e., the number—or percentage—of objects detected with respect to all the objects present. The values for Precision and Recall, as shown in Fig. 5 and Fig. 6, respectively, improve significantly as training progresses.

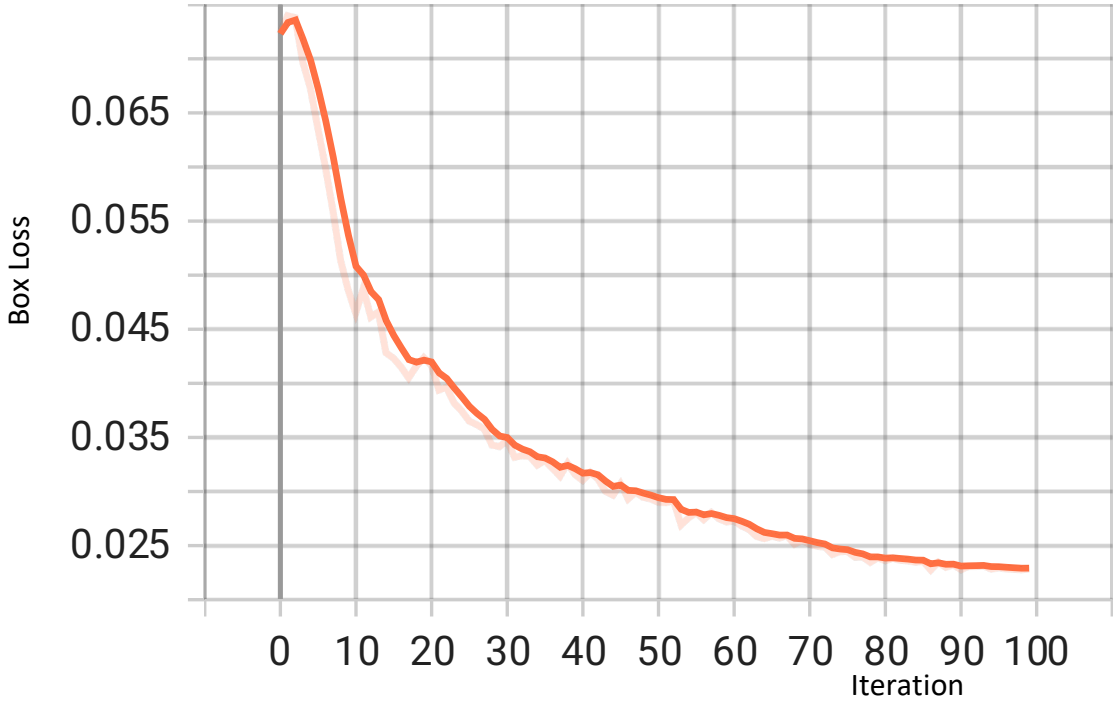

Figure 9: Bounding Box Regression Loss for YOLOv7

Bounding boxes are the rectangles drawn around the detected object, as mentioned in the model architecture the bounding box coordinates are regressed in a branch of the network, using Mean Square Error. The loss of this regression is a strong indicator of how well these boxes are 'bounding' the objects. The values of regression loss decrease continuously to a very low value, showing the improvements as the training progresses.

