# AI-BASED FRAMEWORK FOR AUTOMATICALLY EXTRACTING HIGH-LOW FEATURES FROM NDS DATA TO UNDERSTAND DRIVER BEHAVIOR

---

A Dissertation

presented to

the Faculty of the Graduate School

at the University of Missouri-Columbia

---

In Partial Fulfillment

of the Requirements for the

Degree Doctor of Philosophy

---

by

ARMSTRONG ABOAH

Dr. Yaw Adu-Gyamfi, Dissertation Supervisor

DECEMBER 2022

The undersigned, appointed by the dean of the Graduate School, have examined the dissertation entitled

**AI-BASED FRAMEWORK FOR AUTOMATICALLY EXTRACTING HIGH-LOW FEATURES FROM NDS DATA TO UNDERSTAND DRIVER BEHAVIOR**

Presented by **Aboah Armstrong**,

A candidate for the degree **Doctor of Philosophy**,

And hereby certify that, in their opinion, it is worthy of acceptance.

Dr. Yaw Adu-Gyamfi

Dr. Praveen Edara

Dr. Carlos Sun

Dr. Timothy Matisziw

## DEDICATION

This dissertation is proundly dedicated

To my Parents and all my Friends,

Thanks for your Endless Love, Sacrifices, Prayers, Support and Guidance.

## ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my advisor, Dr. Yaw Adu-Gyamfi, for always keeping his office doors open for me. He would make time for me no matter what time of day it was. Because of his friendly demeanor, I never hesitated to present my ideas, which he always supported and encouraged with his inventiveness. I consider myself fortunate to have been supervised by a professor as kind, intelligent, and lively as him.

I would like to thank my parents as well as my finance. They were my most dependable source of assistance throughout the years. Their words of encouragement and advice helped me overcome many obstacles in my life. It's a blessing to be born into such a loving and caring family. I'd like to thank the members of my dissertation committee for their time, advice, and helpful feedback that helped me finish this PhD dissertation.

Finally, I would like to thank all of my friends for supporting me throughout my PhD program and making my time in Columbia memorable.

# TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

# AI-BASED FRAMEWORK FOR AUTOMATICALLY EXTRACTING HIGH-LOW FEATURES FROM NDS DATA TO UNDERSTAND DRIVER BEHAVIOR

Armstrong Aboah

Dr. Yaw Adu-Gyamfi, Dissertation Supervisor

## ABSTRACT

Our ability to detect and characterize unsafe driving behaviors in naturalistic driving environments and associate them with road crashes will be a significant step toward developing effective crash countermeasures. Due to some limitations, researchers have not yet fully achieved the stated goal of characterizing unsafe driving behaviors. These limitations include, but are not limited to, the high cost of data collection and the manual processes required to extract information from NDS data.

In light of this limitations, the primary objective of this study is to develop an artificial intelligence (AI) framework for automatically extracting high-low features from the NDS dataset to explain driver behavior using a low-cost data collection method. The author proposed three novel objectives for achieving the study's objective in light of the identified research gaps. Initially, the study develops a low-cost data acquisition system for gathering data on naturalistic driving. Second, the study develops a framework that automatically extracts high- to low-level features, such as vehicle density, turning movements, and lane changes, from the data collected by the developed data acquisition system. Thirdly, the study extracted information from the NDS data to gain a better understanding of people's car-following behavior and other driving behaviors in order to develop countermeasures for traffic safety through data collection and analysis.

The first objective of this study is to develop a multifunctional smartphone application for collecting NDS data. Three major modules comprised the designed app: a front-end user interface module, a sensor module, and a backend module. The front-end, which is also the application's user

interface, was created to provide a streamlined view that exposed the application's key features via a tab bar controller. This allows us to compartmentalize the application's critical components into separate views. The backend module provides computational resources that can be used to accelerate front-end query responses. Google Firebase powered the backend of the developed application. The sensor modules included CoreMotion, CoreLocation, and AVKit. CoreMotion collects motion and environmental data from the onboard hardware of iOS devices, including accelerometers, gyroscopes, pedometers, magnetometers, and barometers. In contrast, CoreLocation determines the altitude, orientation, and geographical location of a device, as well as its position relative to an adjacent iBeacon device. The AVKit finally provides a high-level interface for video content playback.

To achieve objective two, we formulated the problem as both a classification and time-series segmentation problem. This is due to the fact that the majority of existing driver maneuver detection methods formulate the problem as a pure classification problem, assuming a discretized input signal with known start and end locations for each event or segment. In practice, however, vehicle telemetry data used for detecting driver maneuvers are continuous; thus, a fully automated driver maneuver detection system should incorporate solutions for both time series segmentation and classification. The five stages of our proposed methodology are as follows: 1) data preprocessing, 2) segmentation of events, 3) machine learning classification, 4) heuristics classification, and 5) frame-by-frame video annotation. The result of the study indicates that the gyroscope reading is an exceptional parameter for extracting driving events, as its accuracy was consistent across all four models developed. The study reveals that the Energy Maximization Algorithm's accuracy ranges from 56.80% (left lane change) to 85.20 % (right lane change) (lane-keeping) All four models developed had comparable accuracies to studies that used similar models. The 1D-CNN model had the highest accuracy (98.99%), followed by the LSTM model (97.75%), the RF model (97.71%), and the SVM model (97.65%). To serve as a ground truth, continuous signal data was annotated. In addition, the proposed method outperformed the fixed time window

approach. The study analyzed the overall pipeline's accuracy by penalizing the F1 scores of the ML models with the EMA's duration score. The pipeline's accuracy ranged between 56.8% and 85.0% overall.

The ultimate goal of this study was to extract variables from naturalistic driving videos that would facilitate an understanding of driver behavior in a naturalistic driving environment. To achieve this objective, three sub-goals were established. First, we developed a framework for extracting features pertinent to comprehending the behavior of natural-environment drivers. Using the extracted features, we then analyzed the car-following behaviors of various demographic groups. Thirdly, using a machine learning algorithm, we modeled the acceleration of both the ego-vehicle and the leading vehicle. Younger drivers are more likely to be aggressive, according to the findings of this study. In addition, the study revealed that drivers tend to accelerate when the distance between them and the vehicle in front of them is substantial. Lastly, compared to younger drivers, elderly motorists maintain a significantly larger following distance. This study's results have numerous safety implications. First, the analysis of the driving behavior of different demographic groups will enable safety engineers to develop the most effective crash countermeasures by enhancing their understanding of the driving styles of different demographic groups and the causes of collisions. Second, the models developed to predict the acceleration of both the ego-vehicle and the leading vehicle will provide enough information to explain the behavior of the ego-driver.

# CHAPTER 1: INTRODUCTION

## 1.1 Background

Road crashes results in more than a million fatalities worldwide each year; on average nearly four thousand people lose their lives every day on roads (ASIRT, 2021). It is predicted that road fatalities will continue to rise to become the fifth leading cause of death in the world by 2030 (Global Status Report on Road Safety, 2021). In fact, road traffic crashes are a leading cause of death in the United States for people aged 1-54 (ASIRT, 2021). Studies have shown that about 50% of fatal road accidents are due to unsafe driving behaviors (Global Status Report on Road Safety, 2021). Our ability to detect and characterize these unsafe behaviors in naturalistic driving settings and associate them with road accidents will be a major step toward developing effective crash countermeasures.

Large-scale naturalistic driving studies are intended to shed light on the factors that lead up to a crash and near-crashes. A review of the data collected from these studies can be used to extract detailed information about driver behavior, performance, and the environment that can be linked to crashes and near-collisions. Several studies have been conducted using naturalistic driving studies (NDS) datasets. Studies such as () have drawn insight into anomaly activities of drivers such as eating, dancing etc. Other studies have investigated car-following behaviors of drivers using NDS data. The NHTSA's 100-car NDS study (100-Car Naturalistic Driving Study, 2006) was the first of many (Antin et al. 2019; Barnard et al. 2018.; Charlton et al. 2019; Fridman et al. 2019; Larue et al. 2018) to collect video, radar, and vehicle telemetry data from a wide range of drivers while driving naturally. The data collection devices installed in the vehicles cost hundreds of thousands of dollars. Consequently, restricting the number of vehicles that could have the device installed and reducing the number of participants that can be involved in the studies.

In order to scale the data collection process to increase participants involvement, there is a need for alternative data-driven approaches that can collect and process high-quality, high-resolution, and high-fidelity data streams at an affordable price in order to find long-term solutions to these problems. The current state of transportation data collection relies on expensive systems that are difficult to scale for continuous data streaming. This endeavor's prohibitive cost could force organizations with limited resources to conduct infrastructure evaluations infrequently necessitating the need for low-cost alternatives to these high-end data collection systems that can produce comparable information. One of the aims of this study is to utilize technological advances in consumer-level devices such as smartphones to scale the collection of transportation data for continuous performance evaluation and decision-making.

Depending on video compression rates, the size of the data collected during NDS studies can range from hundreds of terabytes to several petabytes. To date, driving event extraction from NDS data has been accomplished through a combination of manual and semi-automated processes. As a result, relying on manual processing methods can be time-consuming and costly to scale. There is, therefore, a need for algorithms that can ingest multi-modal NDS data and accurately annotate various driving events useful for understanding crash causation. As a result, this dissertation develops an end-to-end, fully automated pipeline for detecting and analyzing driver maneuvers from naturalistic driving videos and kinematic data on a frame-by-frame basis. Our goal is to extract eight critical driving events for developing crash countermeasures: stop and lane-keeping events, left-right lane changes, left-right turning movements, and left-right horizontal curve maneuvers.

To gain a deeper understanding of the causes of vehicle crashes, it is necessary to link vehicle maneuver detection with the driver's driving behavior such as car-following behavior. This is because rear-ended crashes are the most frequent type of crashes encountered on the highway, according to statistical data pertaining to vehicle crashes. These crashes have a significant negative impact on the flow of traffic and typically have severe consequences (Stipancic et. al., 2017). Examining the sequence of events preceding rear-end crashes can be divided into two distinct scenarios. The first scenario is one in which the following distance between vehicles is so close that, even if the driver behind them can apply their brakes in time, a rear-end collision is possible if the car in front of them suddenly brakes. In the second scenario, the driver is maintaining a relatively safe distance behind the car in front of them; however, when the car in front of them applies the brakes, the driver is either distracted or fatigued and does not notice. This circumstance is comparable to one in which there is no braking whatsoever. This study aims to accurately model car-following behaviors that result in rear-end crashes in order to gain a better understanding of these scenarios from a practical standpoint.

## 1.2 Motivation

The cost of collecting NDS data is prohibitively high, precluding the continuous collection and analysis of data related to driver behaviors. In order to provide a long-term solution to the high-cost of data collection, there is a need for data-driven approaches that can collect and process quality, high-resolution, and high-fidelity streams of data at reasonable cost. The current state of transportation data collection relies on expensive systems that are not easily scalable for continuous data streaming. The cost-prohibitive nature of this effort could lead to infrequent driver behavior assessment for agencies with limited funding. Low-cost data collection alternatives that are able to produce information comparable to these high-end systems are needed. Therefore, there is a need to tap into advancements in consumer-level technologies such as smartphones to scale the collection of naturalistic driving data for continuous performance evaluation and decision making. The current generation of smartphones are enabled with a plethora of sensors such as accelerometers, gyroscopes, compass, and cameras. These sensors can be used to capture various dynamics of the transportation system. Many studies have used accelerometers, for example, to detect various driving maneuvers made by drivers in a naturalistic driving setting. Also, with the aid of computer vision and machine learning techniques, the smartphones video cameras and motion sensors can be used to understand the driving environment: extract vehicle trajectories, car-following behavior, understand lane-changing behaviors, estimate vehicle density, weather conditions, etc. (Aboah and Adu-Gyamfi 2020; Aleadelat et al. 2018; Robinson and Cook 2012; Zeng et al. 2018). The current study, therefore, seeks to explore the use of smartphone-based sensors for collecting NDS data at a cheaper cost.

Furthermore, despite significant progress and appreciable maneuver detection accuracies attained especially from ML-based algorithms, there are still open challenges that remain unsolved. First, all previous ML-based studies have treated vehicle maneuver detection as a time series classification problem. A major challenge that is not addressed by recent approaches is time series segmentation. This is an important step that should precede the development of ML classifiers: it separates any raw, continuous vehicle telemetry signal into a finite set of discrete events and anomalies with unique characteristics that can be used to train ML models for maneuver detection. The time series segmentation problem is straightforward if all the events contained in the continuous signal have a fixed duration: A simple, moving window with fixed time window could be used to define the start and end of each event. For NDS data however, the duration, frequency and amplitudes of events may vary significantly depending on the speed of the vehicle, type of sensor, driving behavior and type of event (lane change or turning movement). A robust time series segmentation algorithm is therefore needed to extract unique events that are needed to train and test ML algorithms for maneuver detection. Second, the robustness and transferability of models developed for maneuver detection have not been well tested: the size of data, number of drivers and events are usually not large enough to deduce the best performing models, or architectures needed for accurate detection of driver maneuvers. For example, Mandalia and Salvucci, (2005) and Yang et al. (2017) reports high accuracies for only 4-431 drivers driving 5-50 miles in the study. Studies have also evaluated these models on only one type of hardware acquisition systems: OBD or mobile phone.

Although a number of studies have been conducted to model the car-following behaviors of drivers, the majority of these studies have relied on simulated data that may not accurately represent incidents that occur in the real world. In addition, very few longitudinal studies have been conducted on the car-following behavior of drivers in naturalistic environments. These studies, however, are limited to developing models that can only estimate the acceleration of the ego-vehicle, which is insufficient to explain the behavior of the ego-driver. This limitation exists because data to model the acceleration of the leading vehicle is rarely available when using naturalistic driving dataset. As such, the current study attempts to address this issue by modeling both the acceleration of the ego-vehicle and the leading vehicle through the development of an AI framework capable of extracting parameters from NDS videos necessary to model the behavior (acceleration) of the leading vehicle as well as the ego-vehicle. In addition, there have been no previous longitudinal studies of the car-following behavior of various demographics in a naturalistic environment. Such research is important because it enhances our understanding of the driving styles of various demographic groups and the causes of crashes. The study addresses this deficiency by conducting longitudinal studies of demographic groups of drivers.

## 1.3 Research Contributions

The main goal of the study is to develop an AI framework for automatically extracting high-low features from the NDS dataset in order to explain driver behavior while using a low-cost data collection method. The author proposed three novel objectives for achieving the study's goal based on the identified research gaps. First, the study builds a low-cost data acquisition system for acquiring naturalistic driving data. Second, the study designs a framework that automatically extracts high to low-level features such as vehicle density, turning movements, lane changes, and other relevant features from the data collected from the developed data acquisition system. Third, the study extracted information from the NDS data to gain a better understanding of people's car-following behavior, and other driving behaviors through data collection and analysis in order to develop countermeasures for traffic safety.

The study has three broad objectives.

- The first objective is to build a low-cost data acquisition system for acquiring naturalistic driving data. The study's aim is to develop and implement a mobile data collection application for collecting naturalistic driving data. A modular software design process was followed to allow for future expansion of the mobile app. The top module is a high-level user interface that uses different layouts, menus, buttons, and notifications to create a user-friendly interface for a wide range of users. A sensor module manages all key technologies for data collection including GPS, Camera, Accelerometer and Gyroscope. The back-end module handles communications between the different modules, data storage and requests. In comparison to previous smartphone data collection apps, this current app is not overly reliant on internet connection to transmit data. This means that data can be gathered and stored temporarily in the app's library before being uploaded to the cloud server for storage when internet connectivity is available. This feature enables data collection from roadways in Wi-Fi dead zones. Also, the use of firebase backend enables data to be streamed from multiple sensors simultaneously and instantly. Recorded data can therefore be displayed instantly via web or other apps synced with the apps' database.
- The second objective is to be able to automatically extract high to low-level features such as vehicle density, turning movements, lane changes, driver distraction, and other relevant features from the data collected from the developed data acquisition system. In this objective, the study developed an end-to-end pipeline for automatic, frame-by-frame labelling of NDS videos

into various driving events by using vehicle telemetry data. To achieve this goal, we formulated the problem as a time series segmentation and classification problem. The segmentation task was achieved by developing a novel segmentation algorithm that utilizes the principle of energy maximization to detect the start and end of any driving event. Furthermore, the performance of both shallow and deep machine learning models for characterizing different types of driver maneuvers are evaluated using a large database of NDS data (200 hours of video and vehicle telemetry data) from three different studies: SHRP2 (Antin et al., 2019), Nebraska Medical Center (Drincic et al., 2018.), and a mobile application (Aboah et al., 2021). Annotating data from multiple sources enables us to evaluate the transferability of the segmentation and classification algorithms developed.

- The third objective is to gain a better understanding of people's car-following behavior, and other driving behaviors in order to develop countermeasures for traffic safety. To achieve this objective, we estimated the depth of objects in image frames using monocular depth estimation. The study's objective is to combine the estimated depth with other computer vision models to explain the car-following behaviors of different drivers in a naturalistic driving.

The rest of the dissertation is structured as follows. Chapter two presents the development of a low-cost data collection app. In this chapter, a smartphone data collection app was developed to enable data to be streamed from multiple smartphone-embedded sensors simultaneously and instantly. In addition, the app was designed to enable data collection from roadways in Wi-Fi dead zones unlike other existing apps. The live recorded data by the app can then be displayed instantly via the web or other apps synced with the apps' database. A modular design process was followed during the app's design to allow future expansion of the mobile app. Chapter three presents the study and results of driver maneuver detection and analysis using time series segmentation and classification. This chapter describes the development of an end-to-end pipeline for the automatic, frame-by-frame labeling of NDS videos into various driving events using vehicle telemetry data. To accomplish this objective, we formulated the problem as a time series segmentation and classification problem. Utilizing the principle of energy maximization, a novel segmentation algorithm was developed in order to accomplish the segmentation task. Chapter four presents the analysis of car-following behavior of ego-vehicles in a naturalistic driving setting. First, the study develops a framework for extracting features relevant to understanding driver behavior in a naturalistic environment. Additionally, the study analyzed the car-following behaviors of various demographic groups. To accomplish this objective, numerous visualization plots and statistical tests were conducted. We look into the factors that best explain the leading and ego-vehicle accelerations. Lastly, the study modeled the acceleration of both the ego-vehicle and the leading vehicle using a machine-learning algorithm. The study utilized the XGBoost algorithm to develop various acceleration models. The study presents its conclusion and recommendation in chapter five.

# CHAPTER 2: MOBILE SENSING FOR MULTIPURPOSE APPLICATIONS IN TRANSPORTATION

## 2.1 Introduction

Among today's transportation issues are congestion, safety, equity, aging infrastructure, energy, sustainability, and security among others. In order to find long-term solutions to these issues, there is a need for data-driven approaches that can collect and process quality, high-resolution, and high-fidelity streams of data at a reasonable cost. The current state of transportation data collection relies on expensive systems that are not easily scalable for continuous data streaming. The Virginia Department of Transportation (VDOT) for example, spends about $1.8 million per year on pavement data collection and evaluation using high-end machines (Sauerwein et al., 2011). The cost-prohibitive nature of this effort could lead to infrequent infrastructure assessments for agencies with limited funding. Low-cost data collection alternatives that are able to produce information comparable to these high-end systems are needed. The goal of this study is to tap into advancements in consumer-level technologies such as smartphones to scale the collection of transportation data for continuous performance evaluation and decision-making.

The developed mobile app will serve multiple purposes, including 1) a low-cost data acquisition system for naturalistic driving studies: this can be achieved by utilizing both front and back camera sensors of the smartphone to record both the driver's activities and the driving environment. 2) The telemetry data recorded by the gyroscope and accelerometer sensors could be used for pavement assessment, 3) the video data from the camera sensors could also be used to collect work zone data for safety analysis, etc.

The current generation of smartphones is enabled with a plethora of sensors such as accelerometers, gyroscopes, compasses, and cameras as shown in **Figure 2.1**. These sensors can be used to capture various dynamics of the transportation system. The accelerometers, for example, detect vibrations caused by a moving vehicle. These vibrations are used as a surrogate to estimate the roughness of the road. Previous studies have estimated the IRI using smartphone data, and the results are comparable to those obtained using high-end machines with an acceptable margin of error (Aboah & Adu-Gyamfi, 2020; Aleadelat et al., 2018; Sauerwein et al., 2011; Zeng et al., 2018). Numerous studies have developed various smartphone applications for collecting transportation data, but these applications are limited in their applications since they are not designed to serve multiple purposes (Aleadelat et al. 2018; Sauerwein et al. 2011; Zeng et al. 2018). Aleadelat et al. (2018) utilized a custom-built smartphone app to collect road surface data for the purpose of calculating IRI values. However, the app is not intended for any other function. The same can be said about Zeng et al. (2018).

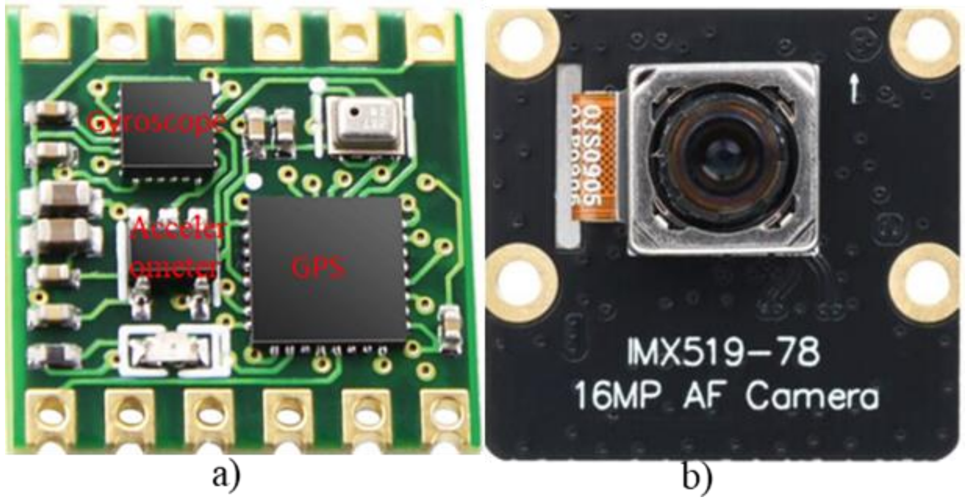


**Figure 2.1: Smartphone imbedded sensors a) A sensor board comprising of GPS, accelerometer, and gyroscope b) Camera sensor**

### 2.1.1 Objectives

In light of the limitations identified in previously developed mobile applications for transportation data collection, the current study aims to address these concerns. The study's aim is to

1. develop and implement a mobile data collection application for collecting transportation data for multiple applications including road condition evaluation, naturalistic driving studies, etc. To achieve this a modular software design process was followed to allow for future expansion of the mobile app. The top module is a high-level user interface that uses different layouts, menus, buttons, and notifications to create a user-friendly interface for a wide range of users. A sensor module manages all key technologies for data collection including GPS, Camera, Accelerometer and Gyroscope. The back-end module handles communications between the different modules, data storage and requests. In comparison to previous smartphone data collection apps, this current app is not overly reliant on internet connection to transmit data. This means that data can be gathered and stored temporarily in the app's library before being uploaded to the cloud server for storage when internet connectivity is available. This feature enables data collection from roadways in Wi-Fi dead zones. Also, the use of firebase backend enables data to be streamed from multiple sensors simultaneously and instantly. Recorded data can therefore be displayed instantly via web or other apps synced with the apps' database. The study further investigates the accuracy of the data collected by the app by synchronizing all sensor data.
2. Collect transportation data required for naturalistic driving studies, pavement evaluation, and road surface roughness evaluation. To achieve this goal, the app was

used to collect data including front and rear videos, telemetry data (gyroscope and accelerometer), and GPS data. The front and rear camera videos were used to analyze naturalistic driving studies, the accelerometer reading was used to develop a deep learning model to estimate the road surface roughness, and finally, the images from the front camera video were used to develop a machine learning model for pavement distress detection. The developed machine learning and deep learning models were assessed using accuracy, precision, and root-mean-percentage-squared-error.

The rest of this Chapter is divided into the following sections. The second section reviews relevant literature on smartphone applications in solving transportation related problems. Section three contains information about the development process, including the design approach, key components, and modulus. The fourth section discusses the data collection process as it relates to the developed mobile application. Section five summarizes the quantitative findings from the collection of field data. Finally, Section six summarizes the research, results the findings, and makes recommendations for future research in section seven.

## 2.2 Related Studies

The emergence of smartphone data collection has created new research opportunities in transportation. A number of past studies have relied on smartphone-collected data for a variety of research works that have produced cutting-edge findings. As such, this section discusses studies that examine the use of smartphone data to solve transportation-related problems. Each study is reviewed for its purpose, data collection technique, and methodology.

The technology to accurately assess pavement roughness with inexpensive sensors has improved greatly. A probe-based monitoring system for slippery and rough road surfaces was developed by MDOT in 2010 (Robinson & Cook, 2012). The vehicle data was collected and transmitted to a backend server running on a Droid platform. It was attached to the windshield in the same manner as a navigation device. Various sensors in the vehicle and on the phone were used to collect data, including the phone's three-axis accelerometer, the external road surface temperature and humidity, and the vehicle's Controller Area Network. During a two-year period, the system was installed in two vehicles driven by MDOT personnel. Over thirty thousand miles and more than 13 gigabytes of data were captured. To represent the pavement's surface irregularities, the vertical accelerometer signal's variation was employed. The accelerometer's sample rate is 100 Hz. Data collection was followed by calibration of the accelerometer readings using a PASER system curve fitting algorithm to the Pavement Surface Evaluation and Rating (PASER) scale. Future iterations of the curve fitting algorithm may incorporate data from the MDOT's annual PASER rating study. Also, researchers at Auburn University examined the use of vehicle-mounted sensors to determine the condition of road pavements (Dawkin, J., et. al. 2010.). The study's primary objective was to compute the IRI through the use of automotive sensors. The IRI was calculated using information gathered from a variety of vehicle sensors, including suspension deflection meters, accelerometers, and gyroscopes. On a 1.7-mile (2,750-meter) long test track, the National Center for Asphalt Technology conducted controlled speed tests (NCAT). The total number of vibrations in a particular section can be determined by calculating the Root Mean Square (RMS) of a signal measurement (i.e., vertical acceleration, gyroscopes, or suspension deflection). Acceleration includes a section on RMS (Root Mean Squared) (Section 3.2). Following that, the aggregated vibrations were compared to the pavement segment's true IRI. The root mean square of vertical accelerations was found to be the most accurate representation of the true IRI. It followed the same general trend as the well-known IRI, with the exception of a few expected magnitude changes. In summary, this study established that the most practical method for calculating the IRI is to use a root mean square algorithm on vertical acceleration readings. In a pilot study, Flintsch et al. (2012) quantified road ride quality and roughness using probing vehicles. Once again, vertical acceleration data was used to estimate vehicle vibration. At the Virginia Smart Road facility in Blacksburg, Virginia, a smoothness profile was created using an inertial-based laser profiler, and vertical accelerometer measurements were taken using an instrumented car. The frequency of the accelerometer was chosen to be 10 hertz. Additionally, GPS coordinates were collected. Acceleration data was collected during four runs on the test track. The study discovered that acceleration runs are highly reproducible. The smoothness profile and acceleration measurements are highly correlated, as determined by the coherence function analysis.

Papadimitriou et al. (2018) used smartphone data to detect and analyze risky driving behaviors. The study examined critical risk indicators such as the number of aggressive driving incidents and cell phone use while driving. The study gathered data on vehicle speed, distance traveled, accelerations, turning maneuvers, braking events, and cell phone use. The study's findings indicate that distraction caused by smartphone use has a significant effect on the number of severe events occurring per kilometer and, consequently, on the relative crash risk. Additionally, smartphone sensor data can be used to accurately detect mobile phone use while driving in more than 70% of cases. Another application of mobile apps is to enhance traffic control devices and to alleviate confusion among motorists passing through work zones. For example, a smartphone-based audio warning message (AWM) was proposed and tested in driving simulators to supplement conventional traffic controls and increase worker safety in work zones (Q. Li et al., 2016). The National Highway Traffic Safety Administration (NHTSA) and state transportation departments have implemented a variety of safety countermeasures aimed at reducing forward collisions in work zones. Traditional countermeasures, on the other hand, frequently fail to prevent crashes in work zones due to the complexity of traffic. Craig et al. (2017) conducted a study in Minnesota to determine the effect of in-vehicle messages on drivers' perceptions of work-zone events. Researchers at Texas Southern University's Innovative Transportation Research Institute developed a warning system application that alerts drivers to hazardous traffic situations through a variety of warning messages, including sound, visual, and voice (Dutzik et al., 2013; Rahman et al., 2016). Also, researchers at the University of Minnesota developed a smartphone app that uses embedded sensors and Bluetooth technology to provide pedestrians with routing instructions when upcoming work zones are detected. When a work zone is detected, the smartphone vibrates to alert users, and the app then broadcasts an audible message to them (Liao, 2014). Azadi et al. (2020) developed and deployed a work zone application for collecting, reporting, and storing real-time work-zone activity information. The study indicated that the precision of sensors such as GPS was within appreciable accuracy of work zone geolocation.

Although the aforementioned literature achieves great results using smartphone data, the data collection apps developed on these smartphones for the various data collection are not flexible enough to be used for other purposes; thus, there is a need for a smartphone app that serves multiple purposes. Also, as discussed in the literature, smartphone applications designed for pavement collection rely heavily on the internet and are difficult to use in areas without internet access. Therefore, it is necessary to develop a smartphone application that is not overly dependent on the internet and can be utilized in offline mode in regions without internet access. The present study aims to fill all these identified gaps.

## 2.3 Design Approach

The current app was designed to collect data from the onboard IOS sensors and video from the onboard camera and then transfer the data instantly to a cloud-based, real-time database. The current design is made up of three major modules: a frontend user interface module, a sensor module, and a backend module. **Figure 2.2** shows the interactions and information flow between these modules.

