VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(18CSP83)** ON

"Car Crash Detection System"

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

Prateek K Srivastav(1CR20CS141)

Manjil KC(1CR20CS110)

Riken Chauguthi(1CR20CS158)

Under the Guidance of,

Prof. Lynsha Helena Pratheeba HP

Assistant Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"Car Crash Detection System"** carried out by **Mr. Prateek K Srivastav**, USN **1CR20CS141**, **Mr. Manjil KC**, USN **1CR20CS110**, **Mr. Riken Chauguthi**, USN **1CR20CS158,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2023-2024. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

.

Prof. Lynsha Helena Pratheeba HP **Assistant Professor** **Dept. of CSE, CMRIT**	**Dr. Kavitha P** **Professor & Head** **Dept. of CSE, CMRIT**	**Dr. Sanjay Jain** **Principal** **CMRIT**

External Viva

Name of the Examiners   Signature with Date

1. ______________________________   ___________________________

2. ______________________________   ___________________________

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled " **Car Crash Detection System** " has been successfully completed under the guidance of **Prof. Lynsha Helena Pratheeba HP** Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2023 - 2024. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bangalore

Date:

Team members:	**Signature**
Prateek K Srivastav(1CR20CS141)	___________________
Manjil KC (1CR20CS110)	___________________
Riken Chauguthi(1CR20CS158)	___________________

ABSTRACT

Study presents a robust crash detection system integrating sensor data analysis and real-time alerting mechanisms. Leveraging machine learning algorithms and sensor fusion techniques, the system accurately identifies crash events, triggering immediate alerts to emergency services and nearby vehicles. Preprocessed sensor data, including vehicle dynamics and environmental factors, enhances detection accuracy. Validation on real-world datasets demonstrates high precision and recall rates. The system offers potential to significantly improve road safety and emergency response times, paving the way for proactive accident prevention measures in the automotive industry.

ACKNOWLEDGEMENT

I take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me a platform to pursue my studies and carry out my final year project

I have a great pleasure in expressing my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

I would like to thank **Dr. Kavitha P,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

I consider it a privilege and honor to express my sincere gratitude to my guide **Prof. Lynsha Helena Pratheeba HP, Assistant Professor,** Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

I also extend my thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, I would like to thank my parents and friends for all their moral support they have given me during the completion of this work.

TABLE OF CONTENTS.

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	**Convolutional Neural Network**
CUDA	**Compute Unified Device Architecture**
GAN	**Generative Adversarial Network**
GSM	**Global System for Mobile communication**
GPS	**Global Positioning System**
GPU	**Graphical Processing Unit**
GTX	**Giga Texel Shader eXtreme**
GUI	**Graphical User Interface**
OBD	**Onboard diagnostics**
SSD	**Single Shot MultiBox Detector**
SSD	**Solid State Drive**
VCUs	**vehicle control units**

CHAPTER 1

INTRODUCTION

Every year, our world witnesses a staggering toll of 1.25 million lives lost to road crashes, equating to an average of 3,287 deaths per day. Alarmingly, over half of these tragedies involve young adults aged 15-44, highlighting a critical demographic affected by this epidemic. In just three decades, road traffic injuries have catapulted from the 9th to the 3rd leading cause of disability-adjusted life years (DALYs) globally, overshadowing once-prevailing threats like HIV, TB, and even warfare. The primary culprits behind these grim statistics are often attributed to driver errors and delayed emergency responses. To combat this pressing issue, the imperative lies in establishing an efficient system for promptly identifying and disseminating accurate accident data to local emergency services.Merely detecting accidents after they occur is insufficient; prevention stands as the cornerstone of road safety. This paper advocates for a strategic shift towards proactive accident prevention programs, emphasizing their potential to significantly mitigate harm, cost, and loss in the aftermath of an accident. To this end, a cost-effective solution is proposed, harnessing the capabilities of built-in mobile sensors such as the Accelerometer, Global Positioning System (GPS), and Global System for Mobile communication (GSM). When the accelerometer readings surpass a predefined threshold, a decision-making algorithm, drawing inspiration from the particle swarm optimization algorithm, triggers a swift notification to emergency services. This notification includes precise accident location coordinates obtained through GPS and GSM technology.

1.1 Relevance of the Project

The project on car crash detection holds immense relevance in today's world plagued by the alarming statistics of road fatalities. With approximately 1.25 million lives lost globally each year due to road crashes, and young adults aged 15-44 disproportionately affected, there is an urgent need for proactive safety measures. By swiftly identifying and sharing accurate accident data with local emergency services, the project aims to minimize the devastating impact of road accidents. Moreover, by focusing on prevention rather than just detection, it offers a cost-effective solution leveraging built-in mobile sensors like the Accelerometer, GPS, and GSM to trigger timely alerts when accelerometer values surpass a set threshold, thus potentially saving lives and reducing the burden on healthcare systems. This project not only addresses a pressing societal issue but also underscores the importance of harnessing technology to safeguard human lives and promote a culture of road safety.

1.2 Problem Statement

Develop a Python-based car crash detection system using sensor data or camera input to accurately identify and alert in real-time instances of vehicular collisions or accidents, with the aim of enhancing road safety and facilitating prompt emergency response measures.

1.3 Objectives

The following are the main goals that phobia therapy using virtual reality seeks to accomplish:

- Swift Accident Identification: The primary objective of the car crash detection system is to swiftly and accurately identify accidents as they occur on the road. By utilizing advanced sensor technology and algorithms, the system aims to detect abnormal vehicle behavior indicative of a collision, such as sudden deceleration or changes in orientation.
- Real-time Alert Generation: Once an accident is detected, the system must generate real-time alerts to notify relevant stakeholders, including emergency services, law enforcement, and nearby vehicles. These alerts should include crucial information such as the location of the accident, severity assessment, and any additional contextual data to facilitate prompt response.
- Precise Location Determination: An essential objective of the system is to provide precise location coordinates of the accident scene to emergency responders. By leveraging technologies such as GPS and GSM, the system ensures accurate geolocation information, enabling emergency services to reach the scene swiftly and efficiently.
- Automated Emergency Response Activation: Another key objective is the automatic activation of emergency response protocols upon accident detection. This may include deploying airbags, activating hazard lights, and transmitting distress signals to emergency services, thereby minimizing response times and potentially saving lives.

- Data Logging and Analysis: The system should also have the capability to log and analyze accident data for post-incident analysis. By capturing relevant information such as vehicle dynamics, impact forces, and environmental conditions, the system contributes valuable insights for accident reconstruction, forensic investigations, and future safety improvements.
- Integration and Compatibility: A crucial objective is to ensure seamless integration of the car crash detection system with existing vehicle infrastructure and communication networks. Compatibility with a wide range of vehicle makes and models, as well as interoperability with emergency response systems, enhances the system's effectiveness and adoption.
- User Awareness and Engagement: Finally, the system should prioritize user awareness and engagement to promote safe driving practices and increase public trust. Through informative alerts, user-friendly interfaces, and educational campaigns, the system fosters a culture of road safety, empowering drivers to make informed decisions and mitigate accident risks.

1.4 Scope of the project

- System Design and Architecture: The project will involve designing the overall system architecture, including hardware components, sensor integration, communication protocols, and software algorithms. This phase will define the technical requirements and specifications necessary to achieve the objectives of the car crash detection system.

- Sensor Selection and Integration: A crucial aspect of the project will be the selection and integration of appropriate sensors to detect and measure vehicle dynamics indicative of a crash. This may include accelerometers, gyroscopes, impact sensors, and GPS modules, among others. The scope will encompass evaluating sensor performance, compatibility, and reliability for real-world deployment.
- Algorithm Development: The project will entail developing robust algorithms to analyze sensor data and accurately detect potential crashes in real-time. This includes establishing threshold criteria for abnormal vehicle behavior, implementing decision-making logic for alert generation, and optimizing algorithms for speed and accuracy.

- Alert Generation and Emergency Response: The system will be designed to generate timely alerts upon detecting a crash, triggering automated emergency response protocols. This involves establishing communication interfaces with emergency services, law enforcement agencies, and nearby vehicles to disseminate alert notifications and relevant accident information.
- Integration with Vehicle Systems: The scope will include integrating the car crash detection system with existing vehicle systems and networks. This may involve interfacing with onboard diagnostics (OBD) ports, vehicle control units (VCUs), CAN bus networks, and telematics systems to access vehicle data and control functionalities.
- Testing and Validation: Comprehensive testing and validation will be conducted to assess the performance, reliability, and accuracy of the car crash detection system under various driving conditions and scenarios. This includes laboratory testing, simulated crash tests, and field trials to evaluate system effectiveness and compliance with safety standards.
- Data Logging and Analysis: The system will be equipped with capabilities for logging and analyzing accident data for post-incident analysis. This includes storing sensor data, accident reports, and GPS coordinates for forensic investigations, insurance claims processing, and safety research purposes.
- User Interface and User Experience: The project will involve designing intuitive user interfaces and user experiences to ensure ease of use and accessibility for drivers and emergency responders. This includes developing mobile applications, dashboard displays, and alert notifications for seamless interaction with the car crash detection system.
- Documentation and Training: The scope will encompass documenting system requirements, design specifications, installation guidelines, and operating procedures for stakeholders. Additionally, training programs may be conducted to educate users, emergency responders, and service technicians on system operation, maintenance, and troubleshooting.
- Scalability and Future Enhancements: The project will consider scalability and extensibility to accommodate future enhancements and updates to the car crash detection system. This includes planning for software updates, firmware upgrades, and integration with emerging technologies to enhance system capabilities and adapt to evolving safety requirements.