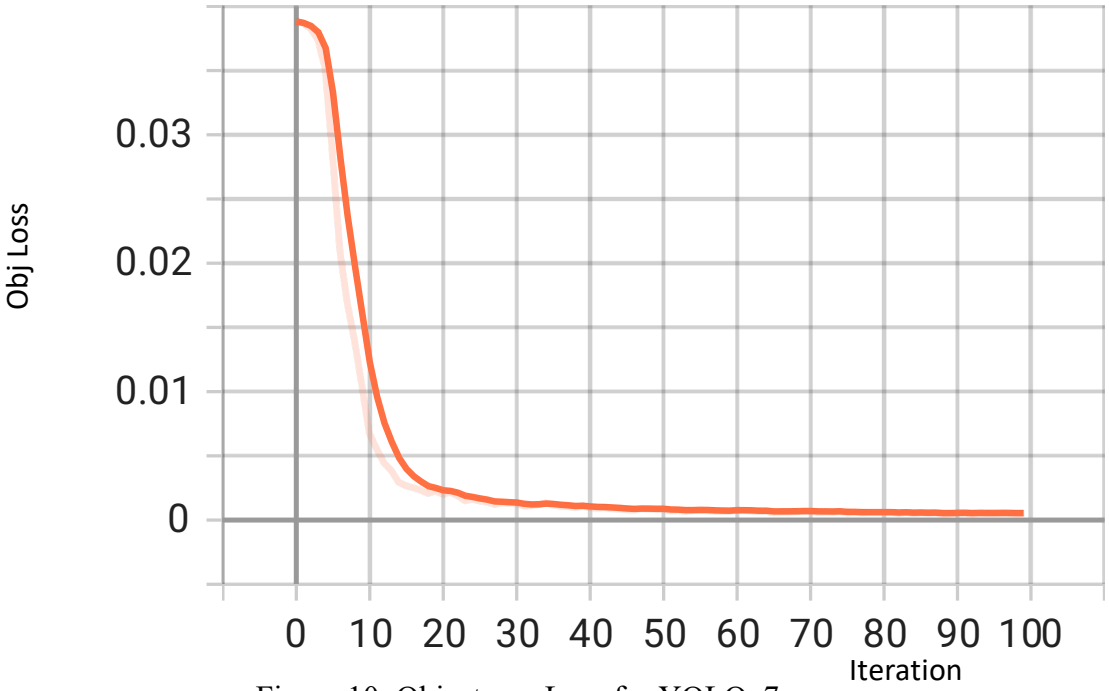

Figure 10: Objectness Loss for YOLOv7

Objectness Loss is another major identifier of a model's performance. Objectness is a

relatively new term in performance metrics. It can be described as the confidence a network has in an object existing in a predicted bounding box. Objectness Loss helps the network predict a correct IOU by using Binary Cross Entropy. The graph above shows a steady decrease as the iterations increase showing the improvement in model's performance.

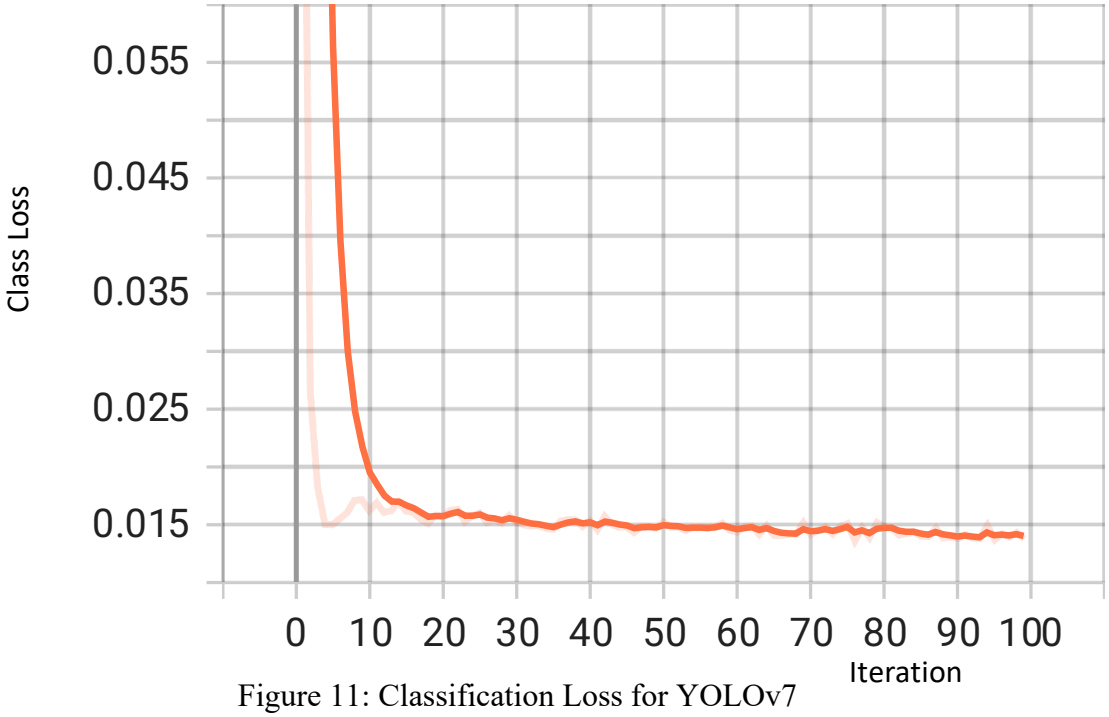

Figure 11: Classification Loss for YOLOv7

YOLOv7 classifies the detected objects into one of the predefined classes. This part of the architecture uses Cross Entropy, and the loss generally trends downwards quite early and stays more or less steady after about 40 iterations or so.

