ABSTRACT

Title of Dissertation:	DISTRIBUTED COMPUTING AND UNSUPERVISED DEEP LEARNING FOR ANALYZING HUMAN TRAVEL BEHAVIORS USING BIG TRAJECTORY DATA
	Peiqi Zhang, Doctor of Philosophy, 2025
Dissertation directed by:	Professor Kathleen Stewart Department of Geographical Sciences

Human travel behaviors, which refer to the movement of either individuals or groups across space and time, has been a research focus for the understanding of not only the behaviors themselves, but also the interaction between humans and their environment. In recent years, with the development of information and communication technologies, passively collected trajectory data (i.e., sequences of sampled geographic coordinates with timestamps collected by GPS sensors in devices such as mobile phones and vehicles) has become more available. This new data type has provided unprecedented opportunities for research in human travel behaviors offering advantages of broad spatial scale, large data volume, and fine granularity compared with traditional actively collected data, such as census data and survey data. This dissertation focused on exploring how advanced research methods, including distributed computing and unsupervised deep learning, can be extended and applied to examine human travel behaviors using passively

collected trajectory data. In this dissertation, three studies are described that examine three different topic that contribute to our understanding of vehicular travel behaviors. The first study involved developing a distributed research framework based on Apache Spark and Sedona to estimate traffic speed and speeding across a state-wide road network and multiple road types using passively collected mobile device data. The study analyzes spatio-temporal patterns in traffic speed and speeding behaviors in the state of California and examines differences in patterns from different road types such as freeways and residential roads. The second study developed a research framework that combines a rule-based distributed parking extracting module and an LSTM-autoencoder model to extract parking trajectories and automatically classify them into different categories (e.g., direct parking and cruising for parking) using in-vehicle trajectory data. Using datasets collected in the Washington, DC metropolitan area, this study further investigates the driving patterns and spatial distribution of cruising trips, identifying areas where parking demand may exceed supply. The third study in the dissertation proposed a research framework that is comprised of a distributed contour plot-building module (designed to capture speed changes compared with free-flow and historical averages), a W-net-based semantic segmentation model to segment the non-recurrent congestion (NRC) impact areas in these plots, and a postprocessing module to ensure that the identified propagation patterns follow the law of shockwaves using passively collected in-vehicle trajectory data. Based on detected NRC impact areas, propagation of NRC over actual road networks in the Washington, DC metropolitan area was examined to identify how these types of congestion move through these road networks. The contributions of this dissertation include providing cutting-edge distributed computing and deep learning

methods to make full use of passively collected trajectory data for human travel behavior analyses and provide feasible means to overcome the challenges brought by the large data volumes, broad spatio-temporal ranges, and lack of ground truth labels. The integration of passively collected trajectory data and new research methods have provided new insights into human travel behaviors over large spatial scales and fine granularities, and lay the foundation for future research on human mobility.

DISTRIBUTED COMPUTING AND UNSUPERVISED DEEP LEARNING FOR ANALYZING HUMAN TRAVEL BEHAVIORS USING BIG TRAJECTORY DATA

By

Peiqi Zhang

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park, in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2025

Advisory Committee:
Professor Kathleen Stewart, Chair
Professor Giovanni Baiocchi
Professor Taylor Oshan
Professor Xianfeng Yang
Professor Yiqun Xie

Dedication

To my parents,

Yuhong Wang and Jingjun Zhang,

My grandparents,

Sulian Li, Guofu Wang, Jiadong Zhang, and Guomin Liu,

And my beloved families

For your endless love, support, and treasured memories along the journey.

Acknowledgements

The Ph.D. is a long journey, lonely and full of pressure for the most part, but the happiness of a few moments is like fireworks, making all the efforts worthwhile.

I would like to express my gratitude to my advisor Prof. Kathleen Stewart, for the opportunities, trust, and guidance she gave throughout my Ph.D. journey. Prof. Stewart gave me almost unlimited freedom to explore the research questions and methods I liked. My Ph.D. journey would not have been possible without Prof. Stewart's consistent support.

Then I would like to express my sincere appreciation to my committee members for their generous suggestions and insights on my research and dissertation. Especially, I want to thank Prof. Giovanni Baiocchi, for always reminding me of the importance of knowing the limitations and biases of my research. To Prof. Taylor Oshan, for critical crucial insights into improving my research questions and methods. To Prof. Xianfeng Yang, for valuable insight from the perspective of transportation engineering. To Prof. Yiqun Xie, for greatly expanding my knowledge and understanding of cutting-edge research methods, such as Apache Sedona and deep learning models.

Thanks to Dr. Yao Li and Dr. Guimin Zhu for their guidance and selfless help since I knew nothing about research, teaching, and Ph.D. life in the US. Thanks to my girlfriend Shuran Zhao for bringing me joy and love when I was most helpless. Thanks to my dear roommates Yuhao Wang and Zhihao Wang for looking after me and for the joyful moments during the past five years. Thanks to all my friends Dr. Zhiyue Xia, Dr. Aolin Jia, Dr. Zheng Liu, Dr. Ruohan Li, Dr. Yiming Zhang, Xinyuan Li, Dr. Yunting

Song, Dr. Jiaming Lu, Dr. Jeffery Sauer, Jiena He, Quan Shen, Mengyu Liao, Qianru Liao, Yingrui Zhao, Yuehui Qian, Weiye Chen, Zhili Li, and Xin Dong.

A special thanks go to my badminton friends, Dr. Weidi Yin, Dr. Jingchuan Wang, Shiyuan Xiang, Shenghao Li, Tianyi (Eddie) Li, Siyuan (Sydnie) Shen, Zhaojie Yin, Songlun Dai, Weifeng Chen, Xuan Su, Ziyang Shen, Dr. Wei Xiao, and Xintong Zhao as they supported me through this lonely PhD journey.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

API: Application Programming Interface

CA: California

DBMSs: Databases Management Systems

DBSCAN: Density-Based Spatial Clustering of Applications with Noise

DTW: Dynamic Time Warping

FCM: fuzzy C-Means

GPS: The Global Positioning System

ITS: Intelligent Transportation Systems

KDB-Tree: K-Dimensional B-tree

KL divergence: Kullback-Leibler divergence

LCSS: Longest Common SubSequence

LSTM: Long Short-Term Memory

MTI: Maryland Transportation Institute

NRC: Non-Recurrent Congestions

OGC: Open Geospatial Consortium

OD: origin and destination

OSM: OpenStreetMap

PeMS: Performance Measurement System

RC: Recurrent Congestions

RDDs: Resilient Distributed Datasets

ReLU: Rectified Linear Unit

RNN: Recurrent Neural Network

SQL: Structured Query Language

SRI: Speed Reduction Index

SRIff: Speed Reduction Index compares with Free Flow speed

SRIhs: Speed Reduction Index compares with Historic average speed

SRDDs: Spatial Resilient Distributed Datasets

Washington, DC: Washington, District of Columbia

Chapter 1: Introduction

1.1 Background and motivation

Human travel behaviors refer to the movement of either individuals or groups over space and time in the context of pursuing different tasks, e.g., trips to work, trips for recreation purposes or other types (C. Chen et al., 2016). Quantitative analysis of human travel behavior, e.g. number of trips and travel speed at road segments level, is of fundamental importance to understand the processes related to human movement and the impact on society and the environment of these movements (Barbosa et al., 2018; Cabanas-Tirapu et al., 2025). Research on human travel behaviors addresses a range of questions—from evaluating how socio-economic, built environment, and travel-related factors influence movement, to quantifying mobility patterns and developing models to predict future behaviors (C. Chen et al., 2016; Hong et al., 2014; Klinger & Lanzendorf, 2016). In the face of recent global challenges such as the COVID-19 global pandemic and natural disasters, studies on human travel behaviors have played a crucial role in facilitating our understand on impact of challenges on different social groups (T. Hu et al., 2021; J. Wang & Hu, 2024; Yabe et al., 2021). For instance, analyzing shifts in travel patterns (e.g., daily trip frequency and travel distance) before, during, and after these events has not only illuminated their impact on daily life but also provided valuable insights into issues of social equity and equality (X. Huang et al., 2022).

With the rapid rise and prevalence of information and communication technology, passively collected big trajectory data such as smart card data, GPS trajectory data (i.e., a GPS trajectory refers to a sequence of sampled geographic coordinates with timestamps

collected by GPS sensors and other related movement information (X. Yang et al., 2018)) collected by in-vehicle sensors, and GPS location data recorded by mobile devices through apps has become more available (C. Chen et al., 2016). Compared with traditional actively collected data types (e.g., survey data and census data), passively collected big trajectory data has the advantages of large data volume, broad temporal and spatial distribution and coverage, and high collection frequency (Chen, et al., 2014; Fan et al., 2019; Yu et al., 2020). With these advantages, research on travel behaviors at fine-granularity and large spatio-temporal scales has been the subject of numerous studies. For example, during the COVID-19 global pandemic, using individual-level passively collected mobile device data, changes in travel behaviors, such as travel frequency and traffic flow were estimated and updated for the whole United States(Xiong et al., 2020). The heterogeneity of travel behavior changes over space and by social groups was further investigated and used to explore systemic social inequity issues, e.g., heterogeneity of people's resilience to the stay-in-home order (X. Huang et al., 2022; Xiong et al., 2020).

Despite the great opportunities that big trajectory data has offered, this new data have also brought new challenges for spatio-temporal analysis, modeling, and prediction (de Almeida et al., 2020). Similar to other big data, trajectory big data also has the properties of 'veracity' and 'value' at the same time, which makes traditional analysis methods for small data inappropriate for data-intensive tasks, such as integration, visualization, querying and analysis for large-scale real-time systems (Anuradha, 2015). For example, although based on Open Geospatial Consortium (OGC) standards (e.g., OGC Web Map Service and Geographic SPARQL Protocol and RDF Query Language), some spatial databases systems (DBMSs), i.e., extended relational DBMSs with spatial

data types that provide support for spatial analysis, suffer from a scalability issue because the massive scale of available geospatial data can't be handled by traditional spatial query processing methods (J. Yu et al., 2019a). Furthermore, for many cases, even if traditional methods could work with big trajectory data, the processing time may be unacceptably long due to the data volume (P. Zhang et al., 2023). Therefore, how to extend current cutting-edge research methods, including the use of distributed computing and deep learning, and propose new research frameworks suitable for big trajectory data are open research questions that are discussed in this dissertation.

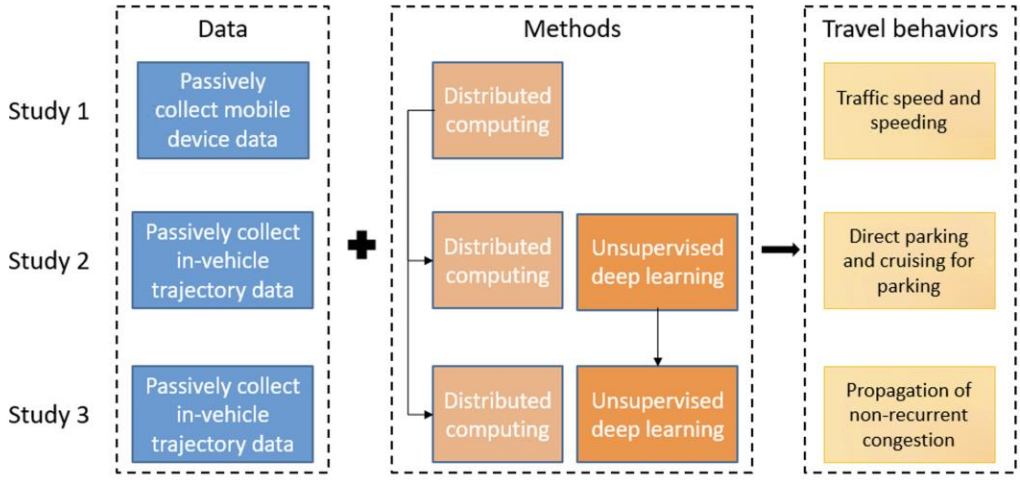


Figure 1. 1 Conceptual flowchart for dissertation topics

In this dissertation, I explore vehicular travel behaviors using trajectory data and innovative research methods, including distributed computing and unsupervised deep learning (Fig. 1.1). The work consists of three distinct studies, each examining a specific aspect that improves our understanding of travel behaviors: traffic speed and speeding, urban parking and cruising for parking, and how non-recurrent congestion (NRC) is propagated across actual road networks. I investigate how distributed computing frameworks can be applied and extended to handle the large volume and broad spatio-temporal coverage of trajectory data. Additionally, I demonstrate how unsupervised deep

learning techniques, such as the autoencoder-decoder structure, can be applied to automatically classify different behaviors without relying on predefined labels in the data.

The first study (Chapter 2) in this dissertation examines traffic speed and speeding behaviors over a state-wide road network that encompasses multiple road types. Traffic speed is an important component of urban mobility, and an accurate estimation of traffic speed and determining where speeding is occurring is critical for the development of intelligent transportation systems (ITS) (Lin et al., 2017). Previous research has primarily relied on actively collected data for travel speed information, such as data collected from traffic monitoring stations and GPS records in travel surveys, which often suffer from limited spatial coverage and volume and tend to focus on freeways (Bakowski et al., 2018; Kwon et al., 2003). As a result, these studies may not fully capture traffic conditions across larger spatio-temporal scales (C. Chen et al., 2016).

For my research in Study 1, making use of the advantages brought by passively collected mobile device data, I proposed a research framework based on a distributed computing environment implemented using Apache Spark and Sedona and that is capable of estimating traffic speed for state-wide road networks. Based on the in-memory Resilient Distributed Datasets (RDDs) supported by Spark and Spatial RDDs (SRDDs) supported by Sedona that extends the regular Spark RDDs, the framework is suitable for determining traffic speed and estimating speeding across a broad spatial scale (e.g., at the scale of large urban agglomerations or at state level) and large data volume (i.e., billions of waypoints) (Zaharia et al., 2016) (J. Yu et al., 2019a). The framework consists of four modules: data preprocessing, map matching, waypoint gap filling, and speed estimation for road segments. The Spark-based preprocessing module filters anomalies at both the

waypoint and trip levels. In the Sedona-based map matching module, hundreds of millions of waypoints are aligned to a state-wide road network by considering both the distances between waypoints and corresponding road segments, as well as the angle between the travel direction and road direction. In the map matching module, both KDB-Tree spatial partitioning (Robinson, 1981) and R-Tree index are applied to provide further acceleration to computing. Given the non-uniform sampling intervals in the trajectory data, gaps inevitably arise in the trip trajectories generated by the map matching module. To address these gaps, a gap filling module that uses Spark and iGraph has been developed to fill in missing road segments along each trip trajectory (Ju et al., 2016). Finally, a speed estimation module aggregates the travel speed information matched to road segments to compute traffic speeds and detect speeding over specified temporal intervals.

Using mobile device data passively collected over a 24-hour period for the State of California (CA) on February 1st, 2020, with around 126 million waypoints, I tested our framework and examined both the speed of traffic in CA. Speed estimates generated at one-hour intervals covered about 600,000 road segments, representing roughly 17% of all CA roads and including 46% of freeway segments. Additionally, by incorporating speed limit data from OpenStreetMap (OSM), I analyzed traffic speeding behaviors and their spatial heterogeneity. This research offers valuable insights into traffic conditions across a large-scale road network and extends our understanding to road types beyond freeways, such as ramps and residential roads.

The second study (Chapter 3) in this dissertation investigated parking behaviors in an urban environment. Parking is a common driving behavior and is a necessary part of

daily driving, which makes parking an important component of good city management (Chai et al., 2019; Lu & Liao, 2020; Millard-Ball et al., 2014) and the development of the ITS (Khanna & Anand, 2016). Certain parking behaviors, specifically searching or *cruising* for parking, may cause a series of negative impacts, such as increasing air pollution, and causing additional congestion, driver frustration, and an excess of vehicle miles travelled (Bagha et al., 2023; Cao et al., 2019; Shoup, 2006; Weinberger et al., 2020). Consequently, detecting locations where cruising for parking occurs and distinguishing this behavior from scenarios involving direct parking, is an important step for mitigating these negative impacts. In previous research on parking, actively collected data types, such as on-street surveys and video recordings have been used (Lee et al., 2017; Van Ommeren et al., 2012). However, these studies mainly focused on small spatial scales (e.g., several streets or parking garages) and fail to fully examine the different behaviors associated with the process of parking in a city including the different categories of parking behaviors that may exist (i.e., direct parking and cruising for parking).

For study 2, I designed a rule-based distributed computing framework using Spark and Sedona to extract parking trajectories (i.e., the ending segments of trajectories involved in either searching for parking or actual parking) from massive passively collected trajectory data and built an LSTM autoencoder-decoder deep clustering model to automatically classify parking trajectories into parking or cursing categories without the use of labelled data (Yu et al., 2015; Zaharia et al., 2010; Zaharia et al., 2016). The parking trajectory extraction framework first generates a set of buffers around the destination of a driving trip using different radii and then applies a set of rules that use

travel speed to the waypoints inside each buffer area and in this way, parking trajectories are extracted. Based on the spatial data types and spatial partitioning supported by SRDDs, the framework is suitable for extracting parking behaviors from overall patterns of driving at city-wide or regional scales and offers high computing efficiency. The LSTM autoencoder deep clustering model takes driving features such as driving speed, angles, and parking distance along with extracted parking trajectories as input and classifies parking trajectories into categories of either cruising or direct parking. Within these two categories, further subcategories are also identified. Taking advantage of the LSTM layers and the autoencoder structure, the model uses driving features along with varying lengths of driving trajectories as input and provides classification in the absence of any prior labeling of trajectories.

Using a trajectory dataset that was collected using in-vehicle GPS sensors for one month in July 2018 with approximately 6.3 million trips and 2.8 billion waypoints, I tested the research framework in the Washington, DC metropolitan area. After classification, four different parking categories were retrieved, including two cruising and two direct parking categories. Cruising trajectories detected by the deep clustering model generally showed relatively longer temporal durations for parking (e.g., searching for more than 5 minutes) and larger spatial ranges. The cruising trips were more likely to have frequent turns during the search for parking, which represented travel behaviors associated with parking. Based on the classification results, the research findings provide valuable insights into areas where parking supply may not meet demand and offers useful information for local drivers and administrations (Gu et al., 2020).

The third study (Chapter 4) focused on non-recurrent congestion (NRC) and their propagation over actual road networks. NRC refers to traffic congestion induced by traffic incidents, such as traffic accidents and road works, and can contribute to multiple types of negative impacts to traffic flow, such as increases in travel time, uncertainties to travel plans, and increased driver frustration (An et al., 2016; Ou et al., 2020; Z. Zheng et al., 2023). Unlike recurrent congestion (RC), which is typically linked to predictable, high-traffic periods (e.g., weekday mornings and evening rush hours) at specific locations, NRC can occur at any time and place, potentially causing more severe impacts. Although previous studies have proposed various methods to model the spatio-temporal impact and propagation of NRC (including static methods, queue-based models, and shockwave approaches), most have focused on limited freeway stretches or simulated networks, thereby neglecting the propagation of NRC over actual road networks.

For my third study, I developed a research framework to investigate the spatio-temporal range of the NRC over actual road networks using passively collected trajectory data and modeled their propagation. The research framework contains a distributed contour plot building module, a W-Net-based unsupervised semantic segmentation model, and a postprocessing module that follows the law of shockwaves. The contour plot building module is designed to first estimate two speed reduction index (SRI) indicators that measure the speed change for road segments at a given time from the free-flow speed and historic average speed respectively, and then generate contour plots for the designed routes using the SRI indicators. By this means, clusters with high SRI values in the contour plots represent the spatio-temporal range of all types of congestion (both RC and NRC) and congestion that is likely to be NRC. Using Apache Spark and Sedona,

the module is designed for data with broad spatial scale and large data volume. To determine the spatio-temporal range of congested areas from the contour plots and to overcome the challenges brought by data noise, a W-Net-based unsupervised image segmentation model is used. The segmentation model takes contour plots as inputs and generates pixel-level classifications for the spatio-temporal range of all congestion including NRC in the contour plots without any labels or predefined thresholds. After segmentation, in addition to the rules that make sure segmented NRC follow the law of shockwaves, we apply a new rule that separates segmented NRC that are more likely to be caused by the fluctuations of RC for further investigation. Neglecting this step might result in an overestimation of NRC and so our approach has developed this additional check to improve our estimates of NRC and their space-time ranges.