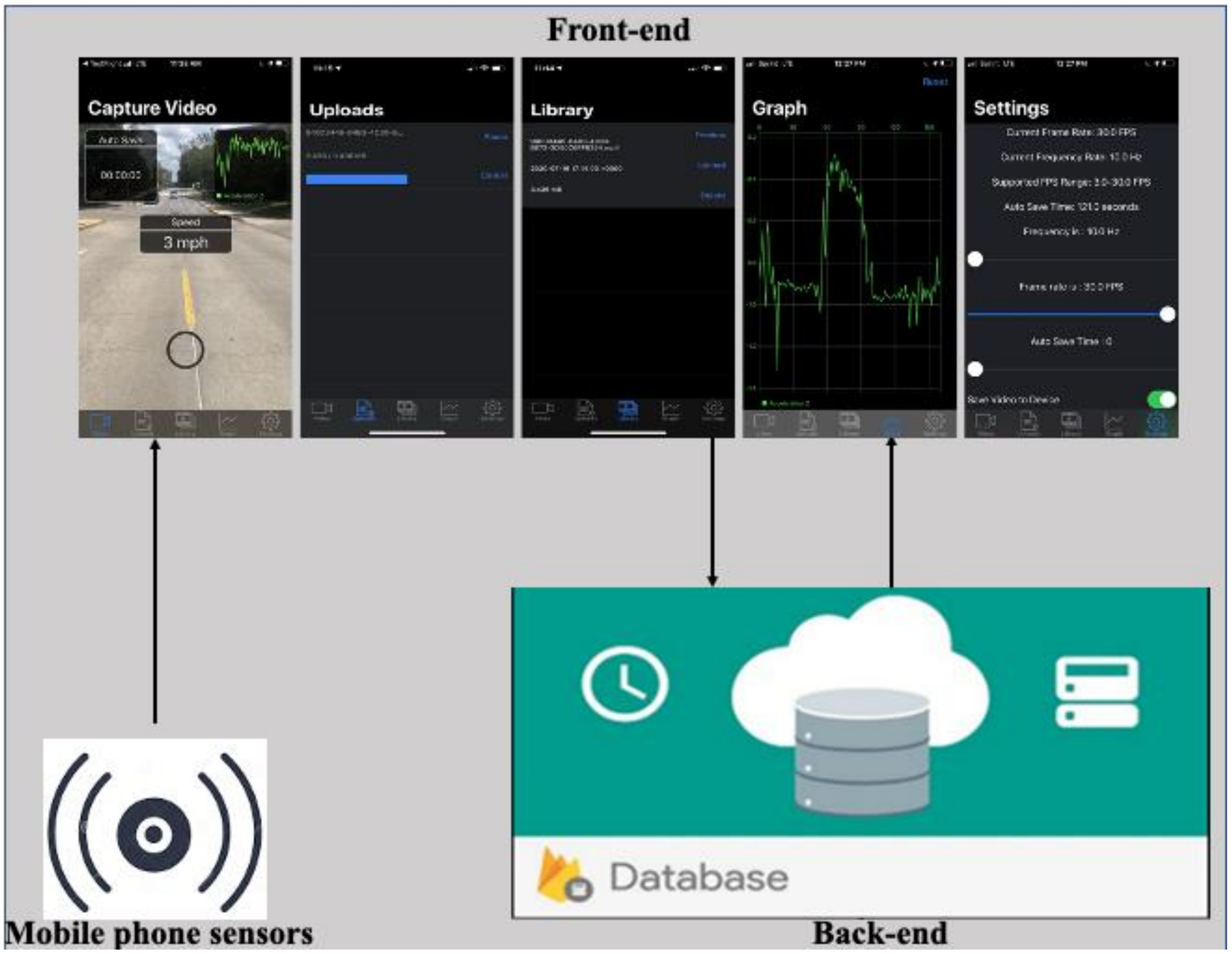
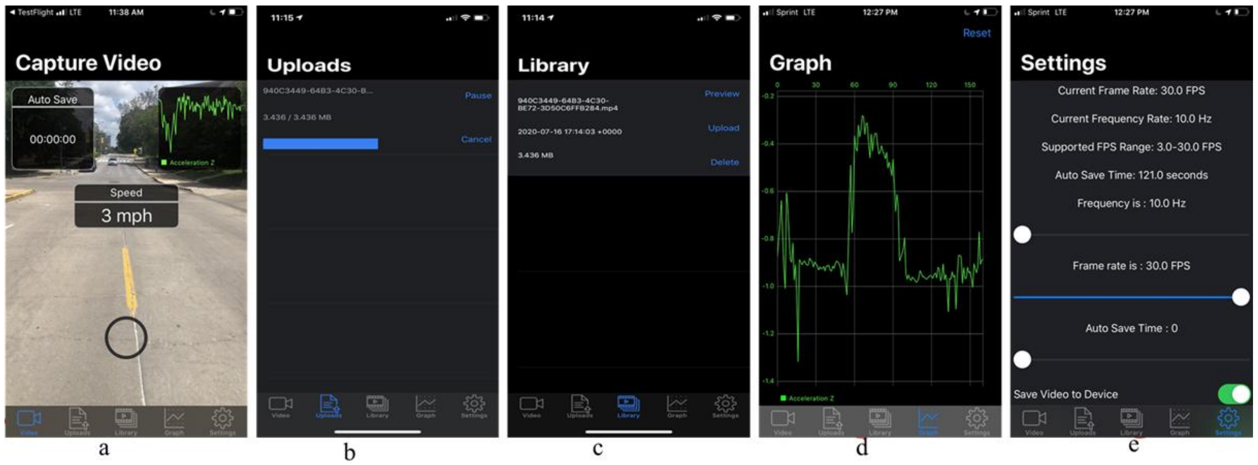


**Figure 2.2: Frontend, and backend modules of the App**

### 2.3.1 Frontend Design

The user interface for this app was designed to provide a streamlined view that exposed key aspects of the application via a tab bar controller. This enables us to compartmentalize the application's various critical components into distinct views. As illustrated in **Figure 2.3**, the tab bar contains a Video Tab, an Uploads Tab, a Library Tab, a Graph Tab, and a Settings Tab. The Video Tab enables the collection of video and sensor data. The Uploads Tab enables the user to keep track of the upload status of each package gathered in the Video Tab. The Library Tab displays all of the packages that have been collected and are currently stored on the device. The Graph Tab displays a live graph of the device's acceleration in the z direction as measured by its sensors. The Settings Tab displays all of the current settings and enables them to be modified as necessary. Adjustable frame rate (0-30fps), frequency rate or data sampling rate (0-15Hz), and auto-save time are included in the setting. The frame rate determines the quality of the captured video data. When the value is set to a large number, the size of the video data grows significantly, necessitating that the collecting device has a large amount of memory to temporarily store the video data before uploading them to the cloud. In addition, adjusting the sampling rate to a high frequency ensures that the app collects enough telemetry data. Utilizing a high sampling rate results in large data sizes, necessitating sufficient storage space on the mobile device to temporarily store them. Using a smaller sampling rate, on the other hand, has a negative impact on data quality but is generous in terms of file size. The autosave setting determines how frequently the user's data is saved. Using a longer autosave time will result in files that are large in size and may require a significant amount of storage space on the device before being uploaded to the cloud.



940C3449-64B3-4C30-B...
3.436 / 3.436 MB

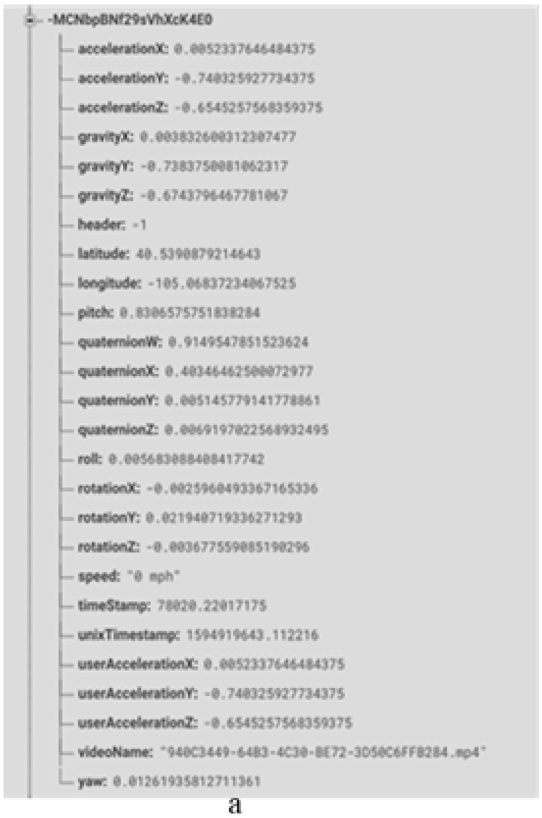
940C3449-64B3-4C30-BE72-3D50C6FFB284.mp4
2020-07-16 17:14:03 +0000
3.436 MB

Current Frame Rate: 30.0 FPS
Current Frequency Rate: 10.0 Hz
Supported FPS Range: 3.0-30.0 FPS
Auto Save Time: 121.0 seconds
Frequency is : 10.0 Hz
Frame rate is : 30.0 FPS
Auto Save Time : 0
Save Video to Device

**Figure 2.3: Key components of the app: (a) Video Tab, (b) Upload Tab, (c) Library Tab, (d) Graph Tab (e) Settings Tab**

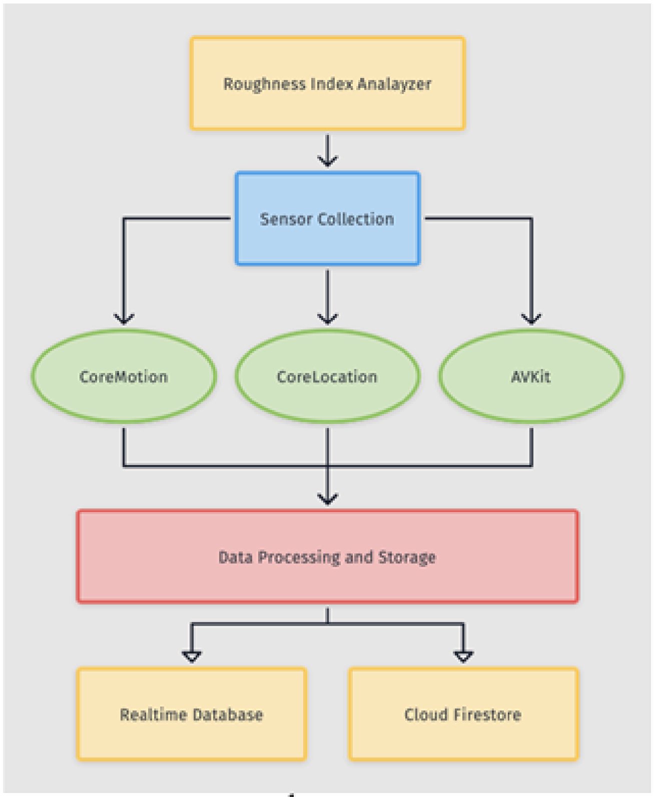
### 2.3.2 Backend Design

The primary function of the backend is to provide computational resources that can be used to accelerate front-end user query responses. The analytics performed on the front end of the application can be computationally expensive. To enable such sophisticated analytics on the front end of the app, we built a scalable, cloud-based backend using cutting-edge big data analytics techniques. The current study's backend is powered by Google Firebase. Firebase is made up of several parts, including a Realtime Database, a Cloud Firestore, and Cloud Storage. The Firebase Realtime Database is a relational database management system that runs in the cloud. The data is stored in JSON format and is synchronized with each connected client in real time. Similarly, the Cloud Firestore, maintains data consistency across client applications and enables offline support for mobile and web applications. Cloud Storage enables massive scalability of file storage. It allows users to upload and download files directly to and from the Cloud Storage "bucket". The developed application stores all sensor data in a Firebase Real-time Database and the videos in Cloud Firestore. The use of both the Firebase Realtime Database and Cloud Firestore ensure that uploaded data are made instantly accessible to other app users. Additionally, they allow multiple users to simultaneously push data to the cloud storage in real-time. Finally, the Cloud Firestore caches data that your app is actively using, allowing the app to write, read, listen to, and query data even when the device is not connected to the internet. The structure of the backend real-time database is shown in **Figure 2.4a**.



```
-MCNbpBNf29sVhXcK4E0
 ├─ accelerationX: 0.0052337646484375
 ├─ accelerationY: -0.740325927734375
 ├─ accelerationZ: -0.6545257568359375
 ├─ gravityX: 0.003832600312307477
 ├─ gravityY: -0.7383750081062317
 ├─ gravityZ: -0.6743796467781067
 ├─ header: -1
 ├─ latitude: 40.5390879214643
 ├─ longitude: -105.06837234067525
 ├─ pitch: 0.8306575751838284
 ├─ quaternionW: 0.9149547851523624
 ├─ quaternionX: 0.40346462500072977
 ├─ quaternionY: 0.005145779141778861
 ├─ quaternionZ: 0.0069197022568932495
 ├─ roll: 0.005683088408417742
 ├─ rotationX: -0.0025960493367165336
 ├─ rotationY: 0.021940719336271293
 ├─ rotationZ: -0.003677559085190296
 ├─ speed: "0 mph"
 ├─ timeStamp: 78020.22017175
 ├─ unixTimestamp: 1594919643.112216
 ├─ userAccelerationX: 0.0052337646484375
 ├─ userAccelerationY: -0.740325927734375
 ├─ userAccelerationZ: -0.6545257568359375
 ├─ videoName: "940C3449-64B3-4C30-8E72-3D50C6FF8284.mp4"
 └─ yaw: 0.01261935812711361
```

a



b

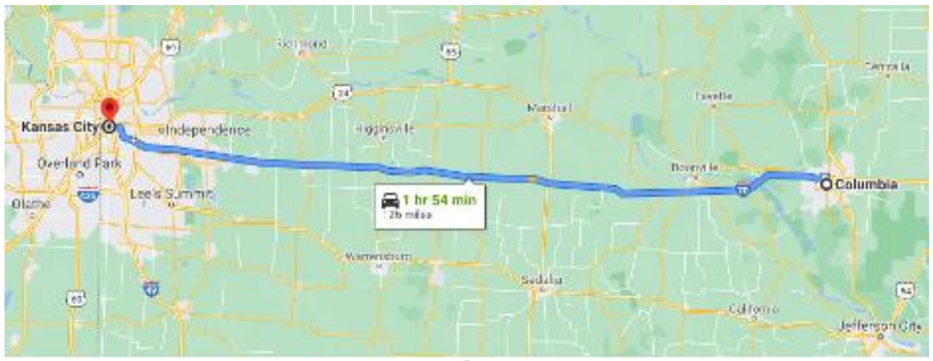
**Figure 2.4: (a) Real-time posting of data collected, (b) Sensor Module**

### 2.3.3 Sensor Module

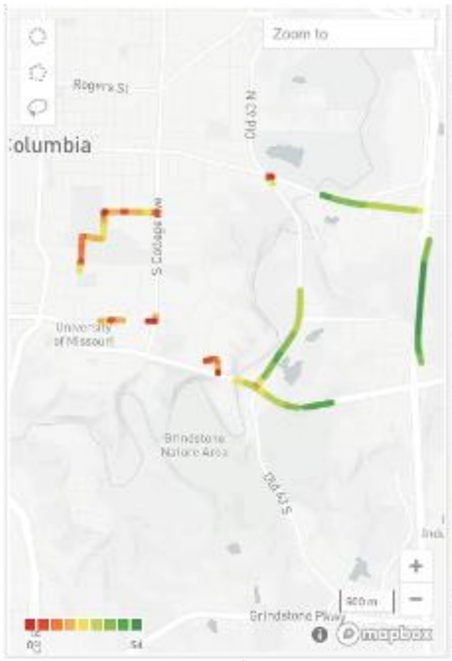
The developed app leverages multiple modules to allow for sensor collection. These modules include CoreMotion, CoreLocation, and AVKit as shown in **Figure 2.4b**. The CoreMotion module collects motion and environmental data from the onboard hardware of iOS devices, including accelerometers and gyroscopes, as well as the pedometer, magnetometer, and barometer. This framework enables the access and utilization of data generated by the hardware. The CoreMotion module provides access to both the unprocessed and raw values recorded by the hardware. The processed values do not contain any forms of bias that could have a negative impact on how you utilize the data. For instance, an accelerometer value that has been processed reflects only the acceleration caused by the user and not the acceleration caused by gravity. The CoreLocation determines the altitude, orientation, and geographical location of a device, as well as its position in relation to a nearby iBeacon device. The framework collects data by utilizing all available hardware components on the device, including barometer, GPS, Wi-Fi, magnetometer, and Bluetooth. The CoreLocation module has a location manager that is responsible for tracking large or minor changes in the user's current location with a configurable degree of accuracy, monitoring distinct regions of interest and generating location events when the user enters or leaves those regions, detecting and locating nearby beacons, and finally reporting heading changes from the onboard compass. Also, the AVKit provides a high-level interface for video content playback. AVKit module has a player view controller that allows you to play media full screen, embedded inline, or in a floating picture-in-picture window. It also includes a view controller, which displays content from a player and displays a native user interface for controlling playback. It finally consists of a protocol that defines the methods that must be implemented in order to respond to player view controller events. These various modules were utilized in this current app to access the device motion data (accelerometers and gyroscopes), the GPS location, and the video data respectively.

## 2.4 Data Collection Using Developed App

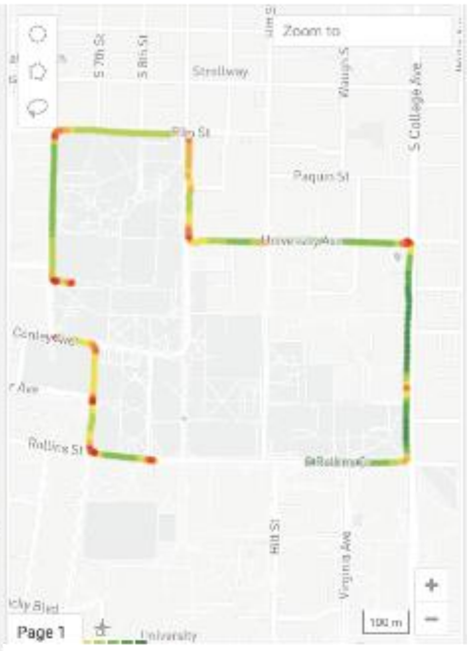
In order to test the various components of the app, we collected data on both freeways and local routes for a variety of applications. **Figure 2.5a** shows the study areas for which data was collected.  The first study was conducted on I-70 West, which connects Columbia, Missouri, and Kansas City, Missouri. The road segment considered for this test was approximately 45 miles in length. The second and third study were conducted in the city of Columbia as shown in **Figures 2.5b** and **2.5c.**



a



b



c

**Figure 2.5a-2.5c: The study area where the app was used to collect pavement roughness information**

To collect data with the app, the mobile phone must first be mounted on the car's windscreen (see **Figure 2.6b**) to record vehicle accelerations, rotations and some other pertinent information. The video data was sampled at a frame rate of 10 frames per second, whereas the accelerometer and vehicle location data were collected at a frame rate of 30 samples per second (30 Hz). As illustrated in **Figure 2.6a**, the vehicle used to collect data was a 2007 Nissan Sentra.

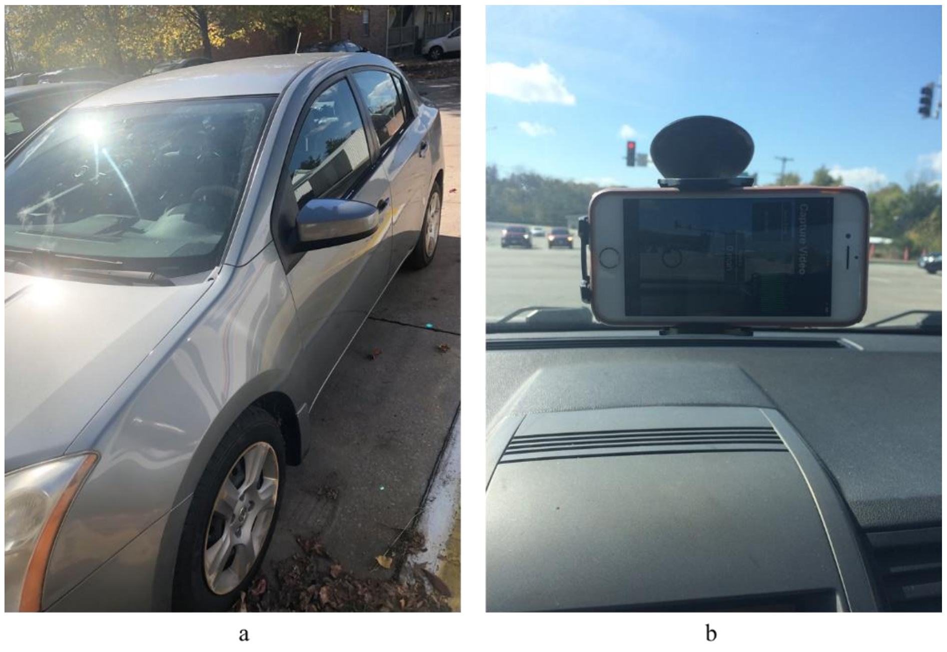


**Figure 2.6a-2.6b. a) 2007 Nissan Sentra vehicle used for the data collection b) The mounting position of the smartphone on the windscreen**

### 2.4.1 An Interactive User Interface for Data Querying using Streamlit

It is critical for the app to be able to sync the numerous pieces of information it collects. To accomplish this, the study used Streamlit to create an interactive user interface for integrating all of the various sensor data (Streamlit, 2020.). Streamlit is a Python library for web application development that is completely free and open-source. Streamlit is compatible with a number of well-known libraries and frameworks, including Keras, OpenCV, Vega-Lite, Pytorch, Tensorflow, and Python. The study used this library because of its ease of use and rapid deployment to create an interactive dashboard that enabled effortless data querying by timestamp.

The dashboard is divided into three sections: the homepage, data querying, and data visualization. Users can query both raw data and specific video frames by timestamp using the data querying page. Users can view changes in accelerometer readings while the vehicle is in motion on the data visualization page.

### 2.4.2 Evaluating the Accuracy of the GPS Coordinate Points in the App

The GPS coordinates must be precise in order to synchronize the video and accelerometer data from the app. This will allow us to map pavement distresses in real time at the exact locations where they occurred. Issues arise when GPS information is not accurate. **Figure 2.7** is a dashboard showing the synchronization of the various data. The dashboard is divided into five sections labeled A-E. Label A displays GPS data colored according to the vehicle's speed. The changes in the accelerometer readings are depicted in Label B, while the changes in the gyroscope readings are depicted in Label C. Label D displays the various trip identifiers, and Label E displays the speed profile. The GPS coordinates are found to be slightly off, which may affect the data collection's synchronization. The spikes in the z-coordinate of the gyroscope data correspond to the various turns made during the journey. Additionally, the orientation of the spikes indicated whether the turn was a left or a right. The spikes in the accelerometer reading (accelerometer z) correspond to extremely rough sections of the roadway.

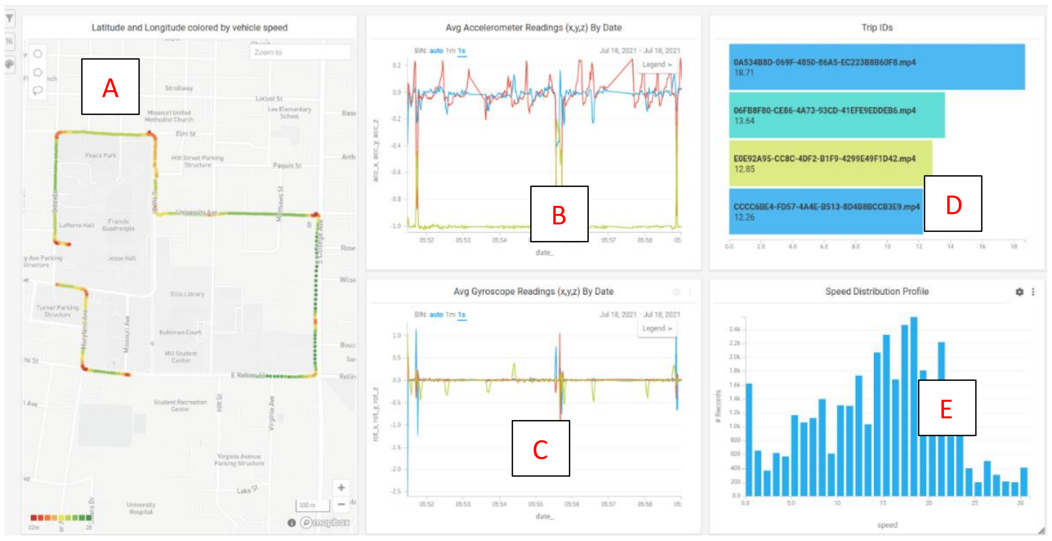

Latitude and Longitude colored by vehicle speed
Avg Accelerometer Readings (x,y,z) By Date
Trip IDs
Avg Gyroscope Readings (x,y,z) By Date
Speed Distribution Profile
A
B
C
D
E
0A534B8D-069F-4850-86A5-EC223B8B60F8.mp4 16.71
06FB8F80-CE86-4A73-93CD-41EFE9EDDEB6.mp4 13.64
E0E92A95-CC8C-4DF2-B1F9-4299E49F1D42.mp4 12.85
CCCC6BE4-FD57-4A4E-B513-8D4B8BCCB3E9.mp4 12.26
date_
speed
# Records

**Figure 2.7: A dashboard showing data synchronization**

## 2.5 Applications of Data Collected from The App

The app collects a variety of data, including accelerometer and gyroscope readings, roadway video, and GPS location data. This section discusses some of the possible uses for the app's data.

### 2.5.1 Estimating Road roughness index (IRI)

Several approaches have been used to develop relationships between accelerometer readings and actual IRI values of road sections. In some studies, the accelerometer readings were transformed into RMS values and then regressed against known IRI values for the road section [Hanson et al. (2014); Zeng et al. (2018)]. Few studies used deep learning approach to develop a model to assess the roughness of a road surface [Attoh-Okine (1994); Alinizzi et al. (2017); Cha et al. (2017); Hossain et al. (2019)]. In this study, a deep learning approach with entity embeddings was used to predict IRI values of road sections. Entity embedding allows for both continuous and categorical variables to be fitted with different functions unlike the traditional approach where both variables were fitted with the same function. This increases the learning ability of the categorical variable. The categorical variable used in this proposed model was the speed of the vehicle. The methodological framework is shown in Fig. 8 below.

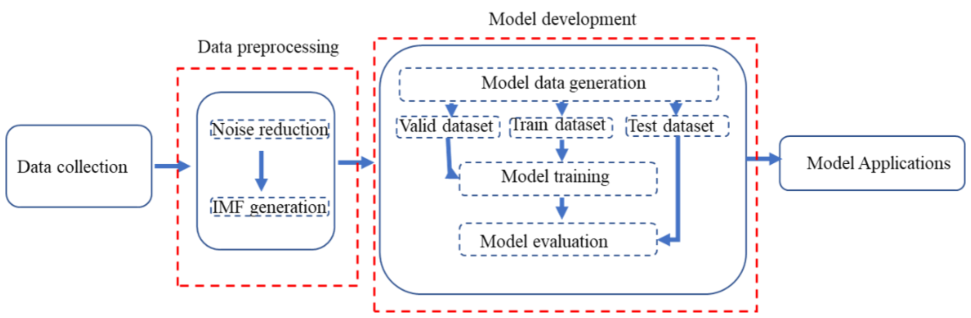


**Figure 2.8: Methodology flowchart**

*2.5.1.1 Data collection*

The accelerometer, gyroscope, and speed data collected by the app was used in the study to forecast IRI values for road segments. The smartphone was mounted on the vehicle's windscreen during data collection. The study also obtained ground truth IRI values from the MoDOT's ARAN viewer portal shown in **Figure 2.9**. The portal includes three tabs for pavement information: Condition, IRI, and Rut. Each tab contains records spanning the years 2009 to 2019. The portal includes a search box for locating roadways with relevant pavement information. To select a section of a roadway, the information for the road section's Start log (Begin log) and End log is entered in the search boxes shown in **Figure 2.9**.

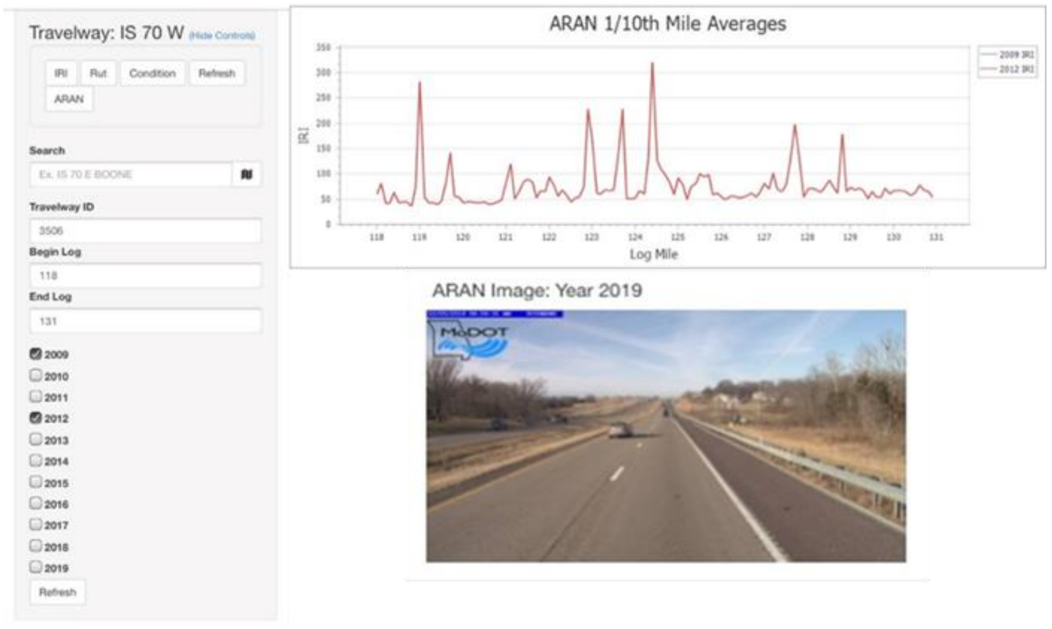
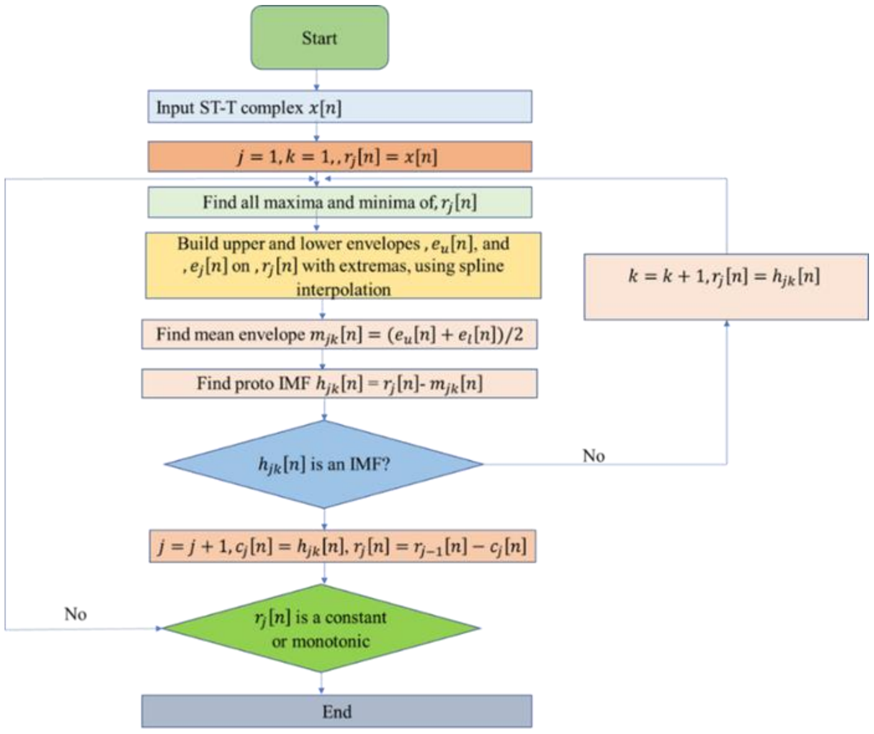


**Figure 2.9: The MoDOT ARAN viewer portal [8]**

*2.5.1.2 Data preprocessing*

The first step of the data preprocessing was reducing the noise in the accelerometer and gyroscope data using simple moving average. The accelerometer data together with the gyroscope information were processed using EMD. The EMD algorithm is used for analyzing nonstationary and nonlinear signals. It decomposes the input signal into several intrinsic mode functions (IMFs) through an iterative sifting procedure as summarized in the flowchart in **Figure 2.10**. The IMFs (shown in **Figure 2.11(a)**) presents low-high resolution information within the dataset. The low-resolution information usually characterizes general terrain information (hills and valleys). Roughness information is characterized by high–medium resolution IMFs. Further analysis of the high and medium IMFs resulting from the EMD yields information that is relevant for characterizing the relationship between accelerometer readings and the IRI-based pavement condition information. In this study, the range of frequencies contained in each IMF were further analyzed via the Fourier transform. **Figure 2.11(b)** is a power spectral density (PSD) showing the strength or power for the range of frequencies contained in the different IMFs. Different parameters were extracted from the PSD diagram to generate the dataset used for model training and validation. The parameters calculated included: the total area under the PSD, maximum power, the frequency corresponding to the maximum power, and other shown in Tables 2.1 and 2.2. A snapshot of the processed data that was used for the model development is shown in Table 2.1. Table 2.2 provides meanings to the column names. In Table 2.1, each row of the data represents averaged information per 1/10th of a mile.



