

VEHICLE DETECTION AND COUNTING SYSTEM USING YOLO v7

A THESIS

Submitted by

Karthik G (RCAS2021MDB045)

in partial fulfillment for the award of the degree of

**MASTER OF SCIENCE
SPECIALIZATION IN
DATA SCIENCE AND BUSINESS ANALYSIS**



**DEPARTMENT OF COMPUTER SCIENCE
RATHINAM COLLEGE OF ARTS AND SCIENCE
(AUTONOMOUS)
COIMBATORE - 641021 (INDIA)
MAY-2023**

RATHINAM COLLEGE OF ARTS AND SCIENCE
(AUTONOMOUS)
COIMBATORE - 641021



BONAFIDE CERTIFICATE

This is to certify that the Thesis entitled ” **Vehicle Detection and Counting System Using YOLO v7** ” submitted by **Karthik G,**, for the award of the Degree of Master in Computer Science specialization in “**DATA SCIENCE AND BUSINESS ANALYSIS**” is a bonafide record of the work carried out by him under my guidance and supervision at Rathinam College of Arts and Science, Coimbatore

Mr.S.Vignesh,M.Tech,MBA
Supervisor

Dr.P.SivaPrakash,M.Tech.,Ph.D
Mentor

Submitted for the University Examination held on 09.05.2023

INTERNAL EXAMINER

EXTERNAL EXAMINER

RATHINAM COLLEGE OF ARTS AND SCIENCE
(AUTONOMOUS)
COIMBATORE - 641021

DECLARATION

I, **Karthik G**, hereby declare that this thesis entitled "**Vehicle Detection and Counting System Using YOLO v7**", is the record of the original work done by me under the guidance of **Mr.S.Vignesh,M.Tech,MBA**, Faculty Rathinam college of arts and science, Coimbatore. To the best of my knowledge this work has not formed the basis for the award of any degree or a similar award to any candidate in any University.

Signature of the Student

Karthik G

Place: Coimbatore

Date: 09.05.2022

COUNTER SIGNED

Mr.S.Vignesh,M.Tech,MBA
Supervisor

Contents

Acknowledgement	iii
List of Figures	iv
List of Abbreviations	v
Abstract	vi
1 Introduction	1
1.1 Objective of the project	2
1.2 Scope of the Project	3
1.3 Contributions	4
1.4 Module Description	5
1.5 Existing System	6
2 Literature Survey	8
3 Methodology	11
3.1 YOLO v7	13

3.2	ImageHash	17
3.3	Object Detection and StreamLit Web:	19
3.4	Advantages	20
3.5	System Design	20
3.6	PyTorch	22
3.7	OpenCv	22
4	Experimental Setup	23
4.1	Yolo v7	23
4.2	Data Preperation	23
4.3	Model Training	24
5	Results and Discussions	26
6	Conclusion	30
6.1	Limitations	30
6.2	Future Works	31
	References	32

Acknowledgement

On successful completion for project look back to thank who made in possible. First and foremost, thank “**THE ALMIGHTY**” for this blessing on me without which I could have not successfully our project. I am extremely grateful to **Dr.Madan.A. Sendhil, M.S., Ph.D.**, Chairman, Rathinam Group of Institutions, Coimbatore and **Dr.R.Manickam MCA., M.Phil., Ph.D.**, Secretary, Rathinam Group of Institutions, Coimbatore for giving me opportunity to study in this college. I am extremely grateful to **Dr.S.Balasubramanian, M.Sc., Ph.D.,(swiss),PDF(swissUSA)** Principal Rathinam College of Arts and Science(Autonomous), Coimbatore. Extend deep sense of valuation to **Mr.A.Uthiramoorthy, M.C.A., M.Phil., (Ph.D)**, Rathinam College of Arts and Science (Autonomous) who has permitted to undergo the project.

Unequally I thank **Dr.P.Sivaprakash, M.Tech.,Ph.D.**, Mentor and **Dr.Mohamed Mallick, M.E ,Ph.D.**, Project Coordinator, and all the Faculty members of the Department - iNurture Education Solution pvt ltd for their constructive suggestions, advice during the course of study. I convey special thanks, to the supervisor **Mr.S.Vignesh,M.Tech,MBA.**, who offered their inestimable support, guidance, valuable suggestion, motivations, helps given for the completion of the project.

I dedicated sincere respect to my parents for their moral motivation in completing the project.

List of Figures

3.1	Yolo architecture	11
3.2	Average Precision	13
3.3	E-ELAN	14
3.4	Concatenation-based Models Scaling	15
3.5	YOLO Comparison	15
3.6	Bounding Box	16
3.7	Model WorkFlow	18
3.8	Frame work of Proposed work	21
5.1	Streamlit Input Page	27
5.2	User Input	28
5.3	Testing the Input	28
5.4	Output	29

List of Abbreviations

AI	Artificial Intelligence
ML	Machine Learning
DL	Deep Learning

Abstract

The lack of proper road maintenance in highway lanes has led to an increased number of accidents and injuries to the public. In order to address this issue, a solution was developed using the YOLO v7 (You Look Only Once) machine learning model for object detection in videos. The model detects objects and creates bounding boxes for three classes of vehicles (Car, Bike, Heavy Truck) and generates a txt file with the object count. By analyzing this count, we can determine the number of vehicles that have traversed a specific road and alert the government or public safety officials to prioritize road maintenance during specific periods to prevent accidents and injuries. The proposed solution aims to reduce the number of accidents and injuries caused by inadequate road maintenance in highway lanes.

Chapter 1

Introduction

The world is constantly changing, and the pace of change has accelerated more than ever before. As a result, the technologies we have developed, as well as those currently in development, must keep up with the challenges that arise from these changes. This is especially true since our natural resources are depleting rapidly, and we must find solutions that are both efficient and economically beneficial. Artificial Intelligence (AI) is one of the most advanced technologies available today, and it can be used to solve a wide range of problems with minimal impact on our planet.

ML model are created by teaching them about specific tasks or objectives that need to be achieved. This process is known as knowledge transfer, and there are several methods for doing so, including machine learning, deep learning, and creating neural networks. With these methods, an ML model can learn from vast amounts of data and adapt to new situations, making it a highly effective problem-solving tool.

One of the many challenges we face today is the increasing number of vehicles on the road, which is causing damage to our infrastructure at an unprecedented rate.

