

Electronic Theses and Dissertations, 2020-

2021

Real-Time Traffic Safety Evaluation in the Context of Connected Vehicles and Mobile Sensing

Pei Li

University of Central Florida

 Part of the [Civil Engineering Commons](#), and the [Transportation Engineering Commons](#)
Find similar works at: <https://stars.library.ucf.edu/etd2020>
University of Central Florida Libraries <http://library.ucf.edu>

This Doctoral Dissertation (Open Access) is brought to you for free and open access by STARS. It has been accepted for inclusion in Electronic Theses and Dissertations, 2020- by an authorized administrator of STARS. For more information, please contact STARS@ucf.edu.

STARS Citation

Li, Pei, "Real-Time Traffic Safety Evaluation in the Context of Connected Vehicles and Mobile Sensing" (2021). *Electronic Theses and Dissertations, 2020-*. 1144.
<https://stars.library.ucf.edu/etd2020/1144>

**REAL-TIME TRAFFIC SAFETY EVALUATION IN THE CONTEXT OF
CONNECTED VEHICLES AND MOBILE SENSING**

by

PEI LI

B.Sc. Tongji University, 2015

M.Sc. Tongji University, 2018

M.Sc. University of Central Florida, 2020

A thesis submitted in partial fulfillment of the requirements
for degree of Doctor of Philosophy
in the Department of Civil, Environmental and Construction Engineering
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Summer Term

2021

Major Professor: Mohamed Abdel-Aty

ABSTRACT

Recently, with the development of connected vehicles and mobile sensing technologies, vehicle-based data become much easier to obtain. However, only few studies have investigated the application of this kind of novel data to real-time traffic safety evaluation. This dissertation aims to conduct a series of real-time traffic safety studies by integrating all kinds of available vehicle-based data sources. First, this dissertation developed a deep learning model for identifying vehicle maneuvers using data from smartphone sensors (i.e., accelerometer and gyroscope). The proposed model was robust and suitable for real-time application as it required less processing of smartphone sensor data compared with the existing studies. Besides, a semi-supervised learning algorithm was proposed to make use of the massive unlabeled sensor data. The proposed algorithm could alleviate the cost of data preparation and improve model transferability. Second, trajectory data from 300 buses were used to develop a real-time crash likelihood prediction model for urban arterials. Results from extensive experiments illustrated the feasibility of using novel vehicle trajectory data to predict real-time crash likelihood. Moreover, to improve the model's performance, data fusion techniques were proposed to integrated trajectory data from various vehicle types. The proposed data fusion techniques significantly improved the accuracy of crash likelihood prediction in terms of sensitivity and false alarm rate. Third, to improve pedestrian and bicycle safety, different vehicle-based surrogate safety measures, such as hard acceleration, hard deceleration, and long stop, were proposed for evaluating pedestrian and bicycle safety using vehicle trajectory data. In summary, the results from this dissertation can be further applied to real-time safety applications (e.g., real-time crash likelihood prediction and visualization system) in the context of proactive traffic management.

ACKNOWLEDGMENT

I would like to convey my heartiest gratitude to my honorable supervisor Dr. Mohamed Abdel-Aty for his excellent supervision and constant support in this dissertation. I still remember when I first met him at Shanghai in 2017. I am very lucky to have the chance to work with an amazing advisor who always gives me full support and guidance. I would also like to extend my sincere thanks to the committee members, Dr. Naveen Eluru, Dr. Mohamed H. Zaki, and Dr. Xin Yan for their valuable suggestions. I would also like to acknowledge the help from my colleagues and friends. In the end, I want to thank my family for always supporting me.

TABLE OF CONTENTS

LIST OF FIGURES	vi
LIST OF TABLES.....	viii
LIST OF ACRONYMS/ABBREVIATIONS	ix
CHAPTER 1: INTRODUCTION.....	1
1.1 Background	1
1.2 Research Objectives.....	4
1.3 Thesis Structure	7
CHAPTER 2: LITERATURE REVIEW	10
2.1 Data Sources for Traffic Safety Analysis.....	10
2.1.1 Infrastructure-based Data.....	10
2.1.2 Vehicle-based Data.....	12
2.2 Real-time Traffic Safety Analysis	15
2.3 Methodologies	16
2.3.1 Statistical Methods	17
2.3.2 Machine Learning Methods	17
CHAPTER 3: A DEEP LEARNING APPROACH TO DETECT REAL-TIME VEHICLE MANEUVERS BASED ON SMARTPHONE SENSORS	20
3.1 Introduction	20
3.2 Sensor Description.....	24
3.3 System Overview.....	26
3.4 Experiment and Results.....	33
3.5 Conclusions	45
CHAPTER 4: DRIVING MANEUVERS DETECTION USING SEMI-SUPERVISED LSTM AND SMARTPHONE SENSORS	47
4.1 Introduction	47
4.2 Research Approach	54
4.3 Experiment and Results.....	61
4.4 Conclusions	70
CHAPTER 5: THE APPLICATION OF NOVEL CONNECTED VEHICLE EMULATED DATA ON REAL-TIME CRASH POTENTIAL PREDICTION FOR ARTERIALS.....	72
5.1 Introduction	72
5.2 Data Preparation	76
5.2.1 Data Description.....	76
5.2.2 Data Preprocessing	78
5.2.3 Crash Labelling and Features Selection.....	82
5.3 Methodologies	84
5.4 Experimental Design and Results	86
5.4.1 Experimental Design	86
5.4.2 Experimental Results.....	89
5.4.3 Model Comparison.....	90
5.5 Conclusions	91
CHAPTER 6: USING BUS CRITICAL DRIVING EVENTS AS SURROGATE SAFETY MEASURES FOR PEDESTRIAN AND BICYCLE CRASHES BASED ON GPS TRAJECTORY DATA.....	94
6.1 Introduction	94
6.2 Research Approach	98

6.2.1	Trajectory Data Preparation	99
6.2.2	Map Matching	102
6.2.3	Correlation Estimation.....	102
6.2.4	Crash Frequency Modeling.....	103
6.3	Data Description	106
6.4	Experimental Results	108
6.5	Conclusions	115
CHAPTER 7: TRAJECTORY FUSION-BASED REAL-TIME CRASH LIKELIHOOD PREDICTION USING LSTM-CNN WITH ATTENTION MECHANISM.....		117
7.1	Introduction	117
7.2	Data Preparation	120
7.2.1	Data Description.....	120
7.2.2	Trajectory Data Preparation	122
7.2.3	Trajectory Data Fusion	125
7.2.4	Crash Data Preparation	126
7.3	Research Methodologies	127
7.3.1	LSTM with Attention Mechanism.....	127
7.3.2	CNN.....	129
7.3.3	TA-LSTM-CNN.....	130
7.4	Experiment and Results.....	131
7.4.1	Site Selection and Data Preparation	131
7.4.2	Experimental Design	133
7.4.3	Results	135
7.4.4	Model Comparison.....	136
7.5	Conclusions	138
CHAPTER 8: CONCLUSIONS.....		140
REFERENCES.....		144

LIST OF FIGURES

Figure 1 Sensor coordinate system and readings	25
Figure 2 System workflow.....	27
Figure 3 Coordinate system reorientation illustration	29
Figure 4 LSTM structure (Christopher, 2015)	32
Figure 5 Smartphone application interface	34
Figure 6 Model architecture.....	35
Figure 7 Model implementation process	37
Figure 8 Normalized confusion matrix.....	39
Figure 9 t-SNEs of raw features and extracted features	40
Figure 10 Results from different models	42
Figure 11 Processing time of different models	42
Figure 12 Model results on the new dataset.....	43
Figure 13 Model results on different new data ratios	44
Figure 14 Research workflow	54
Figure 15 Application interface.....	55
Figure 16 Sensor coordinate systems (Apple, 2020).....	56
Figure 17 Coordinate system alignment	57
Figure 18 Semi-supervised learning algorithm	59
Figure 19 LSTM cell structure (Sanjeevi, 2018).....	60
Figure 20 Network architecture.....	61
Figure 21 Data collection routes and intersections' locations.....	62
Figure 22 Boxplots of model comparison.....	69
Figure 23 Research area.....	78
Figure 24 Illustration of segments and intersections	78
Figure 25 Procedure of data preparation.....	79
Figure 26 Map matching process	80
Figure 27 Average segment speed.....	82
Figure 28 Crash labeling process	83
Figure 29 Variable importance.....	84
Figure 30 LSTM unit structure (Graves et al., 2013)	85
Figure 31 Experiment procedures	87
Figure 32 Network architecture.....	88
Figure 33 t-SNEs of the raw data and extracted features.....	90
Figure 34 Model comparison results	91
Figure 35 Research approach flowchart	98
Figure 36 Bus trip generation illustration	100
Figure 37 Research area.....	108
Figure 38 Distributions of acceleration rates, dwell time, and crashes	110
Figure 39 Trajectory data preparation flowchart.....	122
Figure 40 Trip generation illustration.....	124
Figure 41 LSTM structure (Christopher, 2015)	127

Figure 42 TA-LSTM structure	129
Figure 43 Network architecture.....	130
Figure 44 Selected urban arterials	132
Figure 45 Model training process.....	134
Figure 46 Confusion matrix	136
Figure 47 Model comparison results	137

LIST OF TABLES

Table 1 Experiment summary	33
Table 2 Feature description.....	35
Table 3 Hyperparameters tuning results	37
Table 4 Experiment results	38
Table 5 Results from existing studies and our method.....	40
Table 6 Summary of driving maneuvers detection in existing studies	51
Table 7 Confusion matrix	64
Table 8 Values of hyperparameters	65
Table 9 Performance metrics of different labeled data ratios	68
Table 10 Performance metrics of model comparison	69
Table 11 Data description	77
Table 12 Descriptive Statistics.....	81
Table 13 Confusion matrix	88
Table 14 Hyperparameters tuning	89
Table 15 Model results	89
Table 16 Lynx trajectory data	107
Table 17 Descriptive statistic table.....	111
Table 18 Spearman’s rank correlation coefficients	112
Table 19 Results of Bayesian NB and Bayesian NB-CAR models.....	114
Table 20 Lynx and Lytx data description	121
Table 21 Descriptive statistics table	133
Table 22 Hyperparameters tuning	135

LIST OF ACRONYMS/ABBREVIATIONS

ATMS: Active Traffic Management System
AUC: Area under the ROC Curve
AVI: Automatic Vehicle Identification
BN: Batch Normalization
CAR: Conditional Autoregression
CNN: Convolutional Neural Network
CAN: Controller Area Network
CV: Connected Vehicles
DIC: Deviance Information Criteria
DTW: Dynamic Time Warping
DSRC: Dedicated Short Range Communication
GPS: Global Positioning System
GRU: Gated Recurrent Unit
KNN: K-Nearest Neighbors
LSTM: Long-short Term Memory
NB: Negative Binomial
OBD: On-Board Diagnostics
RNN: Recurrent Neural Network
ROC: Receiver Operating Characteristics
S4A: Signal for Analytics System
SGD: Stochastic Gradient Descent
SMS: Space Mean Speed
SMOTE: Synthetic Minority Over-sampling Technique
SSL: Semi-supervised Learning
SVM: Support Vector Machine

TA: Temporal Attention

t-SNE: t-Distributed Stochastic Neighbor Embedding

WAIC: Watanabe-Akaike Information Criterion

XGBoost: eXtreme Gradient Boosting

CHAPTER 1: INTRODUCTION

1.1 Background

In 2018, 6,734,000 traffic crashes have occurred in the USA, a 4.3% increase from the 6,453,000 crashes in 2017 (NHTSA, 2020c). To enhance traffic safety, various real-time traffic safety studies have been conducted to facilitate the development of proactive traffic safety management systems. While infrastructure-based data were widely used in the existing studies, such as data from loop detectors, Bluetooth detectors, cameras, AVIs, etc. These devices were usually installed at fixed positions to detect vehicles within certain ranges. However, there are several limitations of using these devices. First, these devices usually require additional installation and regular maintenance, which increase the cost of obtaining data. Second, data coverage is limited by the location of devices. For example, a Bluetooth detector installed at an intersection can only detect the vehicles driving through the intersection. Third, some devices are not reliable enough. For example, cameras are sensitive to light and weather conditions, while loop detectors are sensitive to pavement conditions.

Recently, with the development of Connected Vehicles (CV) and mobile sensing techniques, massive vehicle-based data can be efficiently obtained, such as the vehicle's speed, location, acceleration, and other information. Vehicle-based data can be collected through various devices without requiring additional installation, such as OBDs, smartphones, tablets, etc. Using vehicles as data collectors, vehicle-based data provide rich information on the entire trip of each vehicle. In addition, since each vehicle could be connected wirelessly, vehicle-based data can be transferred between vehicles, roadside units, and central servers. This novel type of vehicle-based data has brought new potential for traffic safety evaluation, such as vehicle

maneuvers detection, crash likelihood prediction, surrogate safety analysis, etc. Although the real deployment of the CV system still needs more time, it is currently possible to obtain CV emulated data, such as vehicle trajectory data, mobile sensor data, etc.

Mobile sensor data are usually obtained from mobile devices, such as smartphones, tablets, etc. Nowadays, a common smartphone is already equipped with various sensors, such as the accelerometer, gyroscope, GPS, etc. Sensor data from a smartphone can be used to identify vehicle maneuvers (Bhoraskar et al., 2012; Ferreira et al., 2017; Johnson and Trivedi, 2011; Singh et al., 2017), which can then be applied in traffic safety applications (Stipancic et al., 2018a; Tselentis et al., 2017). Vehicle maneuvers detection is a multi-class classification problem, which aims to detect different vehicle maneuvers (e.g., left turn, right turn, U-turn, etc.) using data from smartphone sensors. Most existing studies usually fixed the positions of smartphones for simplicity. However, in real-life, smartphones are usually in arbitrary positions. The coordinate systems of the smartphone and vehicle need to be aligned before further analysis. In addition, sliding window techniques were widely used to generate statistical features from the raw sensor data, such as mean, standard deviation, median, etc. This process may improve the accuracy but with the price of efficiency and computation cost, which is not favorable for real-time applications. Moreover, almost existing studies developed their models using supervised learning, which required sensor data to be labeled based on vehicle maneuvers. Nevertheless, preparing sensor data is extremely time-consuming due to the amount of data. New algorithms need to be proposed in order to make use of the unlabeled data.

Trajectory data can be used for predicting real-time crash likelihood. However, most existing studies only used this data for aggregated safety analysis, such as crash frequency

prediction (Wang et al., 2015b; Xie et al., 2013). Wang et al. (2019b) developed a Support Vector Machine (SVM) model for predicting crash likelihood on freeways using trajectory data from taxis. The authors considered vehicle platoon characteristics and generated several new features, such as average speed difference and speed difference ratio. Existing studies illustrated the feasibility of using taxi trajectory data for traffic safety research. However, trajectory data from other vehicles (e.g., buses, trucks, etc.,) still require additional research. It is also worthwhile to investigate the application of trajectory data fusion on real-time crash prediction (Basso et al., 2020). Moreover, trajectory data can also be used for extracting novel surrogate safety measures, which provides a convenient way to assess traffic safety status when crash events are rare or unavailable.

In terms of methodologies, statistical (Shi and Abdel-Aty, 2015; Xu et al., 2014; Yu and Abdel-Aty, 2014a; Yu et al., 2017b) and machine learning (Lin et al., 2015; Sun et al., 2014; Yu and Abdel-Aty, 2013) methodologies were the two methodologies primarily used in the previous studies. Recently, with the availability of massive transportation data and the development of computer hardware, deep learning, as one type of machine learning methodologies, is becoming a very powerful tool in transportation studies. Existing studies illustrated the superiority of deep learning methods over other methodologies. For example, Yuan et al. (2019) indicated that Long Short-term Memory (LSTM) outperformed conditional logistics regression for real-time crash likelihood prediction in terms of sensitivity and false alarm rate. Bao et al. (2019) proposed a Spatiotemporal Convolutional LSTM Network (STCL-Net), which accurately predicted citywide short-term crash risk compared with other benchmark methods, such as autoregressive integrated moving average, gradient boosting regression tree, Convolutional Neural Network (CNN), etc. In

addition, Li et al. (2020c) indicated that the performance of LSTM could be improved by augmenting it with a CNN component. The proposed LSTM-CNN achieved the highest accuracy for predicting real-time crash likelihood compared with LSTM, CNN, Bayesian Logistics Regression, and XGBoost.

In summary, with the availability of novel data in the transportation field, it is promising to investigate their performance on real-time traffic safety analysis. New data sources provide different insights to depict vehicle motion, including speed, acceleration, maneuvers, etc. Deep learning methods have been widely applied in the transportation area. However, additional studies need to be done for traffic safety analysis. The detailed objectives of this dissertation are presented in the following section.

1.2 Research Objectives

The primary objective of this dissertation is to propose novel and applicable methods for vehicle maneuvers detection, crash likelihood prediction, and surrogate safety analysis by using vehicle-based data and deep learning techniques. Current research gaps were mitigated by addressing the research objectives as below:

Objective 1: The design of a robust and accurate model for detecting vehicle maneuvers using smartphone sensors.

Objective 2: Proposing an algorithm to use the unlabeled sensor data for identifying vehicle maneuvers.

Objective 3: Developing a model for predicting real-time crash likelihood based on CV emulated data.

Objective 4: Proposing new surrogate safety measures based on trajectory data.

Objective 5: Investigating the application of trajectory fusion to crash likelihood prediction.

The first objective has been achieved in Chapter 3. A deep learning model was built to detect vehicle maneuvers with smartphone sensor data. Results indicated the proposed method could accurately detect various vehicle maneuvers compared with other existing studies and methods. Moreover, the computation cost and the transferability of the proposed model were also investigated. The proposed model could be easily transfer to new drivers/locations with a small portion of the new data. In addition, smartphones have the potential to be used as communication devices for CV systems in the future. The results from this chapter can be implemented to improve traffic safety.

The second objective has been achieved in Chapter 4. A semi-supervised learning algorithm was proposed to learn from the unlabeled sensor data. Six driving maneuvers were detected, including driving straight, left turn, right turn, U-turn, left lane change, and right lane change. Data from the smartphone's accelerometer and gyroscope were collected with a variety of smartphones, vehicles, and locations. Three LSTM models were developed with the proposed semi-supervised learning algorithm. Experimental results indicated that the proposed semi-supervised LSTM could efficiently learn from the unlabeled data and achieve excellent accuracy on detecting vehicle maneuvers. The proposed algorithm can significantly reduce the cost of sensor data preparation while implementing in a vehicle maneuvers detection system.

The third objective has been achieved in Chapter 5. A novel CV emulated data was used to predict the real-time crash likelihood on urban arterials. Trajectory data from three hundred

LYNX[®] buses were obtained with a high frequency. A series of data cleaning and preparation was proposed to transform the vehicle-level data into segment-level data. Several speed-related features were generated, including average speed, the standard deviation of speed, etc. An LSTM model was developed to use the generated features for predicting crash likelihood. The proposed model outperformed other models with higher sensitivity and Area under the ROC Curve (AUC) values. This chapter proved the feasibility of using trajectory data from buses as a new data source for real-time crash likelihood prediction. The developed model is expected to be applied into a proactive traffic safety management system and predict crash likelihood in real-time.

The fourth objective has been achieved in Chapter 6. Three critical driving events were generated using the trajectory data for pedestrian/bicycle crashes, including hard acceleration, hard deceleration, and long stop. Spearman's rank correlation coefficients were used to examine the relationships between critical driving events and the pedestrian/bicycle crashes at the bus stops. Results suggested that these three events could be used as surrogate safety measures for pedestrian/bicycle crashes. A Bayesian negative binomial model incorporating spatial correlation was developed to estimate the frequency of pedestrian/bicycle crashes using the critical events. The models' results are consistent with the results from correlation estimation. Hard acceleration, long stop events, and the number of buses (exposure) were found to be positively related to the crash frequency. The proposed surrogate safety measures could be used to evaluate the safety condition of pedestrian/bicycle at individual bus stops.

The fifth objective has been achieved in Chapter 7. Data fusion technique was used to integrate two real-world trajectory datasets with a variety of vehicle types. The traffic conditions of urban arterials were described with various speed-related features (e.g. average speed,

standard deviation of speed, etc.). To improve the performance of LSTM on predicting crash likelihood, a new deep learning architecture (TA-LSTM-CNN) was developed which containing an LSTM component with temporal attention and a CNN component. Experimental results indicated that the proposed method could achieve outstanding performance (e.g. high sensitivity and low false alarm rate) for the real-time crash likelihood prediction with the help of trajectory data fusion. Further, model comparison results suggested that the proposed model outperformed other state-of-the-art models in terms of various metrics. The proposed methods could be used for integrating different trajectory datasets for real-time crash likelihood prediction.

1.3 Thesis Structure

The rest of the thesis is organized as follows:

In Chapter 2, a detailed literature review was conducted on the data sources and methodologies in the existing real-time traffic safety analysis.

Chapter 3 presented a deep learning method to detect vehicle maneuvers based on smartphone sensors. The constraints on the smartphone's position were released using a coordination system reorientation method. Then, simply filtered sensor data were directly used rather than complicated statistical features. A stacked-LSTM model was proposed to identify the vehicle maneuvers considering the time-dependency of the sensor data. Extensive experimental results indicated that the proposed method can accurately identify different vehicle maneuvers with an average F1-score of 0.98, precision of 0.97, and recall of 0.99, which outperformed the counterparts.

Chapter 4 proposed a semi-supervised learning algorithm to learn from the massive unlabeled sensor data. Three LSTM models were trained with the proposed semi-supervised learning algorithm. Experimental results indicated that semi-supervised LSTM could achieve similar results compared with the supervised method using much less labeled data. Moreover, the proposed method outperformed other machine learning methods (e.g., CNN, XGBoost, and random forest) in terms of precision, recall, F1-score, and AUC value.

Chapter 5 explored the possibility of using novel CV emulated data to predict real-time crash likelihood on urban arterials. The CV emulated data are more cost-friendly and flexible compared with the infrastructure-based data. Crash and CV emulated data were collected from two urban arterials in Orlando, USA. Different data cleaning and preparation techniques were implemented, while various speed-related variables were generated from the CV emulated data. An LSTM model was developed to predict the crash potential for the next 5-10 minutes. The results illustrated the feasibility of using a novel CV emulated data to predict real-time crash likelihood. The average and 50th percentile speed were the two most important variables for the crash likelihood prediction.

Chapter 6 proposed three novel surrogate safety measures based on bus trajectory data. Specifically, three critical driving events were identified based on the bus's acceleration rate and stop time, hard acceleration, hard deceleration, and long stop. The relationships between critical driving events and crashes were examined using Spearman's rank correlation coefficient. All three events were positively correlated with pedestrian and bicycle crashes. Long stop event had the highest correlation coefficient, followed by hard acceleration and hard deceleration. A Bayesian negative binomial model incorporating spatial correlation (Bayesian NB-CAR) was

built to estimate the pedestrian and bicycle crash frequency using the generated events. The results were consistent with the correlation estimation. For example, hard acceleration and long stop events were both positively related to pedestrian and bicycle crashes.

Chapter 7 proposed a Temporal Attention-based LSTM-CNN (TA-LSTM-CNN) for predicting real-time crash likelihood with trajectory data. Two real-world trajectory data were prepared with a series of steps, including trip generation, speed estimation, and data fusion. Extensive experimental results showed that the data fusion scenario achieved better results compared with the single data scenario. Besides, the temporal attention also improved the performance of the LSTM-CNN with higher sensitivity, AUC, and lower false alarm rate. The results from model comparisons indicated that the proposed TA-LSTM-CNN outperformed other benchmark models (e.g., logistics regression, XGBoost, etc.) with various evaluation metrics.

CHAPTER 2: LITERATURE REVIEW

2.1 Data Sources for Traffic Safety Analysis

2.1.1 Infrastructure-based Data

Most existing traffic safety studies used infrastructure-based data from various devices, which were usually installed in fixed positions to probe certain traffic conditions. Loop detectors were widely utilized to collect traffic flow conditions for traffic safety analysis (Abdel-Aty and Abdalla, 2004; Abdel-Aty and Pande, 2005; Lee et al., 2002; Oh et al., 2005). Lee et al. (2002) used the data from 38 loop detector stations to examine crash precursors on freeways. Results suggested the variation of speed and traffic density were statistically significant predictors of crash frequency. Similarly, Abdel-Aty and Abdalla (2004) obtained average speed, volume, and occupancy rate from the loop detectors for safety analysis on freeways. The authors indicated that high variability and low variability in volume could increase the likelihood of crashes. However, the major issue for loop detectors is the reliability. Loop detectors tend to fail due to the very hard environment of pavement and temperature variation (Ahmed and Abdel-Aty, 2012). Therefore, several new nonintrusive devices were introduced, such as the AVI, Bluetooth, camera, etc. For example, the AVI system could capture and calculate the travel time for the vehicles passing through the electronic toll collection. Space Mean Speed (SMS) can then be estimated accordingly. SMS is the average speed of all vehicles occupying a given stretch of the road over some specified period (Shi et al., 2016; Yu and Abdel-Aty, 2014b). Yu and Abdel-Aty (2014b) used the SMS data from the AVI system to analyze crash injury severity. Modeling results demonstrated that large variations of speed before the crash occurrence would increase

the likelihood of severe crashes. Similarly, Bluetooth detectors were used to collect vehicle SMS data (Yuan et al., 2018). Bluetooth detectors can detect the vehicles equipped with Bluetooth devices while the devices are working at discoverable modes. The SMS on a specific segment is calculated as the segment length divided by the travel time of each detected vehicle on the segment based on the detection data of two Bluetooth detectors located at the two contiguous intersections. Yuan et al. (2018) used Bluetooth and other types of data for real-time safety analysis on urban arterials. The results revealed that the average speed had significant effects on the occurrence of crashes. Besides, cameras were also used to obtain data for traffic safety analysis (Zhang et al., 2020a; Zhang et al., 2021). Using object detection and tracking techniques, road users' trajectory information can be extracted from the cameras. This information can then be used for traffic conflicts analysis.

Infrastructure-based data were popular among the existing traffic safety studies since they could depict traffic conditions from different aspects. However, there are several limitations of using this type of data. First, these devices require installation and regular maintenance, which increase the cost of obtaining data. It also takes a long time to deploy these devices in a new area due to the same reason. Second, infrastructure-based data are usually archived first and then extracted. It is hard to obtain the data for real-time traffic safety analysis, which has a high requirement for the data update frequency. Third, the coverage of the infrastructure-based data is limited. For example, the Bluetooth detector at an intersection can only detect the vehicles near it rather than the vehicles on the whole segment. In addition, some devices are not reliable enough. For example, cameras are sensitive to lighting and weather conditions. Loop detectors are sensitive to the surrounding environment as mentioned before.

2.1.2 Vehicle-based Data

Using individual vehicle as the data collector, vehicle-based data are more flexible than infrastructure-based data. This data can be collected through various devices, such as OBDs, smartphones, tablets, etc. Two types of vehicle-based data were reviewed in this section, trajectory and mobile sensor data.

- **Trajectory Data**

Trajectory data were widely applied in transportation management fields. Herring et al. (2010) used taxi trajectory data to estimate and predict arterial travel time distributions in San Francisco. Similarly, Rahmani et al. (2015) utilized the low-frequency taxi trajectory data to estimate the mean and other statistics of the travel time distribution, such as the median and various percentiles. Liu et al. (2013) applied taxi trajectory data to estimate the turn delays at the intersections in Beijing. Kuang et al. (2015) used taxi trajectory data to identify traffic anomalies in urban arterials. Several abnormal traffic events, such as road construction and large exhibitions, were successfully detected by the proposed methods.

However, the application of vehicle trajectory data to traffic safety is still limited, while most studies focused on aggregated safety analysis. Xie et al. (2013) used taxi data to calculate arterial-level travel speed and introduced it as a new explanatory variable for the occurrence of crashes. The authors found that higher average speed along arterials was associated with an increasing of crashes. Similarly, Wang et al. (2015b) examined the relationships between different variables from taxi trajectory data and the occurrence of crashes on urban arterials during peak and off-peak hours. Higher average speed was found to be associated with higher frequency of crashes during peak periods, but not during off-peak periods. Bao et al. (2019) used

the numbers of taxi pick-ups and drop-offs as new variables to predict citywide crash frequency based on deep learning models. Wang et al. (2019b) applied the SVM model to predict crash potential on freeways using trajectory data. Different variables were generated from the trajectory data, such as average speed, speed difference ratio, etc.

In summary, vehicle trajectory data were widely applied in transportation management fields. However, the performance of this data on traffic safety analysis, especially real-time crash likelihood prediction, still requires additional research. Moreover, almost all the existing studies used taxi trajectory data for analysis. Additional studies need to be done on other types of vehicles, such as buses, trucks, etc. In addition, Basso et al. (2020) indicated it was necessary to distinguish vehicle types for crash prediction. Results from a case study on an expressway indicated that having access to disaggregated data by vehicle type could improve the accuracy of prediction up to 30 %.

- **Mobile Sensor Data**

Mobile sensor data can be obtained using different mobile devices, such as smartphones, tablets, etc. Due to the ubiquity of mobile devices, it is now much easier to obtain sensor data. Mobile sensor data are usually complementary to the vehicle trajectory data since it could provide more detailed information on individual vehicle.

The sensor data from the mobile devices can be used to identify vehicle maneuvers. Previous studies showed that some vehicle maneuvers were related to crash likelihood. Stipanovic et al. (2018a) analyzed the correlation between historical crashes and vehicle maneuvers, hard braking and accelerating events. Results from Spearman's correlation coefficient and pairwise Kolmogorov-Smirnov indicated that both maneuvers were positively correlated with crash

frequency at the link and intersection levels. In addition, some vehicle maneuvers, such as harsh turning, harsh acceleration, are used as important driver classification metrics by many insurance companies, as they appear to be related to crash likelihood (Tselentis et al., 2017).

Extensive studies have been conducted for identifying vehicle maneuvers using sensor data over the past decade. Johnson and Trivedi (2011) applied the Dynamic Time Warping (DTW) method to identify vehicle maneuvers (e.g., right turn, left turn, acceleration, and brake) using data from smartphone sensors. The smartphone was mounted in the center of a vehicle's windshield. To eliminate noise in the data, a simple moving average filter was applied. Similarly, Singh et al. (2017) utilized DTW to detect sudden braking and lateral maneuvers using data from a smartphone's accelerometer, gyroscope, gravity sensor, and GPS. The smartphone was mounted on the car's dashboard to collect sensor data. The simple moving average and band pass filter were used to smooth the raw sensor data. The system reached an average accuracy of 94.55% to detect aggressive maneuvers. Bhoraskar et al. (2012) proposed a system called Wolverine, which used SVM to detect bumps and braking events. Accelerometer, GPS, and magnetometer were utilized to collect data. Besides, a coordinate reorientation method was introduced to align the coordinate system. The system achieved a false positive rate of 2.7% and a false negative rate of 21.6% in braking detection. Ferreira et al. (2017) investigated the combination of different sensors to detect aggressive driving behaviors, such as aggressive braking, acceleration, left turn, and right turn. Accelerometer and gyroscope were proved to be the most suitable sensors. Besides, the random forest was found to have the highest AUC value among different methods. Wang et al. (2015b) designed a threshold-based system to detect turning and lane-change movements. Data from gyroscope, accelerometer, and GPS were used in

the study. The proposed system achieved an overall accuracy of 96% for detecting different maneuvers.

To conclude, it is promising to use mobile devices to detect vehicle maneuvers, which can then be used as new factors for traffic safety analysis. Most of the studies obtained sensor data from accelerometer and gyroscope and applied a filter to remove the noise.

2.2 Real-time Traffic Safety Analysis

Real-time traffic safety analysis is expected to play an important role in the Active Traffic Management Systems (ATMS). There are generally two types of real-time traffic safety analysis based on their objectives, one is focusing on crash probability and another one is focusing on crash frequency. This section mainly reviews the studies related to real-time crash likelihood prediction. Real-time crash probability prediction aims to predict the crash probability within a short period (e.g. 5-10 minutes). Existing studies can be divided into two types based on the research locations, which are freeways and urban arterials.

Most of the existing studies are conducted on freeways (Abdel-Aty et al., 2012; Ahmed et al., 2012b; Oh et al., 2005; Yu and Abdel-Aty, 2014a) rather than urban arterials (Theofilatos, 2017; Yuan et al., 2018). One important component of these studies is to identify the relationship between real-time crash likelihood between different types of data, such as transportation, signal timing, weather, road geometry, etc. Speed-related variables were found among the most significant variables in terms of crash risk. The average speed was found to be negatively correlated with the crash likelihood (Abdel-Aty et al., 2012; Ahmed et al., 2012a; Ahmed and Abdel-Aty, 2012; Shi and Abdel-Aty, 2015; Xu et al., 2012; Yu et al., 2016). The standard

deviation of speed was found to have significant positive effects on crash occurrence (Abdel-Aty et al., 2004; Abdel-Aty et al., 2012; Ahmed et al., 2012a; Ahmed and Abdel-Aty, 2012; Xu et al., 2012; Zheng et al., 2010). The number and level of speed mutations were found to be significantly related to the crash risk (Wang et al., 2019b). Speed mutation is defined when the acceleration rate (absolute value) over certain thresholds. Intuitively, higher traffic volume contributes to higher crash risk (Roshandel et al., 2015). Moreover, several studies (Hossain and Muromachi, 2012; Shi and Abdel-Aty, 2015) reported that the congestion index is positively correlated with crash occurrence. In addition, Yuan et al. (2018) found that the downstream green ratio was negatively associated with the occurrence of crashes at urban arterials, this could be explained as the higher downstream green ratio could efficiently reduce the percentage of stop-and-go traffic.

In summary, more studies need to be done on real-time crash analysis, especially for urban arterials. In addition, speed-related variables were found to be the most important variables for crash occurrence. However, the relationships between some speed-related variables and crash occurrence are still not clear, such as different speed percentile, acceleration rate, etc.

2.3 Methodologies

This section reviews the main methodologies used in real-time crash likelihood prediction. Real-time crash likelihood prediction is a classification problem with the dependent variables as ‘crash’ or ‘non-crash’. The modeling methods for real-time crash likelihood prediction can be divided into statistical and machine learning methods. In addition, since the crash is a rare event, data resampling methods also need to be applied before further analysis.

2.3.1 Statistical Methods

Statistical methods were firstly applied for crash prediction. Several methods were commonly used by the existing studies, including logistic models (Abdel-Aty and Pande, 2005; Abdel-Aty et al., 2004; Ahmed and Abdel-Aty, 2012; Xu et al., 2012; Zheng et al., 2010), Bayesian logistical models (Shi and Abdel-Aty, 2015; Wang et al., 2017a; Wang et al., 2015a; Yu et al., 2014), Bayesian random effect logistic models (Shi and Abdel-Aty, 2015; Yu et al., 2016), Bayesian random parameter logistic models (Shi and Abdel-Aty, 2015; Xu et al., 2014; Yu and Abdel-Aty, 2014a; Yu et al., 2017b). The benefits of using statistical models are they can quantify the correlations between crash risk and other types of variables, which can help identify the most crucial factors in terms of crash risk. However, these models are usually built on matched-case control data or have certain assumptions. Existing studies indicated machine learning methods usually had better performance compared with statistical methods, such as higher sensitivity and AUC values (Bao et al., 2019; Yu and Abdel-Aty, 2013).

2.3.2 Machine Learning Methods

With the capability of learning from big data and non-linear relationships, machine learning methods were used to predict real-time crash likelihood. Yu and Abdel-Aty (2013) applied the SVM to predict the real-time crash likelihood prediction at freeways. Results indicated that SVM outperformed the Bayesian logistic regression model in terms of AUC values. Similarly, Sun et al. (2014) used SVM to predict real-time crash likelihood on urban expressways. Results suggested that SVM had better performance than logistics regression, Bayesian networks, and K-Nearest Neighbor (KNN). In addition, the authors also illustrated the

transferability of SVM by applying the model on different expressways. In addition, Lin et al. (2015) found that Tree-based methods can be used for the variable selection process, which could improve the accuracy of real-time crash likelihood prediction.