Start
Input ST-T complex $x[n]$
$j = 1, k = 1,, r_j[n] = x[n]$
Find all maxima and minima of, $r_j[n]$
Build upper and lower envelopes , $e_u[n]$, and , $e_l[n]$ on , $r_j[n]$ with extremas, using spline interpolation
Find mean envelope $m_{jk}[n] = (e_u[n] + e_l[n])/2$
Find proto IMF $h_{jk}[n] = r_j[n]$- $m_{jk}[n]$
$h_{jk}[n]$ is an IMF?
No
$k = k + 1, r_j[n] = h_{jk}[n]$
$j = j + 1, c_j[n] = h_{jk}[n], r_j[n] = r_{j-1}[n] - c_j[n]$
No
$r_j[n]$ is a constant or monotonic
End

**Figure 2.10. A flowchart of EMD algorithm**

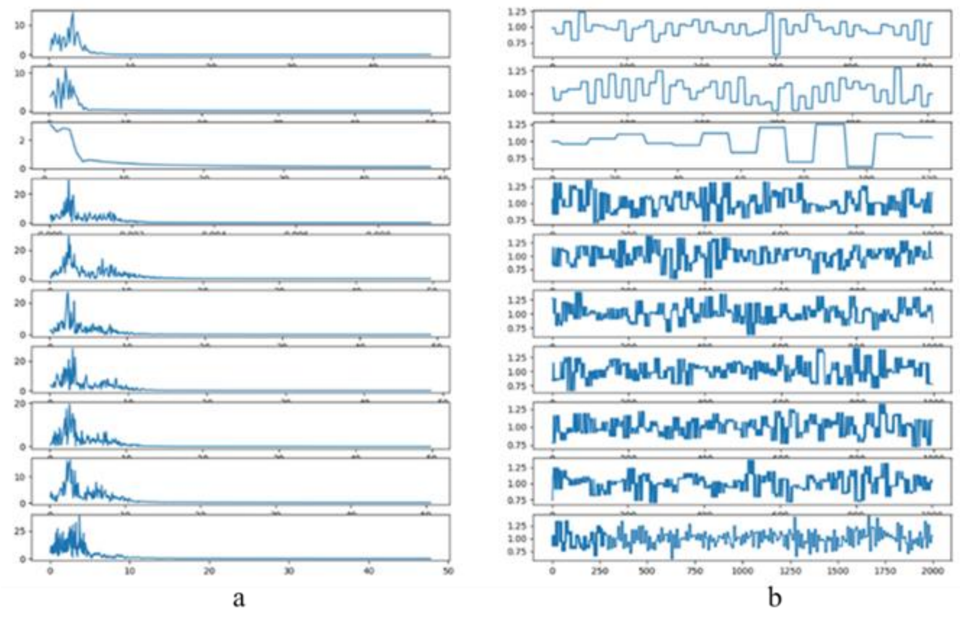
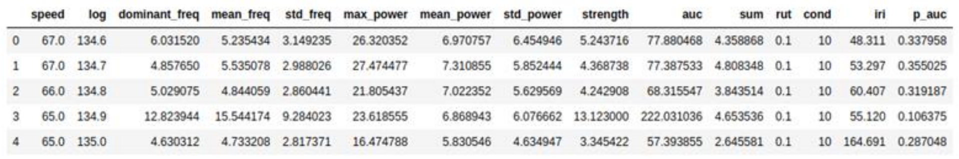


**Figure 2.11a – 2.11b. a) Plot of the highest frequency modes. b) Plot of IMFs after the EMD**

**Table 2. 1: Data for model development**



| | speed | log | dominant_freq | mean_freq | std_freq | max_power | mean_power | std_power | strength | auc | sum | rut | cond | iri | p_auc |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 67.0 | 134.6 | 6.031520 | 5.235434 | 3.149235 | 26.320352 | 6.970757 | 6.454946 | 5.243716 | 77.880468 | 4.358868 | 0.1 | 10 | 48.311 | 0.337958 |
| 1 | 67.0 | 134.7 | 4.857650 | 5.535078 | 2.988026 | 27.474477 | 7.310855 | 5.852444 | 4.368738 | 77.387533 | 4.808348 | 0.1 | 10 | 53.297 | 0.355025 |
| 2 | 66.0 | 134.8 | 5.029075 | 4.844059 | 2.860441 | 21.805437 | 7.022352 | 5.629569 | 4.242908 | 68.315547 | 3.843514 | 0.1 | 10 | 60.407 | 0.319187 |
| 3 | 65.0 | 134.9 | 12.823944 | 15.544174 | 9.284023 | 23.618555 | 6.868943 | 6.076662 | 13.123000 | 222.031036 | 4.653536 | 0.1 | 10 | 55.120 | 0.106375 |
| 4 | 65.0 | 135.0 | 4.630312 | 4.733208 | 2.817371 | 16.474788 | 5.830546 | 4.634947 | 3.345422 | 57.393855 | 2.645581 | 0.1 | 10 | 164.691 | 0.287048 |

**Table 2. 2: Input variables and meaning**

| Variable | Meaning (averaged per 1/10$^{th}$ of a mile) |
|---|---|
| Speed | The average speed of the vehicle |
| Dominant frequency | The frequency corresponding to the highest squared amplitude |
| Mean frequency | The average frequency for road section |
| Std_frequency | The standard deviation of the frequencies of the selected road section |
| Maximum power | The maximum of amplitudes squared for road section |
| Mean power | The average power for a selected road section |
| Std_Power | Standard deviation of the average power for a selected road section |
| Strength | The product of the dominant frequency and the maximum power |
| AUC | The total area under the PSD |
| SUM | Spatial acceleration |
| Rut | The Rut value of the road section from MODOT ARAN Viewer |
| Condition | The Condition value of the road section from MODOT ARAN Viewer |
| IRI | The IRI value of the road section from MODOT ARAN Viewer |

### *2.5.1.3 Model development*

The study utilizes a TabNet model from the Fastai libraries. Fastai is a deep learning framework built on top of PyTorch's libraries. The model was trained on an NVIDIA GTX

1080ti GPU. The number of training sample was 432. Due to the small size of the dataset, it was divided into ratios of 0.80:0.10:0.10, with each ratio representing the training, validation, and testing datasets, respectively. The root-mean-squared-percentage-error (RMSPE) metric was used to evaluate the performance of the developed model. The RMSPE was calculated by subtracting the predicted IRI value from the actual IRI value for each prediction and expressing the result as a percentage. **Figure 2.12** shows a convergence plot of the training and validation loss for the selected attributes. The plot demonstrates that the developed model is capable of rapidly learning the underlying relationships in the dataset. The RMSPE was approximately 0.174 after 18 iterations. This indicates that the error associated with predicting with this model was approximately 17.4 percent. The developed model had training and validation losses of 0.015 and 0.032, respectively.

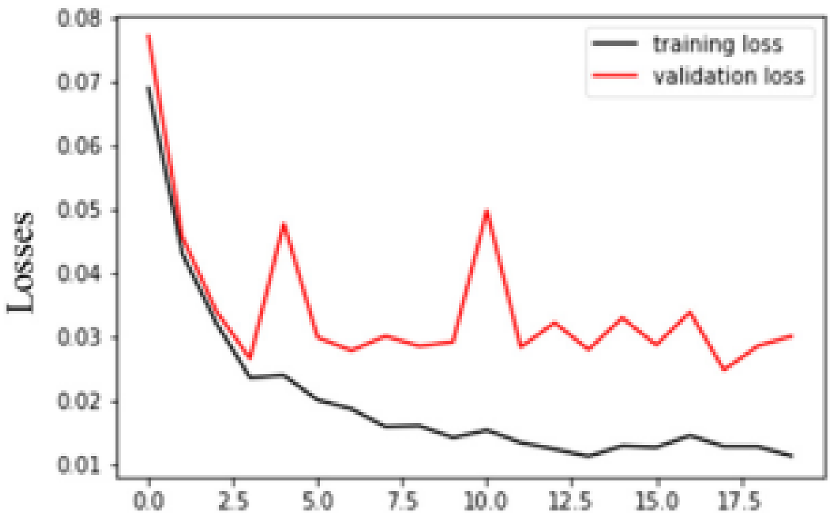


**Figure 2.12: A plot of training and validation losses of the proposed model**

The model was used to estimate the IRI values of the test data. The average-root-mean-squared-error (ARMSE) of the test prediction was 5.6. This means that our predictions are 5.6 units off the mark when compared to the actual IRI estimates provided by road profilers. Furthermore, the root-mean-square-percentage-error (RMSPE) metric indicates that the model's predictions were approximately 17% off target IRI values. As illustrated in **Figure 2.13a**, the road conditions' trends and amplitudes are correctly matched. As illustrated in **Figure 2.13b**, there is a linear relationship between the true and predicted IRI values. The r-squared value of the linear plot was 0.79. This indicates that the true and predicted IRI values are highly correlated.

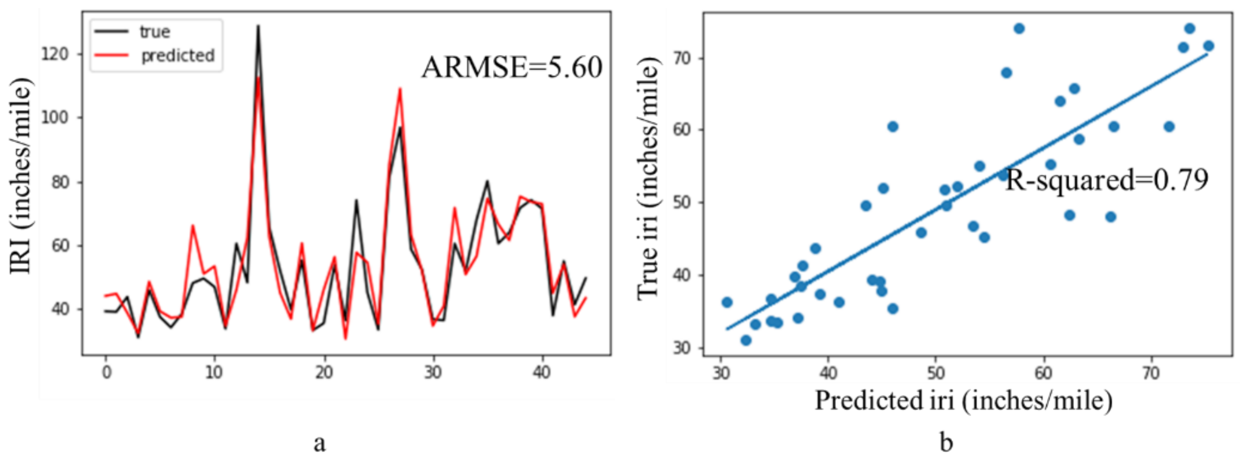


**Figure 2.13a and 2.13b. Proposed Model Performance: a). A Plot of True IRI values and Predicted IRI values and b). Scatter Plot of True IRI values against Predicted IRI values**

### *2.5.1.4 Model application*

The developed model will assist various state departments of transportation in assessing the roughness of road surfaces on a regular basis. This will assist them in prioritizing road maintenance operations.

### 2.5.2 Pavement Distress Detection

Pavement distresses pose a potential threat to the safety of road users. As a result, detecting distresses in a timely manner is regarded as one of the most important steps in limiting further degradation of pavement surfaces. To make the best use of financial resources, it is necessary to assess the condition of pavement surfaces on a regular basis and keep up with maintenance. As a result, the study developed a pavement distress detection model using video images collected by the develop app. We learned the visual and textual patterns associated with the various types of distress using a single-stage object detection algorithm, You Only Look Once (YOLOv5) algorithm. The framework for achieving this objective is shown in **Figure 2.14**.

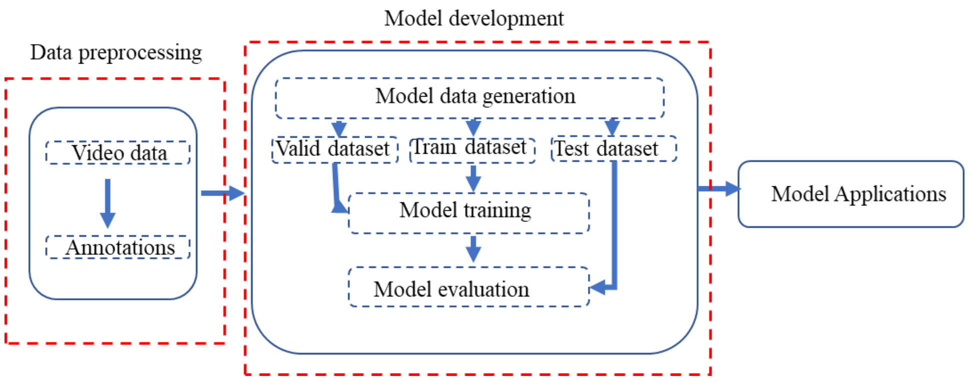


**Figure 2.14: Methodology flowchart**

*2.5.2.1 Data preprocessing*

As one drives with a smartphone mounted on the dashboard of the vehicle, the rear camera captures both the road surface and the vehicle's surroundings. The rear video footage is converted to still images. Distresses found on the images are then annotated with bounding boxes.

*2.5.2.2 Model development*

The state-of-the-art object detection model YOLOv5 was used in building the pavement distress model. YOLOv5 is the most recent version of the YOLO series and is a state-of-the-art single stage object detection algorithm. The YOLOv5 network is divided into three distinct components: the Backbone, the Neck, and the Head. The Backbone is a convolutional neural network that bundles and shapes image representational features at varying granularities. The neck of the architecture is composed of a series of layers that combine and integrate image representational features in order to advance to prediction. Similarly, the head makes use of neck-derived features and acquires box and class prediction functionality. Within YOLOv5, the CSPDarknet53 backbone contains 29 convolutional layers 3x3, a receptive field size of 725x725, and an overall parameter count of 27.6 M. Additionally, the SPP block attached to YOLO's CSPDarknet53 increases the proportion of receptive fields without impairing its operation. Similarly, feature aggregation is accomplished via PANet, which makes use of multiple backbone levels. YOLOv5 pushes the envelope of efficiency by incorporating features such as weighted residual connections, cross-stage partial connections, cross mini-batch, normalization, and self-adversarial training. In this study, we used the PyTorch framework to train and deploy our YOLOv5 model. To improve the YOLOv5 model's performance in detecting vehicles, the following hyperparameters are adjusted: 64-batch size, 0.0005 decay rate for the optimizer's weights, 0.01 initial learning rate, and 0.937 momentum.

The model was trained using 3,000 images obtained from the developed app and tested on 800 images. Four pavement distress types were annotated in this study. They are longitudinal crack (D00), transverse crack (D10), alligator crack (D20), and pothole (D40). The 3,000 images were distributed as follows: D00-900 samples, D10-836 samples, D20-614 samples, and D40-650 samples. We assessed the performance of the model using precision (P), F1 score (F1), and recall value (R). The F-1 score is the harmonic average of the recall and precision values. Precision is defined as the ratio of true positives (tp) to all predicted positives (tp+fp). Similarly, recall is the ratio of true positives to all true positives (tp+fn). The model was trained for 4-hours using an NVIDIA GTX 1080ti GPU.

After 1000 iterations, the results summary is shown in **Figure 2.15**. The training and validation losses converged after 500 iterations as shown in **Figure 2.15a** and **Figure 2.15b** respectively. Also, the best precision and recall values were obtained around the 500ith iteration shown in **Figure 2.15c** and **Figure 2.15d** respectively.

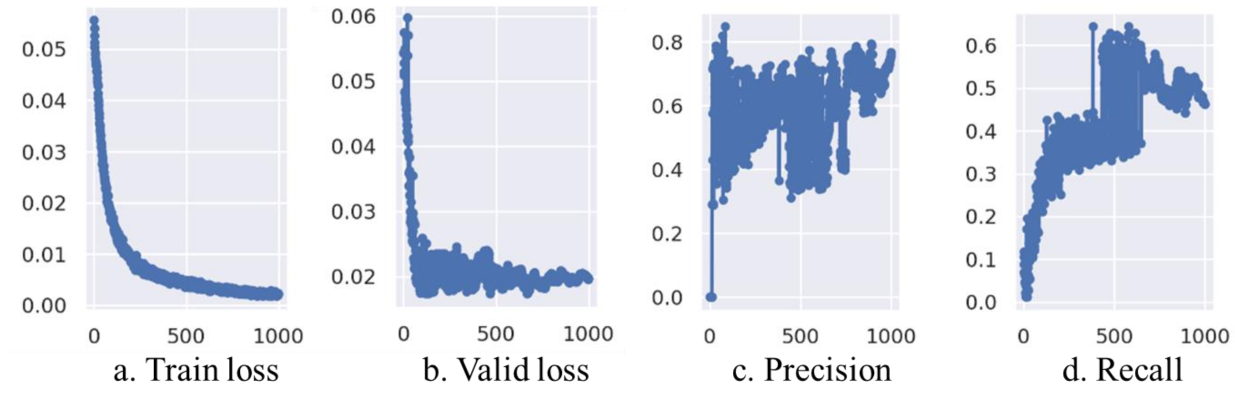


**Figure 2.15: Results from model training a) Training loss, b) Validation loss, c) Precision and d) Recall**

As shown in Table 2.3, the precision scores for all distress types of range between 0.65 and 0.82, while the recall values range between 0.58 and 0.61. Distress type D00 had the highest precision score of 0.82 and recall value of 0.61. The overall F1 score for the developed model was 0.68.

**Table 2. 3: Precision, Recall, F1-score values for the various distress types.**

| Distress Type | Precision | Recall | F1-score |
|---|---|---|---|
| D00 | 0.82 | 0.61 | 0.70 |
| D10 | 0.73 | 0.59 | 0.65 |
| D20 | 0.65 | 0.60 | 0.62 |
| D40 | 0.67 | 0.58 | 0.62 |

**Figure 2.16** shows detected pavement distresses from road images. Three pavement distresses were detected in this figure. They are designated as longitudinal crack (D00-red bounding box), alligator crack (D20-blue bounding box), and pothole (D40-green bounding box). The results show that the app can produce high resolution videos images that can be used by the current generation of machine learning algorithms for pavement evaluation.

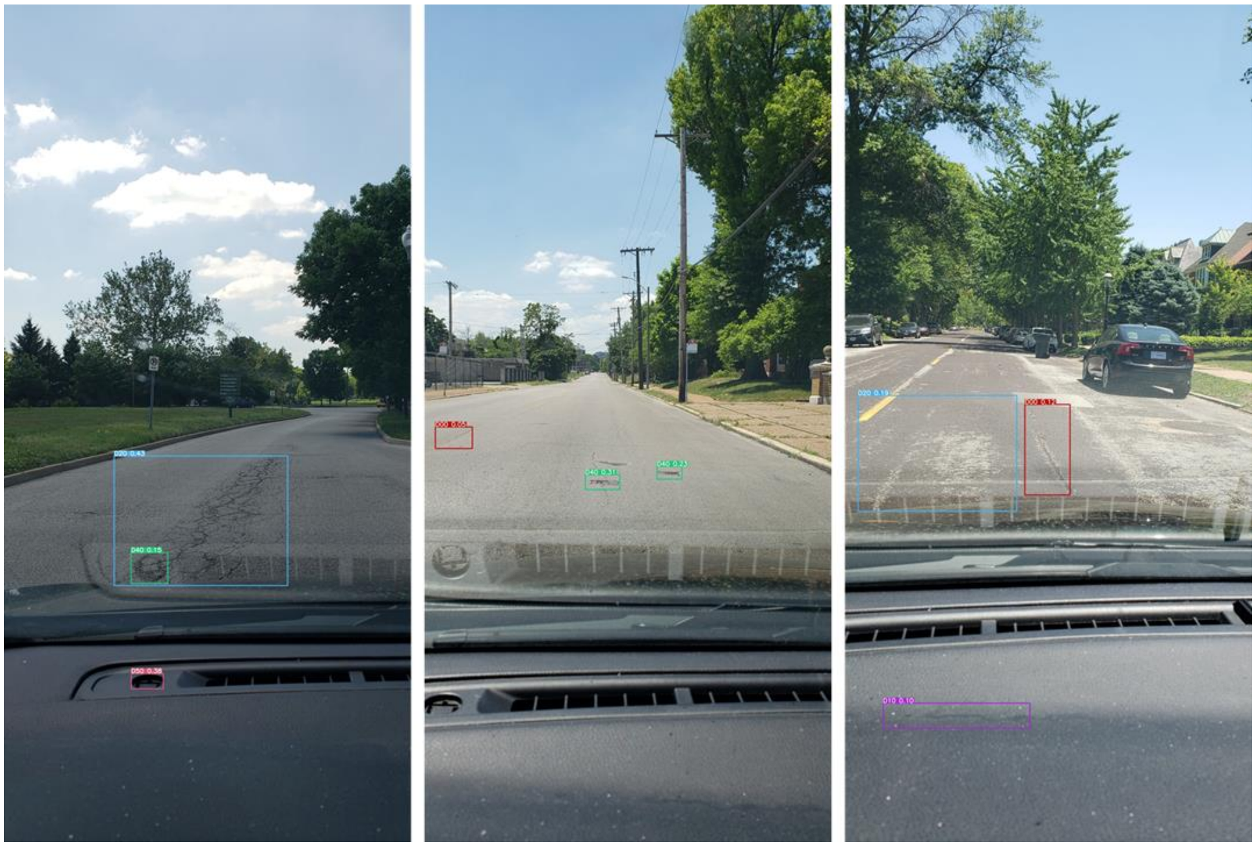


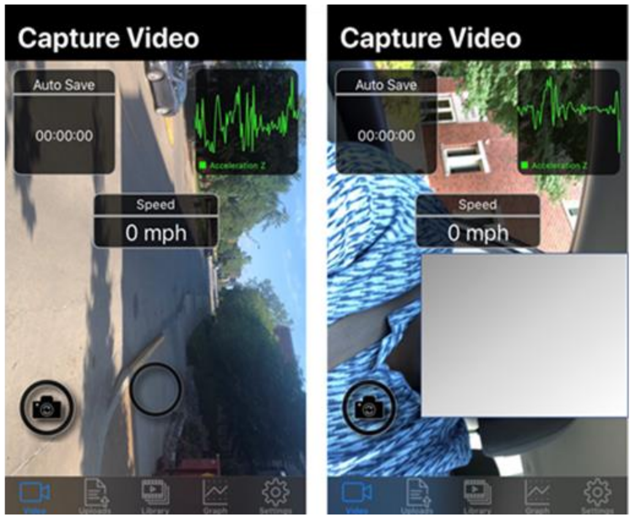
**Figure 2.16: Pavement distresses detected from video data**

### *2.5.2.3 Model Application*

The developed model will be beneficial to both the state and federal Departments of Transportation to detect and assess the severity of the detected distress. This will aid in the timely repair of death trapping distress on the roadway.

## 2.5.3 Naturalistic Driving Studies

Naturalistic driving studies (NDS) are cutting-edge research techniques that involve continuously recording driving data in real-world driving conditions using advanced instrumentation. NDSs enable the assessment of driving risks that would otherwise be impossible to assess using traditional crash databases or experimental methods (Guo, 2019). The NDS findings have a significant impact on policy making, safety research and development of safety countermeasures. The developed application in this study collects NDS data using iPhone’s dual camera system. One camera monitors the driver's environment outside the vehicle, while the other is used to record the driver's activities inside the vehicle as illustrated in **Figure 2.17a** and **2.17b**. This setup enables us to visually identify traffic incidents and correlate them to specific driving behaviors (driver sleeping, distracted, using phone, etc.).



a b

**Figure 2.17: a) Back camera recording activities outside the vehicle b) Front camera recording activities inside the vehicle (face of driver masked).**

The objective of this task is to develop a model for automatically detecting driving events from naturalistic driving videos. To achieve this objective, we formulated the problem as a time series segmentation and classification problem. The segmentation task was achieved by developing a novel segmentation algorithm that utilizes the principle of energy maximization to detect the start and end of any driving event from the telemetry data (gyroscope in the z-direction). This step was necessary to facilitate and expedite the annotation of driving events. All annotated driving events were standardized to a fixed length before feeding it to the model. Seven main classes of events were defined: lane changing (left and right), driver stopped, left turns, left curves, right turns, right curves, and lane-keeping. The model was trained with 86,000 training samples using NVIDIA GTX 1080ti GPU. We used a 50:50 split for model training and testing. The training data was fed through a multi-layered LSTM consisting of 30 hidden layers and 5 LSTM layers.

A convergence plot of the training and validation loss is shown in **Figure 2.18**. After 4000 iterations, the model's accuracy was approximately 94%. This implies that 94% of the time, the model can predict accurately the driving maneuvers of a vehicle. The training and validation losses of the developed model were 0.1402 and 0.2703 respectively. This means that the model is overfitting the training data, but it is not able to generalize correctly to validation data.

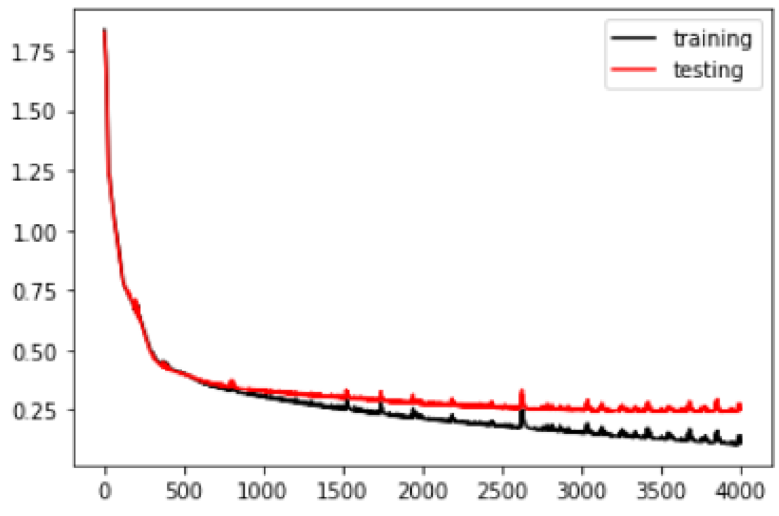


**Figure 2.18: A plot of training and validation losses of the Proposed model**

The developed models were evaluated using accuracy and precision. Accuracy was computed as the ratio of the true positive (TP) predictions to the sum of true positives (TP) and true negative (TN) predictions expressed as a percentage. Precision, on the other hand was computed as the number of true positives divided by the total number of true positives and false positives predictions.

From Table 2.4, precision scores range between 0.889 and 0.978 for all types of driving maneuvers, while recall values range between 0.898 and 0.962. Left turns achieved the highest precision of 0.978, while right turns achieved the highest recall of 0.962. Right lane changes, on the other hand, had the lowest precision score of 0.889, while left lane changes had the lowest recall score of 0.898. Overall, the model received an F1 score of 0.936. The precision, recall, and f1 score of all driving maneuvers are summarized in Table 4.

**Table 2. 4: Precision, Recall, F1-score values for the various driving events.**

| Driving Maneuvers | Performance matrix | | |
|---|---|---|---|
| | Precision | Recall | F1 score |
| Right turn | 0.945 | 0.921 | 0.932 |
| Left turn | 0.978 | 0.942 | 0.960 |
| Right curve | 0.969 | 0.962 | 0.965 |
| Left curve | 0.943 | 0.929 | 0.936 |
| Right lane change | 0.889 | 0.901 | 0.895 |
| Left lane change | 0.907 | 0.898 | 0.902 |
| Lane keeping | 0.949 | 0.938 | 0.943 |
| Stop | 0.961 | 0.944 | 0.952 |

## 2.6 Conclusion

This study developed and deployed a mobile application for collecting data for multiple applications including road condition evaluation, naturalistic driving studies, etc. The app collects road surface data via in-built smartphone sensors. The data can be streamed directly to a cloud-based database or stored in the app's library and uploaded to cloud storage at a later time if Wi-Fi is unavailable. The app's settings allow users to change the sampling frequency and frame rate during data collection. Additionally, while driving, the app is able to display the accelerometer readings on the video tab interface.

The application was developed using a modular approach. The design framework of this app is composed of three major modules: a frontend user interface (UI) component, a sensor component, and a backend component.

The app's functionality was evaluated by collecting data on the road surface on the I-70 W highway connecting Columbia, Missouri and Kansas City. The collected data was used to build a predictive model for estimating IRI values using a deep learning architecture. The accuracy of the model was used as a proxy for the quality of data collected by the app. ARMSE, R-squared, and RMSPE were used as metrics for evaluation. The model predictions indicate that the predicted IRI values from the smartphone data are comparable to those estimated using high-end machines such as the ARAN van. When the predicted IRI values were compared to the ground truth IRI values, a goodness-of-fit value of 0.79 was obtained. This demonstrates a high degree of correlation between them. Also, the video information from the app was used to identify pavement distresses. The pavement distress model was developed using state-of-the-art object detection model YOLOv5. The model achieves reasonably low training loss during the training of the model. The developed distress model demonstrated acceptably high predictive performance on the test dataset. While there were some false alarms, they were caused by dashboard cracks that resembled distress signals. Finally, information from the gyroscope and the accelerometer readings were used to determine the turning and lane-changing maneuvers of vehicles. The shapes of driving events were extracted from the gyroscope signal and annotated. The annotated data was used to train the vehicle maneuver, detection model. The model was developed using the LSTM architecture. The training results show that the model was able to learn to differentiate between the various signals associated with the various driving maneuvers. During training, the model's overall accuracy was 94%. When tested, the model's accuracy ranged from 93% to 97% for detecting various driving events.

In a nutshell, when tested, the developed mobile application for collecting road surface information and estimating road surface roughness demonstrated a high degree of potential for producing accurate and reliable results.

### 2.6.1 Limitations and Recommendations

The developed roughness app's future updates should address the following limitations.

- The first constraint is that the app was designed exclusively for the iOS operating system. Given the large number of Android users, the app's next update should include support for the Android operating system.
- Additionally, when used, the app does not provide IRI values directly. Future updates should incorporate the developed deep learning model into the backend, allowing the app to directly predict the IRI value of road sections in real-time while in use.
- Another limitation observed is that the GPS coordinates are slightly off as shown in **Figure 2.7**. Next update of the app will have an improved GPS.