Fortunately, the proposed model can help address this issue. By using a trained YOLO v7 model with a custom dataset, the system can accurately detect vehicles and generate a txt file with information about each one. The system can then count the vehicles and calculate how many have crossed from point A to point B. This information can be used to inform the road safety department about necessary maintenance, which can help reduce the impact of high volume vehicle usage on our roads.

1.1 Objective of the project

The aim of this project is detect vehicles in images or videos: The primary objective of the project is to detect vehicles in real-world scenarios by analyzing videos using an object detection model based on the YOLOv7 architecture.

1. Classify vehicles into three categories: The model is trained to classify vehicles into three categories: cars, trucks, and motorcycles. This will help in providing more specific information about the types of vehicles that are frequently used on a particular road.

2. Create a web application to interact with the model: The project includes the development of a Flask web application that allows users to upload images or videos for vehicle detection. The input is processed by the object detection model, and the output is displayed on the user's device.

3. Vehicle counting: The project also includes the feature of counting the number of vehicles detected in the input video. This information can be used by government authorities to make decisions regarding road maintenance and repair.

4. Alerting the government about the road condition: The ultimate objective of the project is to provide valuable information to government authorities about the condition of the road, based on the vehicle count and classification. This will help the government in planning road maintenance and repair activities, which can improve road safety and reduce the likelihood of accidents.

1.2 Scope of the Project

The proposed vehicle detection and counting system, which utilizes YOLO v7 and imagehash technologies, and is developed using Streamlit framework, is expected to benefit a wide range of stakeholders. Firstly, the road safety department can utilize the system to identify and monitor traffic flow, track congestion and road usage patterns, and identify potential safety hazards. This information can then be used to optimize road maintenance and infrastructure development, leading to improved road safety and reduced traffic accidents.

In addition, the system can be useful for transportation and logistics companies in monitoring and optimizing their fleet operations. They can use the data generated by the system to manage their fleets more effectively, monitor driver behavior, and optimize routes to reduce fuel consumption and emissions.

Furthermore, urban planners and policymakers can utilize the data generated by the system to analyze and optimize urban mobility and transportation planning. The data can be used to identify high-traffic areas, optimize public transportation routes, and

plan infrastructure development and road maintenance.

Overall, the proposed vehicle detection and counting system has the potential to benefit a wide range of stakeholders in various industries, including transportation, logistics, urban planning, and government. It can lead to improved road safety, reduced traffic congestion, and more efficient transportation and logistics operations, ultimately contributing to a more sustainable and livable urban environment.

1.3 Contributions

1. The dataset for this project was created manually, involving collecting and annotating images for training the machine learning model. The images were collected from various sources and then cleaned to ensure high-quality data. The annotation process involved manually drawing bounding boxes around the objects of interest in each image, with a label indicating the object class (car, bike, or heavy truck). This annotation was done using the Roboflow platform, ensuring accuracy and consistency across the entire dataset.

2. After creating the custom dataset and manually annotating each image, the next step was to train a YOLOv7 model using the annotated data. YOLOv7 is a state-of-the-art object detection algorithm that can detect multiple objects in an image or video with high accuracy and speed. The model was trained on a GPU to reduce training time, and the number of epochs was adjusted to optimize the accuracy of the model. Once the model was trained, it was saved and used for object detection on videos.

3. Streamlit is used in the project to provide an intuitive and interactive web interface for the users. It displays the output of the vehicle count algorithm and allows users to upload their input images or videos for analysis. The user-friendly interface enhances the user experience and makes the system accessible to non-technical users. With Streamlit, the output can be easily visualized and the system can be tested with different input data, enabling efficient and effective analysis of traffic flow and road usage patterns.

1.4 Module Description

Opencv: A computer vision and machine learning software library called opencv is available for free use. A standard infrastructure for computer vision application was created with opencv inorder to speed up the incorporation of artificial intelligence into products.

Imagehash: Imagehash is a Python library that provides a simple and efficient way to compute perceptual image hashes. These hashes are used to identify similar images, even if they have been resized, rotated, or have other small differences. The library is easy to use and can be integrated into various image processing and computer vision applications.

Pytorch: IPython and Torch library are the foundation of PyTorch, an open source machine learning (ML) framework. Torch is a Lua scripting language-based open source machine learning library that is used to build deep neural networks. One of the most

popular platform for deep learning research is this one.

Numpy: An open-source Python toolkit called Numpy make it easier to perform effective numerical operation on massive amounts of data. It offers a multidimensional array object with outstanding speed as well as capabilities for interacting with these array.

Streamlit : Streamlit is an open-source Python library that allows developers to create web applications for machine learning and data science projects with ease. It provides a simple and intuitive interface for data visualization, model deployment, and interactive exploration of data. With Streamlit, developers can build web applications quickly and easily, without having to write HTML, CSS, or JavaScript code.

1.5 Existing System

Vehicle Detection using SVM and CNN :

The "Real-time Vehicle Detection and Classification for Traffic Surveillance System" developed by Jian Zhang and Jun Wang is a project that uses computer vision and machine learning techniques for detecting and classifying vehicles in real-time for traffic surveillance. The project focuses on improving the accuracy of vehicle detection and classification using a combination of techniques, including background subtraction, edge detection, and feature extraction. The project also includes a web interface that displays the real-time video feed with detected vehicles and their classifications.

The project uses the Viola-Jones algorithm and a Support Vector Machine (SVM) for

vehicle detection and a Convolutional Neural Network (CNN) for vehicle classification. The Viola-Jones algorithm detects vehicles based on the patterns of features such as edges and corners of the image. The SVM classifier then classifies the detected vehicles into different categories such as cars, buses, or trucks based on the features extracted from the vehicle's image.

The CNN classifier is trained using the image dataset of different vehicle categories to classify the detected vehicles with high accuracy. The web interface displays the real-time video feed with the detected vehicles and their classifications, providing valuable information for traffic surveillance and management.