4.2.1.1 YOLOv5

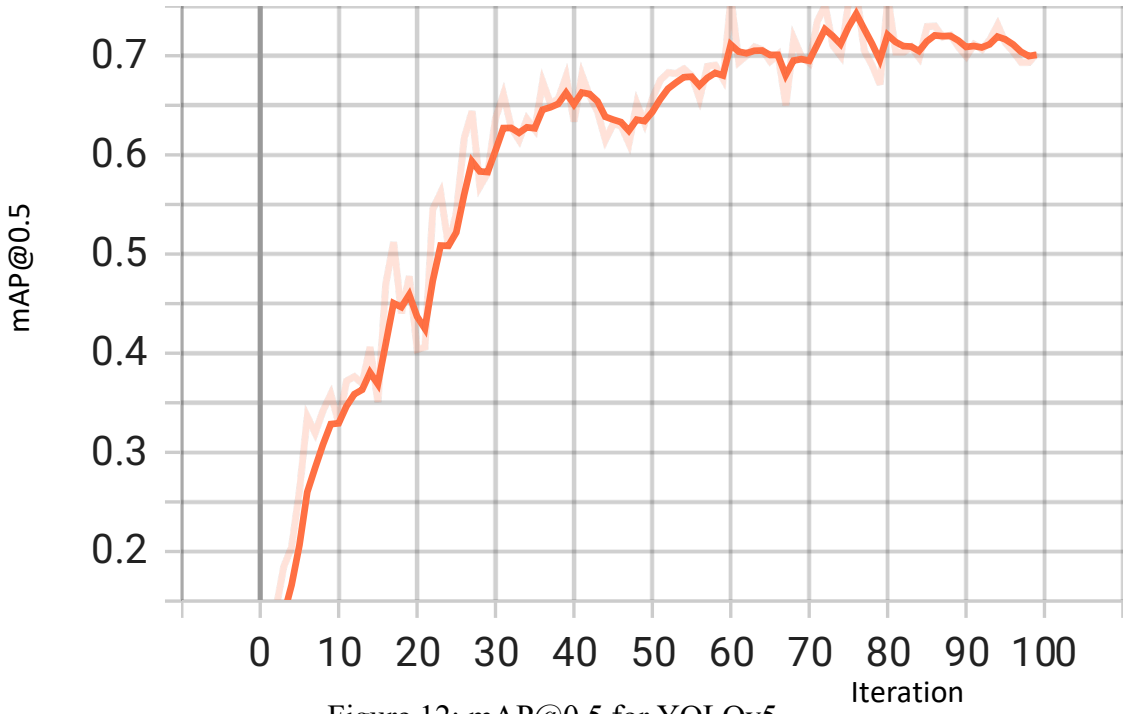

Figure 12: mAP@0.5 for YOLOv5

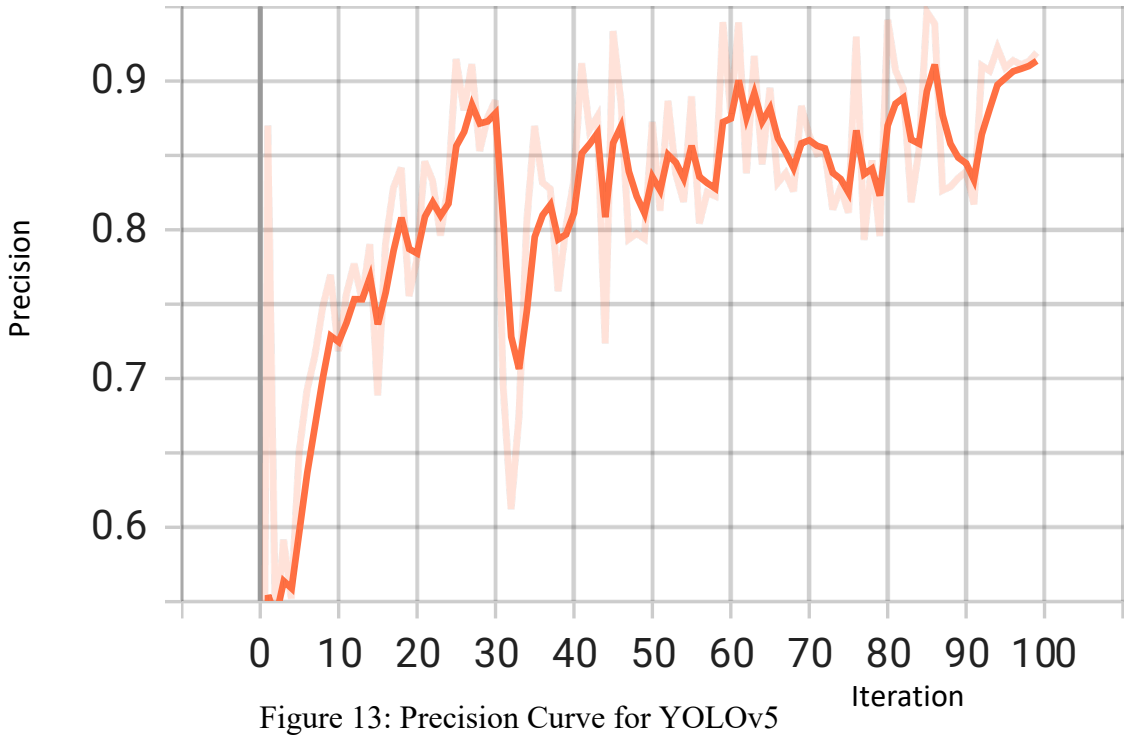

Figure 13: Precision Curve for YOLOv5

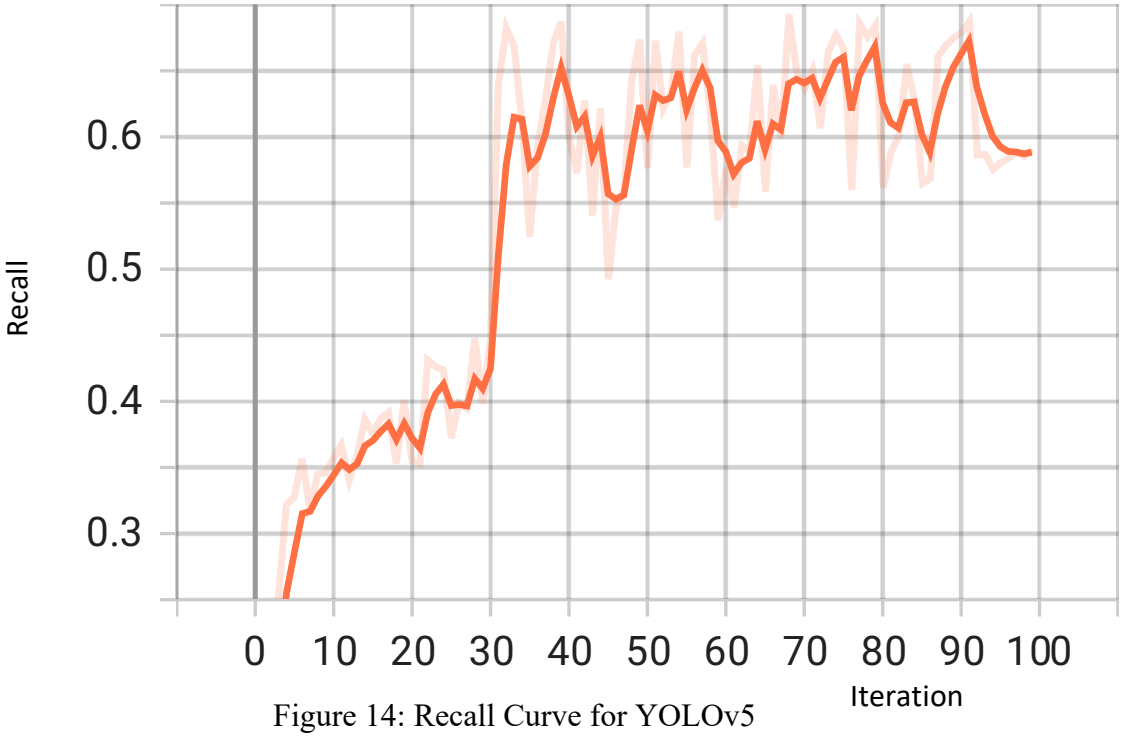