One major branch of machine learning is deep learning, which mainly referred to deep neural networks. The concept of deep neural networks was proposed over decades ago but was not been widely applied due to the limitation of the computer hardware. Recently, with the rapid development of computation power and the availability of massive data, deep learning was proved to be very useful in terms of image classification, face recognition, natural language processing, etc. Deep learning methods are capable to learn from massive data and achieve very high accuracy in both regression and classification problems. There are various applications of deep learning methods in transportation fields. Chen et al. (2016) developed a deep stack denoise autoencoder model to learn from hierarchical features of human mobility and predict traffic crashes in an aggregated way. Ma et al. (2017) implemented a deep CNN model to predict traffic speed in Beijing. Spatial and temporal traffic dynamics are converted to images describing the time and space relations of traffic flow. Results indicated CNN outperformed other common methods such as random forest and KNN.

Transpiration research problems usually contain lots of time-series data, specifically, Recurrent Neural Network (RNN) was proved to be a powerful neural network in terms of learning time-series data (Mahmoud et al., 2021a; Tian and Pan, 2015; Zhao et al., 2017; Zheng et al., 2019). Different from the traditional neural network that only maps the current input vector to the output vector (Tian and Pan, 2015), RNN introduces recurrent connections, which allow information to persist. However, one drawback of RNN is it cannot capture long-term

dependency (Bengio et al., 1994). Thus, LSTM was invented by Hochreiter and Schmidhuber (1997). LSTM improves the performance of RNN by including memory cells and gates, which preserve the information for a long period.

Several existing studies applied LSTM in transportation fields. Zhao et al. (2017) investigated the short-term traffic forecast of Beijing based on LSTM, the proposed LSTM network considers spatio-temporal correlations in the traffic system via a two-dimensional network. The results of LSTM are better than other methods, such as autoregressive integrated moving average model and normal RNN. Similarly, Tian and Pan (2015) utilized LSTM to predict short-term traffic flow. With the ability to memorize long historical data and automatically determine the optimal time lags, LSTM outperformed other common machine learning methods, such as SVM and single layer feed forward neural network. There are currently few papers that used LSTM for real-time crash likelihood prediction. For example, Yuan et al. (2019) utilized LSTM to predict crash risk in real-time, the authors proved the performance of LSTM was much better than the conditional logistic model in terms of sensitivity and false alarm rate.

CHAPTER 3: A DEEP LEARNING APPROACH TO DETECT REAL-TIME VEHICLE MANEUVERS BASED ON SMARTPHONE SENSORS

3.1 Introduction

In 2017, the reported number of fatalities from vehicle crashes was 44,034 in the USA (National Highway Traffic Safety Administration, 2019). Among them, 64.2% involved the vehicle going straight, 7.3% involved turning left, 0.9% involved turning right, etc. To prevent crashes, there is a need to identify vehicle maneuvers in real-time and provide drivers assistance or warning accordingly. As existing studies indicated (Ferreira et al., 2017; Singh et al., 2017), each vehicle maneuver has its unique characteristics, such as its speed, acceleration, and rotation rate. These characteristics can be obtained via various devices, including smartphones, cameras, OBDs, radars, etc. Among them, smartphones are becoming more and more popular because of their high penetration rate, robustness, and low cost.

First, the smartphone is a ubiquitous device. In 2020, 70% of the world's population will use smartphones (Kanarachos et al., 2018). However, radar and other advanced sensors are only available on luxury vehicles (Wang et al., 2015b). Second, some devices, such as the camera, can be easily influenced by the weather and light conditions (Wang et al., 2015b). Third, the smartphones are already equipped with various sensors (e.g., accelerometer, gyroscope). It is not necessary to install extra devices. In addition, current high-speed cellular networks and the incoming 5G technology make it possible to use the smartphone for the vehicle to vehicle and vehicle to pedestrian communication systems. The information of vehicle maneuvers can be

broadcasted to other road users and enhance traffic safety.

Vehicle maneuvers detection is a classification problem based on time-series data. In general, two types of methods were commonly used among the existing studies, which are rule-based and machine learning methods. For the rule-based methods, vehicle maneuvers are detected according to certain thresholds. For example, acceleration, braking, and left/right turn are considered as violations of thresholds imposed on vehicle acceleration of different axes (Paefgen et al., 2012). Turning maneuvers can be detected using certain constraints on the headings obtained from the GPS sensors (Saiprasert et al., 2013). Lane change maneuvers can be identified based on certain threshold values on the rotation rate, which is collected through a gyroscope (Wang et al., 2015b). The rule-based methods are easy to use but are not flexible and accurate enough since the thresholds are usually set manually. Considering these limitations, machine learning methods have become more popular since their high flexibility (Ferreira et al., 2017; Yu et al., 2017a), including random forest, SVM, KNN, etc. Recently, with the rapid development of neural networks, they were implemented to various transportation problems and achieved promising results. Among various neural networks, RNN was proved to be especially useful for learning time-series data (Tian and Pan, 2015; Zhao et al., 2017; Zheng et al., 2019), compared to common machine learning methods. Different from the traditional neural network that only maps the current input vector to output vector (Tian and Pan, 2015), RNN introduces recurrent connections, which allow information to persist. However, one drawback of the RNN is it cannot capture long-term dependency (Bengio et al., 1994). Thus, LSTM, was invented by Hochreiter and Schmidhuber (1997). LSTM improves the performance of RNN by including memory cells and gates, which preserve the information for a long period. There are already

various applications of LSTM in transportation fields, including traffic speed prediction, traffic volume prediction, crash risk prediction, etc. However, there are only a few papers that applied LSTM to vehicle maneuvers detection. Saleh et al. (2017) utilized LSTM to classify drivers' behaviors as normal, aggressive, and drowsy based on data from the accelerometer, gyroscope, GPS, and camera. Mumcuoglu et al. (2019) implemented LSTM to classify different drivers according to the longitudinal and lateral acceleration. However, these studies only provided coarse-grained results and used too many sensors. The performance of LSTM on real-time vehicle maneuvers detection still requires more investigations.

Two types of data, filtered sensor data and statistical features, are used as the inputs for identifying vehicle maneuvers. Specifically, the first one applies sensor filters to remove the noise from the raw data (Wang et al., 2015b). The second one generates statistical features based on a sliding window. With a predefined size, the sliding window goes through the sensor data and estimates the statistical features accordingly. For instance, Ferreira et al. (2017) used the sliding window to obtain the mean, median, standard deviation from the sensor data. Similarly, Yu et al. (2017a) estimated the standard deviation, mean, maximum, and minimum of the sensor readings as features for modeling.

Furthermore, one crucial step for vehicle maneuvers identification is sensor selection. A single sensor can provide valuable but limited information for vehicle motions. For example, the gyroscope can detect aggressive turning. The accelerometer is good at detecting aggressive acceleration and aggressive lane changing (Ferreira et al., 2017). Sensor-fusion combines different sensors as inputs. Johnson and Trivedi (2011) proved it is difficult to detect the U-turn when using the accelerometer or gyroscope alone. However, sensor fusion could improve the

accuracy from 23% to 77%. Moreover, the GPS and accelerometer can be fused to improve the accuracy of the vehicle speed estimation (Chowdhury et al., 2014).

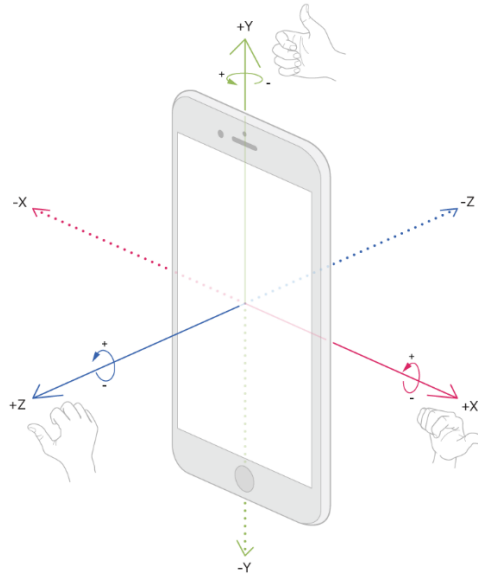
There are still several challenges of using smartphone sensors to detect vehicle maneuvers. First, in previous studies, the smartphone was fixed on a mount (Chowdhury et al., 2014; Singh et al., 2017). This is not consistent with the real-life situation. Free position could introduce more noises and make it difficult to reflect vehicle motions through the smartphone. Second, many studies generate new statistical features based on sensor data or use complex data preprocessing techniques to improve the performance of the model. However, these procedures increase the computational complexity and make them hard to apply in real-time. Third, sensor fusion may improve the model's performance, but using too many sensors could also increase battery consumption. This study tries to mitigate the existing research gaps by proposing a real-time vehicle maneuvers detection system. First, we release the constraints on the smartphone's position by a coordination system reorientation method. Second, the filtered raw sensor data are directly used instead of generating new statistical features, which requires much less computation cost. Third, this study uses readings from only two sensors as the inputs. This could save more battery than fusing too many sensors. Forth, since vehicle maneuvers detection is a typical time series-related sequential prediction process, the long-term dependency of the sensor data can be captured efficiently by the proposed stacked-LSTM. To the authors' best knowledge, this is the first study to conduct real-time vehicle maneuver detection using stacked-LSTM.

3.2 Sensor Description

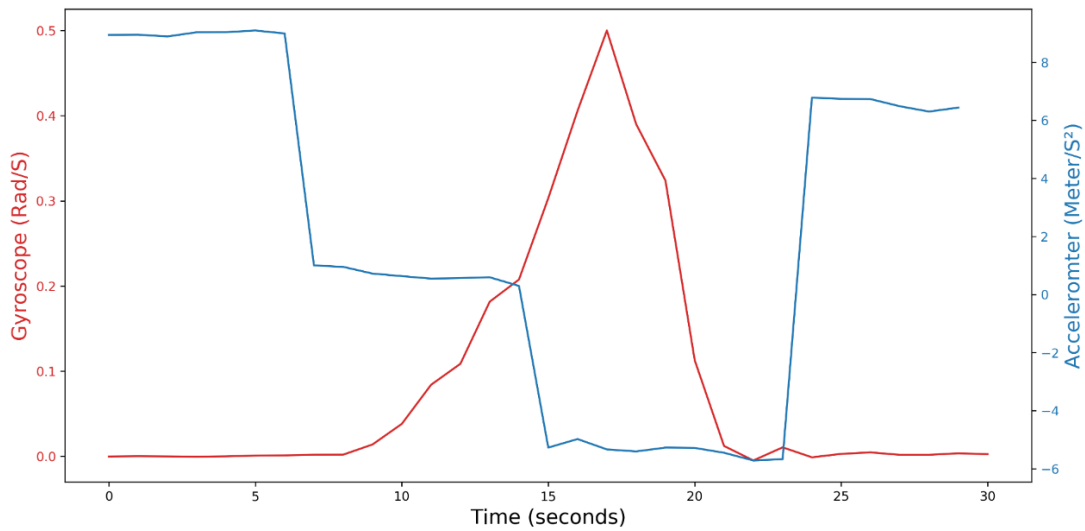
Nowadays, a common smartphone usually has many built-in sensors, such as the accelerometer, gyroscope, magnetometer, GPS, etc. These sensors measure the vehicle motions from different aspects. This part explains their roles in detecting vehicle maneuvers. The sample data used here were collected by an Android application developed by one of the authors. For simplicity, the sensor update frequency is set as 1 Hz. The coordinate system of the smartphone and vehicle are aligned.

3.2.1 Accelerometer

The accelerometer measures the acceleration of the smartphone on three axes with the unit of m/s^2 (Figure 1 (a)). It is widely used in driving maneuvers detection and activities recognition (Paefgen et al., 2012; Saiprasert et al., 2013; Vlahogianni and Barmounakis, 2017). The readings from the accelerometer show significant patterns for different vehicle maneuvers. For instance, if a vehicle is making a left turn, the X-axis of the accelerometer will decrease dramatically ((Figure 1 (b))).



(a) Sensor coordinate system



(b) Sensor readings of left turn

Figure 1 Sensor coordinate system and readings

3.2.2 Gyroscope

Gyroscope reflects the smartphone's rotation rates on three axes in rad/s, its directions are shown by the hands in Figure 1 (a). Gyroscope is helpful in navigation applications as well as some smartphone games. It is one of the most useful sensors to detect vehicle maneuvers.

Gyroscope is a major sensor for reflecting wheel steerings, such as lane change and turning. For instance, the reading of the Z-axis is increasing significantly if a vehicle is making a left turn (Figure 1 (b)).

3.2.3 GPS

GPS sensor provides location information, such as longitude, latitude, bearing, and speed. Specifically, the longitude and latitude reflect the location of the smartphone in degrees. Bearing is the horizontal direction of travel of the smartphone, from 0 to 360 degrees. Speed is the speed of the smartphone in m/s. In general, different vehicle maneuvers have different speeds. A car is expected to have a lower speed while it is turning compared to normal driving.

3.2.4 Magnetometer

Magnetometer estimates the magnetic field at the position of a smartphone on Earth. The magnetometer can be used to obtain the angle between the horizontal component of the magnetic field and the true north, in degrees (i.e., positive means the magnetic field is rotated east that much from true north). The main application of the magnetometer in this study is to align the coordinate systems of smartphones and vehicles.

3.3 System Overview

This study aims to detect real-time vehicle maneuvers with smartphone sensors using a deep learning method. The system workflow is illustrated in Figure 2. First, sensor data are collected. Second, a coordinate system reorientation method and a data filter are utilized to align the smartphone coordination system and remove noise, respectively. Third, data from the accelerometer and gyroscope are used to detect vehicle maneuvers based on a stacked-LSTM

model. Four maneuvers are detected, including going straight, left turn, right turn, and U-turn. These maneuvers' information can be broadcasted to the road users. In terms of communication technologies, Dedicated Short Range Communication (DSRC) (Tahmasbi-Sarvestani et al., 2017), cellular (Chika et al., 2008; Liu et al., 2015), and Wi-Fi (Huang et al., 2016; Liu et al., 2016b) are mainly used by the previous studies. However, DSRC requires the smartphone to be equipped with 802.11p, which is not available for any off-the-shelf smartphone (Sewalkar and Seitz, 2019). Therefore, the work in this study is designed to use cellular and Wi-Fi technologies to transmit maneuvers information.

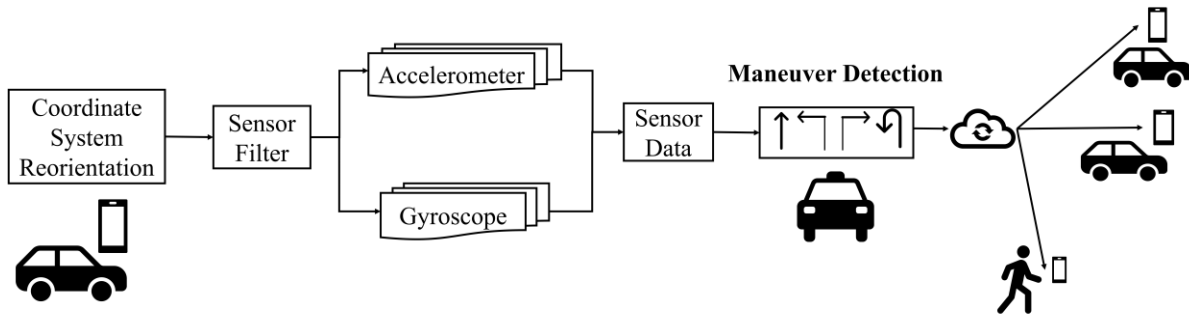


Figure 2 System workflow

3.3.1 Coordinate System Reorientation

Smartphones and vehicles have similar coordinate systems, which are shown in Figure 3 (a) and Figure 3 (b), respectively. In real life, the position of a smartphone to a vehicle is usually arbitrary. Therefore, the sensor reading of a smartphone cannot truly represent the vehicle motions. Most of the previous studies fixed the smartphone for simplicity (Johnson and Trivedi, 2011; Singh et al., 2017). However, it is more realistic to free the smartphone's position and

align its coordinate system with the vehicle's, which can be achieved based on the rotation matrix (Bhoraskar et al., 2012).

The coordinate system reorientation is implemented in two steps. Taking the accelerometer as an example. In the first step (Equation (1)) (Bhoraskar et al., 2012), we multiply the rotation matrix with the three sensor axes. The rotation matrix R is the 3 by 3 matrix obtained from the smartphone. a_x , a_y , and a_z are the X, Y, and Z-axis of the accelerometer. The smartphone coordination system is then transformed into the geometric coordination system (a'_x , a'_y , and a'_z). The geometric coordinate system is shown in Figure 3(c). Specifically, X-axis is tangential to the ground at the device's current location and roughly points east). Y-axis is tangential to the ground at the device's current location and points towards the magnetic North Pole. Z-axis points towards the sky and is perpendicular to the ground.

$$\begin{bmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{bmatrix} \times \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} = \begin{bmatrix} a'_x \\ a'_y \\ a'_z \end{bmatrix} \quad (1)$$

The second step is conducted based on magnetic declination and bearing (Equations (2)-(5)). For example, a'_x from the first step is transformed to a''_x , which is aligned with the vehicle coordinate system. Magnetic declination is the deviation of magnetic north from true north while bearing is the angle between the vehicle's direction and the true north (Figure 3 (d)). These two parameters can be directed obtained through the smartphone's magnetic sensor and GPS, respectively. After these two steps, the smartphone's sensors could better reflect the vehicle motions. Take the accelerometer as an example, after the coordinate system reorientation, the X-

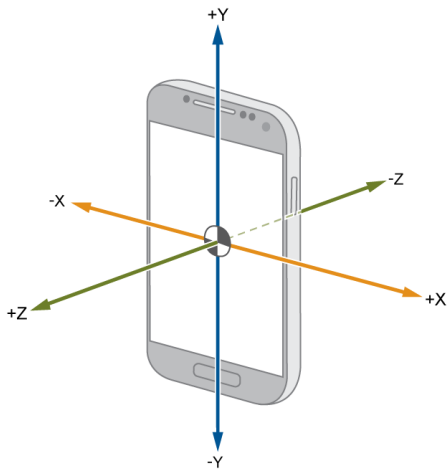
axis of the accelerometer represents the vehicle's lateral acceleration, the Y-axis represents the vehicle's longitudinal acceleration, and the Z-axis represents gravity.

$$\theta = \text{bearing} - \text{magnetic declination} \quad (2)$$

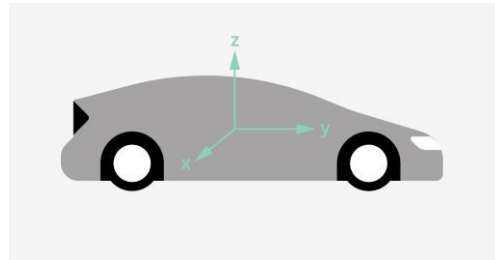
$$a''_x = a'_y * \sin \theta + a'_x * \cos \theta \quad (3)$$

$$a''_y = a'_y * \cos \theta - a'_x * \sin \theta \quad (4)$$

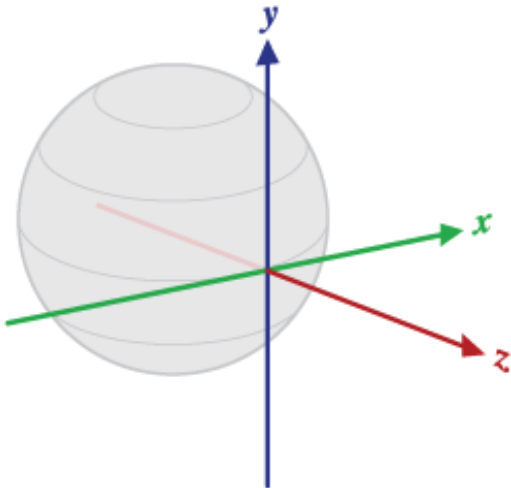
$$a''_z = a'_z \quad (5)$$



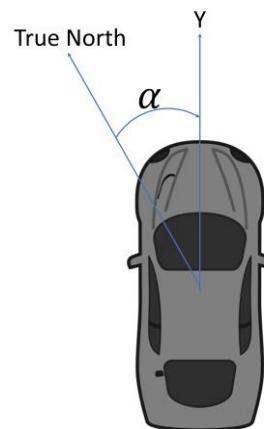
(a) Smartphone coordinate system (MathWorks, 2019)



(b) Vehicle coordinate system



(c) Geometric coordinate system (Google, 2019)



(d) Illustration of bearing

Figure 3 Coordinate system reorientation illustration

3.3.2 Sensor Filter

Smartphone sensor data contain lots of noise. Directly using the raw sensor data will impair the accuracy of maneuvers detection. Most of the existing studies applied sensor filters before further analysis. Among them, moving average filter was widely used and achieved reasonable results (Johnson and Trivedi, 2011; Liu et al., 2016c; Saiprasert et al., 2013; Singh et al., 2017; Wang et al., 2015b). For example, Wang et al. (2015b) applied the moving average filter to remove the noise from raw gyroscope readings. Liu et al. (2016c) utilized the same filter to remove noise from raw acceleration. The moving average filter for a series Y can be calculated based on Equation (6) (Wikipedia, 2019).

$$S_t = (Y_t + Y_{t-1} + \dots + Y_1)/t \quad (6)$$

Where Y_t is the value at a time period t , and S_t is the value after filtering at any time period t . In this study, the Y is the reading from the sensor axis. The moving average filter is applied to each of the sensor axes accordingly.

3.3.3 Long Short-term Memory Neural Network

Deep learning methods were proven to have better performance than other machine learning methods in learning sensor data (Ordóñez and Roggen, 2016; Yuan and Abdel-Aty, 2018). Moreover, existing studies indicated that LSTM had better result than others. For example, Yuan and Abdel-Aty (2018) built an LSTM network for human activity recognition based on smartphone sensors. Results suggested that LSTM had the highest accuracy compared with SVM, CNN, DTW, etc. Ordóñez and Roggen (2016) applied LSTM to the human activity recognition problem. The authors indicated LSTM outperformed CNN with higher accuracy.

LSTM can more efficiently learn from time-series data compared with other neural networks due to its unique design of the memory cells, which contain three gates to control when to forget or remember certain information. In addition, the ability to look back on several time steps also helps LSTM reach better results on time-series data than common neural networks. In terms of different variants of LSTM, Greff et al. (2017) compared the performance of LSTM over its eight variants. Results from 5400 experiment runs (roughly 15 years of CPU time) suggested that none of the variants can improve significantly upon the standard LSTM architecture. Besides, the forget gate and the output activation function were indicated to be the most critical components of LSTM.

This study developed an LSTM model for vehicle maneuvers detection due to its good performance on time-series data. LSTM is one kind of RNNs. Different from traditional RNNs, LSTM is capable of learning long-term dependencies (Hochreiter and Schmidhuber, 1997) since it introduced the memory cells to determine when to forget certain information. The structure of an LSTM layer is shown in Figure 4. LSTM has three different gates compared with the RNN. The forget gate f_t controls how much to forget from the previous step memory cell. The input gate i_t determines which values to be updated. The output gate o_t determines the output for the hidden state h_t . The introduction of these special gates makes it easier for LSTM to preserve information over long timestamps. For instance, if we take the forget gate as 1 and the input gate as 0, then the information of this memory cell is preserved indefinitely.

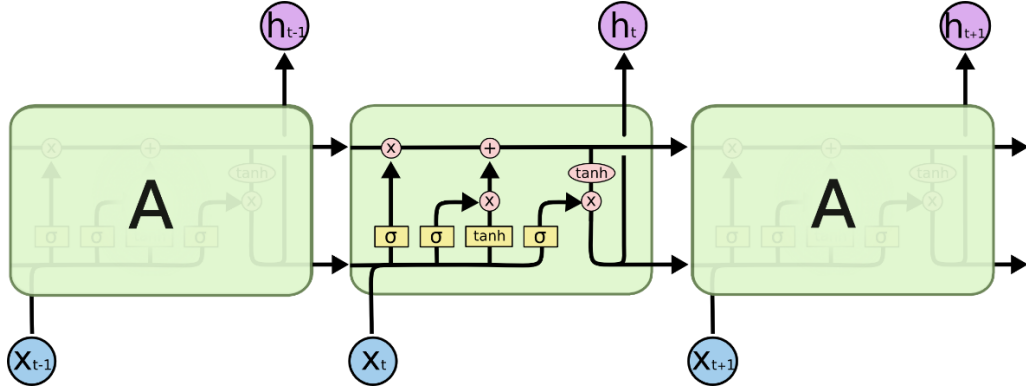


Figure 4 LSTM structure (Christopher, 2015)

If the input of LSTM is denoted as $X = (X_1, X_2, \dots, X_t)$, where t is the prediction period, $h = (h_1, h_2, \dots, h_t)$ is the hidden state, and the $y = (y_1, y_2, \dots, y_t)$ is the output. The equations of LSTM are shown in Equations (7)-(12).

$$i_t = \sigma(W_{ix}X_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) \quad (7)$$

$$f_t = \sigma(W_{fx}X_t + W_{fh}h_{t-1} + W_{fc}c_{t-1} + b_f) \quad (8)$$

$$o_t = \sigma(W_{ox}X_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o) \quad (9)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tanh(W_{cx}X_t + W_{ch}h_{t-1} + b_c) \quad (10)$$

$$h_t = o_t \odot \tanh(c_t) \quad (11)$$

$$y_t = W_{yh}h_{t-1} + b_y \quad (12)$$

Where W represents weight matrices, for example, W_{ix} denotes the weight matrix from the input gate to the input, σ is the logistic sigmoid function, and \odot indicates elementwise product of the vectors. The forget gate f_t controls the extent to which the previous step memory cell is forgotten, the input gate i_t determines how much to update for each unit, and the output

gate o_t controls the exposure of the internal memory state. Since the values of all the gating variables vary for each time step, the model could learn how to represent information over various time steps.

3.4 Experiment and Results

3.4.1 Data Collection

To collect real-world driving data, we conducted extensive experiments from April 2019 to July 2019. The summary of the experiment is shown in Table 1.

Table 1 Experiment summary

Name	Description
Driver number	Four anonymous drivers
Smartphone models	Samsung Note10, Samsung Galaxy S8, Samsung Galaxy S10
Vehicle models	Toyota Corolla, Nissan Altima, Chevrolet Malibu
Sampling rate	10 Hz
Sensor	Accelerometer, Gyroscope, GPS

Furthermore, the interface of the Android application is shown in Figure 5. The unit for the input frequency is in million seconds. Moreover, the user can select the sensors they want to activate, including GPS, accelerometer, gyroscope, barometer, and compass. The output data are saved automatically.

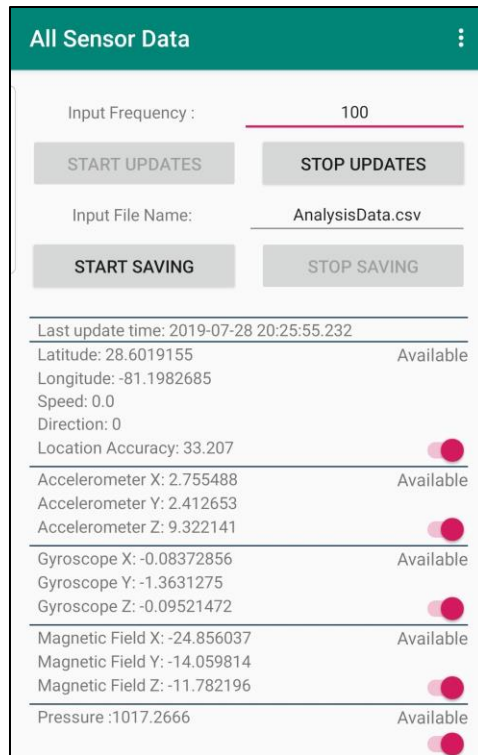


Figure 5 Smartphone application interface

Four anonymous drivers collected the data during their commutes, the average commute time was around 15 minutes to 30 minutes. The drivers were asked to drive on both weekdays and weekends. The sensor update frequency was set as 10 Hz. In total, 77,642 samples were collected, including 71,522 going straight, 2,563 left turn, 2,840 right turn, and 717 U-turn. Regarding the locations of the vehicle maneuvers, most of the turning maneuvers happened at intersections, while segments have most of the going straight maneuvers.

For sensor selection, accelerometer, gyroscope, camera, and GPS were commonly used by previous studies. However, Camera and GPS usually consumes much more battery than the other sensors and are more sensitive to the environment. For example, the battery consumption from the accelerometer is 1mA, for GPS is 15mA (Wu et al., 2014). Besides, the signal strength

of GPS can be easily influenced by the outside environment. Thus, this study does not use GPS as a candidate sensor. The description of the selected features is shown in Table 2.

Table 2 Feature description

Feature	Sensor Name	Description
a_x	X-axis of accelerometer	Vehicle's lateral acceleration
a_y	Y-axis of accelerometer	Vehicle's longitudinal acceleration
g_x	X-axis of gyroscope	Vehicle's pitch rate
g_y	Y-axis of gyroscope	Vehicle's roll rate
g_z	Z-axis of gyroscope	Vehicle's yaw rate

3.4.2 Model Implementation

To better capture the time dependency of the sensor data, this study builds a neural network with two stacked LSTM layers to identify vehicle maneuvers. Dropout layers are added to prevent over-fitting, the architecture of the model is shown in Figure 6. Data of three seconds are stacked to identify the vehicle maneuver in the current time. Specifically, the data from the $T - 2$, $T - 1$, and T seconds are used to identify the maneuver at the T second. Moreover, Softmax function is used as the activation function to generate the output since there are multiple maneuvers.

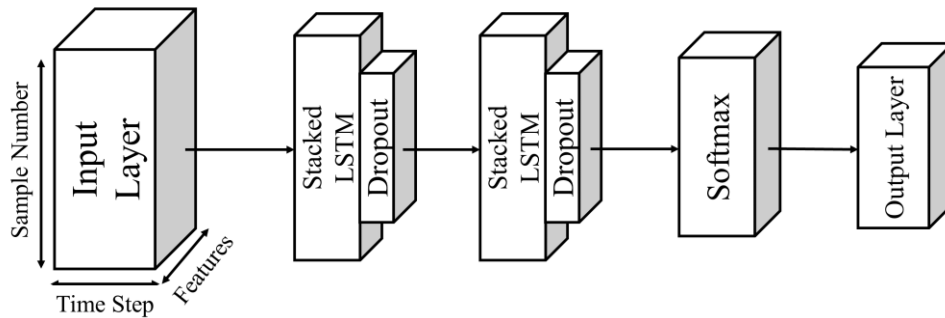


Figure 6 Model architecture

The model implementation process is shown in Figure 7. We firstly divide the dataset into two parts, training (75%) and test (25%) data. The data collected at intersections and segments are all used for model implementation. In the training data, the ratio of different maneuvers, going straight, left turn, right turn, and U-turn is around 97:3:3:1, indicating it is highly imbalanced. Then, Synthetic Minority Over-sampling Technique (SMOTE) is used for re-sampling the training dataset (Chawla et al., 2002). SMOTE creates new minority class instances by interpolating between several minority class instances that lie together and has been utilized widely in the transportation fields. Hyperparameters tuning is conducted based on the training data, the trained model is evaluated on the test dataset. The fine-tuned model is generated according to the selected metrics, including precision, recall, and F1-score:

- Precision: the probability that ‘A’ maneuvers in classification result are true ‘A’.
- Recall: the probability that all ‘A’ maneuvers in ground truth are classified as ‘A’.
- F1-score: F1-score is estimated based on precision and recall (13), a high F1-score usually indicates the model’s good performance.

$$F_1 = 2 \times (Precision \times Recall) / (Precision + Recall) \quad (13)$$

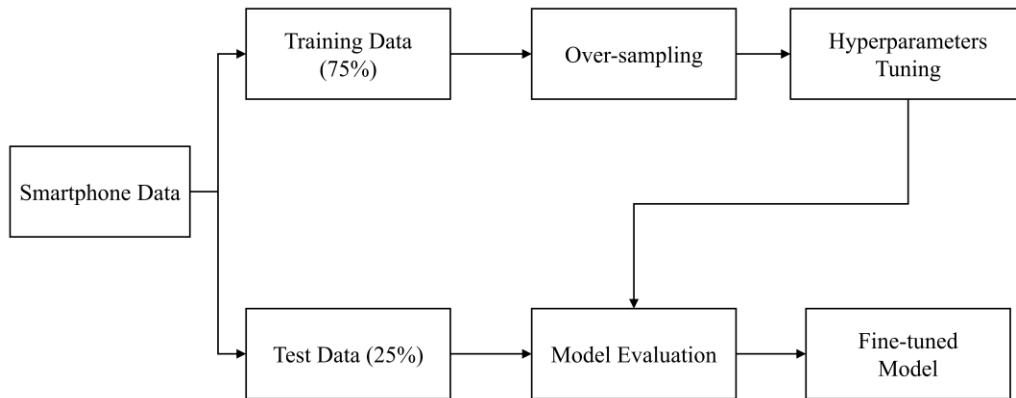


Figure 7 Model implementation process

The proposed model is implemented based on Keras (Chollet, 2015) using NVIDIA GTX 1050 4G GPU. One of the most crucial steps for applying the deep learning model is hyperparameters tuning. Several common hyperparameters are selected (Table 3), including learning rate, batch size, epoch, etc. Besides, the optimization function is also considered as a hyperparameter, three common optimization functions are selected, including Adam, RMSprop, and Stochastic Gradient Descent (SGD).

Table 3 Hyperparameters tuning results

Name	Range	Value
Learning rate	0.1, 0.01, 0.001, 0.0001	0.001
Batch size	100, 200, 500, 1000, 2000	500
Epoch number	10, 50, 100, 200, 500	100
Optimizer function	Adam, RMSprop, SGD	RMSprop
Dropout rate	0.3, 0.5, 0.8	0.3
LSTM unit number	128, 64, 32, 16	64

3.4.3 Results

After tuning hyperparameters and sensor selection. The combination of a_x , a_y , and g_z achieves the best result on the test data set (Table 4). The model has an average precision of 0.97, recall of 0.98, and F1-score of 0.98, The results indicate that the model has good performance over different drivers, vehicles, smartphones, and road facilities. Besides, the confusion matrix of the results is shown in Figure 8. There are 0.11%, 0.14%, and 0.05% of going straight maneuvers that were wrongly detected as the left turn, right turn, and U-turn. For the turning maneuvers, the model wrongly detects 3.04% left turn, 1.76% of right turn, and 5.6% U-turn as going straight. The confusion matrix indicates that the model might fail to distinguish between going straight and turning movements. However, it would not fail to distinguish between different turning movements.

Table 4 Experiment results

Maneuver	Precision	Recall	F1-score
Going straight	0.99	0.99	0.99
Left Turn	0.97	0.97	0.97
Right Turn	0.97	0.98	0.97
U-turn	0.96	0.99	0.97

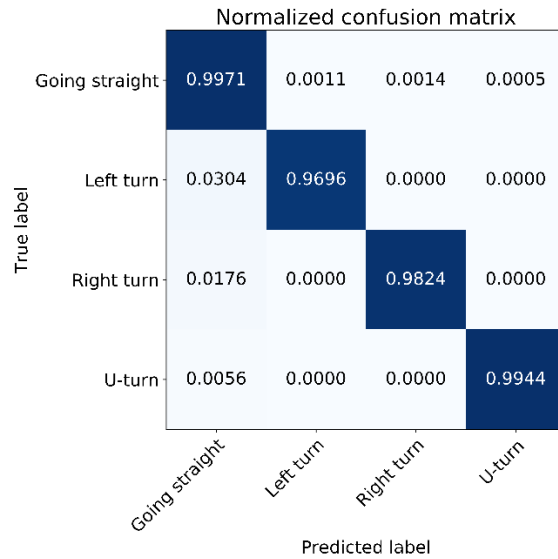


Figure 8 Normalized confusion matrix

To illustrate the model’s capability of distinguishing vehicle maneuvers, t-Distributed Stochastic Neighbor Embedding (t-SNE) is utilized based on the outputs of the model. t-SNE is an advanced technique for high-dimensional data visualization (Maaten and Hinton, 2008). It maps multi-dimensional data to two or more dimensions suitable for human observation. t-SNEs of the raw features and the extracted features from the last layer of the model are shown in Figure 9 (a) and Figure 9 (b), respectively. The vehicle maneuvers are extremely difficult to distinguish in the raw data, their patterns are critically tangled together (Figure 9 (a)). However, Figure 9 (b) indicates the features extracted by our model successfully divide four maneuvers.