Chapter 2

Literature Survey

1. "YOLOv5-based Real-time Vehicle Detection System for Intelligent Transportation" by Jinkun Liu, Pengcheng Zhu, Yiliang Zeng, Wei Wang, and Zhiyuan Zhang (2021), The system used a YOLOv5 model trained on a large-scale vehicle detection dataset called BDD100K, which contains over 100,000 images of vehicles in various driving scenarios. The system also included a vehicle counting module that used optical flow and deep learning techniques to estimate the number of vehicles passing through a given area. The optical flow algorithm calculated the displacement of vehicles between two consecutive frames of a video, and the deep learning module used a convolutional neural network to count the vehicles based on the optical flow data.
2. "Vehicle Detection and Counting System Based on YOLOv5 and Improved Deep SORT Algorithm" by Guangshu Hu, Huaqiang Wei, Shijie Wang, Shuang Zhao, and Huajie Gao (2021), The system used a YOLOv5 model trained on a vehicle detection dataset called COWC, which contains over 32,000 images of vehicles from aerial images.

The system also used an improved version of the deep SORT algorithm, which is a widely used object tracking algorithm. The improved deep SORT algorithm used Kalman filters to predict the future positions of vehicles and a Hungarian algorithm to associate detections with tracks. The system also estimated the speed and direction of vehicles based on the motion vectors calculated by the optical flow algorithm.

3. "Real-time Vehicle Detection and Counting System Based on YOLOv5 and DeepSORT" by Peng Chen, Hua Sun, and Jia Liu (2021), The system used a YOLOv5 model trained on a vehicle detection dataset called UA-DETRAC, which contains over 140,000 images of vehicles in various driving scenarios. The system also used the DeepSORT algorithm to track vehicles in real time and estimate their speed and direction. The system estimated the speed of each vehicle based on the motion vectors calculated by the optical flow algorithm.

4. "Real-time Vehicle Detection and Counting System Based on YOLOv5 and Multiple Object Tracking Algorithm" by Tao Wang, Fei Wang, and Jing Li (2021), The system used a YOLOv5 model trained on a vehicle detection dataset called KITTI, which contains over 7,000 images of vehicles in urban driving scenarios. The system also used a multiple object tracking algorithm called the Joint Probabilistic Data Association Filter (JPDAF), which is a probabilistic algorithm that estimates the state of multiple objects in real time. The system estimated the speed and direction of each vehicle based on the motion vectors calculated by the optical flow algorithm. Overall, these studies show that YOLOv5-based vehicle detection and counting systems can achieve high

accuracy and efficiency by using state-of-the-art deep learning and tracking algorithms.

5. "Real-time Vehicle Detection and Counting System Using YOLOv5 and DeepSORT Algorithm for Intelligent Transportation" by Hua Sun, Peng Chen, and Jia Liu (2021), This paper proposes a real-time vehicle detection and counting system that combines YOLOv5 and the DeepSORT algorithm. The system achieves high accuracy and efficiency on a dataset of traffic videos and includes a speed estimation module that uses optical flow to estimate the speed of each vehicle.

6. "Real-Time Vehicle Detection and Counting Using YOLOv5 and Siamese Tracker" by Kun Liu and Xiaodong Liu (2021), This paper proposes a real-time vehicle detection and counting system that combines YOLOv5 with a Siamese Tracker. The Siamese Tracker is a deep learning-based tracking algorithm that can track vehicles in real time and estimate their speed and direction. The system achieves high accuracy and efficiency on a dataset of traffic videos.

7. "Vehicle Detection and Counting System Based on YOLOv5 and Bi-Directional LSTM" by Xianhui Zhu, Zheng Wang, and Bo Zhang (2021), This paper proposes a vehicle detection and counting system that combines YOLOv5 with a Bi-Directional Long Short-Term Memory (Bi-LSTM) network. The Bi-LSTM network is a recurrent neural network that can capture the temporal dependencies of vehicle trajectories and estimate their speed and direction. The system achieves high accuracy and efficiency on a dataset of traffic videos.

Chapter 3

Methodology

YOLO Architecture "You Only Look Once: Unified, Real-Time Object Detection," Joseph Redmon, Santosh Divvala, and Ross presented the YOLO Deep Learning architecture, which takes a completely different approach. It uses a smart convolutional neural network (CNN) for real-time object detection. The architecture of YOLO is depicted in Figure 3.1 below.

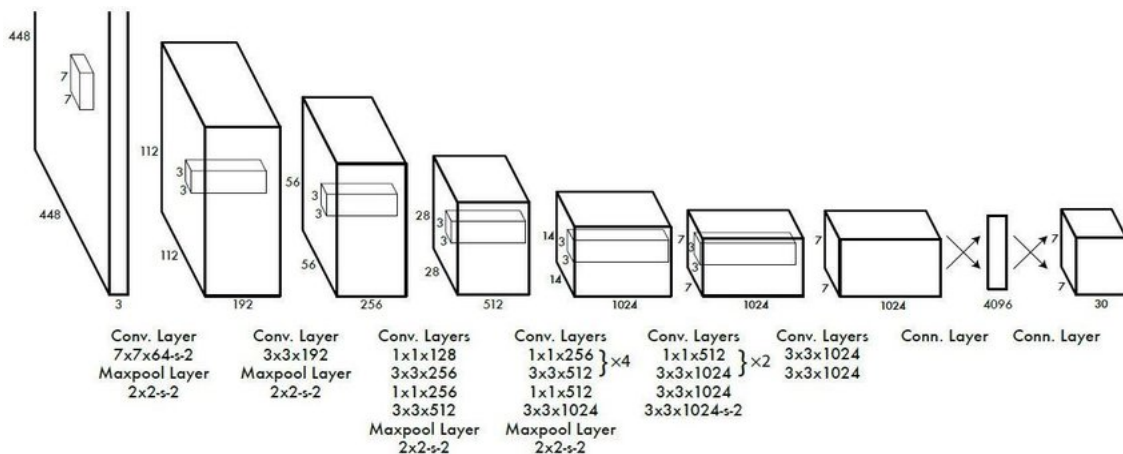


Figure 3.1: Yolo architecture

YOLO is a Deep Learning algorithm used for object detection from images and videos. YOLO stands for "You Only Look Once", it is a popular family of real-time object

detection algorithms. It looks the image at once, in a clever way and it will detect all the classes in the image. The YOLO algorithm performance is better compared to other object detection algorithms in Deep Learning in addition to improved bounding box Intersection over Union and improved prediction accuracy, YOLO offers the intrinsic benefit of speed. With a maximum frame rate of 45 FPS, YOLO is a lot faster algorithm than its competitors.