## 2.7 References

Aboah, A., & Adu-Gyamfi, Y. (2020). Smartphone-Based Pavement Roughness Estimation Using Deep Learning with Entity Embedding. *Https://Doi.Org/10.1142/S2424922X20500072*, *12*(03n04), 2050007. https://doi.org/10.1142/S2424922X20500072

Aboah, A., Boeding, M., & Adu-Gyamfi, Y. (2021). *Mobile Sensing for Multipurpose Applications in Transportation*. http://arxiv.org/abs/2106.10733

Aleadelat, W., Asce, S. M., Ksaibati, K., Cameron, ;, Wright, H. G., & Saha, P. (2018). Evaluation of pavement roughness using an android-based smartphone. *Ascelibrary.Org*, *144*(3). https://doi.org/10.1061/JPEODX.0000058

Antin, J. F., Lee, S., Perez, M. A., Dingus, T. A., Hankey, J. M., & Brach, A. (2019). Second strategic highway research program naturalistic driving study methods. *Safety Science*, *119*, 2–10. https://doi.org/10.1016/J.SSCI.2019.01.016

*ASIRT — Association for Safe International Road Travel*. (n.d.). Retrieved November 16, 2021, from https://www.asirt.org/

Ayres, G., Wilson, B., & LeBlanc, J. (2004). Method for identifying vehicle movements for analysis of field operational test data. *Transportation Research Record, 1886*, 92–100. https://doi.org/10.3141/1886-12

Azadi, F., Adu-Gyamfi, Y., Sun, C., & Edara, P. (2020). Mobile Application Development and Testing for Work Zone Activity Real-Time Data Collection. *Transportation Research Record*, *2674*(6), 351–362. https://doi.org/10.1177/0361198120919118

Bakhit, P., Osman, O., Research, S. I.-T., & 2017, undefined. (2017). Detecting imminent lane change maneuvers in connected vehicle environments. *Journals.Sagepub.Com*, *2645*(1), 168-175. https://doi.org/10.3141/2645-18

Bejani, M., intelligent, M. G.-I. transactions on, & 2019, undefined. (n.d.). Convolutional neural network with adaptive regularization to classify driving styles on smartphones. *Ieeexplore.Ieee.Org*. Retrieved November 10, 2021, from https://ieeexplore.ieee.org/abstract/document/8643569/?casa_token=MvI_iKkT8acAAAAA:xBXpzeqhmsDPX-My6DrsfedZQhkTt0g2jOiHLgv5nHWvLWWSIGdhO-zxfEPAF3gfzbehmcYYKAk

Bhoraskar, R., Vankadhara, N., … B. R.-… systems and networks, & 2012, undefined. (n.d.). Wolverine: Traffic and road condition estimation using smartphone sensors. *Ieeexplore.Ieee.Org*. Retrieved November 10, 2021, from https://ieeexplore.ieee.org/abstract/document/6151382/?casa_token=VflO3GOrlLgAAAAA:nUyKZbgRZfU5VeoENl1N_6G8O8JaDu1sU-CNXxleRgYZuY9-US3hNWL1Xi8Bh08e2XC5CWEp1OY

Carvalho, E., Ferreira, B., … J. F.-… J. C. on, & 2017, undefined. (n.d.). Exploiting the use of recurrent neural networks for driver behavior profiling. *Ieeexplore.Ieee.Org*. Retrieved November 10, 2021, from https://ieeexplore.ieee.org/abstract/document/7966230/?casa_token=X4tGuD0nuS8AAAAA:vhxX1hMfrE7xmQWX3-RNr-JcU5bj0K2XboIvvTz_zVk0esElNjYBoE076TR-JB6oBwKRX3ViKeE

Cortes, C., learning, V. V.-M., & 1995, undefined. (1995). Support-vector networks. *Springer*, *20*, 273–297. https://link.springer.com/article/10.1007/BF00994018

Craig, C., Achtemeier, J., Morris, N., Tian, D., & Patzer, B. (2017). *In-vehicle work zone messages*. https://conservancy.umn.edu/handle/11299/189538

*Dawkin, J., Bevly, D., Powell, B., and Bishop, R.... - Google Scholar*. (n.d.). Retrieved December 30, 2021, from https://scholar.google.com/scholar?hl=en&as_sdt=0%2C26&q=Dawkin%2C+J.%2C+Bevly%2C+D.%2C+Powell%2C+B.%2C+and+Bishop%2C+R.+%282010%29.+Investigation+of+pavement+maintenance+applications+of+intellidrive+SM%2C+CTS+PFS+Report&btnG=

Drincic, A., Rizzo, M., Desouza, C., Health, J. M.-D. D., & 2020, undefined. (n.d.). Digital health technologies, diabetes, and driving (meet your new backseat driver). *Elsevier*. Retrieved January 1, 2022, from https://www.sciencedirect.com/science/article/pii/B978012817485200016X

Dutzik, T., Madsen, T., & Baxandall, P. (2013). *A new way to go: The transportation apps and vehicle-sharing tools that are giving more americans the freedom to drive less*. https://trid.trb.org/view/1264807

Flintsch, G. W., Valeri, S. M., Katicha, S. W., de Leon Izeppi, E. D., & Medina-Flintsch, A. (2012). Probe vehicles used to measure road ride quality: Pilot demonstration. *Transportation Research Record*, *2304*, 158–165. https://doi.org/10.3141/2304-18

Freund, Y., icml, R. S.-, & 1996, undefined. (1996). Experiments with a new boosting algorithm. *Citeseer*. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.6252&rep=rep1&type=pdf

*global status report on road safety - Google Search*. (n.d.). Retrieved November 16, 2021, from https://www.google.com/search?q=global+status+report+on+road+safety&sxsrf=AOaemvKQh0Ha156Iyw31CViTErkRqsxp1Q%3A1637159246232&source=hp&ei=ThGVYaCcC8eQtAbXqLrwCw&iflsig=ALs-wAMAAAAAYZUfXloqwUz6MwX4HSs8BgDRKfcMI4H8&gs_ssp=eJzj4tVP1zc0TMvKM08pLzMyYPRSTs_JT0rMUSguSSwpLVYoSi3ILypRyM9TKMpPTFEoTkxLLakEAMk2EjU&oq=Global+status+report+on+road+safety&gs_lcp=Cgdnd3Mtd2l6EAEYADIFCC4QgAQyBQgAEIAEMgUIABCABDIFCAAQgAQyBQgAEIA

EMgUILhCABDIFCAAQgAQyBQgAEIAEMgUIABCABFAAWABg0hxoAHAAeACAAVKIAVKSAQExmAEAoAECoAEB&sclient=gws-wiz

Guo, F. (2019). Statistical methods for naturalistic driving studies. *Annual Review of Statistics and Its Application*, *6*, 309–328. https://doi.org/10.1146/ANNUREV-STATISTICS-030718-105153

Houenou, A., Bonnifait, P., Cherfaoui, V., & Yao, W. (2013). Vehicle trajectory prediction based on motion model and maneuver recognition. *IEEE International Conference on Intelligent Robots and Systems*, 4363–4369. https://doi.org/10.1109/IROS.2013.6696982

Júnior, J. F., Carvalho, E., Ferreira, B. v., de Souza, C., Suhara, Y., Pentland, A., & Pessin, G. (2017). Driver behavior profiling: An investigation with different smartphone sensors and machine learning. *PLoS ONE*, *12*(4). https://doi.org/10.1371/JOURNAL.PONE.0174959

Kumar, P., Perrollaz, M., … S. L.-2013 I. I., & 2013, undefined. (n.d.). Learning-based approach for online lane change intention prediction. *Ieeexplore.Ieee.Org*. Retrieved November 2, 2021, from https://ieeexplore.ieee.org/abstract/document/6629564/?casa_token=34owklZuo6IAAAAA:2hGYVxddfNWTWoFIWL42SwTCk8kbO-QMZREVPVZMABT71cAATsw62C1XW3E6yNe-wvmMCZG_QKc

Li, P., Abdel-Aty, M., & Islam, Z. (2021). Driving Maneuvers Detection using Semi-Supervised Long Short-Term Memory and Smartphone Sensors. *Transportation Research Record: Journal of the Transportation Research Board*, *2675*(9), 1386–1397. https://doi.org/10.1177/03611981211007483

Li, Q., Qiao, F., Proc., L. Y.-, Congress, 23rd ITS World, & 2016, undefined. (2016). Performance Measures of Smartphone Warning Messages in Work Zones and Intersections. *Researchgate.Net*. https://www.researchgate.net/profile/Qing-Li-89/publication/303750794_Performance_Measures_of_Smartphone_Warning_Messages_in_Work_Zones_and_Intersections/links/57505b6008aefe968db72bed/Performance-Measures-of-Smartphone-Warning-Messages-in-Work-Zones-and-Intersections.pdf

Liao, C.-F. (2014). *Development of a navigation system using smartphone and bluetooth technologies to help the visually impaired navigate work zones safely*. https://conservancy.umn.edu/handle/11299/163204

Mandalia, H. M., & Salvucci, D. D. (2005). Using support vector machines for lane-change detection. *Proceedings of the Human Factors and Ergonomics Society*, 1965–1969. https://doi.org/10.1177/154193120504902217

Miller, R., Technical, G. S.-19th I., & 2005, undefined. (n.d.). Determination of Lane Change Maneuvers Using Naturalistic Driving Data. *Www-Esv.Nhtsa.Dot.Gov*.

Retrieved November 1, 2021, from https://www-esv.nhtsa.dot.gov/Proceedings/19/05-0337-O.pdf

Morris, B., Doshi, A., & Trivedi, M. (2011). Lane change intent prediction for driver assistance: On-road design and evaluation. *IEEE Intelligent Vehicles Symposium, Proceedings*, 895–901. https://doi.org/10.1109/IVS.2011.5940538

Nhtsa, U. (1999). *Evaluation of the intelligent cruise control system: volume 1: study results*. https://rosap.ntl.bts.gov/view/dot/4273

Ohn-Bar, E., Tawari, A., Martin, S., & Trivedi, M. M. (2014). Predicting driver maneuvers by learning holistic features. *IEEE Intelligent Vehicles Symposium, Proceedings*, 719–724. https://doi.org/10.1109/IVS.2014.6856612

Papadimitriou, E., Tselentis, D., Transport, G. Y.-P. of 7th, & 2018, undefined. (n.d.). Analysis of driving behaviour characteristics based on smartphone data. *Nrso.Ntua.Gr*. Retrieved December 30, 2021, from https://www.nrso.ntua.gr/geyannis/wp-content/uploads/geyannis-pc289-1.pdf

Perslev, M., Jensen, M. H., Darkner, S., Jennum, P. J., & Igel, C. (n.d.). U-time: A fully convolutional network for time series segmentation applied to sleep staging. *Arxiv.Org*. Retrieved January 2, 2022, from https://arxiv.org/abs/1910.11162

Rahman, R., Fengxiang Qiao, M., Li, Q., & Yu, L. (2016). *Developing a smartphone based warning system application to enhance the safety at work zones*. https://rosap.ntl.bts.gov/view/dot/32557

Robinson, R., & Cook, S. J. (2012). *Slippery road detection and evaluation*. https://deepblue.lib.umich.edu/handle/2027.42/117505

Sauerwein, P. M., Smith, B. L., & Studies, U. of Virginia. C. for T. (2011). *Investigation of the implementation of a probe-vehicle based pavement roughness estimation system.* https://doi.org/10.21949/1503647

Toledo, Tomer, Zohar, & David. (n.d.). *TRR 1999*. https://doi.org/10.3141/1999-08

TR567, L. B.-U. B., & 1999, undefined. (1999). Random forests. *Machinelearning202.Pbworks.Com*. http://machinelearning202.pbworks.com/w/file/fetch/60606349/breiman_randomforests.pdf

Xuan, Y., & Coifman, B. (2006). Lane change maneuver detection from probe vehicle DGPS data. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 624–629. https://doi.org/10.1109/ITSC.2006.1706811

Yang, D., Qiu, X., Liu, Y., Wen, C., Zhu, L., & Hong, X. (2017). *Modeling the Discretionary Lane-Changing Decision Behavior on Freeways Using Random Forest Theory*. https://trid.trb.org/view/1438691

Yu, J., Chen, Z., Zhu, Y., Chen, Y., … L. K.-I. transactions on, & 2016, undefined. (n.d.). Fine-grained abnormal driving behaviors detection and identification with smartphones. *Ieeexplore.Ieee.Org*. Retrieved November 10, 2021, from https://ieeexplore.ieee.org/abstract/document/7593222/?casa_token=oJaKFhLtCfcAAAAA:ySQwRv4Vi3uh2oMqKTbg2nR32JsOr0Splaqbmlx36M0hfqnWP4i4ihXIrRwpkOvI1j1vlQXB9L0

Zeng, H., Park, H., Smith, B., Engineering, E. P.-K. J. of C., & 2018, undefined. (2018). Feasibility assessment of a smartphone-based application to estimate road roughness. *Springer*, *22*(8), 3120–3129. https://doi.org/10.1007/s12205-017-1008-9

Zheng, J., Suzuki, K., & Fujita, M. (2014). Predicting driver's lane-changing decisions using a neural network model. *Simulation Modelling Practice and Theory*, *42*, 73–83. https://doi.org/10.1016/J.SIMPAT.2013.12.007

# CHAPTER 3: DRIVER MANEUVER DETECTION AND ANALYSIS USING TIME SERIES SEGMENTATION AND CLASSIFICATION

## 3.1 Introduction

Road crashes result in more than a million fatalities worldwide each year; on average nearly four thousand people lose their lives every day on roads (ASIRT, 2021). It is predicted that road fatalities will continue to rise to become the fifth leading cause of death in the world by 2030 (Global Status Report on Road Safety, 2021). In fact, road traffic crashes are a leading cause of death in the United States for people aged 1-54 (ASIRT, 2021). Studies have shown that about 50% of fatal road accidents are due to unsafe driving behaviors (Global Status Report on Road Safety, 2021). Our ability to detect and characterize these unsafe behaviors in naturalistic driving settings and associate them with road accidents will be a major step toward developing effective crash countermeasures.

Large-scale naturalistic driving studies are designed to provide insight into pre-crash causal and contributing factors. A review of the data collected from these studies can be used to extract detailed driver behavior, performance, environment information that can be associated with crashes and near crashes. NHTSA's 100-car NDS study (100-Car Naturalistic Driving Study, 2006) was the first of many (Antin et al. 2019; Barnard et al. 2016.; Charlton et al. 2019; Fridman et al. 2019; Larue et al. 2018) to simultaneously collect video, radar, and vehicle telemetry data from a large variety of drivers under naturalistic driving settings. To leverage this data for developing effective crash countermeasures, there is a need to annotate different types of driving events or behaviors and associate them with critical incidents, near crashes or crashes. To date, the process of driving event extraction from NDS data has been performed by a mixture of manual and semi-automated processes. The size of these datasets typically ranges between hundreds of terabytes to several petabytes depending on video compression rates. Therefore, relying on manual processing methods can be labor intensive and very expensive to scale. There is a need to develop algorithms that can ingest multi-modal NDS data and accurately annotate different driving events useful for understanding crash causality. As a result, this paper develops an end-to-end, fully automated pipeline for frame-by-frame detection and analysis of driver maneuvers from naturalistic driving videos and kinematic data. Our aim is to extract eight driving events that are critical in developing crash countermeasures: stop and lane keeping events, left-right lane changes, left-right turning movements and left-right horizontal curve maneuvers.

Existing algorithms developed for extracting driving maneuvers from NDS data can be grouped into two main categories: rule-based, pattern matching and or machine learning approaches. A rule-based algorithm is a collection of decision rules that facilitate the detection of various driving events. For instance, to distinguish between an aggressive turn and a normal turn, consider the following: if the vehicle's heading is greater than 30 degrees, the turn is considered aggressive; otherwise, the turn is considered normal (Saiprasert et al. 2013). Additionally, (Panichpapiboon and Leakkawn, 2020) used a rule-based algorithm to detect lane changing events. The study indicates that the lane change maneuvers will occur in three phases. The first phase is lane departure, followed by the "into" phase during which the vehicle enters the new lane, and finally the lane keeping phase during which the vehicle returns to its original position. An advantage of using this approach is that it does not require labeling of the dataset. The second class of algorithms used are pattern recognition or matching based. Algorithms are designed to extract driving events in vehicle telemetry data through a matching process against a ground-truth database of referenced driving maneuvers. The matching process is usually implemented via dynamic time warping (DTW), which compares the similarity of an incoming signal to that of a reference signal by computing a cost matrix in the form of Euclidean distance between pairwise points (Atia et al. 2017; Panichpapiboon and Leakkawn, 2020; Saiprasert et al. 2017). The reference signal corresponding to the optimal or lowest cost path is selected as the detected driving event for the incoming or unknown signal. One of the main advantages of this technique is the ability to compare compressed and stretched portions of two signals while accounting for signal length differences.

Most recent studies (Bakhit et al. 2017; Kumar et al. 2013; Mandalia and Salvucci 2005; Yang et al. 2017; Zheng et al. 2014) have explored the use of both shallow and deep machine learning models for driving maneuver detection and have obtain accuracies between 70 percent and 98 percent. The machine learning algorithms could either be supervised as in support vector machine (SVM), artificial neural networks (ANN), and Long-short-term-memory (LSTM) or unsupervised as in k-means (Bejani et al. 2019; Bhoraskar et al. 2012; Carvalho et al. 2017; Júnior et al. 2017; Li et al. 2021; Yu et al. 2016.). Supervised learning algorithms learn and classify events based on ground truth information whereas the unsupervised learning algorithms analyze and cluster unlabeled datasets. The advantages of using ML algorithms in solving these problems are that they aid in the automation of the process of detecting driving maneuvers and produces more reliable models.

### 3.1.1 Motivation

Despite significant progress and appreciable maneuver detection accuracies attained especially from ML-based algorithms, there are still open challenges that remain unsolved. First, all previous ML-based studies have treated vehicle maneuver detection as a time series classification problem. A major challenge that is not addressed by recent approaches is time series segmentation. This is an important step that should precede the development of ML classifiers: it separates any raw, continuous vehicle telemetry signal into a finite set of discrete events and anomalies with unique characteristics that can be used to train ML models for maneuver detection. The time series segmentation problem is straightforward if all the events contained in the continuous signal have a fixed duration: A simple, moving window with fixed time window could be used to define the start and end of each event. For NDS data however, the duration, frequency and amplitudes of events may vary significantly depending on the speed of the vehicle, type of sensor, driving behavior and type of event (lane change or turning movement). A robust time series segmentation algorithm is therefore needed to extract unique events that are needed to train and test ML algorithms for maneuver detection. Second, the robustness and transferability of models developed for maneuver detection have not been well tested: the size of data, number of drivers and events are usually not large enough to deduce the best performing models, or architectures needed for accurate detection of driver maneuvers. For example, Mandalia and Salvucci (2005) and Yang et al. (2017) reports high accuracies for only 4-

431 drivers driving 8-80.4 km in the study. Studies have also evaluated these models on only one type of hardware acquisition systems: OBD or mobile phone.

### 3.1.2 Objectives

As a result of the above limitations, the study develops an end-to-end pipeline for automatic, frame-by-frame labelling of NDS videos into various driving events by using vehicle telemetry data. To achieve this goal, we formulated the problem as a time series segmentation and classification problem. The segmentation task was achieved by developing a novel segmentation algorithm that utilizes the principle of energy maximization to detect the start and end of any driving event. Furthermore, the performance of both shallow and deep machine learning models for characterizing different types of drivers' maneuvers are evaluated using a large database of NDS data (200 hours of video and vehicle telemetry data) from three different studies: SHRP2 (Antin et al. 2019), Nebraska Medical Center (Drincic et al. 2020.), and a mobile application (Aboah et al. 2021). Annotating data from multiple sources enables us to evaluate the transferability of the segmentation and classification algorithms developed.

The rest of this Chapter is structured as follows. A review of relevant literature is discussed in section two. Section three presents the data collection approach and problem statement. The methodology used in this study is presented in section four. Section five presents the results and discussion of the study. The study presents its conclusion and recommendation in section six.

## 3.2 Literature Review

This section first discusses the major naturalistic driving studies that have been conducted for the purpose of understanding driver behavior. Next, we review the different types of kinematic variables that has been used in related studies to detect and analyze driving maneuvers from vehicle on-board diagnostics (OBD) or mobile phone sensors. The last section of the review discusses different machine learning, computer vision and pattern recognition algorithms that have been developed for maneuver detection.  We explore the strengths, limitations, as well as the practical implementation of these algorithms.

### 3.2.1 Naturalistic Driving Studies

The 100-car NDS study is the first to collect extensive data on naturalistic driving of many drivers over an extended period. The primary goal of the study was to provide information about crashes and pre-crash events through the use of environmental and sensor data (100-Car Naturalistic Driving Study, 2006). About 100 passenger vehicles were retrofitted with a data acquisition system consisting of five cameras, a doppler radar antenna, a GPS, accelerometer, alcohol sensor, and an incident push button to continuously collect data under naturalistic driving settings. The study generated petabytes of data from 241 primary and secondary drivers, with about 43,000 hours of video data and over 3.1 million vehicle kilometers driven. The study captured many extreme cases of driving behavior including severe drowsiness, impairment, judgment error, risk taking, willingness to engage in secondary tasks, aggressive driving, and traffic violations.

The Canadian Driving Research Initiative for Vehicular Safety in the Elderly (Candrive) conducted a similar but much larger study, with over 256 drivers, 80.5 million vehicle kilometers driven and 5 million hours of video—a total of approximately 2 petabytes of compressed data. The primary objective of the study was to identify prospectively older drivers who were medically unfit to drive (Charlton et al. 2019). In addition to recorded videos, the study monitored participants' driving patterns by recording location data from a GPS, vehicle's speed, the position of the gas pedal, the engine's speed, and the air temperature. The study found out that elderly citizen that traveled low mileage were less prone to vehicle crashes. The European Naturalistic Driving (UDRIVE) also designed a similar study to collect data on road user behavior in various European regions under normal and near-crash conditions (Barnard et al. 2016.). The study retrofitted vehicles and scooters with DAS consisting of Mobile eye smart cameras, IMU sensors, GPS, CAN data, and a sound level sensor. The type of DAS was slightly modified base on the vehicle type. For example, trucks had 8 cameras instead of 5 cameras for passenger cars. The study collected a total of 87,871 hours of video data. The Australian Naturalistic Driving Study (ANDS) aims to improve understanding of how people behave in routine and safety-critical driving situations (Larue et al. 2018). The data for this study were gathered over a four-month period. The researchers recruited 360 volunteer drivers (180 from New South Wales and 180 from Victoria) and installed a data collection system in their private vehicle. The DAS is analogous to (Antin et al. 2019). The study found out that about 45 percent of the time, drivers were distracted behind the wheel.  Lastly but not the least, we discuss the MIT Advanced Vehicle Technology (MIT-AVT) which aims to set the bar for the next generation of NDS programs by leveraging large-scale computer vision analysis for human behavior (Fridman et al. 2019.). The DAS used in this study is comprised of an IMU, GPS, and CAN messages, as well as three high-definition cameras. The research is currently ongoing and will broaden in scope in the future. 122 individuals have taken part, 15610 days have passed, 823401.5 kilometers have been traveled, and 7.1 billion video frames have been collected. The preliminary result from the study indicates that drivers tend to look at things that non-related to driving more often whiles driving. Lastly, prior NDS emphasized vision-based approaches exclusively, omitting critical psychophysiological factors such as cognition and emotion due to technological and computing constraints (Tavakoli and Heydarian 2021). The primary objective of this study was to establish a human-centered multimodal naturalistic driving study in which driver behaviors and states are monitored using in-cabin and outside video streams, physiological signals such as driver heart rate and hand acceleration (IMU data), ambient noise, light, the vehicle's GPS location, and music logs with song features. This study is currently ongoing with no publication on the outcomes of their study.

### 3.2.1 Kinematic Variables for Detecting Driving Maneuvers

Recent car models have OBDs that are able to transmit high resolution vehicle kinematic information in fractions of a second. Several studies have also explored extracting and analyzing kinematic data from smartphone which tend to have a high penetration rate. Kinematic parameters such as acceleration, deceleration, orientation, yaw rate, and time to collision (TTC) are frequently used in research to identify driving events (Benmimoun et al. 2011; Hankey et al. 2016; McGehee et al. 2007; Lerner et al. 2010; Olson et al. 2009; Pilgerstorfer et al. 2012). The authors of Benmimoun et al. (2011), Hankey et al. (2016), McGehee et al. (2007), Lerner et al. (2010), Olson et al. (2009), Pilgerstorfer et al. (2012) used these parameters to examine vehicle maneuvers from NDS data. Benmimoun et al. (2011), Hankey et al. (2016), and McGehee et al. (2007) used accelerometer values to detect driving behaviors of young teens such as improper turns and curve using a rule-based approach. Olson et al. (2009) used TTC as a surrogate to measure changes in safety of the driver using rule-based approach. Pilgerstorfer et al. (2012) used lateral and longitudinal acceleration as well as TTC to assess the triggered events of truck drivers. Bogard (1999) used GPS data to detect lane changes. Xuan and Coifman (2006) proposed a similar technique for detecting lane changes by analyzing differential global positioning system (DGPS) data for the vehicle's lateral position instead. Although these are promising and much straightforward, the GPS precision levels required are not attainable from the current generation of mobile sensors. Miller et al. (2005) in a study used yaw rate to identify lane changing maneuvers made by heavy vehicles. The researchers hypothesized that changing lanes would produce a yaw rate signal similar to that produced by a noisy sine wave. A study conducted by Ayres et al. (2004) examined both the vehicle’s velocity and yaw rate as potential variables by using a rule-based approach in detecting turns, lane changes, and curves on various types of roads. Li et al. (2021) developed a machine learning model to detect various driving maneuvers using accelerometer and gyroscope reading using a semi-supervised machine learning algorithm.

### 3.2.3 Approaches for Detecting Driving Maneuvers

The approaches used in various literature for detecting driving maneuvers can be group into three categories: vision-based approaches, patten or rule-based approaches and machine learning approaches.

All vision-based approaches begin by detecting road markings before determining any driving maneuver. The color difference between lane markings and road surfaces defines the edge, gradient, and intensity of road features used for lane detection (Gao et al. 2009; Xu et al. 2009). Many researchers have used edge information to find straight lines that could be lane markings in a vision-based approach to identify various driving maneuvers (Xu et al. 2009). A B-spline is a popular mathematical model (Li et al. 2014; Son et al. 2015) which uses potential points derived from the lane markings to detect road lanes. Son et al. (2015) employs Kalman filter in tandem with a B-spline to detect lane markings. The B-spline is also commonly used to convert RGB data to HSI or custom color spaces (Rotaru et al. 2008) or color features (Mohamed et al. 2015). Among the feature-based lane detection approaches are artificial neuron networks (Mohamed et al. 2015), histogram of oriented gradients (HOG) (Naiel et al. 2014), and support vector machine (SVM) classifier (Mandalia and Salvucci 2005). Although these studies produced excellent results, they do have some limitations. First, these classes of algorithms are heavily reliant on visible road markings for driver maneuver detection. Their performance suffers in the absence of lane markings, or when lane marking retroreflectivity is low. Also, vision-based approaches are affected by video quality and driving environment including weather conditions. The resolution of most NDS datasets is usually low due to high compression rates. Robust image enhancement techniques are needed to achieve modest performance for maneuver detection.

A wide variety of machine learning approaches have been used in literature for detecting driving maneuvers. Li et al., (2021) proposed a semi-supervised LSTM model to detect driving maneuvers. In their study, three long short-term memory (LSTM) models were built and trained to evaluate the proposed semi-supervised learning algorithm. According to the experimental results, the proposed semi-supervised LSTM could learn from unlabeled data and deliver impressive results with only a small amount of labeled data. The study compared the performance of the proposed method to other machine learning models. When compared, the proposed model outperformed existing machine learning techniques such as convolutional neural networks, XGBoost, and random forests on several measures, including accuracy, recall, F1-score, and area under the curve. The overall accuracy of the developed model was 99.7%. Bhoraskar et al., (2012) used support vector machine (SVM) to detect braking and road bumps using accelerometer, GPS, and magnetometer data collected by a smartphone and achieved an overall accuracy of 78.37%. Júnior et al., (2017) evaluated the performance of multiple machine learning algorithms for detecting driving maneuvers (e.g., aggressive braking, acceleration, left turn, right turn) by using the area under the curve as a performance measure. The study found out that random forest outperformed other algorithms including SVM and Bayesian network with an accuracy of 99.1%. Yu et al. (2021) employed a fully connected neural network to detect driving maneuvers such as weaving, swerving, and quick braking using the accelerometer and rotation sensors of a smartphone. The findings indicated that the neural network (95.36%) performed more accurately than the SVM (90.34%) at classifying driving events. Bejani et al. (2019) used a convolutional neural network (CNN) to rate drivers as safe or dangerous based on accelerometer data from their smartphone. The findings demonstrated that CNN was capable of accurately classifying diverse driving styles with the use of regularization terms. The developed model achieved an overall accuracy of 95%. Carvalho et al. (2017) used recurrent neural networks to explore the detection of driving movements (RNNs). Unlike CNN, RNN was developed to learn from time series data and has shown potential. The authors compared the performance of a variety of RNN architectures, including long short-term memory (LSTM), gated recurrent unit (GRU), and standard RNN. The findings suggested that LSTM (99.7%) and GRU (99.2%) achieved equivalent results and outperformed traditional RNNs (93%) in recognizing different driving maneuvers accurately.

Some studies have also used rule-based and pattern matching algorithms to detect various driving maneuvers. Saiprasert et al. (2017) proposed both a ruled-based and pattern matching-based algorithms to detect aggressive and normal driving maneuvers. The study concludes that the pattern matching algorithm outperforms the rule-based algorithm in detecting driving maneuvers. (Sun et al. 2021) combined a dynamic time warping (DTW) and bagging tree algorithm to driving events using accelerometer and gyroscope data collected by a smartphone. Atia et al. (2017) compared the performance of K-nearest neighbor and dynamic time warping (DTW) algorithm in detecting various driving maneuvers. The results from the study indicate that k-NN achieved the best accuracy to differentiate between road anomalies and driving behaviors whereas DTW achieved the best accuracy in driving turn behavior classification.

Table 3.1 summarizes the literature by examining various studies, the algorithms that were used, and the kinematic variables that were used.

**Table 3. 1: Literature Summary**

| Papers | Algorithm | Kinematic Variable | Driving Event Detection |
|---|---|---|---|
| (Miller et al. 2005) | Frequency Thresholding | Yaw Rate | Lane changes |
| (Nhtsa, 1999) | Rule-based algorithm | road’s curvature | Lane keeping |

|  |  |  |  |
|---|---|---|---|
| (Ayres et al., 2004) | Algorithm Tunning | Yaw rate, Speed | Turns, curves, and lane change |
| (Xuan & Coifman, 2006) | Pattern Matching | Differential global positioning System (DGPS) | Lane changes |
| (P. Li et al., 2021) | Semi-Supervised LSTM | Accelerometer, Gyroscope | Left-Right turns, Left-Right Lane change |
| (Bhoraskar et al. 2012) | SVM | Accelerometer, GPS, Magnetometer | Break detection |
| (Júnior et al., 2017) | SVM, Bayesian Network, Random Forest, ANN | Accelerometer, Magnetometer, GPS | Aggressive-breaking, acceleration, left turn, right turn, left lane change, right lane change |
| (Yu et al. 2021) | FCNN | Accelerometer, Rotation Sensors | U-turn, swerving, weaving, right turn, left turn |
| (Bejani et al. 2019) | CNN | Accelerometer | Normal drive, dangerous drive |
| (Carvalho et al. 2017) | RNN, LSTM, GRU | Accelerometer | Lane keeping, left turn, right turn, right lane change, left lane change |

## 3.3 Data Collection

Multiple streams of datasets were collected for this study. This include the Blackbox sensor data (Drincic et al. 2020), smartphone collected data (Aboah et al. 2021) and the VTTI NDS dataset (Antin et al. 2019). The Blackbox sensors developed by Digital Artefacts LLC were used to collect the data used in this study. The sensors were installed in individual personal vehicles to continuously record activities that occurred inside and outside of the vehicle. Multiple sensors, including GPS, accelerometer, wireless OBD, infrared, and high-resolution cameras, are embedded in the sensor instrumentation. As shown in **Figure 3.1b**, the windshield-mounted sensor package, which is mounted behind the rear-view mirror. Two cameras in the system continuously capture 1) a forward view of the vehicle and 2) a view of the driver and the interior of the vehicle. The driver's behavior is continuously recorded from the time the vehicle is turned on to the time it is turned off. The study included 77 participants who were observed over a three-month period. A total of 289681.9 kilometers of data was collected across the entire United States. This dataset contains far more detailed information on driver behavior across a wide range of geographic environments than laboratory-based or retrospective studies can. The study used a developed smartphone app to collect data on both freeways and local routes. The smartphone app interface is shown in **Figure 3.1c**. To collect data with the app, the mobile phone must first be mounted on the car's windscreen to record vehicle accelerations, rotations and some other relevant information. The video data was sampled at a frame rate of 10 frames per second, whereas the accelerometer and vehicle location data were collected at a frame rate of 30 samples per second (30 Hz).

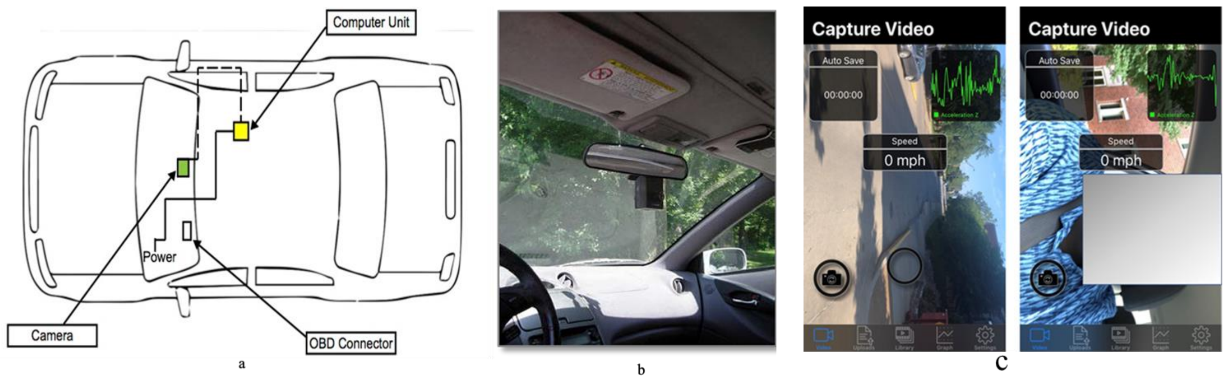


**Figure 3. 1.: a) The Positioning of the Blackbox sensors in the vehicle b) Example of the Blackbox sensor inside a vehicle c) Smartphone Data Collection App Interface**

### 3.3.1 Data Annotation

To be build the benchmark dataset, the gyroscope readings were manually annotated concurrently with the driving video by noting the timestamps associated with each event in the driving video. That is, a human annotator watches the video and records the start and end timestamps of each event, after which the annotator assigns a class number to the corresponding timestamp in the signal dataset. Additionally, the study visualized both the annotated signal and the driving video concurrently to ensure that the annotations corresponded to the actual timestamp of the event. When it is determined that annotations do not correspond to actual events, they are corrected and re-visualized. This process is repeated until all annotations correspond to actual events occurring during the driving video. This method was used to obtain all ground truth labeling for the three NDS datasets that were used in this study.

### 3.3.2 Data Cleaning and Preprocessing

To reduce the amount of noise in the gyroscope data, a simple moving average technique was employed to smooth the signal. As illustrated by the following (Chen, Dongyao, et al.; Karatas, Cagdas, et al.; Kang, Lei, and Suman Banerjee; You, Chuang-Wen, et al.) studies, preprocessed gyroscope signal results in gyroscope drift. The drift is responsible for shifting the actual time at which an event occurred and becomes very critical when dealing with real-time alert systems. Therefore, the study examined the occurrence of these gyroscope drifts and determined that, due to the high sampling rate of our data, the drifts were small and, as a result, did not affect our proposed algorithm's performance for extracting driving events. From **Figure 3.2** below, it can be seen that the gyroscope drift after preprocessing is very small and will have no effect on the study's goal of extracting the various shapes that represent specific driving events using our proposed algorithm.