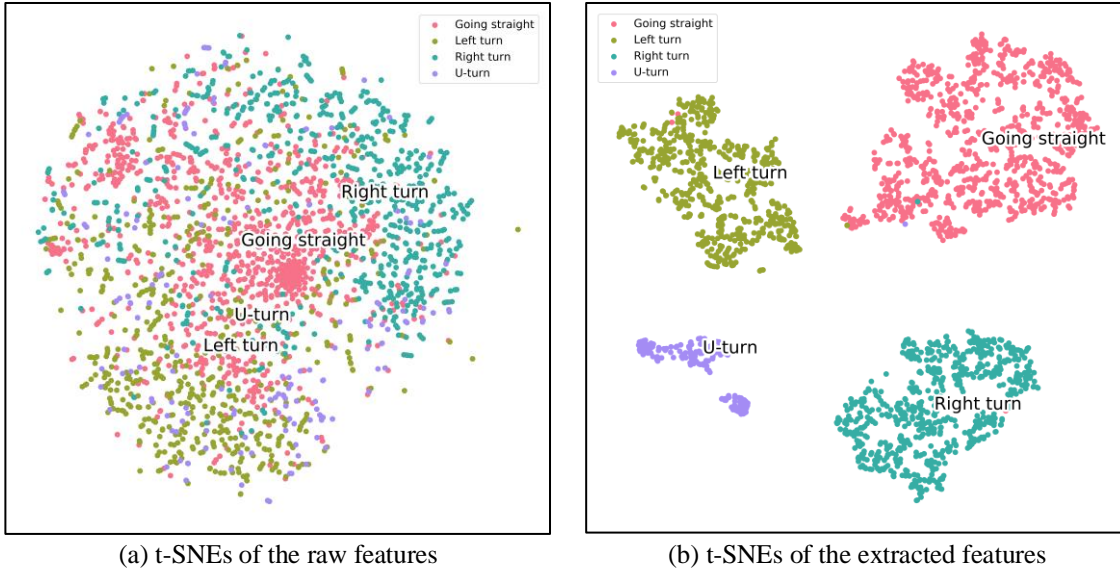


Figure 9 t-SNEs of raw features and extracted features

3.4.4 Model Comparison

To compare the results of the stacked-LSTM with existing studies, we summarize our model and the previous studies from different aspects, such as the selected sensors, statistical features, and the average accuracy (Table 5). Previous studies use multiple sensors and different statistical features to get relatively accurate results. In general, the more features a model has, the results could become more accurate. However, our proposed stacked-LSTM uses only two sensors with three axes, reaches the highest accuracy without any other statistical features.

Table 5 Results from existing studies and our method

Literature	Sensor axes	Statistical features	Accuracy
Johnson and Trivedi (2011)	$a_x, a_y,$ g_x, g_y, o_x	max, range	91.87%
Yu et al. (2017a)	$a_x, a_y,$ $o_x, o_y, time$	range, std, mean, max, min	95.36%
Wang et al. (2016)	a_x, a_y, a_z g_x, g_y, g_z	max	90%
Stacked-LSTM	a_x, a_y, g_z	Do not apply	98%

Note: o_x is the orientation on X-axis, std is the standard deviation, the accuracy is the average accuracy.

The comparison of our results with the previous studies shows that our model reaches the-state-of-the-art performance in terms of average accuracy. It is also worthwhile to investigate the performance of other benchmark methods on the same dataset. Several benchmark machine learning methods are selected, including LightGBM (Ke et al., 2017), random forest, and KNN. To ensure fairness and comparability, they are all fine-tuned based on the same dataset as the stacked-LSTM. Specifically, the benchmark models take the 90-dimensional data as the input and generate the maneuver label as the output. In terms of the hyperparameters for the benchmark methods. The number of leaves and max depth are selected for LightGBM. The number of trees, max number of features, and max depth are selected for random forest. The number of neighbors and Minkowski metric are selected for KNN. Kernel functions, C value, and gamma are used for SVM. The results from different models are summarized in Figure 10. F-1 score is used as the metric since it combines precision and recall. Besides, going straight maneuver is ignored since it is relatively easy to detect and has very high accuracy among all the models. According to Figure 10, the proposed stacked-LSTM outperforms the other models with a significantly higher F-1 score for the three maneuvers. LightGBM has the best results among the benchmark methods, whereas SVM cannot accurately detect different maneuvers. The main reason for the good performance of the stacked-LSTM is the capability of learning from time-series data and capturing long-term dependencies. Besides, the computation time for processing one input is also compared between different methods (Figure 11). LSTM has the minimal processing time (1.976 milliseconds), which is much less than the sensor update frequency (10Hz). The fast processing time of LSTM makes it possible to deploy the model in real-time and generate the results every 0.1 seconds.



Figure 10 Results from different models

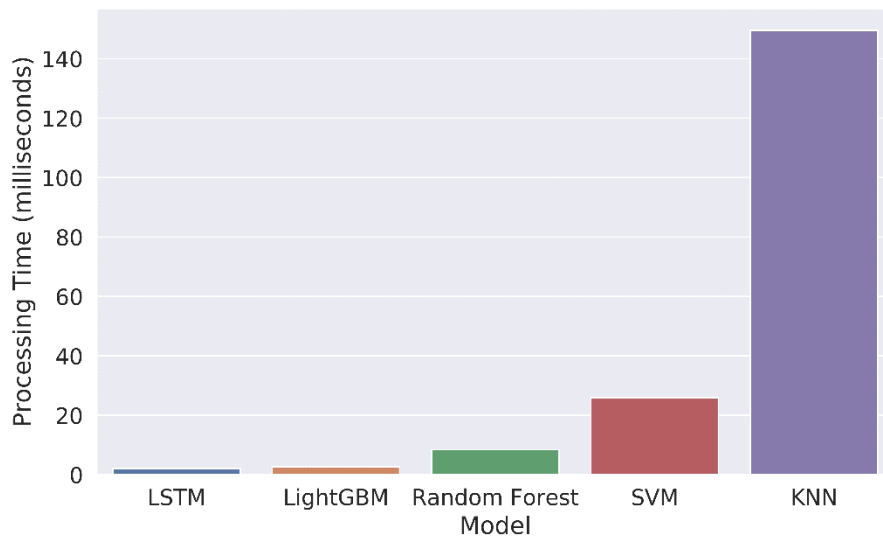


Figure 11 Processing time of different models

3.4.5 Model Transferability

The performance of the proposed model is well-illustrated in the previous sections. However, the model transferability still requires additional investigation. The basic assumption for transferring a model is that different drivers and locations have different characteristics but share common features. The model may have poor performance when it receives completely new data. However, the training process on the new data should be much easier. Existing studies (Ferreira et al., 2017; Ouyang et al., 2019; Saleh et al., 2017; Singh et al., 2017) only test the proposed models on the self-built dataset without evaluating them on new data. Thus, we collected additional data for two new drivers at new locations with around 30,000 records. The performance of the proposed model and the benchmark methods are shown in Figure 12. LSTM still has the best overall results. Among the benchmark methods, LightGBM and random forest have the best results while KNN has the worst results.

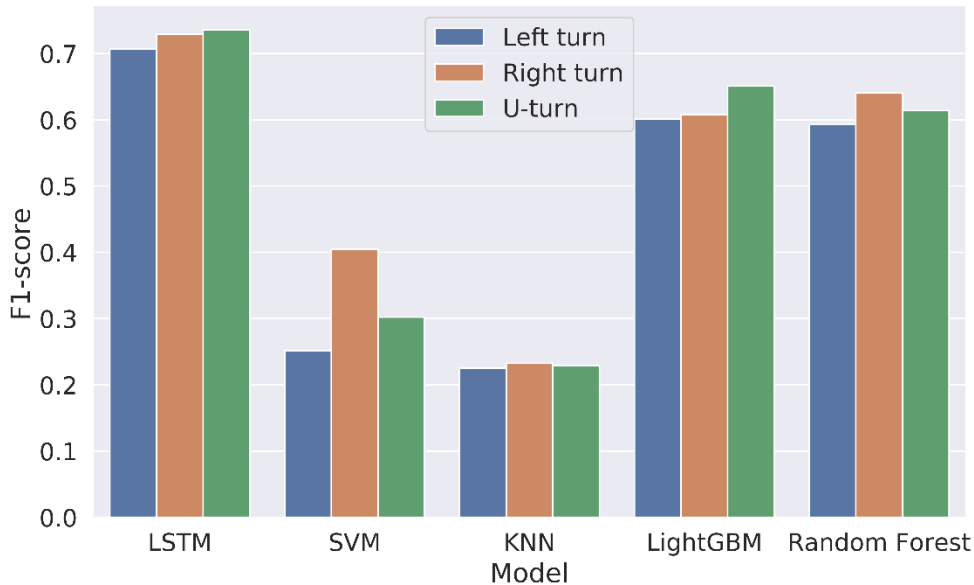


Figure 12 Model results on the new dataset

The performance of the proposed LSTM on the new dataset can be gradually improved by re-training it. Since the model was already trained based on the original dataset, the re-training process only requires much fewer data. The model's average F1-score on the original and new data are shown in Figure 13. The average F1-score is estimated by averaging the F1-scores of the left turn, right turn, and U-turn. The new data ratio is the portion of the new data used to re-train the model. Figure 13 indicates that with only 5% of the new data, the average F1-score can be improved from 0.72 to 0.81. To reach a 0.95 F1-score, the model only requires 35% of the new data. Besides, the model performance on the original data is not impaired during this process.

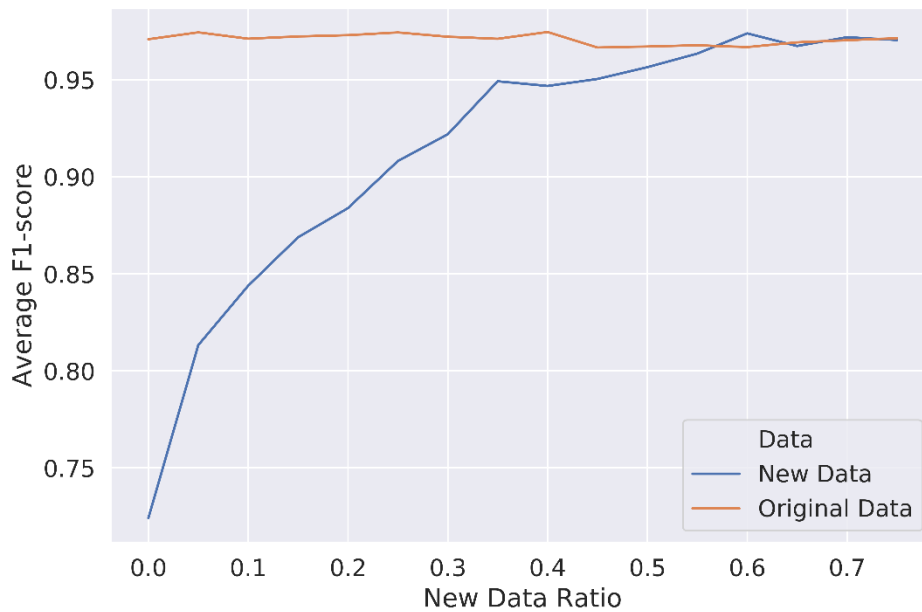


Figure 13 Model results on different new data ratios

3.5 Conclusions

This study proposed a real-time vehicle maneuvers detection system based on smartphone sensors. First, a coordination system reorientation method was applied to free the smartphone's position. Second, data from the gyroscope and accelerometer were used as input after filtering. Third, SMOTE was applied as an over-sampling technique to solve the imbalanced data problem. In the end, a stacked-LSTM model was built to detect vehicle maneuvers. The model is fine-tuned based on the real-world driving data collected from extensive experiments at different locations. According to the experimental results, the main contributions of this study can be summarized as follows: First, instead of using various sensors and extracting additional statistical features, this study uses three filtered sensor data, which decreases the computation complexity and saves battery consumption. Second, the proposed stacked-LSTM model accurately identifies different vehicle maneuvers with the average precision of 0.97, recall of 0.98, and F1-score of 0.98, which reaches the state-of-the-art performance compared with other studies and benchmark models. Third, the model has robust performance in different locations, which is validated based on the data from intersections, segments, and local roads. Forth, the model can be easily transferred to different drivers or locations.

Overall, this study illustrates the feasibility of using stacked-LSTM to identify vehicle maneuvers in real-time. The simply reoriented and filtered sensor data are desirable features for the implementation of the model. The results from this study can be applied to broadcast vehicle maneuvers through the smartphone in a connected vehicle system to prevent crashes. However, there are still several limitations in the current study. First, the performance of stacked-LSTM

can be improved by adding more layers or trying more combinations of different hyperparameters. Second, different vehicle maneuvers may indicate the safety condition of the driver, the differentiation between normal, aggressive, and dangerous driving can be introduced in future research. Third, the model was developed on a relatively limited dataset, which may impair the model's generality while applying it to other drivers. With the development of CV and mobile sensing technologies, massive real-world driving data will become available in the future. The model can be further improved based on these novel data.

CHAPTER 4: DRIVING MANEUVERS DETECTION USING SEMI-SUPERVISED LSTM AND SMARTPHONE SENSORS

4.1 Introduction

In 2018, 42,881 vehicles were involved in single- and two-vehicle fatal crashes in the USA (National Highway Traffic Safety Administration, 2019). Among them, traveling straight accounts for 63.8%, turning left accounts for 7.4%, turning right accounts for 0.9%, and U-turn accounts for 0.4%. Driving maneuvers detection is an important component for proactive traffic safety management systems, such as collision warning systems (Abdel-Aty et al., 2020; Liu et al., 2016c), abnormal driving detection systems (Johnson and Trivedi, 2011; Yu et al., 2017a), etc. Besides, with the development of the connected vehicle systems, the information of driving maneuvers could be shared and transmitted efficiently, which could provide road users with useful assistance. To detect driving maneuvers, driving data are usually collected using a device (e.g. smartphone, OBD, etc.). A method is then applied to learn from the data and detect driving maneuvers accordingly.

Various devices were used in the previous studies, such as OBDs (Meseguer et al., 2013), cameras (Li and Busso, 2016), radars (Doshi and Trivedi, 2010), etc. The OBD can be plugged into the OBD port of the vehicle and collect data from the vehicle's electronic control unit. Some OBDs also have Bluetooth functions which enable them to be connected with other mobile devices. Meseguer et al. (2013) utilized an OBD-II to collect driving information such as speed, acceleration, engine revolutions per minute, throttle position, and vehicle's geographic position. The collected data were then used to detect different driving styles, such as normal, quiet, and aggressive. A camera is another device that was utilized in existing studies. Li and Busso (2016)

used two cameras to capture both the driver's face and road conditions. The driver's head pose and eye movement were extracted from the interior camera. Road optical flow and intensity were obtained through the exterior camera. The authors identified different mirror-checking actions (e.g., left, rear, and right mirror-checking) by using the driver's head pose and eye movement. The authors then introduced mirror-checking actions as new features to identify several driving maneuvers (e.g., turns, lane switch, and straight). The use of these new features significantly improved the results of driving maneuvers detection. Doshi and Trivedi (2010) used the adaptive cruise control radar system to extract the average time gap between the ego vehicle and the leading (preceding) vehicle. This information was then used to identify aggressive drivers from normal drivers. Recently, due to the increasing sensing capacity and high market penetration rate of smartphones, they are gaining attention from both academia and industry. Compared with other devices, the smartphone has its advantages for detecting driving maneuvers. First, a common smartphone has various built-in sensors. For example, the Android system is already equipped with an accelerometer, gyroscope, GPS, and other sensors since 2010 (Li et al., 2021a). Nowadays, almost everyone has a smartphone. However, the use of OBDs or radars usually requires an additional installation, which increases the cost of obtaining data. Second, the smartphone is more flexible and robust. Other devices, such as cameras, usually need to have fixed positions. We can put a smartphone in arbitrary positions and still get most of the sensor data. In addition, the high-speed cellular and Wi-Fi networks enable the possibility of using smartphones for vehicle-to-vehicle and vehicle-to-pedestrian applications. Various studies have used smartphones for driving maneuvers identification. Accelerometer and gyroscope are the two most commonly used sensors. For example, Chen et al. (2015) identified lane change and turn

maneuvers using a smartphone's gyroscope and accelerometer. The readings from the gyroscope were first used to detect lane change and turn maneuvers. The authors then derived the vehicle's horizontal displacement using the gyroscope and accelerometer. This information could help distinguish between lane change, turn, and driving on a curvy road. Singh et al. (2017) utilized data from the smartphone's accelerometer and gyroscope to detect several driving maneuvers. Specifically, the accelerometer was used to detect braking maneuver. The gyroscope was used to distinguish between aggressive turning and normal turning maneuvers. In addition, Ferreira et al. (2017) compared the performance of different sensors (i.e., accelerometer, linear accelerometer, magnetometer, and gyroscope) on detecting driving maneuvers, such as aggressive braking, turning, lane change, etc. The authors indicated that accelerometer and gyroscope are the most suitable sensors to detect driving maneuvers.

Table 6 briefly summarizes the methods used in the existing studies for identifying driving maneuvers. Driving maneuvers detection is a multi-class classification problem with driving data as input and maneuvers as output. Two types of methods were utilized in the existing studies, rule-based and machine learning methods. The first method detects driving maneuvers by using pre-defined rules. For example, Chen et al. (2015) detected lane-changing and turning maneuvers using the smartphone's gyroscope and accelerometer. By setting certain thresholds, the bumps of the gyroscope's readings could be detected. A bump was then classified as a turning or lane-change maneuver based on the values of the accelerometer. The rule-based methods are easy to implement and interpret. However, the readings of sensors may vary from different vehicles, devices, and drivers. The accuracy of the rule-based methods would be impacted if they are implemented under different conditions. Therefore, machine learning

methods became popular because of the high flexibility and accuracy. For example, the SVM was used by Bhoraskar et al. (2012) to detect braking maneuver and bumps in the road using the data from a smartphone's accelerometer, GPS, and magnetometer. Singh et al. (2017) implemented DTW to detect braking, left turn, and right turn maneuvers. Specifically, the accelerometer was used to detect braking maneuver, while the gravity and gyroscope were used to detect turning maneuvers. Ferreira et al. (2017) compared the performance of different machine learning methods on driver maneuver detection (e.g. aggressive braking, acceleration, left turn, right turn). Random forest outperformed other methods (e.g. SVM, Bayesian network) in terms of AUC values. Recently, with the development of computer hardware and the availability of massive data, deep learning methods are widely used in transportation fields. Deep learning is one type of machine learning methods with the ability to learn from high-dimensional data. In terms of driving maneuver identification. Yu et al. (2017a) implemented a fully connected neural network to detect driving maneuvers (e.g. such as weaving, swerving, sudden braking, etc.) with a smartphone's accelerometer and orientation sensors. Results indicated that the neural network outperformed SVM with higher classification accuracy. Bejani and Ghatee (2020) used the CNN to identify normal and dangerous drivers based on the smartphone's accelerometer. Results suggested that CNN accurately identified different driving styles with the help of regularization terms. Carvalho et al. (2017) explored the application of RNNs to driver maneuver identification. Different from CNN, RNN was designed to learn from time-series data and achieved promising results. The authors compared the performance of different RNNs, LSTM, Gated Recurrent Unit (GRU), and standard RNN. Results suggested that LSTM and

GRU had similar results and could accurately detect different driving maneuvers, which are better than the standard RNN.

Table 6 Summary of driving maneuvers detection in existing studies

Author	Year	Device	Method	Results
Meseguer et al. (2013)	2013	OBD-II	Neural Network	Normal, quiet, and aggressive driving styles
Li and Busso (2016)	2016	Camera	Random undersampling boost	Turning and lane change
Chen et al. (2015)	2015	Smartphone	Rule-based method	Lane change and turning
Singh et al. (2017)	2017	Smartphone	DTW	Braking, left turn, and right turn maneuvers
Ferreira et al. (2017)	2017	Smartphone	Random forest	Aggressive braking, acceleration, left turn, right turn
Yu et al. (2017a)	2017	Smartphone	Fully connected neural network	Weaving, swerving, sudden braking, fast U-turn, and sideslipping
Bejani and Ghatee (2020)	2020	Smartphone	CNN	Normal and dangerous drivers
Carvalho et al. (2017)	2017	Smartphone	RNNs	Aggressive and non-aggressive drivers

Almost all the previous studies treat driving maneuver detection as a supervised classification problem, which requires the labels of driving maneuvers. Recently, collecting traffic-related data (e.g., smartphone data, video data, detector data, etc.) becomes much easier due to the development of technologies and infrastructure. However, the process of labeling data is usually tedious and time-consuming, which constrains the implementation of supervised learning methods. In addition, the unused unlabeled data also have great potential and should not be ignored. Therefore, it is necessary to propose novel methods (e.g., unsupervised learning and semi-supervised learning) which rely less on the data labels and could learn from the massive unlabeled data. Unlabeled data usually do not have its explanation, such as label, tag, class, or name for the data. For example, a driving record (e.g., speed, acceleration, etc.) without indication of the vehicle is turning. Some studies investigated the possibilities of utilizing

unlabeled data for driving maneuver identification. Two types of methods were commonly used unsupervised and semi-supervised learning. Unsupervised learning usually does not require labels for the data. It works by finding the similarity between different drivers and then cluster them. Fugiglando et al. (2019) designed an unsupervised technique that clustered drivers in different groups using CAN bus data. principal component analysis was used to generated new features from the CAN bus data (e.g., brake pedal pressure, gas pedal position, speed, etc.). Then, a K-means clustering method was implemented to cluster the drivers into different groups with the generated features. The results of clustering could be used to reflect different driving styles. However, the generated clusters usually do not have ground truth to validate, which may question the accuracy of the approach. Mahajan et al. (2020) proposed an unsupervised learning approach to label lane change maneuvers based on a trajectory dataset. The authors designed a density-based clustering method to label lane-changing and lane-keep maneuvers. An LSTM model was then developed using the generated labels and data. However, due to the lack of ground truth, the authors were not able to comprehensively evaluate the results. Moreover, the authors only detect lane-changing without differentiating right lane-changing, left lane-changing, and other maneuvers. This study aims to accurately identify various driving maneuvers (e.g., left turn, right turn, left lane change, etc.). Unsupervised learning methods work well for driving style detection or driving maneuvers detection with a few maneuvers according to the literature review, which are not suitable for the problems in this study. Another method for using unlabeled data is Semi-supervised Learning (SSL). SSL was designed to learn from the massive unlabeled data with only a small amount of labeled data. SSL was widely used in the computer vision and natural language processing areas. In terms of driving maneuver detection, Wang et

al. (2017b) designed a semi-supervised SVM to detect different driving styles, including normal and aggressive. To learn from the unlabeled data, the authors used the smoothness assumption, which assumes nearby data points should have similar detection results. Liu et al. (2016a) used a semi-supervised learning method to detect driver distraction with data from cameras. Drivers' eye and head movements were extracted as features from the video data. Similarly, the authors introduced a manifold regularization penalty term to learn from the unlabeled data, which penalized a large difference in the predicted class labels with respect to two nearby data points. In summary, these studies successfully illustrate the feasibility of using SSL for driving maneuver-related studies. First, the performance of SSL methods could converge to the performance of the supervised learning methods as the ratio of labeled data increases. SSL methods work by learning from both labeled and unlabeled data. The performance of SSL methods could be improved with additional labeled data. Second, the SSL methods could make use of the unlabeled data and mitigate the costs of labeling data. However, these studies were all developed based on the smoothness assumption, which may not be true in some cases. Moreover, time-dependency was not considered during these studies. Finally, various maneuvers need to be detected, such as right turn, left turn, U-turn, etc.

This study aims to mitigate the current research gaps based on the proposed methods. First, a semi-supervised learning algorithm is designed to learn from the unlabeled data. Second, LSTM is used as the deep learning model to detect driving maneuvers due to its ability to learn time-series data. Third, extensive experiments are conducted to collect real-world driving data from different drivers, smartphones, vehicles, and locations. In the end, the proposed semi-supervised LSTM is compared with a variety of methods to illustrate its performance.

4.2 Research Approach

4.2.1 Sensor Data Collection

This study aims to identify driving maneuvers using smartphone sensors with a semi-supervised LSTM. The research workflow is shown in Figure 14. The sensors' data (accelerometer and gyroscope) are first collected by an Android application. After data preparation, the proposed semi-supervised learning algorithm is used to train the LSTM. Six driving maneuvers are detected, including going straight, left turn, right turn, U-turn, left lane change, and right lane change.

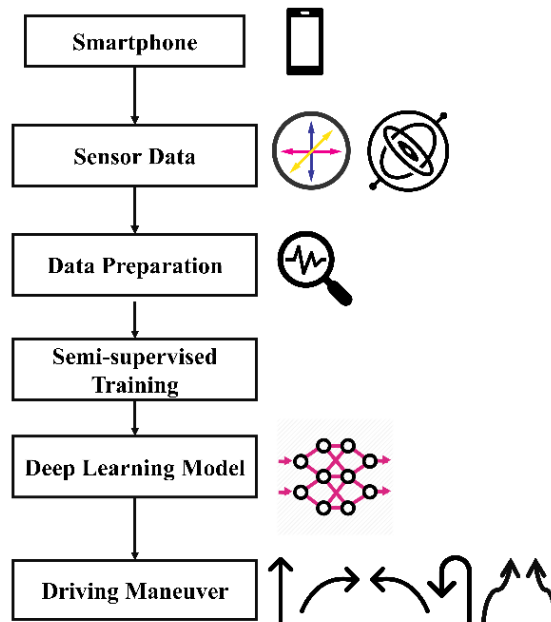


Figure 14 Research workflow

The smartphone sensor data are collected through an Android application. Figure 15 shows the application interface. Different sensor data can be collected from this application,

including gyroscope, accelerometer, GPS, barometer, and magnetometer. The sensor update frequency can be customized with the unit as milliseconds. The data are saved automatically as a CSV file. In addition, the application can efficiently label the sensor data by pressing the earphone's button.

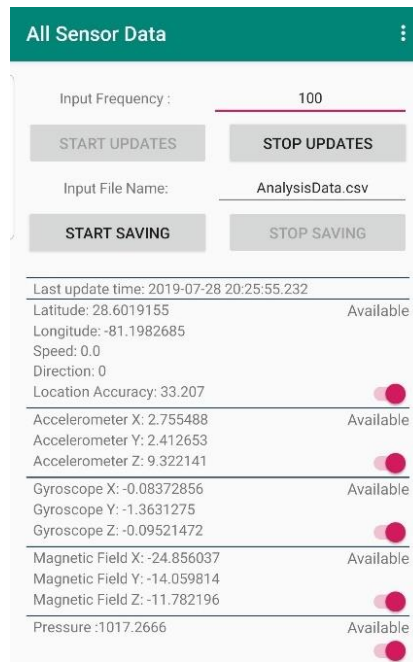


Figure 15 Application interface

Various smartphone sensors were used by previous studies (Islam and Abdel-Aty, 2021a, b; Johnson and Trivedi, 2011; Li et al., 2020a; Saiprasert et al., 2013; Singh et al., 2017), including GPS, accelerometer, gyroscope, etc. This study selects the accelerometer and gyroscope due to their popularity and the consideration of battery consumption. Both the accelerometer and gyroscope are motion sensors, which monitor the motion of a smartphone from different perspectives. Specifically, the accelerometer returns acceleration rates for the three coordinate axes in m/s^2 . Almost every Android-powered handset and tablet has an

accelerometer, and it uses about 10 times less power than the other motion sensors (Google, 2020). The gyroscope measures the rate of rotation in rad/s around a device's X-, Y-, and Z-axis. Rotation is positive in the counter-clockwise direction (Google, 2020). The coordinate systems of the accelerometer and gyroscope are illustrated in Figure 16.

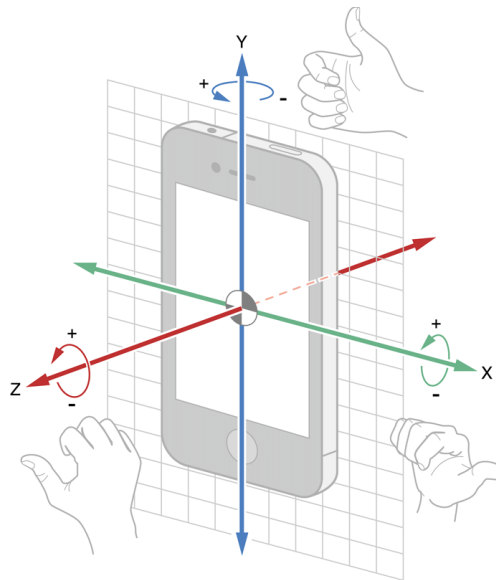


Figure 16 Sensor coordinate systems (Apple, 2020)

4.2.2 Data Preparation

This study used a two-step data preparation procedure, including data filtering and coordinate system alignment. Raw sensor data usually contain a lot of noise due to hardware sensitivity. Existing studies usually utilized digital filters before further analysis. Moving average filter is simple while powerful digital filter that was widely used in previous studies (Johnson and Trivedi, 2011; Liu et al., 2016c; Saiprasert et al., 2013; Singh et al., 2017). Moving average filter operates by averaging several points from the input data to produce each point in the output data with Equation (31).

$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i + j] \quad (14)$$

Where $x[]$ is the input data (e.g. sensor readings from the accelerometer), $y[]$ is the output data, and M is the number of points in the average (i.e., time step in this study). Besides, The smartphone's coordinate system changes with its position. To better measure the motion of a vehicle through smartphone sensors, their coordinate systems need to be aligned (Figure 17). One approach for coordinate system alignment is by using the rotation matrix and vehicle direction (Bhoraskar et al., 2012). The smartphone's coordinate system (X_P, Y_P, Z_P) is first converted to the world's coordinate system using the rotation matrix from the Android API. Then, the world's coordinate system is transformed into the vehicle's coordinate system (X_V, Y_V, Z_V) using vehicle direction and bearing obtained from the smartphone's GPS. For the detailed mathematical equations, the reader is referred to Bhoraskar et al. (2012).

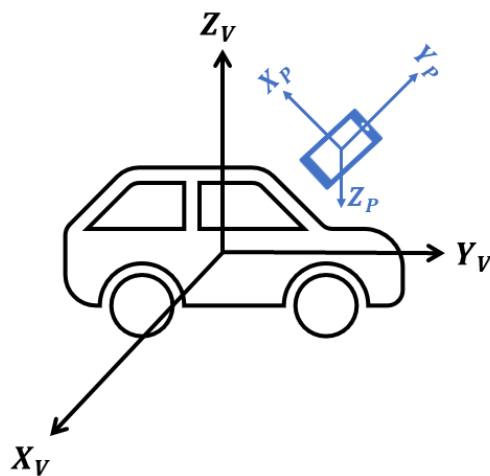


Figure 17 Coordinate system alignment

4.2.3 Semi-supervised Learning

The semi-supervised learning algorithm proposed in this study is designed based on the concepts of multi-view learning (Xu et al., 2013a) and co-training (Blum and Mitchell, 1998). During each co-training process, multiple views are generated from the labeled data, while each view is used to train a weak model. Each model is then applied to the unlabeled data. Some of the unlabeled samples are selected with the results from weak models as their labels using certain criteria. These samples and their labels are added to the labeled data. A model has trained on each view of the new labeled data again. Figure 18 shows the procedure of the semi-supervised learning algorithm proposed by this study. The sensor data are prepared as $N \times 3$ matrices (a_x, a_y, g_z) , where N is the number of the samples, a_x is the X-axis of the accelerometer, a_y is the Y-axis of the accelerometer, and g_z is the Z-axis of the gyroscope. Three views are obtained from the original data, (a_x, a_y) , (a_x, g_z) , and (a_y, g_z) . The proposed algorithm has three inputs, L as the labeled data, U as the unlabeled data, and k represents the difference between models. First, each model from (M_1, M_2, M_3) is trained separately on each view of the labeled data (L). For example, M_1 is trained on (a_x, a_y) . Then, for each sample x in U , p_i is the classified label (driving maneuver) of M_i on x . If all three results are the same (l), x and l are included in L , while x is removed from U . Otherwise, the value of k increases by 1. The algorithm keeps iterating until the unlabeled data are empty or k is smaller than 10.

Algorithm 1 Semi-supervised Learning Algorithm

Input: L, U, k
Output: M_1, M_2, M_3

```
1: repeat
2:    $k = 0$ 
3:   for  $i \in (1, 2, 3)$  do
4:     train  $M_i$  on  $L$ 
5:   end for
6:   for  $x \in U$  do
7:     for  $i \in (1, 2, 3)$  do
8:        $p_i \leftarrow M_i(x)$ 
9:     end for
10:    if  $p_1 = p_2 = p_3 = l$  then
11:      add  $x$  and  $l$  to  $L$ 
12:      remove  $x$  from  $U$ 
13:    else
14:       $k \leftarrow k + 1$ 
15:    end if
16:  end for
17: until  $U$  is empty or  $k < 10$ 
```

Figure 18 Semi-supervised learning algorithm

4.2.4 LSTM

LSTM is one type of RNNs with the ability to learn from long-term dependency.

Traditional RNN usually suffers from the vanishing gradients issue while learning long-sequence data. Hochreiter and Schmidhuber (1997) proposed LSTM to solve this problem by adding additional gates to the RNN cell. LSTM was widely used by similar studies and achieved outstanding results (Carvalho et al., 2017; Zhang et al., 2020c). A standard LSTM cell has three important gates, input, forget, and output gate. These gates control the information flow and decide when to forget or remember certain information. The structure of the LSTM cell at time t is shown in Figure 19, with its equations (Christopher, 2015) shown from Equations (15) to (21).

$$f_t = \sigma(W_f \cdot [h_{t-1}; x_t] + b_f) \quad (15)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}; x_t] + b_i) \quad (16)$$

$$\tilde{c}_t = \tanh(W_r \cdot [h_{t-1}; w_t] + b_c) \quad (17)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (18)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}; x_t] + b_o) \quad (19)$$

$$h_t = o_t \odot \tanh(c_t) \quad (20)$$

$$y_t = W_y h_{t-1} + b_y \quad (21)$$

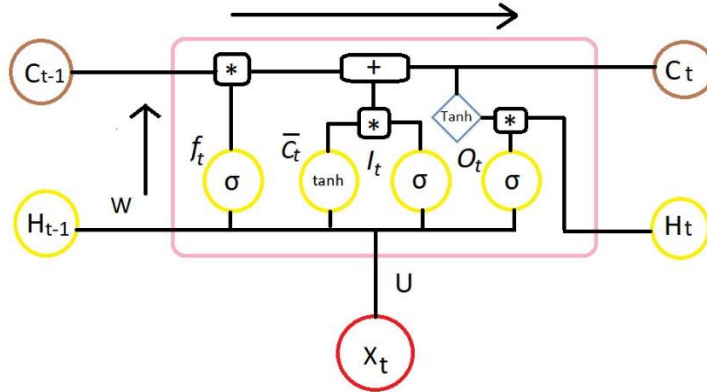


Figure 19 LSTM cell structure (Sanjeevi, 2018)

Where x_t is the input, y_t is the output, i_t is the input gate, f_t is the forget gate, o_t is the output gate. W represents weight matrices, for example, W_i denotes the weight matrix from the input gate to the input, σ is the logistic sigmoid function, c is the context vector, h is the hidden state, and \odot indicates the elementwise product of the vectors.

The network architecture (semi-supervised LSTM) used in this study is shown in Figure 20. Three identical LSTM models are developed based on three views of the labeled data. The LSTM model contains two LSTM layers with regularization terms. Dropout layers are used to prevent overfitting. Softmax is used to generate the output since driving maneuver detection is a multi-class classification problem. The “label” is the model’s results on each view of the unlabeled data.