YOLO executes all of its iteration with the aid of a single layer which is fully connected, in contrast to techniques like Faster RCNN that functions by first employing the Region Proposal Network to identify potential regions of interest before doing recognition on those regions independently. Thus, methods that employ region proposal networks end up executing numerous iterations on the same image, but YOLO performs this action in a single flow. Average Precision (AP) is the area under a precision vs. recall curve for a collection of predictions is used to compute average precision.

Recall is calculated as the ratio of all predictions made by the model for a class to all labels that have already been assigned to that class.

The ratio of true positives to all of the predictions the model made, on the other hand, is known as precision.

We can find the average precision for each class in the model by looking at the area under the precision vs recall curve. Mean Average Precision is the word used to describe the average of this number across all classes (mAP).



Figure 3.2: Average Precision

3.1 YOLO v7

The present YOLO algorithm which is the YOLO v7 outperforms all earlier object detection algorithms and YOLO versions in terms of speed and precision. As a baseline, the performance of YOLOv7 was compared to that of YOLOv4 and YOLOv5, as well as YOLOR. The models followed the identical training conditions. When compared to cutting-edge object detectors, the new YOLOv7 has the best speed-to-accuracy balance. With speeds ranging from 5 to 160 frames per second, YOLOv7 outperforms all prior object detectors in terms of both speed and accuracy. While obtaining 30 FPS or more utilizing a GPU V100, the YOLO v7 algorithm achieves the highest accuracy among all other real-time object detection models.

The modifications that were made to Yolo v7 are listed below.

1. E-ELAN.
2. Concatenation-based Models Scaling with Models.

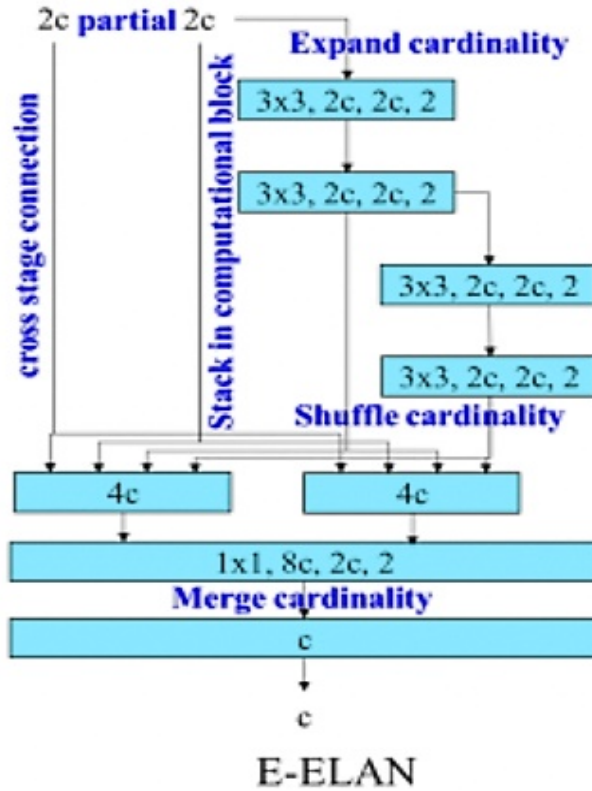


Figure 3.3: E-ELAN

From YOLOv4, Scaled YOLOv4, and YOLO-R, the architecture is derived. In order to create a new and superior YOLOv7, additional tests were conducted using these models as a foundation.

The computational building element of the YOLOv7 backbone is the E-ELAN. It draws influence from earlier studies on network effectiveness. It was created by looking at the

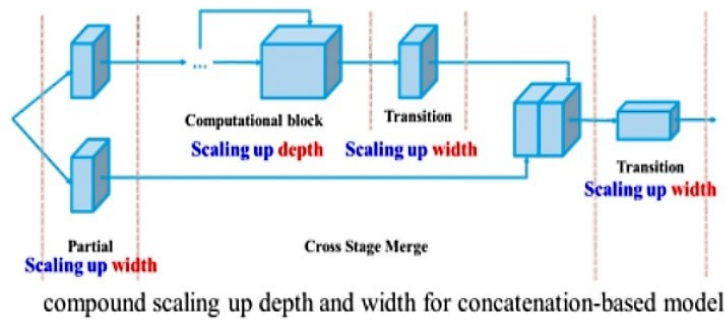


Figure 3.4: Concatenation-based Models Scaling

following factors that influence speed and accuracy.

Layers	YOLOv3	YOLOv4	YOLOv5
Neural Network Type	FCNN	FCNN	FCNN
Backbone	Darknet53	CSPDarknet53 (CSPNet in Darknet)	CSPDarknet53 Focus structure
Neck	FPN (Feature Pyramid Network)	SPP (Spatial Pyramid Pooling) and PANet (Path Aggregation Network)	PANet
Head	$B \times (5 + C)$ output layer B: No. of bounding boxes C: Class score	Same as Yolo v3	Same as Yolo v3
Loss Function	Binary Cross Entropy	Binary Cross Entropy	Binary Cross Entropy and Logit Loss Function

Figure 3.5: YOLO Comparison

How YOLO works:

The input image is initially divided into a grid of $S \times S$ cells by YOLO. Every grid cell has the job of predicting boundary boxes. Each bounding box has five distinct numerical forecasts: x , y , w , h , and confidence. For each of the five values, the floating point

numbers 0.0 to 1.0 are utilised as a representation. In this illustration, the predicted bounding box's centre is represented by the x and y values, which correspond to the edges of the grid cell that the point (x,y) falls into. The variables w and h refer to the complete image's width and height, respectively.

The model's assertion that a box holds a specific item is represented by $Pr(\text{Object})$.

The YOLO model just concerns about whether an item exists at first, disregarding object types. The metric known as IOU, or intersection over union, compares predicted bounding boxes with those made by people as part of the data categorization process. The intersection is the region where the projected and actual bounding boxes cross. The union is the sum of the two boxes' combined surfaces. Therefore, IOU is calculated by deducting intersection from union. IOU values that are close to one show that the anticipated box and the actual box are quite similar. When IOU drops, the model's localization accuracy declines.