Figure 14: Recall Curve for YOLOv5

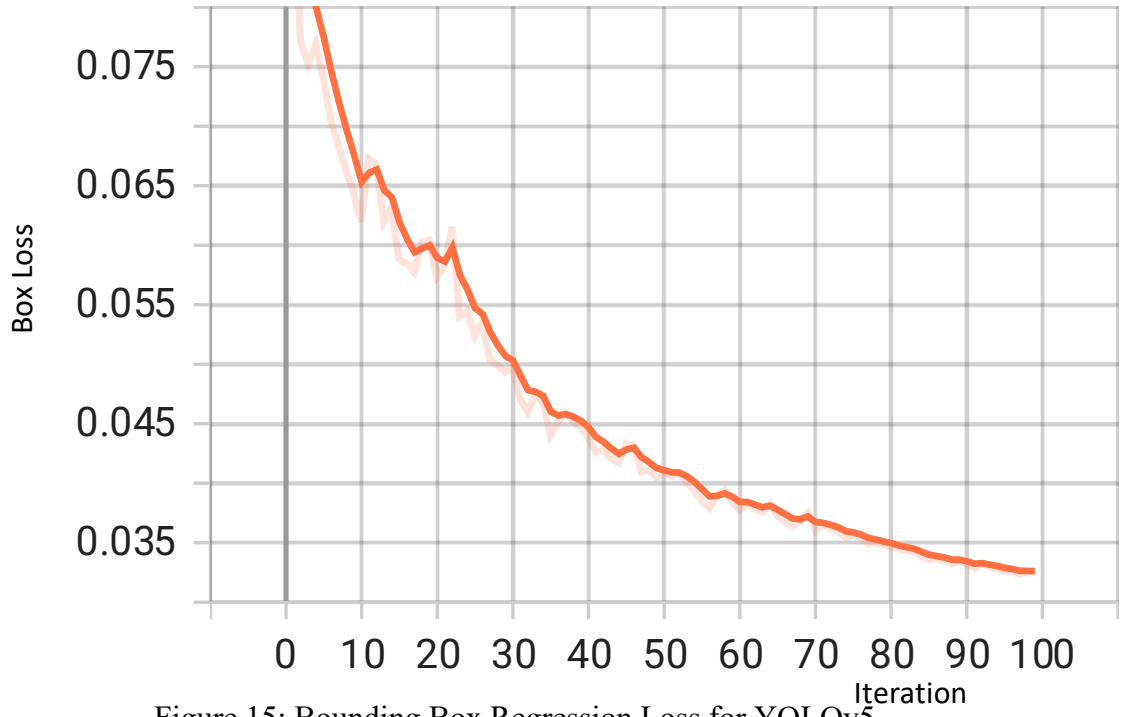

Figure 15: Bounding Box Regression Loss for YOLOv5

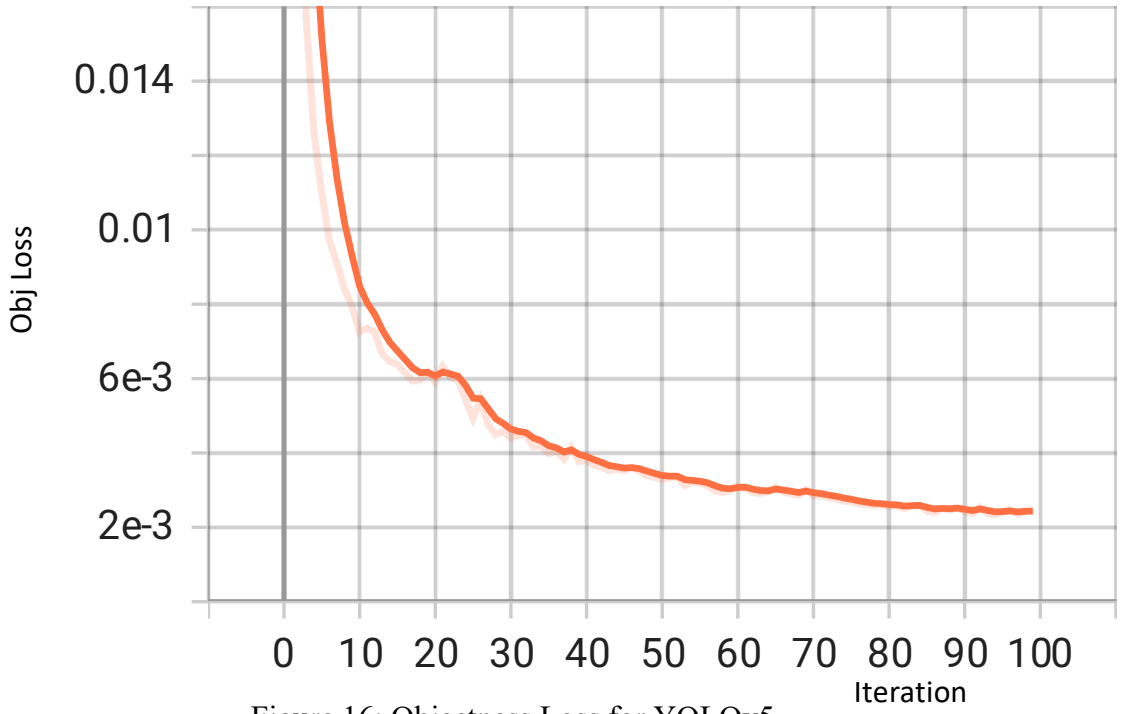

Figure 16: Objectness Loss for YOLOv5

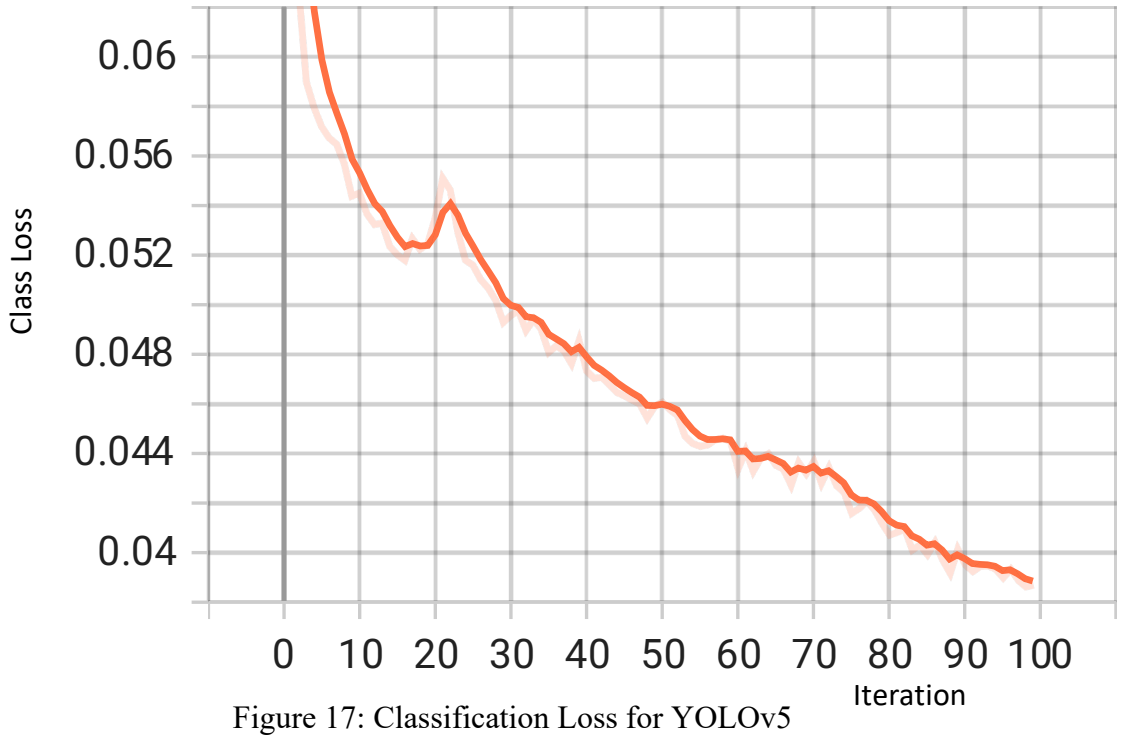

Figure 17: Classification Loss for YOLOv5

4.3 Discussion

The novelty of this research lies in the collection of vast amounts of data pertaining to Pakistani traffic types. This data as mentioned in detail above has been collected from several cities while having a diverse range of visual features and artifacts. These include images with a varying range of exposure, brightness, and occlusion. Furthermore, this dataset is annotated