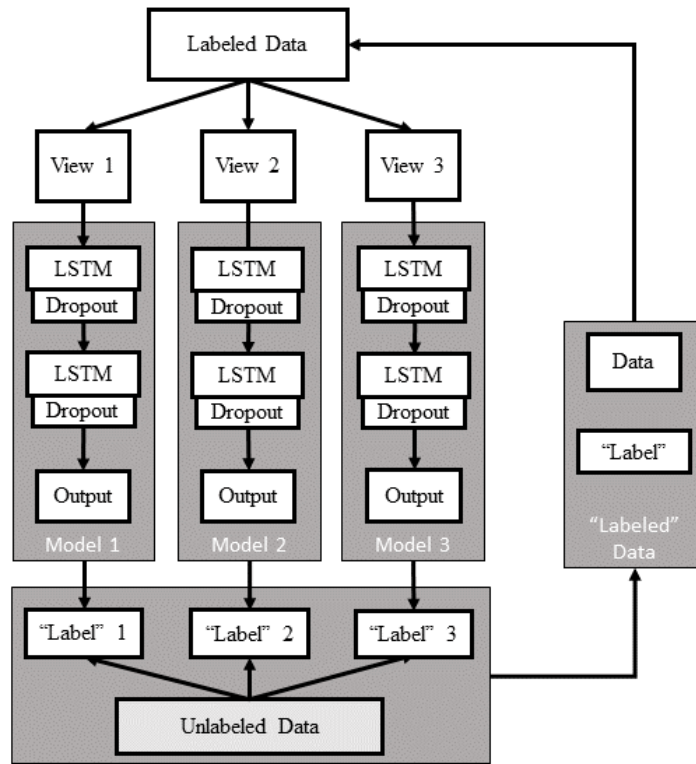


Figure 20 Network architecture

4.3 Experiment and Results

4.3.1 Experimental Design

Driving data from four volunteers were collected during their daily commute, they could put their smartphones in any position. To increase the generality of the method, data were collected by different smartphones (i.e., Samsung Note10, Samsung Galaxy S8, and Samsung Galaxy S10) with different vehicles (i.e., Toyota Corolla, Nissan Altima, and Chevrolet Malibu). Data from the accelerometer and gyroscope were collected with an update frequency of 10 Hz. The experiment ranged from May 2019 to October 2019. In total, in this research we had collected 156,463 records, with 144,461 going straight, 4,699 left turn, 4,729 right turn, 1,506 U-turn, 506 left lane change, and 562 right lane change. Most of the turning maneuvers happened at

intersections, while most of the lane change and going straight happened at road segments. The data were mostly collected near the main campus of the University of Central Florida. Figure 21 shows the spatial distribution of the collected data, which includes different intersections and road segments.

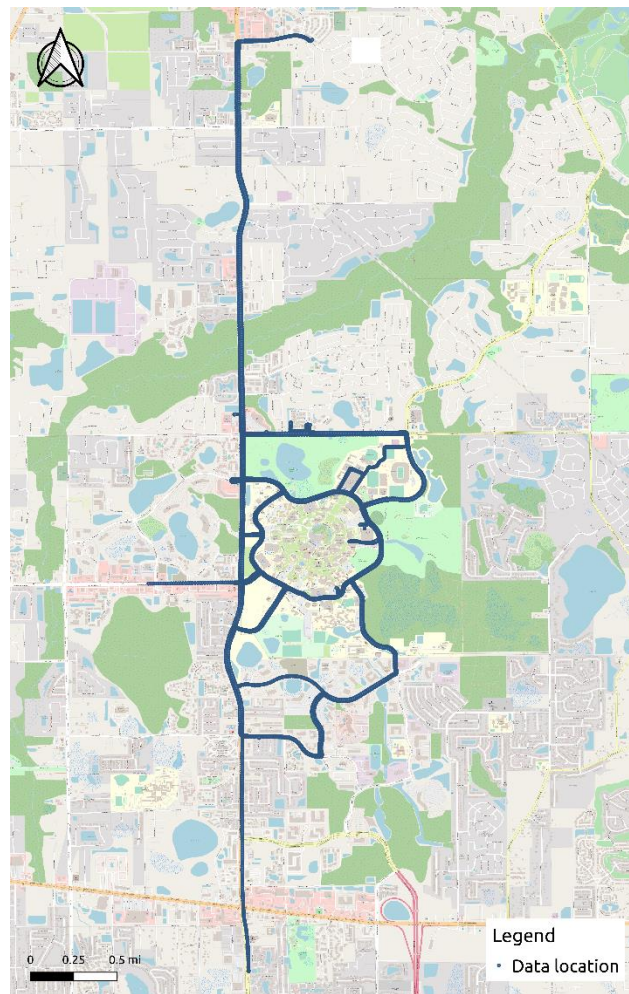


Figure 21 Data collection routes and intersections' locations

The collected data were smoothed by a moving average filter with a time-step as 30 (3s). Data of $T - 30$ time-step were stacked to identify the driving maneuver at time T . The experiment was designed as follows: The prepared data were randomly divided into training

(70%) and test (30%) data ten times. The model's results on the training and test data were averaged over the ten splits. Training data were then divided into two parts, one retained the labels as labeled data, another one discarded the labels as unlabeled data. The ratio of labeled data to training data was selected from 10%, 20%, 30%, 40%, 50%. In addition, the dataset had more going straight events than other events. Directly training the proposed model on this imbalanced data may result in poor performance. Therefore, SMOTE was utilized as the over-sampling method (Chawla et al., 2002). Besides, to make sure unlabeled data and test data could reflect the realistic situation, this study only applied SMOTE to the labeled data. After this process, the proposed semi-supervised LSTM was trained on the balanced labeled data and unlabeled data. To avoid over-fitting, the model was evaluated on the test data. For the model's results on test data, the estimated class probability of each model was multiplied. Four metrics were used for model evaluation, precision, recall, F1-score, and AUC. Precision (Equation (22)) and recall (Equation (23)) are estimated based on the classification confusion matrix (Table 7). F1-score (Equation (24)) combines precision and recall, which can evaluate the model more comprehensively. AUC measures the area under the ROC curve, which plots the true positive rates (Equation (23)) and false positive rates (Equation (25)) under different thresholds. Besides, to compare the performance of semi-supervised learning and traditional supervised end-to-end learning. An LSTM model was built with the same structure and parameters as the proposed model. Differently, this LSTM model was trained on the entire training data and evaluated on the test data. The proposed methods were trained and evaluated on a single NVIDIA GeForce RTX 1080 Ti with 11 GB Memory.

Table 7 Confusion matrix

	Actual Positive	Actual Negative
Predicted Positive	True Positive (TP)	False Positive (FP)
Predicted Negative	False Negative (FN)	True Negative (TN)

$$\text{Precision} = \frac{TP}{TP + FP} \quad (22)$$

$$\text{Recall (True Positive Rate)} = \frac{TP}{TP + FN} \quad (23)$$

$$\text{F1 score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \quad (24)$$

$$\text{False Positive Rate} = \frac{FP}{FP + TN} \quad (25)$$

4.3.2 Results

The hyperparameters used in this study are shown in Table 8. The number of LSTM units is the number of hidden units of an LSTM layer, which controls the dimensionality of its output space. The dropout rate is the rate of setting inputs to 0, which aims to prevent overfitting. Batch size and epoch number decided how the model is trained. Specifically, the batch size is the number of samples the model is trained during one time. The epoch number is the number of total training iterations. RMSprop (Tieleman and Hinton, 2012) is used in this study as an optimization function. RMSprop was widely used in transportation-related problems and achieved promising results (Li et al., 2020c). The results of the semi-supervised LSTM on different labeled data ratios are shown in Table 9. The model's performance is increasing with the increase of the labeled data ratio. When the labeled data ratio is 0.5, the model has the best results for identifying driving maneuvers. The model has better performance on turning and going straight maneuver than lane change maneuver. The main reason may be because the lane change maneuver has relatively fewer samples compared with other maneuvers. Besides, the

advantage of using more labeled data is directly reflected by the increase of recall and F1-score. Although the model has high precision when the labeled data ratio is 0.1, it has very low recall and F1-score values. The results from semi-supervised and traditional supervised LSTMs are also compared in Table 10. The semi-supervised LSTM achieved similar results compared with the supervised LSTM. For example, for the maneuver of going straight, the semi-supervised LSTM has the F1-score of 0.997, the supervised LSTM has the F1-score of 0.997. For left lane change, the semi-supervised LSTM has the F1-score of 0.937, the supervised LSTM has the F1-score of 0.933. The results indicated that the proposed semi-supervised LSTM could accurately detect different driving behaviors with a small portion of labeled data.

Table 8 Values of hyperparameters

Name	Value
Number of LSTM units (first layer)	128
Number of LSTM units (second layer)	64
Dropout rate	0.3
Batch size	500
Epoch number	100
Optimization function	RMSprop
Learning rate	0.001

4.3.3 Model Comparison

Three benchmark methods are used to compare with the proposed method, XGBoost, random forest, and CNN. Both XGBoost and random forest are decision tree-based methods. Specifically, XGBoost is a boosting method, each tree in the XGBoost learns from the previous tree and affects the next tree to improve the model's performance. XGBoost was widely used in the machine learning areas due to its scalability and the ability to handling sparse data

(Mahmoud et al., 2021b). Random forest built its decision tree differently, each tree is built independently on a random sample of the data without affecting others. This method was also popular among the existing studies (Ferreira et al., 2017). CNN is a deep learning method, which was originally designed for the computer vision area. The key component of CNN is its convolutional layer, where a filter is applied to obtain features from the input data. Recently, due to its ability to learn from high-dimensional data, CNN became popular in transportation areas and achieved promising results (Bejani and Ghatee, 2020). The selected benchmark methods were also widely used to detect driving maneuvers. For example, Bejani and Ghatee (2020) utilized CNN for driving styles (i.e., dangerous and normal) detection based on data from smartphones. The proposed CNN outperformed other machine learning methods such as SVM, K-NN, etc. Ferreira et al. (2017) implemented random forest to identify driving maneuvers (e.g., aggressive braking, aggressive acceleration, aggressive left turn, etc.) using smartphone sensors. Random forest was compared with other methods (i.e., SVM, Multi-layer Perceptron, etc.) and reached the best classification accuracy in terms of AUC. Mousa et al. (2018) compared the performance of different tree-based methods for detecting lane-change maneuvers. Specifically, decision tree, random forest, gradient boosting, and XGBoost were used in this study. Results suggested that XGBoost reached higher accuracy compared with tree-based methods.

The selected benchmark methods are trained and tested on the same dataset as the proposed method, using supervised learning. In terms of hyperparameters, three hyperparameters are tuned for XGBoost, including the number of trees (200), maximum depth of the tree (20), and learning rate (0.1). Two hyperparameters are tuned for the random forest, including the number of trees (200) and the maximum depth of the tree (20). A two layers CNN is

implemented with its filter size (128), kernel size (5), batch size (500), epoch number (100), and learning rate (0.001) are selected accordingly. RMSprop is used as the optimization function for CNN. The results of all four models are shown in Table 10. The proposed semi-supervised LSTM outperforms other methods with higher precision, recall, F1-score, and AUC. In addition, it is difficult for XGBoost and random forest to detect lane change maneuvers. Both of them have very poor performance in detecting lane change. CNN has the best results among the benchmark methods. However, it is still less accurate compared with the proposed semi-supervised LSTM. Figure 22 shows the boxplots of each model's F1-score for the ten random splits. The proposed semi-supervised LSTM has a much more stable performance compared with other models. The results of other models, such as CNN and XGBoost, vary in each split. The results from Figure 22 confirm the promising performance of the semi-supervised LSTM on different data sets.

Table 9 Performance metrics of different labeled data ratios

			Going straight	Left turn	Right turn	U-turn	Left lane change	Right lane change
Labeled Data Ratio	0.1	Precision	0.975	0.935	0.895	0.953	0.968	0.990
		Recall	0.996	0.701	0.775	0.763	0.098	0.160
		F1-score	0.985	0.802	0.831	0.848	0.177	0.275
		AUC	0.865	0.879	0.906	0.902	0.629	0.645
	0.2	Precision	0.985	0.945	0.940	0.969	0.989	0.969
		Recall	0.997	0.864	0.851	0.870	0.345	0.423
		F1-score	0.991	0.902	0.893	0.917	0.510	0.587
		AUC	0.936	0.970	0.962	0.961	0.823	0.888
	0.3	Precision	0.991	0.964	0.952	0.967	0.969	0.973
		Recall	0.997	0.916	0.914	0.890	0.616	0.688
		F1-score	0.994	0.939	0.933	0.927	0.753	0.805
		AUC	0.971	0.991	0.994	0.981	0.965	0.961
	0.4	Precision	0.994	0.975	0.962	0.976	0.963	0.969
		Recall	0.998	0.942	0.943	0.935	0.758	0.852
		F1-score	0.996	0.958	0.953	0.955	0.848	0.906
		AUC	0.987	0.998	0.995	0.994	0.995	0.987
	0.5	Precision	0.996	0.977	0.969	0.993	0.955	0.985
		Recall	0.998	0.953	0.954	0.966	0.921	0.943
		F1-score	0.997	0.965	0.961	0.979	0.937	0.964
		AUC	0.989	0.996	0.994	0.996	0.987	0.990

Table 10 Performance metrics of model comparison

		Going straight	Left turn	Right turn	U-turn	Left lane change	Right lane change
Semi-supervised LSTM	Precision	0.996	0.977	0.969	0.993	0.955	0.985
	Recall	0.998	0.953	0.954	0.966	0.921	0.943
	F1-score	0.997	0.965	0.961	0.979	0.937	0.964
	AUC	0.989	0.996	0.994	0.996	0.987	0.990
Supervised LSTM	Precision	0.998	0.960	0.957	0.966	0.902	0.885
	Recall	0.996	0.978	0.980	0.982	0.968	0.974
	F1-score	0.997	0.969	0.968	0.974	0.933	0.926
	AUC	0.999	0.999	0.999	0.999	0.998	0.999
CNN	Precision	0.997	0.902	0.859	0.874	0.854	0.843
	Recall	0.989	0.952	0.956	0.986	0.962	0.971
	F1-score	0.993	0.926	0.904	0.926	0.904	0.902
	AUC	0.997	0.999	0.998	0.990	0.998	0.990
XGBoost	Precision	0.989	0.737	0.739	0.817	0.416	0.445
	Recall	0.975	0.904	0.901	0.839	0.384	0.324
	F1-score	0.982	0.812	0.812	0.828	0.399	0.374
	AUC	0.976	0.994	0.991	0.995	0.970	0.952
Random forest	Precision	0.994	0.824	0.843	0.920	0.331	0.360
	Recall	0.977	0.942	0.951	0.899	0.747	0.752
	F1-score	0.985	0.879	0.893	0.909	0.459	0.487
	AUC	0.962	0.971	0.974	0.951	0.873	0.875

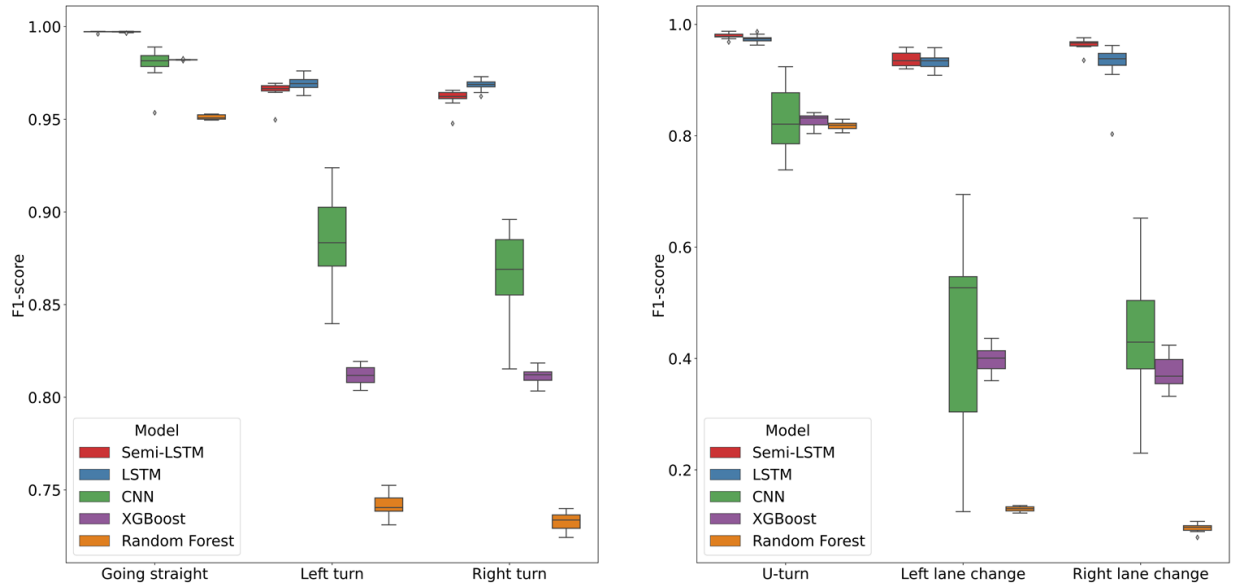


Figure 22 Boxplots of model comparison

4.4 Conclusions

Driving maneuver detection was widely investigated by existing studies, while most of them used supervised methods, which heavily rely on the labeled data. The development of mobile devices (i.e., smartphones) greatly mitigates the cost of collecting driving-related data. However, labeling data is a very time-consuming and tedious process, especially for large-scale data. Similar problems also exist in the computer vision and natural language processing areas, where SSL was proposed to make use of the massive unlabeled data. This study designed a semi-supervised LSTM to identify six driving maneuvers (i.e., going straight, left turn, right turn, U-turn, left lane change, and right lane change) based on the smartphone's accelerometer and gyroscope data. The proposed method was trained and validated on real-world driving data collected by different drivers with various phones, vehicles, and locations. Experimental results suggested that the proposed semi-supervised LSTM could achieve similar results compared with the supervised LSTM, although it was trained on a small portion of the labeled data. Moreover, the proposed method also outperformed other benchmark machine learning methods (e.g. CNN, XGBoost, random forest) in terms of precision, recall, F1-score, and AUC.

The main contribution of this study is the design of a semi-supervised LSTM to detect driving maneuvers. The proposed method requires a very small amount of the labeled data and could achieve similar results with the supervised methods. The costs of labeling data are expected to be significantly reduced by using the proposed method, especially when massive data are available in the future. The developed method has several possible applications in transportation fields. For example, a safety application could be developed based on the proposed method. Potential conflicts between road users could be detected if drivers' maneuvers

are detected at intersections or road segments. The proposed method could also be integrated into a driving performance evaluation system for fleet management. Traffic managers can monitor drivers' maneuvers and coach drivers with abnormal maneuvers. In addition, the smartphone can be used for Vehicle-to-everything (V2X) applications due to its wireless communication ability. The information of driving maneuvers could be transferred to other drivers, pedestrians, and roadside units as long as they are connected. This information can be used to improve efficiency and traffic safety in the future. With the development of connected and autonomous vehicles, massive driving-related data will become available soon. It is predicted that connected vehicles may generate 259 TB of data per day (Cutulenco, 2019). The proposed method can be applied to this kind of data without requiring the typical data labeling process. In the future, this study could be further improved by 1) collecting more data from smartphones with different manufacturers 2) Investigating the model's transferability by testing it with data from different drivers and vehicles.

CHAPTER 5: THE APPLICATION OF NOVEL CONNECTED VEHICLE EMULATED DATA ON REAL-TIME CRASH POTENTIAL PREDICTION FOR ARTERIALS

5.1 Introduction

In 2018, traffic crashes caused 33,654 fatalities in the USA (IIHS, 2019), while 41.6% of them happened on urban arterials. Improving traffic safety, especially for urban arterials, is becoming a major concern for traffic engineers and researchers. Real-time crash potential prediction is one of the effective methods for enhancing traffic safety. Different from the traditional crash frequency prediction based on aggregated data, real-time crash potential prediction aims to predict the crash probability during a short time interval. However, most of the existing studies on real-time crash potential prediction are limited to freeways (Abdel-Aty et al., 2012; Ahmed et al., 2012b; Xu et al., 2013b; Yu and Abdel-Aty, 2014a) rather than urban arterials (Li et al., 2020c; Wang et al., 2015b; Yuan and Abdel-Aty, 2018). Urban arterials usually have more complicated traffic conditions, which require various data sources to predict the real-time crash potential, such as traffic, signal, and weather data. Traditional safety studies usually obtained these data from the fixed infrastructure-based devices, including loop detectors, Bluetooth detectors, microwave sensors, and cameras (Hassan and Abdel-Aty, 2013; Wang et al., 2015a; Zhang, 2020; Zhang et al., 2020b). However, these devices require extra installation costs and regular maintenance. In addition, some devices, such as cameras, are sensitive to lighting, weather conditions, etc. Also, the detection range of these fixed devices is limited to their locations.

Recently, the concept of CV provides a novel way to obtain large-scale vehicle-based data with high flexibility and low cost. Different from the traditional sensor data, the CV data are easy to obtain and maintain. In addition, the data can be collected continuously in a wide range. It is possible to depict the traffic conditions of the whole city with large-scale vehicles. The real deployment of the CV system still needs more time. However, with the help of mobile sensing technologies, it is possible to obtain CV emulated data. CV emulated data can provide similar vehicle information as CV data, such as vehicle location, speed, etc. Some studies have been conducted to explore the application of the CV emulated data in transportation, such as anomaly detection (Kuang et al., 2015; Pang et al., 2013), traffic conditions estimation (Herring et al., 2010; Rahmani et al., 2015), etc. Nevertheless, only a few studies applied this new data source to the traffic safety field. Xie et al. (2013) used taxi data to calculate arterial-level travel speed and introduced speed as an explanatory variable to investigate intersection safety in Shanghai. The authors found higher average speeds along arterials were associated with increased intersection crashes. Similarly, Wang et al. (2015b) examined the relationship between different variables from taxi GPS data and traffic safety for urban arterials during peak and off-peak hours. Higher average speeds were found to be associated with higher crash frequencies during peak periods, but not during off-peak periods. Bao et al. (2019) used the numbers of taxi pick-ups and drop-offs as new variables to predict citywide crash frequency based on deep learning models. Wang et al. (2019b) applied the SVM model to predict crash potential on freeways based on taxi data. Different variables were generated, such as average speed, speed difference ratio, etc. In addition, SVM was found to have better performance than the logistic regression model in terms of sensitivity and AUC values.

Two types of models are available for the real-time crash potential prediction, statistical and machine learning models. Statistical models include logistic regression, Bayesian logistics regression (Ahmed et al., 2014; Ahmed et al., 2012b), etc. These models were usually built on matched-case control data and had certain assumptions. Considering these limitations, the applications of machine learning methods were explored, such as SVM (Yu and Abdel-Aty, 2013), random forest (Lin et al., 2015), etc. The performance of these methods was proven to be better than the statistical methods. For example, Yu and Abdel-Aty (2013) indicated SVM outperformed Bayesian logistic regression in terms of AUC value. Recently, the availability of massive transportation data and the development of computer hardware accelerate the implementation of deep learning. Deep learning is one class of machine learning methods. It was utilized to solve various transportation problems. Moreover, RNN was proven to be especially useful for learning time-series transportation data (Zhang et al., 2020c). Different from the traditional neural network that only maps the current input vector to the output vector, RNN introduces recurrent connections, which allow information to persist. However, one drawback of the RNN is that it cannot capture long-term dependencies (Hochreiter, 1991). Thus, LSTM was invented by Hochreiter and Schmidhuber (1997). LSTM improves the performance of RNN by including memory cells and gates, which preserve the information for a long period. Some new studies have applied LSTM in transportation safety. Yuan et al. (2019) utilized LSTM to predict crash potential in real-time, the authors claimed that their models achieved better sensitivity than the conditional logistic model. Bao et al. (2019) implemented a spatiotemporal convolutional LSTM to predict the citywide crash frequency based on multiple data sources, such as taxi trip data, road network attributes, and land-use features.

There are still several research gaps that need to be filled. First, the existing traffic safety studies with CV emulated data mainly focused on crash frequency analysis (Bao et al., 2019; Wang et al., 2015b; Xie et al., 2013) rather than crash potential prediction (Wang et al., 2019b). It is necessary to investigate the feasibility of using CV emulated data for real-time crash potential, especially for urban arterials. Second, almost all the studies utilized taxis for traffic safety analysis. More efforts need to be done on other types of vehicles. Previous studies successfully detected traffic anomalies based on bus data (Kong et al., 2017; Zhang et al., 2019). It is promising to investigate the applications of bus data on traffic safety.

Different from the other vehicles such as taxis, buses have their unique advantages. For example, a bus usually has a fixed route and schedule. The trajectory of the bus is more stable and cannot be affected by the drivers' preferences and characteristics. Moreover, a bus usually runs around the urban area, which can depict the city-wide traffic conditions extensively. Third, the studied periods are restricted in the existing studies, which are not favorable for realistic applications. For example, Wang et al. (2015b) only analyzed the crashes during peak and off-peak periods. Wang et al. (2019b) only selected the periods from 5 to 10 minutes (and 10 to 15 minutes) before the events (crash and non-crash cases) to conduct analysis. Although the authors claimed the non-crash cases were randomly picked, the information of the unselected non-crash cases is still important to the model. It is necessary to build a generic model based on the entire data set.

The main objective of this study is to explore the feasibility of utilizing novel CV emulated data to predict the real-time crash potential for arterial road segments. Two major urban arterials in Orlando, FL are selected to conduct a case study. Various speed-related

variables are generated from the CV emulated data. In addition, different data preparation, map-matching techniques will be explored. A deep learning methodology is proposed to predict the real-time crash potential with variables from the CV emulated data. The proposed method will be compared with different benchmark methods based on various evaluation metrics.

5.2 Data Preparation

5.2.1 Data Description

Two data sets are used in this study, CV emulated data and crash data. The CV emulated data have three parts: vehicle trajectory data, routes data, and stops data. All of them are obtained by the data collection API from the DoubleMap. The API requests are made with an HTTP GET request, and the data are returned in JSON format. There are around 300 LYNX[®] buses and 50 UCF shuttles in the vehicle trajectory data. The data are collected in real-time and updated every three seconds. The geographical location, heading, stop, and ID of the vehicle can be obtained from this data (Table 11). In addition, the decimal points of 90% latitude and longitude data range from 5 to 15, which can achieve up to the millimeter accuracy. The routes data provide the information of the active LYNX[®] routes, such as the route ID, route name, and the route stops. The stops data are complementary to the routes data and provide the geographical location, ID, and name of each stop.

Table 11 Data description

Name	Description
ID	Unique integer for each vehicle.
Name	A text name for the vehicle.
Latitude	Latitude for the current position of the vehicle.
Longitude	Longitude for the current position of the vehicle.
Heading	The direction of movement, in degrees (0-360).
Route	The ID of the route that this vehicle is currently assigned to.
Laststop	The ID of the stop that this vehicle was most recently at, or its current stop.
Lastupdate	The UNIX timestamp of the last GPS update from the vehicle.
Source	LYNX [®] or UCF.

The vehicle trajectory data have an extremely high update frequency, with almost 3,000,000 records are generating every day. Therefore, it is difficult to conduct safety analysis for the entire city. Two urban arterials (Figure 23), are selected in this study considering road geometry, vehicle density, crash frequency, etc. The experimental time range is from April 2019 to July 2019. The target of this study is the road segment, which is defined as the road facility between two consecutive intersections with a certain direction (Figure 24). Each road segment has its unique ID. In total, the selected area has 126 arterial segments, 66 intersections, and 15 bus routes. The crash data of the selected area are obtained from the S4A system. S4A provides detailed information for each crash event, including crash time, location, severity, and type.

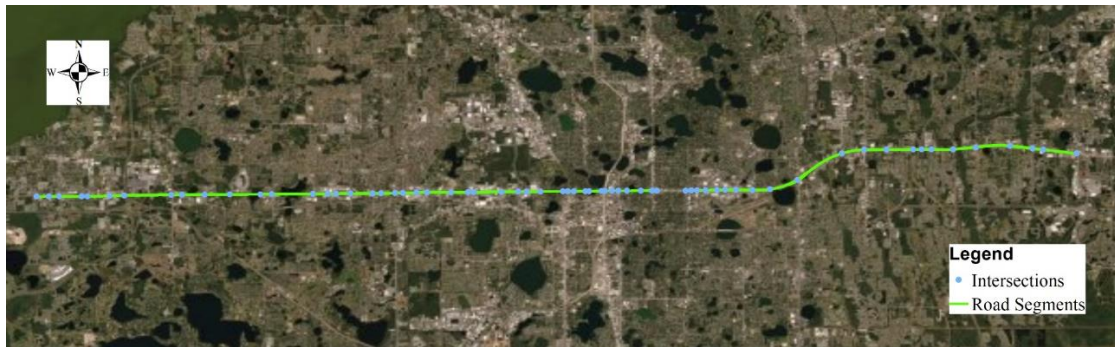


Figure 23 Research area

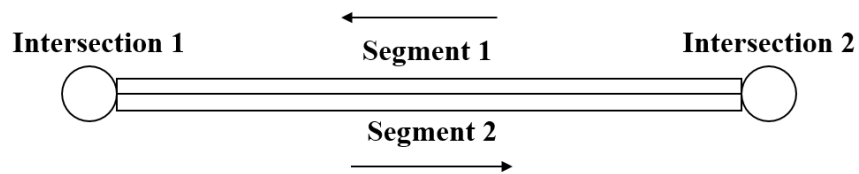


Figure 24 Illustration of segments and intersections

5.2.2 Data Preprocessing

The procedure of the CV emulated data preparation is shown in Figure 25. One common drawback for using CV emulated data is the instability of the trajectory. Previous studies spent extensive time correcting the trajectory data based on different map-matching methods. The benefit of using bus data is that it usually has a fixed route. In addition, we can obtain direction information to help the map matching process.

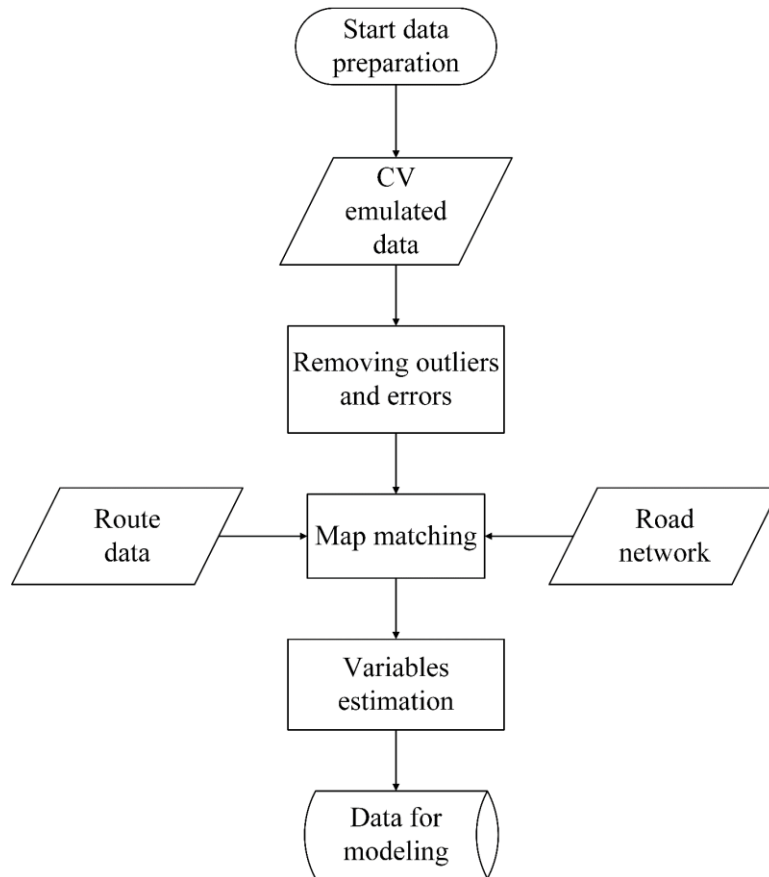


Figure 25 Procedure of data preparation

In terms of data preparation, first, the outliers and errors are removed from the original data, such as the GPS points outside of the studied area and the duplicated records. The obtained data are matched to the road segments for further analysis (Figure 26). *ArcMap* 10.6 is used for map matching. Since buses have fixed routes, the segment map is generated using a combination of bus routes and the existing road network. This process greatly mitigates the computational costs compared with matching mixed trajectory data to a large road network. Using the spatial join function provided by *ArcMap*, the CV emulated data are converted to the data of road segments. Moreover, the vehicle directions are also considered to improve the map matching accuracy.

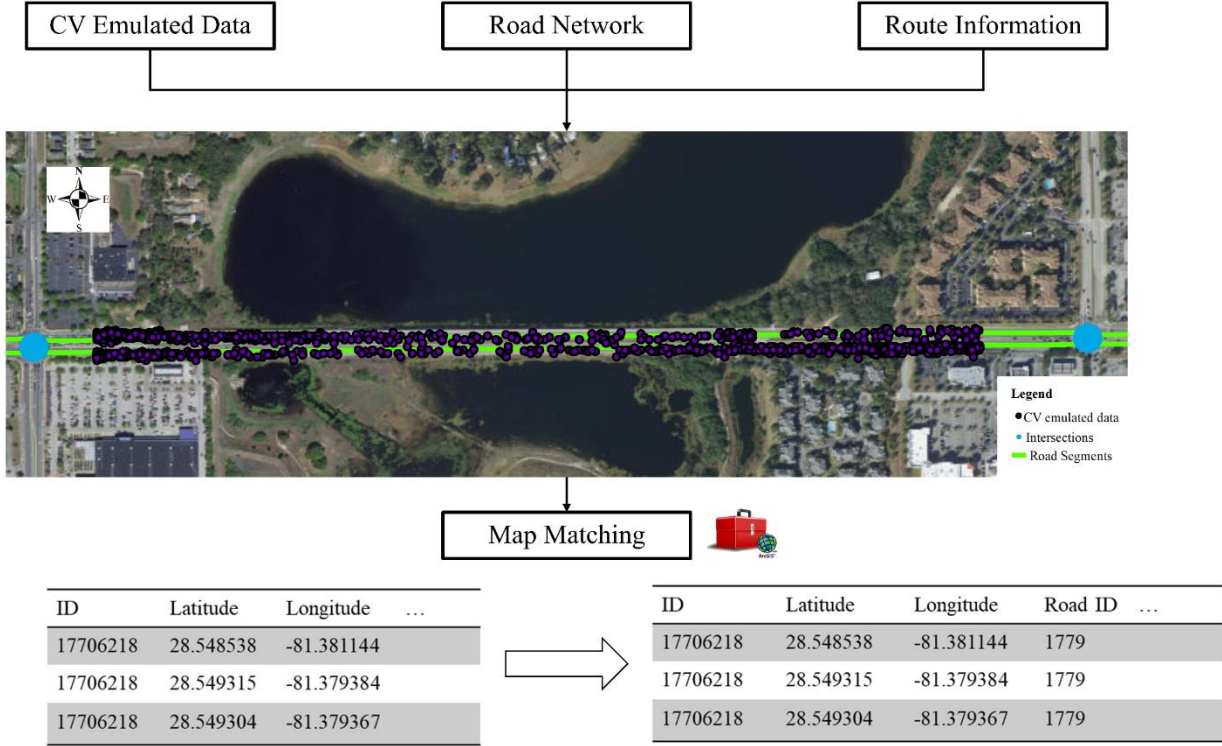


Figure 26 Map matching process

Second, the speed of each vehicle is estimated based on two continuous GPS points. For a vehicle i , its speed at the time t_1 is estimated as:

$$V_i = \frac{\text{distance}(GPS_{t_1}, GPS_{t_2})}{t_1 - t_2} \quad (26)$$

Where GPS_{t_1} is its location at the time t_1 , and GPS_{t_2} is its location at the time t_2 . The distance d between two locations is estimated based on the Haversine formula (Veness, 2019):

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_1) \cdot \cos(\varphi_2) \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (27)$$

$$c = 2 \cdot \arctan2(\sqrt{a}, \sqrt{1-a}) \quad (28)$$

$$d = R \cdot c \quad (29)$$

Where φ and λ are latitude and longitude in radians, R is the earth's radius. After we get the vehicle speed, several variables for the road segments can be generated, including average speed, speed standard deviation, 85th percentile speed, 50th percentile speed, and 15th percentile speed. All these variables are generated in the 1-minute time interval. The descriptive table of the variables is shown in Table 12.