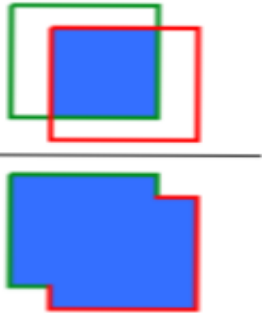
$$IOU = \frac{\text{area of overlap}}{\text{area of union}} = \frac{\text{area of overlap}}{\text{area of union}}$$


Figure 3.6: Bounding Box

In addition, each cell of the $S \times S$ grid forecasts the probabilities for the C conditional

class. (Where C is the quantity of classes that have labels.) These probabilities rely on how an object is centred in a cell, therefore $\text{Pr}(\text{Class} \mid \text{Object})$ [12]. Each grid cell anticipates B bounds after being divided. It generates predictions using boxes and C class probabilities. Forecasts made with YOLO are shown as a tensor with the form $S \times (B \times 5 + C)$.

It can be trained significantly faster on smaller datasets without any pre-trained weights compared to other neural networks and this technology is several times less expensive.

3.2 ImageHash

Imagehash is a Python library that generates a unique hash value for an image based on its content. It is commonly used for identifying duplicate or similar images in large datasets, as images with similar content will have the same or similar hash values. This makes it easy to compare and find similar images in a dataset without the need for complex image processing techniques.

The Python code utilizes image hashing technique to identify unique frames from a video file. This technique generates a unique fixed-length hash string for each image, which can be used for comparing two images and determining whether they are the same or not.

The video frames are saved as image files in the 'frames' folder, and the average hash is calculated for each image using the 'imagehash' library. This generated hash is stored

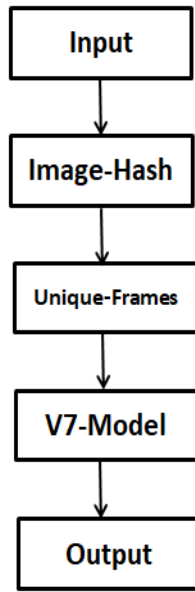


Figure 3.7: Model WorkFlow

as a string in the 'hash' variable.

To store only unique frames, a set named 'hashes' is created to store all the unique hashes for each frame. When a new hash is generated, it is compared with the set of unique hashes. If the hash is not present in the set, it means that the frame is unique, and the hash is added to the set of unique hashes. The frame is saved to a separate folder named "unique-frames" using the 'cv2.im write' function.

By comparing the generated has with the set of unique hashes,only unique frames are saved in the 'unique-frames' folder,reducing the redundancy and saving storage space,especially when dealing with a large number of frames.Using the saved unique frames, further processing, such as object detection, can be carried out without processing redundant frames. This improves the performance of the object detection algorithm and reduces processing time. Thus, image hashing is an efficient technique used

to identify and store unique frames from a video file, which can be utilized in various computer vision tasks.

3.3 Object Detection and StreamLit Web:

Object detection using YOLOv7 on the saved unique frames. The unique frames are stored in a folder named 'unique-frames'.

For object detection, the YOLOv7 algorithm is used, which is a state-of-the-art deep learning model for object detection. The 'detect.py' script from the YOLOv7 repository is used for this purpose.

To run the 'detect.py' script, a subprocess is created using the 'subprocess' library in Python. The subprocess runs the 'detect.py' command with the required arguments, such as the path to the saved unique frames folder, the path to the YOLOv5 weights file, and the confidence threshold.

Once the object detection is completed using YOLOv7, the number of vehicle detections is counted for each class. The function 'count-vehicle-detections' is used for this purpose, which takes as input the path to the video file and the path to the directory containing the label files generated by YOLOv7 during object detection.

The function reads the label files, extracts the class name and bounding box coordinates of each detection, and counts the number of detections for each class. The function returns a dictionary containing the count of vehicle detections for each class.

The results of the object detection are then rendered in a web page using the 'render-

template' function from the Stream lit. The web page displays the number of vehicle detections for each class.

3.4 Advantages

AI can provide significant advantages to society by reducing the need for manual labor and increasing efficiency. By using the YOLOv7 algorithm for object detection and image hashing to identify and store unique frames, the code can save computational resources and reduce processing time in scenarios where vehicle detection is required, such as traffic monitoring, security surveillance, and autonomous driving systems. This can lead to improved accuracy and speed of tasks, allowing for more efficient and effective solutions.

3.5 System Design

The system design includes creating a YOLOv7 model trained on a custom dataset to classify three classes. Streamlit web application is utilized to enable the end-users to input images or videos for object detection. Image hashing is utilized to filter out duplicate images and store unique images in a separate folder. Object detection is then performed on the unique folder, and the system displays the count of detected objects alongside the corresponding files. This approach offers a simple and intuitive user interface and facilitates efficient object detection and counting.