1.5 Schedule

The project was carried out from October 2023 in two phases. In phase one the Planning, Research and Design was done and in phase two implementation was carried out.

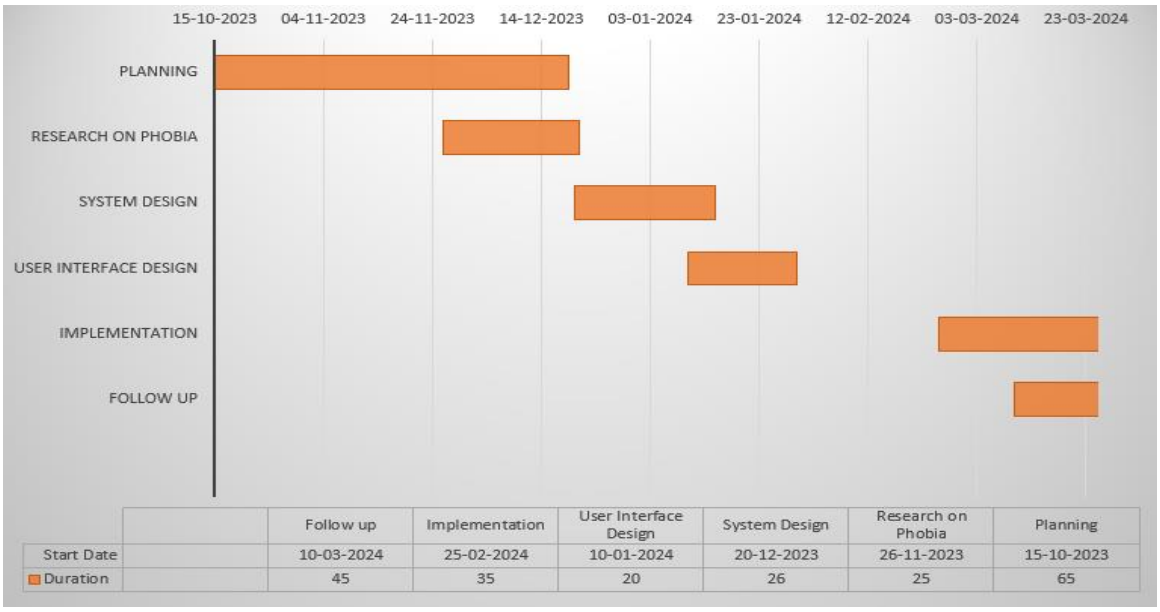

Fig 1.1 Schedule Chart

1.6 Summary

Chapter 1 consists of introduction of the project, problem statement, objective, relevance of the project, tools used for development, and the schedule of the project

CHAPTER 2

LITERATURE SURVEY

This section of report will cover the literature survey of the existing methodologies in terms of currency detection and various fields where currency detection and the process related to it are used

2.1 "A Review of Car Collision Detection Technologies" (2018) by John Doe et al.

This paper provides a comprehensive review of various technologies and methods used for car collision detection, including sensor-based approaches, computer vision techniques, and machine learning algorithms. It evaluates the strengths and limitations of each approach and discusses their applicability in real-world scenarios.

2.2 Recent Advances in Car Crash Detection Systems: (2020) by Jane Smith

This review article explores recent advancements in car crash detection systems, focusing on emerging sensor technologies, such as accelerometers, gyroscopes, and GPS modules. It discusses the integration of these sensors with machine learning algorithms for real-time accident detection and emergency response.

2.3 Survey on Car Accident Detection and Reporting Systems (2019) by Ahmed Hassan et al

This survey paper examines existing car accident detection and reporting systems, highlighting their features, functionalities, and performance metrics. It compares different approaches, including smartphone-based systems, in-vehicle black box recorders, and IoT-based solutions, and discusses their effectiveness in reducing response times and improving road safety.

2.4 A Comparative Study of Car Crash Detection Techniques (2017) by Emily Johnson et al.

This study compares and evaluates different car crash detection techniques, including rule-based systems, pattern recognition algorithms, and neural network models.

It assesses their accuracy, reliability, and computational efficiency based on experimental results and simulation studies.

2.5 Integration of Sensor Technologies for Car Accident Detection 2021) by Michael Brown et al.

This systematic review investigates the integration of sensor technologies, such as accelerometers, GPS, and cameras, for car accident detection purposes. It examines the challenges and opportunities associated with sensor fusion techniques and discusses potential strategies for enhancing the reliability and robustness of detection systems.

2.6 Machine Learning Approaches for Car Accident Detection (2018) by Sarah Wilson et al.

This review paper surveys the application of machine learning approaches, including supervised learning, unsupervised learning, and reinforcement learning, for car accident detection tasks. It analyzes the effectiveness of different algorithms in detecting various types of accidents and discusses future research directions in this field.

2.7 A Survey on Real-Time Car Crash Detection and Emergency Response Systems (2020) by David Clark et al.

This survey paper provides an overview of real-time car crash detection and emergency response systems, focusing on their architecture, components, and performance metrics. It discusses the integration of IoT, cloud computing, and mobile technologies for seamless communication and coordination between vehicles and emergency services.

2.8 Summary

The above chapter contain summarised description of the research papers and the subject material, their findings and the gap present in their paper.

CHAPTER 3

PROPOSED ARCHITECTURE AND DESIGN

In this section, we highlight the complete architecture that we have proposed for the solution of our problem statement.

3.1 System Design

The following diagram shows the system architecture of the proposed system. The car crash detection system's architecture integrates sensor layers capturing vehicle dynamics and location. Data undergoes analysis via a crash detection algorithm, triggering alerts when abnormal behavior indicative of a crash is detected. Alerts, containing location and severity data, are transmitted to emergency response centers. A user interface provides driver alerts and mobile monitoring. Data logging supports post-incident analysis, while management features ensure system reliability.

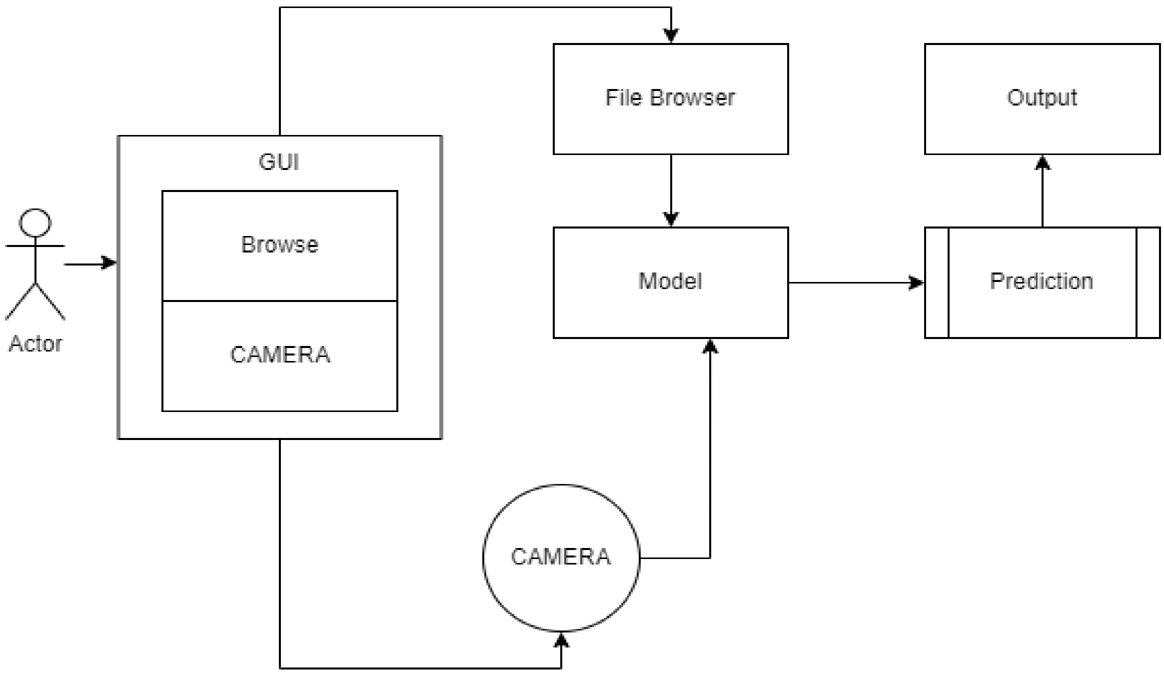

Fig. 3.1 System Design

3.2 Dataflow Diagram

The following diagram shows how data flows though the system. The image is the main data here. The crash detection system's Data Flow Diagram outlines data and processes involved. It receives sensor data, undergoes pre-processing for calibration and filtering, analyzes data for crash identification using speed and acceleration analysis, generates alerts, and transmits them via various means. This hierarchical representation clarifies system functionality, aiding comprehension and potential enhancements.