Using trajectory data collected for 6 months from April to September 2018 in the Washington, DC metropolitan area, I examined NRC propagation for 7 different travel routes. Approximately 17,000 NRCs in total were extracted in the study area and scenarios where NRC propagated from one freeway to another were investigated. The examination results demonstrate that the research framework could be used to examine the propagation of NRC over complex actual road networks and facilitate our understanding about congestion in transportation systems. This research also laid the foundation for future research on NRC propagation including predicting congestion duration and maximum impact range over actual road networks.

1.2 Dissertation structure and research questions

Overall, this dissertation focuses on exploring new research methods, e.g., distributed computing and unsupervised deep learning, to examine human travel behaviors using passively collected mobility data.

This dissertation is composed of five chapters. Chapter 1 introduces the broader research context for these studies and outlines the dissertation structure. Chapter 2 presents a research framework for analyzing traffic speed and speeding behaviors across a state-level road network using passively collected mobile device data. Chapter 3 describes a research method for automatically classifying parking trajectories—based on characteristics such as speed, turning behavior, and proximity to road networks—to identify trajectories involved in cruising for parking and direct parking, using an LSTM-autoencoder deep clustering model. Chapter 4 introduces a research framework for modeling the spatio-temporal propagation of NRC over actual road networks, effectively distinguishing the impact areas of NRC from RC. Finally, Chapter 5 concludes the dissertation by summarizing the key findings and discussing future research directions related to the exploration of human travel behaviors using innovative research methods.

The research questions of Chapter 2 are:

1. What measurements should a distributed map matching module consider for matching noisy mobile device data to dense road networks accurately while maintaining a high computing efficiency?

2. What benefits does an Apache Sedona-based framework bring over a Spark-based framework for estimating travel speeds across a state-wide road

network using big mobile device trajectory data?

3. How well do estimations of travel speed using big passively collected mobile device data compare with estimates derived from more traditional data sources, such as loop detector data?

The research questions of Chapter 3 are:

1. Using a distributed computing framework, how can we identify and extract parking trajectories along with their driving features using passively collected trajectory data?

2. What are the benefits of an LSTM-autoencoder model in automatically classifying different types of parking trajectories according to their driving behaviors along trajectories compared with baseline models?

3. What insights can be gained about available parking resources in the study area based on the spatio-temporal distribution and durations of cruising for parking trip trajectories?

The research questions of Chapter 4 are:

1. Using big trajectory device data, what methods can be used to capture change of traffic status for actual road networks at certain time intervals compare with free-flow traffic status and historic average traffic status?
2. What are the benefits of a W-Net-based deep learning model in automatically segmenting congestion areas from noisy contour plots compared to traditional clustering algorithms?

3. What methods can be used to distinguish the impact areas of NRC from the RC and overcome the challenges of both traffic fluctuations and RC impact area fluctuations?

Chapter 2: Estimating traffic speed and speeding using passively collected big mobility data and a distributed computing framework

2.1 Abstract:

With the increasing availability of location-aware devices, passively collected big GPS trajectory data offers new opportunities for analyzing human mobility. Processing big GPS trajectory data, especially extracting information from billions of trajectory points and assigning information to corresponding road segments in road networks is a challenging but necessary task for researchers to take full advantage of big data. In this research, we propose an Apache Spark and Sedona-based computing framework that is capable of estimating traffic speeds for state-wide road networks from GPS trajectory data. Taking advantage of Spatial Resilient Distributed Datasets supported by Sedona, the framework provides high computing efficiency while using affordable computing resources for map matching and waypoint gap filling. Using a mobility dataset of 126 million trajectory points collected in California, and a road network inclusive of all road types, we computed hourly speed estimates for approximately 600,000 segments across the state. Comparing speed estimates for freeway segments with speed limits, our speed estimates showed that speeding on freeways occurred mostly during the nighttime, while analysis of travel on residential roads showed that speeds were relatively stable over the 24-hour period.

2.2 Introduction

With the rapid development of intelligent transportation systems (ITS) (Lin et al., 2017), the acquisition, evaluation, and prediction of urban traffic conditions, i.e., traffic

speed and traffic volume have become a major focus of research (Yu et al., 2020; Yu & Gu, 2019; Zhang et al., 2015). Traffic speed, an important component of urban mobility, is not only critical for applications relating to ITS, but also for the development of sustainable urban transit systems, and for improving understanding of interactions between people and cities (Wang et al., 2020).

In prior research, mobility data collected from monitoring stations and travel surveys that contain GPS records have been widely used to estimate traffic conditions (Kwon et al., 2003; Yokoo & Levinson, 2019; J. Zhang et al., 2015). However, these data are often limited to relatively small spatial coverages and sample volumes (e.g., selected major roads or freeways in a city and several hundred users for survey data), and may fall short of reflecting traffic conditions over larger spatio-temporal scales (C. Chen et al., 2016). In recent years, big GPS trajectory data that is passively collected by mobile devices using location-aware sensors in smart phones and in-vehicle sensors, have provided new possibilities for larger-scale and higher-precision speed estimation (C. Chen et al., 2016). While big GPS trajectory data is promising for the development of ITS, how to extract traffic speed estimates over billions of raw GPS waypoints and assign speed information to each road segment, is a serious computing challenge (C. Chen et al., 2016). A computing framework that can effectively estimate travel speeds on a road network using big trajectory data at larger spatial scales (e.g., at the scale of large urban agglomerations or at state level) still needs to be specified.

In this article, we discuss the design and development of a distributed computing framework that uses Apache Spark and Apache Sedona (Armbrust et al., 2015; J. Yu et al., 2015; Zaharia et al., 2010) to extract traffic speed data from billions of waypoint data

generated from smart devices, and assign these values to corresponding road segments to provide hourly estimates of traffic speed for all road segments. The computing framework is comprised of four modules: data preprocessing, map matching, waypoint gap filling, and speed estimation for road segments. The computing framework is designed using Apache Spark and Apache Sedona that are in-memory distributed computing environments. The in-memory Resilient Distributed Datasets (RDDs) supported by Spark, make this environment more effective than traditional Map Reduce for complex data analysis (Zaharia et al., 2016). Spatial RDDs (SRDDs), which are supported by Sedona extend the regular Spark RDDs and offer support for spatial data types and spatial indexing (J. Yu et al., 2019a). This makes Sedona even more suitable for handling large scale (i.e., large extent) geospatial data. The Spark-based data preprocessing module is designed to filter raw GPS trajectory data by detecting anomalies and data redundancy for both the mobility dataset and the road network, and to compute traffic speeds for the waypoints. Using the SRDDs and KDB-Tree spatial partitioning (Robinson, 1981), hundreds of millions of waypoints are aligned to a state-level road network, and Apache Spark and iGraph (Ju et al., 2016) are both used to connect adjacent trajectory points that have been matched to road segments in each trip. With these latter two steps, traffic speeds for segments that connect adjacent trajectory points can be estimated. Finally, traffic speeds for all road segments in California are estimated at hourly time intervals by aggregating the travel speed information that has been matched to road segments through Spark partitioning.

The approach is tested using mobility data collected over a 24-hour period for the State of California (3.4 million road segments and 126 million mobile trajectory points)

on 02/01/2020. By applying the computing framework on this dataset as well as a road network dataset for CA, we return hourly speed estimates for 596,186 road segments that represent approximately 17% of all CA roads and 46% of freeway road segments. Traffic speed estimates for many of these segments would not otherwise be available due to the pattern of monitoring stations on State roads. Traffic speeds over the 24-hour period and for freeways and residential roads are analyzed using this framework. An analysis of speeding behaviors (travel that exceeds the posted speed limit) using two different thresholds–travel that exceeds the speed limit by 20% and by 50% for freeways in CA returns segments that could be monitored for managing speeding behaviors.

The rest of the paper is organized as follows: Section 2 discusses related work on estimating traffic speed using big data; Section 3 introduces the study area and datasets used in this research; Section 4 elaborates on the design of the four modules for the traffic speed estimation framework that use Spark and Sedona. Section 5 discusses the speed estimate results for the study area; and Section 6 presents our conclusions and ideas for future research.

2.3 Related work

The estimation of traffic conditions, such as traffic speed and traffic volume, are important for studying human travel behaviors and people's interactions with urban environments. Data from traffic detectors, for example, loop detectors and radar have been used for designing a real-time truck volume estimation algorithm that used a single loop detector in California (Kwon et al., 2003), and to study the uncertainty and variability of traffic volume estimates using data from a monitoring station equipped with radar and sound-level meters in Poland (Bakowski et al., 2018). Data from a single loop

detector was also used to estimate mean vehicle length and mean travel speed (Y. Wang & Nihan, 2000). These three studies contributed methods for more accurate traffic condition estimation at the lane and intersection level, and for examining travel modes in cities and their patterns over time. However, detector data has the limitation of generating a smaller data sample volume, and being sparsely deployed on a road network, making these data less effective for understanding travel behaviors at larger spatial scales (Seo et al., 2017).

Researchers have also applied actively collected GPS data to estimate traffic conditions, for example, a GPS-enabled cellphone-based traffic monitoring system was tested using 100 vehicles in the San Francisco Bay area in California (Herrera et al., 2010). Travel survey data with GPS records for 152 users over a 7-day period in Minneapolis-St. Paul, MN was used to identify and analyze road segments where speeding occurred. Researchers in this study also investigated the relationship between different factors such as road segment length and speed limits, with drivers' characteristics (e.g., age, educational background and gender) and frequency of speeding (Yokoo & Levinson, 2019).

With the development of technology and the increasing availability of location awareness equipment, passively collected big trajectory data has become available for transportation studies. A cell-based map-matching algorithm was proposed to link GPS points with road segments, and innovatively applied a Schatten p-norm matrix completion algorithm to address the data limitations (J. Yu et al., 2020). The authors of this study used trajectory data and road network data for Chengdu, China as a case study to reveal congestion in road segments using their results. Kan et al., (2019) proposed a

turn-level traffic congestion estimation method and used taxis' GPS trajectory data for one day in Wuhan, China to prove the feasibility of their algorithm. Zhao et al., (2019) proposed an optimized Gate Recurrent Units algorithm to predict the driving speed of trucks using truck trajectory data for highways in part of an urban expressway in Beijing, China. The research methods applied in these studies were designed for city-scale or more local scales and are not as suitable for trajectory data and road network data for larger urban agglomerations and state-level studies. Even fewer studies have proposed a distributed computing approach that is capable of estimating traffic conditions at larger spatial scales. Fan et al., (2019) proposed an Apache Spark-based geo-computing framework capable of estimating vehicle miles traveled using big GPS data for the State of Maryland and balanced the map-matching accuracy and the computing time.

Recently, Uber Movement released travel time, aggregated travel speed, and mobility heatmap datasets for certain large cities around the world. The travel speed dataset is aggregated for one-hour intervals for road segments, and the travel time dataset is aggregated for small zones. Both datasets are based on GPS trajectories collected from Uber drivers. Researchers have applied Uber Movement datasets in urban mobility and transportation studies. Roy et al., (2020) examine the daily mobility pattern for weekdays and weekends in the Miami Metropolitan area from 2016 to 2019 and performed spatial clustering for Uber trips in the study area. Using this dataset, the researchers found spatial and temporal clustering along the major traffic arteries. Sun et al., (2020) conducted research on commuting times using the Uber Movement travel time data as a proxy for commuting time data using Greater Boston as the study area.

In recent years, different kinds of distributed computing frameworks optimized for big spatial data management and analysis have emerged, such as Hadoop-based approaches including Hadoop-GIS and Spatial Hadoop, and Spark-based approaches including Apache Sedona, Spatial Spark and GeoMesa (Aji et al., 2013; Eldawy & Mokbel, 2015; Hughes et al., 2015; You et al., 2015). In this research, we chose Spark-based approaches over Hadoop-based ones because although extended Hadoop approaches provided support for spatial data types and spatial indexing, the limitation of relatively slow reading and writing speed on disks remains for the traditional map-reduce models (J. Yu et al., 2019b). For the limitations that Spatial Spark does not offer support for spatial SQL, and GeoMesa does not remove duplicated spatial objects introduced by partitioning the data and therefore can potentially produce inaccurate spatial query results, we chose Apache Sedona for this research (J. Yu et al., 2019b). Using the combination of KDB-Tree spatial partitioning and R-Tree index, the framework proposed in our study achieved an improved computing efficiency while keeping a high accuracy when estimating traffic speed at the spatial scale of a statewide network.

2.4. Study area

In this research, a detailed road network dataset from OSM for the State of California, and a passively collected mobile device dataset provided by the Maryland Transportation Institute (MTI) at the University of Maryland were used to design the travel speed estimation framework. The OSM road network data offered the advantages of timely updates, being open-source, and containing rich metadata, such as road type, road direction, and speed limit information (OpenStreetMap contributors, 2020). In OSM, the road type is represented in the “highway” attribute, and selected road types included

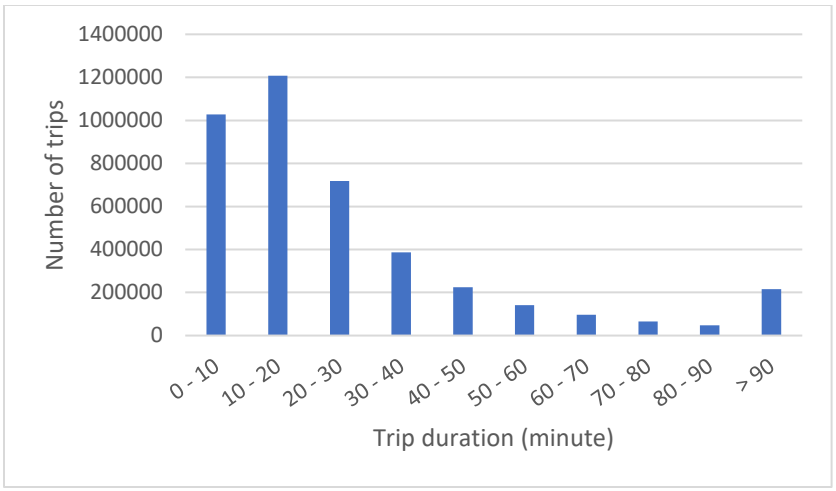
motorway, motorway_link, primary, primary_link, residential, road, secondary, secondary_link, tertiary, tertiary_link, trunk, trunk_link, and unclassified (Table 1). Road direction information is expressed by the "oneway" attribute. If this attribute is “True” or “Yes”, it means that a vehicle traveling on the corresponding segment can only travel from the start node to an end node of the segment. If this attribute is “No”, then the segment is not a one-way street, and drivers can drive from either the start to end node or *vice versa*. If the attribute is “-1”, drivers may only drive from an end node to a start node. If the attribute is “None” or the segment does not have this attribute, the segment is defined as a two-way street.

Table 2. 1 Number and percentage of road segments for different road types in California

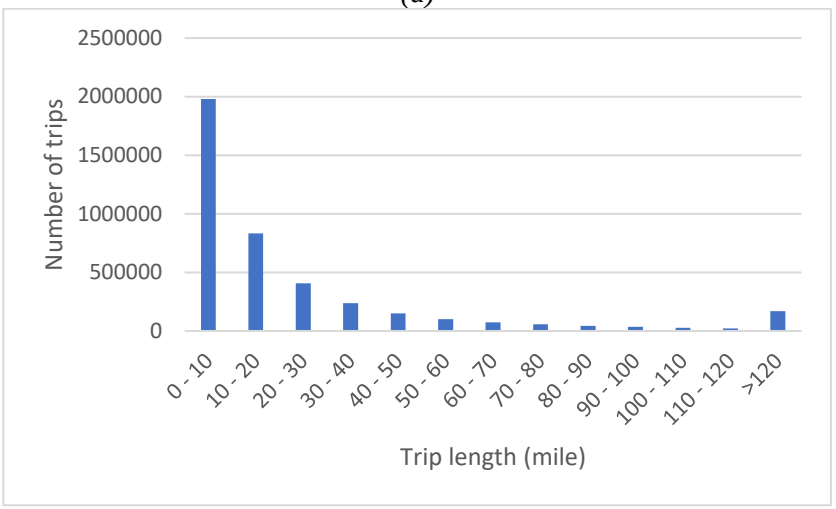
Road type	Number of road segments	Percentage of road segments (%)
Motorway	19920	1.22
Motorway link	31001	1.90
Primary	74562	4.58
Primary link	4163	0.26
Residential	650515	39.92
Road	923	0.06
Secondary	108136	6.64
Secondary link	4484	0.28
Service	636138	39.04
Tertiary	65770	4.04
Tertiary link	1669	0.10
Trunk	6802	0.42
Trunk link	1840	0.11
Unclassified	23441	1.44
Total	1629364	100

The mobility dataset used in this research was provided by the Maryland Transportation Institute and was obtained from different leading data vendors that collect anonymized mobile device location data. The dataset represents passively collected GPS waypoints contributed by location-based apps used by smart devices that recorded locations for the vehicular traffic. Each record in the mobility dataset is a waypoint, and

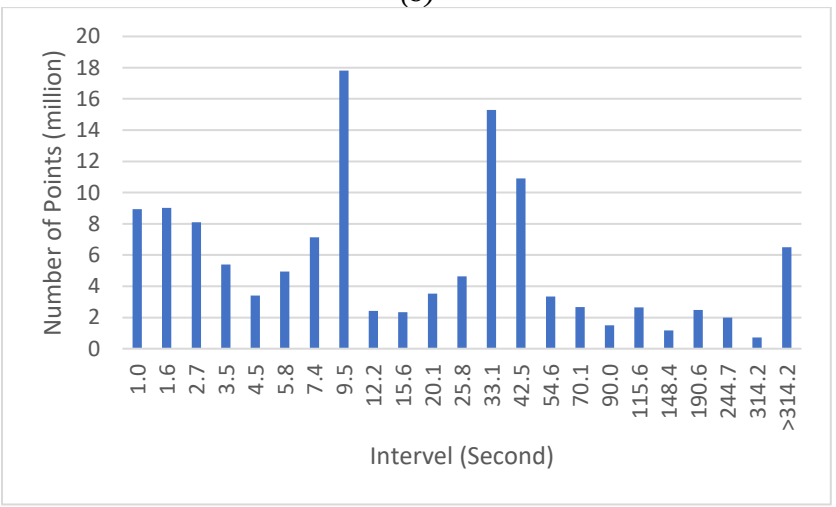
contains attributes longitude, latitude, device ID, trip ID, time stamp, start and end longitude and latitude for each trip, trip distance, and trip duration. Each trip is identified by a unique trip id, with a trip having an average of 30.70 waypoints. The average duration of trips on the day of data collection was 30.73 minutes (fig 2.1 a), with an average distance of 13.88 miles (fig 2.1b). Fifty-one percent of the waypoints in the mobility dataset had a sampling frequency ≤ 10 seconds, 11% had a sampling frequency between 10 - 30 seconds, and 22% had a sampling frequency between 30 seconds and 1 minute. The remaining 16% had frequencies between 1 minute and 5 minutes long (less than 3% of waypoints in our dataset had more than 5-minute sampling frequency) (fig 2.1 c, the horizontal axis applies the power of universal constant). This indicates that overall, this dataset was characterized by a high sampling frequency (Z. Huang et al., 2021). For the accuracy of our speed estimation results, waypoints that had more than 90 seconds sampling frequency were not used to estimate travel speeds. By randomly sampling 5% of origin and destination points (OD points), and 1% of all waypoints in the mobility dataset, a heatmap of OD points shows that most of the OD points were in cities (fig 2.2 a), and waypoints were mainly located near or on freeways (fig 2.2 b).