Table 12 Descriptive Statistics

Variable Name	Description	Mean	Std	Min	Max
Avg_speed	Average speed of the segment	15.91	11.70	0.00	50.85
Std_speed	Speed standard deviation of the segment	9.37	4.71	0.00	28.42
Per85	85th percentile speed of the segment	21.88	8.80	0.00	49.03
Per50	50th percentile speed of the segment	15.33	9.49	0.00	47.64
Per15	15th percentile speed of the segment	8.58	8.40	0.00	47.64

Note: all the values are in the unit of mph

In addition, to illustrate the results from the speed estimation, the average bus speed of one road segment is shown in Figure 27. The speed reaches two low points at 8:00 and 17:00, which correspond to the peak hours of daily commutes. The bus usually has a constant low speed from 8:00 to 17:00. The speed gradually increases after 17:00 and reaches the highest point around midnight.

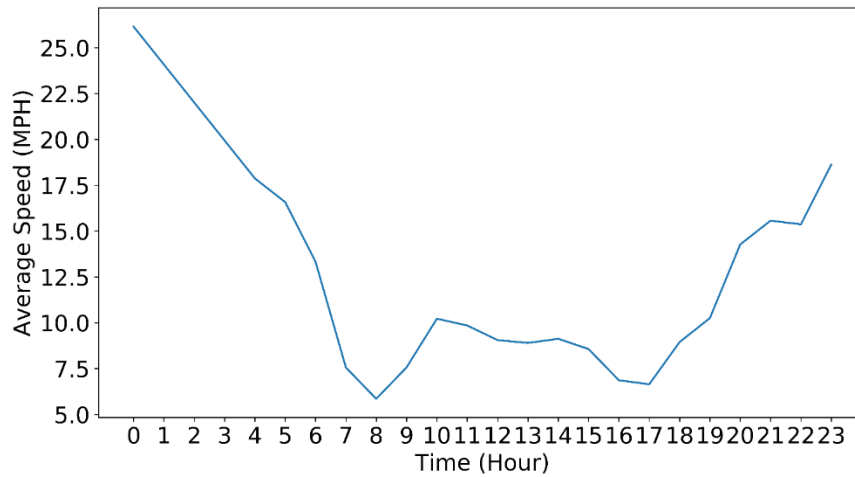


Figure 27 Average segment speed

To prepare the crash data, the crashes which are within the intersection influence area (within 250 feet of intersection) are excluded from the original data set. In addition, alcohol and drug-related crashes are also removed. After these processes, there are 180 crashes in total. These crashes are then matched to the corresponding segments according to the geographical locations.

5.2.3 Crash Labelling and Features Selection

This study aims to utilize the features from the CV emulated data to predict real-time crash potential. After the vehicle features and crashes are matched to the corresponding road segments, data of each segment are prepared in chronological order. The time range of the data is from 4/16/2020 to 7/2/2020. Therefore, each road segment has roughly 112,320 records. This study aims to predict the crash potential in the next 5-10 minutes. As Figure 28 shows, if a crash happened at 00:10, the traffic safety statuses from 00:00 to 00:05 are labeled as '1', indicating that a crash will occur in the next 5-10 minutes from the current time. Otherwise, the traffic

safety status is labeled as ‘0’. In addition, since a crash event usually causes turbulence to the traffic conditions, data within 120 minutes after a crash are removed.

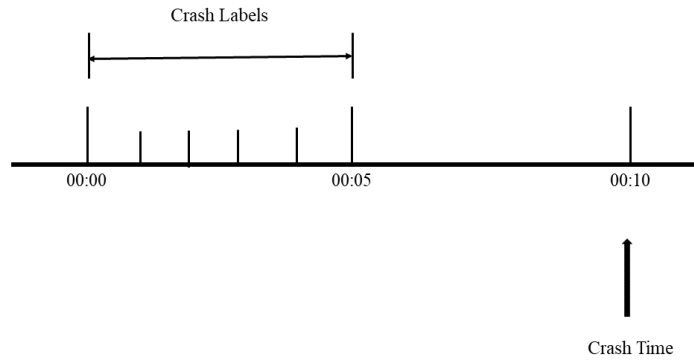


Figure 28 Crash labeling process

The variables used by this study were commonly applied by the previous traffic safety studies. The average speed and speed standard deviation were proven to be closely related to crash potential (Yu and Abdel-Aty, 2013; Yu and Abdel-Aty, 2014b). Park and Saccomanno (2006) indicated it is necessary to include 85th percentile speed for crash risk evaluation. Muchuruza and Mussa (2005) pointed out the missing of low-speed vehicles among the existing studies. The authors introduced 15th percentile speed as a new variable for traffic safety analysis. In addition, acceleration may also be utilized as an indicator for crashes. Stipancic et al. (2018a) introduced hard braking and accelerating events as surrogate safety measures to investigate their correlations with historical collisions. However, acceleration is usually calculated in high frequency and this study is using the 1-min interval. In the end, average speed, speed standard deviation, 85th percentile speed, 50th percentile speed, and 15th percentile speed are selected in this study. To illustrate the feasibility of using these variables, the variables’ importance is

estimated based on the Extra-tree classifier (Geurts et al., 2006). Extra-tree classifier fits several randomized decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting (Pedregosa et al., 2011). The variable importance of Table 12 is shown in Figure 29. The average speed is the most important variable for crash potential prediction. While the 50th percentile speed is the second important variable. The speed standard deviation has similar importance as the 15th percentile speed. Based on the results from Figure 29 and existing studies, this study utilizes all these five variables to predict crash potential.

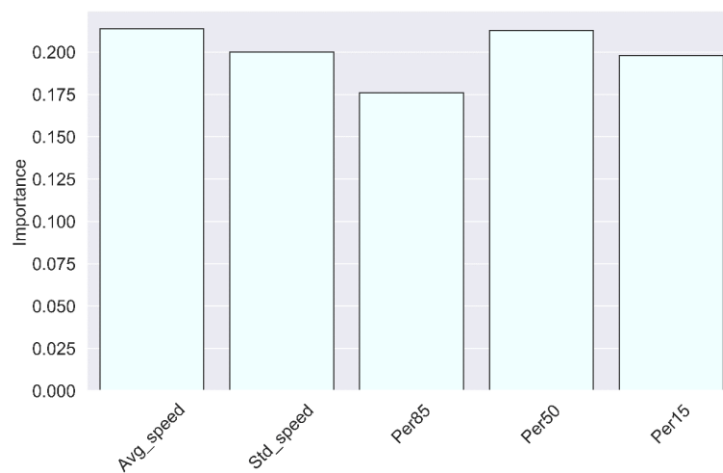


Figure 29 Variable importance

5.3 Methodologies

Recently, with the rapid development of deep learning methods, the RNN has proven to be an efficient method for processing time-series data. RNN allows cyclical or recurrent connections, which can map the whole historical input data to each output and allow information to persist (Tian and Pan, 2015). However, RNN is usually less effective in learning long-term

data sequences due to the problem of vanishing gradient. Therefore, the LSTM neural network was developed to address this problem (Hochreiter and Schmidhuber, 1997), which introduced the memory cells to determine when to forget certain information. An LSTM network is composed of the input layer, the hidden layers, and the output layer. The main characteristics of the LSTM are the memory cells in its hidden layers, which contain memory blocks rather than traditional neuron nodes. Each memory block has several self-connected memory cells and three multiplicative units, input, output, and forget gates. These gates provide continuous analogues of write, read and reset operations on the cells. The structure of an LSTM unit at each time step is shown in Figure 30.

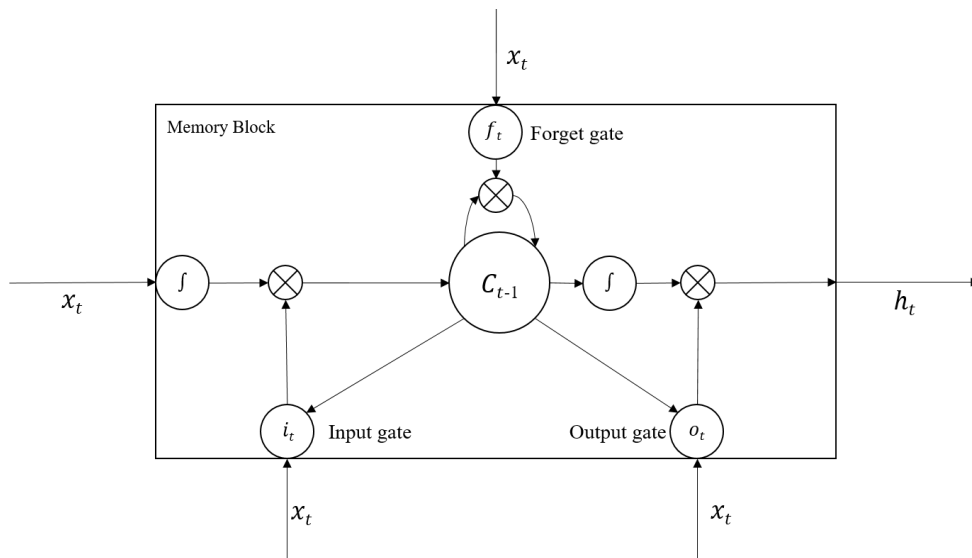


Figure 30 LSTM unit structure (Graves et al., 2013)

The LSTM generates a mapping from an input sequence vectors $X = (X_1, X_2, \dots, X_N)$ to an output probability vector by calculating the network unit activations using the following equations (Graves et al., 2013), iterated from $t = 1$ to N :

$$\mathbf{i}_t = \sigma(\mathbf{W}_{ix}\mathbf{X}_t + \mathbf{W}_{ih}\mathbf{h}_{t-1} + \mathbf{W}_{ic}\mathbf{c}_{t-1} + \mathbf{b}_i) \quad (30)$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{fx}\mathbf{X}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f) \quad (31)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{ox}\mathbf{X}_t + \mathbf{W}_{oh}\mathbf{h}_{t-1} + \mathbf{W}_{oc}\mathbf{c}_t + \mathbf{b}_o) \quad (32)$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tanh(\mathbf{W}_{cx}\mathbf{X}_t + \mathbf{W}_{ch}\mathbf{h}_{t-1} + \mathbf{b}_c) \quad (33)$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \quad (34)$$

$$\mathbf{y}_t = \mathbf{W}_{yh}\mathbf{h}_{t-1} + \mathbf{b}_y \quad (35)$$

Where W represents weight matrices, for example, W_{ix} denotes the weight matrix from the input gate to the input, σ is the logistic sigmoid function, and \odot indicates the elementwise product of the vectors. The forget gate f_t controls the extent to which the previous step memory cell is forgotten, the input gate i_t determines how much to update for each unit, and the output gate o_t controls the exposure of the internal memory state. Since the values of all the gating variables vary for each time step, the model could learn how to represent information over multiple time steps.

5.4 Experimental Design and Results

5.4.1 Experimental Design

The procedure of the experiment is shown in Figure 31. The data are first divided into training (75%) and test (25%). As crashes are rare events, the data are highly imbalanced. The ratio of non-crash events to crash events is around 9,000:1 in the training data. Directly applying the model to the training data will result in a model with bad performance. Therefore, the crash resampling method should be implemented before training the model. Matched-case control is a traditional under-sampling method, which creates non-crash events based on several control

factors from the crash events, such as the time of the day, day of the week, etc. However, the information of the non-crash events may be lost during this process. SMOTE is a popular data resampling method (Chawla et al., 2002) and has been widely applied to similar problems. SMOTE is a data over-sampling method that synthesizes new minority samples between existing minority samples. The advantage of the SMOTE is the number of majority samples will not change, which retains their information completely. After the training data is resampled by SMOTE, the proposed model is trained on the balanced data. Finally, the model is evaluated on the test data with different metrics.

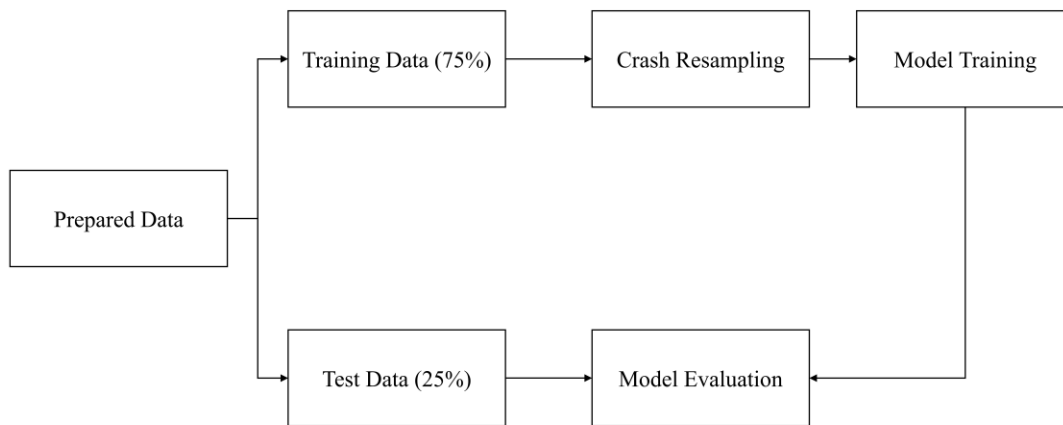


Figure 31 Experiment procedures

In terms of the evaluation metrics, sensitivity, false alarm rate, and AUC value are applied. Sensitivity and false alarm rate are estimated according to the confusion matrix (Table 13) and Equations (36)-(37). High sensitivity means the model can predict most of the crash cases correctly, while a low false alarm rate indicates the model predicts most non-crash cases correctly. AUC is a comprehensive metric that is estimated based on the Receiver Operating Characteristic (ROC) curve. The ROC curve plots true positive rate and false positive rate, which

can be estimated from Equations (36) and (37). AUC measures the entire two-dimensional area underneath the entire ROC curve from (0, 0) to (1, 1).

Table 13 Confusion matrix

	True Crash	True Non-Crash
Predicted Crash	True Positive (TP)	False Positive (FP)
Predicted Non-Crash	False Negative (FN)	True Negative (TN)

$$\text{True Postive Rate (Sensitivity)} = \frac{TP}{TP + FN} \quad (36)$$

$$\text{False Postive Rate (False Alarm Rate)} = \frac{FP}{TN + FP} \quad (37)$$

The LSTM used in this study is designed to take three-dimensional data as the input and generate a single output. Due to the special requirement of the LSTM, the input data are reshaped with a dimension as (sample size * time step * feature number). Specifically, the data from the previous T-10 minutes are stacked together as the input to predict the crash potential at the time T. The dropout layer is introduced to prevent over-fitting. The sigmoid function is used to generate the output since the crash prediction is a binary classification problem. The network architecture of the neural network is shown in Figure 32.

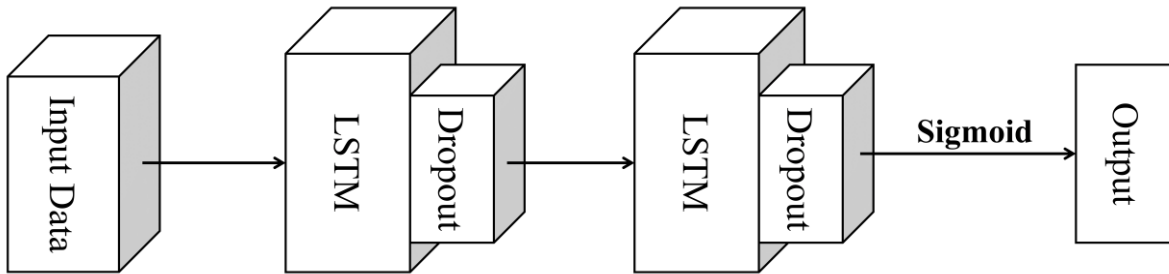


Figure 32 Network architecture

5.4.2 Experimental Results

The performance of the deep learning models is heavily dependent on the selections of different hyperparameters. Six common hyperparameters (Table 14) are tuned within the given range, including LSTM unit number, dropout rate, optimization function, learning rate, epoch number, and batch size.

Table 14 Hyperparameters tuning

Name	Range	Value
LSTM unit number	128, 64, 32, 16	32
Dropout Rate	0.3, 0.4, 0.5	0.5
Optimization Function	SGD, Adam, RMSprop	RMSprop
Learning rate	0.1, 0.01, 0.001, 0.0001	0.001
Batch size	1000, 5000, 10000	5000
Epoch number	50, 100, 150, 200	50

After training the model, the experiment results of the model on the test data are shown in Table 15. The model could successfully predict 79% of the total crashes, with a relatively low false alarm rate of 21%. The results illustrate the feasibility of using CV emulated data to predict crash potential in real-time.

Table 15 Model results

Name	Value
AUC	0.79
Sensitivity	0.79
False Alarm Rate	0.21

In addition, the t-SNEs of the raw data and extracted features are shown in Figure 33 (a) and Figure 33 (b), respectively. t-SNEs is an effective data visualization technique, especially for

high-dimensional data (Maaten and Hinton, 2008). The extracted features are obtained through the last layer of the LSTM. Crash and non-crash events are extremely tangled together and hard to distinguish in the raw data. On the contrary, the features extracted by the LSTM (Figure 33 (b)) successfully separate these two events, which illustrates the good performance of the model.

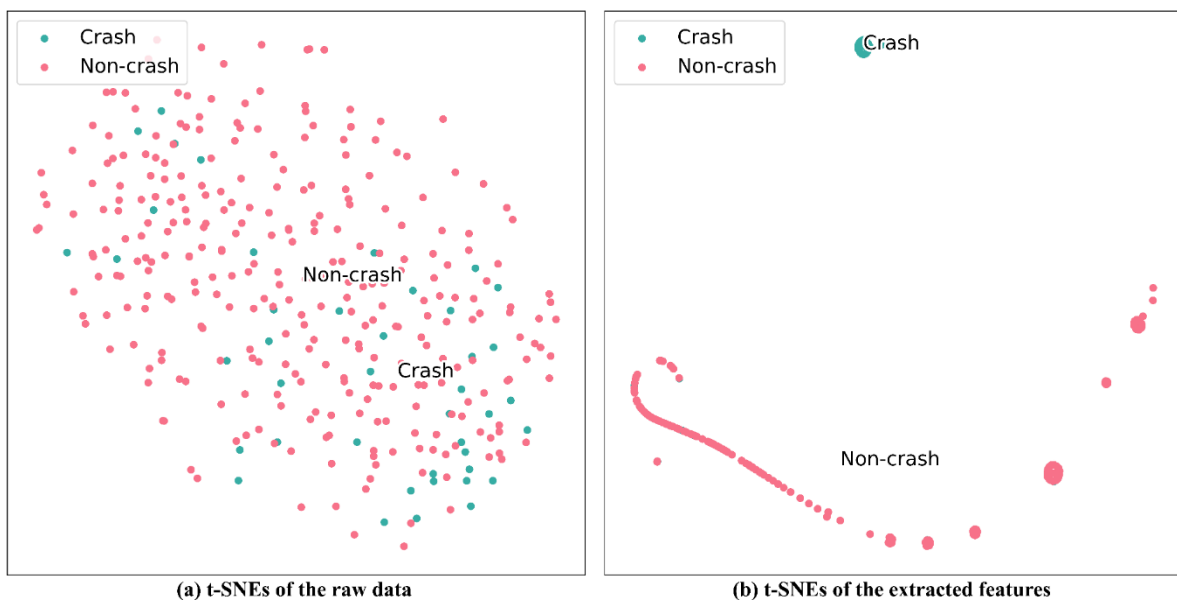


Figure 33 t-SNEs of the raw data and extracted features

5.4.3 Model Comparison

This section compares the performance of the proposed LSTM with two types of benchmark methods, statistical methods and machine learning methods. Bayesian logistics regression is used as the statistical method. This method was widely applied to the real-time crash potential prediction (Ahmed et al., 2012b; Yu and Abdel-Aty, 2013). Different from the basic logistics regression, in the Bayesian approach, the parameters are treated as random variables and the data are used to update beliefs about the behavior of the parameters to assess their distributional properties (Ahmed et al., 2012b). For machine learning methods, XGBoost is

selected due to its good performance on similar problems (Chen and Guestrin, 2016). These two methods are trained based on the same dataset as the LSTM. Several hyperparameters of XGBoost are tuned, such as the number of trees, maximum tree depth, etc. The results of the model comparison are shown in Figure 34. LSTM outperforms the other two methods in terms of sensitivity, false alarm rate, and AUC.

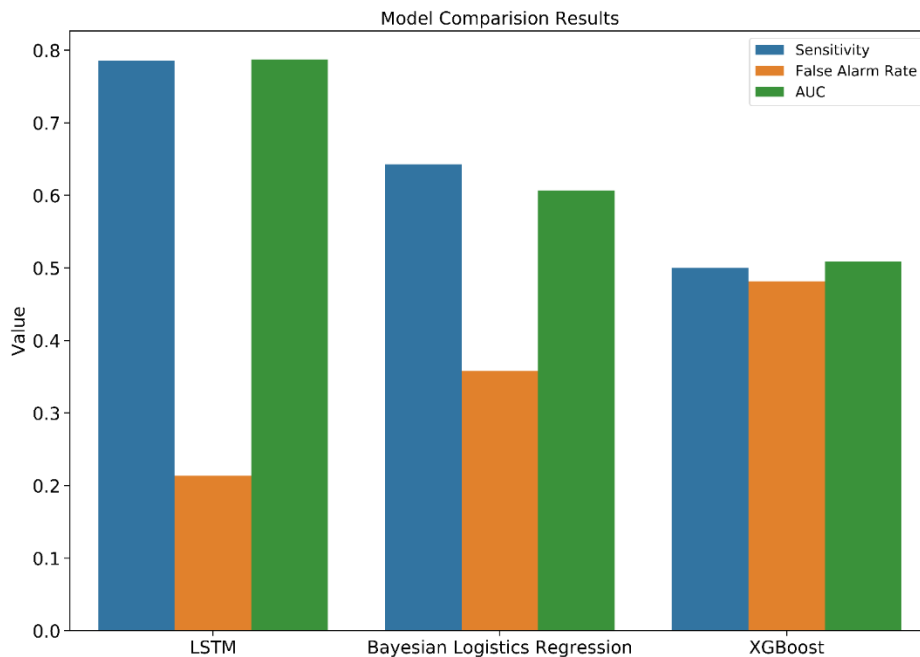


Figure 34 Model comparison results

5.5 Conclusions

This study applied a new CV emulated data source to predict the real-time crash potential for urban arterials. Two urban arterials in Orlando, USA were selected to conduct a case study. Crash and CV emulated data were obtained for three months. The CV emulated data were used to generate different speed-related variables, such as average speed, speed standard deviation, 85th percentile speed, etc. After data cleaning and preparation, an LSTM model was proposed to

predict the crash potential in the next 5-10 minutes. The model was trained on the data after over-sampling and validated based on different evaluation metrics. Results illustrated the feasibility of using CV emulated data for real-time crash potential. In addition, the model comparison results indicated the LSTM outperformed other methods, including Bayesian logistics regression and XGBoost in terms of sensitivity, false alarm rate, and AUC. Several key findings from this study can be summarized as below:

- Different speed-related variables can be generated from the CV emulated data, such as the average speed, speed standard deviation, 85th percentile speed, etc. These variables can be updated in a high frequency to better reflect the continuous traffic conditions.
- With the help of map-matching methods, the variables from bus data can be transformed into the variables of the road segments. Different from other types of vehicles such as the taxi, a bus usually has a fixed route and schedule. The computation costs of map matching are greatly mitigated for bus data.
- The CV emulated data can be used as a new data source to predict real-time crash potential. The data have higher flexibility and wider coverage compared with the traditional sensor data. With the rapid development of the CV, the results from this study can be generalized to other types of vehicles.
- LSTM outperforms other types of models in terms of sensitivity, false alarm rate, and AUC values. With the help of the over-sampling methods, the LSTM can learn time-series data comprehensively. In addition, due to the rapid development of computer hardware, the implementations of deep learning models will be much easier in the future.

There are still several improvements that can be done in the future. First, buses are one type of vehicles. It is very promising to explore the fusion with other vehicle types, such as taxis, private vehicles, trucks, etc. Second, the impact of the different variables on crash potential prediction also needs further investigation, a proper variables generation and selection process could improve the performance of the model. Finally, different deep learning architectures can be explored in the future to improve the results of the current model.

CHAPTER 6: USING BUS CRITICAL DRIVING EVENTS AS SURROGATE SAFETY MEASURES FOR PEDESTRIAN AND BICYCLE CRASHES BASED ON GPS TRAJECTORY DATA

6.1 Introduction

The safety of pedestrians and bicyclists is a serious concern for traffic researchers, while we are experiencing an increase in pedestrian and bicycle crashes. In 2018, 6,283 pedestrians were killed by traffic crashes in the USA, a 3.4% increase from the 6,075 pedestrian fatalities in 2017 (NHTSA, 2020b). Similarly, there were 857 cyclists deaths in 2018, an increase of 6.3% from 2017 (NHTSA, 2020a). During the past decades, numerous pedestrian and bicycle safety studies were conducted at the micro- and macro-level, including intersections, road segments, and Traffic Analysis Zone (TAZ) (Lee et al., 2015; Schepers et al., 2011; Ukkusuri et al., 2011; Xuesong Wang, 2016). Specifically, a TAZ is a special-purpose geographic entity delineated by Metropolitan Planning Organizations (MPOs) and state Departments of Transportation (DOTs) for tabulating traffic-related data from the decennial census (U.S. Census Bureau, 2011). Several traffic, demographic, and geometry-related factors were found to be highly correlated with the pedestrian and bicycle crashes, such as traffic volume, road density, and population. In addition, recent studies also indicated that the presence of bus stops was correlated with pedestrian and bicycle crashes. (Hess et al., 2004; Xie et al., 2017). Bus stops were also considered as possible crash hotspots due to the attraction of pedestrians and bicyclists (Truong and Somenahalli, 2011). However, only a few studies considered the influence of buses' characteristics (Zhou and Bromfield, 2007) on pedestrian and bicycle crashes, such as speed, acceleration, and dwell time.

Over the last decades, various studies have been conducted to identify the relationship between pedestrian and bicycle crash occurrence and various factors. Schepers et al. (2011) indicated that the volume of through motorized vehicles and the volume of cyclists crossing the major road were positively related to the bicycle crashes at intersections. Lee et al. (2015) found that the TAZ with higher Vehicle Miles Traveled (VMT) and population were associated with higher pedestrian and bicycle crashes. Some studies investigated the impact of bus stops on pedestrian and bicycle crashes. For example, Xie et al. (2017) showed that bus stop density was positively associated with pedestrian crashes in urban areas. Chen and Zhou (2016) also indicated that the number of bus stops in a TAZ would result in a higher pedestrian crash frequency. In summary, these reactive traffic safety studies provided valuable insights into the contributing factors to pedestrian and bicycle crashes from different perspectives. Reactive safety studies usually require the occurrence of a significant number of crashes for the analysis. However, the number of pedestrian and bicycle crashes is much less compared to non-crash events or other types of crashes. In addition, collecting and investigating real crash data is also time-consuming. Regarding these problems, the concept of proactive traffic safety analysis is becoming popular among traffic studies. Proactive traffic safety analysis is complementary to the traditional active safety analysis, which requires fewer historical crash events. (Vedagiri and Killi, 2015). Proactive traffic safety analysis could evaluate traffic safety status through other safety-related indicators, such as surrogate safety measures.

One purpose of using surrogate safety measures is to assess traffic safety status when crash events are rare or unavailable. Several surrogate safety measures were widely used by the previous studies for pedestrian and bicycle safety analysis (Fu et al., 2016; Zangenehpour et al.,

2016; Zhang et al., 2020c), such as time to collision, post-encroachment time, and gap time. Most studies used video data to generate these surrogate safety measures. However, the quality of video data could be easily influenced by the surrounding conditions. In addition, large-scale video data are also hard to obtain and process, thus most studies only selected a small area (e.g. one or two intersections) to conduct analysis. Due to the development of mobile sensing technologies and the popularity of mobile devices (e.g. smartphones, tablets, etc.), city-wide GPS data can now be easily obtained at a low cost. Recently, some studies explored the possibility of using GPS data to extract surrogate safety measures. For example, Stipancic et al. (2018a) extracted hard decelerating and accelerating events from GPS travel data and compared them to historical vehicle crashes data. Results indicated that these two events were positively correlated with crash frequency at the link and intersection levels. Similarly, Boonsiripant et al. (2011) estimated different speed-related variables from GPS trajectory data. The authors indicated that the acceleration rate and driver stop frequency were positively correlated with the vehicle crashes at the corridor level. In terms of pedestrian and bicycle crashes, Strauss et al. (2017) estimated the deceleration rates from a large sample of cyclist GPS data. Results from Spearman's rank correlation coefficient indicated that the deceleration rates were positively associated with the bicycle crashes at intersections and road segments. In addition, existing studies also suggested some bus behaviors may be associated with crashes. For example, Zhou and Bromfield (2007) found crashes were prone to happen for a stopped bus where the dwell time exceeded 30 seconds. af Wåhlberg (2004) indicated that the bus's acceleration behavior is related to crash frequency. Cafiso et al. (2013) found several bus driving events were most relevant to the road safety status, including hard braking, stopping, etc. Similarly, Cheranchery et

al. (2016) indicated that the bus's dwell time is related to the safety level of bus stops. It is promising to use bus trajectory data to extract new surrogate safety measures. However, none of the existing studies were conducted for pedestrian and bicycle crashes from literature review (Li et al., 2021b).

There are some limitations in the existing studies. First, pedestrian and bicycle surrogate safety measures have not been thoroughly developed with GPS data. Second, only a few existing studies considered bus behaviors as possible surrogate safety measures. Third, most surrogate safety studies were conducted at limited locations while a city-wide analysis is crucial. With the capability of accessing real-time bus trajectory data in Orlando, Florida, this study aims to investigate the feasibility of using critical bus driving events as pedestrian and bicycle surrogate safety measures. The contributions of this study can be summarized as below:

- This study extracts various variables from bus trajectory data to reflect buses' operation status around the bus stops.
- This study proposes new surrogate safety measures from the critical bus driving events for the pedestrian and bicycle crashes at the bus stops.
- The proposed surrogate safety measures are validated on a city-wide trajectory dataset. In addition, the correlations between surrogate safety measures and crashes are investigated per crash type (i.e., pedestrian and bicycle)
- A Bayesian negative binomial model incorporating spatial correlation is proposed to model the crash frequency using the generated bus driving events.

6.2 Research Approach

The approach used in this study has several steps (Figure 35). For trajectory data, bus trips are first generated. After filtering the trip data, the bus's speed, acceleration, and dwell time are computed accordingly. Three critical driving events are then generated from the acceleration rate and dwell time. The prepared trajectory and crash data are matched to the bus stop geographical data using *QGIS* (QGIS.org, 2016). The correlations between crash events and the critical driving events are examined with Spearman's rank correlation coefficient. Besides, a Bayesian negative binomial model incorporating the spatial correlation is used to model crash frequency using variables generated from the bus trajectory data.

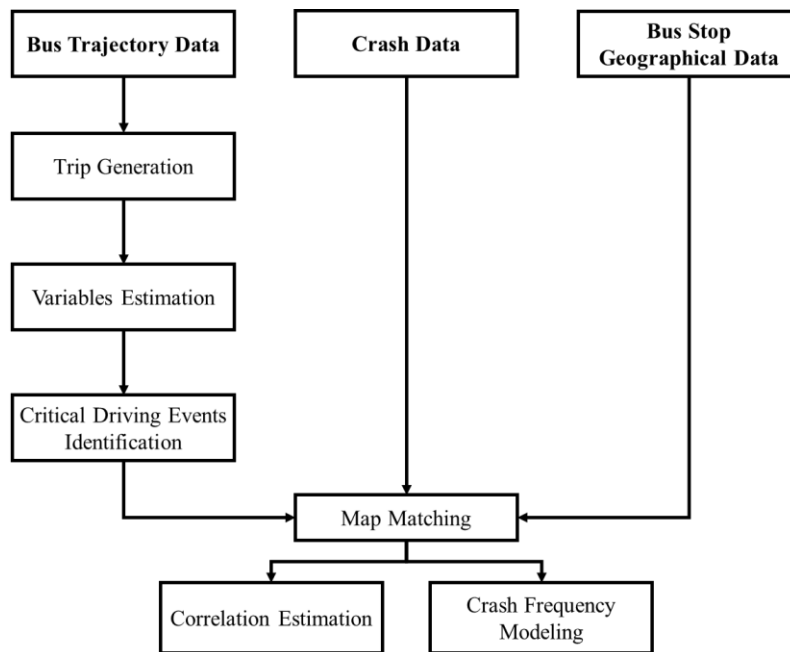


Figure 35 Research approach flowchart

6.2.1 Trajectory Data Preparation

This section presents the process of bus trajectory data preparation. Bus trips are first extracted from the trajectory data. The speed and acceleration of buses are estimated using the generated trips. Three critical events are identified using speed and acceleration, including hard acceleration, hard deceleration, and long stop.

- **Bus Trip Generation**

Duplicated records (e.g., same bus vehicle ID, time, longitude, and latitude) are first removed from the trajectory dataset. To estimate speed-related variables, the bus trajectory data are divided into multiple trips. A trip is defined as the collection of consecutive GPS coordinates for one bus. In the Lynx system, each bus is running continuously on its assigned route. Therefore, a trip is generated using three control parameters: bus ID, time, and route. Figure 36 shows an example of generating trips for a bus with ID as '17706388'. This bus has the 1st trip as route 541 from '2019-12-30 18:59:40' to '2019-12-30 19:21:36'. It has the 2nd trip as route 544 from '2019-12-30 19:22:14' till '2019-12-30 20:55:08'. After this process, two types of trips are removed: 1) Short trip, a trip that has less than 100 GPS coordinates or lasts less than 1 minute is considered as a short trip. 2) Stationary trip, a trip is removed when the bus stays at the same location information during the entire trip.

	id	time	route	lat	lon	
0	17706388	2019-12-30 18:59:40	541	28.4501791667	-81.4297721667	} Trip 1
1	17706388	2019-12-30 18:59:51	541	28.4501656667	-81.4288470000	
2	17706388	2019-12-30 19:00:10	541	28.4501653333	-81.4287498333	
3	17706388	2019-12-30 19:01:40	541	28.4494376667	-81.4280050000	
4	17706388	2019-12-30 19:02:00	541	28.4463965000	-81.4279718333	
...	
54	17706388	2019-12-30 19:21:06	541	28.4447541667	-81.3993501667	} Trip 2
55	17706388	2019-12-30 19:21:36	541	28.4435733333	-81.4000973333	
56	17706388	2019-12-30 19:22:14	544	28.4457706667	-81.4003768333	
57	17706388	2019-12-30 19:23:08	544	28.4457743333	-81.4003386667	} Trip 2
58	17706388	2019-12-30 19:31:47	544	28.4458793333	-81.4003036667	
			...			} ...
						} Trip N

Figure 36 Bus trip generation illustration

- **Variables Estimation**

The instantaneous speed of the bus is estimated from the generated trips. For a bus with its trip, the travel distance d between two consecutive points (GPS_1, GPS_2) is estimated using Haversine Distance (equation (38) to (42)) (Veness, 2019) since the GPS coordinates are in the World Geodetic System (WGS84). Specifically, φ and λ are latitude and longitude in radians. For example, φ_1 and λ_1 are the latitude and longitude of GPS_1 in radians. R is earth's radius (6,371 km).

$$\Delta\varphi = \varphi_2 - \varphi_1 \quad (38)$$

$$\Delta\lambda = \lambda_2 - \lambda_1 \quad (39)$$

$$c = 2 \cdot \arctan2(\sqrt{a}, \sqrt{1-a}) \quad (40)$$

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_{t1}) \cdot \cos(\varphi_{t2}) \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (41)$$

$$d = R \cdot c \quad (42)$$

The bus speed is estimated as the distance (d) divided by the time difference. After computing speed, the speed is smoothed with a Savitzky-Golay filter. This filter was widely adopted by the studies related to GPS trajectory data and achieved promising results (Stipancic et al., 2018a; Zaki et al., 2014). This filter has two crucial parameters, the order of the polynomial used to fit the samples and the length of the filter window. In this study, the order of the polynomial is selected as 2, and window length is selected as 3 based on their performance on smoothing the trajectory data and the results from previous studies.