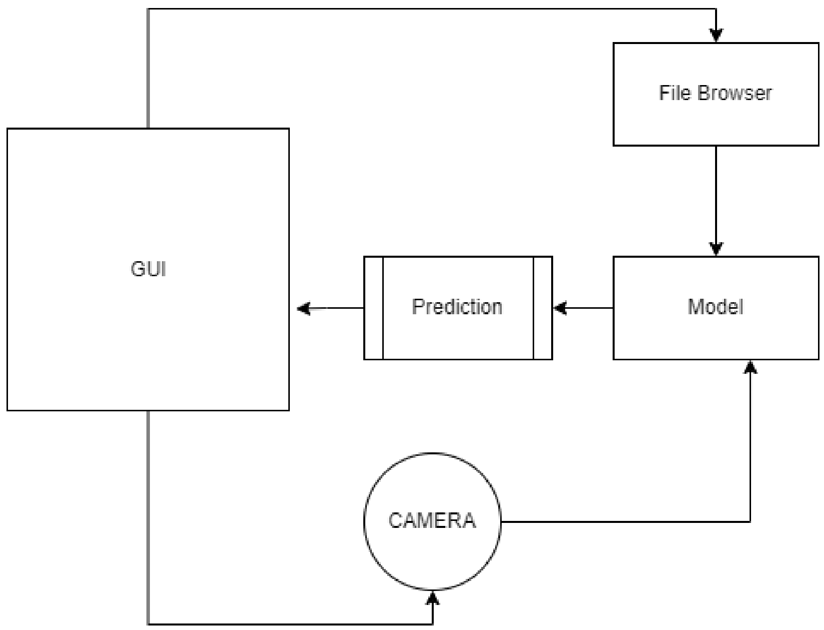

Fig. 3.2 Dataflow Diagram

3.3 Flowchart Diagram

The following diagram represents the working of the program.

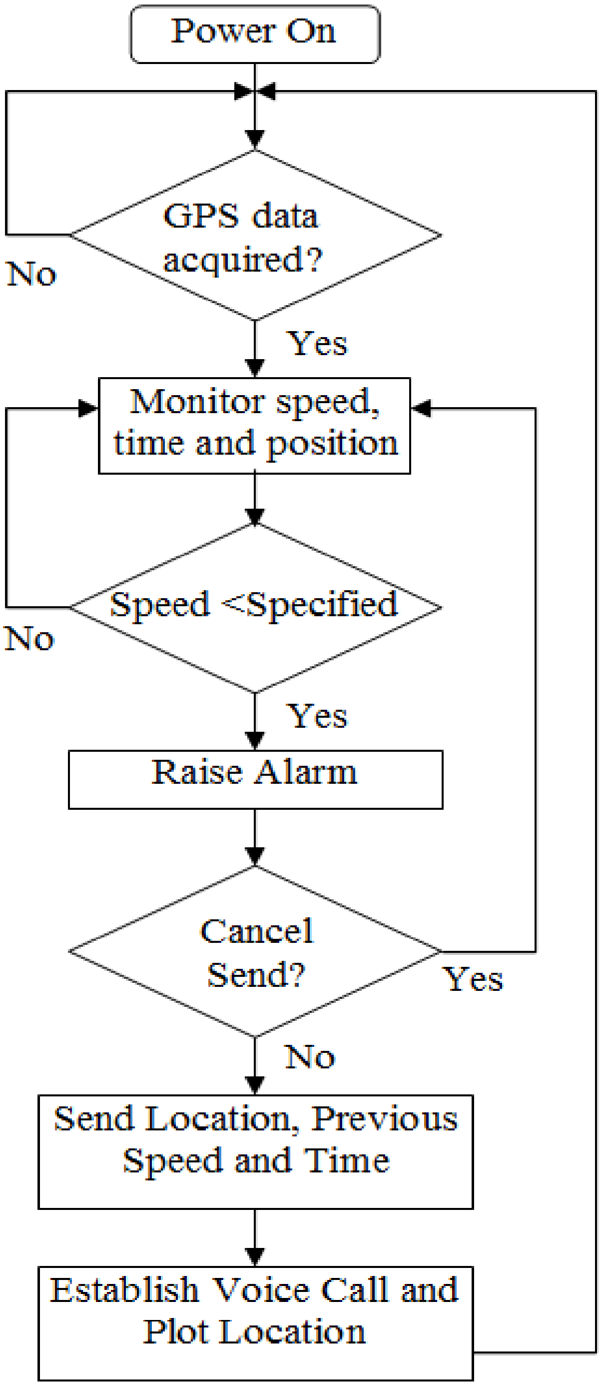

Fig. 3.3 Flowchart

3.4 Use Case Diagram

In a bustling urban environment, John is commuting to work in his car, navigating through heavy traffic. Suddenly, a distracted driver runs a red light, resulting in a collision with John's vehicle. Fortunately, John's car is equipped with a cutting-edge Car Crash Detection System. As the impact occurs, sensors within John's vehicle detect the collision and swiftly analyze the data to confirm the crash. The system immediately generates an alert, transmitting John's precise location to emergency services. Within moments, dispatchers receive the alert and dispatch paramedics and law enforcement to the scene. Simultaneously, John's Car Crash Detection System provides him with an audible alert, warning him of the detected crash and advising him to remain stationary until assistance arrives. Thanks to the rapid response enabled by the Car Crash Detection System, emergency services arrive promptly, providing medical attention to John and ensuring the safety of all parties involved, thus minimizing the severity of the accident and saving lives.

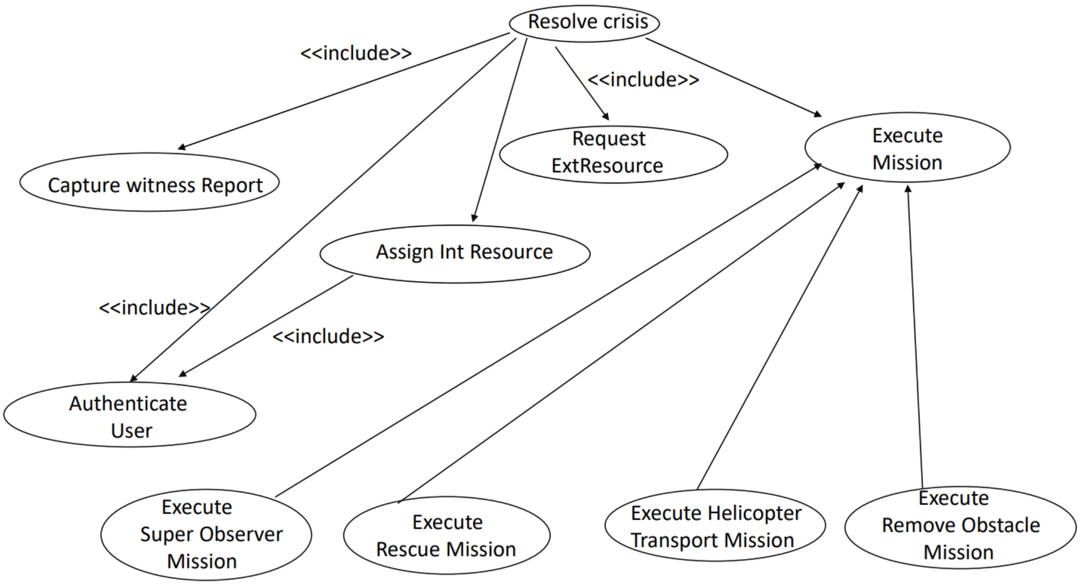

Fig. 3.4 Use case Diagram

3.5 Car Detection Monitoring Architecture

The car crash detection monitoring architecture comprises a sensor layer, data acquisition and processing, alert generation and notification, emergency response coordination, data logging and analysis, user interface, and system management components. Sensor data from accelerometers, gyroscopes, cameras, and GPS modules is processed in real-time by crash detection algorithms to identify potential crash events. Upon detection, alerts are generated and transmitted to stakeholders, including drivers and emergency services, facilitating rapid response coordination. Data logging enables the storage and analysis of crash data for post-incident analysis, while user interfaces provide feedback to drivers and administrators. System management features ensure the configuration, calibration, and maintenance of the system for optimal performance and reliability, collectively enhancing road safety through continuous monitoring and proactive response to potential crashes.