(a)

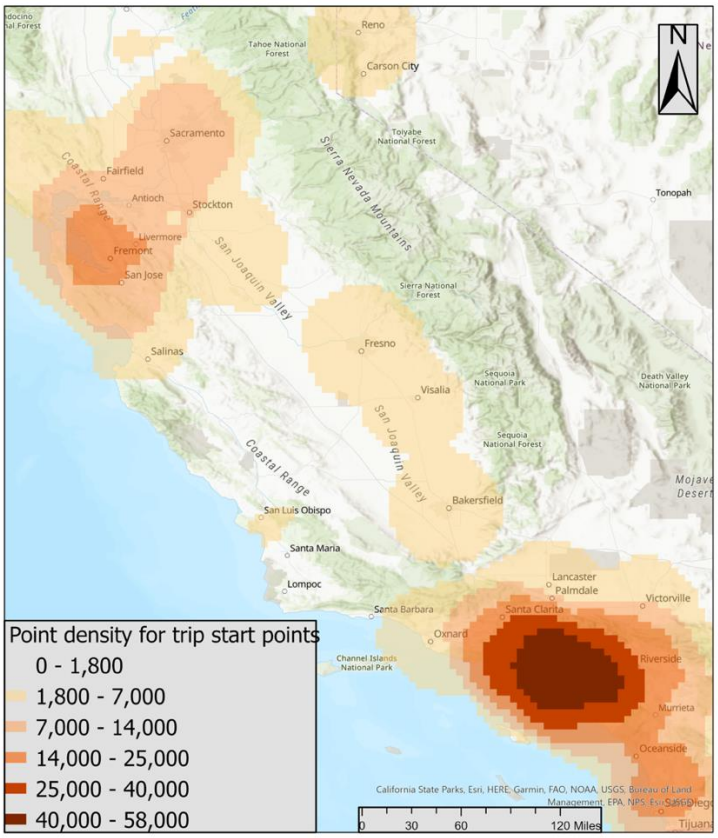


(b)

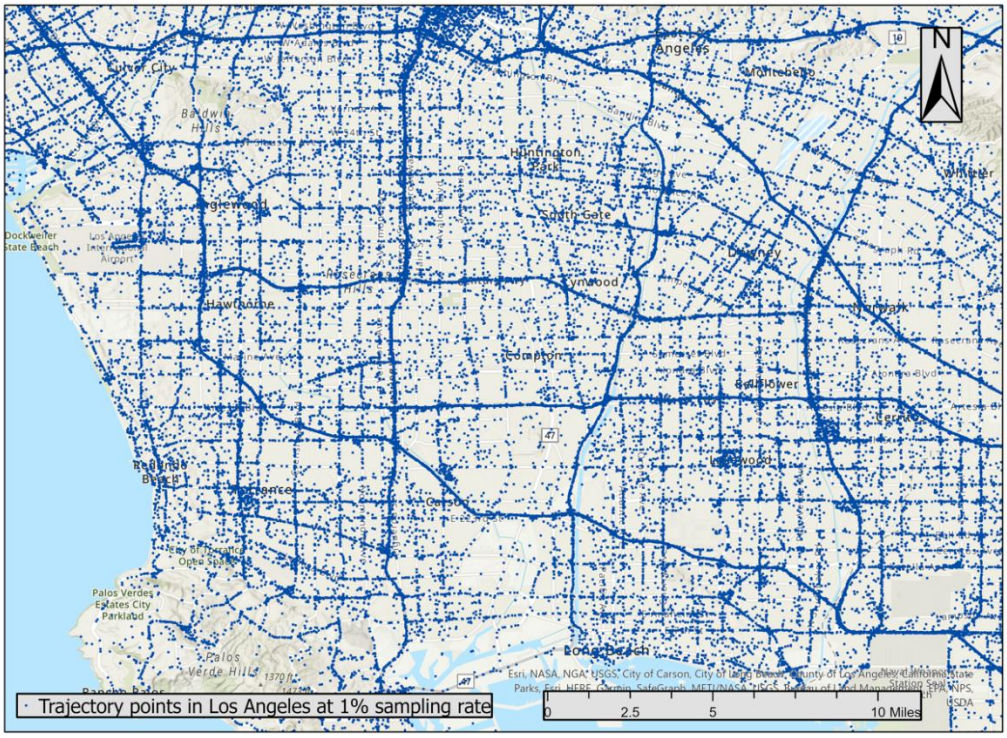


(c)

Figure 2. 1 Distributions of (a) trip durations; (b) travel distance; (c) sampling intervals of waypoints.


Reno
Tahoe National Forest
Carson City
Sacramento
Toiyabe National Forest
Sierra Nevada Mountains
Fairfield
Coastal Range
Antioch
Stockton
Tonopah
Livermore
Fremont
San Jose
San Joaquin Valley
Sierra National Forest
Fresno
Salinas
Sequoia National Park
Death Valley National Park
Visalia
Sequoia National Forest
Coastal Range
San Joaquin Valley
Bakersfield
San Luis Obispo
Santa Maria
Lompoc
Lancaster
Palmdale
Victorville
Santa Barbara
Santa Clarita
Oxnard
Channel Islands National Park
Riverside
Murrieta
Oceanside
Point density for trip start points
0 - 1,800
1,800 - 7,000
7,000 - 14,000
14,000 - 25,000
25,000 - 40,000
40,000 - 58,000
0
30
60
120 Miles

(a)



(b)

Figure 2. 2 (a) Heatmap of origin and destination points of the mobility dataset for 02/01/2020 in California, sampling 5% of the entire mobility dataset and (b) spatial distribution of mobility waypoints for 02/01/2020 California, sampling 1% of the entire mobility

2.5 Traffic speed estimation using a distributed computing approach

To compute travel speed, speed information was extracted from each waypoint, and assigned to the corresponding road segment. The results were aggregated for each road segment. To overcome the challenge of massive data volumes, large spatial coverage, and complexity of spatial computation, we designed a distributed computing framework that is comprised of four modules as described in the following sections.

2.5.1 The preprocessing module

The preprocessing module was designed to detect and remove anomalies, such as redundant and incomplete records in both the mobility dataset and OSM road network,

and extract speed, and travel directions from the mobility dataset and "one-way" information, speed limits, and segment directions from the road network dataset.

The anomalies in the mobility dataset were classified into trip anomalies and waypoint anomalies. Trip anomalies refer to trips with less than a minimum threshold of 2 waypoints, i.e., where there were too few waypoints to extract speed and travel direction. These anomalous records were removed from the trajectory dataset. Waypoint anomalies referred to cases of redundant waypoints and timestamp errors. For example, in a single trip if there were two consecutive waypoints or waypoints separated by no more than 3 waypoints with the same coordinate values (this threshold was selected after manually checking hundreds of trips) then these points could be anomalies. Waypoint anomalies were discovered by testing whether the coordinates of two consecutive points were the same or whether there were two consecutive 180-degree turns within four waypoints (the number was also selected after checking hundreds of trips) creating redundancies in the dataset. The speed and direction of each waypoint n_i was calculated based on the coordinate and timestamp of waypoint n_i and the next point in the same trip n_{i+1}:

$$V(n_i) = \frac{\Delta(n_i, n_{i+1})}{\Delta(t_i, t_{i+1})} \quad (1)$$

Where $\Delta(\cdot)$ refers to the difference method

$$Dir(n_j, n_{j+1}) = \frac{\Delta(Y_{i+1}, Y_i)}{\Delta(X_{i+1}, X_i)} \quad (2)$$

For the last waypoint in each trip, the speed and direction of that point was set to be equal to that of the previous waypoint. Waypoint anomalies were removed from each trajectory in the trajectory dataset after detection.

The most common anomalies in the OSM road network data were topology errors, typically road segments that were not divided into separated segments at intersections, as well as some redundant road segments. In the preprocessing module, road segments were split at intersections and all the redundant road segments were deleted. The direction of each segment S_j was calculated using the start and end node of the segment:

$$Dir(S_j) = \frac{\Delta(Y_{end}, Y_{start})}{\Delta(X_{end}, X_{start})} \quad (3)$$

2.5.2 The map matching module

Map-matching is a commonly applied step in trajectory reconstruction (C. Chen, Zhang, et al., 2014; M. Li et al., 2019). The main purpose of map matching is to align trajectory waypoints that represent the position of mobile devices at a corresponding timestamp to the most appropriate road segment. The main challenge in this process was due to sampling errors generated by the GPS sensors in the mobile devices (7-13 meters horizontal error on average with a maximum error within 100 meters (Merry & Bettinger, 2019)), with the result that waypoint coordinates were unlikely to be located accurately with respect to their corresponding road segment. In addition, because some locations had dense road networks, especially multi-lane road networks in cities and urban areas, waypoints were often closer to a non-corresponding road segment than the correct segment.

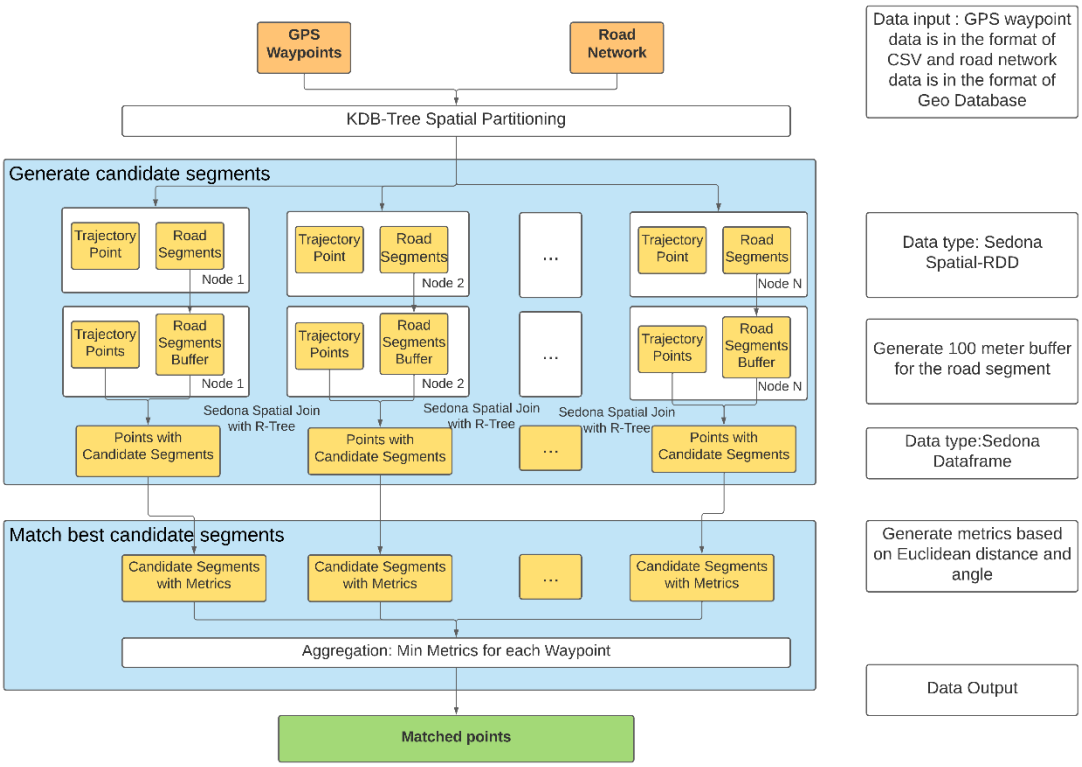


Figure 2. 3 Structure of the map-matching module. Orange rectangles represent input data, yellow rectangles represent intermediary data, and the green rectangle represents output data. Blue rectangles represent the two main steps of the map matching module, and the nodes are computing nodes in the computational cluster.

In previous studies, features such as the distance between trajectory points and candidate road segments, topology, and speed of waypoints have been applied as parameters in the map matching methods (Bernstein & Kornhauser, 1996; Hsueh & Chen, 2018; White et al., 2000). The majority of previous map matching methods used datasets of relatively small data volume (no more than 10 million waypoints) and spatial coverage (city-level), and as such are not so suitable for the task of map matching big trajectory data. Fan et al. (Fan et al., 2019) proposed a map matching method using Apache Spark that considered both the distance between waypoints and the candidate segments and the angle between the trip and the candidate segments. They also applied an R-Tree index in order to balance the matching accuracy and the computing efficiency

when handling a big trajectory map matching task.

Using Apache Sedona for this research we designed a distributed map matching module that also uses the Euclidean distance between waypoints and road segments, and the angle between the trip direction and the direction of segments. However, we extended this approach by considering lane information from the road network, and we used KDB-Tree spatial partitioning that is supported by Sedona to first assign trajectory points and the road network as Point RDDs and Line RDDs respectively for higher computing efficiency (fig. 2.3).

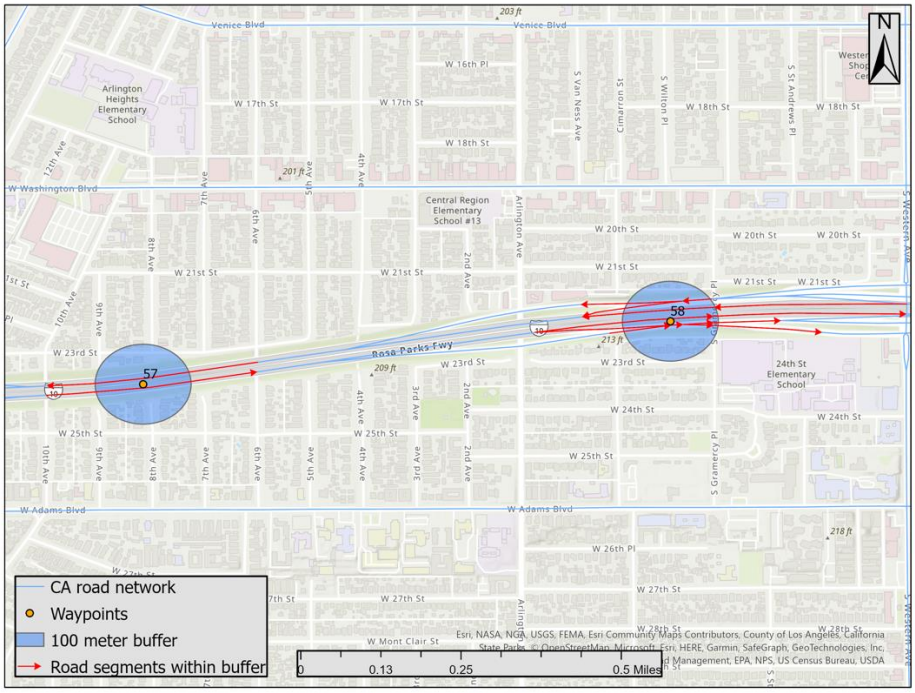


Figure 2. 4 Waypoints and candidate segments for matching. Directions of candidate segments are shown with red arrows.

The map-matching algorithm consists of two steps:

a) Generate candidate segments: Search all road segments within 100 meters of the waypoint n_i as candidate segments. We used an R-Tree index as a local spatial

index in this step to improve computing efficiency (fig 2.4).

b) Match best candidate segment: Calculate Euclidean distance between each waypoint and all candidate segments, and the angle between trip direction at the current point n_i: $dir(n_i, n_{i+1})$ and direction of candidate segments $dir(Segment_j)$. This angle is calculated as:

$$Angle(n_i, Segment_j) = \frac{\arccos\left[(dir(n_i, n_{i+1}) \cdot dir(Segment_j)\right)}{|dir(n_i, n_{i+1})| * |dir(Segment_j)|}] \quad (4)$$

Because there is no n_{i+1} for the last waypoint of each trip, the direction of the last waypoint was assigned the same value as the second-to-last waypoint. Lane information (e.g., one-way, or two-way) was also considered for candidate segments:

$$Angle(n_i, Segment_j) = Angle(n_i, Segment_j), when\ oneway == yes;$$

$$Angle(n_i, Segment_j) = 180 - Angle(n_i, Segment_j), when\ oneway == -1;$$

$$Angle(n_i, Segment_j) = \min\left(Angle(n_i, Segment_j), 180 - Angle(n_i, Segment_j)\right),$$
$$when\ oneway == None;$$

Based on these elements (Euclidean distance, angular measurement) a compound metric was calculated for each point n_i, and its candidate segments $Segment_j$ as:

$$M(n_i, Segment_j) = distance(n_i, Segment_j), when\ Angle(n_i, Segment_j) < 45$$

$$M(n_i, Segment_j) = distance(n_i, Segment_j) * 10,$$
$$when\ 45 < Angle(n_i, Segment_j) < 90$$

$$M(n_i, Segment_j) = distance(n_i, Segment_j) * 100, when\ Angle(n_i, Segment_j) < 90$$

This metric emphasizes the role of the angle between the trip and the road segment. If this angle is less than 90 degrees, the Euclidean distance between waypoints and segment was multiplied by a small penalty coefficient of 10, otherwise a larger penalty coefficient of 100 was applied. The module returns this value and the segment with the lowest value for each waypoint is selected as the most appropriate segment for matching.

The module is designed using Apache Sedona, and we transformed the datasets to the format of Sedona SRDDs, with the mobility data as Point RDD and the road network data as Line RDD respectively. For the generate candidate segments step described above, we applied SedonaSQL for spatial joins using the Python API for Sedona to generate 100 meter buffers around waypoints and to extract all segments that intersected with these buffers. Distances and angles were calculated between waypoints and their candidate road segments. To further accelerate the module, an R-Tree spatial index was applied as part of the spatial join process.

2.5.3 The waypoint gap filling module to achieve a continuous trajectory

Because of the irregular nature of the waypoint sampling intervals as discussed previously, smart mobile devices only record the coordinates of certain locations along a trajectory, leaving gaps between waypoints. By closing the gaps between waypoints that have been matched to the correct segments in a single trip, and by assigning travel speed information to these road segments, we were able to compute the speed for an entire

trajectory instead of separate segments (fig 2 .5). For this reason, a waypoint gap filling module is necessary for this analysis, we assumed vehicles traveled at a constant speed between two consecutive waypoints in a single trip.

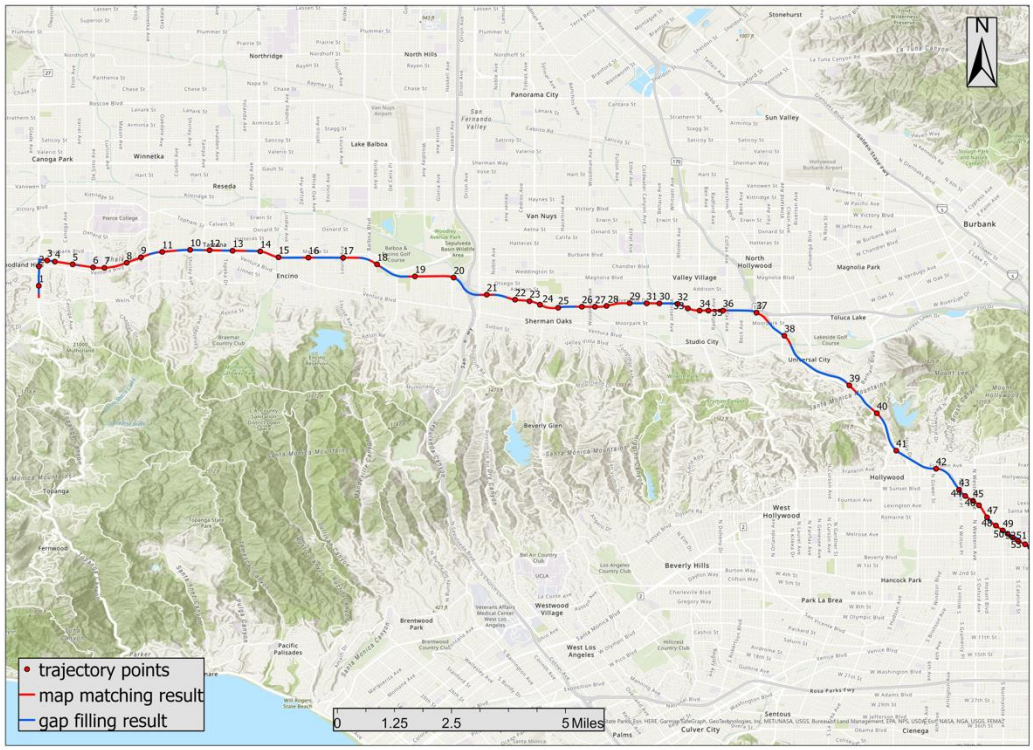


Figure 2. 5 Results after map matching and waypoint gap filling for a single trip from Woodland Hills to Downtown Los Angeles, CA. The digits on waypoints represent the sequence of points in the trip.