- **Critical Driving Events Identification**

Bus's acceleration rates are first estimated. One additional advantage of the Savitzky-Golay filter is it smooths not only the original data (speed), but also its derivatives (acceleration) (Stipancic et al., 2018a). Therefore, the bus's acceleration rates are derived using this filter. Then, a bus's dwell time is estimated as the cumulative time while its speed equals zero among consecutive GPS coordinates. In addition, the dwell time that is less than three seconds is removed. After these two steps, an interquartile range filter is applied to dwell time and acceleration rate to remove the outliers. Three critical driving events are generated using the acceleration rate and dwell time, hard acceleration, hard deceleration, and long stop. Existing studies provide some references for the choices of thresholds. Duarte et al. (2013) suggested using 0.3 m/s² and -0.3 m/s² as the thresholds for hard acceleration and deceleration,

respectively. Zhou and Bromfield (2007) found that crashes were prone to happen for a stopped bus where the dwell time exceeded 30 seconds. Therefore, the thresholds for the hard acceleration and deceleration are selected as 0.3 m/s² and -0.3 m/s², respectively. The threshold for the long stop is set as 30 seconds.

6.2.2 Map Matching

The bus trajectory data, crash data, and bus stop data are geocoded using *QGIS* as three shapefiles. A circular buffer is created around each stop. Bus trajectory and crash data are matched to a bus stop if they are within its buffer. In addition, to prevent duplicated matches, trajectory and crash data are matched to the closest bus stop if they fall into multiple buffers.

6.2.3 Correlation Estimation

This study utilizes Spearman's rank correlation coefficient (Spearman's rho (ρ)) to examine the correlations between bus critical events and crashes around the bus stops. Spearman's rho measures the strength and direction of the association between two ranked variables. It was widely used by previous studies to explore the relationship between possible surrogate safety indicators and crashes (Stipancic et al., 2018a; Strauss et al., 2017). Spearman's rho (ρ) is computed using equation (43). Where $d_i = Crash_i - Event_i$ is the difference between the ranks of stop i based on the number of crashes and the number of critical events. n is the number of bus stops.

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)} \quad (43)$$

6.2.4 Crash Frequency Modeling

This section introduces the models used in this study for crash frequency estimation. The negative binomial model is adopted due to its ability to handle over-dispersed data. Spatial correlation is considered during the modeling process with Bayesian inference. In addition, two goodness-of-fit metrics are applied to evaluate the performance of the model.

- **Negative Binomial Model**

Poisson model was used for modeling crash frequency during the early decades (Lord and Mannering, 2010), it has the assumption as equal mean and variance of the data distribution. However, crash data are usually over-dispersed with a long tail (i.e., a few samples have an extremely high number of crashes). The Negative Binomial (NB) model relaxes the equal mean and variance assumption of the Poisson model, which can account for overdispersion resulting from unobserved heterogeneity and temporal dependency. NB model was widely used by the previous studies (Lee et al., 2019; Stipanovic et al., 2018b; Zou et al., 2015) to model crash frequency with other explanatory variables. The NB model introduces an extra parameter ε_i to the Poisson model which allows the variance to differ from the mean (equation (44)). For each observation, bus stop i , $\exp(\varepsilon_i)$ is a Gamma-distributed disturbance term with mean 1 and variance α . \mathbf{X}_i is a vector of the explanatory variables and $\boldsymbol{\beta}$ is a vector of estimable parameters. λ_i is the expected number of crashes for bus stop i . The NB model has the form shown in equation (45), where $\Gamma(\cdot)$ is a gamma function, y_i is the number of crashes for bus stop i .

$$\ln(\lambda_i) = \boldsymbol{\beta}\mathbf{X}_i + \varepsilon_i \quad (44)$$

$$P(y_i) = \frac{\Gamma\left(\frac{1}{\alpha} + y_i\right)}{\Gamma(y_i + 1)\Gamma\left(\frac{1}{\alpha}\right)} \left(\frac{\frac{1}{\alpha}}{\frac{1}{\alpha} + \lambda_i}\right)^{\frac{1}{\alpha}} \left(\frac{\lambda_i}{\frac{1}{\alpha} + \lambda_i}\right)^{y_i} \quad (45)$$

Note: the equations are based on Washington et al. (2020).

- **Spatial Correlation**

Bayesian approaches were widely adopted to consider the spatial correlation for modeling crash frequency. For example, Truong and Currie (2019) developed a spatial Bayesian model with conditional autoregression (CAR) prior to predict the crash frequency by considering the spatial correlation at the TAZ level. Similarly, Xie et al. (2014) introduced a CAR prior to the negative binomial model to analyze the crash frequency at the intersections. The proposed model reached better results compared with the regular negative binomial model. To account for the spatial correlation between neighboring bus stops, this study adopts a Bayesian NB-CAR model. CAR prior was widely utilized by existing studies related to traffic safety analysis and achieved promising results (Gu et al., 2020; Truong and Currie, 2019; Xie et al., 2014). There are different versions of CAR priors, the intrinsic CAR prior proposed by Besag et al. (1991) are used in this study due to its simplicity and popularity. With the introduction of CAR prior, equation (44) can be rewritten as equation (46), where the spatial autocorrelation is modeled via ϕ . ϕ_{-i} is the set of ϕ_j for any $j \neq i$. θ is a normally distributed term with mean 0 and variance σ^2 . \mathbf{W} is weight matrix representing the spatial relationship between each bus stop. w_{ij} has a non-negative value if stop i and j are adjacent, or 0 otherwise.

$$\ln(\lambda_i) = \boldsymbol{\beta} \mathbf{X}_i + \theta_i + \phi_i \quad (46)$$

$$\phi_i | \phi_{-i}, \mathbf{W}, \tau^2 \sim N\left(\frac{\sum_{j=1}^n w_{ij} \phi_j}{\sum_{j=1}^n w_{ij}}, \frac{\tau^2}{\sum_{j=1}^n w_{ij}}\right) \quad (47)$$

$$\theta_i \sim N(0, \sigma^2) \quad (48)$$

$$\tau^2, \sigma^2 \sim \Gamma^{-1}(a, b) \quad (49)$$

Note: the equations are based on Lee (2013).

Parameters of the NB and NB-CAR models are estimated under the Bayesian context using R with Markov Chain Monte Carlo (MCMC) simulation. Twenty thousand (20,000) iterations are set as a burn-in period. Parameters of the models are estimated based on the one hundred thousand (100,000) samples. Without sufficient prior knowledge, the following non-informative priors are used based on the results from the previous studies (Gu et al., 2020; Truong and Currie, 2019): $\tau^2 \sim \Gamma^{-1}(1, 0.01)$, $\sigma^2 \sim \Gamma^{-1}(1, 0.01)$. The models are implemented based on the *CARBayes* (Lee, 2013) and *runjags* (Denwood, 2016) packages.

- **Performance Evaluation**

Two goodness-of-fit measures are used in this study to evaluate the model performance are Watanabe-Akaike Information Criterion (WAIC, Watanabe and Opper (2010)) and Deviance Information Criteria (DIC, Spiegelhalter et al. (2002)). WAIC is a Bayesian approach for estimating the out-of-sample expectation (Gelman et al., 2014). WAIC can be estimated using equation (50), where LPPD is the log posterior predictive density calculated using equation (51). p_D is the effective number of parameters (Satria et al., 2020). DIC was designed to evaluate the model complexity based on its ability to explain the variation in the data. DIC is calculated using equation (52), where \bar{D} is the posterior deviance and measures the model fit and p_D is the

effective number of parameters and measures the model complexity (Satria et al., 2020). In general, a model with lower WAIC and DIC is preferred.

$$WAIC = -2 * (LPPD - p_D) \quad (50)$$

$$LPPD = \sum_{j=1}^n \log \int p(y_i|\theta) p_{post}(\theta) d\theta \quad (51)$$

$$DIC = \bar{D} + p_D \quad (52)$$

6.3 Data Description

Two types of data are used in this study, Lynx data and crash data. Lynx provides the public transportation service for Orange, Seminole, and Osceola counties in Orlando, Florida. Lynx bus data, including bus trajectory and stop data, are obtained through the data collection API of DoubleMap®. The API requests are made with an HTTP GET request, and the data are returned in JSON format. The trajectory dataset contains around 300 buses, with their information updated around every three seconds. The buses are located via an Automatic Vehicle Locator (AVL) system with three meters' location accuracy, which is acceptable for transportation research according to previous studies (Li et al., 2020b; Ochieng and Sauer, 2002). The trajectory dataset description is shown in Table 16. The bus stop data contain detailed information of 4,326 stops in the Lynx system, including stop ID, longitude, latitude, and name. The pedestrian and bicycle crash data are obtained from the Signal Four Analytics (S4A) system. This system archives the crashes in Florida with detailed information, such as crash time, location, type, severity, etc.

Table 16 Lynx trajectory data

Name	Description
ID	The unique integer for each bus.
Name	A text name for the bus.
Latitude	Latitude for the bus's current position.
Longitude	Longitude for the bus's current position.
Heading	The direction of the vehicle, in degrees (0-360).
Time	The time of the GPS update from the bus.
Route	The unique ID for the bus route

This study selects part of Orange County, FL as the research area considering the data availability and distribution (Figure 37). Bus data were collected from December 2019 to February 2020. Ten years (2010-2020) pedestrian and bicycle crash data were obtained from the Signal Four Analytics (S4A) system. It provides detailed information for all reported crashes, such as crash time, location, type, and severity. In total, the selected area has 6,716,869 GPS records, 1,363 bus stops, 1,889 bicycle crashes, and 2,910 pedestrian crashes. According to the results from the previous studies (Stipanovic et al., 2018b), two scenarios are designed based on buffer size for the bus stops, which are 100 m (100 meters) and 200 m (meters).

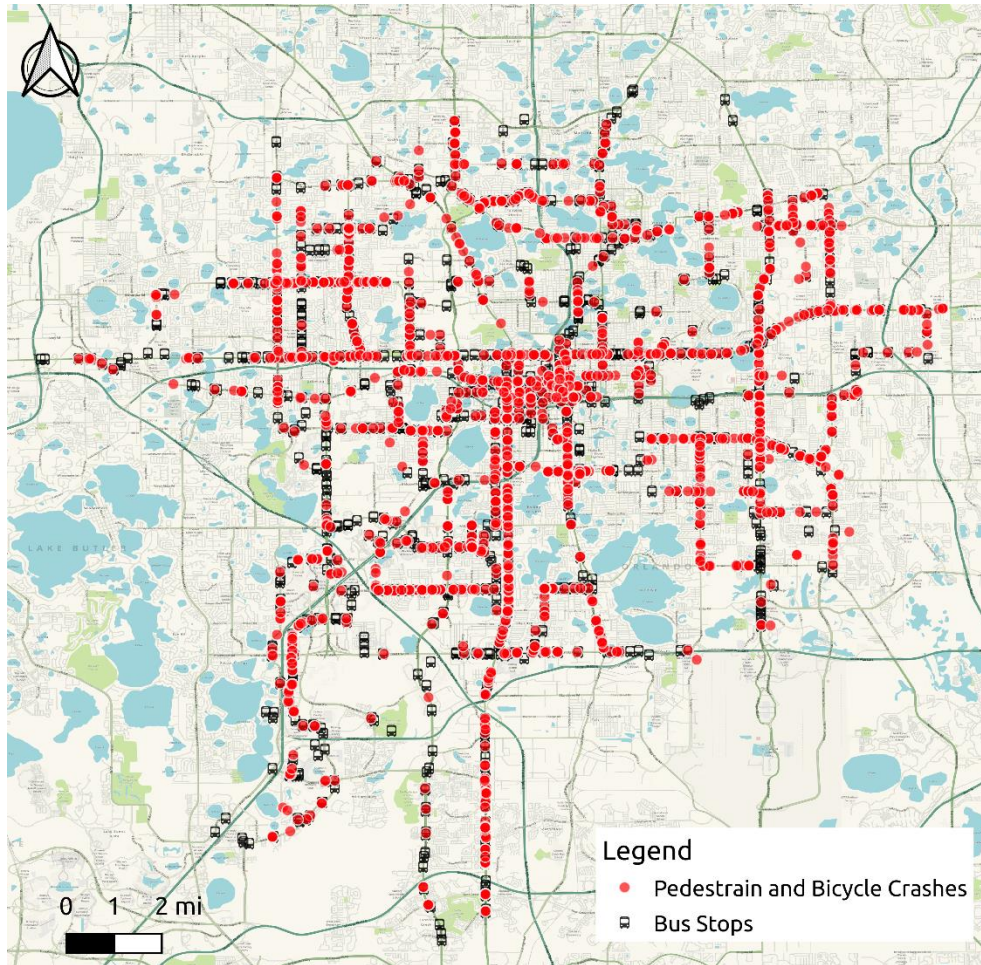
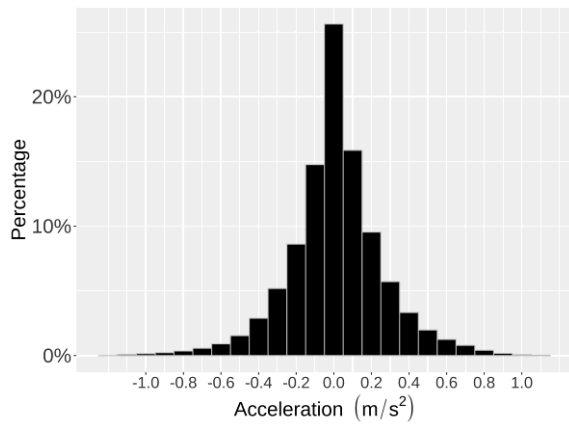


Figure 37 Research area

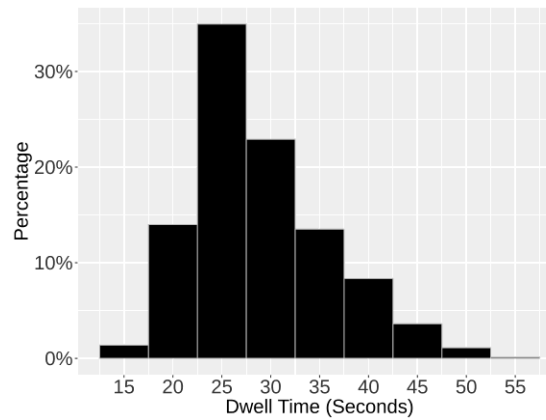
6.4 Experimental Results

This section summarizes the results of this study. First, the results from critical events identification are presented with distribution plots and descriptive statistic tables. The correlation coefficients between critical events and crashes are also presented per crash type. Two models are developed using the critical events to predict the pedestrian and bicycle crashes, Bayesian NB and Bayesian NB-CAR model. The posterior means and Bayesian credible intervals are presented for each variable of these two models. Besides, the performance of the two models is compared using WAIC and DIC.

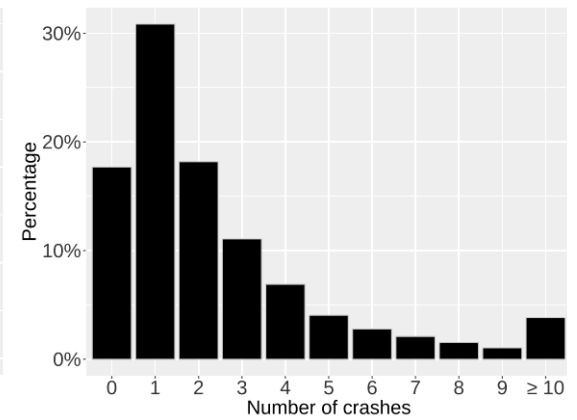
The collected bus trajectory and crash data are processed using the proposed approach. After matching the trajectory and crash data to the bus stops, Figure 38 shows the distribution of the acceleration rate, dwell time, and crash for the two scenarios. The buses' acceleration rates have relatively symmetrical distributions, while most of the buses have the acceleration rates as zero, indicating that buses either maintained constant speeds through the stops or stopped at the stops. The dwell time has right-skewed distributions over these two scenarios, around 50% average dwell time is smaller than 30 seconds. In terms of pedestrian and bicycle crashes, most of the bus stops have no more than 2 crashes. Moreover, the 200 m scenario has a relatively higher portion of the large number of crashes (e.g. more than 10) compared with the 100 m scenario. The descriptive statistics of these variables are shown in Table 17. The 200 m scenario has higher mean values for the number of crashes and critical events, which is reasonable since it covers wider areas. In addition, the number of critical events significantly varies for different stops.



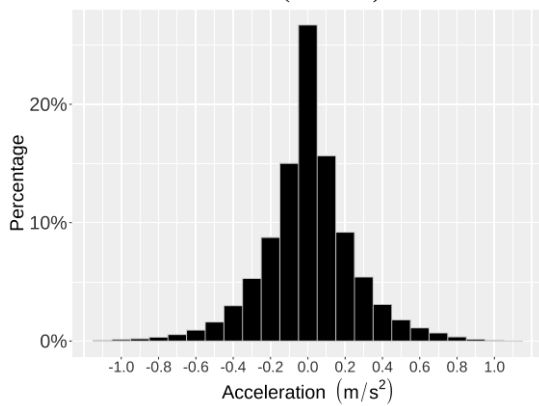
a. Acceleration rates (100 m)



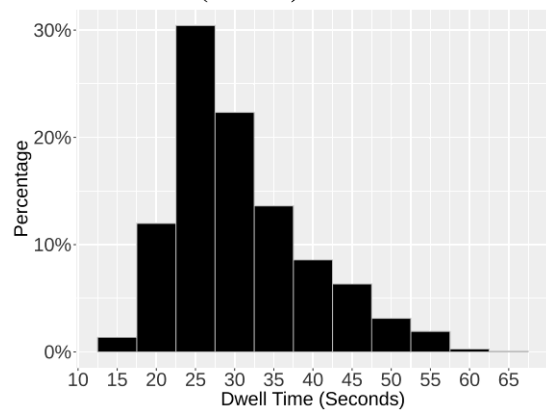
b. Dwell time (100 m)



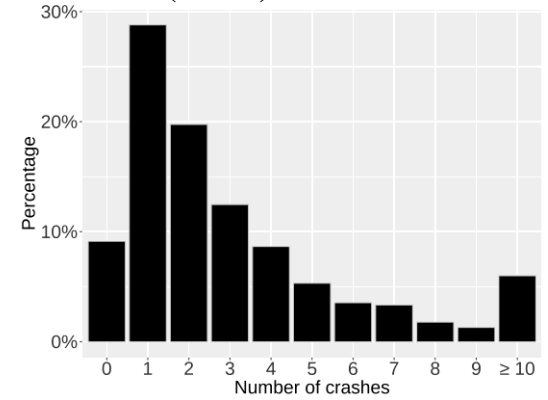
c. Crashes (100 m)



d. Acceleration rates (200 m)



e. Dwell time (200 m)



f. Crashes (200 m)

Figure 38 Distributions of acceleration rates, dwell time, and crashes

Table 17 Descriptive statistic table

Variables	100 m				200 m			
	Mean	Std. Dev.	Min	Max	Mean	Std. Dev.	Min	Max
Stop time (seconds)	28.94	6.98	15	55	30.77	8.61	15	65
Acceleration rates (m/s²)	0.01	0.25	-1.18	1.12	0.01	0.25	-1.14	1.09
Number of crashes	2.55	3.10	0	24	3.27	3.81	0	41
Number of hard accelerations	155	176.49	0	1345	179.70	210.65	0	1915
Number of hard decelerations	1284.73	1325.25	70	15762	1617.23	2472.94	70	71816
Number of long stops	73.40	121.71	0	1276	90.42	191.14	0	4235

The Spearman's rank correlation coefficients of these two scenarios are shown in Table 18. The correlations between crashes and critical driving events for the 200 m scenario are stronger compared with the 100 m scenario. For example, the hard acceleration event has a correlation coefficient of 0.427 with pedestrian crashes under the 200 m scenario, which is larger than 0.374 under the 100 m scenario. The long stop event also has a larger correlation coefficient (0.462) with pedestrian crashes under the 200 m scenario than the 100 m scenario (0.403). Besides, the authors tested the larger buffer sizes and the correlations start to decrease. Therefore, the authors suggest that the 200 m is a desirable choice as a buffer size for bus stops. Long stop has the highest correlation coefficient among the three critical driving events, while hard acceleration has a higher correlation coefficient compared with hard deceleration. The reason for the high correlation coefficient of the long stop may be due to that it relatively indicates higher pedestrian and bicycle volume. The longer a bus stops, the more passengers it may have. Also, a longer stop could be causing a blind spot for seeing pedestrians/bicyclists by incoming vehicles. A stopped bus could also be contributing to the disruption of traffic. In terms of crash types, the pedestrian has higher correlation coefficients with all three critical driving events than

the bicycle crashes. In summary, the results indicate that bus stops with a great number of critical bus driving events may also have more pedestrian and bicycle crashes.

Table 18 Spearman’s rank correlation coefficients

	100 m			200 m		
	All	Pedestrian	Bicycle	All	Pedestrian	Bicycle
Critical driving events						
Long stop	0.381	0.403	0.253	0.436	0.462	0.304
Hard acceleration	0.371	0.374	0.253	0.419	0.427	0.294
Hard deceleration	0.345	0.361	0.255	0.385	0.405	0.288

Two models are built to estimate the number of crashes using the critical bus driving events, Bayesian NB and Bayesian NB-CAR model. Hard deceleration is removed from the explanatory variables since it is highly correlated with the hard acceleration and less correlated with the crashes. To account for the influence of bus volume, the number of hard accelerations and long stops are divided by the number of buses. Besides, the number of buses is used as the proxy for exposure, which is converted using the natural log.

The results of the Bayesian NB and Bayesian NB-CAR models are shown in Table 19, where the variables that are statistically significant at 95% Bayesian Credible Interval (BCI) are highlighted in red. The Bayesian NB-CAR outperformed Bayesian NB with lower WAIC and DIC values, indicating the necessity of considering spatial correlation during the crash frequency modeling process. The posterior means for the number of hard accelerations/the number of buses (HAc/Bus) and the number of long stops/the number of buses (LS/Bus) are positive over the four models, which are consistent with the results from the correlation estimation. Positive posterior means suggest that the increase of critical events will increase pedestrian and bicycle crashes. A bus stop with more hard acceleration events may indicate that it has relatively complicated traffic conditions (e.g., heavy traffic volume, congestion), which could increase the probability of

having pedestrian and bicycle crashes. Besides, LS/Bus has a higher posterior mean than HAc/Bus among all the models, which highlights the impact of a stopped bus on traffic safety. A stopped bus is a relatively large obstacle for other road users. Drivers behind the bus may try to change lanes without observing pedestrians and bicycles covered by the bus. Besides, the posterior mean of the number of buses (exposure) is also positive among the four models as expected. A bus stop with more buses could have a relatively large pedestrian and bicycle volume, which could increase the possibility of a crash happening. In terms of the crash type, HAc/Bus, LS/Bus, and the number of buses are all statistically significant for modeling pedestrian crashes. Similarly, LS/Bus and the number of buses are statistically significant for modeling bicycle crashes. The only exception is that LS/Bus is statistically significant for modeling bicycle crashes in the NB model, while it is not significant in the Bayesian NB-CAR model. Existing studies also have similar findings while the variables' significance changes after the usage of spatial correlation (Stipancic et al., 2018b; Truong and Currie, 2019). The variables' significance may be better validated by increasing the data size or expanding the research area in the future. In summary, the results from the crash modeling confirm the findings from correlation estimation. The generated bus critical driving events are positively correlated with the crash frequency. The long stop event is found to have a higher posterior mean than the hard acceleration event.

Table 19 Results of Bayesian NB and Bayesian NB-CAR models

Variables	Bayesian NB model				Bayesian NB-CAR model			
	Pedestrian		Bicycle		Pedestrian		Bicycle	
	Mean	95% BCI	Mean	95% BCI	Mean	95% BCI	Mean	95% BCI
Intercept	-2.636	(-3.065, -2.183)	-1.558	(-2.020, -1.111)	-8.827	(-9.567, -8.111)	-7.963	(-9.042, -6.553)
HAc/Bus	1.853	(1.070, 2.637)	1.596	(0.463, 2.562)	3.712	(1.935, 5.812)	1.650	(0.117, 3.586)
LS/Bus	5.977	(4.744, 7.389)	3.819	(2.597, 5.374)	7.557	(5.552, 9.663)	2.481	(-0.863, 4.991)
Log(number of buses)	0.398	(0.336, 0.451)	0.239	(0.176, 0.301)	0.569	(0.472, 0.655)	0.476	(0.301, 0.631)
τ^2					0.002	(0.002, 0.003)	0.003	(0.002, 0.004)
σ^2					0.005	(0.002, 0.012)	0.003	(0.001, 0.009)
WAIC	4733.2		3568.8		4137.3		3065.2	
DIC	4732.0		3568.0		4339.4		3282.3	

Note: Variables significant at 95% Bayesian credible interval are highlighted in red

6.5 Conclusions

The purpose of this study is to explore the possibility of using critical bus driving events as pedestrian and bicycle surrogate safety measures around bus stops. This study presents a series of approaches to prepare trajectory data in terms of trip generation, trip cleaning, and variables estimation. A city-wide bus trajectory dataset was collected from Orlando, Florida. Three critical driving events were generated using the proposed methods, hard acceleration, hard deceleration, and long stop. Spearman's rank correlation coefficients were used to examine the relationships between critical driving events and the pedestrian and bicycle crashes at the bus stops. The NB models are built to model the crash frequency with the critical driving events. All three events were positively related to pedestrian and bicycle crashes. Specifically, the long stop has a correlation coefficient of 0.436, hard acceleration has a correlation coefficient of 0.419, and hard deceleration has a correlation coefficient of 0.385. Two hundred meters buffer was suggested as a desirable choice for the bus stops. Besides, the results vary between crash types. The correlations between pedestrian crashes and critical driving events are relatively higher than the correlations between bicycle crashes and critical driving events. Two models (Bayesian NB and Bayesian NB-CAR) were developed to model the crash frequency using the generated critical events, with the number of buses as a proxy for exposure. The models' results are consistent with the results from correlation estimation. Hard acceleration, long stop events, and the number of buses (exposure) were found to be positively related to the crash frequency. A bus stop with more hard acceleration and long stop events is expected to have more crashes. Moreover, model evaluation results suggested that the Bayesian NB-CAR outperformed the

Bayesian NB with lower WAIC and DIC. Several suggestions can be summarized from the study's results. For example, it is suggested that the bus company should coach drivers if they are inclined to have aggressive behaviors such as hard acceleration or deceleration. Moreover, the design of bus stops should consider the impact caused by the stopped buses.

The critical driving events proposed by this study can be eventually used for network screening, safety evaluation, etc. Similar logic can be extended to other facility types (e.g., intersection and segment) with vehicle trajectory data. However, it is worth mentioning that this study also has some limitations. More validation can be done in the future considering the influence of geometry, exposure, and other factors. Moreover, existing studies (Washington and Oh, 2006) indicated that the usage of informative priors could improve the performance of the model. The informative priors could be formulated using two-stage Bayesian updating in the future. In the end, the variables' significance can be better examined by extending the research area or increasing the data size.

CHAPTER 7: TRAJECTORY FUSION-BASED REAL-TIME CRASH LIKELIHOOD PREDICTION USING LSTM-CNN WITH ATTENTION MECHANISM

7.1 Introduction

Real-time crash likelihood prediction is a major component of Active Traffic Management Systems. Different from the traditional crash frequency prediction using aggregated data (e.g. by day, month, and year), real-time crash likelihood prediction predicts the crash likelihood during a short period (e.g., 5 minutes). In 2018, according to the data from the insurance institute for highway safety, motor vehicle crashes caused 36,560 fatalities in the USA (IIHS, 2019). Among the 19,498 deaths in urban areas, 78% of them happened at arterials. However, most of the previous studies about real-time crash likelihood prediction focused on freeways (Abdel-Aty et al., 2012; Ahmed et al., 2012b; Xu et al., 2013b) rather than urban arterials (Wang et al., 2015b; Yuan and Abdel-Aty, 2018). Improving traffic safety, especially for urban arterials, is becoming a major concern for traffic engineers and researchers.

Traffic data plays a crucial role in real-time crash likelihood prediction. Existing studies usually utilized fixed devices to obtain traffic data, such as Loop Detectors (Abdel-Aty et al., 2004; Xu et al., 2013b), Automatic Vehicle Identification (Ahmed et al., 2012b), Bluetooth detectors (Yuan and Abdel-Aty, 2018; Yuan et al., 2018), Cameras (Wang et al., 2019a), etc. Nevertheless, these devices usually require extra installation and regular maintenance, which increases the cost of obtaining data. Moreover, some devices are not reliable enough. For example, cameras are sensitive to lighting and weather conditions, while loop detectors are sensitive to pavement conditions. In addition, these devices are not flexible to depict the city-wide traffic conditions since they are fixed in certain positions. Recently, with the development

of mobile sensing technologies and mobile devices, it is now much easier to obtain vehicle-based data (e.g. trajectory data, smartphone data, CV data). Using the vehicle as the data collector, vehicle-based data can be conveniently obtained in wide range with low cost.

Vehicle trajectory data were widely used in the transportation management area, such as traffic congestion estimation (Kong et al., 2016), travel time reliability estimation (Uno et al., 2009), queue estimation (Cheng et al., 2011), etc. Nevertheless, only a few studies applied this new data to traffic safety analysis while most of them focused on analyzing the relationship between crashes and trajectory data (Wang et al., 2019b; Wang et al., 2019c; Wang et al., 2015b). For example, Wang et al. (2019c) utilized quasi-vehicle-trajectory data to investigate the relationship between transportation parameters and the crash occurrence on expressways. The authors indicated that speed difference-related variables had the significantly positive impact on crash occurrence, which reflected the vehicles' speed fluctuation. Similarly, Wang et al. (2015b) explored the relationship of taxi trajectory data to safety during peak and off-peak periods. Higher average speeds were found to be associated with higher crash frequencies during peak periods. Wang et al. (2019b) utilized the taxi trajectory data to predict crash likelihood at expressways. Variables such as average speed and speed difference ratio were generated from the trajectory data. The authors indicated that traffic status of the period between 5 min to 10 min before the crash affected the occurrence of crashes. In addition, existing studies only used one type of vehicles. The necessity of having a variety of vehicle types is illustrated by the previous studies. Basso et al. (2020) indicated that having access to data of different vehicle types could significantly improve the prediction accuracy. The application of vehicle trajectory data fusion on the real-time crash likelihood prediction still needs substantial investigation.

Two types of methods were commonly used by the existing studies on real-time crash likelihood prediction, which are statistical and machine learning. Statistical methods, such as Logistics Regression, Bayesian Logistics Regression, and Conditional Logistics Regression, were widely used during the early stages (Ahmed et al., 2014; Ahmed et al., 2012b). However, these methods usually have strong assumptions or high dependence on data preparation techniques. In addition, existing studies also indicated that the statistical methods were less accurate compared with the other two kinds of methods. Several traditional machine learning methods, such as SVM and random forest, have been explored by existing studies. Wang et al. (2019b) indicated that SVM outperformed Logistic Regression for crash likelihood prediction in terms of sensitivity and AUC value. While Yu and Abdel-Aty (2014a) found random forest could help the process of feature selection. However, it is still hard for these traditional machine learning methods to model massive high-dimensional traffic data (Guo et al., 2019). Recently, several studies (Guo et al., 2019; Islam et al., 2021; Khan and Ahmed, 2020; Li et al., 2020c; Yuan et al., 2019) applied deep learning methods to high-dimensional traffic data and achieved promising results compared with both statistical and traditional machine learning methods. Deep learning is a subfield of machine learning, with the ability to efficiently learn from large-scale and high-dimensional data. There are some popular deep learning architectures used by the literature, such as CNN, RNN, and LSTM. While LSTM was found to be especially useful for learning time-series traffic data because of its unique design (Khan and Ahmed, 2020; Yuan et al., 2019). In addition, existing studies also indicated that the performance of LSTM can be improved by augmenting it with a CNN component (Karim et al., 2018; Li et al., 2020c). However, there are currently no existing studies that applied the attention mechanism on crash

likelihood prediction, which could improve the performance of neural networks for learning long time-series data (Bahdanau et al., 2014; Guo et al., 2019).

In this study, to address the aforementioned shortcomings, we focus on predicting real-time crash likelihood for arterials using the deep learning and trajectory data fusion techniques. Trajectory data from different vehicles are collected and prepared. A Temporal Attention-based LSTM-CNN (TA-LSTM-CNN) is proposed for crash likelihood prediction. Experiments based on the real-world dataset indicate that the proposed methods can achieve promising prediction results. Our contributions can be summarized as follows:

- We investigate the application of trajectory data fusion on crash likelihood prediction with two real-world trajectory datasets.
- We design a dedicated method to extract traffic-related variables from the city-wide trajectory datasets.
- We propose an LSTM-CNN structure considering temporal attention, i.e., TA-LSTM-CNN, to improve prediction accuracy.
- Various baseline methods (e.g. logistics regression, XGBoost, random forest, LSTM-CNN) are developed to compare with the proposed method.

7.2 Data Preparation

7.2.1 Data Description

Two types of data are used in this study, crash data and trajectory data. Crash data are obtained through the S4A system. S4A archives the crash events of Florida with detailed information, such as crash time, location, severity, type, etc. The trajectory data have two data

sources, Lynx buses and Lytx fleet. Specifically, Lynx provides the public transportation service for Orange, Seminole, and Osceola counties in Florida. The Lynx trajectory data is obtained via the real-time data collection API from the DoubleMap®. This dataset contains 300 Lynx buses localized via an automatic vehicle locator system, with their locations, headings, route names, vehicle IDs, and other information updated every three seconds. Lytx provides industry-leading fleet and compliance management solutions for various partners. Lytx is cooperating with Orange Country, Florida for monitoring its fleet of 2,000 vehicles with various types, e.g. van, sedan, and truck. The county uses the Lytx's DriveCam integrated with a high-accuracy GPS unit (Nemat-Nasser and Smith, 2014) to obtain audio and video recordings inside and outside of the vehicle, as well as its real-time trajectory. The trajectory data are obtained from the Lytx Fleet Tracking Platform with an update frequency of 30 seconds. Since the Lynx and Lytx datasets have a similar data structure, this study combines their data description for simplicity (Table 20).

Table 20 Lynx and Lytx data description

Name	Description
ID	Unique integer for each vehicle.
Name	A text name for the vehicle.
Latitude	Latitude for the vehicle's current position.
Longitude	Longitude for the vehicle's current position.
Heading	The direction of the vehicle, in degrees (0-360).
Time	The time of the GPS update from the vehicle.
Route	The integer for bus route (only for Lynx)
Type	The type for vehicle status (only for Lytx)

7.2.2 Trajectory Data Preparation

Figure 39 shows the flowchart of trajectory data preparation. Lynx and Lytx trajectory data are prepared separately with a series of trip generation, trip cleaning, and speed estimation. A data fusion process is then implemented to fuse these two trajectory datasets spatially and temporally to generate the segment traffic data.