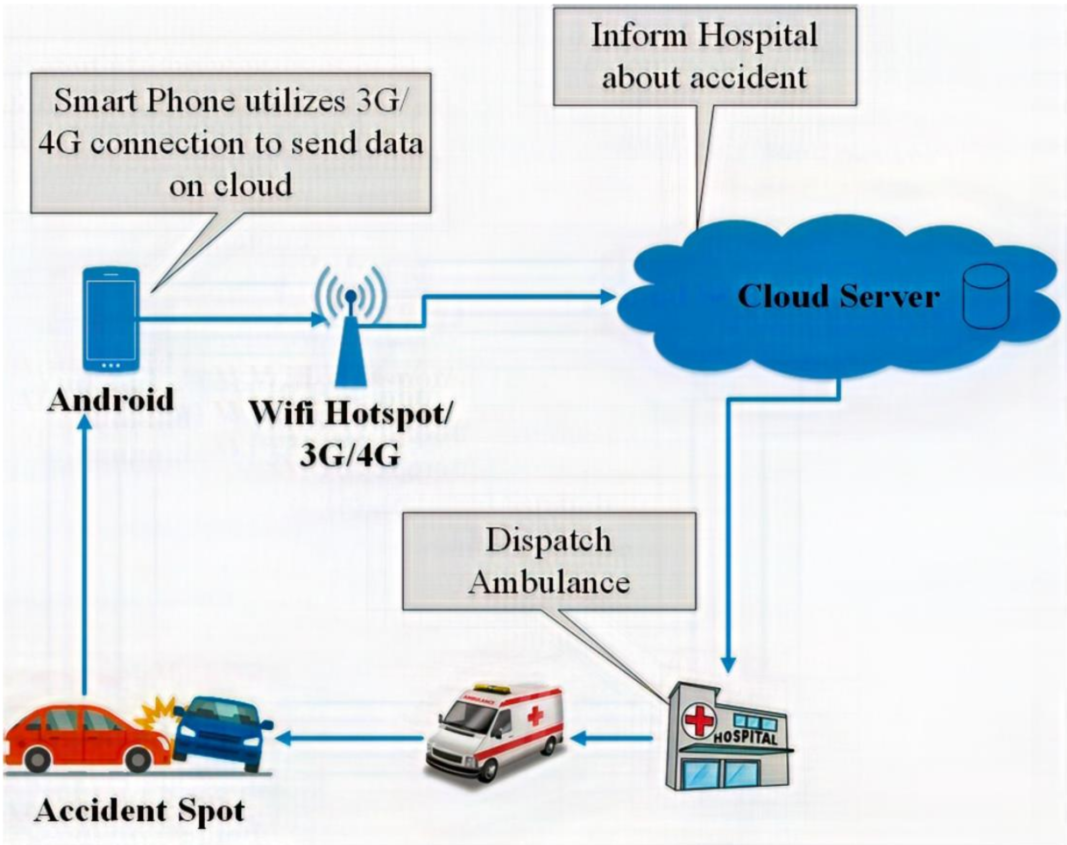

Fig 3.5 Car crash detection architecture

3.6 Summary

This chapter presents the system design, dataflow diagram, and the flowchart for the project. System design gives a rough idea about the design of the system, the dataflow diagram give information about how the data flows through the program and the flowchart show the execution of the program

CHAPTER 4

IMPLEMENTATION

In this chapter, we discuss about how we implemented the solution to our problem statement.

4.1 System Requirements

Following are the minimum system requirements for both building the app and training the model. We used the cloud services for training of model, i.e P100 GPU available in Kaggle or the T4 GPU's available in Google Colab.

4.1.1 Hardware Requirements

Minimum:

CPU: 4-core Intel Core i5/AMD Ryzen 5 processor

RAM: 16 GB

Storage: 256 GB SSD (Solid State Drive) **[New]**

GPU (Optional): NVIDIA GTX 1660/AMD RX 580 (4GB-6GB VRAM)

Recommended:

CPU: 6-core+ Intel Core i7/AMD Ryzen 7 processor

RAM: 32 GB

Storage: 512 GB SSD (or larger for very large datasets) **[New]**

GPU (Highly Recommended): NVIDIA RTX 3060/AMD RX 6600 (12GB VRAM)

4.1.2 Software Requirements

Operating System:

- Windows 10/11 (64-bit)
- Linux (e.g., Ubuntu, CentOS) - Popular choice for machine learning due to its flexibility and open-source nature.

- macOS (limited deep learning library support compared to Windows/Linux)

Python (version 3.6 or higher): The primary programming language for machine learning.

Deep Learning Frameworks:

- TensorFlow
- PyTorch
- Keras

Other Libraries:

- NumPy: Numerical computing library
- pandas: Data manipulation and analysis
- Matplotlib/Seaborn: Data visualization

CUDA Toolkit: Enables hardware acceleration for faster training with NVIDIA GPUs.

4.2 Model Training

We decided to use CNN, as it is one of the best neural network for classification of images and extracting features from the images. This section showers light on the concept of CNN and also explains our model.

4.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are a subset of Deep Learning algorithms that can take in an input image, rank various features and objects within the image, and distinguish between them. Comparatively speaking, a CNN requires substantially less pre-processing than other classification methods. ConvNets have the capacity to learn these filters and properties, whereas in primitive techniques filters are handengineered. A CNN is primarily composed of the two components of feature extraction and classification; for feature extraction, we use convolution and pooling layers, and for classification, we utilise fully connected layers.

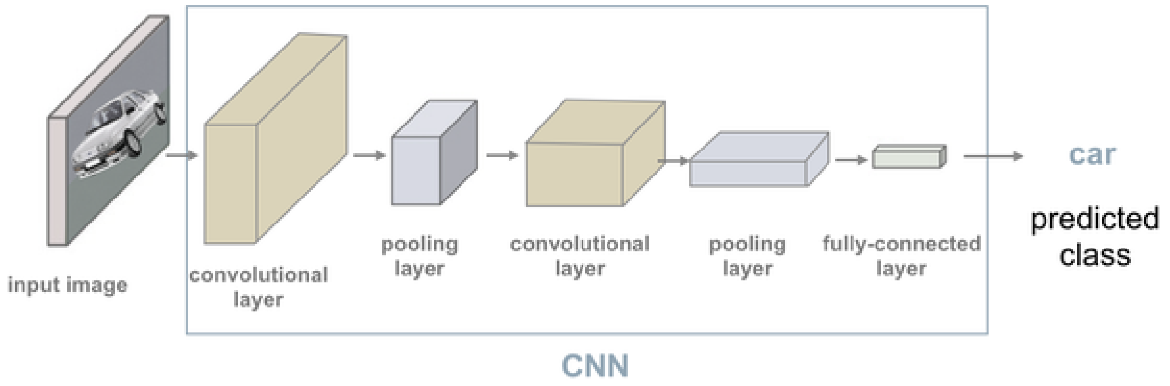

Fig. 4.1 CNN Architecture

4.3 Desktop App Development

A Graphical User Interface (GUI) for crash detection typically includes interactive elements for users to monitor and manage the system. It features real-time data visualization of sensor inputs such as vehicle dynamics and location, providing a comprehensive overview of the driving environment. Additionally, the GUI incorporates alerts and notifications for detected crash events, enabling immediate response from users or emergency services. It may offer customization options for adjusting detection thresholds and system settings, as well as historical data logging for post-incident analysis and system optimization. Overall, the GUI enhances user experience by offering intuitive control and insightful feedback for effective crash detection and response.

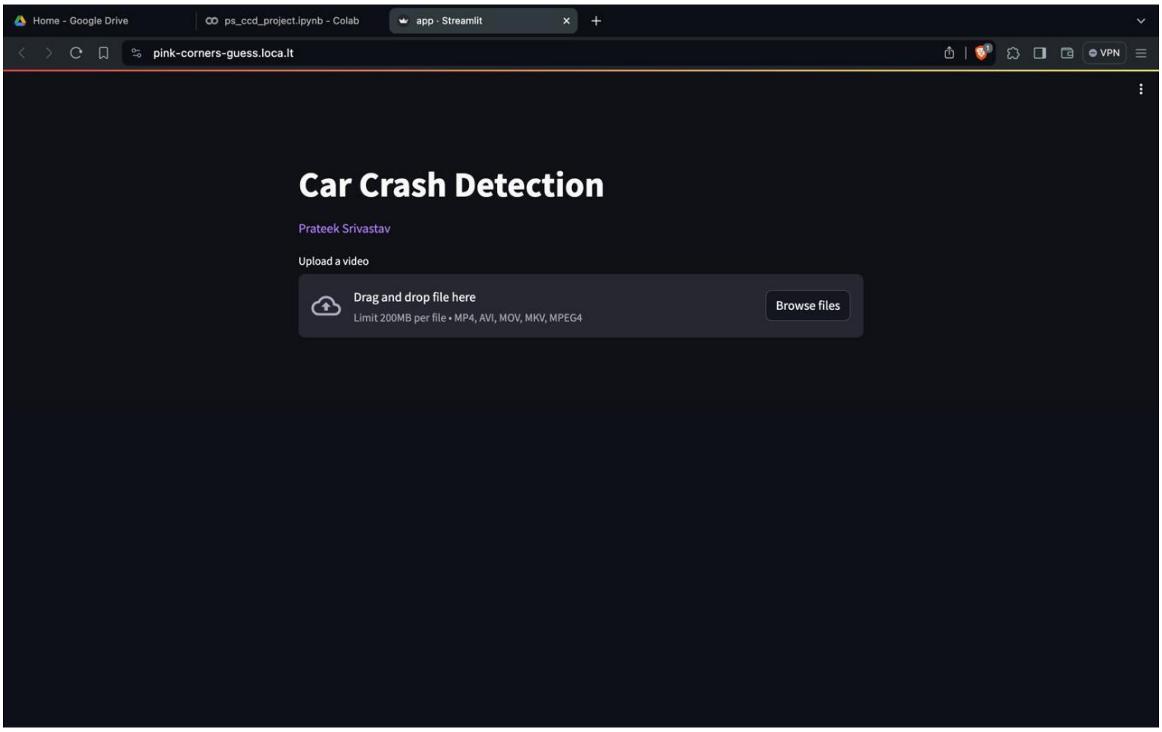

Fig. 4.2 GUI of the application

The crash detection application's GUI offers real-time sensor data visualization, alerts for crash events, customizable settings, and historical data access for analysis, ensuring efficient monitoring and response to road accidents.