The shortest path algorithm based on iGraph and Spark was used for the waypoint gap filling module. Using Python API for Spark, we applied the directed graph of the road network generated by iGraph as a broadcast variable, which could be accessed by all executors in the computing cluster. Then, the shortest path algorithm in iGraph was used to retrieve segments where there were gaps between adjacent segments. In this research, we used a weighted and directed network with approximately 2.6 million vertices and 6.4

million edges to represent the entire CA road network. Given the size of road network, a large percentage of the computing time for all the tasks in the framework was spent on this module.

2.5.4 The speed estimation module

The final step of estimating traffic speed for road segments is accomplished in the speed estimation module. The results generated by map matching and gap filling are structured using a Spark Dataframe that stores the corresponding relationships between waypoints and their matched road segments.

Then, grouping by road segment ID and time interval, two methods for calculating mean values, namely using an arithmetic mean and a harmonic mean were tested in this module to determine the speeds of all trajectory points assigned to certain segments at different time intervals.

The arithmetic means and harmonic mean for segment S for time interval t $S^{A}_{i,j,t}$, and $S^{H}_{i,j,t}$ are calculated as:

$$S^{A}_{i,j,t} = \frac{\sum_{i=1}^{n} x_i + \sum_{j=1}^{m} f_j}{m+n} \tag{5}$$

$$S^{H}_{i,j,t} = \frac{m+n}{\sum_{i=1}^{n} \frac{1}{x_i} + \sum_{j=1}^{m} \frac{1}{f_j}} \tag{6}$$

Where x_i represents speed information of waypoints that has been matched to segment S for time interval t, and f_j represents the filling speed information that generated in the gap filling module.

In this way, speeds of road segments were aggregated at hourly time intervals. For this analysis, road segments with fewer than 5 waypoints in a time interval were not assigned a speed (i.e., were labelled missing) to avoid producing results that were possibly biased due to insufficient data.

Applying our framework on a Hadoop cluster of 100 CPU cores and 10Gbs network bandwidth between nodes, we analyzed the mobility dataset for trips made on 02/01/2020 in CA. The overall processing time for estimating travel speeds for the entire CA road network was approximately 7 hours and 45 minutes. Taking advantage of Apache Sedona, the map matching module took approximately 25 minutes, approximately 4 times faster than the map matching module based on Spark and using R-Tree index where it took approximately 2 hours (Fan et al., 2019). We also tested our map matching module without KDB-Tree spatial partitioning for the map-matching task, and it took approximately 53 minutes, which we believe showed the importance of spatial partitioning in the map matching module. The majority of processing time was spent on the gap filling processing that took approximately 7 hours and where approximately 45 million gaps in trips were filled to produce continuous trips.

2.6 Estimating travel speeds for California roads on 02/01/2020

In this study, hourly speed estimates were generated for 596,186 road segments representing 17.3% of the total state network (not all segments had speed estimates for each hourly interval) (fig 2.6). Approximately 46% of all freeway segments had valid speed estimates including 72.36% of motorway segments, which are the most traveled road types. Since the data was collected for a Saturday, our results represent weekend

trips and do not reflect commuting patterns with morning and evening peaks corresponding to home-work trips, but rather capture driving patterns on weekends.

From the speed estimation results, traffic speed estimates for freeways were generally faster than other road types, such as residential and service roads. City centers, for example, downtown Los Angeles, had more residential segments with valid speed estimates, but the speeds were slower for those segments.

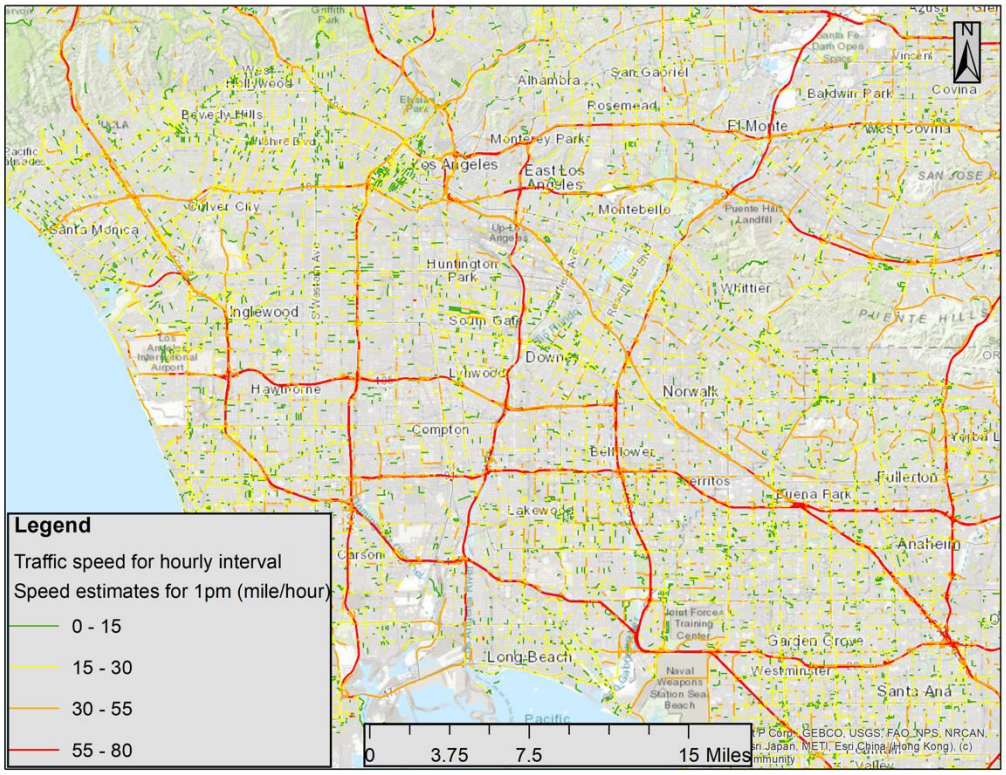


Figure 2. 6 Traffic speed estimates for 1PM, 02/01/2020.

The availability of hourly speed estimates for road segments based on the number of waypoints varied throughout the day. There was an increase in the number of segments for which speed estimates could be made between 6am and 4pm, and travel speeds for a relatively high number of segments could also be estimated for time slots between 4pm

and 8pm (Table 2.2). Between 8pm and 12am, however, the number of road segments for which valid speed estimates could be made decreased, and this decrease continued until 6am, when the lowest number of speed estimates were available.

Table 2. 2 Number of segments with speed estimates by hour for 02/01/2020.

Hour	Number of segments	Percentage of segments (%)	Hour	Number of segments	Percentage of segments (%)
12am	76025	3.29	12pm	126998	5.49
1am	59764	2.58	1pm	151121	6.53
2am	39382	1.7	2pm	169519	7.33
3am	21663	0.94	3pm	182719	7.9
4am	14810	0.64	4pm	182824	7.9
5am	10532	0.46	5pm	176255	7.62
6am	8549	0.37	6pm	169096	7.31
7am	11759	0.51	7pm	174706	7.55
8am	19469	0.84	8pm	171674	7.42
9am	34225	1.48	9pm	153005	6.61
10am	59395	2.57	10pm	116543	5.04
11am	93951	4.06	11pm	90256	3.9

2.6.1 Traffic speeds for different road types

Analysis of travel speeds for different road types (Table 2.3) showed that freeway road types including motorway, trunk, primary way, and secondary way road types had a relatively higher number of waypoints assigned to each hourly time interval. Over the course of the day, we calculated speed estimates for approximately 72% of motorways, 57% of trunk roads, 49% of primary ways (i.e., mostly highway and some state roads that connected large towns/cities, or major roads in an urban area), and 38% of secondary way (i.e., state highways or major urban streets that were not classified as primary) road segments (*Key:Highway*, n.d.). Contrary to freeway road types, fewer waypoints were generally available for residential roads, with only about 8% of residential road segments

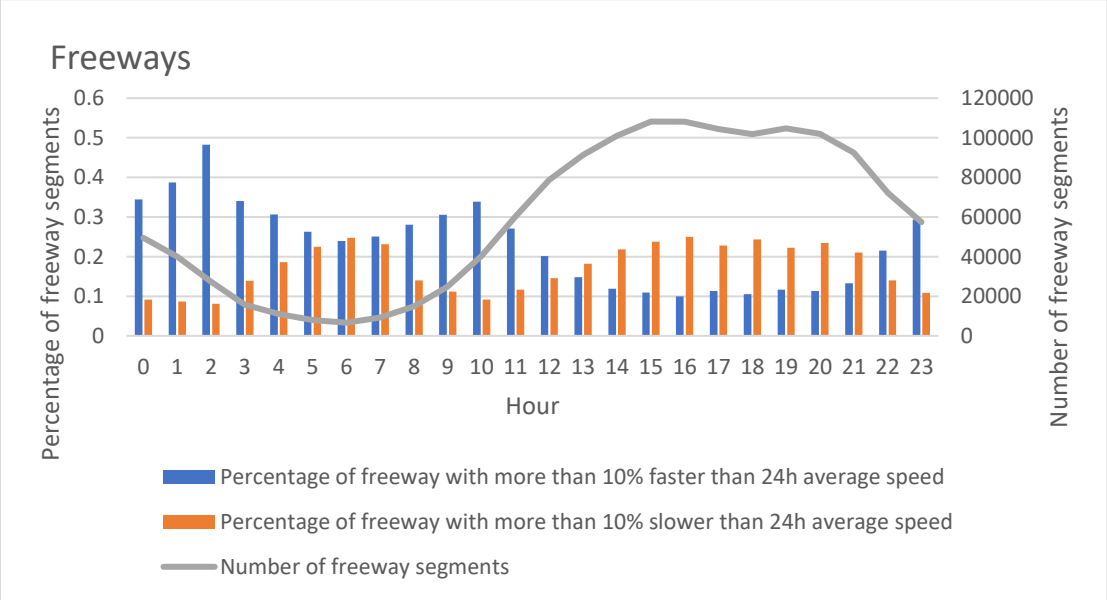
having valid speed estimates. For this reason, while we believe our speed estimates hold for the majority of freeway road types, they may not be representative of all residential roads and road types such as service and tertiary roads.

Table 2. 3 Number of road segments with speed estimates by road type

Freeway or residential road types	Road type	Number of segments with speed estimates	Segments by road type with speed estimates (%)
Freeway	Motorway	33241	72.36
	Trunk	7366	56.6
	Primary	69956	49.02
	Secondary	91614	38.32
Residential road	Residential	118653	7.87
Other roads	Motorway link	21208	46.55
	Primary link	701	15.2
	Road	41	2.79
	Secondary link	528	10.99
	Service	67028	5.87
	Tertiary	52478	23.17
	Tertiary link	121	6.67
	Trunk link	429	19.82
	Unclassified	5148	7.78

Analysis of traffic speed (fig 2.7 a) revealed different patterns by hour for traffic speeds on freeways *vs.* residential roads. To further analyze driving behaviors with respect to travel speed, we applied a threshold of 10% of the daily average speed to capture speeds that were 10% higher or lower than usual for each hour. For all categories of freeway road, there were significantly more road segments with faster than average speeds between 8am and 12pm, between 10pm and 11pm, and between 12am and 5am.

Between 1pm and 9pm, there were relatively higher numbers of segments with 10% slower than average speeds. From 2am to 6am, the proportion of freeway segments that were faster than average had a significant drop and then bounced back from 7am to 10am. Residential roads showed a different pattern of travel speed over time, with residential roads having a relative stable proportion of segments that were 10% faster or 10% slower than the average over the day, and more segments with slower speeds during nighttime hours. The number of segments for which speeds could be calculated for each hourly interval for residential roads followed a similar trend to freeways (i.e., like a trigonometric curve, with peaks and valley values at 6am and 5pm, respectively) (fig 2.7 b).



(a)

(b)

Figure 2. 7 Traffic speed over time for (a) freeways and (b) residential ways

2.6.2 Validating the speed estimates

We validated the speed estimates using freeway detector data from the Performance Measurement System (PeMS) available from the California Department of Transportation. PeMS provides average speeds for one-hour and five-minute intervals from various types of vehicle detector stations, including inductive loops, side-fire radar, and magnetometers over California freeways. We extracted hourly average speed estimates for Saturday, 02/01/2020 from 11,398 stations and assigned these stations to the CA freeway road network using the map-matching module, and using the coordinates and directions PeMS provides for each station. We were able to calculate average travel speeds for 7,775 segments, approximately 17% of all freeway road segments in the CA OSM dataset.

Using the arithmetic mean for aggregation in the speed estimation module produced the results that more closed to the stationary speeds than using the harmonic

mean. Using the arithmetic mean, 46% of segments had differences of less than 5 mph with the station data, 48% had differences of 5 - 15 mph, and only about 6% had differences of more than 15 mph.

We believe one reason for these differences is that our speed estimates were made for all road segments traversed by mobile device users, while stationary data estimates traffic speeds for all vehicles passing through the station during a certain time interval. For this reason, speed estimates for some time intervals may not be fully captured using mobile device data.

Two factors, namely the variance of the speed over each one-hour interval, and the number of waypoints matched to the segments had an impact on the differences that we found with our estimates. By applying the five-minute stationary data to measure the variance of segments in each hourly interval, Pearson coefficients showed a weak to moderately positive linear correlation of 0.30. We also used a Spearman's correlation test to measure the correlation between the number of waypoints assigned to a segment and the difference. The result was a weak non-linear correlation of 0.10. It is possible that the speed estimates could be sensitive to speeds of vehicles in different lanes of the same segment, but due to the accuracy of GPS sensors in smart devices as well as the unavailability of lane level road networks, we were not able to verify this aspect.

We also compared our results with the Uber Movement speed estimate product for February 1st, 2020, for San Francisco, the only city that Uber Movement provides a speed estimates product for in CA. Because the road network that Uber Movement uses for its speed estimates is not the same as the OSM road network that we used, we first matched the Uber Movement speeds to the OSM road segments. Although Uber

Movement provides a mapping between *Movement Segment IDs* to *OSM Way IDs*, the road networks they used are not one-to-one aligned with the OSM road network. And due to the fact that Uber Movement provides only speeds for road segments at one-hour intervals, we could not estimate speeds for those OSM segments that represent more than one Uber segment. After the extraction, approximately 262,000 speed estimates for approximately 30,000 segments were matched to the OSM road network, and around 27,000 segments had both our speed estimation results and Uber results.

After comparation, 42% of segments had differences of less than 5 mph with the Uber results, 50% had differences of 5 - 15 mph, and only about 8% had differences of more than 15 mph. In order to better understand the similarity between our speed estimates and the Uber Movement speed estimates, we used a paired t-test to test whether there is mean difference between our speed estimates and the Uber Movement product for different road types.

For primary and secondary roads, we could not reject the null hypothesis that the mean difference is zero ($P = 0.73 > 0.05$). But for motorways and trunk roads, results showed a P value less than 0.01, which means that the mean of the paired differences does not equal zero. For residential roads, we also had a P value less than 0.01, which means the paired differences does not equal zero for these road types also. These results suggest that for the primary and secondary roads, our speed estimates are very close to Uber Movement results.

One possible reason for the difference between our speed estimates and Uber Movement datasets is the difference in the source of GPS data providers. Uber Movement datasets were collected from Uber drivers instead of general mobile device users, and

therefore have a potential bias that is different to that associated to normal drivers. From previous research, Taxi drivers often drive faster than the general public especially on free-flow roads (Hochmair, 2016). Using the Greater Sydney region as a study area, results in Wu (2018) also showed that travel time from Uber Movement is systematically shorter than results extracted from Google API.

Another difference between the Uber Movement dataset and the speed estimates from our framework is that Uber Movement provides only aggregated speed estimates for one-hour intervals for certain large cities. Using our framework, researchers can get speed estimates for intervals with any duration, and also get individual-level traffic information such as percentage of speeding.

2.6.3 Speeding by drivers on CA roads

One advantage of applying big mobile device data in the field of transportation and mobility analysis is that this data can be used to reveal the travel behavior of individuals. As the majority of road segments (84.9%) with speed limit information in the OSM dataset were freeway segments, we analyzed speeding on this road segment type. We measured speeding based on the proportion of waypoints that exceeded a certain percentage of the speed limit as compared to all waypoints in a one-hour time interval.

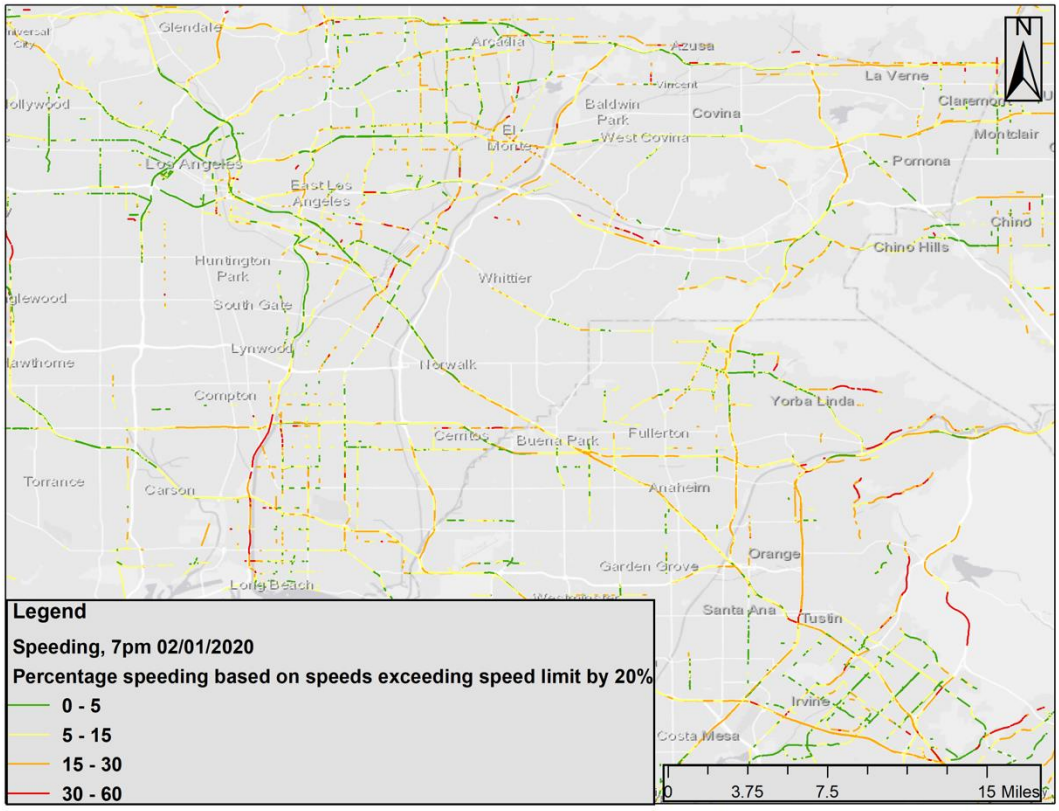
The speeding index is defined as:

$$I_{threshold} = \frac{N_{r(speed > speed\ limit*(1+\ threshold))}}{N_{r(all)}} \quad (7)$$

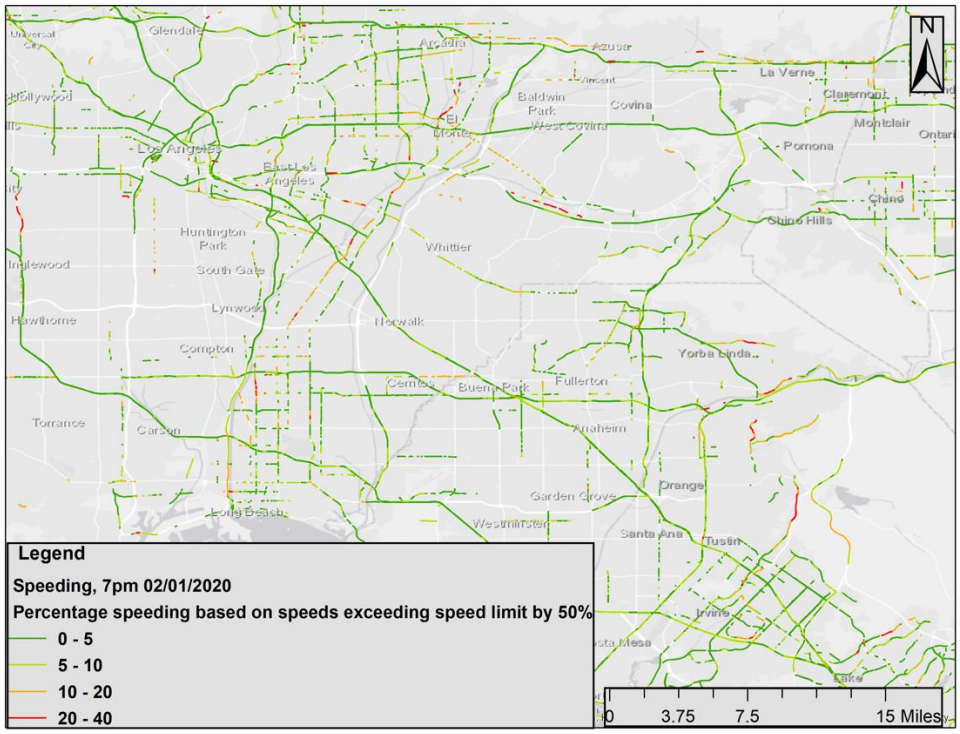
Where $I_{threshold}$ is the speeding indicator for a certain threshold, we examined speeding based on two different thresholds, speeds 20% and 50% faster than the speed limit. The first case represents travel speeds that may be more common (e.g., traveling at 66mph in a 55mph zone), while the second case represents faster driving (e.g., 83mph in a 55mph zone). $N_{r(all)}$ is the number of waypoints available per segment in a one-hour time interval, and $N_{r(speed > speed\ limit*(1+\ threshold))}$ is the number of waypoints that exceeded the speed limit by either a 20% or 50% threshold.