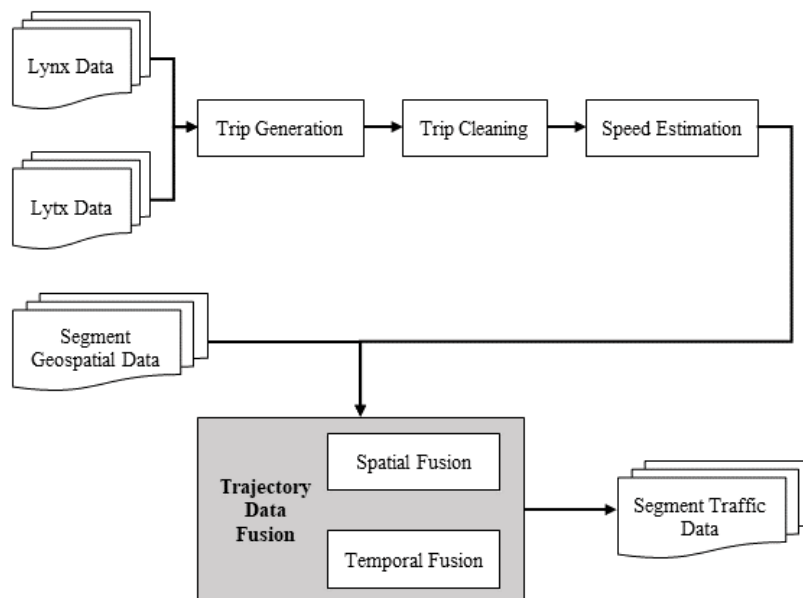


Figure 39 Trajectory data preparation flowchart

- **Trip Generation and Cleaning**

The duplicated records of the trajectory data are first removed, i.e., same vehicle ID, time, latitude, and longitude. Then, the data outside of the selected area were also removed. To estimate vehicles' speed, the trajectory data are divided into multiple trips. A trip is the collection of GPS coordinates for one vehicle during a continuous period. In the Lynx system, each vehicle is running continuously over its assigned route. A Lynx trip is generated based on

three control parameters, time, route, and vehicle ID (Figure 40 (a)). For example, the vehicle with ID as '17706388' has the 1st trip as route 541 from '2019-12-30 18:59:40' to '2019-12-30 19:21:36'. The Lytx data have one attribute as 'Type' with three values indicating the trip status. 'Trip Start' and 'Trip End' define the start and end point of one trip, while 'GPS Trial' represents the GPS coordinates of this trip (Figure 40 (b)). For example, the vehicle with ID as '14795' has the 1st trip from '2020-04-01 14:25:44' to '2020-04-01 15:25:10'. After trip generation, two types of trips are removed: 1) Short trip, a trip with less than 100 GPS coordinates or lasts less than 1 minute. 2) Stationary trip, a trip with all same GPS coordinates.

	id	time	route	lat	lon	
0	17706388	2019-12-30 18:59:40	541	28.4501791667	-81.4297721667	} Trip 1
1	17706388	2019-12-30 18:59:51	541	28.4501656667	-81.4288470000	
2	17706388	2019-12-30 19:00:10	541	28.4501653333	-81.4287498333	
3	17706388	2019-12-30 19:01:40	541	28.4494376667	-81.4280050000	
4	17706388	2019-12-30 19:02:00	541	28.4463965000	-81.4279718333	
...	
54	17706388	2019-12-30 19:21:06	541	28.4447541667	-81.3993501667	} Trip 2
55	17706388	2019-12-30 19:21:36	541	28.4435733333	-81.4000973333	
56	17706388	2019-12-30 19:22:14	544	28.4457706667	-81.4003768333	
57	17706388	2019-12-30 19:23:08	544	28.4457743333	-81.4003386667	
58	17706388	2019-12-30 19:31:47	544	28.4458793333	-81.4003036667	
...	} Trip N

(a) Lynx trip generation

	ID	Time	Type	Latitude	Longitude	
0	14795	2020-04-01 14:25:44	Trip Start	28.6152330	-81.4274180	} Trip 1
1	14795	2020-04-01 14:26:18	GPS Trail	28.6141000	-81.4274100	
2	14795	2020-04-01 14:26:49	GPS Trail	28.6129380	-81.4315220	
3	14795	2020-04-01 14:27:19	GPS Trail	28.6127070	-81.4321830	
4	14795	2020-04-01 14:27:50	GPS Trail	28.6107550	-81.4291250	
...	
64	14795	2020-04-01 15:23:13	GPS Trail	28.4926500	-81.4318370	} Trip N
65	14795	2020-04-01 15:23:44	GPS Trail	28.4953150	-81.4352020	
66	14795	2020-04-01 15:24:14	GPS Trail	28.4991950	-81.4323180	
67	14795	2020-04-01 15:24:45	GPS Trail	28.4994980	-81.4328200	
68	14795	2020-04-01 15:25:10	Trip End	28.4994300	-81.4328730	
...	

(b) Lytx trip generation

Figure 40 Trip generation illustration

- **Speed Calculation**

The vehicle's instantaneous speed is estimated from the generated trips. For each trip, the speed of vehicle i between two continuous GPS coordinates is calculated according to equation (53).

$$Speed_i = \frac{distance(GPS_{t_1}, GPS_{t_2})}{t_2 - t_1} \quad (53)$$

Where GPS_{t_1} is the vehicle's location at the time t_1 in longitude and latitude, and GPS_{t_2} is its location at the time t_2 in longitude and latitude. The distance d is estimated using the Haversine formula (equations (54)-(58)) (Veness, 2019) since this study uses the World Geodetic System (WGS84). Specifically, φ and λ are latitude and longitude in radians. For example, φ_{t_1} and λ_{t_1} are the latitude and longitude of GPS_{t_1} in radians. R is the earth's radius (6,371 km). After the speed is estimated, the Savitzky-Golay filter is applied to smooth the speed and remove noise. This filter was commonly used in similar studies and achieved promising results (Stipancic et al., 2018a; Zaki et al., 2014).

$$\Delta\varphi = \varphi_{t_2} - \varphi_{t_1} \quad (54)$$

$$\Delta\lambda = \lambda_{t_2} - \lambda_{t_1} \quad (55)$$

$$c = 2 \cdot \arctan2(\sqrt{a}, \sqrt{1-a}) \quad (56)$$

$$a = \sin^2\left(\frac{\Delta\varphi}{2}\right) + \cos(\varphi_{t_1}) \cdot \cos(\varphi_{t_2}) \cdot \sin^2\left(\frac{\Delta\lambda}{2}\right) \quad (57)$$

$$d = R \cdot c \quad (58)$$

7.2.3 Trajectory Data Fusion

The benefit of data fusion is to better reflect road segments' traffic conditions with a variety of vehicles. After speed estimation, the two trajectory datasets are fused spatially and temporally, which transform the vehicle-based data into segment-based data. First, QGIS (QGIS.org, 2016) is applied to geocode the trajectory and segment data. The trajectory data are

matched to the closest road segments with *R* (R Core Team, 2013), utilizing the spatial analysis package *sf* (Pebesma, 2018). In addition, the data within 250 feet of intersections are removed to exclude the impact of intersections. Then, for each road segment, the Lynx and Lytx road are fused based on the time of each record. To describe the road segments' traffic conditions from different perspectives, several descriptive variables (e.g. average speed, the standard deviation of speed, and different percentages of speed) are generated in 5-min intervals using the instantaneous speed of Lynx and Lytx. These variables were commonly used by the previous studies and were found to be associated with the crash occurrence (Li et al., 2020b; Wang et al., 2019b; Wang et al., 2019c; Wang et al., 2015b).

7.2.4 Crash Data Preparation

First, crashes caused by drugs and alcohol were removed, since these crashes are not usually related to the traffic characteristics. Second, considering the influence of the intersections, crashes within 250 feet of intersections are removed. Crashes are then matched to the corresponding road segments based on their location using QGIS (QGIS.org, 2016). To predict the crash likelihood, this study uses the traffic 5–10 min before the crash time as potential crash contributing traffic conditions (Wang et al., 2019b; Wang et al., 2019c). For example, if a crash happened at 11:50, the traffic safety status from 11:40 to 11:45 is labeled as '1', indicating that a crash will occur in the next 5-10 minutes. Otherwise, the traffic safety status is labeled as '0'. The traffic data are then used to model traffic safety status. In addition, due to the impact of crash events on traffic conditions, traffic data within two hours after crash events are removed.

7.3 Research Methodologies

7.3.1 LSTM with Attention Mechanism

LSTM is one type of RNNs, which was designed to solve the vanishing gradients problem by introducing memory gates to control when to forget and remember certain information (Hochreiter and Schmidhuber, 1997). Although there are many variants of LSTM, existing studies also indicated that none of them could significantly improve the performance of the standard LSTM (Greff et al., 2017). Therefore, the standard LSTM is adopted in this study. The structure of LSTM cells is shown in Figure 41. The LSTM has three gates to control the information flow: 1) Forget gate f_t controls how much to forget from the previous step memory cell. 2) Input gate i_t determines which values to be updated. 3) Output gate o_t determines the output for the hidden state h_t .

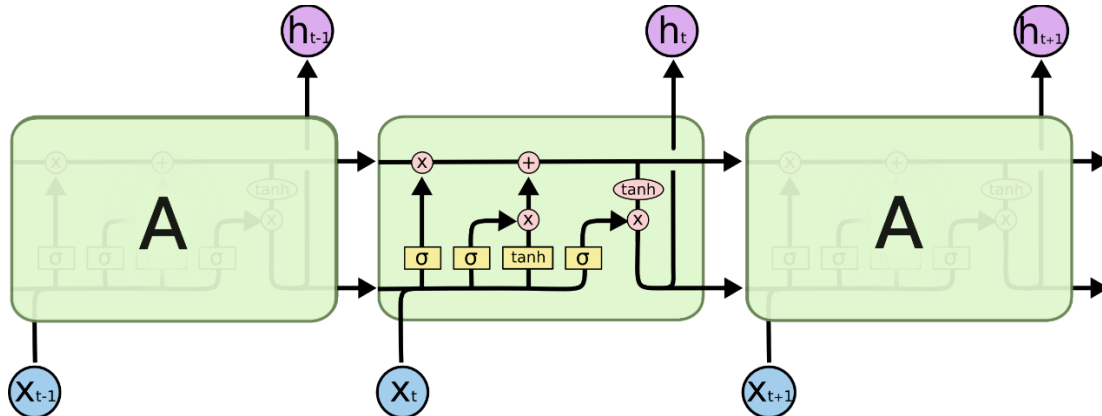


Figure 41 LSTM structure (Christopher, 2015)

The essential equations of LSTM are shown below. If the input of LSTM is denoted as $X = (x_1, x_2, \dots, x_t)$, where t is the prediction period, $h = (h_1, h_2, \dots, h_t)$ is the hidden state, and the

$y = (y_1, y_2, \dots, y_t)$ is the output. The essential equations for LSTM are shown from equations (59)-(65) (Christopher, 2015).

$$f_t = \sigma(W_f \cdot [h_{t-1}; x_t] + b_f) \quad (59)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}; x_t] + b_i) \quad (60)$$

$$\tilde{c}_t = \tanh(W_r \cdot [h_{t-1}; w_t] + b_c) \quad (61)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (62)$$

$$o_t = \sigma(W_o \cdot [h_{t-1}; x_t] + b_o) \quad (63)$$

$$h_t = o_t \odot \tanh(c_t) \quad (64)$$

$$y_t = W_y h_{t-1} + b_y \quad (65)$$

Where W represents weight matrices, for example, W_i denotes the weight matrix from the input gate to the input, σ is the logistic sigmoid function, c is the context vector, h is the hidden state, and \odot indicates the elementwise product of the vectors. In addition, to help LSTM better learn from long time-series data, Bahdanau et al. (2014) proposed the Attention mechanism, which originally aimed to solve machine translation problems. The purpose of using the attention mechanism is to select the information that is relatively critical to the current task from all inputs (Guo et al., 2019). This study applies a temporal Attention mechanism (equation (66)-(70)) from the proposed two-layer LSTM (Figure 42) (Luong et al., 2015).

$$\alpha_s = \text{softmax}(\text{score}(h_t, h_s)) \quad (66)$$

$$\text{score}(h_t, h_s) = h_t^T h_s \quad (67)$$

$$c_t = \sum_{s=1}^t \alpha_s h_s \quad (68)$$

$$\tilde{h}_t = \tanh(W_c [c_t; h_t]) \quad (69)$$

$$y_t = W_y \tilde{h}_t + b_y \quad (70)$$

Where α_s is the attention weights, c_t is the context vector, a_t is the attention vector, h_t is the target hidden state, h_s is the source hidden states, and \tilde{h}_t is the attention hidden state. After using the attention mechanism, the output of the LSTM is updated from (65) to (70).

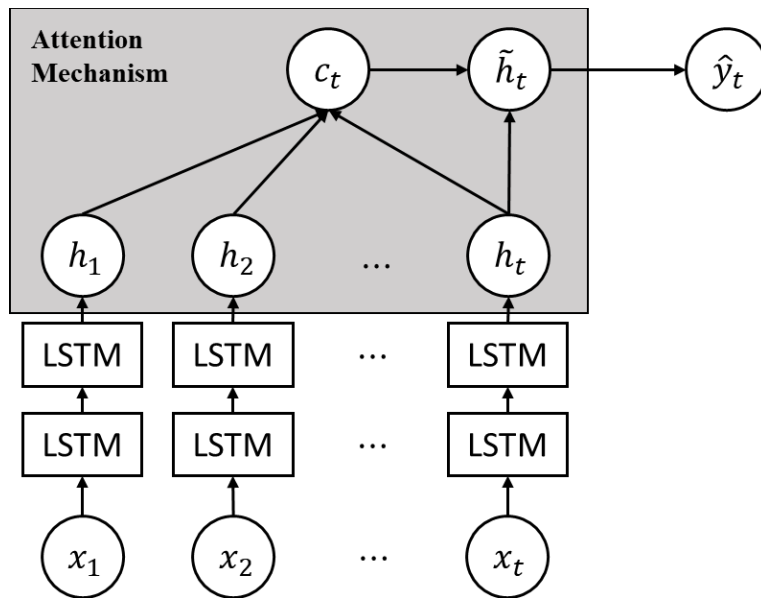


Figure 42 TA-LSTM structure

7.3.2 CNN

The concept of CNN was originally developed for computer vision areas. Recently, some studies also indicated that CNN could also be utilized for time-series data with promising results (Li, 2020; Li et al., 2020c). The key component for CNN is its convolution layer, where CNN applies a filter to extract features through the input data. This study adopts a 1-D CNN as the feature extractor with ReLU as the activation function. The benefit of using CNN for time-series data is its weight sharing ability, which allows CNN to learn features that are invariant across the time dimension.

7.3.3 TA-LSTM-CNN

The network architecture of the proposed TA-LSTM-CNN is shown in Figure 43. The network has one input layer and one output layer, sigmoid function is used as the activation function since the crash prediction is a binary classification problem. The network's hidden layers have two components, LSTM and CNN. Specifically, two LSTM layers are stacked together, followed by one attention layer and a dropout layer to prevent overfitting. The CNN component has two CNN layers, each followed by a Batch Normalization (BN) layer (Ioffe and Szegedy, 2015) to improve the training speed and network stability. In addition, a pooling layer is added to reduce the number of parameters. The results from the LSTM and CNN components are concatenated to generate the outputs. In addition, the L2-norm regularization term is used for both LSTM and CNN to prevent overfitting issues.

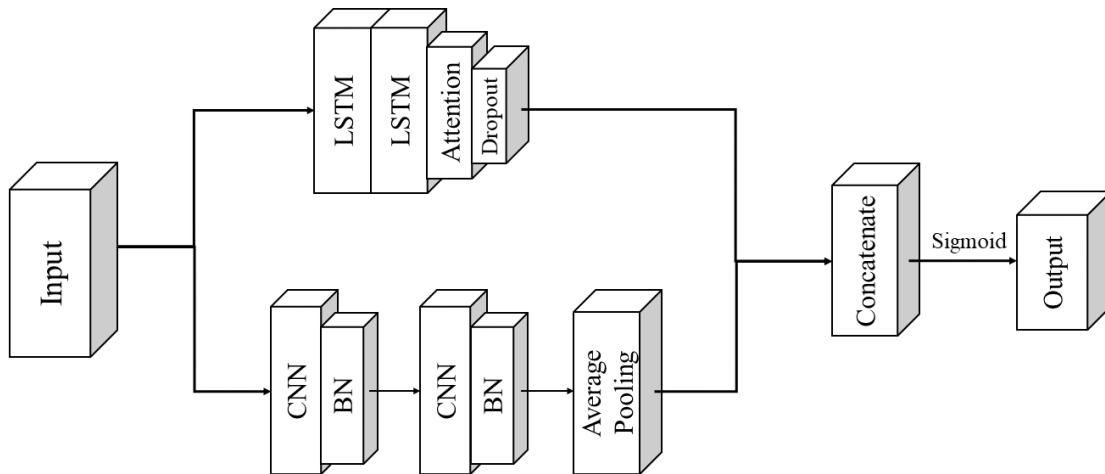
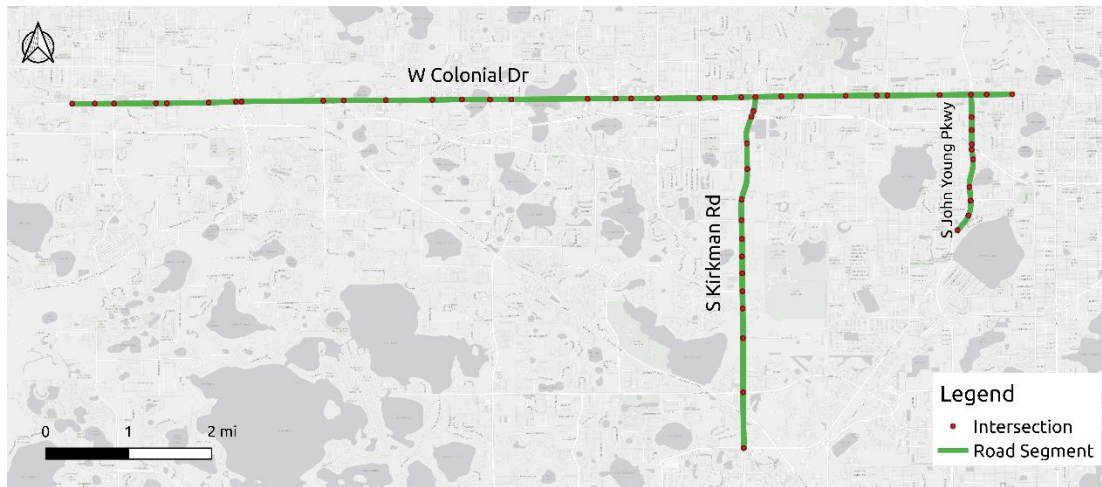


Figure 43 Network architecture

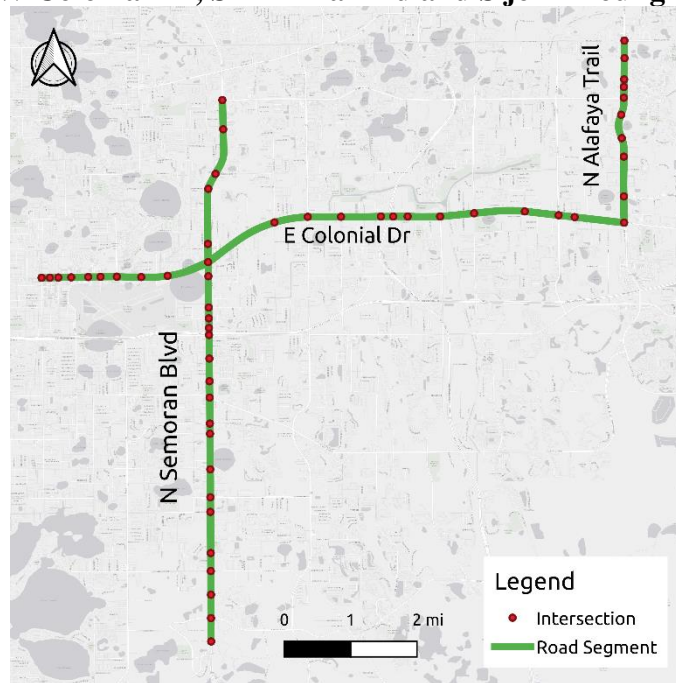
7.4 Experiment and Results

7.4.1 Site Selection and Data Preparation

This study aims to predict the crash likelihood for urban arterial road segments. The selected arterials are shown in Figure 44. Considering the data availability and geometry characteristics, six arterials are selected from Orlando, FL, including 208 road segments and 110 signalized intersections. A segment is defined as the road facility between two consecutive intersections in a certain direction. The road segment geospatial data are obtained from the Florida Department of Transportation to ensure data accuracy. Trajectory and crash data are collected from December 2019 to June 2020, data from March and April are removed due to the impact of COVID-19. In total, this study collects 3,151,169 Lynx records, 138,285 Lytx records, and 162 crash events.



(a) W Colonial Dr, S Kirkman Rd and S John Young Pkwy



(b) E Colonial Dr, N Semoran Blvd, and N Alafaya Trail

Figure 44 Selected urban arterials

The obtained trajectory and crash data are processed using the proposed data preparation approaches. After matching the trajectory data to the corresponding road segments, the descriptive statistics of road traffic variables are shown in Table 21.

Table 21 Descriptive statistics table

Variables	Description	Mean	Std. Dev.
Avg_speed	Average speed	18.98	10.88
Std_speed	The standard deviation of speed	9.03	6.36
85th_speed	85 th percentile speed	22.25	11.33
50th_speed	50 th percentile speed	18.86	11.15
15th_speed	15 th percentile speed	15.72	11.36

Note: All the variables have the unit as mph

7.4.2 Experimental Design

Since TA-LSTM-CNN requires 3-D data as input. The traffic-related data are prepared as (*sample size, time step, feature number*), where the sample size is the number of the input data, the time step is 10 minutes, and the number of features is 6. The prepared data (Figure 45) are first divided into training (70%) and test (30%). In addition, since crashes are extremely rare events, this study uses SMOTE (Chawla et al., 2002) to generate new crash events and balance the crash to non-crash ratio. SMOTE uses a nearest neighbors' algorithm to generate new data, which synthesizes new minority instances between real minority instances. This method was widely utilized by similar studies (Li et al., 2020b; Yuan et al., 2019) and achieved promising results. After crash oversampling, the proposed model is implemented on the training data and evaluated on the test data. The model's performance on the test data is used for hyperparameters tuning, while the optimal model is selected accordingly. In addition, to illustrate the effect of trajectory data fusion, the proposed model is implemented on two scenarios, one is data fusion, while another one is a single dataset containing only Lynx data. The proposed methods are trained and evaluated on a single NVIDIA GeForce RTX 1080 Ti with 11 GB Memory.

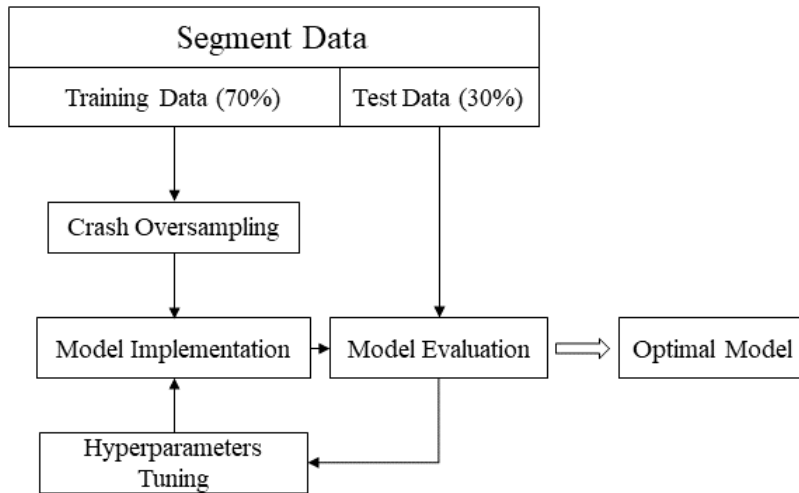


Figure 45 Model training process

Three common metrics are used for model evaluation, including sensitivity, AUC, and false alarm rate. Sensitivity (36) and false alarm rate (37) are derived from the classification confusion matrix (Table 13). AUC measures the area under the Receiver Operating Characteristic (ROC) curve, which plots the true positive rate and the false positive rate at different classification thresholds. Moreover, hyperparameters tuning is a crucial step for implementing deep learning methods. Nine hyperparameters are tuned for the proposed model, which are shown in Table 22. The early stopping strategy is adopted to accelerate training speed and prevent overfitting.

Table 22 Hyperparameters tuning

Name	Range	Value
Learning Rate	0.1, 0.001, 0.0001	0.0001
Optimization Function	SGD, Adam, RMSprop	RMSprop
LSTM Unit Number 1	128, 64, 32, 16	64
LSTM Unit Number 2	128, 64, 32, 16	32
Dropout Rate	0.3, 0.5, 0.6, 0.8	0.6
CNN Filter Size 1	128, 64, 32, 16	64
CNN Filter Size 2	128, 64, 32, 16	32
Batch Size	100, 200, 500, 1000, 3000	200
Epoch Number	50, 80, 100, 150	50

In addition, two types of methods (i.e., statistical and machine learning methods) are used as the baseline models to compare with the proposed method. Logistics regression is selected as the statistical method. XGBoost and random forest are used as the machine learning methods, while standard LSTM-CNN without attention mechanism is used as the deep learning method. The baseline models are trained and tested on the same data as the proposed method. In terms of hyperparameters tuning, seven hyperparameters are tuned for XGBoost using grid search, including the number of trees, maximum tree depth for base learners, gamma (minimum loss reduction required to make a further partition on a leaf node of the tree), learning rate, etc. For random forest, three hyperparameters are tuned, which are the number of trees, maximum tree depth, and min_samples_split (minimum number of samples required to split an internal node). For LSTM-CNN, the same hyperparameters (Table 4) as the proposed TA-LSTM-CNN are tuned.

7.4.3 Results

After training and testing, the results of hyperparameters tuning are shown in Table 4. To obtain the best classification threshold for estimation evaluation metrics, this study selects the

optimal threshold when the difference between sensitivity and specificity reaches the smallest value. For the data fusion scenario, the proposed TA-LSTM-CNN has a sensitivity of 0.847, AUC as 0.848, and False Alarm Rate as 0.151 on the test data. For the same model on the single data scenario, the model has a sensitivity as 0.792, AUC as 0.792, and False Alarm Rate as 0.208 on the test data. The results indicate that trajectory data fusion could improve the accuracy of real-time crash likelihood prediction since it better reflects the traffic conditions from a broader perspective. In addition, the model’s confusion matrix on the data fusion scenario is shown in Figure 46.

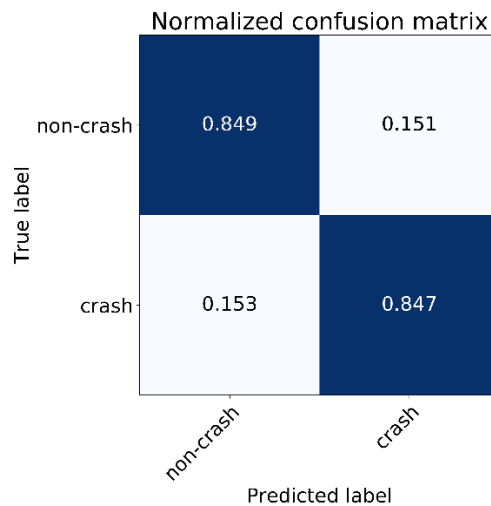
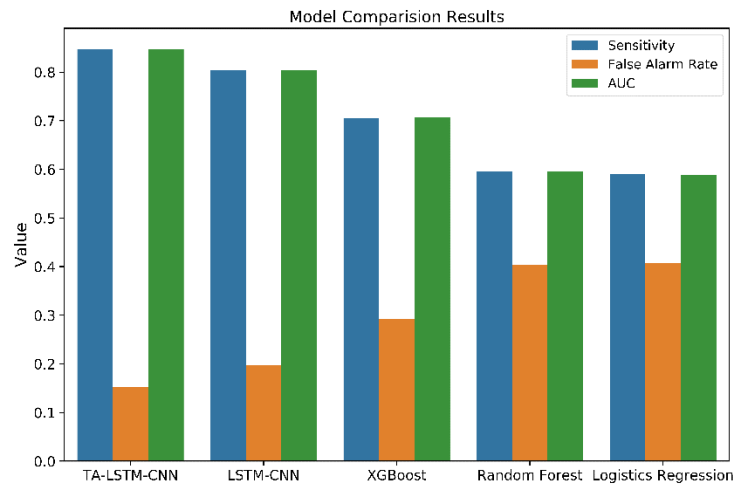


Figure 46 Confusion matrix

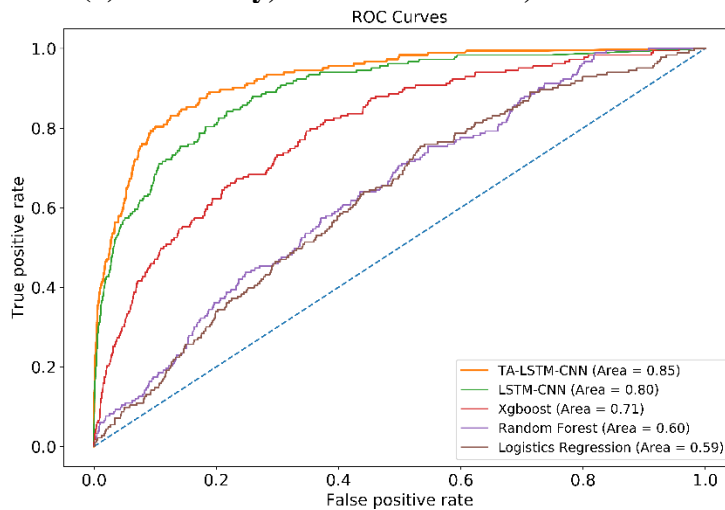
7.4.4 Model Comparison

The results from the model comparison are presented in Figure 47. The proposed TA-LSTM-CNN model has the highest Sensitivity and AUC with the lowest False Alarm Rate. The introduction of the temporal attention mechanism improves the performance of the standard LSTM-CNN. For example, the sensitivity is increased from 0.803 to 0.847 and the AUC is

improved from 0.803 to 0.848. XGBoost has the best performance (e.g. sensitivity: 0.705, AUC: 0.706) among the machine learning and statistical methods, while random forest and logistics regression have similar results. The reason for the relatively poor performance of logistics regression may be due to the difficulty of learning high-dimensional data.



(a) Sensitivity, False Alarm Rate, and AUC



(b) ROC Curves

Figure 47 Model comparison results

7.5 Conclusions

Real-time crash likelihood prediction is an important component of the proactive traffic safety management system. Detector-based data were widely used in the previous studies, while few of them investigated the application of novel trajectory data. In addition, deep learning methods were used by some studies recently with outstanding results compared with traditional statistical and machine learning methods. The temporal attention mechanism is a powerful component for learning time-series data while no existing studies considered it for prediction real-time crash likelihood.

This study proposed an LSTM-CNN with a temporal attention mechanism (TA-LSTM-CNN) for predicting real-time crash likelihood at urban arterials. Two real-world trajectory data were prepared with a series of trip generation, speed estimation, and data fusion. The vehicle-based datasets were transformed to segment data while describing traffic conditions from different perspectives. The TA-LSTM-CNN was fine-tuned with various hyperparameters. Extensive experimental results showed that the data fusion scenario achieved better results compared with the single data scenario. Besides, the temporal attention mechanism could improve the performance of the standard LSTM-CNN with higher sensitivity, AUC, and lower false alarm rate. The results from model comparisons also indicated that the proposed TA-LSTM-CNN outperformed other baseline models (e.g. logistics regression, XGBoost, etc.) with various evaluation metrics.

In summary, by integrating trajectory data from different types of vehicles spatially and temporally, data fusion was proved to be a desirable approach to prepare large-scale trajectory data for crash prediction application. In addition, the temporal attention mechanism can help the

LSTM learn long time-series data, which is a suitable choice for learning high-dimensional traffic data. The proposed approaches have the potentials to be extended to other types of vehicles also since they share common characteristics. With the development of connected vehicle systems, it will be much easier to obtain vehicle trajectory data in the future. In the future, additional improvements will be conducted on improving the model to incorporate spatial features. Besides, the transferability of the proposed methods can be further validated by implementing them for different locations.

CHAPTER 8: CONCLUSIONS

This dissertation aims to conduct real-time traffic safety evaluation in the context of connected vehicles and mobile sensing. This dissertation has conducted a comprehensive study in terms of the application of vehicle-based data (e.g., trajectory data, sensor data) to traffic safety. Different machine learning methods were proposed for identifying driving maneuvers and predicting crash likelihood. The conclusions of this dissertation are summarized as below:

In Chapter 3, this dissertation developed a vehicle maneuvers detection system using a common smartphone with GPS, gyroscope, accelerometer, and magnetometer sensors. A stacked-LSTM model was built to detect the vehicle maneuvers considering the time-dependency of the sensor data. This dissertation compared the performance of the proposed system with previous studies and various machine learning methods, including LightGBM, KNN, SVM, and random forest. Extensive experimental results indicated that the proposed system accurately detected different vehicle maneuvers with an average F1-score of 0.98, precision of 0.97, and recall of 0.98, which outperformed the counterparts. Moreover, the model can be easily transferred to different drivers and locations. The system is robust and suitable for real-time application as it requires simple processing of smartphone sensor data.

In Chapter 4, this dissertation proposed a semi-supervised deep learning method to learn from the massive unlabeled sensor data. Data from a smartphone's accelerometer and gyroscope were collected by different drivers with a variety of smartphones, vehicles, and locations. Three LSTM models were trained with the proposed semi-supervised learning algorithm. Experimental results indicated that the proposed semi-supervised LSTM could learn from the unlabeled data and achieve outstanding results with only a small portion of the labeled data. Using much fewer

labeled data, semi-supervised LSTM could achieve similar results compared with the supervised method. Moreover, the proposed method outperformed other machine learning methods (e.g. convolutional neural network, XGBoost, random forest) in terms of precision, recall, F1-score, and AUC. More and more traffic-data will be available in the future, the proposed method is expected to make use of the undiscovered potential from the massive unlabeled data.

In Chapter 5, this dissertation introduced a novel CV emulated data for real-time crash potential prediction. Different from the fixed devices' data, CV emulated data have high flexibility and can be obtained continuously with relatively low cost. Crash and CV emulated data were collected from two urban arterials in Orlando, USA. Crash data were archived by the S4A, while the CV emulated data were obtained through the data collection API with a high frequency. Different data cleaning and preparation techniques were implemented, while various speed-related variables were generated from the CV emulated data. An LSTM neural network was trained to predict the crash potential in the next 5-10 minutes. The results from the model illustrated the feasibility of using a novel CV emulated data to predict real-time crash potential. The average and 50th percentile speed were the two most important variables for the crash potential prediction. In addition, the proposed LSTM outperformed Bayesian logistics regression and XGBoost in terms of sensitivity, AUC, and false alarm rate. With the rapid development of connected vehicle systems, the results from this dissertation can be extended to other types of vehicles and data, which can significantly enhance traffic safety.

In Chapter 6, this dissertation utilized critical bus driving events extracted from GPS trajectory data as pedestrian and bicycle surrogate safety measures for bus stops. A city-wide trajectory data from Orlando, Florida was used, which contains around 300 buses, 6,700,000

GPS records, and 1,300 bus stops. Three critical driving events were identified based on the buses' acceleration rates and stop time; hard acceleration, hard deceleration, and long stop. The relationships between critical driving events and crashes were examined using Spearman's rank correlation coefficient. All three events were positively correlated with pedestrian and bicycle crashes. Long stop event has the highest correlation coefficient, followed by hard acceleration and hard deceleration. A Bayesian negative binomial model incorporating spatial correlation (Bayesian NB-CAR) was built to estimate the pedestrian and bicycle crash frequency using the generated events. The results were consistent with the correlation estimation. For example, hard acceleration and long stop events were both positively related to pedestrian and bicycle crashes. Moreover, model evaluation results indicated that the proposed Bayesian NB-CAR outperformed the standard Bayesian negative binomial model with lower WAIC and DIC values. In conclusion, this dissertation suggests the use of critical bus driving events as surrogate safety measures for pedestrian and bicycle crashes, which could be implemented in a proactive traffic safety management system.

In Chapter 7, this dissertation proposed an LSTM-CNN with temporal attention mechanism (TA-LSTM-CNN) for predicting real-time crash likelihood at urban arterials. Two real-world trajectory data were prepared with a series of trip generation, speed estimation, and data fusion. The vehicle-based datasets were transformed to segment data while describing traffic conditions from different perspectives. The TA-LSTM-CNN was fine-tuned with various hyperparameters. Extensive experimental results showed that the data fusion scenario achieved better results compared with the single data scenario. Besides, the temporal attention mechanism could improve the performance of the standard LSTM-CNN with higher sensitivity, AUC, and

lower false alarm rate. The results from model comparisons also indicated that the proposed TA-LSTM-CNN outperformed other baseline models (e.g. logistics regression, XGBoost, etc.) with various evaluation metrics.