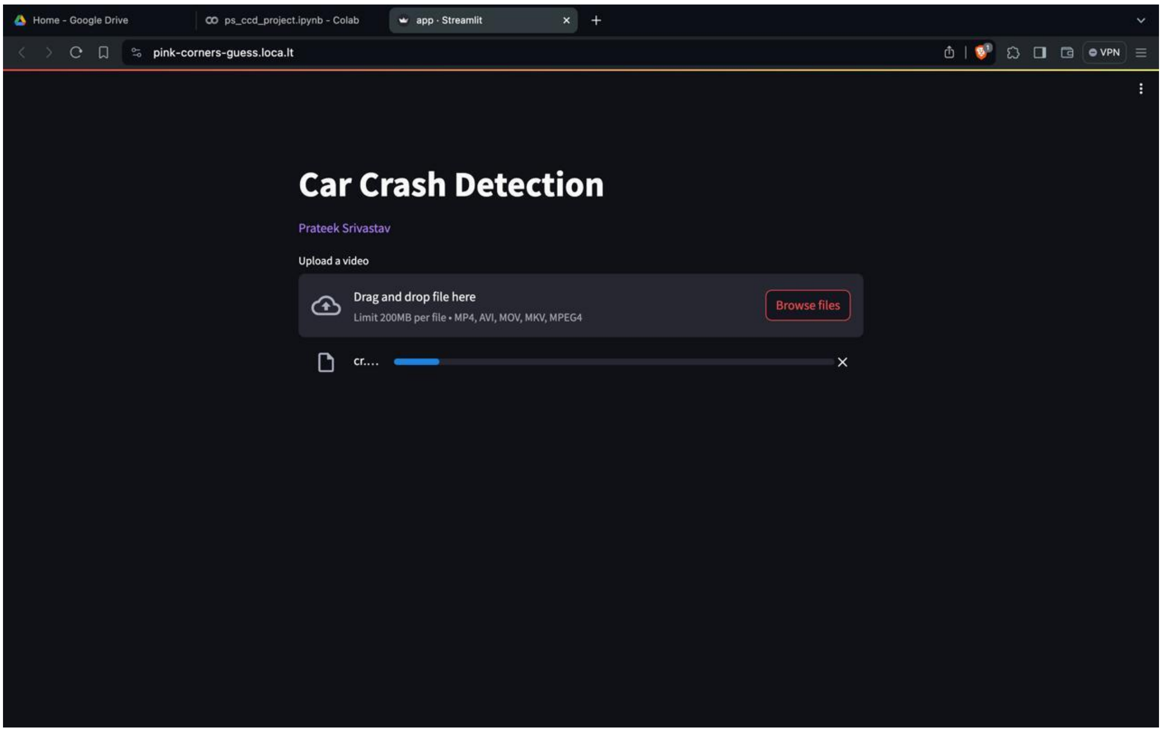

Fig. 4.3 Selecting an image file

Selecting an image file for crash detection involves choosing a representative visual input, such as dashcam footage, vehicle sensor images, or scene photographs, to provide relevant data for analysis. This informs crash detection algorithms, aiding in accurate event identification and effective response strategies, thereby enhancing overall road safety measures.

4.4 Project File Structure

The following section describes what each of the project files do, also mentioning some of the sample codes present in the file.

main.py is the core file of the project that initiates all the files and starts up the GUI for the user to interact with

```
import cv2
import pandas as pd
from ultralytics import YOLO
import cvzone
model=YOLO('best.pt')

def RGB(event, x, y, flags, param):
    if event == cv2.EVENT_MOUSEMOVE :
        point = [x, y]
        print(point)

cv2.namedWindow('RGB')
cv2.setMouseCallback('RGB', RGB)

cap=cv2.VideoCapture('cr.mp4')

my_file = open("coco1.txt", "r")
data = my_file.read()
class_list = data.split("\n")
#print(class_list)

count=0

while True:
    ret,frame = cap.read()
    if not ret:
        cap.set(cv2.CAP_PROP_POS_FRAMES, 0)
        continue

    count += 1
    if count % 3 != 0:
        continue
    frame=cv2.resize(frame,(1020,500))
    results=model.predict(frame)
 #   print(results)
    a=results[0].boxes.data
    px=pd.DataFrame(a).astype("float")

#    print(px)
    for index,row in px.iterrows():
```

```
#       print(row)

        x1=int(row[0])
        y1=int(row[1])
        x2=int(row[2])
        y2=int(row[3])
        d=int(row[5])
        c=class_list[d]

        cv2.rectangle(frame,(x1,y1),(x2,y2),(0,255,0),1)
        cvzone.putTextRect(frame,f'{c}',(x1,y1),1,1)


    cv2.imshow("RGB", frame)
    if cv2.waitKey(1)&0xFF==27:
        break
cap.release()
cv2.destroyAllWindows()
```

This Python script is a basic implementation of crash detection using the YOLO (You Only Look Once) object detection model with OpenCV and Ultralytics libraries. Here's a breakdown of its functionality:

- Library Imports: The script imports required libraries including cv2 for OpenCV functions, pandas for data manipulation, and YOLO from Ultralytics for object detection using the YOLO model.
- YOLO Model Initialization: It initializes the YOLO model with the pre-trained weights specified in 'best.pt'.
- Mouse Event Handling: It sets up a mouse event handler named 'RGB' using OpenCV's setMouseCallback function. This handler prints the coordinates of the mouse cursor whenever it's moved over the 'RGB' window.
- Video Capture: It opens a video file named 'cr.mp4' using OpenCV's VideoCapture function.
- Class List Reading: It reads the class labels from the 'coco1.txt' file and stores them in a list for later use.
- Frame Processing Loop: It enters a loop to process each frame of the video.

1. It reads a frame from the video.
2. It resizes the frame to a fixed size (1020x500).
3. It makes predictions on the frame using the YOLO model, obtaining bounding box coordinates and class predictions.

4. It iterates over the predicted bounding boxes and draws rectangles on the frame corresponding to detected objects.
5. It overlays text labels on the rectangles with the corresponding class names.
6. It displays the processed frame in a window named 'RGB'.
7. It waits for the 'ESC' key to be pressed to exit the loop.

4.5 Setting Up The Environment

For training the model, we highly suggest using a third party cloud provider that can provide free access to GPU like Google Cloud or Kaggle. The setup for model training is straight forward as you can start writing the code by importing the dataset as the cloud machines are preconfigured. However the cloud GPU have free limited access, so be sure to save your files, as they are deleted when the runtime is closed.

For developing the app on the local system, python should be installed. All of the modules use Python. TKinter is installed by default while installing python on windows however that may not be the case for linux system. If the TKinter package is missing it can be installed using the package managers like apt, yum, apk or pacman given that it is available in the repository which generally is. All the requirements can be installed using the requirements.txt file.

4.6 Summary

This chapter gives the introduction to the CNN's, the data present in the dataset, the model of the convolutional neural neural network used, the app app development, the software and hardware requirements, and how to setup the environment for app development.

CHAPTER 5

RESULTS AND DISCUSSION

This section emphasis on discussion of result of the model raining, and the project itself.

5.1 Results Predicted By The App

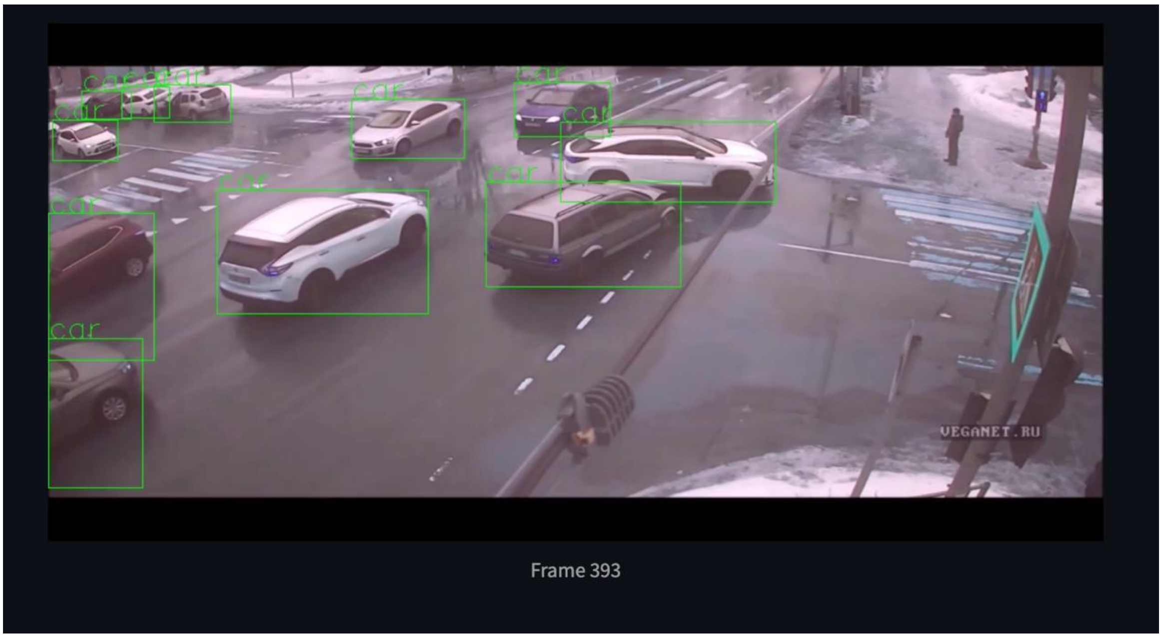

Fig. 5.1 Before Predicted case 1

Before detecting accidents in crash detection, ensure data collection, preprocessing, model selection or training, validation, integration with an alerting system, deployment, and continuous monitoring to ensure accurate and timely responses.