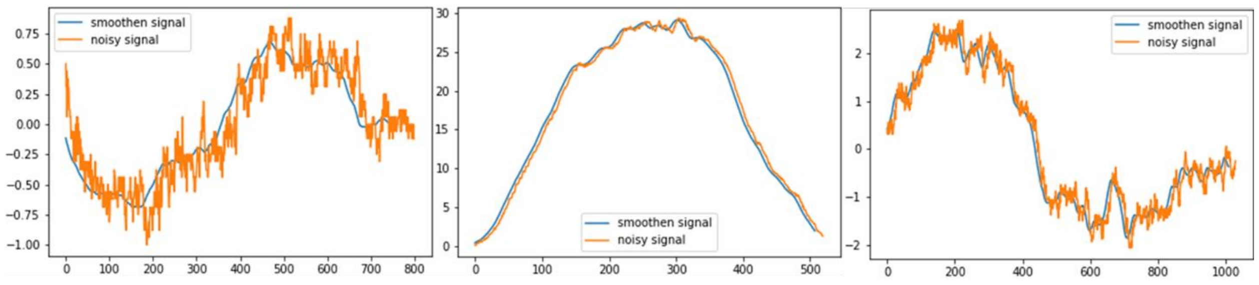


**Figure 3.2: A comparison of raw gyroscope data to processed gyroscope data**

### 3.3.3 Training and Validation Dataset for Developing the Classification Models

The study used the Nebraska NDS dataset to develop all four models. All four models were trained on an NVIDIA GTX 1080ti GPU with 16,233 training samples. For all developed models, we used a 70:30 split for model training and validation. The training samples are distributed as follows; right turns- 4,362 samples, left turns- 3,968 samples, right curves- 2,051 samples, left curves- 1,947 samples, right lane change- 1,895 samples and left lane change- 2,010 samples.

## 3.4 Methodology

### 3.4.1 Problem Formulation and Overview

Most existing methods for driver maneuver detection formulates the problem purely as a classification problem, assuming a discretized input signal with known start and end locations for each event or segment. However, in practice, vehicle telemetry data used for detecting driver maneuvers are continuous, therefore, a fully automated driver maneuver detection system should implement solutions for both time series segmentation and classification. The method proposed in this paper maps a continuous sequence into a dense segmentation followed by event classification using machine learning and a heuristic algorithm.

Specifically, let $x \in \mathbb{R}^{\tau S \times C}$ represent a vehicle telemetry dataset with $C$ sensors or channels, sampling at a rate $S$ for a period of $\tau$ minutes. Let $\mathbf{v} = \{v_1, v_2, v_3, \dots, v_n\}$ be a video sequence of frames that corresponds to each sample in $x$. The goal is to map $x$ into finite set of segments, $\lfloor \tau \cdot e \rfloor$ where $e$ is the segmentation frequency. Compared to other segmentation approaches (Perslev et al., 2019.) where $e$ is fixed, in the current method, the parameter is variable and adaptively selected based on input signal features. Each segment is passed through a model $f(\tau \cdot e; \theta): \mathbb{R}^{T \times i \times C} \to \mathbb{R}^{T \times K}$ with parameters $\theta$ that maps each segment $\lfloor \tau \cdot e \rfloor$ to one of $K$ class labels including: stop and lane keeping events, lane changes, left-right turning movements and horizontal curve maneuvers. Finally, frame-by-frame annotation of video sequence is achieved by mapping classified segments to the image domain.

The general methodology adopted for automatic detection of driver maneuvers consists of 5 distinct steps as shown in **Figure 3.4**. First, multi-modal data is pre-processed and standardized, followed by segmentation of kinematic data into main driving events, anomalies and lane keeping events. Anomalies and lane keeping events are passed through a heuristic's algorithm which further classifies these events into anomalies, stop and lane-keeping events. Only driving events are passed through a machine learning classifier. By training only main driving events, the ML classifiers were able to learn the unique characteristics of lane changing and turning movement events without confusing them with other features such as lane-keeping, lane-incursions events and anomalies caused by road roughness or erratic driving behaviors. The outputs of the classifiers and heuristics are finally used for frame by frame driving event annotation of raw video feeds. Each step of the methodology is further discussed in the sections below.

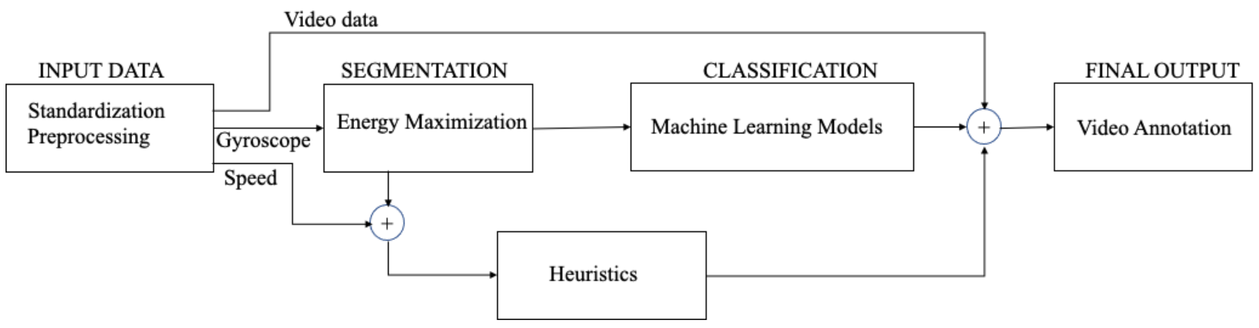


**Figure 3. 3: Flowchart of Methodology**

### 3.4.2. Input Data Normalization

Although previous studies used kinematic variables such as yaw rate, accelerometer readings to detect limited driving maneuvers, the current study determined that gyroscope reading (which measures the orientation and angular velocity of the vehicle), and the vehicle's speed data are the two key input variables that can be used to characterize all driving maneuvers consistently across different hardware measurement systems. The gyroscope readings (z-axis) were smoothed using a simple moving average and subsequently feature-scaled with the mean normalization equation defined in Equation 1. The raw speed data will be used to develop heuristics for detecting lane keeping and stopped events whereas the standardized gyroscope readings are pushed through a time series segmentation algorithm for driving maneuver event detection.

$$\hat{x} = \frac{x_i - \bar{x}}{x_{max} - x_{min}} \qquad \text{Equation 1}$$

Where $x_i$ is the gyroscope reading at timestamp $i$ , $\bar{x}$ is the mean of all data, $x_{max}$ is the maximum data value and $x_{min}$ is the minimum data value.

### 3.4.3 Time Series Segmentation

The segmentation step involves the extraction of driving events using the energy maximization algorithm (EMA). The fundamental assumption driving EMA is that the sum of the energy from the start of an event will continuously increase until the end of the event is reached. At each time step, $t$ , we dilate a moving window at different rates of $w$. For each dilation rate, $w_i$, the energy of $X\left[t - \frac{w_i}{2} : t + \frac{w_i}{2}\right]$, is computed using Equation 2. In Equation 2, the computed energies are scaled by a factor of $\frac{S}{N}$. This factor takes into account the duration of the event (S) and the number of data points (N) in the signal so that events that are not fully captured but has a greater energy could be penalized.

$$e_n = \frac{S}{N} \sum_{n=0}^{n+1} X[n]^2 \qquad \text{Equation 2}$$

We then determine if an event is present based on the calculated energies. The dilation rate, $w$, increases by a factor of 0.25s and continues to dilate until the computed energy is a maxima, i.e., $E_{t-n} \leq E_t \geq E_{t+n}$. If more than one maximum is detected, we use non-maximum suppression to remove overlapping signals. The whole process is repeated for each time step until the final time step for the input signal. The output from the segmentation step is either an event or non-event as shown in **Figure 3.5**.

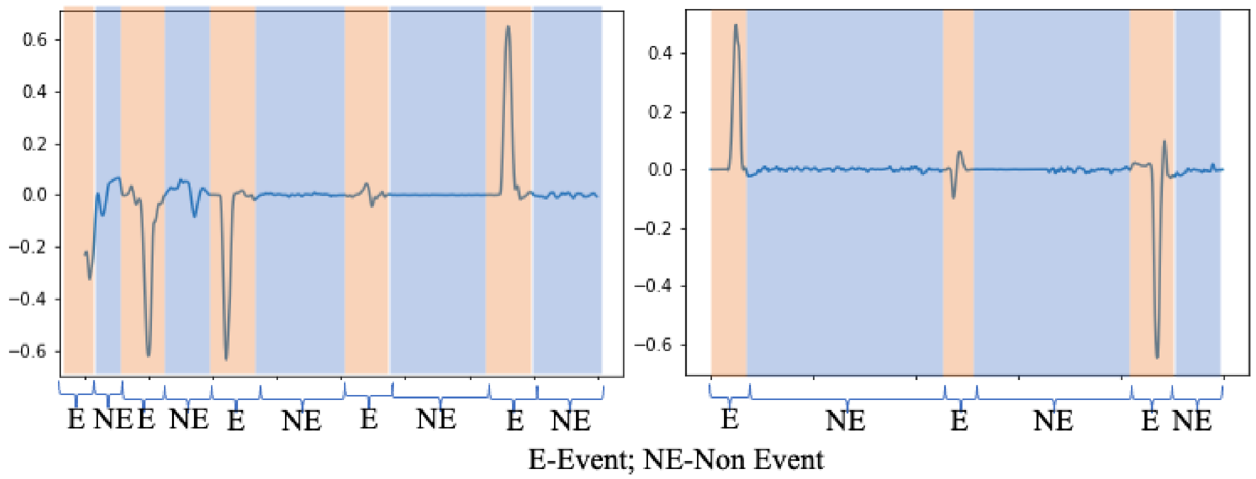


**Figure 3. 4: Segmentation of Signal into Events and Non-events**

**Figure 3.6.** shows an example of a time step $t$ and a plot of energies computed at different dilation rate, $w$. The step in performing the segmentation is summarized in Table 3.2 below. Events detected through the energy-maximization process are events are passed through the ML models for classification into lane changes and turning movements. Non-events are passed through a heuristics algorithm for classifying lane-keeping and stop events.

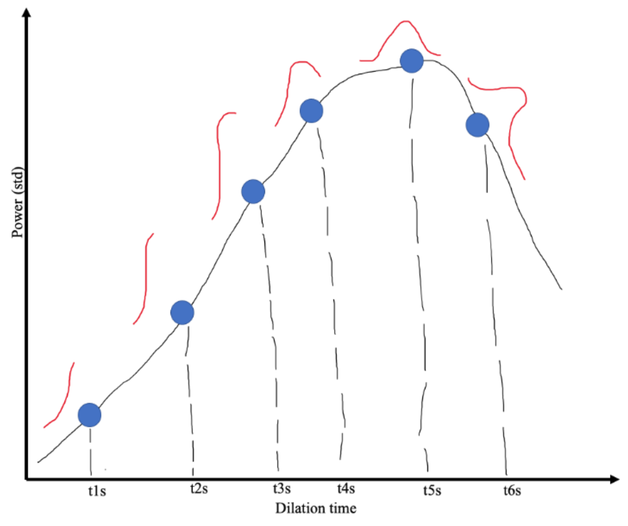


**Figure 3. 5: An illustration of the dilating time window**

**Table 3. 2: Pseudo code for Performing Event Segmentation**

| Steps | Energy Maximization Algorithm (EMA) |
|---|---|
| 1 | **Require:** $x, h_O$: gyroscope and sampling frequency |
| 2 | **Require:** $w, d$: window length and dilation threshold |
| 3 | **Require:** $ds$: dilation step |
| 4 | $t \leftarrow 0$ (Initialize timestep) |
| 5 | $i \leftarrow 0(Initialize\ dilation\ step)$ |
| 6 | $e_{max} \leftarrow [\,] - initialize\ energy\ maximas$ |
| 7 | $x \leftarrow f\left(x, w_s = {}^{h_o}/_{3}\right)$ (low pass filter input ) |
| 8 | **Loop** |
| 9 | **While** $i < d$ **do** |
| 10 | $t \leftarrow t + 1; e \leftarrow [\,] -$ initialize energy vector |
| 11 | $e_i \leftarrow \frac{S}{N}\sum_{n=0}^{n+1} X[n]^2$ (Compute energy for each dilating window) |

```
12              if $e_{i-n} \leq e_i \geq e_{i+n}$. Then
13                  $e_{max}[t] = e_i$
14              $i \leftarrow i + ds$ – (increment dilation, $default = 0.25sec.$)
15          end while
16      $\boldsymbol{\hat{e}_{max}}, \boldsymbol{i_{max}} \leftarrow \boldsymbol{\nabla} g(e_{max}, i)$ − compute gradient and apply non −
17  maximum suppression

    Output: $\boldsymbol{\hat{e}_{max}}, \boldsymbol{i_{max}} \leftarrow$ events and corresponding indices
```

### 3.4.4 Classification

Supervised ML algorithms were used to classify the events extracted from the EMA into six maneuvers: left-right lane changes, left-right turning movements and left-right horizontal curves movements. Four main classifiers were evaluated including: LSTM, SVM, 1D-CNN, Random Forest. The model architectures, and structure of the input data used for training each machine learning algorithm are explained in detail below.

#### *3.4.4.1. Input Data Structure*

The extracted events from the segmentation step are restructured before being passed to the classification algorithms. For the deep learning algorithms, the raw samples from the energy maximization algorithm are passed through them for classification, whereas for the machine learning algorithms, the raw data is first passed through a principal component analysis (PCA) to reduce dimensionality before being passed through them.

The number of input features and output features are 50 and 1 respectively for all trained models. A summary of the training parameters used to build the various models are presented in Table 3.3 below.

**Table 3. 3: Training Parameters**

| Parameters | LSTM | SVM | 1D-CNN |
|---|---|---|---|
| Number of features | 1 | 1 | 1 |
| RNN Layers | 3 | X | X |
| Hidden Layers | 20 | X | 250 |
| Learning rate | 0.001 | X | 0.001 |
| Loss function | Cross entropy loss | X | Cross entropy loss |
| Optimizer | Adam | X | Adam |
| Activation function | X | X | ReLU and Sigmoid |
| Epoch | 600 | X | 40 |
| Kernel | X | Linear | X |
| Gamma | X | Auto | X |
| Kernel size | X | X | 3 |
| Filters | X | X | 250 |

*3.4.4.2 Long-short-term-memory (LSTM) model*

LSTM is a supervised deep learning architecture which is used for both classification and regression. LSTM is a special type of recurrent neural network (RNN) that is capable of learning long-term dependencies (P. Li et al., 2021). It has a single cell state that runs the length of the chain. The cell state can be modified by either adding or removing information. The LSTM architecture comprises of three gates that protect and control information that pass through the cell state. These gates are the forget gate, the input gate, and the output gate.

The forget gate deletes information from the cell state that is not required to pass through to the input gate. This is accomplished by the equation below.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \qquad \text{Equation 3}$$

At the input gate, new information is stored, and values are updated in the cell state. This results in the creation of a vector of new candidate values, $\tilde{C}_t$. The mathematical representation of what happens in the input gate is shown in the equations below.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \qquad \text{Equation 4}$$

$$\tilde{C}_t = tanh(W_c \cdot [h_{t-1}, x_t] + b_c)$$

The cell state is updated by adding the previously deleted information, $f_t * C_{t-1}$ to the newly added information $i_t * \tilde{C}_t$ . The updated cell state can be expressed mathematically as

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t \qquad \text{Equation 5}$$

Finally, the output gate outputs the relevant portions of the cell state.

$$o_t = \sigma(W_0 \cdot [h_{t-1}, x_t] + b_0) \qquad \text{Equation 6}$$

$$h_t = o_t * \tanh(C_t)$$

### *3.4.4.3 Support Vector Machine (SVM)*

Support Vector Machines (SVMs) are a type of supervised machine learning technique that can be used to solve both regression and classification problems. SVMs are designed to strike a balance between fitting the training data and reducing model complexity (Cortes et al., 1995). This method of defining a loss function is known as structural risk minimization (SRM), and it typically yields a better model generalization than the empirical risk minimization approach of defining a loss function. SVMs were originally developed to solve two-group classification problems; therefore, applying it to multi-label classification problems results in the input data being highly dimensional. As a result of the high dimensionality of the input data, computational issues such as handling large vectors and overfitting occur. These issues are resolved by the addition of a kernel function. A kernel function returns the dot product of the original data points' feature space mappings. SVMs employ a variety of kernel functions, including linear, polynomial, and Gaussian RBF. The algorithm for performing multiclass classification using support vector machines involves transforming the input vector into a higher-dimensional feature space. In the feature space, a linear decision surface called a hyperplane is constructed (Cortes et al., 1995). The hyperplane represents the greatest separation between any two classes. In addition, two parallel hyperplanes are constructed on either side of the hyperplane to segregate the data. A separating hyperplane is one that minimizes the distance between two parallel hyperplanes as shown in Equation 7.

$$\max_{v} \frac{1}{2}\left(\min_{x_i \in C_1} v^T(x_i - x_0) - \min_{x_j \in C_2} (-v)^T(x_j - x_0)\right) \qquad \text{Equation 7}$$

$$s.t\ ||v||_2 = 1.$$

Where $v$ is the unit vector, $C_1\ and\ C_2$ are contants, and $x_i \in \mathbb{R}^k$. The idea is that the larger the margin or distance between these parallel hyperplanes, the smaller the generalization error of the classifier.

### *3.4.4.4 1D-CNN*

A one-dimensional convolutional neural network (1D-CNN) is a deep learning architecture that can either be supervised or unsupervised and can be used for both regression and classification problems. The A 1D-CNN applies a kernel along a one-dimension input data. The input data is usually a signal data with two dimensions. The first dimension is the time-steps whereas the second dimension is the signal values. **Figure 3.7** shows the flowchart of the 1D-CNN. Mathematically, a one-dimensional convolutional neural network is composed of an input vector $x \in R^p$ and a filter $w \in R^k$ where $k \leq p$. The 1D-CNN takes $w^T x[i: i + k]$ for each surrounding set of k elements of $xx[i: i + k]$ and gives one node of the convolutional layer.

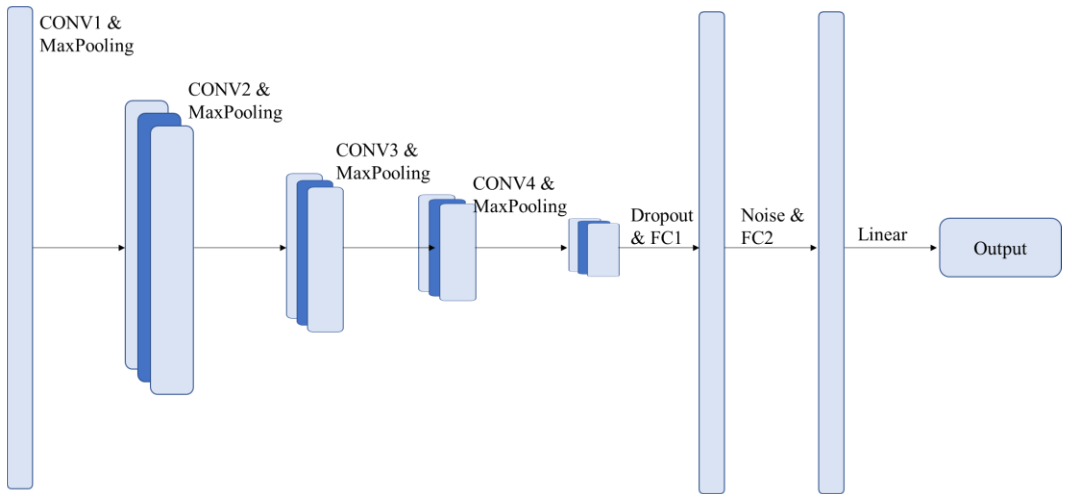


**Figure 3. 6.: 1D-CNN Architecture**

### *3.4.4.5. Random Forest*

A random forest is a supervised machine learning algorithm that leverages ensemble learning to solve complex problems involving regression or classification. Random forests are a collection of tree predictors in which the values of a random vector are sampled independently and uniformly across the forest to determine the values of each tree (Breiman 2001). As the number of trees in a forest increase, the generalization error converges to a limit (Breiman 2001). The generalization error of a forest of tree classifiers is proportional to the strength of the trees in the forest and their correlation. By randomly splitting each node on a tree, we obtain error rates that are comparable to Adaboost's (Freund et al., 1996), but more robust to noise in this case (Breiman 2001). Internal estimates of error, strength, and correlation are used to demonstrate the effect of increasing the number of features used in the splitting process. Internal estimates are also used to determine the significance of variables. Each class in the dataset is determined by letting all the trees in the forest vote for a class. The most voted class becomes the classification of the data points.

For each internal nodes of the tree, it takes a subset of features at random and utilizes that information to compute the centers of various classes present in the data at the current node. For example, given two classes, 0 and 1, the centers of the classes will be denoted as Left-Center and Right-Center respectively.

$$left_{center}[k] = \frac{1}{n}\sum_{i=1}^{n} x_{ik} I(y=0) \quad \text{Equation 8}$$

$$right_{center}[k] = \frac{1}{n}\sum_{i=1}^{n} x_{ik} I(y=1) \quad \text{Equation 9}$$

Where $I(y=0)$ and $I(y=1)$ are the dictator functions. Each record in the dataset is assigned to the appropriate class at the present node by computing the Manhattan distance between the center and the record as illustrated in Equation 10.

$$Distance\ (center, record) = \sum_{i \in sub} |ceter[i] - record[i]| \quad \text{Equation 10}$$

Where $sub$ is the subset attributes randomly selected from the dataset. Each tree, therefore, grows without pruning.

*3.4.4.6. Heuristics*

The characteristic patterns of vehicle telemetry data especially during stop and lane keeping events vary widely even for the same driver. As a result, it generates high false positive rates when fed through machine learning models. In the current study we developed a heuristic algorithm based on the vehicle speed and an adaptive thresholding technique to classifying lane-keeping and stop events. A stopped event occurs when the speed of the vehicle is zero. To detect lane-keeping events, we draw from a probability distribution curve. The assumption here is that lane-keeping is the most dominant event in every trip. Therefore, all gyroscope readings about $k$ standard deviations from the mean should belong to this class. A value of $k = 2$ was used in this study. The equation below summarizes the heuristic algorithm.

$$e_l = \begin{cases} stop & if\ speed \cong 0 \\ lane - keeping & \mu + k\sigma \leq x \leq \mu + k\sigma \end{cases} \qquad \text{Equation 11}$$

**3.4.5. Video Annotation**

The goal of this methodology is to automate the frame-by-frame annotation of driving events of NDS dataset. The machine learning classification outputs the start and end time of the event which is the same as the heuristics. The classification outputs and indices from the machine learning models and heuristics are combined and transferred into the time domain for frame-by-frame video annotation.

## 3.5 Results

### 3.5.1 Performance Measures

The efficiency and accuracy of the Energy Maximization Algorithm and the various machine learning models were evaluated using various performance measures. We assessed the performance of the machine learning models using precision (P), F1 score (F1), and recall value (R). The F-1 score is the harmonic average of the recall and precision values. Precision is defined as the ratio of true positives (tp) to all predicted positives (tp+fp), as shown in Equation 12. Similarly, recall is the ratio of true positives to all true positives (tp+fn) is defined in Equation 13.

$$Precision = \frac{tp}{tp+fp} \qquad \text{Equation 12}$$

$$Recall = \frac{tp}{tp+fn} \qquad \text{Equation 13}$$

$$F1 = \frac{2*Precision*Recall}{Precision+Recall} \qquad \text{Equation 14}$$

On the other hand, we evaluated the accuracy of the EMA using a duration score (DS) metrics computed as

$$e_l = \begin{cases} 1 & if\ |A_i - P_i| > 1 \\ |\ A_i - P_i| & else\ \ |A_i - P_i| < 1 \end{cases}$$

$$DS = 1 - \frac{1}{n}\sum_{i=1}^{n} e_l \qquad \text{Equation 15}$$

Where $A_i$ is the $ith$ actual event duration, $P_i$ is the $ith$ extracted event duration by the EMA, and $n$ is the number of total events. Finally, the overall accuracy of the pipeline was calculated using Equation 16. The F1 scores are multiplied by the duration score in this performance metric.

$$Overall\ Accuracy = F1\ score * DS \qquad \text{Equation 16}$$

## 3.5.2 Segmentation Outputs

The segmentation step outputs either an event (turns, lane changes, and curves) or a non-event (lane keeping and stop) as show in **Figure 3.4**. The EMA in the segmentation step extract very distinctive shapes from the signal as shown in **Figure 3.8.** From **Figure 3.8.**, it can be observed that the full extent of signals corresponding to turns, lane changes, curve negotiation, are extracted at different length of time. Also, all the various driving events have varying amplitudes as shown in the **Figure 3.8**. Events that occur in both directions (right and left) have similar shape and amplitude but occur at different phases. Example is the right turn and left turn. The algorithm's ability to extract the full shape of the signal before taken into the model alleviate the limitation encountered with the fixed time window approach leading to a lot of false positives. Additionally, lane changes are clearly differentiated from lane keeping due to their distinctive shapes. The extracted events are classified using machine learning models, while the non-events are classified using heuristics.

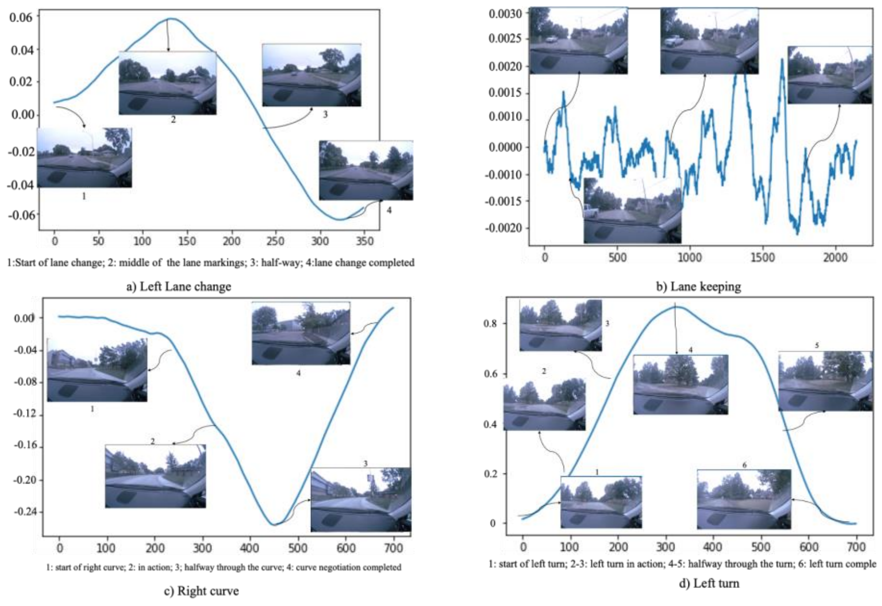


**Figure 3. 7.: Extracted Events Shape by the EMA**

### *3.5.2.1 Evaluating the Accuracy of The Energy Maximization Algorithm*

The actual durations of events were compared to the EMA-derived durations. According to **Figure 3.9.**, the distribution of event durations for actual and extracted events was similar for right turns, right curves, and left curves. Additionally, some extracted left turns had durations that were significantly longer than the actual left turn durations. This can be explained by the fact that some left curve or right curve negotiations are immediately followed by a left turn in which the EMA records a portion of those events as left turns, resulting in the increased duration of some left turns. In general, it is observed that the durations of events extracted via EMA are significantly longer than the durations of events manually extracted.

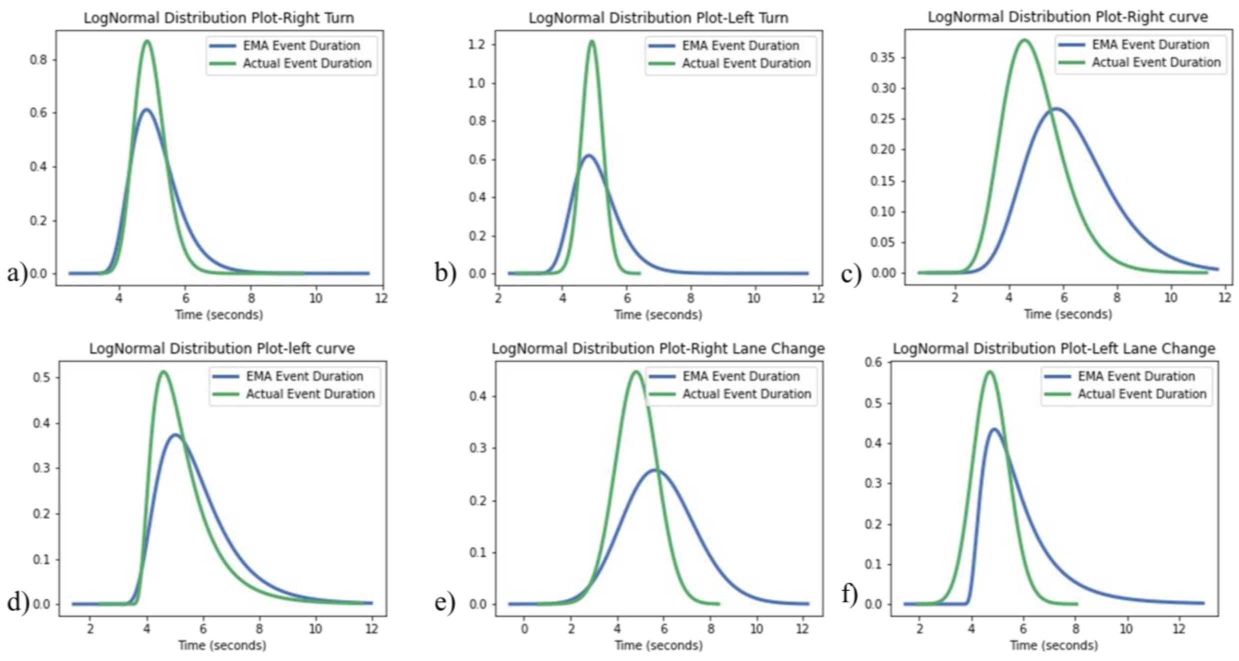


**Figure 3. 8.: Comparative Analysis of Lognormal Distribution of Event Duration a) right-turns b) left-turn c) right-curves d) left-curve e) right-lane-change f) left-lane-change**

Additionally, the extracted time distributions were compared to time distributions extracted for a variety of events in previous research. According to a study by Toledo et al. (1999), lane change durations range between 3 and approximately 7 seconds, which is consistent with the results shown in **Figure 3.10e and Figure 3.10f**. Additionally, the majority of right and left turns occur within the range of 4-6 seconds. For right curves, the majority of durations fell within the range of 4-6 seconds, but a sizable portion fell within the range of 7-10 seconds. These variations are explained by the varying lengths of right curves observed at various locations. Certain right curves are longer than others, requiring vehicles to negotiate for a longer period of time. For left curves, the same is true. Additionally, the majority of lane changes occurred within the range of 3-5 seconds, which is consistent with the findings of (Toledo et al. 1999).

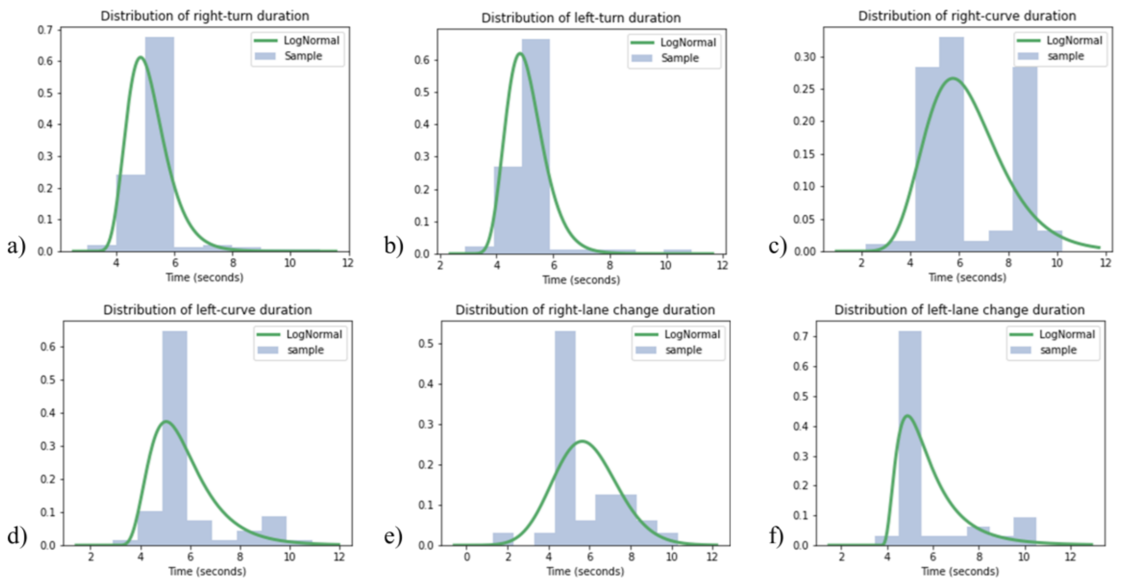


**Figure 3. 9.: Lognormal Distribution of Event Duration a) right-turns b) left-turn c) right-curves d) left-curve e) right-lane-change f) left-lane-change**

Finally, using Equation 15, the accuracy of the EMA was calculated, and the results are summarized in Table 3.4. According to Table 3.4, right and left turns, as well as right and left changes, had significantly high accuracies. On the other hand, the accuracies of the left and right curves were relatively low. The low accuracies can be attributed to a variety of factors, including the algorithm treating two consecutive events as one, as it is typically observed when a left curve follows a right curve or vice versa.