Using the lower 20% threshold, we found that approximately 47,802 freeway segments per hour had more than 30% of waypoints that met this threshold (fig. 2.8 a), while 124,794 freeway segments had 20 to 30% of waypoints per hour that met this threshold, and another, 307,901 segments had 10 to 20% of waypoints per hour exceeding the speed limit by 20%. Looking at the case where drivers exceeded the speed limit by over the 50%, results showed that almost 99,491 freeway segments had more than 10% of waypoints per hour at this higher threshold (fig. 2.8 b).

From a spatial perspective, speeding on this day did not appear to indicate a significant spatial pattern for either the 20% or 50% threshold. However, there were certain roads with a higher proportion of drivers that were speeding, for example, North Main Street in central Los Angeles (fig. 2.8), where 45% of segments on this street had at least 30% of the trips classified as speeding at more than 20%, and approximately 32% of segments on this street had at least 30% of the trips with speeding at more than 50% above the speed limit.


Legend
Speeding, 7pm 02/01/2020
Percentage speeding based on speeds exceeding speed limit by 20%
0 - 5
5 - 15
15 - 30
30 - 60
0
3.75
7.5
15 Miles
N
Glendale
Arcadia
Azusa
La Verne
Claremont
Baldwin Park
Covina
West Covina
El Monte
Montclair
Pomona
Los Angeles
East Los Angeles
Chino
Chino Hills
Huntington Park
Whittier
South Gate
Lynwood
Norwalk
Compton
Yorba Linda
Cerritos
Buena Park
Fullerton
Torrance
Carson
Anaheim
Orange
Garden Grove
Long Beach
Santa Ana
Tustin
Irvine
Costa Mesa

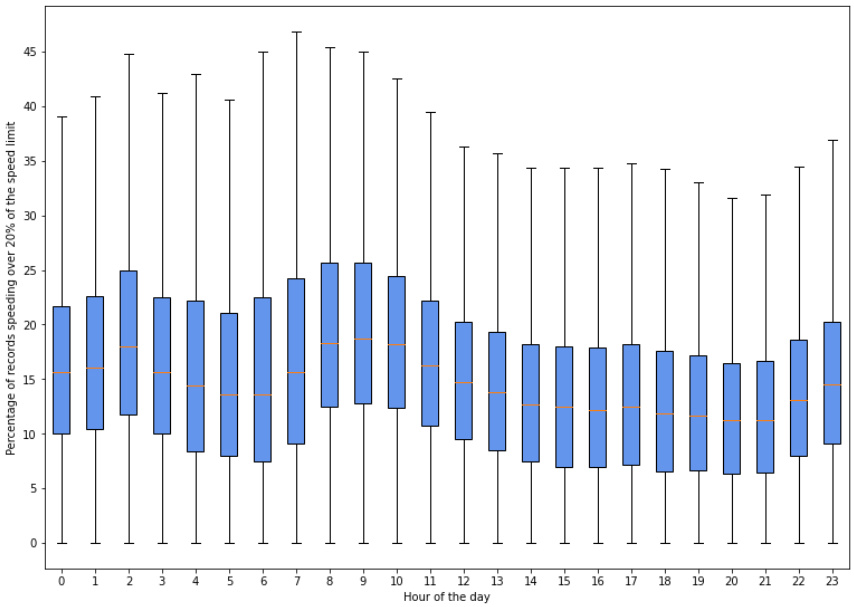
(a)


Legend
Speeding, 7pm 02/01/2020
Percentage speeding based on speeds exceeding speed limit by 50%
0 - 5
5 - 10
10 - 20
20 - 40
0
3.75
7.5
15 Miles
N
Glendale
Arcadia
Azusa
La Verne
Claremont
Baldwin Park
Covina
West Covina
El Monte
Montclair
Pomona
Los Angeles
East Los Angeles
Chino
Chino Hills
Huntington Park
Whittier
South Gate
Lynwood
Norwalk
Compton
Yorba Linda
Cerritos
Buena Park
Fullerton
Torrance
Carson
Anaheim
Orange
Garden Grove
Long Beach
Westminster
Santa Ana
Tustin
Irvine

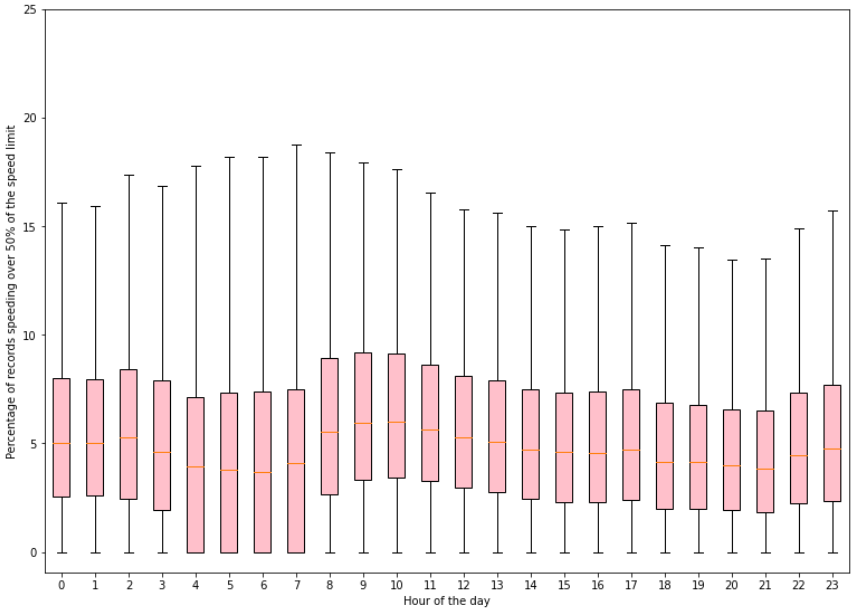
(b)

Figure 2. 8 Speeding on California freeways at 7PM, 02/01/2020 where (a) travel speeds were 20% over the speed limit and (b) 50% over the speed limit.

An analysis of speeding on freeways at different hours of the day showed that driving at more than 20% over the speed limit was more common than the case of traveling at over 50% over the speed limit on CA freeways. The proportion of waypoints where travel speeds exceeded 20% of the speed limit by hour occurred more during the nighttime with a drop in the early morning hours (fig. 2.9 a). Significantly fewer drivers traveled at speeds exceeding 50% of the limit for any hour during this particular day (fig. 2.9 b).



(a)



(b)

Figure 2. 9 Speeding on California freeways over time where (a) speeding exceeded 20% over the speed limit, and (b) exceeded 50% over the speed limit.

2.7 Conclusions and future work

Big GPS trajectory data brings new opportunities to large scale spatio-temporal traffic speed estimation, which is essential for researchers working on topics relating to urban mobility, ITS, and smart cities. In this paper, we discuss a computing framework that uses Apache Spark and Sedona to estimate traffic speeds for a state-level road network using passively collected GPS waypoint data. Taking advantage of RDDs supported by Spark and SRDDs by Sedona, the framework approach extracted traffic speed information from hundreds of millions of trajectory points for the state-level road network while maintaining a good balance between computing efficiency and accuracy.

Mobility and road network datasets of 3.4 million road segments and 126 million trajectory points collected in California were used in this research, generating hourly speed estimates for more than 596,000 road segments. The results showed that for the given day of data there were sufficient waypoints captured between 10am to 11pm to

make valid speed estimates for both freeways and residential roads. For freeways, we found more segments associated with faster speeds during nighttime hours than the daily average, and a pattern where faster freeway driving mostly occurred between 11 pm and 2 am and between 7 am and 10 am. This finding is in line with previous research (Y. Huang et al., 2018) that found that drivers were more likely to have a higher speeding rate during night hours than during the daytime using taxi data from New York and Shanghai. More vehicles on freeways were exceeding the speed limit by either 20% or 50% during the nighttime, although the majority of freeway segments had fewer drivers that were traveling at over 50% above the speed limit. This was in contrast to the pattern for residential roads that showed a relatively stable pattern of travel over the 24-hour period with more segments that were slower than average during the night. We validated our speed estimates results using freeway monitoring station data from CA PeMS, and found that 48% of speed estimates calculated using our framework had less than 10% difference with the monitoring station data.

While big mobility data holds much promise, it should be noted that there are limitations to using such data. One possible source of bias is that the users of any location-based App may not be uniformly sampled across all populations, which may impact the representativeness of the data for all vehicle drivers in the California (S. Hu et al., 2021). To help mitigate this type of bias, further study could employ additional data types such as stationary data or survey data that contains weighting information to further enhance traffic speed and speeding estimates. In addition, the mobile device data we used in this study was from only one day in California, and further testing is needed for a

broader temporal sample to validate how broadly the travel patterns we discovered apply to other days for the CA road network.

There remain several open questions for future research, for example, studying the variance of speeds on road segments has been found to have a more significant impact on traffic accident fatality rate than average traffic speed (Lave, 1985) and a future study could examine variance of speeds using passively collected mobility data. Another possible topic that requires further consideration is estimating traffic speed in more complex setting, for example, for a lane level road network to understand how traffic speeds vary across lanes on the same road segment (Ke et al., 2020). One additional topic is combining online map-matching and routing algorithms with a distributed computing framework and providing support for real-time traffic condition estimation for a broad spatial scale and large data volume.

Chapter 3: LSTM-autoencoder deep clustering of big GPS trajectory data for parking behaviour classification

3.1 Abstract

Certain types of driving behaviours (e.g., searching or cruising for a parking space) may contribute to a series of urban issues including congestion and air pollution among others. While some studies have analysed the impact of factors on cruising times, fewer studies have developed methods for detecting different parking-related behaviours in the trajectory datasets. In this research, using detailed driving behaviours captured by passively collected in-vehicle GPS trajectory data, we introduce a computing framework that extracts parking trajectories from driving trajectories and classifies them into cruising or direct parking categories. The framework consists of a preprocessing module, a rule-based distributed parking trajectory extraction module, and an LSTM-autoencoder deep clustering model. The preprocessing and parking trajectory extraction module is designed to extract trajectories that involve parking from raw driving trajectories. The deep clustering model is designed to take feature-augmented parking trajectories of varying lengths and without labels as input and returns two broad categories for cruising and direct parking. The results returned two types of cruising and two types of direct parking using in-vehicle trajectory data collected passively. The framework can be applied to trajectory data collected for any city and will be useful for identifying locations in an urban environment with cruising times that could be reduced to ease traffic flow and parking demand.

3.2 Introduction

As a common driving behaviour and an important part of most daily driving trips, parking has been a research focus for the contexts of better city management (Chai et al., 2019; Lu & Liao, 2020; Millard-Ball et al., 2014) and the development of Intelligent Transportation System (ITS) (Khanna & Anand, 2016). Certain driving behaviours associated with parking, for example, cruising while searching for a parking space or empty cruising by private autonomous vehicles, may contribute to a series of urban issues, including increased air pollution, additional congestion, driver frustration, and an excess of vehicle miles travelled (Bagha et al., 2023; Cao et al., 2019; Shoup, 2006; Weinberger et al., 2020). To mitigate the impact brought by frequent cruising, detecting these driving behaviours and understanding their spatio-temporal distribution is important.

In previous studies on parking behaviours, actively collected data types, e.g., video recordings and on-street surveys, have been used as data source (Axhausen & Polak, 1991; Chaniotakis & Pel, 2015; Weinberger et al., 2020). These studies mostly focused on a small spatial scale (e.g., several streets or parking garages) and a relatively small data volume due to the limitation of the data sources (Shoup, 2006). In the past few years however, with the development of geo-tracking technologies in mobile devices and in-vehicle sensors, passively collected trajectory datasets sourced from these devices bring new possibilities for both larger-scale and finer-grained human mobility analyses (C. Chen et al., 2016). Previous studies did not fully examine the different behaviours associated with the process of parking in a city including the different categories of parking-related behaviours that may exist (i.e., direct parking and cruising for parking).

How to identify and automatically classify driving trajectories that reflect different driver behaviours using big unlabelled GPS trajectory data is still an open research challenge, and one that is addressed in this study.

For this research, to identify and extract trajectories from big passively collected in-vehicle GPS trajectory data (refer to as trajectory data in later sections) for *post-hoc* analyses of parking and congestion, and to overcome the challenges brought by massive data volumes and larger spatial coverages (i.e., millions of trips and billions of waypoints that make many traditional geospatial data analysis methods difficult to apply and time consuming), we propose a rule-based distributed computing framework to extract *parking trajectories* (i.e., the ending segments of trajectories involved in either searching for parking or actual parking) from massive passively collected trajectory data based on Apache Spark and Sedona (Yu et al., 2015; Zaharia et al., 2010; Zaharia et al., 2016). The framework first generates a set of buffers around the destination point of a driving trip using different radii, and then applies a set of rules that use travel speed to the waypoints inside the buffer area and in this way, parking trajectories are extracted. Based on the spatial data types and spatial partitioning supported by Sedona's Spatial Resilient Distributed Dataset (SRDDs), the framework is suitable for extracting parking behaviours from overall patterns of driving at city-wide or regional spatio-temporal scales and offers faster computing speeds over the baseline model.

To classify the parking trajectories further based on driver behaviours and overcome the challenge of an absence of labels for the categories, we designed an LSTM autoencoder-decoder deep clustering model. The deep clustering model uses driving features including driving speed, angles, parking distance, time duration, and locations in

the format of longitude and latitude along with extracted parking trajectories as input and classifies parking trajectories into two main categories of either cruising or direct parking and within these categories, further subcategories are identified. Taking advantage of the LSTM layers, the model takes driving features along with varying lengths of driving trajectories as input and needs no interpolation. In this way, classification happens in the absence of any labeling of categories of trajectories. Cruising generally refers to trajectories that showed a relatively long temporal duration for parking (e.g., searching for more than 5 minutes) and longer spatial ranges, and were likely to have frequent turns during the search for parking. Direct parking, on the other hand, refers to trajectories where drivers stopped at a destination point in a relatively short temporal duration (often 1-2 minutes) and had shorter spatial ranges without excessive circling. The categories from the deep clustering model showed clearly better classification results than the baseline models using manual labeled parking trips as verification data.

The distributed parking extraction framework and the proposed deep clustering framework was tested on a passively collected trajectory dataset with approximately 6.3 million trips and 2.8 billion waypoints collected during July 2018 in the Washington DC metropolitan area (fig. 3.1). Applying the LSTM autoencoder deep clustering model on the extracted parking trajectories, we retrieved four different parking categories in total, including two cruising and two direct parking categories. Detected trajectories in different categories showed different driving behaviours, distinguished by different temporal durations and the number of turns. Based on the classification results, identification of where cruising behaviours are concentrated may represent a potential

shortage in parking resources compared to demand in an area or provide useful information about available parking for drivers (Gu et al., 2020).

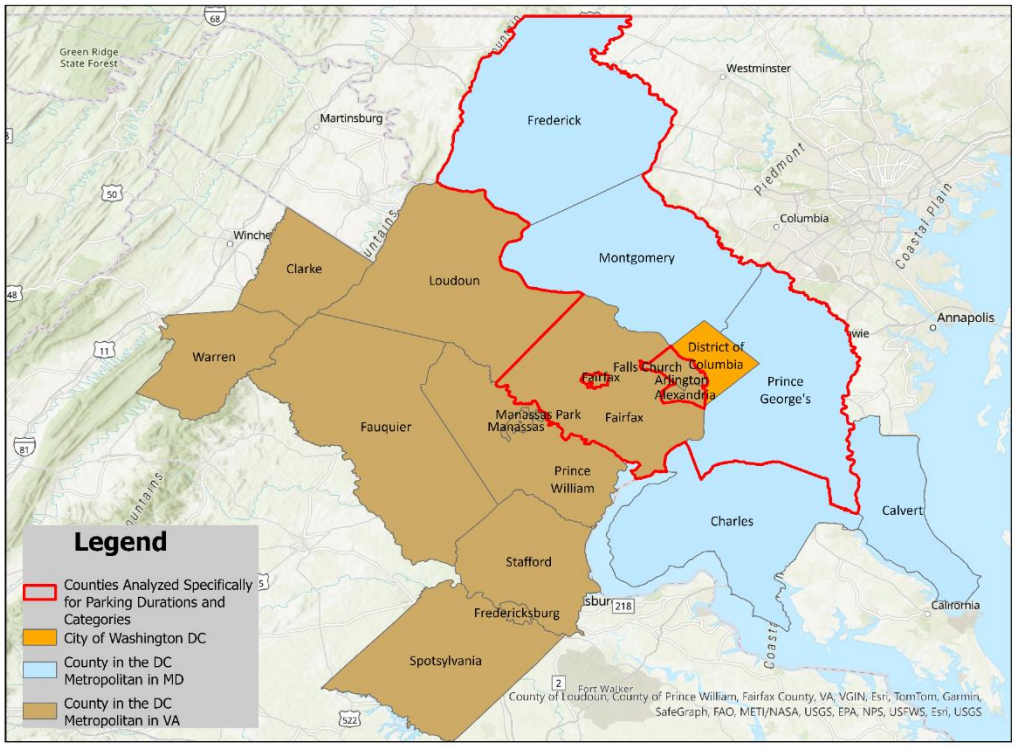


Figure 3. 1 The City of Washington, DC (shown in orange) and the surrounding Washington DC metropolitan area (shown in brown for counties in VA and blue for counties in MD), counites and cities specifically analysed for parking duration and categories were bounded.

The rest of the paper is organized as follows: Section 2 discusses related work on parking trajectory analysis; Section 3 introduces the study area and datasets used in this research; Section 4 describes the design of the distributed computing framework used for parking trajectory extraction, the LSTM autoencoder-decoder for deep clustering, and the baseline models; Section 5 discusses the parking trajectory extraction process and presents classification results for the study area; and Section 6 presents the conclusions, limitation of this research and ideas for future research.