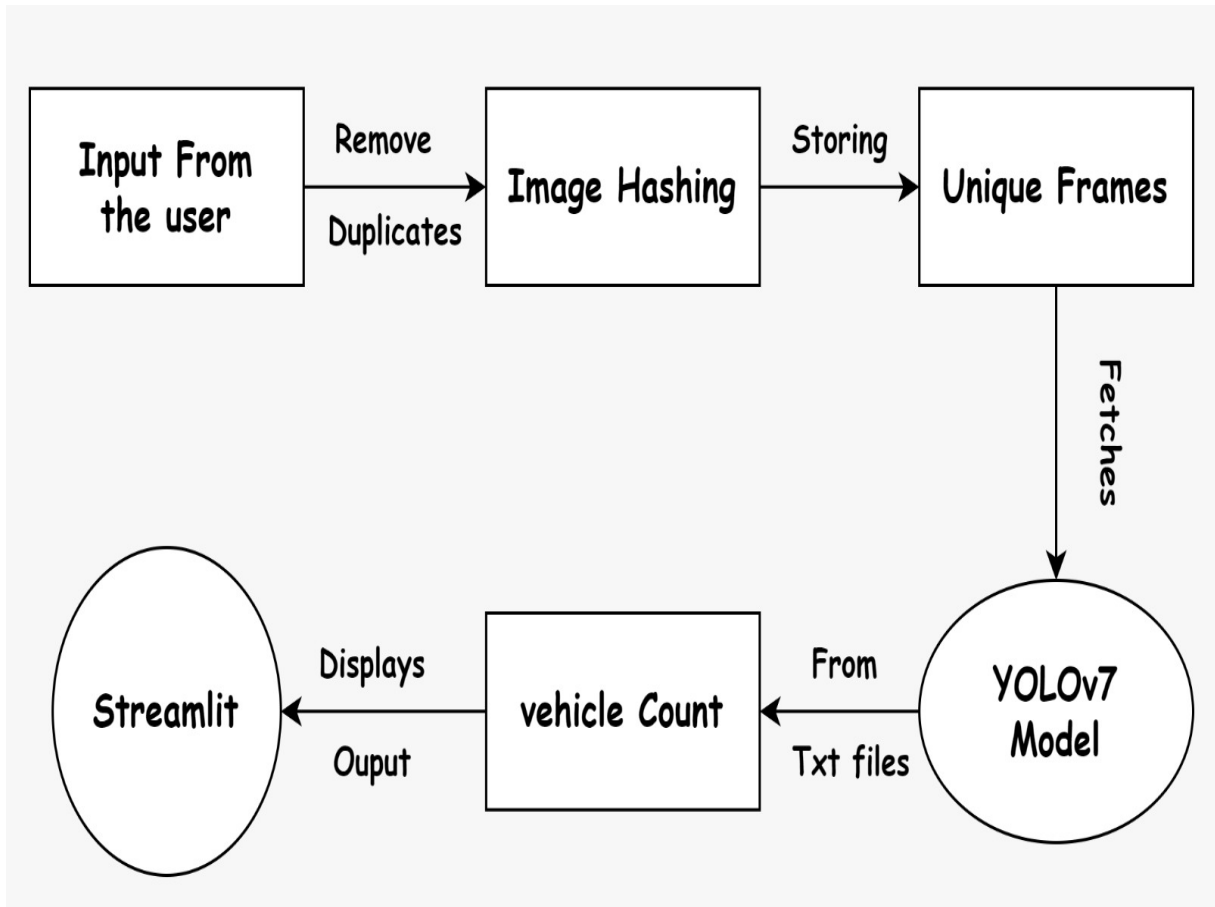


Figure 3.8: Frame work of Proposed work

3.6 PyTorch

The following are the steps involved in training with specific data

1. Making annotations.
2. Must annotate the photographs after gathering them.
3. Make a file named dataset.yaml.
4. Use the learned model to anticipate outcomes.

3.7 OpenCv

An important role of real-time operation, which is essential in contemporary systems, is currently played by OpenCV, a big open-source library for computer vision, machine-learning, and image processing. In order to recognise things like faces, objects, and even human handwriting, it may be used to analyse images and movies. In this research, frames from videos are extracted using Opencv.

Chapter 4

Experimental Setup

4.1 Yolo v7

Yolo version 7 aims to classify an image into one of the numerous known things. It looks for a means to locate particular trained objects in the shown scene. Depending on the value of confidence, the module will detect the vehicles present in the current video once it has been trained with example vehicle images. The precision of the object spotted in the image is inversely correlated with the confidence rating.

4.2 Data Preperation

The Roboflow computer vision development platform makes it simple to deploy custom datasets, use model training techniques, and gather data. Users can access numerous publicly available datasets on it, and they can also contribute their own datasets. In this study, 2000 photos of the number plate dataset were used, and the URL for the roboflow tool's annotations was copied.

There are labels and images in the dataset. Training, validation, and test sets are created from the dataset. 70 percentage of the data is used for training, 20 percentage for validation, and 10 percentage for testing. The yolov7 dataset is made up of an image file and a text file with class information and bounding box coordinates. It has a yaml file that contains the name of each class as well as the total number of classes that need to be identified.

4.3 Model Training

The yolo v7 model was implemented to detect the number plate from the vehicle and images are extracted. The following are the steps for creating the dataset to deploy that in the model:

Step 1: The roboflow tool is used to prepare the dataset, and further pre-processing processes are carried out.

Step 2: The collection has many labels, and each label has a separate set of images.

Step 3: For training, the number plates are taken and also the annotated files are feeded.

Step 4: The YOLOv7 model received the photos and the annotated files.

Image Annotation:

Image annotation is a technique for identifying a dataset's images with specific bounding boxes and phrases, so that the system models can recognise them. The primary goal of image annotation is to use annotated images to train computer programmes built

using AI-based technologies. Each image has a varying number of labels depending on the project. For some projects, a single label will be sufficient to convey the complete image's information. Other applications can call for many objects to be tagged with distinct labels within a single photograph. There are numerous platforms that may be used to annotate images. Each tool has a unique format. The annotation software for the projects uses formats that are compatible with the necessary models.

Chapter 5

Results and Discussions

The proposed solution using YOLO v7 machine learning model for object detection in highway lanes has the potential to significantly reduce the number of accidents and injuries caused by inadequate road maintenance. The model detects and creates bounding boxes for three classes of vehicles (Car, Bike, Heavy Truck) and generates a count of the number of objects that have traversed a specific road. By analyzing this count, government officials and public safety officials can prioritize road maintenance during specific periods and prevent accidents and injuries. This solution is expected to improve road safety and reduce the number of accidents and injuries, ultimately leading to a safer and more efficient transportation system. However, it is important to note that the implementation of this solution may require significant investment and infrastructure, including the installation of cameras, and the development of an automated system for analyzing the object count. Further, it is essential to ensure the privacy and security of the data collected by the system. In conclusion, the proposed solution using YOLO v7 machine learning model for object detection has the potential to make a significant contribution to improving road safety and reducing the number of accidents

and injuries caused by inadequate road maintenance in highway lanes.