**Table 3. 4: Accuracy of the EMA based on Durations of Extracted Events**

| Driving Maneuvers | Accuracy of EMA |
|---|---|
| Right turn | 0.820 |
| Left turn | 0.843 |
| Right curve | 0.654 |
| Left curve | 0.690 |
| Right lane change | 0.618 |
| Left lane change | 0.593 |
| Lane keeping | 0.856 |
| Stop | 0.848 |

### 3.5.3 Classification Results

#### *3.5.3.1 Model Comparison*

In this study, four machine learning models were developed. Our analysis revealed that all four models had accuracies comparable to those reported in studies that trained similar models using a variety of kinematic variables (Bakhit et al. 2017; Kumar et al. 2013; Mandalia and Salvucci 2005; Zheng et al. 2014). It can be deduced that the gyroscope reading is sufficiently sensitive to detect all driving events, as seen when other kinematic variables are combined to perform the same task (Bhoraskar et al. 2012). Additionally, when comparing the number of iterations required to train the deep learning models, the

1D-CNN model converges after 20 epochs, whereas the LSTM model converges after 300 epochs. The 1D-CNN model, therefore, trains faster than the LSTM model. When the accuracies of all four models were compared, the overall accuracy of the 1D-CNN model was 98.99 percent, followed by the LSTM model at 97.75 percent, then RF model at 97.71 percent, and the SVM model at 97.65 percent that are comparable to accuracies obtained by (Bakhit et al. 2017; Kumar et al. 2013; Mandalia and Salvucci 2005; Zheng et al. 2014). The consistency of the accuracies obtained for all four models indicates that the EMA is effective at capturing all driving events. Furthermore, we evaluated the performance of all models using the F1 score, precision, and recall values for each driving maneuver as shown in Table 3.5. Lane change maneuvers (both left and right) had low F1 scores across all models. This is because of the false negatives caused by missed lane change events, which are particularly prevalent on highways with relatively high speeds. Additionally, right turn maneuvers had the highest average F1 scores across all models, ranging from 0.991 to 0.998. Similar scores were observed for left turns and right-left curve negotiations. Lane keeping on the other hand, had a high rate of false positives due to missed lane changes, particularly on highways and also stops. In summary, all models performed similarly well at predicting all types of driving maneuvers, with fewer false positives and negatives. Table 3.5 summarizes the models' predictions for specific driving events.

**Table 3. 5: Precision, Recall and F1 Values Obtained From LSTM, SVM, 1D-CNN, and RF**

| Driving Maneuvers | LSTM | | | SVM | | | 1D-CNN | | | RF | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 | P | R | F1 |
| Right turn | 0.994 | 0.996 | 0.995 | 0.989 | 0.991 | 0.990 | 0.997 | 0.996 | 0.996 | 0.998 | 0.996 | 0.997 |
| Left turn | 0.988 | 0.985 | 0.986 | 0.988 | 0.987 | 0.987 | 0.996 | 0.993 | 0.994 | 0.997 | 0.995 | 0.996 |

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Right curve | 0.989 | 0.990 | 0.989 | 0.989 | 0.984 | 0.986 | 0.986 | 0.991 | 0.988 | 0.998 | 0.993 | 0.995 |
| Left curve | 0.981 | 0.982 | 0.981 | 0.983 | 0.982 | 0.982 | 0.992 | 0.994 | 0.992 | 0.995 | 0.991 | 0.993 |
| Right lane change | 0.993 | 0.933 | 0.962 | 0.994 | 0.933 | 0.962 | 0.995 | 0.965 | 0.980 | 0.989 | 0.985 | 0.987 |
| Left lane change | 0.990 | 0.926 | 0.957 | 0.991 | 0.935 | 0.962 | 0.989 | 0.991 | 0.990 | 0.998 | 0.991 | 0.994 |
| Lane keeping | 0.951 | 0.976 | 0.963 | 0.951 | 0.976 | 0.963 | 0.996 | 0.994 | 0.995 | 0.994 | 0.993 | 0.993 |
| Stop | 0.975 | 0.975 | 0.975 | 0.974 | 0.976 | 0.975 | 0.989 | 0.994 | 0.991 | 0.995 | 0.996 | 0.995 |

The study further examined the overall accuracy of the developed pipeline using Equation 16, and the results are summarized in Table 3.6. In this performance metric, the F1 scores were penalized by the duration scores. Overall accuracy per driving event ranges between 0.645 and 0.852, as shown in Table 3.6. Right and left curves both exhibits relatively low overall accuracy, owing to their low duration score values.

**Table 3. 6: Overall Accuracy of Pipeline**

| Driving Maneuvers | LSTM | SVM | 1D-CNN | RF |
|---|---|---|---|---|
| Right turn | 0.816 | 0.812 | 0.817 | 0.818 |
| Left turn | 0.831 | 0.832 | 0.838 | 0.840 |
| Right curve | 0.647 | 0.645 | 0.646 | 0.651 |
| Left curve | 0.677 | 0.678 | 0.684 | 0.685 |
| Right lane change | 0.595 | 0.595 | 0.606 | 0.609 |
| Left lane change | 0.568 | 0.570 | 0.587 | 0.589 |
| Lane Keeping | 0.824 | 0.824 | 0.852 | 0.850 |
| Stop | 0.827 | 0.827 | 0.840 | 0.844 |

### *3.5.3.1 Comparative Analysis: Proposed Methodology vs Fixed Time Window Approach*

To further investigate the effectiveness of the EMA in extracting driving events and its relevance in the proposed methodology, the study compared the detection outcome of an EMA-extracted event to the detection outcome of a fixed time moving window approach on a continuous signal, which has been used in several studies (Houenou et al., 2013; Morris et al., 2011; Ohn-Bar et al., 2014). We considered two different fixed time window methods: the three-second moving time window approach and the five-second moving time window approach. The results indicate that the energy maximization algorithm produced consistent results across all three models, whereas the fixed time window approach did not. Also, as shown in Table 3.7, the 5-second fixed moving time window performed better than the 3-second moving time window, which is consistent with results in studies (Houenou et al., 2013; Morris et al., 2011; Ohn-Bar et al., 2014). On a continuous signal, the proposed methodology outperforms fixed moving time window approaches for detecting driving events.

**Table 3. 7: Overall Test Accuracy**

| Model | EMA | moving window (3 seconds) | moving window (5 seconds) |
|---|---|---|---|
| LSTM | 97.75% | 75.91% | 86.52% |
| SVM | 97.65% | 63.22% | 75.87% |
| 1D-CNN | 98.99% | 78.65% | 88.13% |
| RF | 97.71% | 73.98% | 85.89% |

### 3.5.4 Test of Model's Transferability

To test the transferability of the models developed, specifically the 1D-CNN, the study evaluated the developed model on events extracted from three different data sources, namely the SHRP2 NDS dataset (Antin et al., 2019), the Nebraska Medical Center NDS dataset, and data collected via a smartphone (Aboah et al., 2021). The study analyzed about 150 video hours of SHRP 2 dataset, 200 hours of Nebraska Medical Center NDS dataset and 100 hours of smartphone collected dataset. Table 3.8 summarizes the outcomes of the predictions for all three datasets. The F1 scores were high and consistent across all three datasets. The results implies that the developed model and algorithm are easily transferable to predict driving event from signal data from different sensor types.

**Table 3. 8: Comparison of Precision, Recall and F1 Values Obtained All Three Datasets**

| Driving Maneuvers | SHRP2 dataset | | | Smartphone collected dataset | | | Nebraska Medical Center NDS dataset | | |
|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 |
| Right turn | 0.990 | 0.992 | 0.991 | 0.945 | 0.921 | 0.932 | 0.997 | 0.996 | 0.996 |
| Left turn | 0.991 | 0.981 | 0.986 | 0.978 | 0.942 | 0.960 | 0.996 | 0.993 | 0.994 |
| Right curve | 0.992 | 0.993 | 0.992 | 0.969 | 0.962 | 0.965 | 0.986 | 0.991 | 0.988 |
| Left curve | 0.989 | 0.992 | 0.990 | 0.943 | 0.929 | 0.936 | 0.992 | 0.994 | 0.992 |
| Right lane change | 0.986 | 0.956 | 0.971 | 0.889 | 0.901 | 0.895 | 0.995 | 0.965 | 0.980 |
| Left lane change | 0.982 | 0.966 | 0.973 | 0.907 | 0.898 | 0.902 | 0.989 | 0.991 | 0.990 |
| Lane keeping | 0.991 | 0.987 | 0.989 | 0.949 | 0.938 | 0.943 | 0.996 | 0.994 | 0.995 |
| Stop | 0.982 | 0.985 | 0.983 | 0.961 | 0.944 | 0.952 | 0.989 | 0.994 | 0.991 |

Additionally, all four models were combined to create a decision tree-like structure. Where each branch of the tree is a representation of a different model. Extracted events from the segmentation step are classified by passing them through each branch. Following that, the branches vote on the most frequent class. The Table 3.9 below summarizes the analysis's findings. As seen in the Table 3.9, there is a slight improvement in both precision and recall values for all classes on average compared to relying on a single model prediction as illustrated in Table 3.8 for all three datasets.

**Table 3. 9: Combined Model Performance All Three Datasets**

| Driving Maneuvers | SHRP2 dataset | | | Smartphone collected dataset | | | Nebraska Medical Center NDS dataset | | |
|---|---|---|---|---|---|---|---|---|---|
| | P | R | F1 | P | R | F1 | P | R | F1 |
| Right turn | 0.997 | 0.996 | 0.993 | 0.961 | 0.921 | 0.940 | 0.998 | 0.997 | 0.998 |
| Left turn | 0.996 | 0.993 | 0.994 | 0.992 | 0.942 | 0.966 | 0.998 | 0.996 | 0.997 |
| Right curve | 0.997 | 0.995 | 0.997 | 0.989 | 0.962 | 0.975 | 0.992 | 0.994 | 0.993 |
| Left curve | 0.995 | 0.993 | 0.994 | 0.980 | 0.929 | 0.953 | 0.995 | 0.995 | 0.995 |
| Right lane change | 0.991 | 0.989 | 0.990 | 0.902 | 0.901 | 0.902 | 0.996 | 0.985 | 0.990 |
| Left lane change | 0.988 | 0.981 | 0.984 | 0.951 | 0.903 | 0.926 | 0.992 | 0.993 | 0.993 |
| Lane keeping | 0.993 | 0.991 | 0.992 | 0.959 | 0.948 | 0.953 | 0.997 | 0.996 | 0.997 |
| Stop | 0.991 | 0.998 | 0.994 | 0.978 | 0.972 | 0.975 | 0.995 | 0.996 | 0.996 |

### 3.5.5 End-to-End Pipeline for Annotating NDS Videos

Finally, the study developed an end-to-end pipeline that takes the NDS video, gyroscope reading, and vehicle speed as inputs and outputs an annotated video of driving events, as illustrated in [**https://youtu.be/JAuCfRGnLBI**]. To annotate each video frame, the indices of the time series (segmented and classified) are aligned with the video stream, taking into account differences in sampling rates. We interpolate and upsample the vehicle telemetry data if its sampling frequency is higher than the videos frame rate and vise-versa.

### 3.5.6 Application of Research Findings

The primary application of this research is to develop crash countermeasures by better understanding drivers’ behaviors in naturalistic settings, specifically, the drivers’ environment. The results from this study, when conducted on large-scale, will provide insight and the extraction of some critical information such as drivers' lane-changing behaviors (which have recently been the cause of the majority of vehicle crashes on the highway) and turning maneuvers, as well as aggressive driving behaviors, for the purpose of improving traffic safety. The framework for this study is both fast and scalable. As such the framework developed in this study is going to facilitate the annotations of large-scale of NDS videos into various driving events. The extraction of these events will allow for more rapid analysis of conflict zone crashes, especially at intersections (i.e., the extraction of right and left turning events).

## 3.6 Conclusion

To effectively use NDS data to deduce crash causation, algorithms must be developed that can ingest multi-modal NDS data and annotate various driving events pertinent to deducing crash causation. Recent studies have examined the use of shallow and deep machine learning models for driving maneuver detection, obtaining accuracies ranging from 70% to 98%. A significant limitation that these ML approaches do not address is the time series segmentation problem. The current study addressed this limitation by 1) developing an energy maximization algorithm (EMA) that is capable of extracting distinct shapes of driving events from telemetry data. Also, the effectiveness of the EMA was further investigated through the development of four machine learning models.

Multiple sources of data were used in this study including Blackbox sensor data, smartphone data and VTTI dataset. The study accomplished its objectives through the development of a five-stage methodology: 1) preprocessing of data, 2) event segmentation, 3) machine learning classification, 4) heuristics classification, and 5) frame-by-frame annotation of video. To begin, the input data is standardized and smoothed. The resulting output is segmented and then classified using both machine learning (main driving events) and heuristics (stops and lane-keeping). The study separated the detection of stops and lane-keeping from the rest of the driving events because the two can be easily identified using simple thresholding and to reduce false negatives when using only ML to classify all driving events.

The result from the study indicates that the gyroscope reading is a very good parameter to be use in extracting driving events since it showed consistent accuracy across all four developed models. The study shows that the accuracy of the Energy Maximization

Algorithm ranges from 56.80% (left lane change) to 85.20% (lane-keeping) All four models developed had comparable accuracies to studies that used similar models (Bakhit et al. 2017; Kumar et al. 2013; Mandalia and Salvucci 2005; Zheng et al. 2014). The 1D-CNN model had the highest accuracy of 98.99%, followed by the LSTM model at 97.75%, the RF model at 97.71%, and the SVM model at 97.65%. To serve as a ground truth, continuous signal data was annotated. Also, the proposed methodology outperformed the fixed time window approach when compared. The study further analyzed the accuracy of the overall pipeline by penalizing the F1 scores of the ML models with the duration score of the EMA. The overall accuracy of the pipeline was in the range of 56.8% to 85.2%. To test the model's transferability, the developed models were used to detect driving events from multiple streams of datasets. The F1 scores were high and consistent across all three datasets used. The predicted results were compared to the ground truth annotations. Using the LSTM model, the test was 91% accurate.

The study did not take advantage of large database of video data acquired; Future work should consider integrating the video data, with other predictive models such as eye detection model, and object detection models to better understand the driver’s behavior.

### 3.6.1 Limitations to Study

One of the challenges encountered in this study was dealing with outliers due to anomalous behaviors of drivers. The data outliers are due to false spikes in the gyroscope readings caused by the driver's activity in the vehicle. For instance, a spike in the gyroscope reading can be observed when the driver is dancing or drinking while keeping a lane or at a stop. While these spikes are not considered events, the EMA will extract them as events and pass them through the classification algorithm. These outliers contribute to the increased detection of false positives.

## 3.7 References

Aboah, A., Boeding, M., and Adu-Gyamfi, Y. (2021). “Mobile Sensing for Multipurpose Applications in Transportation.”

Antin, J. F., Lee, S., Perez, M. A., Dingus, T. A., Hankey, J. M., and Brach, A. (2019). “Second strategic highway research program naturalistic driving study methods.” *Safety Science*, Elsevier, 119, 2–10.

“ASIRT — Association for Safe International Road Travel.” (n.d.). <https://www.asirt.org/> (Nov. 16, 2021).

Atia, A., Mostafa, M.-S., Hamdy Ali, A., and Sami, M. (2017). “Recognizing Driving Behavior and Road Anomaly using Smartphone Sensors Driving Events Recognition using Smartphone Sensors.” *Article in International Journal of Ambient Computing and Intelligence*.

Ayres, G., Wilson, B., and LeBlanc, J. (2004). “Method for identifying vehicle movements for analysis of field operational test data.” *Transportation Research Record,* National Research Council, (1886), 92–100.

Bakhit, P., Osman, O., Research, S. I.-T., and 2017, undefined. (2017). “Detecting imminent lane change maneuvers in connected vehicle environments.” *journals.sagepub.com*, SAGE Publications Ltd, 2645(1), 168-175.

Barnard, Y., Utesch, F., van Nes, N., Eenink, R., & Baumann, M. (2016). The study design of UDRIVE: the naturalistic driving study across Europe for cars, trucks and scooters. *European Transport Research Review*, *8*(2), 14.

Bejani, M. M., & Ghatee, M. (2019). “Convolutional neural network with adaptive regularization to classify driving styles on smartphones”. IEEE transactions on intelligent transportation systems, 21(2), 543-552.

Benmimoun, M., Fahrenkrog, F., Zlocki, A., & Eckstein, L. (2011, June). “Incident detection based on vehicle CAN-data within the large-scale field operational test “euroFOT”. In 22nd Enhanced Safety of Vehicles Conference (ESV 2011), Washington, DC/USA.

Bhoraskar, R., Vankadhara, N., Raman, B., & Kulkarni, P. (2012). “Wolverine: Traffic and road condition estimation using smartphone sensors”. In 2012 fourth international conference on communication systems and networks (COMSNETS 2012) (pp. 1-6). IEEE.

Bogard, S. (1999). “Analysis of data on speed-change and lane-change behavior in manual and ACC driving.”

Borkar, A., Hayes, M., and Smith, M. T. (2012). “A novel lane detection system with efficient ground truth generation.” *IEEE Transactions on Intelligent Transportation Systems,* 13(1), 365–374.

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5-32.

Carvalho, E., Ferreira, B. V., Ferreira, J., De Souza, C., Carvalho, H. V., Suhara, Y., ... & Pessin, G. (2017, May). Exploiting the use of recurrent neural networks for driver behavior profiling. In *2017 International Joint Conference on Neural Networks (IJCNN)* (pp. 3016-3021). IEEE.

Charlton, J. L., Koppel, S., D'Elia, A., Hua, P., st. Louis, R., Darzins, P., di Stefano, M., Odell, M., Porter, M., Myers, A., Tuokko, H., and Marshall, S. (2019). "Changes in driving patterns of older Australians: Findings from the Candrive/Ozcandrive cohort study." *Safety Science*, Elsevier, 119, 219–226.

Chen, D., Cho, K. T., Han, S., Jin, Z., & Shin, K. G. (2015, May). Invisible sensing of vehicle steering with smartphones. *In Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services* (pp. 1-13).

Cortes, C., learning, V. V.-M., and 1995, undefined. (1995). "Support-vector networks." *Springer*, Kluwer Academic Publishers, 20, 273–297.

Drincic, A., Rizzo, M., Desouza, C., Health, J. M.-D. D., and 2020, undefined. (n.d.). "Digital health technologies, diabetes, and driving (meet your new backseat driver)." *Elsevier*.

Freund, Y., icml, R. S.-, and 1996, undefined. (1996). "Experiments with a new boosting algorithm." *Citeseer*.

Fridman, L., Brown, D. E., Glazer, M., Angell, W., Dodd, S., Jenik, B., ... & Reimer, B. (2019). MIT advanced vehicle technology study: Large-scale naturalistic driving study of driver behavior and interaction with automation. *IEEE Access*, *7*, 102021-102038.

Gao, D., Li, W., Duan, J., and Zheng, B. (2009). "A practical method of road detection for intelligent vehicle." *Proceedings of the 2009 IEEE International Conference on Automation and Logistics, ICAL 2009*, 980–985.

"Global status report on road safety - Google Search." (2021.). <https://www.google.com/search?q=global+status+report+on+road+safety& (Nov. 16, 2021).

Hankey, J., Perez, M., and McClafferty, J. (2016). "Description of the SHRP 2 naturalistic database and the crash, near-crash, and baseline data sets."

Houenou, A., Bonnifait, P., Cherfaoui, V., and Yao, W. (2013). "Vehicle trajectory prediction based on motion model and maneuver recognition." *IEEE International Conference on Intelligent Robots and Systems*, 4363–4369.

Júnior, J. F., Carvalho, E., Ferreira, B. v., de Souza, C., Suhara, Y., Pentland, A., and Pessin, G. (2017). "Driver behavior profiling: An investigation with different smartphone sensors and machine learning." *PLoS ONE*, Public Library of Science, 12(4).

Kang, L., & Banerjee, S. (2017, November). Practical driving analytics with smartphone sensors. *In 2017 IEEE Vehicular Networking Conference (VNC)* (pp. 303-310). IEEE

Karatas, C., Liu, L., Li, H., Liu, J., Wang, Y., Tan, S., Yang, J., Chen, Y., Grusteser, M., & Martin, R. (2016, April). Leveraging wearables for steering and driver tracking. *In IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications* (pp. 1-9). IEEE.

Kong, H., Audibert, J. Y., & Ponce, J. (2010). General road detection from a single image. *IEEE Transactions on Image Processing*, *19*(8), 2211-2220.

Kumar, P., Perrollaz, M., Lefevre, S., & Laugier, C. (2013). Learning-based approach for online lane change intention prediction. In 2013 IEEE Intelligent Vehicles Symposium (IV) (pp. 797-802). IEEE.

Larue, G. S., Demmel, S., Ghasemi Dehkordi, S., Rakotonirainy, A., Grzebieta, R., Williamson, A., Charlton, J., Haworth, N. L., Wooley, J., Senserrick, T., and Young, K. (2018). "Australian Naturalistic Driving Study (ANDS): Using 20,000 trips to get a glimpse at locations and speeds where data was collected.” *2018 Australasian Road Safety Conference*.

Li, P., Abdel-Aty, M., and Islam, Z. (2021). “Driving Maneuvers Detection using Semi-Supervised Long Short-Term Memory and Smartphone Sensors.” *Transportation Research Record: Journal of the Transportation Research Board*, SAGE Publications, 2675(9), 1386–1397.

Li, W., Gong, X., Wang, Y., and Liu, P. (2014). “A lane marking detection and tracking algorithm based on sub-regions.” *ICCSS 2014 - Proceedings: 2014 International Conference on Informative and Cybernetics for Computational Social Systems*, Institute of Electrical and Electronics Engineers Inc., 68–73.

Mandalia, H. M., and Salvucci, D. D. (2005). “Using support vector machines for lane-change detection.” *Proceedings of the Human Factors and Ergonomics Society*, 1965–1969.

McGehee, D. V., Raby, M., Carney, C., Lee, J. D., & Reyes, M. L. (2007). Extending parental mentoring using an event-triggered video intervention in rural teen drivers. *Journal of safety research*, *38*(2), 215-227.

Miller, R. J., & Srinivasan, G. (2005, June). Determination of Lane Change Maneuvers Using Naturalistic Driving Data. In *19th International Technical Conference on the Enhanced Safety of Vehicles (ESV), Washington, DC* (No. 05-0337).

Mohamed, A., Issam, A., Mohamed, B., and Abdellatif, B. (2015). “Real-time Detection of Vehicles Using the Haar-like Features and Artificial Neuron Networks.” *Procedia Computer Science*, Elsevier B.V., 73, 24–31.

Morris, B., Doshi, A., and Trivedi, M. (2011). “Lane change intent prediction for driver assistance: On-road design and evaluation.” *IEEE Intelligent Vehicles Symposium, Proceedings*, 895–901.

Lerner, N., Jenness, J., Singer, J., Klauer, S., Lee, S., Donath, M., ... & Ward, N. (2010). An exploration of vehicle-based monitoring of novice teen drivers: Final report. *Report number DOT HS*, *811*, 333.

Naiel, M. A., Ahmad, M. O., and Swamy, M. N. S. (2014). “Vehicle detection using TD2DHOG features.” *2014 IEEE 12th International New Circuits and Systems Conference, NEWCAS 2014*, Institute of Electrical and Electronics Engineers Inc., 389–392.

Nhtsa, U. (1999). “Evaluation of the intelligent cruise control system: volume 1: study results.”

Ohn-Bar, E., Tawari, A., Martin, S., and Trivedi, M. M. (2014). “Predicting driver maneuvers by learning holistic features.” *IEEE Intelligent Vehicles Symposium, Proceedings*, Institute of Electrical and Electronics Engineers Inc., 719–724.

Olson, R., Hanowski, R., Hickman, J., and Bocanegra, J. (2009). “Driver distraction in commercial vehicle operations.”

Panichpapiboon, S., & Leakkaw, P. (2020). Lane Change Detection with Smartphones: A Steering Wheel-Based Approach. IEEE Access, 8, 91076-91088.

Perslev, M., Jensen, M. H., Darkner, S., Jennum, P. J., & Igel, C. (2019). U-time: A fully convolutional network for time series segmentation applied to sleep staging. arXiv preprint arXiv:1910.11162.

Pilgerstorfer, M., Runda, K., Brandstätter, C., and Christoph, M. (2012). “Deliverable 6.3, Report on Small Scale Naturalistic Driving Pilot.” *Small Scale Naturalistic Driving Pilot, Deliverable,* 6(3).

Rotaru, C., Graf, T., and Zhang, J. (2008). “Color image segmentation in HSI space for automotive applications.” *Journal of Real-Time Image Processing*, 3(4), 311–322.

Saiprasert, C., Pholprasit, T., & Pattara-Atikom, W. (2013). Detecting driving events using smartphone. In *Proceedings of the 20th ITS World Congress* (Vol. 11, p. 11).

Saiprasert, C., Pholprasit, T., and Thajchayapong, S. (2017). “Detection of Driving Events using Sensory Data on Smartphone.” *International Journal of Intelligent Transportation Systems Research*, Springer New York LLC, 15(1), 17–28.

Son, J., Yoo, H., Kim, S., and Sohn, K. (2015). “Real-time illumination invariant lane detection for lane departure warning system.” *Expert Systems with Applications*, Elsevier Ltd, 42(4), 1816–1824.

Sun, R., Cheng, Q., Xie, F., Zhang, W., Lin, T., & Ochieng, W. Y. (2019). Combining machine learning and dynamic time wrapping for vehicle driving event detection using smartphones. IEEE Transactions on Intelligent Transportation Systems.

Tavakoli, A., and Heydarian, A. (2021). “Multimodal Driver State Modeling through Unsupervised Learning.”

“The 100-Car Naturalistic Driving Study Phase II-Results of the 100-Car Field Experiment.” (2006).

Xu, H., Wang, X., Huang, H., Wu, K., and Fang, Q. (2009). “A fast and stable lane detection method based on B-spline curve.” *Proceeding 2009 IEEE 10th International Conference* *on Computer-Aided Industrial Design and Conceptual Design: E-Business, Creative Design, Manufacturing - CAID and CD'2009*, 1036–1040.

Xuan, Y., and Coifman, B. (2006). "Lane change maneuver detection from probe vehicle DGPS data." *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 624–629.

Yang, D., Qiu, X., Liu, Y., Wen, C., Zhu, L., and Hong, X. (2017). "Modeling the Discretionary Lane-Changing Decision Behavior on Freeways Using Random Forest Theory."

Yang, M., Wang, X., & Quddus, M. (2019). Examining lane change gap acceptance, duration and impact using naturalistic driving data. *Transportation research part C: emerging technologies*, *104*, 317-331.

You, C. W., Lane, N. D., Chen, F., Wang, R., Chen, Z., Bao, T. J, et al. (2013, June). Carsafe app: Alerting drowsy and distracted drivers using dual cameras on smartphones. *In Proceeding of the 11th annual international conference on Mobile systems, applications, and services* (pp. 13-26).

Yu, J., Chen, Z., Zhu, Y., Chen, Y., Kong, L., & Li, M. (2016). Fine-grained abnormal driving behaviors detection and identification with smartphones. *IEEE transactions on mobile computing*, *16*(8), 2198-2212.

Zheng, J., Suzuki, K., and Fujita, M. (2014). "Predicting driver's lane-changing decisions using a neural network model." *Simulation Modelling Practice and Theory*, 42, 73–83.

# CHAPTER 4: AI-BASED FRAMEWORK FOR UNDERSTANDING CAR-FOLLOWING BEHAVIORS OF DRIVERS IN A NATURALISTIC DRIVING ENVIRONMENT

## 4.1 Introduction

### 4.1.1 Background

The rear-end crash is the most common type of accident encountered on the highway, according to statistical data for related accidents. These crashes have a significant negative impact on the flow of traffic and typically result in serious repercussions (Zheng and Sarvi, 2016). An examination of the sequence of events that lead up to rear-end crashes can be broken down into two distinct scenarios. The first scenario is one in which the following distance between vehicles is so close that if the car in front of them suddenly brakes, it is possible for the driver behind them to be involved in a rear-end crash, even if they are able to apply their brakes in time. In the second scenario, the driver is maintaining a car-following distance that is relatively safe; however, when the car ahead of them applies the brakes, the driver is either distracted or fatigued, and as a result, the driver does not notice that the car ahead of them is applying the brakes. This circumstance is very similar to the one in which there is absolutely no braking at all. To gain a better understanding of these scenarios from a practical standpoint, it is necessary to accurately model car-following behaviors that lead to rear-end crashes and that is what this study seeks to achieve.

The aforementioned scenarios of rear-end crashes each have a significantly high incidence rate. As a result, some automobile companies have begun encouraging the installation of a warning system for rear-end crashes. Alarm systems for rear-end crashes are able to monitor the motion of the car in front of them by using radars or visual sensors, and they can sound an alarm if they detect an impending risk. Some drivers may develop a hostile attitude toward other drivers during car following if the alarms are set off too frequently (Hoogendoorn et al., 2010). This is because different drivers have different driving styles. In order to reduce the likelihood of the driver becoming distracted by the alarm, these systems typically set the timer for the alarm to go off when the vehicle is already in a relatively hazardous state. However, the timing of the warnings has a significant impact on how well the warning system works. There is a potential for an increase in the number of accidents if the warning time is delayed (Tang and Yip, 2010). This discussed challenge can easily be resolved if acceleration of the leading vehicle can be correctly estimated relative to the acceleration of the ego-vehicle. The current study thereby seeks to develop models that are capable of predicting the acceleration of the leading vehicle as well as the ego-vehicle.

### 4.1.2 Motivation

Although a number of studies have been conducted to model the car-following behaviors of drivers, the majority of these studies have relied on simulated data that may not accurately represent incidents that occur in the real world. In addition, very few longitudinal studies have been conducted on the car-following behavior of drivers in naturalistic environments. These studies, however, are limited to developing models that can only estimate the acceleration of the ego-vehicle, which is insufficient to explain the behavior of the ego-driver. This limitation exists because data to model the acceleration of the leading vehicle is rarely available when using naturalistic driving dataset. As such, the current study attempts to address this issue by modeling both the acceleration of the ego-vehicle and the leading vehicle through the development of an AI framework capable of extracting parameters from NDS videos necessary to model the behavior (acceleration) of the leading vehicle as well as the ego-vehicle. In addition, there have been no previous longitudinal studies of the car-following behavior of various demographics in a naturalistic environment. Such research is important because it enhances our understanding of the driving styles of various demographic groups and the causes of crashes. The study addresses this deficiency by conducting longitudinal studies of different demographic groups of drivers.

### 4.1.3 Objectives

In light of the gaps identified in previous studies, the study seeks to primarily develop an AI framework to extract parameters from naturalistic videos that provides better insight to drivers' behavior in a naturalistic environment. To achieve this goal, we formulated three objectives described below.

1. First, the study develops a framework for extracting features pertinent to comprehending the behavior of drivers in natural environments. To achieve this objective, we investigated monocular depth estimation techniques for determining the distance between the leading vehicle and the ego-vehicle. Moreover, we determine the relative velocities of the leading vehicle and the ego-vehicle using optical flow. The study further relied on the relative velocity, car-following distance and the acceleration of the ego-vehicle to estimate the acceleration of the leading vehicle.

2. Second, we analyzed the car-following behaviors of various demographic groups. To accomplish this objective, numerous visualization plots and statistical tests were conducted. The study employed the sample mean t-statistics to determine whether different demographic groups exhibit distinct driving behavior.
3. Third, we modeled the acceleration of both the ego-vehicle and the leading vehicle using a machine learning algorithm. We utilized the XGBoost algorithm to develop the various acceleration models. The explanatory variables used in the models were car-following distance, relative velocity, and ego-vehicle acceleration (only used to model the acceleration of the leading vehicle). We investigate further the variables that best explain the leading and ego-vehicle accelerations.

## 4.2 Literature Review

Understanding driver car-following behaviors is a critical step in developing crash countermeasures to reduce rear-end collisions. Over the last two decades, there has been an enormous amount of research into understanding and modelling driver car-following behavior. As a result, this section provides reviews of car-following studies as well as other relevant literature pertaining to the current research work. First, we review various studies on car-following behavior conducted in the past two decades. Second, we go over the various monocular depth estimation approaches that have been used in previous studies.