3.3 Related work

In previous studies on parking behaviours, actively collected data including video recordings of vehicles parking and on-street parking surveys have been a major data source (Shoup, 2006). For example, using a parking survey conducted in Brisbane, Australia, researchers reported an average cruising time of 14.5 minutes, and the perception of high costs for parking instead of the cost itself was one of the leading factors that impacted drivers' choices related to cruising for on-street parking (Lee et al., 2017). In another study, questionnaire data from the Dutch National Travel Survey for 2005 to 2007, was analysed by researchers who found that cruising times were in line with the benefits of cruising (e.g., lower parking prices) (Van Ommeren et al., 2012). It is also notable that the duration of cruising in the Netherlands study was very short, with an average of 37 seconds, which is different to cruising times found in studies based elsewhere in the world. For example, van der Waerden et al., (2015) undertook an exploratory study regarding the duration of searching for parking using GPS logs to analyse parking in an urban area, analyzing 97 GPS trajectories from the City of Turnhout, Belgium. According to their results, the average time spent searching for a parking space was about 1 minute and 18 seconds. Weinberger et al., (2020) provided a quantitative definition for cruising for parking based on analysis of a GPS dataset collected in Ann Arbor, MI and San Francisco, CA by the University of Michigan Transportation Research Institute, and a commercial anonymous GPS dataset. According to their analysis of parking in these two cities, cruising for parking had at least a 200 meter longer driving path than the ideal shortest path to the destination point, when a vehicle first entered a 400-meter buffer area based on the destination point. We explore

these distance relationships for parking more fully in this study and investigate different categories of driving behaviours associated with parking vehicles that has not been addressed in previous studies.

Clustering as one fundamental category of machine learning, has been commonly adopted in unsupervised trajectory classification tasks (Y. Zheng, 2015). Classic clustering models (e.g., K-Means (Hartigan & Wong, 1979) and DBSCAN (Ester et al., 1996)) have been broadly used in early days' research of grouping trajectories into different groups. Among these studies, different trajectory distance or similarity metrics have been applied, such as warping-based distance (e.g., Dynamic Time Warping (DTW) (Yi et al., 1998), Longest Common SubSequence (LCSS) (Vlachos et al., 2002)), and shape-based distance (e.g., Hausdorff) (Besse et al., 2016). However, because of the nature of the high dimensionality of trajectories and properties including a variety of trajectory lengths and different sampling intervals, the distance measurement metrics could fail, and shallow clustering methods may lose their discriminative power (Yue et al., 2019).

With the success of the Recurrent Neural Network (RNN) (Hopfield, 1982) based model on time series data, researchers developed deep representation learning for trajectory clustering. Compared with shallow (traditional) clustering methods, deep clustering methods provide better support for high-dimensional data classification, which makes them especially suited for trajectory datasets (Yue et al., 2019). Li et al., (2018) first developed a sequence-to-sequence model (t2vec) for trajectory similarity computing using RNN autoencoder. The researchers adopted a random dropping out (a drop-out rate from 0 to 0.6 in their experiments) as part of the deep representation learning to

overcome the challenges brought by varying sampling rates and low sampling rate in the trajectory data. Extended the t2vec, Fang et al., (2021) applied the K-Means clustering on the representation information extracted by the t2vec from raw trajectory data and formed an RNN deep trajectory clustering model (E^2DTC). Yue et al., (2019) proposed a Deep Embedded Trajectory Clustering network for travel purpose classification (DETECT). After summarizing trajectories using stop points along the trips, the researchers adopted an LSTM autoencoder structure to encode the geo-augmented trajectories with varying lengths and then performed unsupervised classification. Inspired by natural language processing, (C. Wang et al., 2024) treated spatio-temporal trajectories as tokens using grid-based or density-based system to represent the spatio-temporal information and applied a pretrained LSTM autoencoder proposed in Park et al., (2018) to extract the latent representations. The aforementioned research established a solid foundation for RNN-based autoencoder models in trajectory deep representation learning. In our study, we investigate the use of an LSTM autoencoder for extracting latent representation from geo-augmented trajectories in parking scenarios. Compared with the research introduced above which mainly focused on the similarity and classification of entire trajectories, parking mainly happened on a much smaller spatial scale. Therefore, the framework proposed in this research focuses more on extracting detailed travel behaviours (e.g., frequent turns and speed change) from parking trajectories and classifying these trajectories into different parking categories.

3.4 Data and study area

The trajectory data used in this research was collected passively through in-vehicle GPS sensors by INRIX and was collected in the Washington, DC metropolitan

area for one month in July 2018. The Washington, DC metropolitan area includes the city of Washington, D.C., and parts of Maryland, Virginia, and West Virginia (fig.3.1).

The dataset used in this research was provided by the Maryland Transportation Institute (MTI) at the University of Maryland and was comprised of trajectories that represent vehicle trips. The trips were predefined by the data vendor, where the ending point of the trip is known but not the actual destination of the trip. This is not uncommon among research that uses passively collected trajectory data for parking studies (Millard-Ball et al., 2020; Weinberger et al., 2020). Trips that were not likely to contain parking were excluded from the preprocessing model.

Each trajectory in the dataset is comprised of a set of GPS waypoints. Each waypoint contains attributes including longitude, latitude, device ID, trip ID, timestamp, raw speed, the time difference from the previous point of the same trip, and distance from the previous point of the same trip.

The raw dataset contains around 6.3 million trips and over 2.8 billion waypoints and has an average of approximately 400 waypoints per trip. The average duration of trips in the month of data collection was approximately 34 minutes. The sampling interval in this dataset was not uniform and range from 1 second to over 5 minutes. Overall, the dataset has a short sampling frequency where 49% of waypoints had a sampling frequency less than 10 seconds, 8% had a sampling frequency between 10 - 30 seconds, and 16% were between 30 seconds and 1 minute. The remaining 23% of waypoints had frequencies between 1 minute and 5 minutes long, and less than 2% of waypoints in our dataset had more than a 5-minute sampling frequency (fig. 3.2). Shorter sampling frequencies help to better capture the travel behaviours within trips.

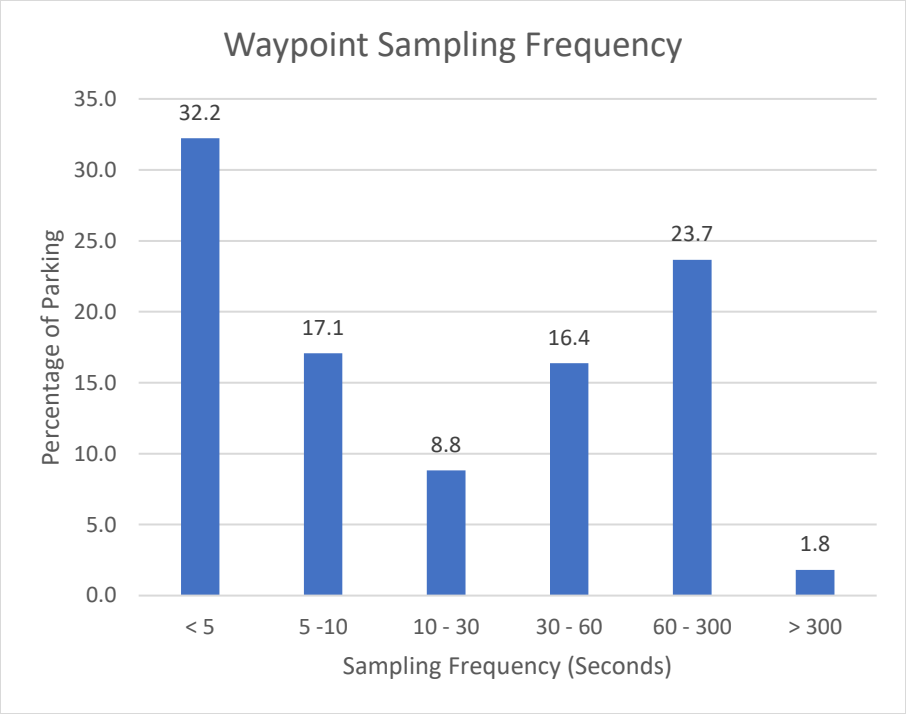


Figure 3. 2 Sampling frequency of waypoints

The road network dataset for the DC metropolitan area used in this research is from Open Street Map (OSM). The road network dataset contains 22,377 road segments for the city of Washington, DC and around 1.2 million road segments for the Washington DC metropolitan area.

3.5 Methods

To identify and classify parking trajectories from massive passively collected trajectory data and retain travel behaviours that occurred over different spatio-temporal ranges for analysis, we propose a three-step model for parking trajectory identification and parking behaviour classification. For the first step, we designed a data preprocessing and filtering model to extract trajectories that were more likely to contain a parking process. We built a rule-based model to extract parking trajectories according to a set of rules that use driving speeds associated with a trajectory within a certain distance of the

destination point. A distributed computing environment including Apache Spark and Sedona was used in the rule-based model design and application to address the challenges brought by the large data volume and broad spatio-temporal coverage of the trajectory data. For the last step, we developed an LSTM autoencoder deep embedded clustering model to classify identified parking trajectories based on their driving features. These steps will be discussed in detail in the sections that follow.

3.5.1 Preprocessing and trip filtering

The preprocessing module was designed to exclude anomalous waypoints and trajectories (trajectories that unlikely to capture a parking process), and then generate a subset of trajectories in the study area that were analyzed with respect to a parking range and classified as different categories of parking behaviours. The filtering process was designed to extract trajectories with higher sampling frequency that were likely to capture more features for parking behaviour classification.

In this study, anomalous waypoints refer to stop points (points that have speed equal to 0 mph) at the end of trajectories and individual noise waypoints (waypoints "floating" away from the trajectory). The stop points at the end of trajectories could be caused by location-aware sensors that keep tracking after the parking process has ended. We excluded all stop points at the end of each trajectory in the preprocessing model. Waypoint data noises could be caused by poor positioning signals or sensor noise. The trajectory dataset used in this research showed very few waypoints-data noise (less than 1 waypoint per trajectory according to our manual examination of the map), we believe it is because the dataset was collected by in-vehicle sensors instead of commonly used mobile devices. We had a threshold based on speed (calculated by distance from the previous

and after point and difference in timestamps instead of speed provided by the datasets as an attribute) to exclude waypoint noises.

An anomalous trajectory refers to a trajectory that captures where a location collecting service terminated but does not capture that a trip stopped at the actual destination of the traveler, and so is not considered part of parking process. These trajectories could occur for multiple reasons, such as the location tracking service exiting before the trip ended and is not an uncommon issue with passively collected trajectory data. In the preprocessing model, we applied a rule-based threshold based on travel speed to exclude these trajectories using the understanding that the acceleration rate of the vehicle would be negative (i.e., the vehicle is slowing) since the driver started parking (Stewart et al., 2018).

In the preprocessing model, we first detected and excluded all trips where the end of the trajectories included waypoints during the last 30 seconds of each trip that were still at an average speed above a threshold of 15 mph determined to indicate deceleration and parking. Preprocessing also involved extracting trajectories that corresponded to the final time when vehicles entered a 1500-meter area around the destination waypoint (i.e., final stop point). We refer to this time period as the ending phase of a trip.

We then conducted trip-level filtering on the ending phase trip trajectories to extract trajectories that were more likely to contain enough features for the parking behaviour classification. We applied a sampling interval threshold of 60 seconds and the number of waypoints-thresholds of 3 points to the ending phase of trajectories and kept trajectories only if the sampling interval of all waypoints was within this threshold. In

addition, the dataset contains trips that only passed through the study area. We also extracted only the trips that ended in the study area in the preprocessing.

3.5.2 Distributed parking trajectory extraction

The process of *post-hoc* searching for parking is commonly understood to start at or near the destination of a trip, where vehicles have maintained relatively low speeds. Two methods used in previous studies (Stewart et al., 2018; Weinberger et al., 2020) to extract parking trajectories from GPS trajectory data are discussed here:

1. Extract driving trajectories once a vehicle is within a certain radius of the destination.
2. Apply one or more thresholds of travel speed or acceleration rates to the end of a driving trajectory.

We used a combination of these two approaches to extract parking trajectories from different trips. Using Apache Spark and Sedona, we developed a rule-based model to first extract the final driving portion of driving trajectories using a set of buffers with different radii, and then determine the most appropriate buffer distance using a set of rules based on vehicle speed and apply these steps for different parking scenarios (fig. 3.3).

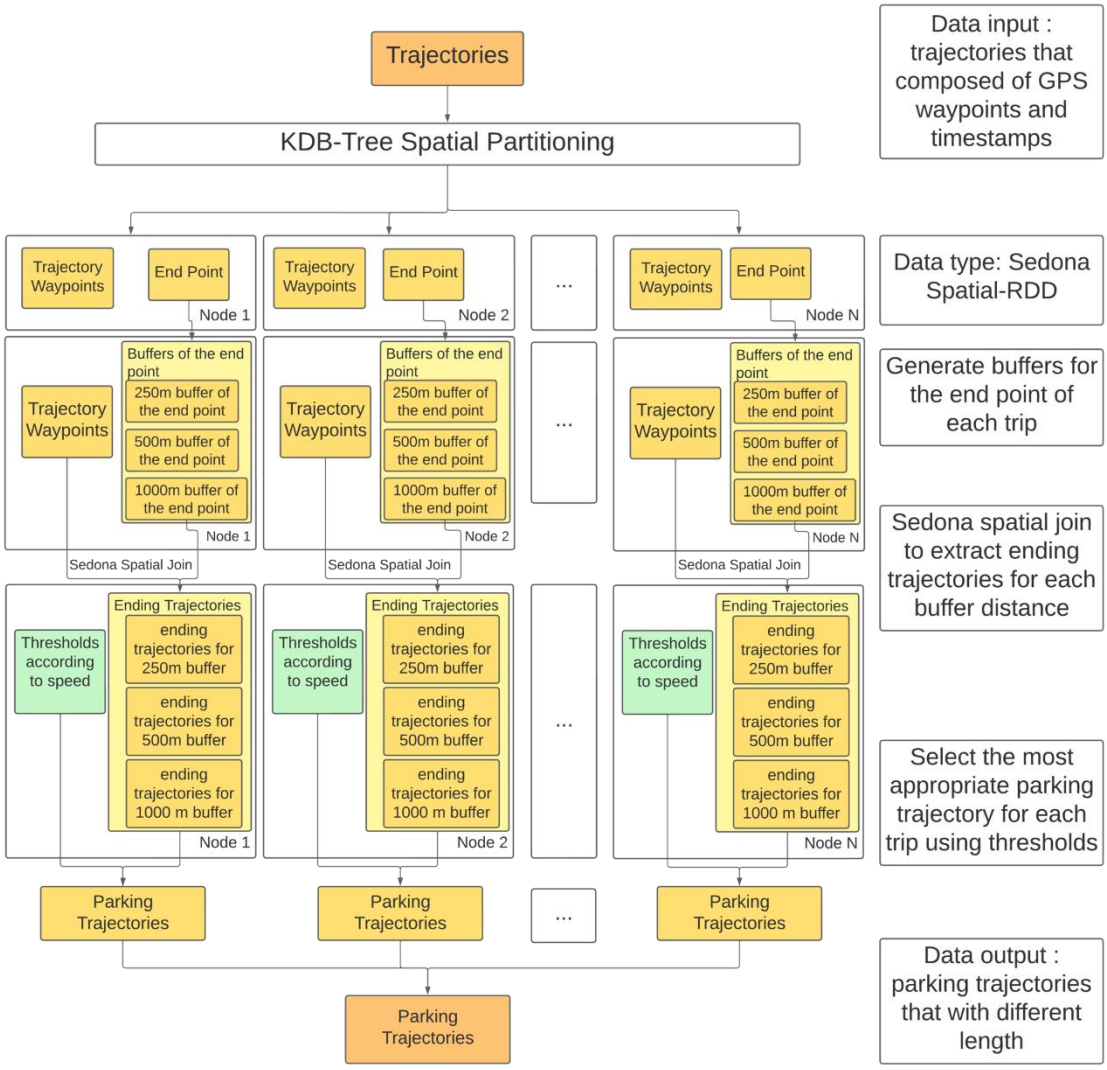


Figure 3. 3 Structure of the distributed parking trajectory extraction model

A previous study used a buffer with a radius of 400 meters from the final stay point to extract parking trajectories in Ann Arbor, MI and San Francisco, CA (Weinberger et al., 2020). For this research, we assumed that searching for parking would be impacted by the block size and network density of a city. The City of Washington, DC does not follow a strict grid pattern despite an original block size design of 400 feet (122 meters). We found that the length of most blocks in Washington, DC varied between 300 to 500 feet (91 to 152 meters), and based on this, we used 250 meters as an approximation of a block's diameter and designed the radii used for parking trajectory

extraction using this measurement. We also found that a single buffer with a fixed radius (as used in earlier studies) would be inappropriate for parking scenarios with different spatial ranges (i.e., the distance traveled for parking and ending the trip, fig. 3.4). In this study, we investigated a set of three buffer distances (250 meters for shorter distances, 500 meters for medium distances, and 1000 meters for longer distances) for parking trajectory extraction.

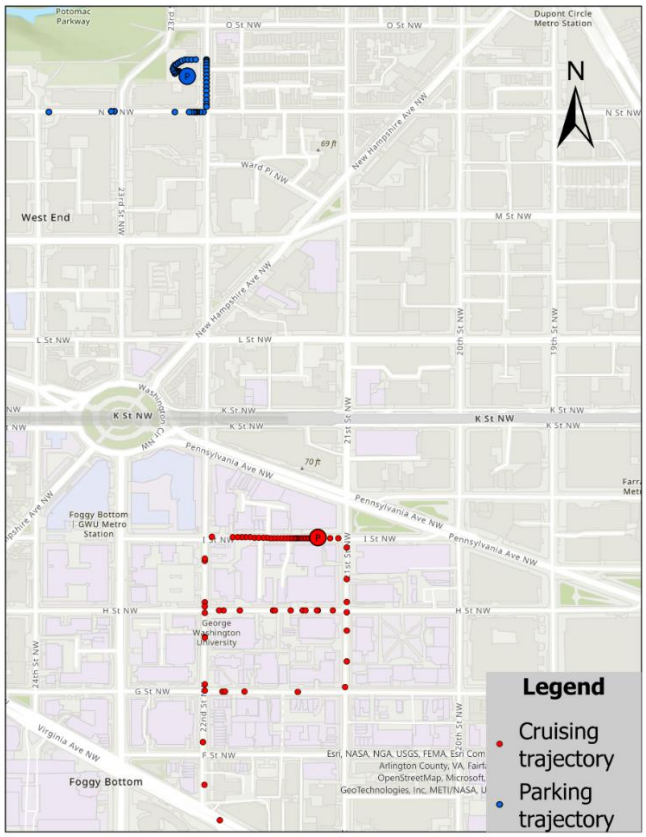


Figure 3. 4 Parking and cruising trajectories in Washington, DC with two different temporal and spatial ranges

For scenarios where a vehicle slowed down within the shorter or medium distance buffers (i.e., the vehicle was quite close to its destination point), we extracted trajectories using a speed threshold based on both the average and maximum travel speed. For scenarios where vehicles entered the longer distance buffer and maintained driving at a

relatively slow speed (less than 30 mph), and didn't show any significant acceleration beyond the applied speed threshold, we classified these sub-trajectories as parking trajectories.

To overcome the challenges brought by large data volumes and broad spatio-temporal scales (i.e., over 2.8 billion waypoints and 1.2 million road segments), we used Apache Spark and Sedona for the model design and development. We transformed the datasets to the format of Sedona Spatial Resilient Distributed Dataset (SRDDs) with the mobility data as Point RDD. To extract parking trajectory waypoints using the buffer distances, we applied spatial joins between waypoints and buffer distances using Sedona SQL and used the Python API for Sedona to generate the short, medium, and long-distance buffers around destination waypoints, and extracted ending trajectories with different buffers (Fig. 4). To further accelerate the model, KDB-Tree spatial partitioning was applied as part of the spatial join process (Robinson, 1981).

Taking advantage of Spark and Sedona, the computing time for the extraction section took around 3 minutes on our datasets and was approximately 7 times faster than scripts using the Geo-Pandas library based on Python. We believe the advantage of improved computing speeds and support for big geospatial data brought by Spark and Sedona will become more significant as the data volume increases.

After ending trajectories were extracted using buffers, we generated parking trajectories by selecting the most appropriate buffer radius for different parking scenario using a set of rules based on driving speed (fig. 3.5).