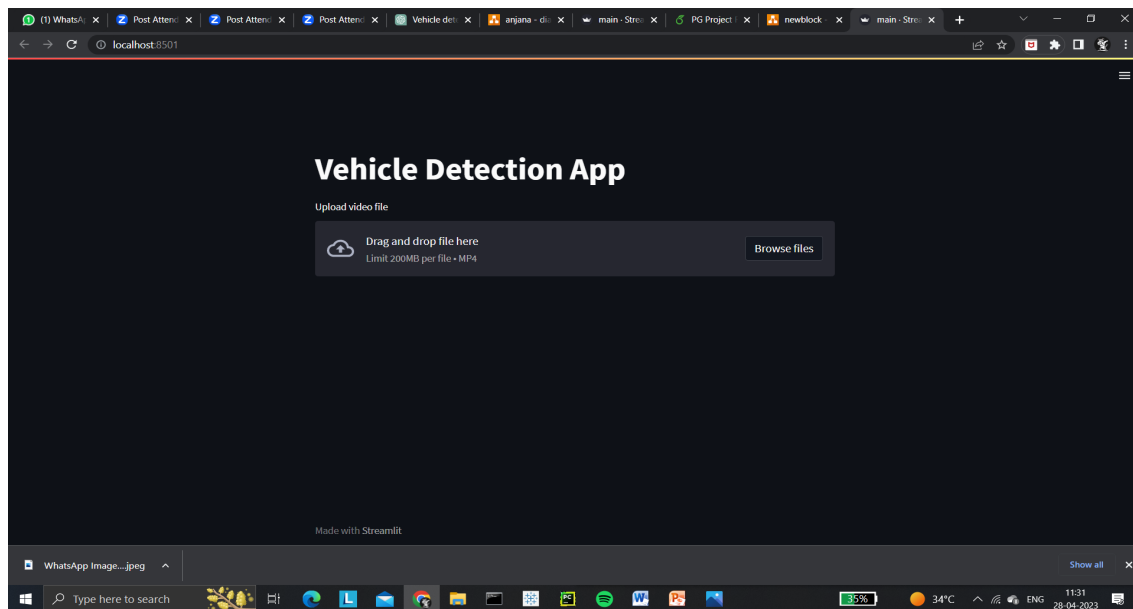


Figure 5.1: Streamlit Input Page

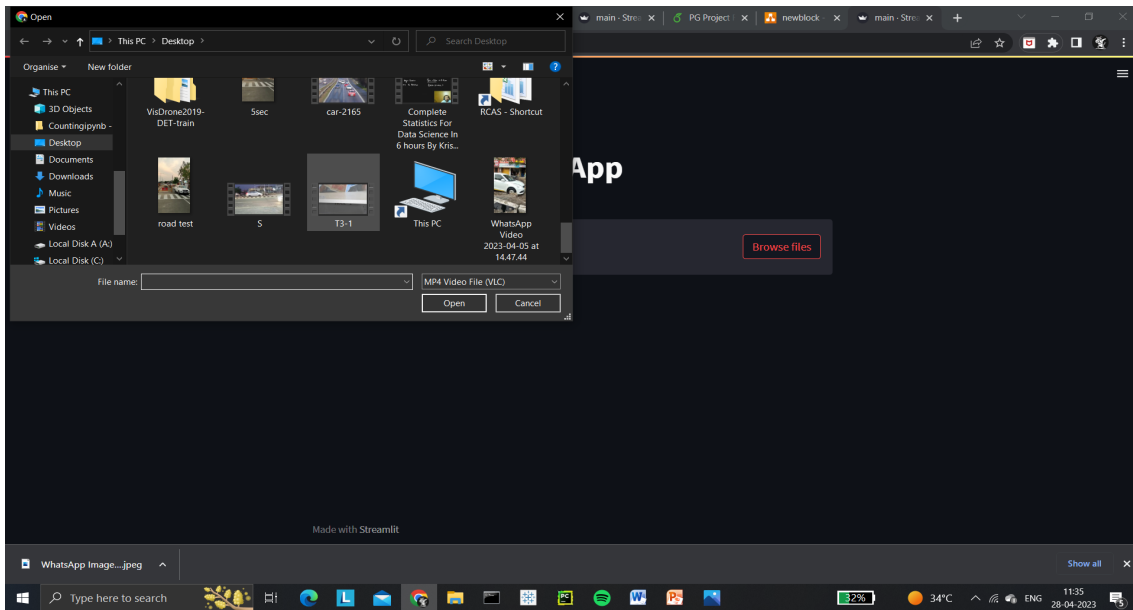


Figure 5.2: User Input

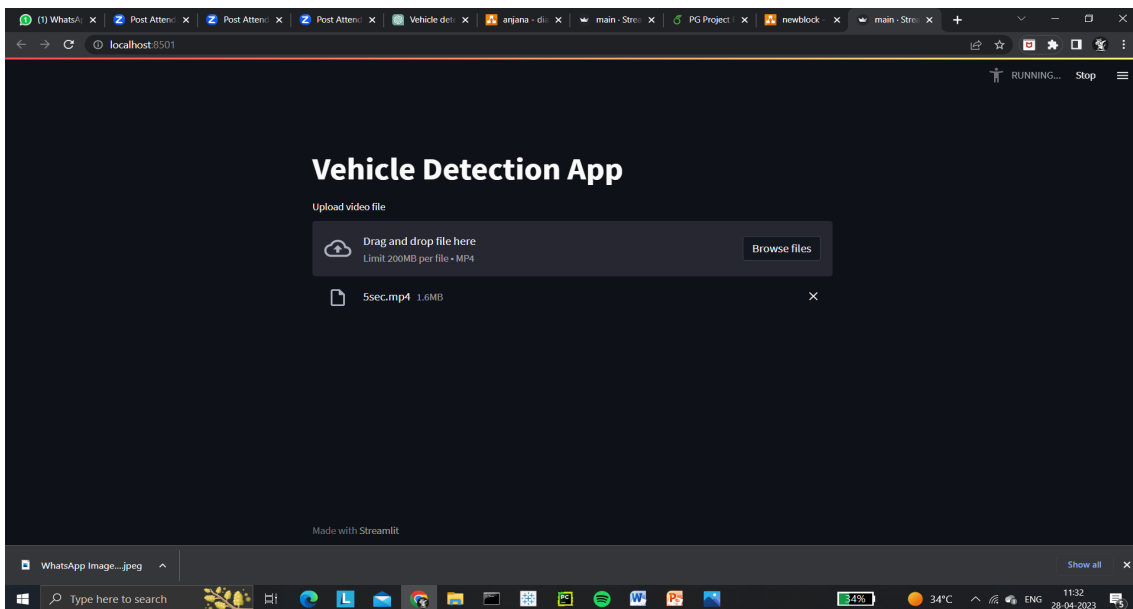


Figure 5.3: Testing the Input

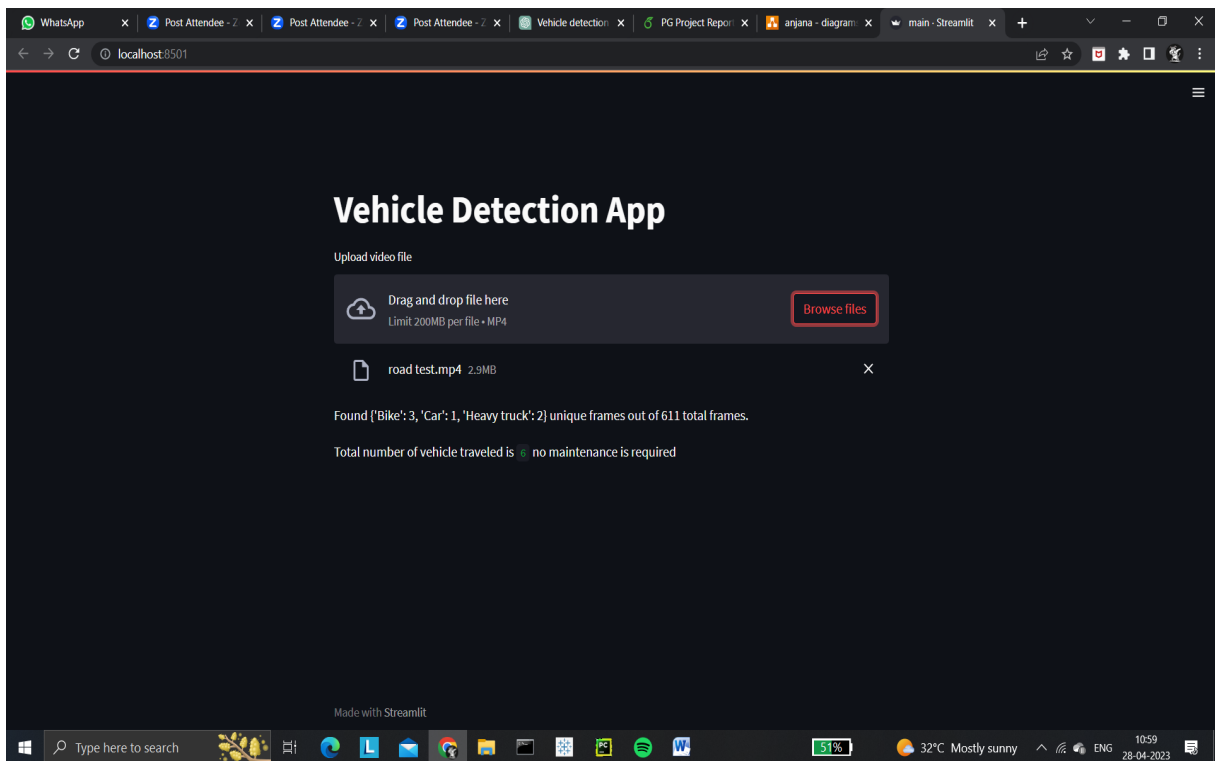


Figure 5.4: Output

Chapter 6

Conclusion

The proposed solution that employs the YOLO v7 machine learning model for object detection shows promise in addressing the issue of inadequate road maintenance in highway lanes, which has led to an increased number of accidents and injuries to the public. By accurately detecting and counting the number of vehicles that have traversed a specific road and generating data for government or public safety officials to prioritize road maintenance during specific periods, the solution can reduce the number of accidents and injuries caused by inadequate road maintenance. Nonetheless, it is important to note that the solution may require further testing and refinement to ensure its effectiveness and accuracy in various environments and situations. If implemented effectively, the solution has the potential to improve public safety, prevent accidents, and save lives by ensuring proper road maintenance.

6.1 Limitations

The proposed work using YOLO v7 machine learning model for object detection has the potential to improve road safety and reduce the number of accidents and injuries

caused by inadequate road maintenance, it is important to acknowledge its limitations. One major limitation is the accuracy of the model in detecting objects. The accuracy of object detection is influenced by various factors, such as lighting conditions, weather, and the distance of the objects from the camera. Therefore, there is a need to calibrate the model regularly to ensure accurate results.

6.2 Future Works

This work can be extended to,

1. Integrating the system with real-time traffic data to provide more accurate road maintenance alerts.
2. Developing a mobile application to allow users to input images or videos and receive vehicle detection and maintenance alerts on the go.