for traffic type detection and can be used by researchers to develop and improve their models for the roads of Pakistan.

Furthermore, this is pioneering research in the implementation of multiple object detection-based systems to recognize the Pakistani traffic type. It is the first of its kind research in Pakistan which has a model trained on frames from video footage from the streets. It also scores impressively in all performance metrics used internationally to characterize related models.

CHAPTER 5: FUTURE WORK

This research was meant as an initial foray into enabling better traffic conditions in Pakistan. Even though the research resulted in favorable outcomes and performance metrics there is always room for improvement. The performance metrics can be improved by the use of data augmentation to help account for class imbalance. More training iterations/epochs can be used in case of the availability of better computational resources. Another case of improvement can be the use of all frames instead of extracting keyframes and using them at a higher resolution to improve results.

Object detection is a hot research topic hence recently published models can be used for higher performance scores. RCNN and image segmentation models to can also be used to obtain pixel-wise detection but, that would require an even more tedious annotation process which could result in better real-world results when deploying the model. The research could be compounded by the development of specialized hardware to help turn the research project into a commercially viable product.

CHAPTER 6: CONCLUSION

The motivation behind conducting this research was to lay a foundation for a dataset and a model which can be used by self-driving vehicles and existing vehicles to improve the traffic conditions in the country. During this research, a first-of-its-kind dataset was collected from the roads of Pakistan and across various cities including, but not limited to, Islamabad, Quetta, Lahore, and Karachi. The dataset amounted to **5 hours, 42 minutes and 1 second** of video footage, and 109,463 images of keyframes. The footage was annotated using rectangular bounding boxes and **4 distinct classes** which were pedestrians, bikes, LTVs, and HTVs.

Consequently, a deep learning model was trained for the traffic signs. It was YOLOv7 from the YOLO architecture family. It was trained and tested to detect and classify traffic types at an mAP of **87.20%.** Using predictions from this model, an ADAS algorithm was designed to assist in a wide array of issues while driving.

REFERENCES

[1] T. Jochem, D. Pomerleau, B. Kumar, and J. Armstrong, "PANS: A portable navigation platform," in *Proceedings of the Intelligent Vehicles' 95. Symposium*, 1995, pp. 107–112.

[2] X. J. Zhu, "Semi-supervised learning literature survey," 2005.

[3] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," *Mach. Learn.*, vol. 39, no. 2, pp. 103–134, 2000.

[4] "Riding shotgun in Tesla's fastest car ever," *Engadget*. https://www.engadget.com/2014-10-09-tesla-d-awd-driver-assist.html (accessed Jul. 28, 2022).

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[6] X. Yin, J. Han, J. Yang, and P. S. Yu, "Crossmine: Efficient classification across multiple database relations," in *Constraint-Based mining and inductive databases*, Springer, 2006, pp. 172–195.

[7] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: a multi-class classification competition," in *The 2011 international joint conference on neural networks*, 2011, pp. 1453–1460.

[8] "Alarming figures of traffic accidents need attention." https://www.thenews.com.pk/print/910436-alarming-figures-of-traffic-accidents-need-attention (accessed Jul. 29, 2022).

[9] V. K. Kukkala, J. Tunnell, S. Pasricha, and T. Bradley, "Advanced driver-assistance systems: A path toward autonomous vehicles," *IEEE Consum. Electron. Mag.*, vol. 7, no. 5, pp. 18–25, 2018.

[10] M. Galvani, "History and future of driver assistance," *IEEE Instrum. Meas. Mag.*, vol. 22, no. 1, pp. 11–16, 2019.

[11] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale convolutional networks," in *The 2011 international joint conference on neural networks*, 2011, pp. 2809–2813.