Algorithm 1 Parking Trajectories Extraction

Require: all_trip_id, $Trajectory\_Short$, $Trajectory\_Medium$, $Trajectory\_Long$

```
for all trip_id_i in all_trip_id do
    if AVG(Trajectory_Medium[trip_id_i].Speed) ≥ 25 OR
      MAX(Trajectory_Medium[trip_id_i].Speed) ≥ 45 then
        return Trajectory_short[trip_id_i]
    else if AVG(Trajectory_Long[trip_id_i].Speed) ≥ 30 OR
      MAX(Trajectory_Long[trip_id_i].Speed) ≥ 50 then
        return Combine(Trajectory_Short[trip_id_i],
            Trajectory_Medium[trip_id_i])
    else
        return Combine(Trajectory_Short[trip_id_i],
            Trajectory_Medium[trip_id_i],
            Trajectory_Long[trip_id_i])
    end if
end for
```

Figure 3. 5 Pseudo code for parking trajectory extraction algorithm

In algorithm 1, *all_trip_id* represents the set of all trip IDs in the dataset, and *trip_id_i* represents each of the trip IDs. *Trajectory_Short* (represents waypoints within the short buffer area (red dots) in fig. 3.6), *Trajectory_Medium* (representing waypoints within the medium buffer (green dots)), *and Trajectory_ Long* (representing waypoints within the longer distance buffer (blue dots)) represents sets of trajectories extracted using buffers with the short, medium, and longer radii respectively (Fig. 6). For each of the trajectories extracted using buffers with medium and long radii, trajectories that also intersected short and medium buffers were excluded, which means that the three trajectories for the same trip ID have no intersection. *Trajectory_Short['trip_id_i'], Trajectory_Medium['trip_id_i'],* and *Trajectory_Long['trip_id_i'* represents each of the ending trajectories for a specific trip where trip ID equals to *trip_id_i. AVG(*.Speed)* that represents the average speed of waypoints in the trajectories for different buffers and *MAX(*.Speed)* represents the maximum speed of waypoints in the trajectories for a particular buffer. To facilitate the execution of the above algorithm in the distributed

system, we transferred this algorithm into the format of Spark-SQL (Armbrust et al., 2015).

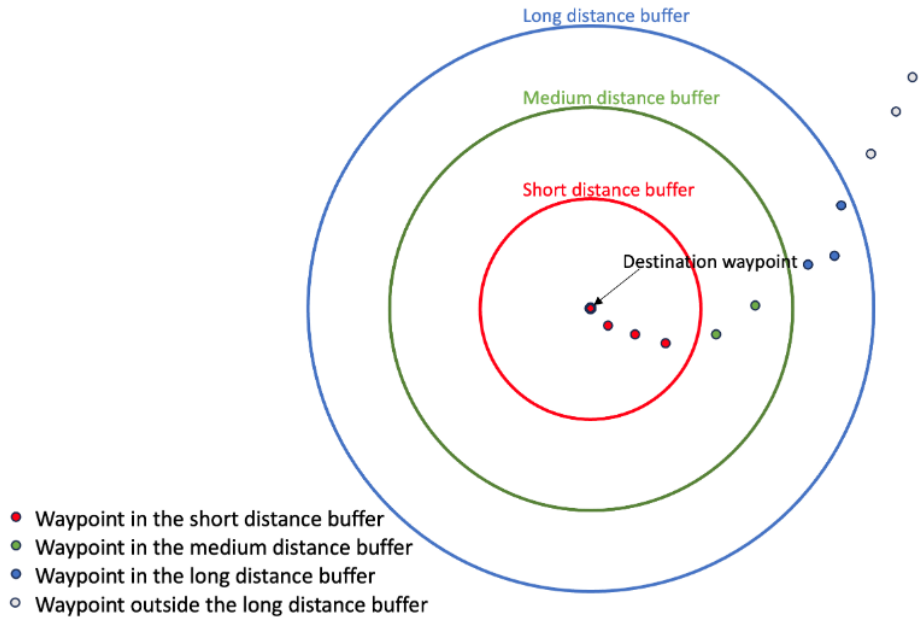


Figure 3. 6 Buffers with different radii used for parking trajectory extraction

3.5.3 Feature augmentation

We classified driving trajectories into two different behaviours, i.e., cruising and direct parking based on the features of trajectories and reflecting mobility behaviours during the parking process. For example, a trajectory representing cruising for parking generally showed a pattern of maintaining a lower speed over a relatively longer time duration and may also have had frequent turns corresponding to, e.g., a vehicle circling the block while searching for a space. However, raw trajectory data contains only a series of chronologically ordered waypoints and does not have any explicit features such as travel speed and turning angles from previous waypoints (Dabiri & Heaslip, 2018; Zhu et al., 2022). For this reason, we used a feature augmentation process to enrich each waypoint in the raw trajectory data with 9 features including driving features as well as global features that were applied to the entire parking trajectory.

A parking trajectory with m waypoints extracted using the parking trajectory extraction model is represented as $G = \{g_1, g_2, \dots, g_m\}$, where each g_i is a GPS waypoint that contains longitude, latitude, and a timestamp.

$$g_i = [longitude_i, latitude_i, timestamp_i]$$

For each waypoint, local driving features included driving speed, current driving angle, angle to the destination point, and distance to the closest road segment. The current driving speed was collected by the in-vehicle GPS sensors and was provided with the trajectory data. The driving angle of each waypoint g_i was calculated based on the coordinates and timestamp of waypoint g_i and the next point in the same parking trajectory g_{i+1}:

$$C\_Ang(g_i, g_{i+1}) = \frac{\Delta(latitude_i, latitude_{i+1})}{\Delta(longitude_i, longitude_{i+1})}$$

Where $\Delta(\cdot)$ refers to the difference method, and the angle to the destination point was calculated as the angle between the current waypoint g_i and the destination point of the parking trajectory g_m.

$$D\_Ang(g_i, g_{i\mathrm{m}}) = \frac{\Delta(latitude_i, latitude_m)}{\Delta(longitude_i, longitude_m)}$$

For the last waypoint in each trip, the direction of that point was set to 0.

To retrieve the distance from a waypoint to the road segment that it was more likely driving on, we used an Apache Sedona-based distributed map matching module that could detect the most appropriate road segment according to the distance from the waypoint to the road segments and the direction of driving (P. Zhang et al., 2023).

In addition to the local driving features for each waypoint, we used the duration of the entire parking process as a global feature, including the total distance the vehicle traveled during parking, and the Euclidian distance between the first point of the trajectory and the destination point. To use these global features with the other local driving features in the time series trajectory, we repeat these features for every waypoint in each trip.

3.5.4 LSTM Autoencoder

We designed an LSTM autoencoder to compress and encode the driving behaviours of the feature-augmented trajectories into a low-dimensional latent embedding, and then reconstructed the trajectories from the embedding layer. Similar to a conventional autoencoder (Baldi, 2012), the LSTM autoencoder contains both an encoder and a decoder (fig. 3.7).

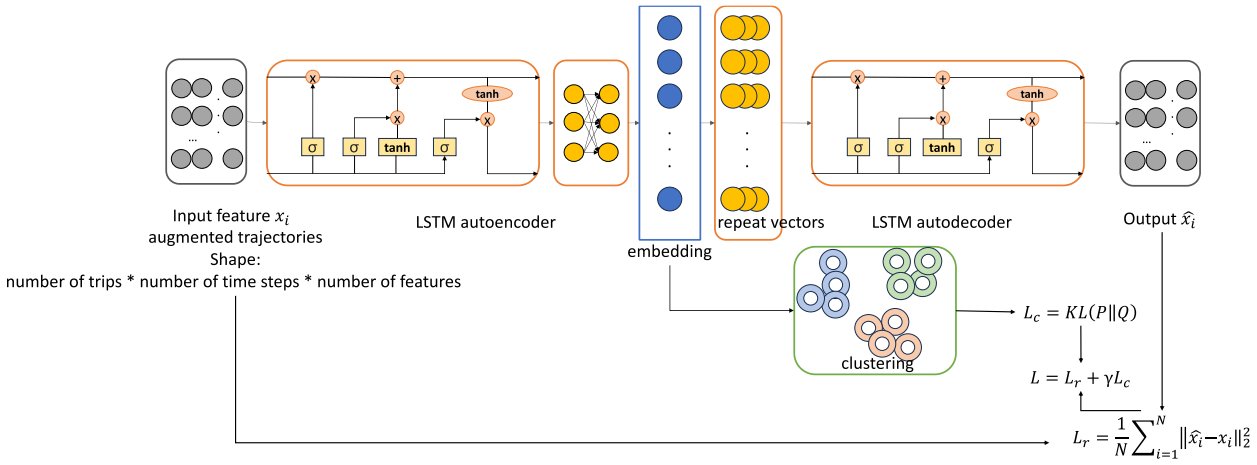


Figure 3. 7 The structure of the LSTM autoencoder deep clustering model

Where $x_i = \{x_i\}_n^{T_i}$ is a feature-augmented trajectory with T_i time-steps and n features. During the encoding process, x_i is fed sequentially to several LSTM layers as

well as a fully connected layer to generate an embedded layer *h* that is assumed to contain all the information for the feature-augmented trajectory. The process of encoding is represented as:

$$h = f_{encoding}(x_i)$$

Then the LSTM decoder reconstructs x_i such that:

$$\widehat{x_i} = f_{decoding}(h)$$

$$= f_{decoding}\big(f_{encoding}(x_i)\big)$$

The LSTM autoencoder and decoder were trained together by minimizing the mean square error (MSR) of the reconstruction error L_r:

$$L_r = \frac{1}{N}\sum_{i=1}^{N} \|\widehat{x_i} - x_i\|_2^2$$

$$= \frac{1}{N}\sum_{i=1}^{N} \big\|f_{decoding}\big(f_{encoding}(x_i)\big) - x_i\|_2^2$$

where N is the total number of parking trajectories in the dataset. During the training process, the embedded layer learns the characteristics of the features of the augmented trajectories.

3.5.5 LSTM Deep Embedded Clustering

The deep clustering model used in this research is composed of the LSTM autoencoder decoder and a clustering layer that is connected to the embedded layer. Since applying a shallow clustering method directly to the embedded layer that contains the compressed trajectory features may not generate the most appropriate clusters for the parking behaviour classification task, we followed (Guo et al., 2018; Xie et al., 2016) for

deep embedded clustering and used an optimization process to optimize both the clusters and the deep learning parameters.

For K classes in total and using cluster centers (i.e., the arithmetic mean value of all the points belonging to the cluster) $\{\mu_j\}_1^K$, the cluster layer maps each embedded point z_j in the embedded layer h into a soft label q_i using a Student-t distribution (Student, n.d.) to measure similarity between cluster centers and each embedded point:

$$q_{ij} = \frac{(1 + \|z_j - \mu_j\|^2)^{-1}}{\sum_j (1 + \|z_j - \mu_j\|^2)^{-1}}$$

Where q_{ij} is the *jth* entry of the soft label q_i and represents the probability of mapping an embedded point z_j that contains the compressed driving information for a parking trajectory, to the *jth* cluster.

To measure the difference between the distribution of soft labels that are generated and the predefined target distribution *P*, we applied a Kullback-Leibler divergence (KL divergence) measure (Kullback & Leibler, 1951) and we defined a clustering loss function as follows:

$$L_c = KL(P\|Q)$$

$$= \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}}$$

Where *P* is an auxiliary target cluster distribution calculated from high confidence prediction of *Q*:

$$p_{ij} = \frac{{q_{ij}}^2 / \sum_i q_{ij}}{\sum_j ({q_{ij}}^2 / \sum_i q_{ij})}$$

The objective function that determines the total loss during the training process for the entire model is defined as

$$L = L_r + \gamma L_c$$

Where L_r and L_c are reconstruction loss variables for the LSTM autoencoder and the clustering loss respectively, and γ is a coefficient that controls the proportion of clustering loss in the overall objective function. The autoencoder-decoder structure and clustering could therefore be optimized together by minimizing L during the iterative model training.

For the LSTM autoencoder, we applied 2 bidirectional LSTM layers (Graves et al., 2005; Hochreiter & Schmidhuber, 1997) with 256 and 64 units respectively and a fully connected layer with 64 units. The LSTM layers in the encoders and decoders were activated using the Rectified Linear Unit (ReLU) function (Agarap, 2018). After pre-training the LSTM autoencoder decoder by setting $\gamma = 0$ for 50 epochs with an Adam optimizer (Kingma & Ba, 2014), we performed K-Means clustering (Hartigan & Wong, 1979) on the embedded layer to retrieve the initial cluster centers. Then the LSTM autoencoder and the cluster layer were jointly trained by setting $\gamma = 0.1$. For the number of parking classes or categories K in the cluster layer returned by our deep learning approach, we tested different numbers from 7 for the most appropriate parking categories.

3.5.6 Baseline models

To evaluate the advantage of the deep clustering model for parking trajectories classification we proposed in this research, we introduced 6 classic baseline models and 2 deep learning baselines that were commonly used in previous works(Fang et al., 2021; Yue et al., 2019). The classic baselines we selected are classic clustering models (e.g., K-Means and DBSCAN) with distance metrics specifically designed for time series data, including:

DBA-Kmeans (Petitjean et al., 2011, 2014): DBA stands for Dynamic Time Warping Barycenter Averaging. This model uses DTW distance (i.e., a similarity measurement designed for time series) instead of the commonly used Euclidean distance as the distance metric for the K-Means clustering. We set number of the clusters (K) as 4 in and used parking trajectory extracted after extraction (with only longitude and latitude as features) as input.

DBA-Kmeans*: In order to separately analyse the advantage brings by the feature augmentation we designed for parking classification, we also applied the DBA-Kmeans on the feature augmented parking trajectories.

LCSS-DBSCAN (Morris & Trivedi, 2009): LCSS stands for Longest Common SubSequence (Vlachos et al., 2002), which is a similarity measurement designed for trajectories by giving more weight to the similar portions of the sequences. We tested different combinations of the DBSCAN parameters (Epsilon and minimum points (minPts)) in comparison experiments on parking trajectories with only longitude and latitude.

LCSS-DBSCAN*: For the LCSS-DBSCAN, we also tested it on the parking feature augmented trajectories.

LCSS-Kmeans (Choong et al., 2017): Similar to the LCSS-DBSCAN, we applied the LCSS for the similarity measurement of the K-Means clustering for the parking trajectories with only longitude and latitude.

LCSS- Kmeans *: For the LCSS-Kmeans, we also tested it on the parking feature augmented trajectories.

For the deep learning baselines, we followed previous work and introduced two RNN-based autoencoder based deep representation learning and applied K-means clustering directly on the latent embedding (Fang et al., 2021; Yue et al., 2019), including:

LSTM autoencoder + Kmeans (X. Li et al., 2018b; Yao et al., 2017): We examined the simple LSTM autoencoder on the parking trajectories without the joint optimization process for the latent embedding compression and directly applied K-Means on the latent embedding.

LSTM autoencoder + Kmeans*: In this set of experiment, we also applied the LSTM autoencoder on the parking feature augmented parking trajectories. The only difference between the deep clustering model we proposed in this research is the jointly optimization process.

3.6 Results

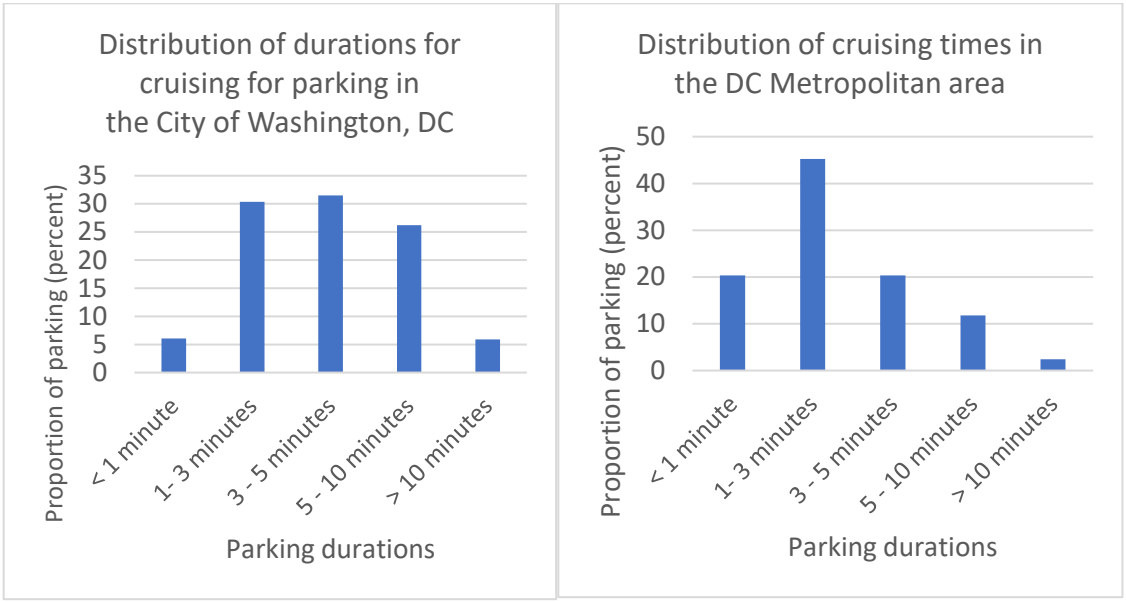
Before the data preprocessing module, we first excluded all trips that passed the Washington, DC metropolitan area but didn't end there (around 1.8 million trips left after

this process). After the data preprocessing of excluding anomalous points and trajectories (trajectories that didn't stop at the end), and filtering on the ending phase trip trajectories extracted approximately 170,000 trajectories in total. Of these trajectories, 43% ended in Montgomery County, MD, 22% ended in the City of Washington DC, and 15% ended in Prince George's County, with the remaining trajectories ending in other counties in the DC Metropolitan area.

After parking trajectory extraction using the distributed parking trajectory extraction model, the distribution of time-to-park (i.e., cruising for parking) showed spatial heterogeneity over the study area. There was a higher proportion of longer durations for parking (i.e., cruising for parking that lasted longer than 5 minutes) in Washington DC, than in the DC metropolitan area in general. The average time-to-park in the City of Washington, DC was 4 minutes and 29 seconds, with almost 26% taking from 5 to 10 minutes (approximately four thousand trips), and 6% (approximately one thousand trips) taking longer than 10 minutes (fig. 3.8 a). In the greater Washington DC metropolitan area, parking times were shorter with an average cruising for parking duration of 2 minutes and 53 seconds, and with only 12% taking from 5 to 10 minutes (approximately six thousand trips), and less than 3% taking longer than 10 minutes to park (approximately one thousand trips) (fig. 3.8 b).

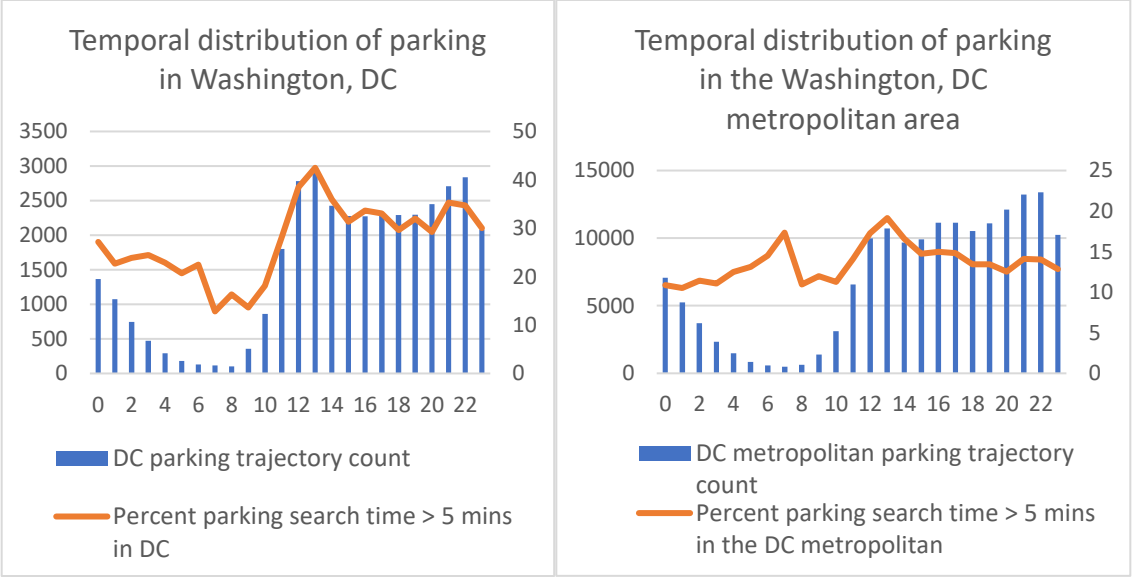
For the City of Washington DC, the peak number of parking trajectories were retrieved for the time period of 12pm to 1pm and 10pm to 11pm, while there was a continuous and stable percentage of around 6 to 7% of the total parking trajectories extracted for the period 2pm to 8pm. Longer durations for parking in the City of Washington DC were generally much higher than in the overall metropolitan area, with

the highest number of longest time-to-park durations occurring between 12pm to 1pm (fig. 3.9 a). For the Washington DC metropolitan area, compared with the city, the change of the number of parking trajectories and longer-duration parking over the 24 hours are generally gentler. For longer-duration parking, the highest peak also occurs at noon, but the peak at 7 am is more obvious. (fig. 3.9 b).