### 4.2.1 Modelling Car-following Behaviors

On the topic of car following safety, a great number of studies have been conducted, each of which has approached the topic from a slightly different direction. Kometani, (1959) came up with the idea for a car-following model that was based on the safety distance. This distance is the shortest one that must be traveled in order to avoid a crash in the event that the vehicle in front of you begins to use its emergency brakes. Car-following was broken up into two phases according to Treitere et al., (1974) who began their research with the General Motors model as their foundation. This process involves both quickening and slowing down of speed at various points. Car-following, according to Aron, (1988), can be broken down further into 3 distinct stages: deceleration, acceleration, and maintaining. Helly, (1959) recommended making use of a linear model that takes into account the distance traveled in addition to the acceleration and the relative speed. Peter, (1998) made a suggestion for a model for the desired spacing based on the findings of the research that concerned the spacing distance that was desired. Numerous studies concentrated their attention on the car-following model with regard to the speed of the vehicles and the distance between them. Following the experiment, Jiang et. al. (2015) carried out a motorcade in order to investigate the relationship between the speed of a car and the distance it kept behind another car in a number of different types of traffic. When following a vehicle, it is extremely important, as stated by Tang et al., to take into consideration both the forward and the backward safe distances. The findings of the study indicate that models which took into account both types of safe distances performed appreciably better than those which did not (Tang et. al., 2017). Gipps, (1981) laid the groundwork for subsequent research on car following by providing a theoretical framework in which he proposed a driver model that could be simulated, and which followed a car. After that, he carried out a correlation analysis by making use of parameters derived from the actual flow of traffic (Gipps, 1981). Because of this, future research on car following will have a theoretical foundation to build upon. Zhou et al. developed a more reliable model for car following by taking into account the motion characteristics of the vehicle in front of them (Zhou et. al., 2014). According to the findings of Tang et al. (2014), Yang et al. (2017), and Tordeux et al. (2010), the headway is an important parameter that plays a role in the evaluation of the driver's level of risk. In addition to the rate of driving, another factor to take into account is the time headway (THW).

### 4.2.2 Monocular Depth Estimation

Over the recent years, there have been many deep learning networks and their variants that perform monocular depth estimation (Eigen et al. (2014); Zheng et al. (2018); Yin et al. (2019); Ranftl et al. (2020); Wang et al. (2020)). These networks provide superior performance with various architectural transformation, data augmentation strategies and innovative cost functions. Many of these neural networks however do not focus on providing high resolution/quality of dense depth map (Silberman et al. (2012); Wadhwa et al. (2018); Wang et al. (2018)). Some of them which produce high quality depth map do require complex architectures. These complex networks use numerous deep layers, residual modules, guiding modules, sequential modules, attention-based modules etc. (Laina et al. (2016); Xu et al. (2017); Xu et al. (2018); Fu et al. (2018); Swami et al. (2020)) to achieve the complex task of monocular depth estimation. Laina et al., (2016) uses a deep layer CNN confining residual learning to improve output depth map resolution. Xu et al., (2017) proposes a novel sequential framework that fuses multi-scale convolutional neural network with continuous conditional random field module for accurate depth maps. Hao et al., (2018) model use dilated convolutions and attention mechanism for extracting multi-scale feature from input image while maintaining dense feature maps and fuse multi-scale features respectively to produce high quality depth map. Xu et al., (2018) uses multi-scale convolutional neural network along with conditional random field and structural attention module. Fu et al., (2018) proposes a deep ordinal regression network that discretizes depth and remodel learning of the depth estimation network as an ordinal regression problem. Swami et al., (2020) is an improved version to the Fu et al., (2018) which use fully differentiable ordinal and pixel-wise regression network along with depth refinement module for improved depth estimation. Recently, there has also been a lot of research to achieve high quality monocular depth estimation using Transformer architecture (Xie et al. (2020); Ranftl et al. (2021)) due to its success achieved in the other computer vision related tasks like image classification, image segmentation etc. Also, the same was observed with deep encoder-decoder (Ummenhofer et al. (2017); Zhou et al. (2018)) type of structures. However, these approaches are computationally expensive which impedes its embedding in the edge technology. Therefore, we aim to analyze simple architectures with less computation and comparable state-of-the-art performance for the task of monocular depth estimation (Ibraheem et al. (2018); Doyeon et al. (2022); Miangoleh et al. (2021)).

## 4.3 Data

The data collection for this study was carried out with the assistance of Blackbox sensors, which were created by Digital Artifacts LLC. The sensors were put into individual, privately owned vehicles with the purpose of continuously recording activities that took place both inside and outside of the vehicle. The sensor instrumentation incorporates a number of sensors, including high-resolution cameras, infrared sensors, high-precision GPS, and wireless onboard diagnostics (OBD). The sensor package that is mounted on the windshield can be seen in **Figure 4.1**. This package is mounted behind the rear-view mirror. The system is equipped with two cameras, one of which captures 1) a view of the roadway in front of the vehicle at all times, and

2) a view of the driver as well as the interior of the vehicle. From the moment the ignition key is turned to the moment it is turned off, the behavior of the driver is being continuously recorded. The research involved 77 people who took part and was conducted over the course of three months. Data were gathered on a total distance of 289681.9 kilometers across the entirety of the United States. This dataset contains much more detailed information on driver behavior across a wide range of geographic environments than laboratory-based or retrospective studies are able to provide.

In this study, we only utilized data information from four individuals. They were made up of two elderly drivers and two young drivers.

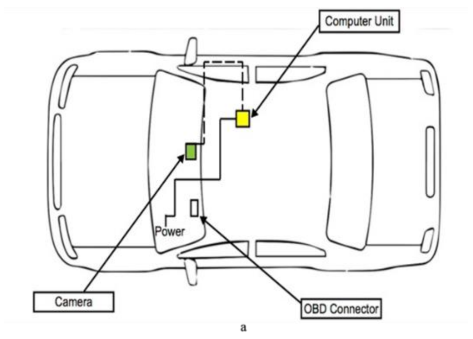


**Figure 4. 1: The Positioning of the Blackbox sensors in the vehicle**

## 4.4 Methodology

The general methodological framework shown in **Figure 4.2** can be group into two main stages. First, we extract the car following distance using distance obtained from the monocular depth estimation and tracking of the leading vehicle (LV). Second, we estimate the acceleration of the LV by combining the estimated car-following distance, relative velocity and the acceleration of the ego-vehicle.

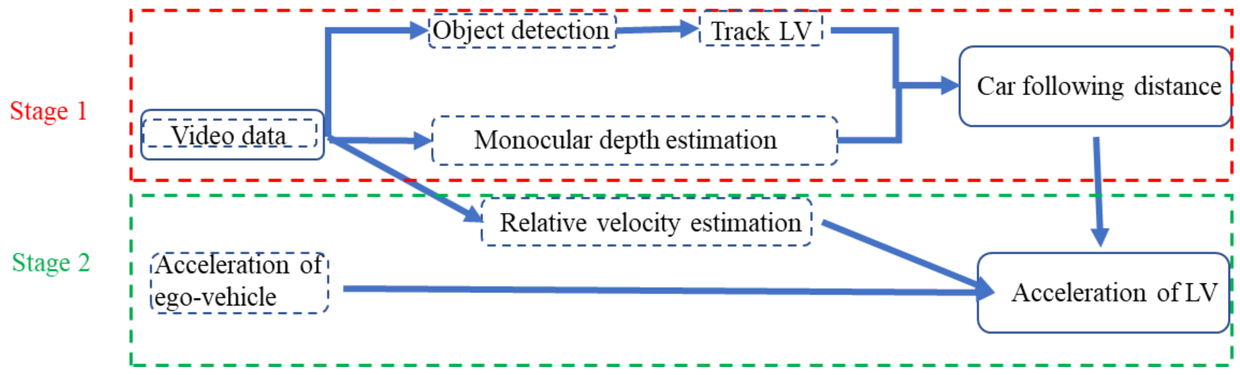


**Figure 4. 2: Methodological Framework**

### 4.4.1 Extraction of car-following distances

The process involved in the extraction of car-following distances can be divided into three major steps. First, training an object detection model to detect vehicles and traffic signs. Second, using a trained monocular depth estimation model to estimate the scene depth of the video images. Finally, combine the first two steps with a heuristic algorithm for tracking leading vehicles to estimate the car-following distance.

#### *4.4.1.1 Object detection*

The state-of-the-art single stage object detection algorithm used for the developing the detection model is YOLOv5. The YOLOv5 network consists of three main pieces viz. Backbone, Neck and Head. The Backbone consists of a convolutional neural network that bundles and forms image representational features at contrasting granularities. The architecture's neck consists of a series of layers which blends and integrates image representational features to proceed further with prediction.  Similarly, the head utilizes features from the neck and gets hold of box and class prediction functionality. CSPDarknet53 backbone within YOLOv5 contains 29 convolutional layers 3 × 3, receptive field size of 725 × 725 and altogether 27.6 M parameters. Besides, the SPP block attached over YOLO's CSPDarknet53 expands the proportion of receptive fields without influencing its operating speed. Likewise, the feature aggregation is performed through

PANet by exploiting different levels of backbone. YOLOv5 pushes state-of-the-art by using features such as the weighted-residual-connections, cross-stage partial-connections, cross mini-batch, normalization and self-adversarial training, making it exceptionally efficient. In the current study, we trained and deployed our YOLOv5 model on the PyTorch framework. To further accomplish the task of vehicle detection, the YOLOv5 model is fine-tuned by adjusting to the following hyperparameters: batch-size 64, the optimizer weight decay value of 0.0005, setting the initial learning rate of 0.01 and keeping the momentum at 0.937.

*4.4.1.2 Monocular depth estimation*

The study utilized different monocular depth estimation models developed by [1,2,3] to estimate the car's following distance. Each model is explained in detail in the sections below

a) **High quality monocular depth estimation via transfer learning**: A simple encoder-decoder UNet type of machine learning model was used for the task of high-quality depth estimation. Fig. 3. shows the encoder-decoder neural network model [1]. The encoder used in the model is the truncated encoder structure of high-performing DenseNet169 architecture. For this encoder, the DenseNet169 model is truncated with the top layers leaving the bottom deep layers. The DenseNet169 is pretrained on the ImageNet dataset for the object classification task. This process of leveraging a high-performance neural network trained on different tasks for initializing the model with the more complex tasks is called transfer learning.

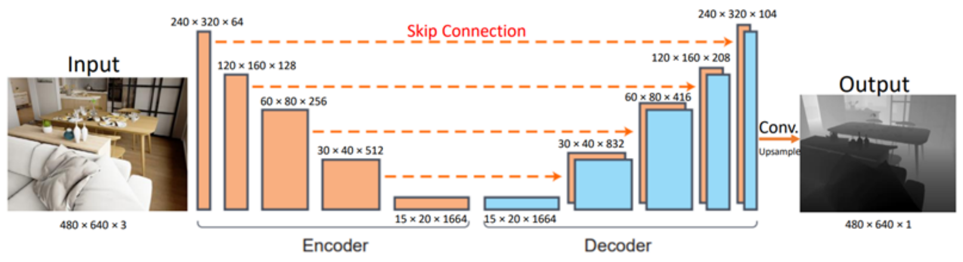


**Figure 4. 3: Pipeline of the Proposed Method**

The encoder consists of the 3x3 convolution layers and downsampling layers which decrease the feature size and increase the number of feature maps along the process to get the final bottleneck features of the image. The encoder also consists of the dense and transition block which is one of the advantages of the DenseNet169 networks. The dense block performs convolution and pooling operation where each processing node gets the input from all the previous node outputs. The dense block, therefore, encourages feature propagation and reuse, which can be useful for understanding. The dense block reduces the feature size and increases the number of feature maps which can increase the trainable parameters. Therefore, to take care of this, the transition block reduces the number of feature maps of the dense block output. Once, we get the bottleneck features from the encoder. These features are given as input to the decoder.

The decoder consists of the up-sampling layers, which use bilinear sampling, and convolution layers. The convolution layers get the feature input from the previous layer of the decoder output and from the same scale encoder convolution layer output. At each layer the decoder performs up-sampling and convolution which decreases the number of feature maps to half and increases the feature map size. The model also consists of skip connections which concatenate the encoder features to the decoder. The skip connections help to regain the details that can be lost during down sampling. It also helps to avoid vanishing and exploding gradient issue as the gradient can travel directly from decoder to the corresponding scale encoder using skip connection during backpropagation. Finally, the decoder predicts the depth map at the output.

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y}) \quad (1)$$

$$L_{depth}(y, \hat{y}) = \frac{1}{n} \sum_{\mathrm{p}}^{n} \left| y_p - \hat{y}_p \right| \qquad (2)$$

$$L_{grad}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} \left| g_x(y_p, \hat{y}_p) \right| + \left| g_y(y_p, \hat{y}_p) \right| \qquad (3)$$

$$L_{SSIM}(y, \hat{y}) = 1 - \frac{SSIM(y, \hat{y})}{2} \qquad (4)$$

The loss function used for the training has a significant effect on the overall performance of the model. For this model, three terms in the loss function are used as shown in equation (1). The first term $L_{depth}$ shown in equation (2) defines the general depth reconstruction error that tries to minimize the difference between the ground truth and the predicted depth map. This is the common loss term used by the depth estimation models, where n is the total number of pixels in depth map, is the ground truth pixel depth value, is the predicted pixel depth value, and λ weight is taken equal to 0.1. The second loss term $L_{grad}$ shown in equation (3) penalizes the model if there are distortions of high-frequency details like boundaries of the objects in the scene. This is done by calculating depth map gradient g() in x and y directions. The loss term then will ensure that the depth map gradient of the ground truth and prediction are as close as possible. The third loss term $L_{SSIM}$ shown in equation (4) uses SSIM function to calculate the similarity between the ground truth and the predicted depth map. 1-SSIM means the dissimilarity value which ensures that the dissimilarity between the ground truth and the predicted depth map is as low as possible.

b) **Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth**: The study used an encoder-decoder model shown in **Figure 4.4**. The model uses a transformer encoder and decoder with skip connections called selective feature fusion (SFF) module to capture the global image context and local connectivity respectively. The proposed model is a global-local path network that extracts significant features on diverse scales and effectively delivers them throughout the network.

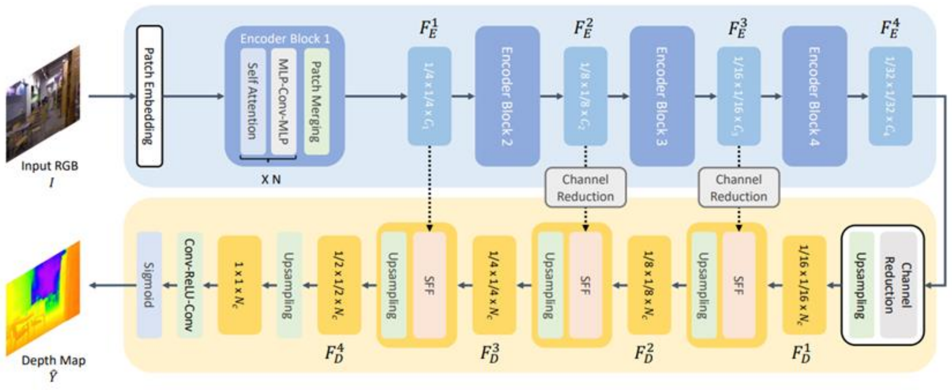


**Figure 4. 4**: **Pipeline of the Proposed Method**

The hierarchical transformer is used as the encoder to capture global relationships. First, the input image is embedded as a sequence of patches with a 3 × 3 convolution operation. These embedded patches are then used as an input to the transformer block, which consists of multiple sets of self-attention and the MLP-Conv-MLP layer with a residual skip. Finally, the transformer output undergoes patch merging with overlapped convolution. The model uses four transformer blocks to get multi-scale features.

The decoder with an effective fusion module is then used to capture local features to produce a fine depth map while preserving structural details. For this the bottle neck features channel dimensions from the encoder are first reduced using 1 x 1 convolution. Then the decoder uses consecutive bilinear up-sampling to enlarge the feature size and SFF module. Finally, the output is passed through two convolution layers and a sigmoid function to predict the depth map.

To exploit the local structures having fine details, the network combines the encoded and decoded features using skip connections with the input-dependent feature fusion module SSF. The SFF module helps the network to selectively focus on the salient regions by estimating the attention map for both encoder and decoder features. For this first the channel dimensions of the decoded features FD and encoded features FE are matched with the convolution layer and provided as input to the SFF module. The SFF then concatenates these features along the channel dimension and pass it through two 3 × 3 Conv-batch_normalization-ReLU layers. Finally, the convolution and sigmoid layers produce a two-channel attention maps which are then multiplied to each local and global feature to focus on the significant locations. These multiplied features are then added elementwise to construct a hybrid feature.

$$L = {}^{1}/_{n} \sum_{i} d_i^2 - {}^{1}/_{2n^2} \sum_{i} d_i^2 \quad (5)$$

$$d_i = \log y_i - \log y_i^* \quad (6)$$

The loss function for the model uses scale-invariant log scale loss as shown in equation (5) and (6) where and indicates the i$^{th}$ pixel value in the ground truth and predicted depth map.

C) **Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging**: The architecture of the model is such that a double-depth-estimation network is analyzed that combines the two depth estimations of the same image at different resolutions adaptive to the image content to generate a result with high-frequency details while maintaining the structural consistency. It is observed that the low-resolution input to the network produces structural consistent depth maps as they learn the overall global content in the image while the high-resolution input captures the high frequency details but loses the overall structure of the scene generating low-frequency artifacts in the depth estimate. The proposed model therefore embeds the high-frequency depth details of the high-resolution patches into the structural consistent depth of the small resolution input that provides a fixed range of depths for the full image.

First the high-resolution inputs are created by selecting the patches from the input image of resolutions adaptive to the local depth cue density to be feed to the network. For creating image adaptive patch resolution containing contextual cues image edge map is computed, using gradients and thresholding, as edge maps are correlated to the contextual hues. The edge map is then used to determine the maximum resolution where every pixel in the patch has contextual information. Finally, the patches are selected by initiating their size equal to the receptive field. Their size is then increased if the edge density is higher than the image density until their density becomes equal while the patches with less edge density are discarded.

The created high-resolution patches and the low-resolution image are now provided as input to the network to produce depth estimates.  Then the depth estimate of the patches are combined onto a low-resolution structurally consistent base depth estimate to achieve a highly detailed high-resolution depth estimation. For this the model uses a standard Pix2Pix network architecture with a 10-layer U-Net. The 10-layer U-Net aims to increase the training and inference resolution, as this merging network will be used for a wide range of input resolution. The network is finally trained to transfer the fine-grained depth details from the high-resolution input patches to the low-resolution input depth estimate.

*4.4.1.3 Determining the leading vehicle*

We relied on a set of heuristic algorithms to determine the leading vehicle among all vehicles identified in an image. To begin, we determined that the image's center was roughly the vanishing point of the road lane. Multiple iterations revealed that multiplying the image's base by a factor of 0.2 and 0.8 yields a triangulation that corresponds to the lane of the ego-vehicle. We compute the gradient for each base coordinate point and the center point (vanishing) of the line. Then, we determine if the predicted vehicle bounding boxes' base coordinates lie within the triangle. If both base points fall within the triangle, then that vehicle is the identified as the leading vehicle.

### 4.4.2 Estimating the Acceleration of the Leading Vehicle

In order to estimate the acceleration of the leading vehicle, we first estimated the relative velocities between the leading vehicle and ego-vehicle using optical flow. We then adopted Newton’s second equation of motion which relates distance, velocity and acceleration as shown in Equation 1. We modified the Equation 7 to reflect what is seen in Equation 8. Where the change in distance is the car following distance between the ego-vehicle and the leading vehicle. Whereas the relative velocity between the ego-vehicle and the LV was used as the change in velocity in that equation. Finally, the change in acceleration was model as the difference between the acceleration of the ego-vehicle minus the acceleration of the LV. From Equation 8, we are able to estimate the acceleration of the leading vehicle.

$$s = ut + \frac{1}{2}at^2 \quad (7)$$

$$\Delta s = \Delta ut + \frac{1}{2}\Delta at^2 \quad (8)$$

## 4.5 Results

### 4.5.1 Comparative analysis of monocular depth estimation methods

In this study, three monocular depth estimation models were evaluated and compared as shown in **Figure 4.5**. The depth estimation of each model was compared to the true depth as determined by a lidar scan. We calculate the root mean squared error depth between the ground truth and each model's estimated depth. The model with the least root mean squared error was selected as the optimal model for the study.

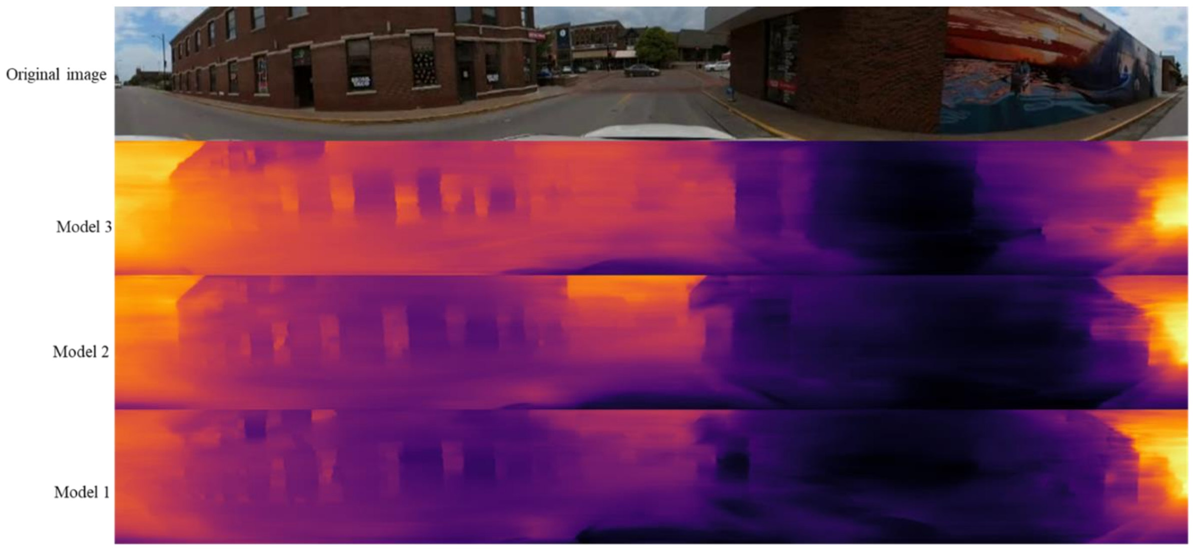


**Figure 4. 5**: **Comparing depth estimation of the three monocular depth estimation models**

**Figure 4.6** compares the estimated distances to the actual distances for each model. Model 1 performs better at predicting the depth of objects when they are closer, but utterly fails when they are farther away. The same could be said for Model 2, with the exception that it performs marginally better than Model 1 when predicting depth of distant objects, as illustrated in **Figure 4.6**. Model 3 has the greatest correlation between predicted and actual distances when compared to Models 1 and 2. The study further analysis these models by comparing the root mean square error.

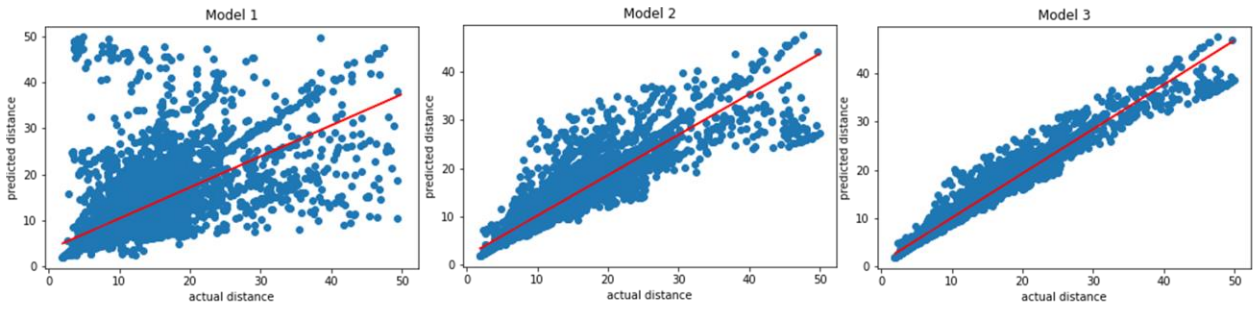


**Figure 4. 6: Comparative analysis of the prediction accuracy of**

Table 4.1 compares the root mean squared error of the three models. Model 3 had the least squared error of 1.79. Followed by model 2 of root mean squared error of 4.58 and lastly model 1 of root mean squared error of 6.23. Based on the results from Table 1. The study settles on model 3 for the rest of its analysis.

**Table 4. 1: Root mean squared error of three monocular depth estimation**

| model | Model 1 | Model 2 | Model 3 |
|---|---|---|---|
| Root-mean-squared-error | 6.23 | 4.58 | 1.79 |

### 4.5.2 Estimation of car following distance

The estimation of the car-following distance was a multi-step approach as described in the methodology. First, we estimated the scene depth using monocular depth estimation (Model 3). **Figure 4.7** below shows the input image and output image after the input image was passed through the monocular depth estimation model to estimate the scene depth.

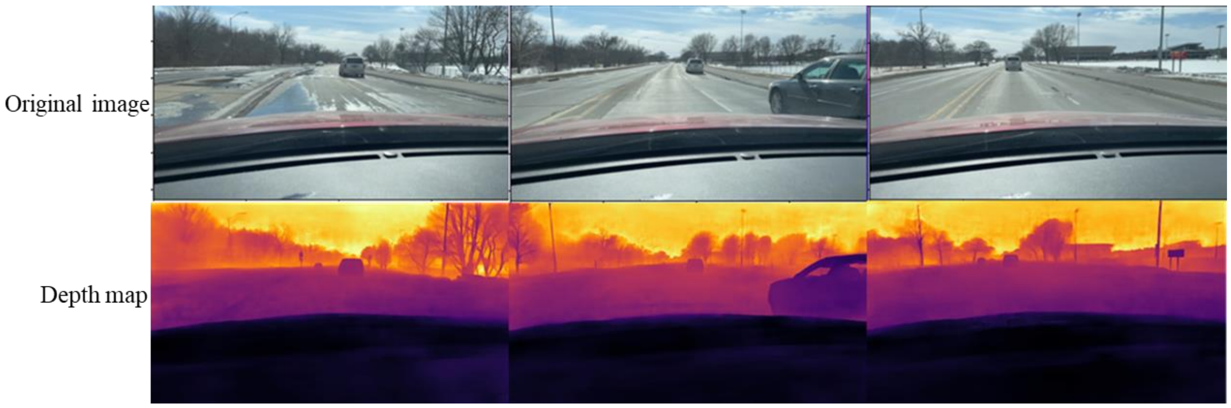


**Figure 4. 7**: **Input image (row 1) and output image (row 2) after the input image was passed through the monocular depth estimation model to estimate the depth map**

Second, we detected relevant objects in the images by using a single-stage object detection model YOLOv5. The results of the detection stage are shown in **Figure 4.8** below. The objects identified in this stage were cars, trucks, traffic signs and traffic signals.

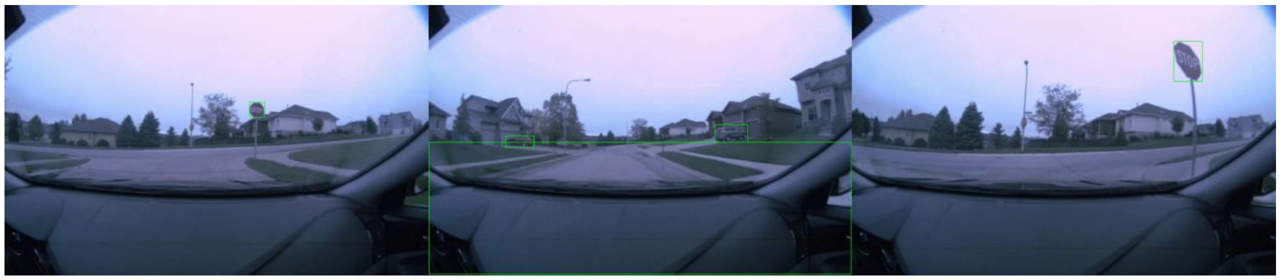


**Figure 4. 8**: **Object detection using YOLOv5**

Finally, we combined the first and second results with a heuristics algorithm to determine if a detected car or truck is in front of the ego-vehicle or not. The results of that are shown in **Figure 4.9** below.

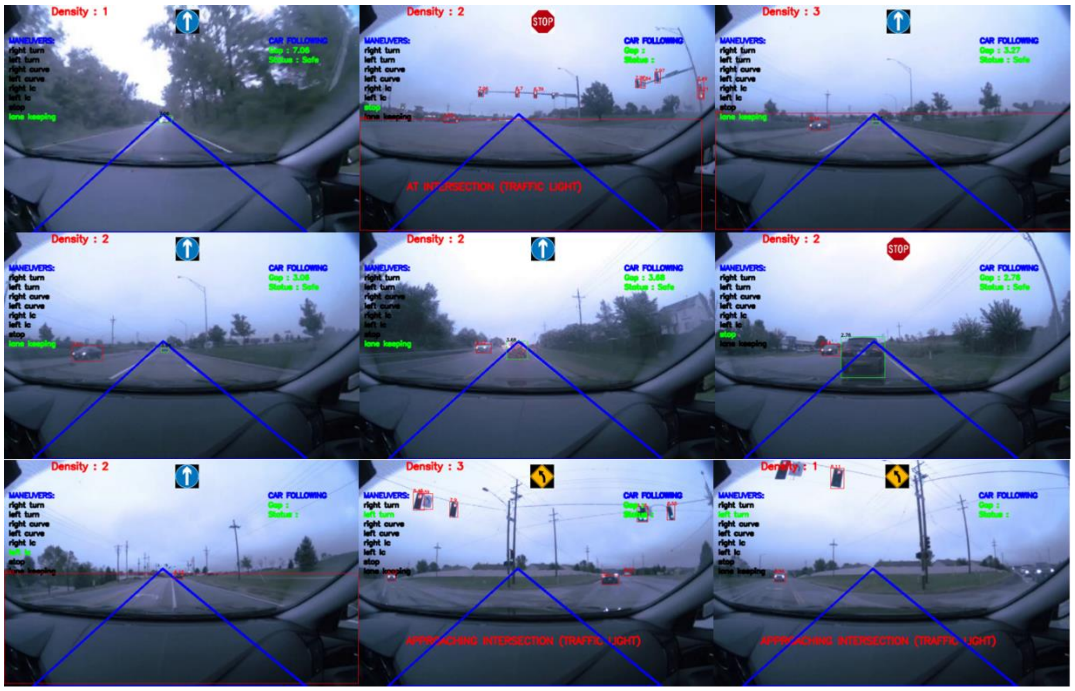


**Figure 4. 9: A framework for detecting leading vehicles, safe car-following distance and driver maneuvers.**

### 4.5.3 Comparison of Car Following Distances of Different Driving Groups

As shown in the probability density plot in **Figure 4.10**, we compared the car following distance of the four participants studied in this project. On average, the car following distance can range from greater than zero to less than ten meters, with the majority of numbers falling somewhere in the middle. Additionally, it was discovered that the elderly drivers (man and woman) maintained greater car following distances than the two younger drivers. Even though the data used for this analysis does not represent a substantial proportion of each driving group, we can still conclude from the results that older drivers are more safety-conscious than younger drivers. The authors are also cognizant of the fact that this conclusion may or may not hold true if large samples of each driving group are analyzed further.