[12] Z. Malik and I. Siddiqi, "Detection and recognition of traffic signs from road scene images," in *2014 12th International conference on frontiers of information technology*, 2014, pp. 330–335.

[13] K. T. Islam, R. G. Raj, and G. Mujtaba, "Recognition of traffic sign based on bag-of-words and artificial neural network," *Symmetry*, vol. 9, no. 8, p. 138, 2017.

[14] R. Malik, J. Khurshid, and S. N. Ahmad, "Road sign detection and recognition using

colour segmentation, shape analysis and template matching," in *2007 international conference on machine learning and cybernetics*, 2007, vol. 6, pp. 3556–3560.

[15] J. Kim, S. Lee, T.-H. Oh, and I. S. Kweon, "Co-domain embedding using deep quadruplet networks for unseen traffic sign recognition," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018, vol. 32, no. 1.

[16] H.-Y. Lin, J.-M. Dai, L.-T. Wu, and L.-Q. Chen, "A vision-based driver assistance system with forward collision and overtaking detection," *Sensors*, vol. 20, no. 18, p. 5139, 2020.

[17] D. Fernández Llorca, A. Hernández Martínez, and I. García Daza, "Vision-based vehicle speed estimation: A survey," *IET Intell. Transp. Syst.*, vol. 15, no. 8, pp. 987–1005, 2021.

[18] J. A. Stark, "Adaptive image contrast enhancement using generalizations of histogram equalization," *IEEE Trans. Image Process.*, vol. 9, no. 5, pp. 889–896, 2000.

[19] J. Greenhalgh and M. Mirmehdi, "Real-time detection and recognition of road traffic signs," *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 4, pp. 1498–1506, 2012.

[20] R. Asgarian Dehkordi and H. Khosravi, "Vehicle type recognition based on dimension estimation and bag of word classification," *J. AI Data Min.*, vol. 8, no. 3, pp. 427–438, 2020.

[21] Y.-Y. Wu and C.-M. Tsai, "Pedestrian, bike, motorcycle, and vehicle classification via deep learning: Deep belief network and small training set," in *2016 International Conference on Applied System Innovation (ICASI)*, 2016, pp. 1–4.

[22] R. Carvalho Barbosa, M. Shoaib Ayub, R. Lopes Rosa, D. Zegarra Rodríguez, and L. Wuttisittikulkij, "Lightweight PVIDNet: A priority vehicles detection network model based on deep learning for intelligent traffic lights," *Sensors*, vol. 20, no. 21, p. 6218, 2020.

[23] S. Arabi, A. Haghighat, and A. Sharma, "A deep-learning-based computer vision solution for construction vehicle detection," *Comput. Civ. Infrastruct. Eng.*, vol. 35, no. 7, pp. 753–767, 2020.

[24] K. Bayoudh, F. Hamdaoui, and A. Mtibaa, "Transfer learning based hybrid 2D-3D CNN for traffic sign recognition and semantic road detection applied in advanced driver assistance systems," *Appl. Intell.*, vol. 51, no. 1, pp. 124–142, 2021.

[25] B. T. Nugraha and S.-F. Su, "Towards self-driving car using convolutional neural network and road lane detector," in *2017 2nd international conference on automation, cognitive science, optics, micro electro-mechanical system, and information technology (ICACOMIT)*, 2017, pp. 65–69.

[26] L. Deng and D. Yu, "Deep learning: methods and applications," *Found. Trends® Signal*

Process., vol. 7, no. 3–4, pp. 197–387, 2014.

[27] R. Dechter, "Learning while searching in constraint-satisfaction problems," 1986.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 25, 2012.

[29] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, 2014, pp. 818–833.

[30] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[31] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ArXiv Prepr. ArXiv14091556*, 2014.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[33] "Medium," *Medium*. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional- (accessed Jul. 29, 2022).

[34] J. Murphy, "An overview of convolutional neural network architectures for deep learning," *Microway Inc*, pp. 1–22, 2016.

[35] G. Jocher *et al.*, "Ultralytics/yolov5: V6. 0—YOLOv5n 'Nano'models, Roboflow integration, TensorFlow export, OpenCV DNN support," *Zenodo Tech Rep*, 2021.

[36] 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," *ArXiv Prepr. ArXiv220702696*, 2022.