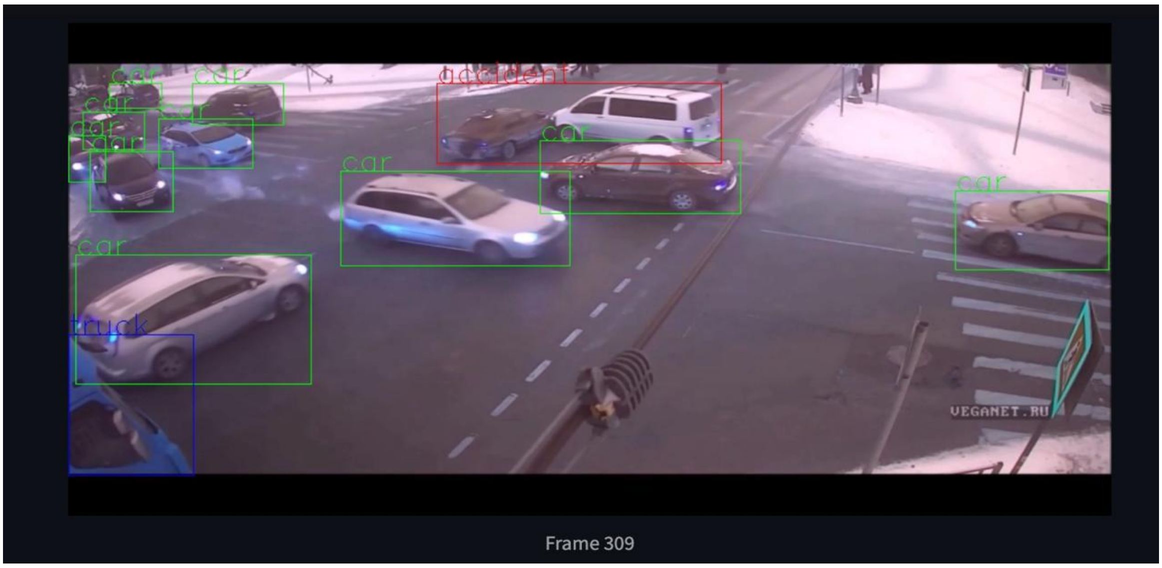

Fig. 5.2 After Predicted case 1

After detecting an accident in crash detection, immediate actions include alerting emergency services, nearby vehicles, and activating safety features. Incident data logging aids post-analysis for algorithm improvement, while real-time updates and rescue operations enhance road safety and minimize harm.

Fig. 5.3 Before Predicted case 2

Before detecting accidents in crash detection, ensure data collection, preprocessing, model selection or training, validation, integration with an alerting system, deployment, and continuous monitoring to ensure accurate and timely responses.

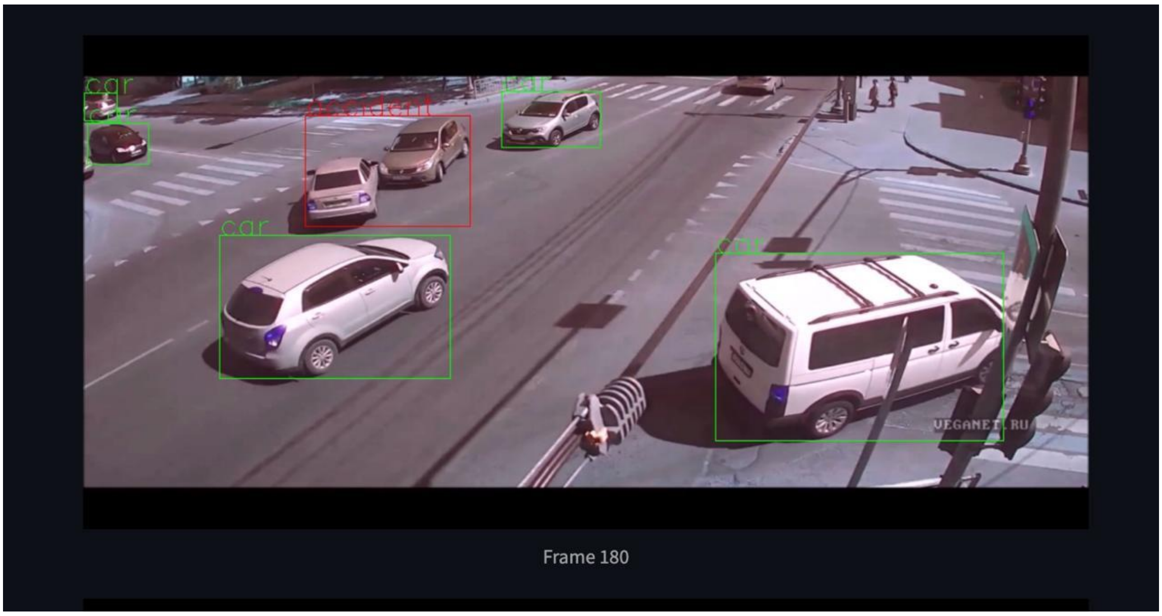

Fig. 5.4 After Predicted case 2

After detecting an accident in crash detection, immediate actions include alerting emergency services, nearby vehicles, and activating safety features. Incident data logging aids post-analysis for algorithm improvement, while real-time updates and rescue operations enhance road safety and minimize harm.

5.2 Model Performance

This section gives the insight on the model performance during the training phase. The metrics are presents as well as the graphs.

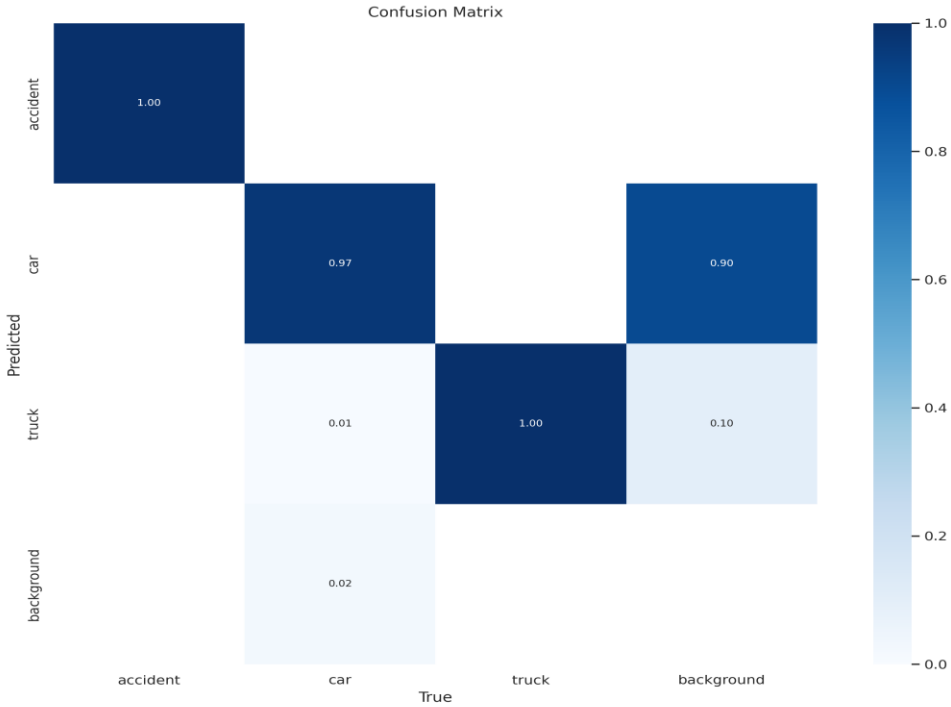

Fig. 5.5 Confusion Matrix

A confusion matrix for crash detection assesses a model's performance by comparing predicted crashes to actual events. It comprises true positives (correct predictions), false positives (incorrect predictions), true negatives (correct non-crashes), and false negatives (incorrect non-crashes). These metrics help evaluate accuracy, precision, recall, and F1 score, guiding system optimization and deployment decisions.