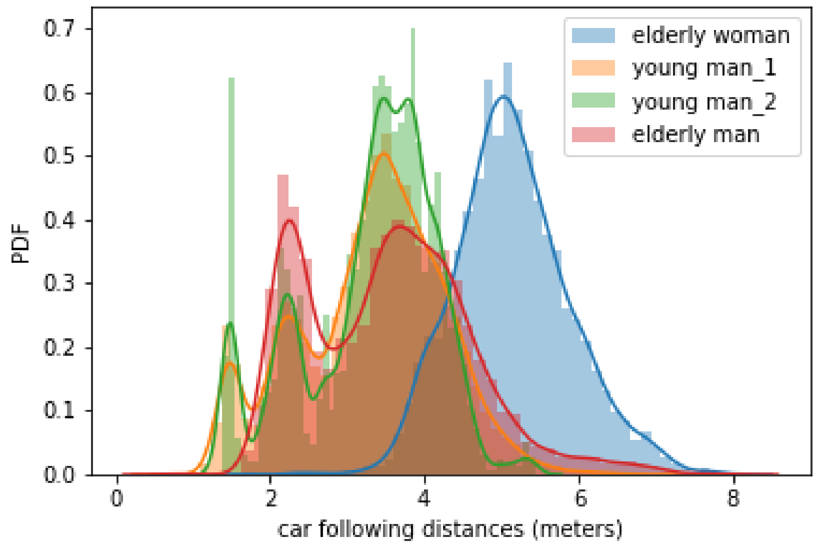


**Figure 4. 10: A PDF of car-following distances of different individuals**

To further investigate the observations made from **Figure 4.10**, statistical t-tests were performed on pairs of driving groups to determine if there were significant differences in the mean car following distance between the various driving groups. This test is necessary in order to determine if different demographic groups exhibit the same car-following behavior. We assume an equal variance in our analysis if the ratio of variances between any two driving groups is 4:1. The fundamental assumption (null hypothesis) for test is that the mean car following distance of different driving groups are the same. A significant level ($\alpha$) 0.05 is used to infer if there is enough evidence to reject the null hypothesis. Mathematically we formulated the test as,

Let,

$$\mu_i = the\ mean\ car\ following\ distance\ for\ group\ i$$

$$\alpha = significant\ level$$

then,

$$\text{Null Hypothesis: } \mathrm{H}_0 \rightarrow\ \mu_1 =\ \mu_2$$

$$\text{Alternate Hypothesis: } \mathrm{H}_a \rightarrow\ \mu_1 \neq\ \mu_2$$

If the computed p-value < $\alpha$, then we have enough evidence to reject the null hypothesis, else we do not reject the null hypothesis.

With the exception of the comparison between the two young drivers, which yielded a p-value of 0.84, all other comparisons yielded a p-value of 0.00, as shown in Table 4.2. Given that the p-value for comparing the two young drivers was 0.84, we can conclude that there was insufficient evidence to reject the null hypothesis; thus, the two young drivers' mean car following distance is identical. Also, the results indicate that the elderly female driver and the elderly male driver have different car following behaviors. In addition, by comparing the car following distances of young and elderly drivers, we discovered that the two groups were also different.

**Table 4. 2: Statiscal T-test Anslysis of different individuals**

| comparison | P-value | Decision |
|---|---|---|
| Elderly woman vs Elderly man | 0.00 | Reject the null hypothesis |
| Elderly woman vs Young man 1 | 0.00 | Reject the null hypothesis |
| Elderly woman vs Young man 2 | 0.00 | Reject the null hypothesis |
| Elderly man vs Young man 1 | 0.00 | Reject the null hypothesis |
| Elderly man vs Young man 2 | 0.00 | Reject the null hypothesis |
| Young man 1 vs Young man 2 | 0.84 | Do not reject the null hypothesis |
| Young vs Elderly | 0.00 | Reject the null hypothesis |

### 4.5.4 Comparing the Acceleration-Deceleration Behavior of Young Drivers and Elderly Drivers

As indicated by our earlier analysis, there is a significant disparity between the car-following distances of young and elderly drivers. In this section we further investigate the aggressiveness of each driving group. **Figure 4.11** below presents the relative velocity–acceleration mapping obtained from the two driving groups (young drivers and elderly drivers). The two driving groups have similar data length (young drivers, 1800sec; elderly drivers,2000sec). The deceleration of young drivers (**Figure 4.11a**) is greater than that of elderly drivers (**Figure 4.11b**) when their vehicle approaches the leading vehicle. This hard deceleration could be used to infer aggressive driving of young drivers. When the distance between vehicles is opening, the acceleration of young drivers is greater than that of elderly drivers. Thus, the acceleration-deceleration rate of young drivers indicates a tendency opposite that of elderly drivers.

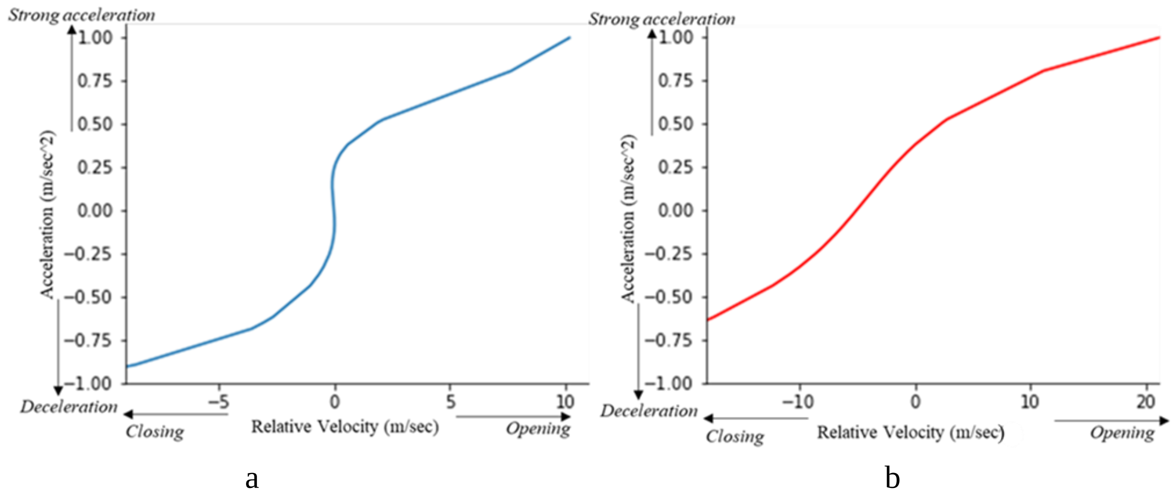


**Figure 4. 11: Acceleration against relative velocity of two driving groups a) Young drivers b) Elderly drivers**

### 4.5.5 Modeling the Acceleration of the Ego-Vehicle

Modeling the acceleration of the ego-vehicle enables us to comprehend how the ego vehicle accelerates and decelerates depending on its distance from the leading vehicle. In **Figure 4.12**, we show a general relationship between acceleration of car-following distance of the ego-vehicle. Generally, deceleration occurs when the ego-vehicle is close to the leading vehicle whereas acceleration occurs when the distance between the ego-vehicle and the leading vehicle is greater.

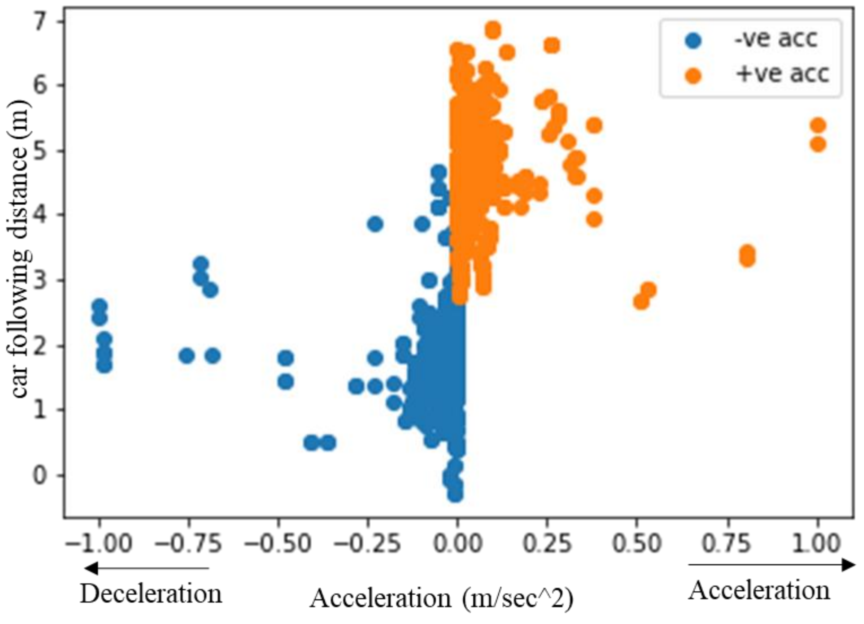


**Figure 4. 12: A general relationship between acceleration of car-following distance of the ego-vehicle**

To develop the acceleration model for the ego-vehicle, an Extreme Gradient Boosting (XGBoost) algorithm was used. XGBoost is a variant of gradient boosting machines that predicts errors using Gradient Boosting Trees (gbtree). It begins with a simple predictor that predicts an arbitrary number (typically 0.5) irrespective of the input. Typically, the model begins with a high error rate and gradually learns to reduce it. Training an error prediction model in XGBoost does not involve optimizing the predictor on (feature, error) pairs. The data used to train the model was divided into training (5834 data points) and test (2,500 data points) samples in a ratio of 0.7:0.3.

From the results of the model building, relative velocity clearly stands out as the most important predictor of the ego-vehicle acceleration compared to the car following distance as shown in the **Figure 4.13**.

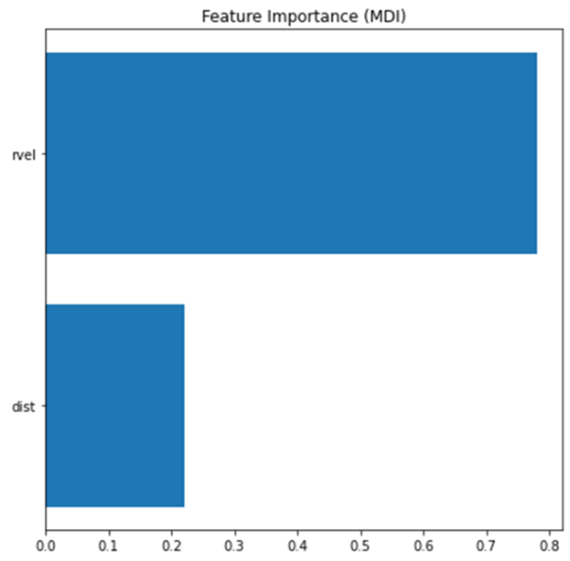


**Figure 4. 13: A plot of feature importance from the developed model**

Subsequently, the developed model was used to predict the acceleration of the ego-vehicle, as shown in **Figure 4.14**. The model was able to detect the general trend of the acceleration of the ego-vehicle as well as the peaks in the trend. The performance of the model was evaluated using the root mean square error (RMSE). The RMSE of the model was 0.0245. A model with RMSE close to zero is commonly regarded as a good model, and since our developed model is in the hundredth decimal, it can be considered to be performing well.

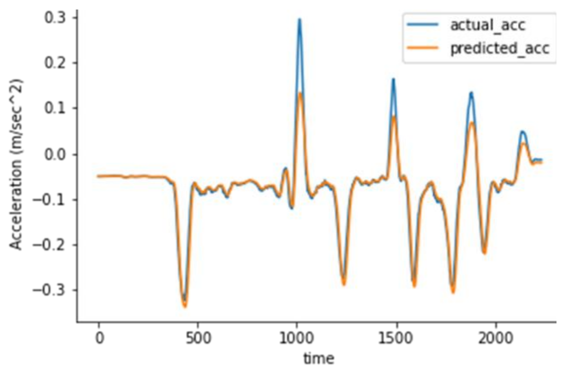


**Figure 4. 14: A plot of actual acceleration versus predicted acceleration of the ego-vehicle**

### 4.5.6 Modeling the Acceleration of the Leading Vehicle

The acceleration of the leading vehicle was modeled similar to the acceleration of the ego-vehicle using XGBoost. The difference here is that the acceleration of the ego-vehicle was an input variable to this model in addition to car-following distance and relative velocity. Similar to the model developed above, the data used to train the model was divided into training (5834 data points) and test (2,500 data points) samples in a ratio of 0.7:0.3. The results of the model showing the feature importance are shown in the **Figure 4.15**. The car-following distance and relative velocity were found to be the most important features to predict the acceleration of the leading vehicle. The acceleration of ego-vehicle was found not be influential in predicting the acceleration of the leading vehicle.

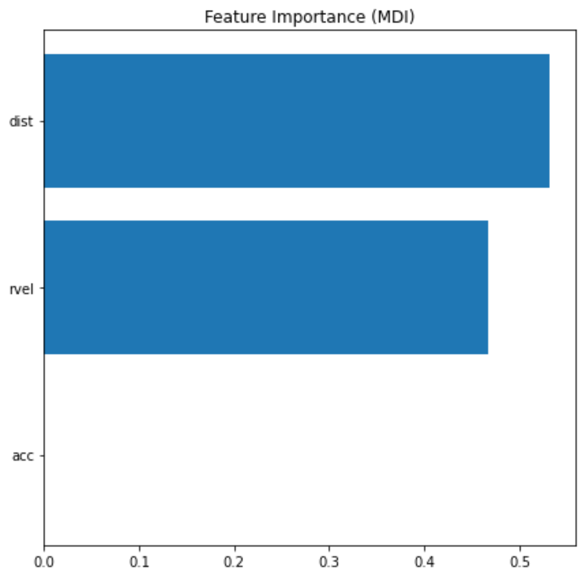


**Figure 4. 15: A plot of feature importance from the developed model**

As shown in **Figure 4.16**, the developed model was used to predict the acceleration of the leading vehicle. The model's generalization ability has been demonstrated. The model was able to detect the general trend of the leading vehicle's acceleration as well as the peaks in the trend. The performance of the model was evaluated using the root mean square error (RMSE). The RMSE of the model was 0.0105. A model with RMSE close to zero is commonly regarded as a good model, and since our developed model is in the hundredth decimal, it can be considered to be performing well.

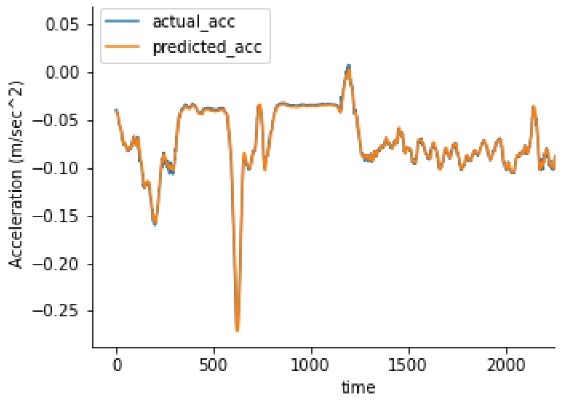


**Figure 4. 16: A plot of actual acceleration versus predicted acceleration of the leading-vehicle**

## 4.6 Conclusion

Rear-end crashes are the most common type of accidents encountered on the highway. These crashes have a significant negative impact on the flow of traffic and typically result in serious repercussions. To gain a better understanding of these scenarios from a practical standpoint, it is necessary to accurately model car-following behaviors that lead to rear-ends crashes. Numerous studies have been conducted to model the car-following behaviors of drivers; however, the majority of these studies have relied on simulated data that may not accurately represent real-world incidents. Also, most studies are limited to modeling the acceleration of the ego-vehicle which is insufficient to explain the behavior of the ego-vehicle.

As a result, the current study attempts to address these issues by developing a framework for extracting features pertinent to comprehending the behavior of drivers in naturalistic environment. Moreover, the study modeled the acceleration of both the ego-vehicle and the leading vehicle through the development of an AI framework capable of extracting the parameters from NDS videos required to model the behavior (acceleration) of both vehicles. Additionally, the study performed longitudinal studies of the car-following behavior of different demographics in a naturalistic environment.

The result from the analysis shows that young individuals are more likely to be aggressive drivers compared to elderly drivers. Also, in modelling the acceleration of the ego-vehicle, relative velocity between the ego-vehicle and the leading vehicle was found to be more important than the distance between the two vehicles (car-following distance). This means that the accelerations of drivers in ego vehicles are affected more by the relative velocity between them and the leading vehicle. The opposite occurred when modelling the acceleration of the leading vehicle.

## 4.7 Limitations and recommendation

The current study has the following limitations.

1. The heuristic method utilized by the study to track the leading vehicle is not always effective at intersections or when the driver is negotiating a curve. Future research should track the leading vehicle using an automated algorithm.
2. Developing our own model with the lidar data would have improved the accuracy of the predicted car-following distances. Despite the fact that the monocular depth estimation model gave us a comparable result. Future research should develop its own model for predicting car-following distance utilizing Lidar data.

## 4.8 References

Z. Zheng, M. Sarvi, Modeling, calibrating, and validating car following and lane changing behavior. Transportation Research Part C: Emerging Technologies 71, 182–183 (2016)

R. Hoogendoorn et al., Mental workload, longitudinal driving behavior, and adequacy of car-following models for incidents in other driving lane. Transportation Research Record 2188(1), 64–73 (2010) https://doi.org/10.3141/2188-08

A. Tang, A. Yip, Collision avoidance timing analysis of DSRC-based vehicles. Accident Analysis & Prevention 42(1), 182–195 (2010) https://doi.org/10.1016/j.aap.2009.07.019

Kometani, E. “Dynamic behavior of traffic with a nonlinear spacing-speed relationship.” Theory of Traffic Flow (Proc. of Sym. on TTF (GM)) (1959): 105-119.

R.E. Chandler, R. Herman, E.W. Montroll, Traffic dynamics: studies in car following. Operations research 6(2), 165–184 (1958) https://doi.org/10.1287/opre.6.2.165

J. Treiterer, J. Myers, The hysteresis phenomenon in traffic flow. Transportation and traffic theory 6, 13–38 (1974)

Aron, Maurice. “Car following in an urban network: simulation and experiment. ” Proceedings of Seminar D, 16^< th> PTRC Meeting, 1988. 1988.

W. Helly, Simulation of bottlenecks in single-lane traffic flow (1959)

Hidas, Peter. “A car-following model for urban traffic simulation.” Traffic engineering & control 39.5 (1998).

R. Jiang, M.B. Hu, H.M. Zhang, et al., On some experimental features of carfollowing behavior and how to model them. Transportation Research Part B: Methodological 80, 338–354 (2015) https://doi.org/10.1016/j.trb.2015.08.003

Tang T Q, Zhang J, Chen L. "Analysis of vehicle’s safety envelope under carfollowing model." Physica A: Statistical Mechanics and its Applications, 2017,474, 127-133. https://doi.org/10.1016/j.physa.2017.01.076

P.G. Gipps, A behavioural car-following model for computer simulation. Transportation Research Part B: Methodological 15(2), 105–111 (1981) https://doi.org/10.1016/0191-2615(81)90037-0

Zhou T, Sun D, Kang Y, et al. "A new car-following model with consideration of the prevision driving behavior." Communications in Nonlinear Science and Numerical Simulation ,2014,19(10): 3820-3826. https://doi.org/10.1016/j.cnsns.2014.03.012

Y. Yang, K. Wada, T. Oguchi, et al., Variability of observed drivers’ carfollowing behavior on expressway basic segment. Transportation Research Procedia 25, 1503–1532 (2017) https://doi.org/10.1016/j.trpro.2017.05.179

A. Tordeux, S. Lassarre, M. Roussignol, An adaptive time gap car-following model. Transportation Research Part B Methodological 44(8–9), 1115–1131 (2010) https://doi.org/10.1016/j.trb.2009.12.018

T.Q. Tang, W.F. Shi, H.Y. Shang, An extended car-following model with consideration of the reliability of inter-vehicle communication. Measurement 58(11), 286–293 (2014) https://doi.org/10.1016/j.measurement.2014.08.051

Tosi, Fabio, Filippo Aleotti, Matteo Poggi, and Stefano Mattoccia. "Learning monocular depth estimation infusing traditional stereo knowledge." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9799-9809. 2019.

H. Laga, L. V. Jospin, F. Boussaid and M. Bennamoun, "A Survey on Deep Learning Techniques for Stereo-Based Depth Estimation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 44, no. 4, pp. 1738-1764, 1 April 2022, doi: 10.1109/TPAMI.2020.3032602.

Li, Zhaoshuo, Xingtong Liu, Nathan Drenkow, Andy Ding, Francis X. Creighton, Russell H.Taylor, and Mathias Unberath. "Revisiting stereo depth estimation from a sequence-tosequence perspective with transformers." In Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 6197-6206. 2021.

David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Proc. NeurIPS, 2014.

Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. T2net: Synthetic-to-realistic translation for solving single-image depth estimation tasks. In Proc. ECCV, 2018.

Wei Yin, Yifan Liu, Chunhua Shen, and Youliang Yan. Enforcing geometric constraints of virtual normal for depth prediction. In Proceedings of the IEEE International Conference on Computer Vision, pages 5684–5693, 2019.

Rene Ranftl, Katrin Lasinger, David Hafner, Konrad ´ Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. IEEE Trans. Pattern Anal. Mach. Intell., 2020.

Lijun Wang, Jianming Zhang, Yifan Wang, Huchuan Lu, and Xiang Ruan. CLIFFNet for monocular depth estimation with hierarchical embedding loss. In Proc. ECCV, 2020

Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In Proc. ECCV, 2012.

Neal Wadhwa, Rahul Garg, David E Jacobs, Bryan E Feldman, Nori Kanazawa, Robert Carroll, Yair MovshovitzAttias, Jonathan T Barron, Yael Pritch, and Marc Levoy. Synthetic depthoffield with a single-camera mobile phone. ACM Trans. Graph., 37(4):1–13, 2018.

Lijun Wang, Xiaohui Shen, Jianming Zhang, Oliver Wang, Zhe Lin, Chih-Yao Hsieh, Sarah Kong, and Huchuan Lu. DeepLens: Shallow depth of field from a single image. ACM Trans. Graph., 37(6), 2018.

I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. 2016 Fourth International Conference on 3D Vision (3DV), pages 239–248, 2016.

D. Xu, E. Ricci, W. Ouyang, X. Wang, and N. Sebe. Multiscale continuous crfs as sequential deep networks for monocular depth estimation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 5354–5362, 2017.

Detail preserving depth estimation from a single image using attention guided networks. 2018 International Conference on 3D Vision (3DV), pages 304–313, 2018.

D. Xu, W. Wang, H. Tang, H. W. Liu, N. Sebe, and E. Ricci. Structured attention guided convolutional neural fields for monocular depth estimation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 3917– 3925, 2018.

H. Fu, M. Gong, C. Wang, N. Batmanghelich, and D. Tao. Deep ordinal regression network for monocular depth estimation. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2002–2011, 2018

Kunal Swami, Prasanna Vishnu Bondada, and Pankaj Kumar Bajpai. Aced: Accurate and edge-consistent monocular depth estimation. In 2020 IEEE International Conference on Image Processing (ICIP), pages 1376–1380. IEEE, 2020.

Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers.

Rene Ranftl, Alexey Bochkovskiy, and ´ Vladlen Koltun. Vision transformers for dense prediction. arXiv preprint arXiv:2103.13413, 2021.

B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. Demon: Depth and motion network for learning monocular stereo. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 5622–5631, 2017.

H. Zhou, B. Ummenhofer, and T. Brox. Deeptam: Deep tracking and mapping. In Proceedings of the European Conference on Computer Vision (ECCV), pages 822–838, 2018.

Alhashim, Ibraheem, and Peter Wonka. "High quality monocular depth estimation via transfer learning." arXiv preprint arXiv:1812.11941 (2018).

Kim, Doyeon, Woonghyun Ga, Pyungwhan Ahn, Donggyu Joo, Sehwan Chun, and Junmo Kim. "Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth." arXiv preprint arXiv:2201.07436 (2022).

Miangoleh, S. Mahdi H., Sebastian Dille, Long Mai, Sylvain Paris, and Yagiz Aksoy. "Boosting monocular depth estimation models to high-resolution via content-adaptive multi-resolution merging." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9685-9694. 2021.

# CHAPTER 5: CONCLUSION

Our ability to detect and characterize unsafe driving behaviors in naturalistic driving settings and associate them with road crashes will be a major step toward developing effective crash countermeasures. Researchers to date have not fully achieved the state goal of being able to characterize unsafe driving behaviors due to certain limitations. These limitations include but are not limited to, the high cost of data collection and the manual processes required to extract information from NDS data.

The cost of collecting naturalistic driving studies (NDS) data is prohibitively high, precluding the continuous collection and analysis of data related to driver behaviors. To provide a long-term solution to the high cost of data collection, there is a need for data-driven approaches that can collect and process quality, high-resolution, and high-fidelity streams of data at a reasonable cost. The current approach to transportation data collection relies on expensive systems that are not easily scalable for continuous data streaming. The cost-prohibitive nature of this effort could lead to infrequent driver behavior assessment for agencies with limited funding. Low-cost data collection alternatives that are able to produce information comparable to these high-end systems are needed. Moreover, the process of driving event extraction from NDS data has been performed by a mixture of manual and semi-automated processes. The size of these datasets typically ranges between hundreds of terabytes to several petabytes depending on video compression rates. Therefore, relying on manual processing methods can be labor-intensive and awfully expensive to scale. There is a need to develop algorithms that can ingest multi-modal NDS data and accurately annotate different driving events useful for understanding crash causality. Additionally, many rear-end near-crashes have been attributed to drivers' car-following behavior. Only a few studies have investigated car-following behavior using NDS data. As such it is necessary for more investigations into cars following behaviors of drivers in a naturalistic driving environment.

The main goal of the study is to develop an AI framework for automatically extracting high-low features from the NDS dataset in order to explain driver behavior while using a low-cost data collection method. The author proposed three novel objectives for achieving the study's goal based on the identified research gaps. First, the study builds a low-cost data acquisition system for acquiring naturalistic driving data. Second, the study designs a framework that automatically extracts high to low-level features such as vehicle density, turning movements, lane changes, and other relevant features from the data collected from the developed data acquisition system. Third, the study extracted information from the NDS data to gain a better understanding of people's car-following behavior, and other driving behaviors through data collection and analysis in order to develop countermeasures for traffic safety.

To achieve the first objective, the study develops a multipurpose smartphone application for collecting NDS data. The designed app was made up of three major modules: a front-end user interface module, a sensor module, and a backend module. The front-end, which is also the user interface of the app, was designed to provide a streamlined view that exposed key aspects of the application via a tab bar controller. This enables us to compartmentalize the application's various critical components into distinct views. The backend module provides computational resources that can be used to accelerate front-end user query responses. The backend of the developed app was powered by Google Firebase. The sensor modules included CoreMotion, CoreLocation, and AVKit. The CoreMotion module collects motion and environmental data from the onboard hardware of iOS devices, including accelerometers and gyroscopes, as well as the pedometer, magnetometer, and barometer. Whereas, the CoreLocation determines the altitude, orientation, and geographical location of a device, as well as its position in relation to a nearby iBeacon device. Finally, the AVKit provides a high-level interface for video content playback. The developed app was used to collect data on both freeways and arterials. The app utilizes both the front and rear cameras of the smartphone to collect video data of the driver's activity in the vehicle and the vehicle's surroundings during data collection. The data gathered by the application was then used to extract driving events from videos, conduct pavement distress detection analysis and predict IRI values of road surfaces. The results of the road roughness analysis indicate that the accuracy of estimating IRI values from smartphone data is comparable to that of high-end machines. Consequently, the developed smartphone application is a suitable substitute for the high-end machine, allowing the State Department of Transportation to save money on data collection. In addition, the high accuracy obtained from the pavement distress detection model will increase confidence in the model's dependability in providing pavement engineers with current information about road conditions.

To accomplish objective two, we formulated our problem as both a classification and time-series segmentation problem. This is because most existing methods for driver maneuver detection formulate the problem purely as a classification problem, assuming a discretized input signal with known start and end locations for each event or segment. However, in practice, vehicle telemetry data used for detecting driver maneuvers are continuous, therefore, a fully automated driver maneuver detection system should implement solutions for both time series segmentation and classification. Our proposed methodology can be divided into five stages: 1) preprocessing of data, 2) event segmentation, 3) machine learning classification, 4) heuristics classification, and 5) frame-by-frame annotation of video. To begin, the input data is standardized and smoothed. The resulting output is segmented and then classified using both machine learning (main driving events) and heuristics (stops and lane-keeping). The study separated the detection of stops and lane-keeping from the rest of the driving events because the two can be easily identified using simple thresholding and to reduce false negatives when using only ML to classify all driving events. The result from the study indicates that the gyroscope reading is an exceptionally good parameter to be used to extract driving events since it showed consistent accuracy across all four developed models. The study shows that the accuracy of the Energy Maximization Algorithm ranges from 56.80% (left lane change) to 85.20% (lane-keeping) All four models developed had comparable accuracies to studies that used similar models (Bakhit et al. 2017; Kumar et al. 2013; Mandalia and Salvucci 2005; Zheng et al. 2014). The 1D-CNN model had the highest accuracy of 98.99%, followed by the LSTM model at 97.75%, the RF model at 97.71%, and the SVM model at 97.65%. To serve as a ground truth, continuous signal data was annotated. Also, the proposed methodology outperformed the fixed time window approach when compared. The study further analyzed the accuracy of the overall pipeline by penalizing the F1 scores of the ML models with the duration score of the EMA. The overall accuracy of the pipeline was in the range of 56.8% to 85.2%. To test the model's transferability, the developed models were used to detect driving events from multiple streams of datasets. The F1 scores were high and consistent across all three datasets used. The predicted results were compared to the ground truth annotations. Using the LSTM model, the test was 91% accurate. Based on the results of this study, conducting the study on a large scale, will provide insight and the extraction of critical information regarding drivers' lane-changing behaviors (which have recently caused the majority of vehicle crashes on the highway) and turning maneuvers, as well as aggressive driving behaviors, in order to improve traffic safety. This study's underlying structure is both rapid and extensible. As a result, the framework developed in this study will facilitate the annotation of a large number of NDS videos into diverse driving events. This will allow for more rapid analysis of collisions in conflict zones, particularly at intersections (i.e., the extraction of right and left turning events).

This study's final objective was to extract variables from naturalistic driving videos that will aid in the comprehension of driver behavior in a naturalistic driving environment. In order to reach this objective, three sub-goals were developed. First, we developed a framework for extracting features relevant to understanding the behavior of drivers in natural environments. Second, we analyzed the car-following behaviors of various demographic groups based on the extracted features. Thirdly, using a machine learning algorithm, we modeled the acceleration of both the ego-vehicle and the leading vehicle. According to the study's findings, young drivers are more likely to be aggressive than older drivers. In addition, the study revealed that drivers tend to speed up when the gap between them and the leading vehicle is large. Finally, compared to younger drivers, elderly drivers keep a much larger car-following distance. The findings of this study have numerous safety implications. First, the analysis of the driving behavior of different demographic groups will improve safety engineers' understanding of the driving styles of different demographic groups and the causes of collisions, enabling them to develop the most effective crash countermeasures. Second, the models developed for predicting the acceleration of both the ego-vehicle and the leading vehicle will provide sufficient information to explain the ego-driver's behavior.

## 5.1 Limitations of the Study

No study is devoid of limitations. Several limitations were encountered while developing the various frameworks proposed in this study. For each objective, we highlight the various limitations encountered.

The limitations encountered during the development of smartphone data acquisition includes

1. The GPS coordinates of the developed smartphone app are slightly off. The next update of the app should extend versions of newer smartphone devices with more accurate GPS coordinate values.
2. Also, the developed app was designed exclusively for the iOS operating system. Given the substantial number of Android users, the app's next update should include support for the Android operating system.
3. In the analysis of the International Roughness Index (IRI), we assumed that the observed spikes in vertical accelerations were caused solely by pavement distress. This assumption is not always accurate, as other driving activities can also result in such a significant huge spike. Future research should include multimodal analysis of video images and sensor readings. This will greatly lead to a reduction in false positives during the analysis.
4. The current study utilized the TabNet model to develop the IRI model. Even though the study's results were favorable, the model did not make use of the temporary information

present in the data. Using more sophisticated sequential modeling architecture in the future could help improve the results accuracy.

Similar to objective one, objective two was poised with some limitations which are highlighted below.

1. The energy maximization algorithm assumes that all enormous peaks of greater magnitude were due to driving events. This is not always the case, as some drivers' activities, such as eating, dancing, or shaving, may also result in significant amount spikes comparable to driving events. Future research should consider analyzing both video data and gyroscope data simultaneously. This will help reduce false positives by identifying anomalous spikes caused by the driver's actions.
2. The approach developed in this study cannot be used in the real-time detection of vehicle maneuvers. For real-time applications, future studies should consider detecting driver maneuvers from image frames directly.

Finally, the limitations of the third objectives are as follows;

1. The heuristic method utilized by the study to track the leading vehicle is not always effective at intersections or when the driver is negotiating a curve. Future research should track the leading vehicle using an automated algorithm.

2. Developing our own model with the lidar data would have improved the accuracy of the predicted car-following distances. Despite the fact that the monocular depth estimation model gave us a comparable result. Future research should develop its own model for predicting car-following distance utilizing Lidar data.

# VITA

Armstrong Aboah received B.S. degree in civil engineering from Kwame Nkrumah University of Science and Technology, Ghana, and the M.S. degree in transportation engineering from Tennessee Technological University, USA. He entered the Graduate School at University of Missouri Columbia in January 2020 to purse the degree of Doctor of Philosophy in Civil Engineering. During this time as a graduate student as well as a research assistant, Armstrong was involved with several funded research projects by NSF and MoDOT. He received several awards for his work in student competitions, conferences, and journal publications.