The implications of this dissertation are summarized as follows: first, this dissertation proved the feasibility of using vehicle-based data (i.e. trajectory data) for real-time crash likelihood prediction. Besides, models developed using one vehicle-based data could be conveniently extended to other vehicle-based data due to the similarity of data structure. Second, the proposed deep learning models could be implemented into a real-time crash likelihood prediction system, which takes the trajectory data and predicts the crash likelihood every minute. The prediction results could be used by traffic operators to improve traffic safety in a proactive way. Third, several critical driving events were proposed and verified as surrogate safety measures for pedestrian and bicycle crashes. The number of pedestrian and bicycle crashes could be estimated using the proposed measures based on trajectory data. Decision makers could use the estimated results to identify dangerous locations and propose certain strategies to improve pedestrian and bicycle safety. In summary, this dissertation brings novel insights into the application of vehicle-based data to traffic safety. The results from this dissertation are ready to be implemented into a traffic safety evaluation system, which aims to assist decision makers and traffic operators to improve traffic safety.

REFERENCES

- Abdel-Aty, M., Abdalla, M.F., 2004. Linking Roadway Geometrics and Real-Time Traffic Characteristics to Model Daytime Freeway Crashes: Generalized Estimating Equations for Correlated Data. *Transportation Research Record* 1897(1), 106-115.
- Abdel-Aty, M., Pande, A., 2005. Identifying crash propensity using specific traffic speed conditions. *Journal of safety Research* 36(1), 97-108.
- Abdel-Aty, M., Uddin, N., Pande, A., Abdalla, F., Hsia, L., 2004. Predicting freeway crashes from loop detector data by matched case-control logistic regression. *Transportation Research Record: Journal of the Transportation Research Board*(1897), 88-95.
- Abdel-Aty, M.A., Cai, Q., Agarwal, S., Islam, Z., Li, P., Zhang, S., Hasan, D., Huang, J., 2020. Using Smartphone as On-board unit (OBU) Emulator Implementation Study.
- Abdel-Aty, M.A., Hassan, H.M., Ahmed, M., Al-Ghamdi, A.S., 2012. Real-time prediction of visibility related crashes. *Transportation Research Part C: Emerging Technologies* 24, 288-298.
- af Wåhlberg, A.E., 2004. The stability of driver acceleration behavior, and a replication of its relation to bus accidents. *Accident Analysis & Prevention* 36(1), 83-92.
- Ahmed, M., Abdel-Aty, M., Yu, R., 2012a. Bayesian Updating Approach for Real-Time Safety Evaluation with Automatic Vehicle Identification Data. *Transportation Research Record: Journal of the Transportation Research Board* 2280, 60-67.
- Ahmed, M.M., Abdel-Aty, M., Lee, J., Yu, R., 2014. Real-time assessment of fog-related crashes using airport weather data: a feasibility analysis. *Accident Analysis & Prevention* 72, 309-317.
- Ahmed, M.M., Abdel-Aty, M., Yu, R., 2012b. Assessment of Interaction of Crash Occurrence, Mountainous Freeway Geometry, Real-Time Weather, and Traffic Data. *Transportation Research Record* 2280(1), 51-59.
- Ahmed, M.M., Abdel-Aty, M.A., 2012. The Viability of Using Automatic Vehicle Identification Data for Real-Time Crash Prediction. *IEEE Transactions on Intelligent Transportation Systems* 13(2), 459-468.
- Apple, 2020. Developer Documentation.
- Bahdanau, D., Cho, K., Bengio, Y., 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint*

- Bao, J., Liu, P., Ukkusuri, S.V., 2019. A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data. *Accident Analysis & Prevention* 122, 239-254.
- Basso, F., Basso, L.J., Pezoa, R., 2020. The importance of flow composition in real-time crash prediction. *Accident Analysis & Prevention* 137, 105436.
- Bejani, M.M., Ghatee, M., 2020. Convolutional Neural Network With Adaptive Regularization to Classify Driving Styles on Smartphones. *IEEE Transactions on Intelligent Transportation Systems* 21(2), 543-552.
- Bengio, Y., Simard, P., Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks* 5(2), 157-166.
- Besag, J., York, J., Mollié, A., 1991. Bayesian image restoration, with two applications in spatial statistics. *Annals of the Institute of Statistical Mathematics* 43(1), 1-20.
- Bhoraskar, R., Vankadhara, N., Raman, B., Kulkarni, P., 2012. Wolverine: Traffic and road condition estimation using smartphone sensors, *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*. IEEE, pp. 1-6.
- Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training, *Proceedings of the eleventh annual conference on Computational learning theory*. Association for Computing Machinery, Madison, Wisconsin, USA, pp. 92–100.
- Boonsiripant, S., Rodgers, M.O., Hunter, M.P., 2011. Speed Profile Variation as a Road Network Screening Tool. *Transportation Research Record: Journal of the Transportation Research Board* 2236(1), 83-91.
- Cafiso, S., Di Graziano, A., Pappalardo, G., 2013. Road safety issues for bus transport management. *Accident Analysis & Prevention* 60, 324-333.
- Carvalho, E., Ferreira, B.V., Ferreira, J., Souza, C.d., Carvalho, H.V., Suhara, Y., Pentland, A.S., Pessin, G., 2017. Exploiting the use of recurrent neural networks for driver behavior profiling, *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3016-3021.
- Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P., 2002. SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 16, 321-357.
- Chen, D., Cho, K.-T., Han, S., Jin, Z., Shin, K.G., 2015. Invisible Sensing of Vehicle Steering with Smartphones, *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, Florence, Italy, pp. 1-13.

- Chen, P., Zhou, J., 2016. Effects of the built environment on automobile-involved pedestrian crash frequency and risk. *Journal of Transport & Health* 3(4), 448-456.
- Chen, Q., Song, X., Yamada, H., Shibasaki, R., 2016. Learning deep representation from big and heterogeneous data for traffic accident inference, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. AAAI Press, Phoenix, Arizona, pp. 338-344.
- Chen, T., Guestrin, C., 2016. XGBoost: A Scalable Tree Boosting System, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Association for Computing Machinery, San Francisco, California, USA, pp. 785–794.
- Cheng, Y., Qin, X., Jin, J., Ran, B., Anderson, J., 2011. Cycle-by-Cycle Queue Length Estimation for Signalized Intersections Using Sampled Trajectory Data. *Transportation Research Record: Journal of the Transportation Research Board* 2257(1), 87-94.
- Cheranchery, M.F., Bhattacharyya, K., Maitra, B., Boltze, M., 2016. Assessing safety level of bus stops in the absence of crash data, *Transportation Research Board 95th Annual Meeting*.
- Chika, S., Yasuhisa, N., Takuya, H., 2008. Prototype of pedestrian-to-vehicle communication system for the prevention of pedestrian accidents using both 3G wireless and WLAN communication, *2008 3rd International Symposium on Wireless Pervasive Computing*, pp. 764-767.
- Chollet, F., 2015. Keras.
- Chowdhury, A., Chakravarty, T., Balamuralidhar, P., 2014. A novel approach to improve vehicle speed estimation using smartphone's INS/GPS sensors, *Proceedings of the 8th International Conference on Sensing Technology*, pp. 2-4.
- Christopher, O., 2015. Understanding lstm networks.
- Cutulenco, G., 2019. Making Connected Vehicles Generates ~10x Data as Driving Them, engineering.com.
- Denwood, M., 2016. runjags: An R package providing interface utilities, model templates, parallel computing methods and additional distributions for MCMC models in JAGS. *Journal of Statistical Software* 71(9), 1-25.
- Doshi, A., Trivedi, M.M., 2010. Examining the impact of driving style on the predictability and responsiveness of the driver: Real-world and simulator analysis, *2010 IEEE Intelligent Vehicles Symposium*, pp. 232-237.
- Duarte, G.O., Gonçalves, G.A., Farias, T.L., 2013. Vehicle monitoring for driver training in bus companies – Application in two case studies in Portugal. *Transportation Research Part D: Transport and Environment* 18, 103-109.

- Ferreira, J.J., 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), e0174959.
- Fu, T., Miranda-Moreno, L., Saunier, N., 2016. Pedestrian Crosswalk Safety at Nonsignalized Crossings During Nighttime: Use of Thermal Video Data and Surrogate Safety Measures. *Transportation Research Record* 2586(1), 90-99.
- Fugiglando, U., Massaro, E., Santi, P., Milardo, S., Abida, K., Stahlmann, R., Netter, F., Ratti, C., 2019. Driving Behavior Analysis through CAN Bus Data in an Uncontrolled Environment. *IEEE Transactions on Intelligent Transportation Systems* 20(2), 737-748.
- Gelman, A., Hwang, J., Vehtari, A., 2014. Understanding predictive information criteria for Bayesian models. *Statistics and Computing* 24(6), 997-1016.
- Geurts, P., Ernst, D., Wehenkel, L., 2006. Extremely randomized trees. *Machine learning* 63(1), 3-42.
- Google, 2019. Motion sensors.
- Google, 2020. Developer Guides.
- Graves, A., Mohamed, A., Hinton, G., 2013. Speech recognition with deep recurrent neural networks, *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645-6649.
- Greff, K., Srivastava, R.K., Koutník, J., Steunebrink, B.R., Schmidhuber, J., 2017. LSTM: A Search Space Odyssey. *IEEE Transactions on Neural Networks and Learning Systems* 28(10), 2222-2232.
- Gu, X., Yan, X., Ma, L., Liu, X., 2020. Modeling the service-route-based crash frequency by a spatiotemporal-random-effect zero-inflated negative binomial model: An empirical analysis for bus-involved crashes. *Accid Anal Prev* 144, 105674.
- Guo, S., Lin, Y., Feng, N., Song, C., Wan, H., 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 922-929.
- Hassan, H.M., Abdel-Aty, M.A., 2013. Predicting reduced visibility related crashes on freeways using real-time traffic flow data. *Journal of safety research* 45, 29-36.
- Herring, R., Hofleitner, A., Abbeel, P., Bayen, A., 2010. Estimating arterial traffic conditions using sparse probe data, *13th International IEEE Conference on Intelligent Transportation Systems*, pp. 929-936.

- Hess, P., Moudon, A., Matlick, J., 2004. Pedestrian Safety and Transit Corridors. *Journal of Public Transportation* 7(2), 73-93.
- Hochreiter, S., 1991. Untersuchungen zu dynamischen neuronalen Netzen. Technische Universität München.
- Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. *Neural computation* 9(8), 1735-1780.
- Hossain, M., Muromachi, Y., 2012. A Bayesian network based framework for real-time crash prediction on the basic freeway segments of urban expressways. *Accident Analysis & Prevention* 45, 373-381.
- Huang, K., Chiu, P., Tsai, H., Kuo, C., Lee, H., Wang, Y.F., 2016. RedEye: Preventing Collisions Caused by Red-Light Running Scooters With Smartphones. *IEEE Transactions on Intelligent Transportation Systems* 17(5), 1243-1257.
- IIHS, 2019. Fatality Facts 2018 Urban/rural comparison.
- Ioffe, S., Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.
- Islam, Z., Abdel-Aty, M., 2021a. Real-Time Vehicle Trajectory Estimation Based on Lane Change Detection using Smartphone Sensors. *Transportation Research Record*, 0361198121990681.
- Islam, Z., Abdel-Aty, M., 2021b. Sensor-Based Transportation Mode Recognition Using Variational Autoencoder. *Journal of Big Data Analytics in Transportation* 3(1), 15-26.
- Islam, Z., Abdel-Aty, M., Cai, Q., Yuan, J., 2021. Crash data augmentation using variational autoencoder. *Accid Anal Prev* 151, 105950.
- Johnson, D.A., Trivedi, M.M., 2011. Driving Style Recognition Using a Smartphone as a Sensor Platform. *Ieee Int C Intell Tr*, 1609-1615.
- Kanarachos, S., Christopoulos, S.-R.G., Chroneos, A., 2018. Smartphones as an integrated platform for monitoring driver behaviour: The role of sensor fusion and connectivity. *Transportation Research Part C: Emerging Technologies* 95, 867-882.
- Karim, F., Majumdar, S., Darabi, H., Chen, S., 2018. LSTM Fully Convolutional Networks for Time Series Classification. *IEEE Access* 6, 1662-1669.
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y., 2017. Lightgbm: A highly efficient gradient boosting decision tree, *Advances in Neural Information Processing Systems*, pp. 3146-3154.

- Khan, M.N., Ahmed, M.M., 2020. Trajectory-level fog detection based on in-vehicle video camera with TensorFlow deep learning utilizing SHRP2 naturalistic driving data. *Accid Anal Prev* 142, 105521.
- Kong, X., Song, X., Xia, F., Guo, H., Wang, J., Tolba, A., 2017. LoTAD: long-term traffic anomaly detection based on crowdsourced bus trajectory data. *World Wide Web* 21(3), 825-847.
- Kong, X., Xu, Z., Shen, G., Wang, J., Yang, Q., Zhang, B., 2016. Urban traffic congestion estimation and prediction based on floating car trajectory data. *Future Generation Computer Systems* 61, 97-107.
- Kuang, W., An, S., Jiang, H., 2015. Detecting Traffic Anomalies in Urban Areas Using Taxi GPS Data. *Mathematical Problems in Engineering* 2015, 1-13.
- Lee, C., Saccomanno, F., Hellinga, B., 2002. Analysis of Crash Precursors on Instrumented Freeways. *Transportation Research Record* 1784(1), 1-8.
- Lee, D., 2013. CARBayes: AnRPackage for Bayesian Spatial Modeling with Conditional Autoregressive Priors. *Journal of Statistical Software* 55(13), 24 %J Journal of Statistical Software.
- Lee, J., Abdel-Aty, M., Jiang, X., 2015. Multivariate crash modeling for motor vehicle and non-motorized modes at the macroscopic level. *Accid Anal Prev* 78, 146-154.
- Lee, J., Abdel-Aty, M., Shah, I., 2019. Evaluation of surrogate measures for pedestrian trips at intersections and crash modeling. *Accid Anal Prev* 130, 91-98.
- Li, N., Busso, C., 2016. Detecting Drivers' Mirror-Checking Actions and Its Application to Maneuver and Secondary Task Recognition. *IEEE Transactions on Intelligent Transportation Systems* 17(4), 980-992.
- Li, P., 2020. A Deep Learning Approach for Real-time Crash Risk Prediction at Urban Arterials. University of Central Florida.
- Li, P., Abdel-Aty, M., Cai, Q., Islam, Z., 2020a. A Deep Learning Approach to Detect Real-Time Vehicle Maneuvers Based on Smartphone Sensors. *IEEE Transactions on Intelligent Transportation Systems*, 1-10.
- Li, P., Abdel-Aty, M., Cai, Q., Yuan, C., 2020b. The application of novel connected vehicles emulated data on real-time crash potential prediction for arterials. *Accident Analysis & Prevention* 144, 105658.

- Li, P., Abdel-Aty, M., Islam, Z., 2021a. Driving Maneuvers Detection using Semi-Supervised Long Short-Term Memory and Smartphone Sensors. *Transportation Research Record*, 03611981211007483.
- Li, P., Abdel-Aty, M., Yuan, J., 2020c. Real-time crash risk prediction on arterials based on LSTM-CNN. *Accid Anal Prev* 135, 105371.
- Li, P., Abdel-Aty, M., Yuan, J., 2021b. Using bus critical driving events as surrogate safety measures for pedestrian and bicycle crashes based on GPS trajectory data. *Accident Analysis & Prevention* 150, 105924.
- Lin, L., Wang, Q., Sadek, A.W., 2015. A novel variable selection method based on frequent pattern tree for real-time traffic accident risk prediction. *Transportation Research Part C: Emerging Technologies* 55, 444-459.
- Liu, T., Yang, Y., Huang, G., Yeo, Y.K., Lin, Z., 2016a. Driver Distraction Detection Using Semi-Supervised Machine Learning. *IEEE Transactions on Intelligent Transportation Systems* 17(4), 1108-1120.
- Liu, X., Lu, F., Zhang, H., Qiu, P., 2013. Intersection delay estimation from floating car data via principal curves: a case study on Beijing's road network. *Frontiers of Earth Science* 7(2), 206-216.
- Liu, Z., Liu, Z., Meng, Z., Yang, X., Pu, L., Zhang, L., 2016b. Implementation and performance measurement of a V2X communication system for vehicle and pedestrian safety. 12(9), 1550147716671267.
- Liu, Z., Pu, L., Meng, Z., Yang, X., Zhu, K., Zhang, L., 2015. POFS: A novel pedestrian-oriented forewarning system for vulnerable pedestrian safety, *2015 International Conference on Connected Vehicles and Expo (ICCVE)*, pp. 100-105.
- Liu, Z., Wu, M., Zhu, K., Zhang, L., 2016c. SenSafe: A Smartphone-Based Traffic Safety Framework by Sensing Vehicle and Pedestrian Behaviors. *Mobile Information Systems* 2016, 13.
- Lord, D., Mannering, F., 2010. The statistical analysis of crash-frequency data: A review and assessment of methodological alternatives. *Transportation Research Part A: Policy and Practice* 44(5), 291-305.
- Luong, M.-T., Pham, H., Manning, C.D., 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*.
- Ma, X., Dai, Z., He, Z., Ma, J., Wang, Y., Wang, Y., 2017. Learning Traffic as Images: A Deep Convolutional Neural Network for Large-Scale Transportation Network Speed Prediction. *Sensors* 17(4).

- Maaten, L.v.d., Hinton, G., 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9(Nov), 2579-2605.
- Mahajan, V., Katrakazas, C., Antoniou, C., 2020. Prediction of Lane-Changing Maneuvers with Automatic Labeling and Deep Learning. *Transportation Research Record* 2674(7), 336-347.
- Mahmoud, N., Abdel-Aty, M., Cai, Q., Yuan, J., 2021a. Estimating cycle-level real-time traffic movements at signalized intersections. *Journal of Intelligent Transportation Systems*, 1-24.
- Mahmoud, N., Abdel-Aty, M., Cai, Q., Yuan, J., 2021b. Predicting cycle-level traffic movements at signalized intersections using machine learning models. *Transportation Research Part C: Emerging Technologies* 124, 102930.
- MathWorks, 2019. Matlab Documentation.
- Meseguer, J.E., Calafate, C.T., Cano, J.C., Manzoni, P., 2013. DrivingStyles: A smartphone application to assess driver behavior, *2013 IEEE Symposium on Computers and Communications (ISCC)*, pp. 000535-000540.
- Mousa, S.R., Bakhit, P.R., Osman, O.A., Ishak, S., 2018. A Comparative Analysis of Tree-Based Ensemble Methods for Detecting Imminent Lane Change Maneuvers in Connected Vehicle Environments. *Transportation Research Record* 2672(42), 268-279.
- Muchuruza, V., Mussa, R., 2005. Traffic operation and safety analyses of minimum speed limits on florida rural interstate highways, *Proceedings of the 2005 Mid-Continent Transportation Research Symposium*. Citeseer, pp. 1-10.
- Mumcuoglu, M.E., Alcan, G., Unel, M., Cicek, O., Mutluergil, M., Yilmaz, M., Koprubasi, K., 2019. Driving Behavior Classification Using Long Short Term Memory Networks, *2019 AEIT International Conference of Electrical and Electronic Technologies for Automotive (AEIT AUTOMOTIVE)*, pp. 1-6.
- National Highway Traffic Safety Administration, 2019. Fatality analysis reporting system (FARS) encyclopedia.
- Nemat-Nasser, S.C., Smith, A.T., 2014. Driver identification based on face data. Google Patents.
- NHTSA, 2020a. Traffic Safety Facts, 2018 Data, Bicyclists and Other Cyclists. National Highway Traffic Safety Administration.
- NHTSA, 2020b. Traffic Safety Facts, 2018 Data, Pedestrians. National Highway Traffic Safety Administration.
- NHTSA, 2020c. Traffic Safety Facts Annual Report.

- Ochieng, W.Y., Sauer, K., 2002. Urban road transport navigation: performance of the global positioning system after selective availability. *Transportation Research Part C: Emerging Technologies* 10(3), 171-187.
- Oh, J.-S., Oh, C., Ritchie Stephen, G., Chang, M., 2005. Real-Time Estimation of Accident Likelihood for Safety Enhancement. *Journal of Transportation Engineering* 131(5), 358-363.
- Ordóñez, J.F., Roggen, D., 2016. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. *Sensors* 16(1).
- Ouyang, Z., Niu, J., Liu, Y., Liu, X., 2019. An Ensemble Learning-Based Vehicle Steering Detector Using Smartphones. *IEEE Transactions on Intelligent Transportation Systems*, 1-12.
- Paefgen, J., Kehr, F., Zhai, Y., Michahelles, F., 2012. Driving behavior analysis with smartphones: insights from a controlled field study, *Proceedings of the 11th International Conference on Mobile and Ubiquitous Multimedia*. ACM, Ulm, Germany, pp. 1-8.
- Pang, L.X., Chawla, S., Liu, W., Zheng, Y., 2013. On detection of emerging anomalous traffic patterns using GPS data. *Data & Knowledge Engineering* 87, 357-373.
- Park, Y.-J., Saccomanno, F.F., 2006. Evaluating speed consistency between successive elements of a two-lane rural highway. *Transportation Research Part A: Policy and Practice* 40(5), 375-385.
- Pebesma, E., 2018. Simple features for R: Standardized support for spatial vector data, p. 439.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12(Oct), 2825-2830.
- QGIS.org, 2016. QGIS geographic information system.
- R Core Team, 2013. R: A language and environment for statistical computing. Vienna, Austria.
- Rahmani, M., Jenelius, E., Koutsopoulos, H.N., 2015. Non-parametric estimation of route travel time distributions from low-frequency floating car data. *Transportation Research Part C: Emerging Technologies* 58, 343-362.
- Roshandel, S., Zheng, Z., Washington, S., 2015. Impact of real-time traffic characteristics on freeway crash occurrence: systematic review and meta-analysis. *Accident Analysis & Prevention* 79, 198-211.
- Saiprasert, C., Pholprasit, T., Pattara-Atikom, W., 2013. Detecting driving events using smartphone, *Proceedings of the 20th ITS World Congress*.

- Saleh, K., Hossny, M., Nahavandi, S., 2017. Driving behavior classification based on sensor data fusion using LSTM recurrent neural networks, *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pp. 1-6.
- Sanjeevi, M., 2018. Chapter 10.1: DeepNLP — LSTM (Long Short Term Memory) Networks with Math.
- Satria, R., Agüero-Valverde, J., Castro, M., 2020. Spatial analysis of road crash frequency using Bayesian models with Integrated Nested Laplace Approximation (INLA). *Journal of Transportation Safety & Security*, 1-23.
- Schepers, J.P., Kroeze, P.A., Sweers, W., Wust, J.C., 2011. Road factors and bicycle-motor vehicle crashes at unsignalized priority intersections. *Accid Anal Prev* 43(3), 853-861.
- Sewalkar, P., Seitz, J.J.S., 2019. Vehicle-to-pedestrian communication for vulnerable road users: Survey, design considerations, and challenges. 19(2), 358.
- Shi, Q., Abdel-Aty, M., 2015. Big Data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. *Transportation Research Part C: Emerging Technologies* 58, 380-394.
- Shi, Q., Abdel-Aty, M., Yu, R., 2016. Multi-level Bayesian safety analysis with unprocessed Automatic Vehicle Identification data for an urban expressway. *Accident Analysis & Prevention* 88, 68-76.
- Singh, G., Bansal, D., Sofat, S., 2017. A smartphone based technique to monitor driving behavior using DTW and crowdsensing. 40(C), 56-70.
- Spiegelhalter, D.J., Best, N.G., Carlin, B.P., Van Der Linde, A., 2002. Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 64(4), 583-639.
- Stipancic, J., Miranda-Moreno, L., Saunier, N., 2018a. Vehicle manoeuvres as surrogate safety measures: Extracting data from the gps-enabled smartphones of regular drivers. *Accident Analysis & Prevention* 115, 160-169.
- Stipancic, J., Miranda-Moreno, L., Saunier, N., Labbe, A., 2018b. Surrogate safety and network screening: Modelling crash frequency using GPS travel data and latent Gaussian Spatial Models. *Accident Analysis & Prevention* 120, 174-187.
- Strauss, J., Zangenehpour, S., Miranda-Moreno, L.F., Saunier, N., 2017. Cyclist deceleration rate as surrogate safety measure in Montreal using smartphone GPS data. *Accident Analysis & Prevention* 99(Pt A), 287-296.

- Sun, J., Sun, J., Chen, P., 2014. Use of Support Vector Machine Models for Real-Time Prediction of Crash Risk on Urban Expressways. *Transportation Research Record* 2432(1), 91-98.
- Tahmasbi-Sarvestani, A., Mahjoub, H.N., Fallah, Y.P., Moradi-Pari, E., Abuchaar, O., 2017. Implementation and Evaluation of a Cooperative Vehicle-to-Pedestrian Safety Application. *Ieee Intelligent Transportation Systems Magazine* 9(4), 62-75.
- Theofilatos, A., 2017. Incorporating real-time traffic and weather data to explore road accident likelihood and severity in urban arterials. *Journal of safety research* 61, 9-21.
- Tian, Y., Pan, L., 2015. Predicting Short-Term Traffic Flow by Long Short-Term Memory Recurrent Neural Network, *2015 IEEE International Conference on Smart City/SocialCom/SustainCom (SmartCity)*, pp. 153-158.
- Tieleman, T., Hinton, G., 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2), 26-31.
- Truong, L., Somenahalli, S., 2011. Using GIS to Identify Pedestrian-Vehicle Crash Hot Spots and Unsafe Bus Stops. *Journal of Public Transportation* 14(1), 99-114.
- Truong, L.T., Currie, G., 2019. Macroscopic road safety impacts of public transport: A case study of Melbourne, Australia. *Accident Analysis and Prevention* 132(August), 105270-105270.
- Tselentis, D.I., Yannis, G., Vlahogianni, E.I., 2017. Innovative motor insurance schemes: A review of current practices and emerging challenges. *Accident Analysis & Prevention* 98, 139-148.
- U.S. Census Bureau, 2011. Census Traffic Analysis Zone Program MAP/TIGER Partnership Software Participant Guidelines, *U.S. Department of Commerce Economics and Statistics Administration*.
- Ukkusuri, S., Hasan, S., Aziz, H.M.A., 2011. Random Parameter Model Used to Explain Effects of Built-Environment Characteristics on Pedestrian Crash Frequency. *Transportation Research Record: Journal of the Transportation Research Board* 2237(1), 98-106.
- Uno, N., Kurauchi, F., Tamura, H., Iida, Y., 2009. Using Bus Probe Data for Analysis of Travel Time Variability. *Journal of Intelligent Transportation Systems* 13(1), 2-15.
- Vedagiri, P., Killi, D.V., 2015. Traffic Safety Evaluation of Uncontrolled Intersections using Surrogate Safety Measures under Mixed Traffic Conditions. *Transportation Research Record* 2512(1), 81-89.
- Veness, C., 2019. Calculate distance, bearing and more between Latitude/Longitude points.

- Vlahogianni, E.I., Barmponakis, E.N., 2017. Driving analytics using smartphones: Algorithms, comparisons and challenges. *Transportation Research Part C: Emerging Technologies* 79, 196-206.
- Wang, C., Xu, C., Dai, Y., 2019a. A crash prediction method based on bivariate extreme value theory and video-based vehicle trajectory data. *Accident Analysis & Prevention* 123, 365-373.
- Wang, J., Luo, T., Fu, T., 2019b. Crash prediction based on traffic platoon characteristics using floating car trajectory data and the machine learning approach. *Accid Anal Prev* 133, 105320.
- Wang, L., Abdel-Aty, M., Lee, J., 2017a. Safety analytics for integrating crash frequency and real-time risk modeling for expressways. *Accident Analysis & Prevention* 104, 58-64.
- Wang, L., Abdel-Aty, M., Ma, W., Hu, J., Zhong, H., 2019c. Quasi-vehicle-trajectory-based real-time safety analysis for expressways. *Transportation Research Part C: Emerging Technologies* 103, 30-38.
- Wang, L., Abdel-Aty, M., Shi, Q., Park, J., 2015a. Real-time crash prediction for expressway weaving segments. *Transportation Research Part C: Emerging Technologies* 61, 1-10.
- Wang, W., Xi, J., Chong, A., Li, L., 2017b. Driving Style Classification Using a Semisupervised Support Vector Machine. *IEEE Transactions on Human-Machine Systems* 47(5), 650-660.
- Wang, X., Fan, T., Chen, M., Deng, B., Wu, B., Tremont, P., 2015b. Safety modeling of urban arterials in Shanghai, China. *Accident Analysis & Prevention* 83, 57-66.
- Wang, Y., Chen, Y.J., Yang, J., Gruteser, M., Martin, R.P., Liu, H., Liu, L., Karatas, C., 2016. Determining Driver Phone Use by Exploiting Smartphone Integrated Sensors. *IEEE Transactions on Mobile Computing* 15(8), 1965-1981.
- Washington, S., Karlaftis, M.G., Mannering, F., Anastasopoulos, P., 2020. *Statistical and econometric methods for transportation data analysis*. CRC press.
- Washington, S., Oh, J., 2006. Bayesian methodology incorporating expert judgment for ranking countermeasure effectiveness under uncertainty: example applied to at grade railroad crossings in Korea. *Accid Anal Prev* 38(2), 234-247.
- Watanabe, S., Opper, M., 2010. Asymptotic equivalence of Bayes cross validation and widely applicable information criterion in singular learning theory. *Journal of machine learning research* 11(12).
- Wikipedia, 2019. Moving average, Wikipedia.

- Wu, X., Miucic, R., Yang, S., Al-Stouhi, S., Misener, J., Bai, S., Chan, W., 2014. Cars Talk to Phones: A DSRC Based Vehicle-Pedestrian Safety System, *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, pp. 1-7.
- Xie, K., Ozbay, K., Kurkcu, A., Yang, H., 2017. Analysis of Traffic Crashes Involving Pedestrians Using Big Data: Investigation of Contributing Factors and Identification of Hotspots. *Risk Analysis* 37(8), 1459-1476.
- Xie, K., Wang, X., Huang, H., Chen, X., 2013. Corridor-level signalized intersection safety analysis in Shanghai, China using Bayesian hierarchical models. *Accident Analysis & Prevention* 50, 25-33.
- Xie, K., Wang, X., Ozbay, K., Yang, H., 2014. Crash frequency modeling for signalized intersections in a high-density urban road network. *Analytic Methods in Accident Research* 2, 39-51.
- Xu, C., Liu, P., Wang, W., Li, Z., 2012. Evaluation of the impacts of traffic states on crash risks on freeways. *Accident Analysis & Prevention* 47, 162-171.
- Xu, C., Liu, P., Wang, W., Li, Z., 2014. Identification of freeway crash-prone traffic conditions for traffic flow at different levels of service. *Transportation research part A: policy and practice* 69, 58-70.
- Xu, C., Tao, D., Xu, C., 2013a. A survey on multi-view learning. *arXiv preprint*.
- Xu, C., Tarko, A.P., Wang, W., Liu, P., 2013b. Predicting crash likelihood and severity on freeways with real-time loop detector data. *Accident Analysis & Prevention* 57, 30-39.
- Xuesong Wang, J.Y., Chris Lee, Zhuoran Ji, Shikai You., 2016. Macro-level safety analysis of pedestrian crashes in Shanghai, China. *Accident Analysis & Prevention* 96, 12-21.
- Yu, J., Chen, Z., Zhu, Y., Jennifer Chen, Y., Kong, L., Li, M., 2017a. Fine-Grained Abnormal Driving Behaviors Detection and Identification with Smartphones. *IEEE Transactions on Mobile Computing* 16(8), 2198-2212.
- Yu, R., Abdel-Aty, M., 2013. Utilizing support vector machine in real-time crash risk evaluation. *Accid Anal Prev* 51, 252-259.
- Yu, R., Abdel-Aty, M., 2014a. Analyzing crash injury severity for a mountainous freeway incorporating real-time traffic and weather data. *Safety Science* 63, 50-56.
- Yu, R., Abdel-Aty, M., 2014b. Using hierarchical Bayesian binary probit models to analyze crash injury severity on high speed facilities with real-time traffic data. *Accident Analysis & Prevention* 62, 161-167.

- Yu, R., Abdel-Aty, M.A., Ahmed, M.M., Wang, X., 2014. Utilizing microscopic traffic and weather data to analyze real-time crash patterns in the context of active traffic management. *IEEE Transactions On Intelligent Transportation Systems* 15(1), 205-213.
- Yu, R., Wang, X., Abdel-Aty, M., 2017b. A Hybrid Latent Class Analysis Modeling Approach to Analyze Urban Expressway Crash Risk. *Accident Analysis & Prevention* 101, 37-43.
- Yu, R., Wang, X., Yang, K., Abdel-Aty, M., 2016. Crash risk analysis for Shanghai urban expressways: A Bayesian semi-parametric modeling approach. *Accident Analysis & Prevention* 95(Pt B), 495-502.
- Yuan, J., Abdel-Aty, M., 2018. Approach-level real-time crash risk analysis for signalized intersections. *Accident Analysis & Prevention* 119, 274-289.
- Yuan, J., Abdel-Aty, M., Gong, Y., Cai, Q., 2019. Real-Time Crash Risk Prediction using Long Short-Term Memory Recurrent Neural Network. *Transportation Research Record* 2673(4), 314-326.
- Yuan, J., Abdel-Aty, M., Wang, L., Lee, J., Yu, R., Wang, X., 2018. Utilizing bluetooth and adaptive signal control data for real-time safety analysis on urban arterials. *Transportation Research Part C: Emerging Technologies* 97, 114-127.
- Zaki, M.H., Sayed, T., Shaaban, K., 2014. Use of Drivers' Jerk Profiles in Computer Vision-Based Traffic Safety Evaluations. *Transportation Research Record* 2434(1), 103-112.
- Zangenehpour, S., Strauss, J., Miranda-Moreno, L.F., Saunier, N., 2016. Are signalized intersections with cycle tracks safer? A case-control study based on automated surrogate safety analysis using video data. *Accident Analysis & Prevention* 86, 161-172.
- Zhang, S., 2020. Prediction of Pedestrians' Red Light Violations Using Deep Learning. University of Central Florida.
- Zhang, S., Abdel-Aty, M., Cai, Q., Li, P., Ugan, J., 2020a. Prediction of pedestrian-vehicle conflicts at signalized intersections based on long short-term memory neural network. *Accident Analysis & Prevention* 148, 105799.
- Zhang, S., Abdel-Aty, M., Wu, Y., Zheng, O., 2020b. Modeling pedestrians' near-accident events at signalized intersections using gated recurrent unit (GRU). *Accident Analysis & Prevention* 148, 105844.
- Zhang, S., Abdel-Aty, M., Wu, Y., Zheng, O., 2021. Pedestrian Crossing Intention Prediction at Red-Light Using Pose Estimation. *IEEE Transactions on Intelligent Transportation Systems*, 1-9.

- Zhang, S., Abdel-Aty, M., Yuan, J., Li, P., 2020c. Prediction of Pedestrian Crossing Intentions at Intersections Based on Long Short-Term Memory Recurrent Neural Network. *Transportation Research Record* 2674(4), 57-65.
- Zhang, X., Zhang, X., Verma, S., Liu, Y., Blumenstein, M., Li, J., 2019. Detection of Anomalous Traffic Patterns and Insight Analysis from Bus Trajectory Data, *PRICAI 2019: Trends in Artificial Intelligence*, pp. 307-321.
- Zhao, Z., Chen, W., Wu, X., Chen, P.C.Y., Liu, J., 2017. LSTM network: a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems* 11(2), 68-75.
- Zheng, Z., Ahn, S., Monsere, C.M., 2010. Impact of traffic oscillations on freeway crash occurrences. *Accident Analysis & Prevention* 42(2), 626-636.
- Zheng, Z., Yang, Y., Liu, J., Dai, H., Zhang, Y., 2019. Deep and Embedded Learning Approach for Traffic Flow Prediction in Urban Informatics. *IEEE Transactions on Intelligent Transportation Systems*, 1-13.
- Zhou, H., Bromfield, S., 2007. Moving the Bus Back Into Traffic Safely – Signage and Lighting Configuration Phase I.
- Zou, Y., Wu, L., Lord, D., 2015. Modeling over-dispersed crash data with a long tail: Examining the accuracy of the dispersion parameter in Negative Binomial models. *Analytic Methods in Accident Research* 5-6, 1-16.