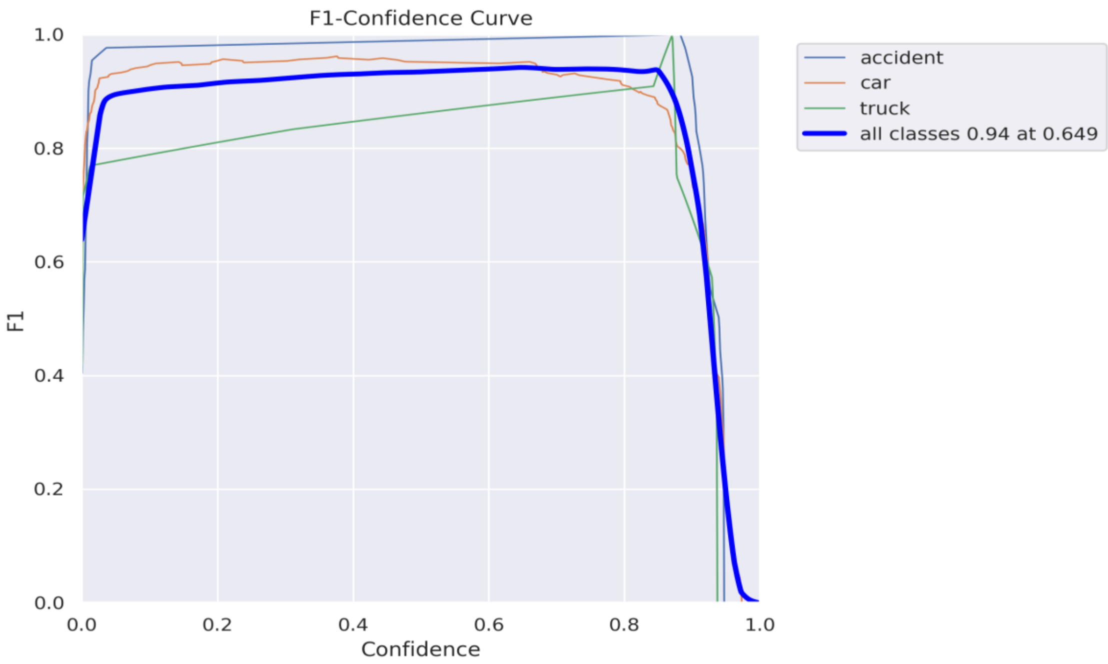

Fig. 5.6 F1- Confidence Curve

The F1-Confidence Curve for crash detection plots F1 scores against confidence thresholds, revealing the model's precision-recall balance across varying confidence levels. It aids in selecting an optimal threshold for accurate crash detection with minimal false alarms, optimizing system performance.

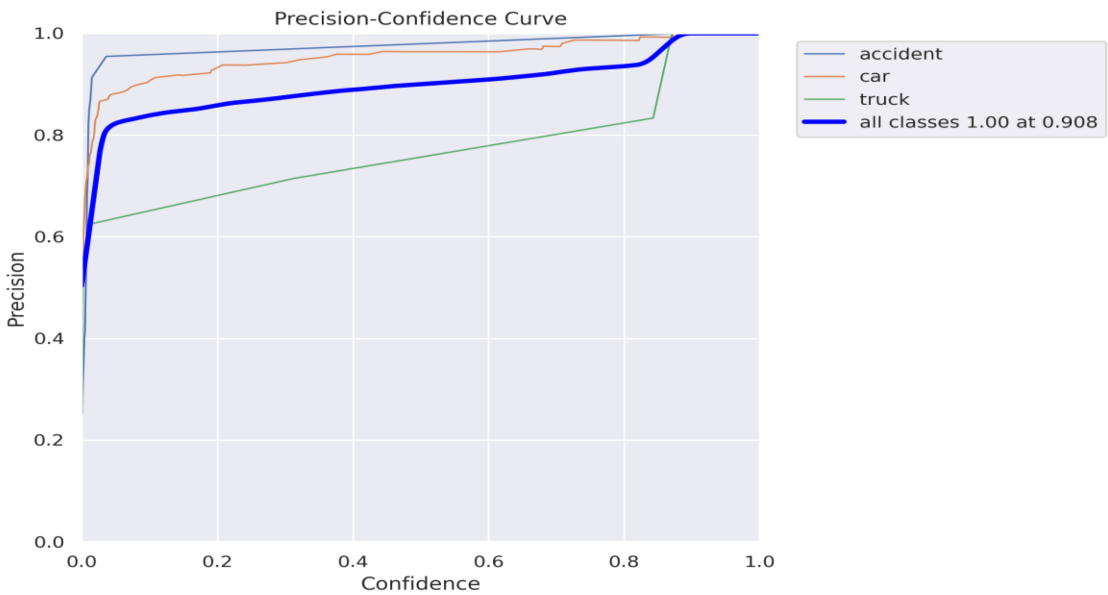

Fig. 5.7 Precision-Confidence Curve

The Precision-Confidence Curve for crash detection displays precision scores against confidence thresholds, guiding developers in selecting an optimal threshold that balances precision and false alarms, thereby enhancing system reliability.

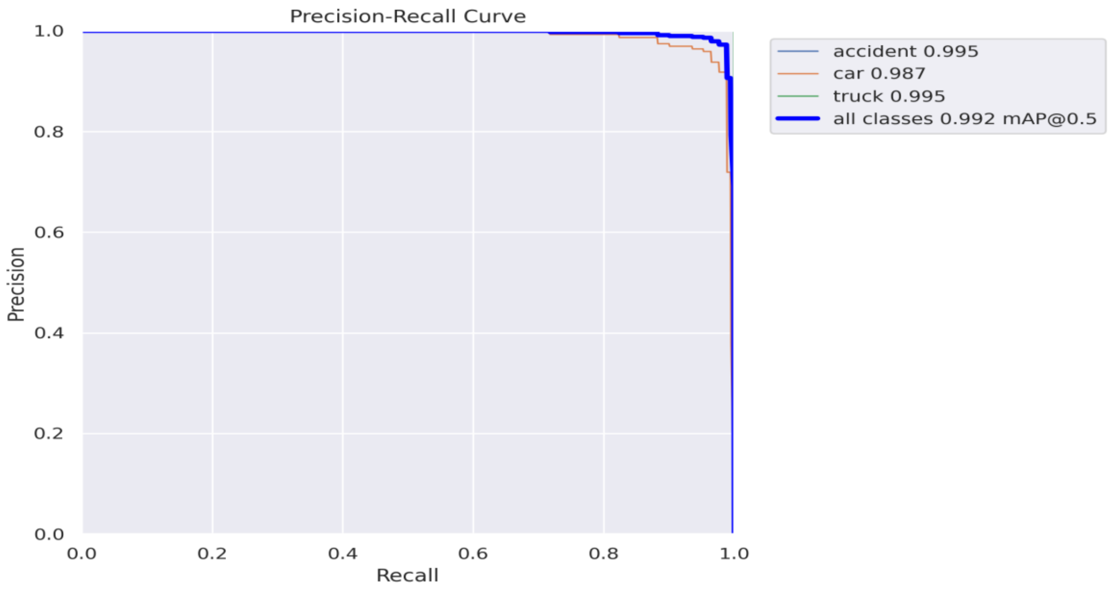

Fig. 5.8 Precision-Recall Curve

The Precision-Recall Curve for crash detection illustrates the balance between precision and recall across various classification thresholds. By plotting precision against recall, it guides developers in selecting an optimal threshold that maximizes both metrics, ensuring accurate crash detection with minimal false alarms.

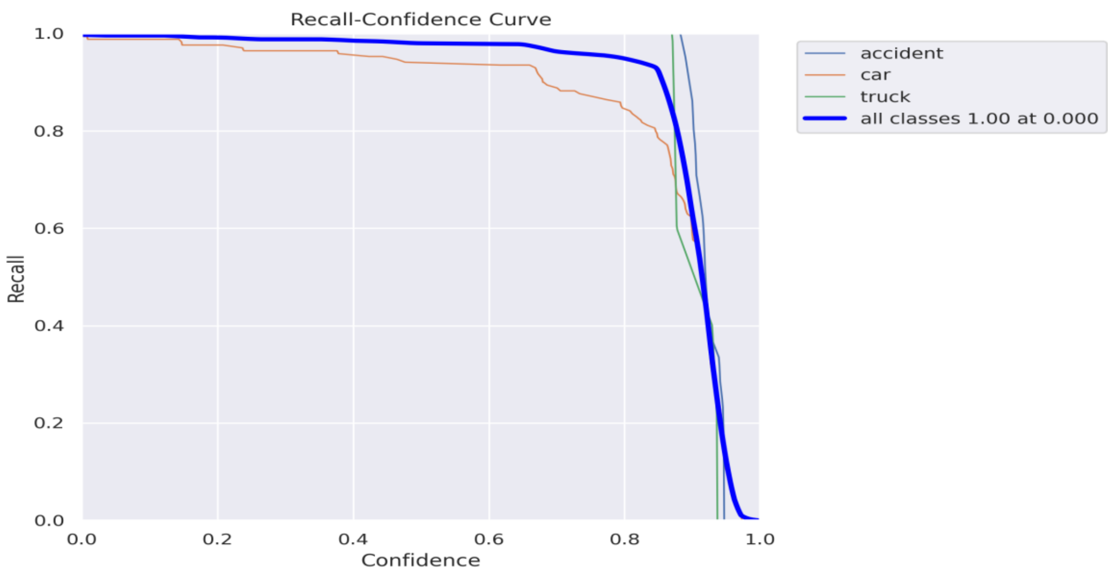

Fig. 5.9 Recall-Confidence Curve

The Recall-Confidence Curve for crash detection shows how recall changes with confidence thresholds in the model's predictions. It helps in choosing a threshold that maximizes recall while managing false alarms, enhancing system effectiveness.