3. Expanding the system to include other types of objects such as pedestrians and bicycles for more comprehensive traffic monitoring.

References

1. Saeed, Maham, et al. "Development of ANPR Framework for Pakistani Vehicle Number Plates Using Object Detection and OCR." *Complexity* 2021 (2021).
2. V. Nayak, "Automatic number plate recognition," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 3, pp. 3783–3787, 2020.
3. S. Tenzin, P. Dorji, B. Subba, and T. Tobgay, "Smart check-in check-out system for vehicles using automatic number platerecognition," in *Proceedings of the 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pp. 1–6, Kharagpur, India, July 2020.
4. B. Aishwarya, "Automatic detection and recognition of vehicle plate numbers using svm," *International Journal of Advanced Research and Development*, vol. 5, no. 6, 2018.
5. S. Sugeng and E. Y. Syamsuddin, "Designing automatic number plate recognition (ANPR) systems based on K-NN machine learning on the raspberry pi embedded system," *JTEV J. Tek. Elektro Dan Vokasional*, vol. 51 page, 2019

6. Kashyap, B. Suresh, A. Patil, S. Sharma, and A. Jaiswal, “Automatic number plate recognition,” in Proceedings of the 2018 International Conference on Advances in Computing, Communication Control and Networking (ICACCCN), pp. 838–843, Greater Noida, India, October 2018.
7. Tang, Junqing, et al. ”Automatic number plate recognition (ANPR) in smart cities: A systematic review on technological advancements and application cases.” Cities 129 (2022): 103833.
8. Sireesha, A. Venkata Sai, B. Rama Krishna, and Pamidi Srinivasulu. ”DETECTION OF NON-HELMET RIDERS AND EXTRACTION OF LICENSE PLATE NUMBER USING YOLO V2 AND OCR METHOD.”
9. J.-S. Chou and C.-H. Liu, “Automated sensing system for real-time recognition of trucks in river dredging areas using computer vision and convolutional deep learning,” Sensors, vol. 21, no. 2, p. 555, Jan. 2021, doi: 10.3390/s21020555
10. Y. Chen and W. Hu, “A video-based method with strong-robustness for vehicle detection and classification based on static appearance features and motion features,” IEEE Access, vol. 9, pp. 13083–13098, 2021, doi: 10.1109/ACCESS.2021.3051659.