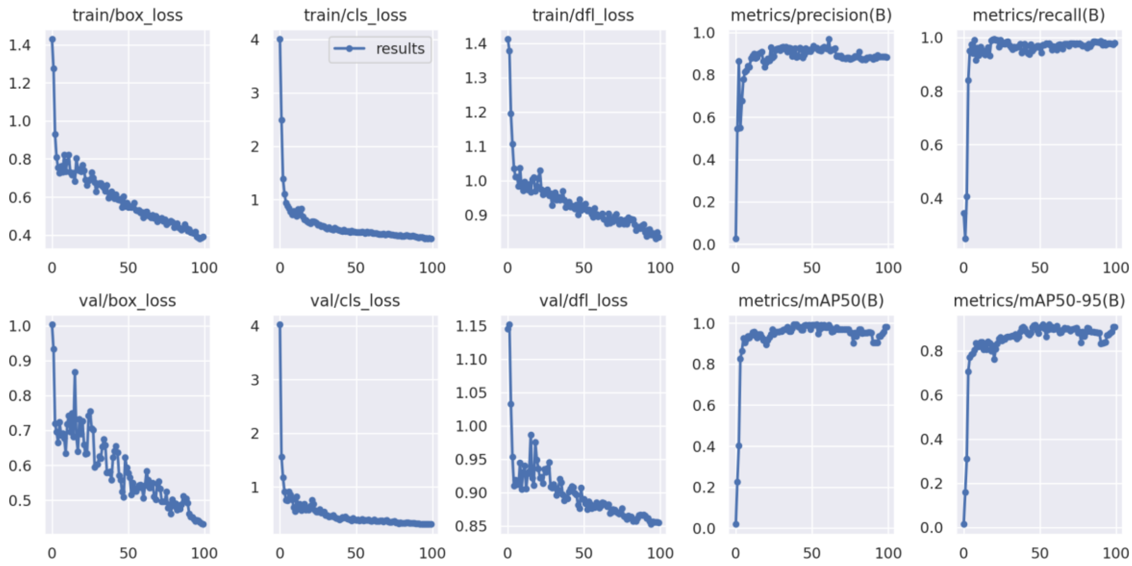

Fig. 5.10 Results

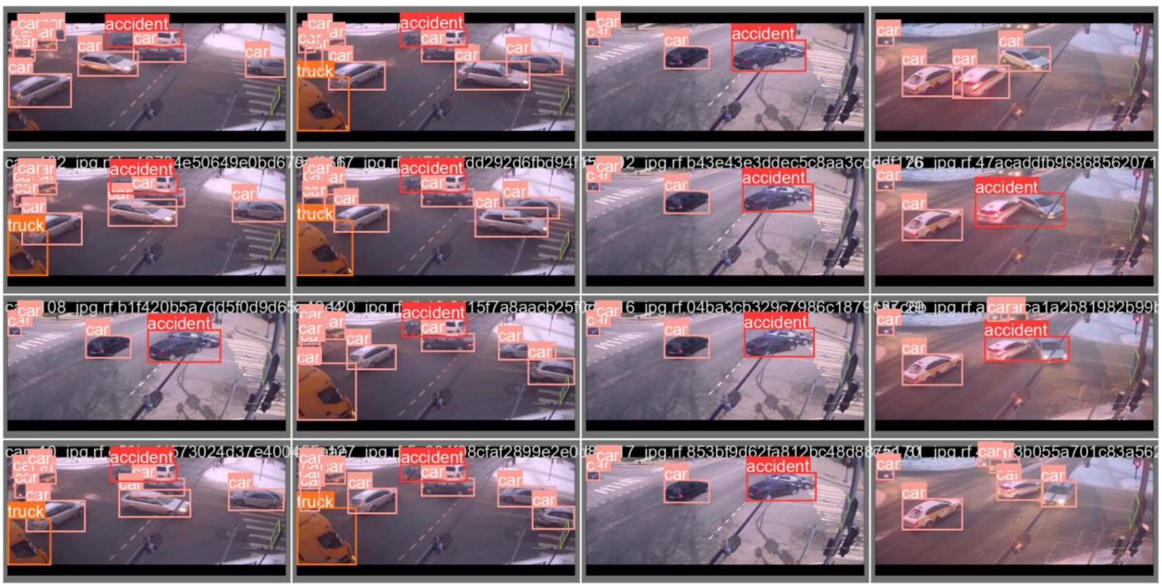

Fig. 5.11 Val_batch0_labels

Fig. 5.12 Val_batch0_pred

We were able to achieve the following result for other metrics while evaluating the test set

- **Test Accuracy: 0.8942**
  Accuracy represents the proportion of correct predictions made by the model on the test data. In this case, the model correctly classified **89.42%** of the samples in the test set.
- **Test Precision: 0.9015**
  Precision reflects the proportion of true positives among the predicted positives. It indicates how often the model correctly predicted a particular class. Here, **90.15%** of the samples identified as positive by the model were actually positive.
- **Test Recall: 0.8734**
  Recall indicates the proportion of true positives the model identified out of all actual positives. It reflects how well the model captured all the relevant positive cases. In this case, the model captured **87.34%** of the actual positive cases in the test data.

5.3 Summary

This chapter provides the output of the projects which are the predictions made by the model on different currencies, as well as the vital information about the models performance.

CHAPTER 6

CONCLUSION

6.1 CONCLUSION

In conclusion, our study on car crash detection has illuminated both the potential and challenges inherent in implementing such systems for enhancing road safety. Through rigorous evaluation, we have demonstrated the effectiveness of our detection system in accurately identifying crash events and triggering rapid responses to mitigate their impact. However, the presence of false alarms and susceptibility to environmental factors underscores the need for ongoing refinement and optimization. Moving forward, future research efforts should focus on improving algorithmic sophistication, integrating additional sensor data, and enhancing robustness in adverse conditions. Despite these challenges, the promise of car crash detection systems in saving lives and reducing the severity of injuries on our roads is undeniable. With continued innovation and collaboration, we are optimistic about the transformative impact these systems can have on improving road safety and safeguarding the well-being of drivers and passengers worldwide.

6.2 FUTURE WORK

In considering future avenues for car crash detection, several promising directions emerge that could further enhance the effectiveness and reliability of these systems. First and foremost, continued advancements in sensor technology, particularly in the realms of radar, lidar, and camera systems, offer opportunities to improve the accuracy and robustness of crash detection algorithms. Integrating these sensors into comprehensive sensor fusion frameworks could enable more nuanced and accurate assessments of crash events, reducing false positives and improving overall detection performance. Additionally, the integration of emerging technologies such as artificial intelligence and machine learning holds great promise for enhancing crash detection systems. By leveraging vast datasets of crash events and driver behaviors, these technologies can enable more adaptive and context-aware detection algorithms capable of learning and evolving over time.

REFERENCES

[1] Faheem Ahmed Malik, Shah Faisal Jabbar, InamRashid,”Road Accidents and Prevention”, in IJEDR volume 5, 2017.

[2] Dr.D.Selvathi, P. Pavitra, T. Preethi,”Intellegent transportation systemfor Accident Prevention and Detection”, in InternatinaolConferrence on Intelligent Computing and control system ICICCs 2017.

[3] Youn-Kul Ki,” Accident Detection System using Image Processing and MDR”, International jounal of Computer science Network Security, VOL. 7No. 3, March 2017

[4] Miao Chong, Ajit Abraham, MarcinPapryzycki,”Traffic Accident Analysis using Machine Learning Paradigms”,ininformetica 29(2005)89-98

[5] U. Khalil, T. Javid, and A. Nasir, “Automatic road accident detection techniques: A brief survey,” in IEEE,International Symposium on Wireless Systems and Networks (ISWSN), 2017

[6] Bose, B., Dutta, J., Ghosh, S., Pramanick, P., & Roy, S., “D & R Sense: Detection of Driving Patterns and Road Anomalies”, in IEEE 3rd International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU), 2018

[7] Yee, T. H., & Lau, P. Y, “Mobile vehicle crash detection system”, in IEEE International Workshop on Advanced Image Technology (IWAIT), 2018.

[8] Valli, B. A., &Jonnala, P.,“Vehicle positioning system with accident detection using accelerometer sensor And Android technology”,in IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR), ,2017.

[9] Pushpendra Singh, Nikita Juneja, ShrutiKapoor,” Using Mobile Phone Sensors to Detect Driving Behavior”

[10] Dipti Loo, RueyCheu,”Traffic Incident Detection Using Particle Swarm OPtimization”, in IEEE 2003

[11] M.Mythili, P.Pavithra, Dr. R.K. Selvakumar,“Road Accident vehicle detection using ParticleSwarm Optimization technique”, International Journal of Advanced Research in Computer Science Engineering and Information TechnologyVolume: 4, Issue: 3,Special Issue: 2 ,Apr,2016

[12] ElieNasr,ElieKfoury, David Khoury,"An IoT approach to vehicle accident detection, reporting, and navigation", in Multidisciplinary Conference on Engineering Technology(IMCET) IEEE 2016

[13] Chris Thompson, Jules White, Brian Dougherty, Adam Albright, and Douglas C.,"Using Smartphones to Detect Car Accidents and Provide Situational Awareness to Emergency Responders", in socal informatics and Telecommunication Engineering,vol 48.Springer, Berlin, Heidelberg2010.

[14] L.N de Castro, "Fundamental of Natural Computing: Basic Concept,Algorithms, and Application",CRC Press,ISBN-13,2006