(a) *(b)*

Figure 3. 8 Distribution of cruising times for (a) the city of Washington, DC., and (b) the Washington, DC metropolitan area.



(a) *(b)*

Figure 3. 9 Temporal distribution of parking for (a) the city of Washington DC, and (b) Washington, DC metropolitan area.

3.6.1 Parking trajectory classification results

After the parking trajectory extraction, the trajectory dataset used for training showed an even higher sampling frequency with more than half of the waypoints having sampling interval as 1 second. Therefore, we believe the sampling interval unbalance is not significant in the datasets. After testing and comparing different number of categories (i.e., the *K* value) using the dataset collected in the Washington DC metropolitan area, four different categories representing different driving behaviours were identified using our deep learning clustering approach (Table 3.1) and confirmed based on manual inspection of the mapped trajectories. With relatively fewer parking categories (i.e., 2 or 3 categories), the deep clustering model clearly confused some of the parking categories, such as trips with a longer time to park but no obvious circling, and trips with a relatively short time to park. With a larger number of categories (i.e. 5-7 categories), the model

generated two or more categories that were difficult to distinguish by inspection on the map.

Table 3. 1 Information for different parking categories

class	Avg. parking duration (seconds)	Avg. parking distance (meters)	Percentage of trips (%)	Avg. number of waypoints
1	102.81	583.95	15.69	4.66
2	111.94	533.54	35.57	38.27
3	186.06	1039.44	25.94	14.11
4	298.68	1255.15	22.80	64.86

The first category corresponds to what we refer to as *direct parking* with an average time-to-park duration of less than 2 minutes and an average parking distance of 583 meters (Table 3. 1). The trajectories in this group were mostly comprised of parking trajectories with very few waypoints due to the relatively long sampling intervals between waypoints (trajectories in this category are shown in green in Fig. 10). Because of the relatively long sampling interval, there could be unrecorded driving behaviors (i.e., turns or short stops), but it is unlikely considering the short time-to-park duration.

The second category was also associated with shorter times-to-park with an average of slightly less than two minutes, but these trajectories typically contained many more waypoints per trajectory compared with the first category (trajectories in this category are shown in violet in Fig. 10). These parking trajectories showed a short distance and contained few circling and turnings during the parking process. This category fits many regular parking scenarios, and these trajectories also represent *direct parking*.

These two categories showed fairly similar time-to-park durations and spatial distances. The major difference between the two categories was the number of waypoints

in each parking trip caused by different sampling intervals. In this research, we did not consider the sampling interval as a travel feature but a general data quality issue, therefore we consider both categories as scenarios of *direct parking*.

The third category returned by the deep clustering model was our first *cruising* category and was associated with cruising or searching times with an average time of approximately 3 minutes (trajectories of this category are shown in blue in Fig. 10). Although the searching for parking time was not generally much longer than the first two categories, this category showed much longer parking distances compared with the direct parking categories (i.e., approximately two times the first two categories on average). Parking trajectories in this category generally showed a process of searching considering the relatively longer time-to-park durations from the travel behaviour perspective but did not show a pattern of circling or frequent turning.

The fourth category included trajectories with longer cruising times that averaged approximately 5 minutes or longer and longer parking distances than all other categories, i.e., approximately 0.8 miles in average for spatial ranges (trajectories in this category are shown in red in Fig. 10). Unlike the third category, these trajectories were associated with different mobility features and contained more turns and circling, especially for locations that were near the destination point. The circling near the destination point generally involved two scenarios, namely circling around city blocks, and circling in parking lots or parking garages. This category represented behaviours associated with the longest times for cruising.

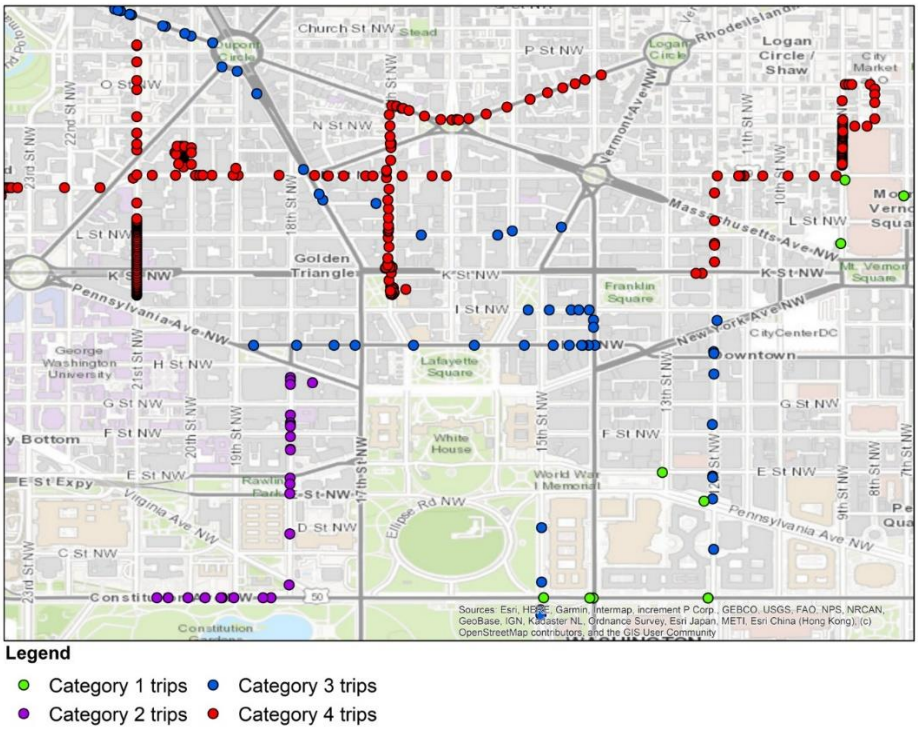


Figure 3. 10 Four categories of driving behaviours were associated with parking where each colour represents a different category of parking behaviour. Categories 1 and 2 represent 2 types of direct parking and Categories 3 and 4 represent two types of cruising for parking.

In general, there were more trajectories representing cruising or searching behaviours (i.e., categories 3 and 4) in the city of Washington DC (fig. 3. 11 a). We find this result to be in line with our analyses where there were numerous trips where the parking process took longer than 5 minutes. Inside the city of Washington DC, the parking trips ended in the commercial and entertainment areas showed a relatively long time to park duration (an average of more than 5 minutes) and generally more cruising for parking behaviors (more than half of the parking trips were classified as cruising). The major political areas in Washington DC such as Capitol Hill did not show much of the cruising trips.

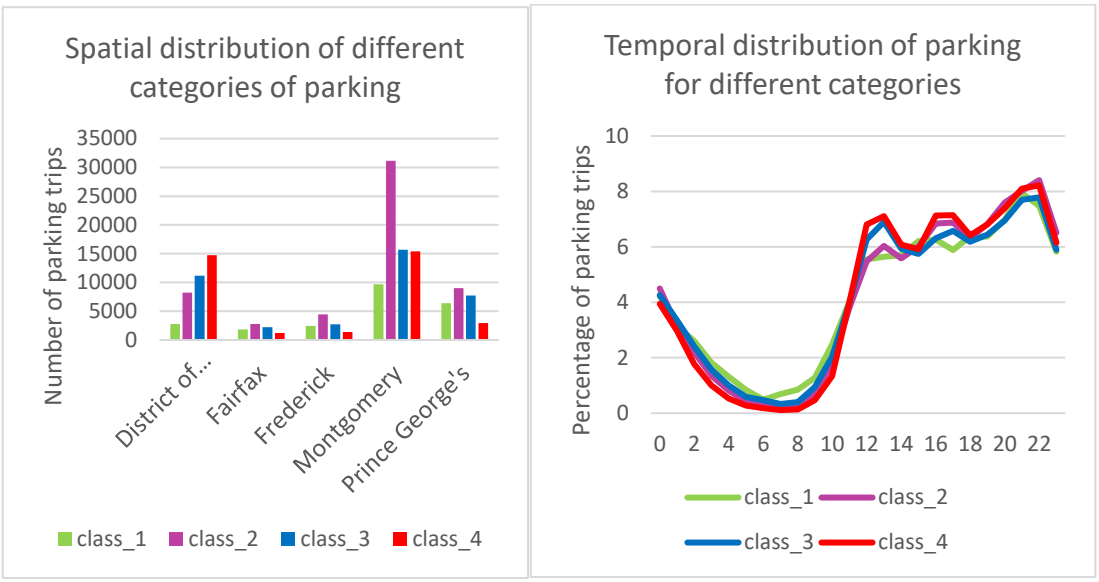
Outside of the City of Washington, DC, all counties in the broader study area showed more direct parking trajectories (i.e., category 1 and 2) than the two cruising

categories (category 3 and 4). We believe this distribution of different parking categories shows that the more serious parking issues were found in the City of Washington, DC where population density and retail and employment opportunities are higher and possibly more parking restrictions are in place, than in the Metropolitan region. The latter results showed significantly more trajectories in category 2 (i.e., direct parking where there were more waypoints and a shorter time to park), around 43% of total parking in Montgomery County.

To investigate possible reasons for the significant differences in the distribution of parking in category 2 among counties, we investigated land use of the destination points for parking trips in 3 counties in Maryland (Prince George's County, Frederick County, and Montgomery County, see Figure 1). Montgomery County showed a higher proportion of parking ending in residential areas, around 45.6% of parking ended in residential areas in Montgomery County and only 33.2% and 28.2% for Prince George's County and Frederick County respectively. In addition, for trips in category 2, these trips in Montgomery County also had a higher proportion that ended in residential areas compared with the other two counties in Maryland, 46.6% of trips in Montgomery County, and 37.7% and 31.4% for Prince George's County and Frederick County respectively. From this analysis, we believe that a higher proportion of trips in Montgomery County were trips going home and involved quick and direct parking in a residential area.

From the temporal perspective, the different parking categories showed a similar distribution to each other over a 24-hour period (fig. 3. 11 b). Category 4 with the longest

times for cruising reached a slightly higher maximum during the peak hour periods and were also slightly lower during the nighttime hours.



(a) *(b)*

Figure 3. 11 (a) Spatial and (b) temporal distribution of different parking categories for counties and cities that contained more than 3% of the data in (a).

3.6.2 Validation of modelling results

After classification using the deep clustering model, we performed a validation step by manually inspecting and classifying 1000 parking trajectories randomly sampled from the dataset. For the manual labelling process, we inferred the categories of parking by inspecting the driving behaviours of trip trajectories on a road map and we classified the trips into one of the four categories based on manual interpretation of the travel behaviours. However, it should be noted that unlike the deep clustering model that takes quantitative features such as time-to-park duration in seconds, parking distance in meters, and degree of turns as input, the manual classification was based mostly on an intuitive or experiential understanding of the parking process in different scenarios. The labelling process was conducted by 4 volunteers with graduate-level geospatial data science

backgrounds. All volunteers are drivers and have experience driving and parking in the Washington DC metropolitan area and the City of Washington, DC. Each one of the volunteers labelled independently.

Table 3. 2 Comparison between deep clustering categories and manual labelled categories

Deep Clustering Categories \ Manual Labeled Categories	Category 1	Category 2	Category 3	Category 4
Category 1	125	9	29	0
Category 2	1	283	32	44
Category 3	8	43	202	13
Category 4	0	40	26	143

In Table 3. 2, columns represent categories from the manually labeled results and rows represents categories from the deep clustering model where Category 1 and Categories 2 represent the two *direct parking* categories, and Category 3 and Category 4 are the *searching* and *cruising* categories. The diagonal cells of Table 2 show the degree to which the two classification results agree with each other while the remaining cells show the degree to which these two results confuse each other. Due to the trips that used for verifications were independently sampled from the dataset for each of the volunteer, 2 trips were double assigned to the volunteers, and there were 998 trips in total showed in this table (both of these trips received the same labels from different volunteers).

Comparing the manually labeled dataset and the classification results overall, 76% of the manually identified parking trajectories were classified as the same category as the deep clustering model. By category, 76.7% of Category 1 trajectories identified by manual inspection agreed with the model, while 78.6% of Category 2, 75.9% of Category 3 and 68.4% of Category 4 agreed with the model results.

The largest difference between the deep clustering classification categories and the manual classification results was between Category 4 (longest cruising), and Category 2 (direct parking with more waypoints). This may indicate that distance was counted more highly among manual classifiers. Another difference could be due to the fact that during manual labeling, turning in a small spatial range was interpreted as a cruising category, while the deep clustering approach might have been trained to compress and extract this part of travel information along the waypoints.

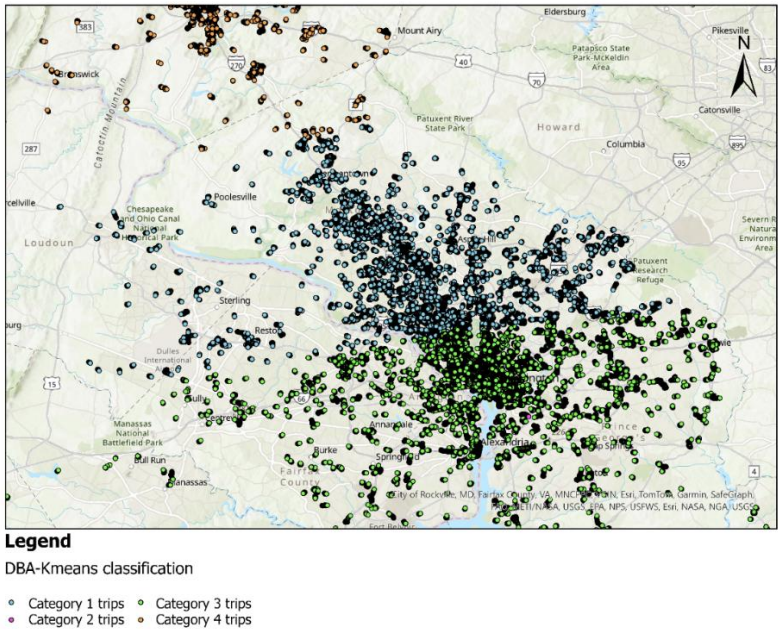
3.6.3 Model comparison results

We then compared the parking classification from our deep clustering model with the baselines. Since we do not have ground truth data to measure the model accuracy, we used the labelled verification data to examine the difference among models in addition to manual examination on the map.

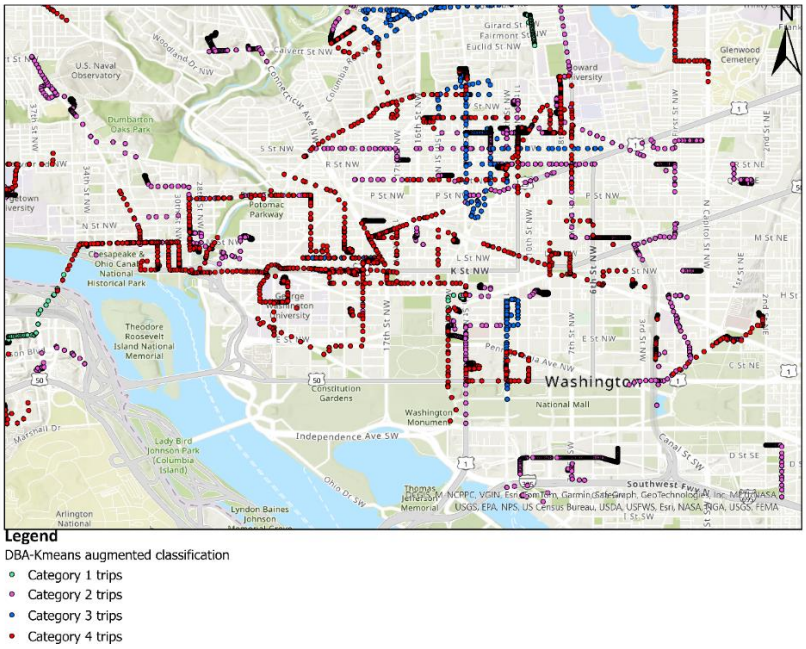
From the map, categories of parking trips from the DBA-Kmeans showed three mutually exclusive clusters in the study area (fig. 3. 12 a). We believe it is because the DBA-Kmeans classified parking trajectories based on their spatial location and failed to capture the travel behaviours (e.g., number of turns, duration of parking) along the parking.

Classification from the DBA-Kmeans* showed a clear bias in the number of trips in different categories with around 74% of parking trajectories classified into one class (we named it Category 3), and the rest 3 categories contain around 23%, 4%, and 1% of the trips respectively. After examination the categories on the road map, compared with clustering based on the longitude and latitude, DBA-Kmeans* on the feature augmented trajectories captured some of the parking behaviours. The model successfully recognized

some of the cruising trips (category 4 trips as a cruising type shown in red in fig. 3. 12 b) but it also showed many confusions among the types. We then compared the classification from the DBA-Kmeans* with our labelled verification dataset (Table 3.3). From the comparison, although in categories 2 and 4 (one direct parking type and the cruising type), the DBA-KMeans* generated a similar classification as our labelled data, it classified the majority of trips into category 3, which is in line with what we found via the map examination.



(a)



(b)

Figure 3. 12 Four parking categories classified by (a) DBA-Kmeans model and (b) DBA-Kmeans model*

Table 3. 3 Comparison between DBA-Kmeans categories and manual labelled categories*

DBA-Kmeans* Categories \ Manual Labeled Categories	Category 1	Category 2	Category 3	Category 4
Category 1	0	0	2	7
Category 2	1	192	9	22
Category 3	133	182	273	136
Category 4	0	1	5	35

None of the clustering models with the LCSS as the distance metric (LCSS-DBSCAN, LCSS-DBSCAN*, LCSS-KMeans, and LCSS-KMeans*) works well on parking trajectory classification. By setting Epsilon as 0.5 and minPts as 100, the LCSS-DBSCAN classified parking trajectories into 6 categories but one category accounts for more than 92% of all parking trajectories. LCSS-DBSCAN*, LCSS-KMeans, and LCSS-KMeans* classify all parking trajectories into only one category on all combinations of

the parameters. We believe it could be due to LCSS is not suitable for the relatively short trajectories with detailed travel behaviour in the parking scenario.

As for the deep learning baselines, the LSTM autoencoder + K-Means generated a more balanced classification but LSTM autoencoder + K-Means* generated an unbalanced result. The LSTM autoencoder + K-Means model classified 19%, 43%, 17%, and 21% of the trips into 4 categories respectively (Table 3.4). However, the LSTM autoencoder + K-Means* model classified 31%, 40%, 24%, and 5% of the trips into the 4 categories respectively (Table 3.5). After comparing with our verification dataset, for category 1, category 2 (two direct parking categories), and 3 (the searching type), the deep learning-based models generated categories closer to our manual labels. We believe it is because the LSTM autoencoder works well on extracting the travel behaviours along with the parking, which also proved the advantage of the deep learning model in extracting high dimensional latent information compared with traditional distance-based metrics. Compared with the LSTM autoencoder + K-Means that only used longitudes and latitudes as features, the LSTM autoencoder + K-Means* that used 7 features clearly generated an even closer to our labels results in category 2-4, which further proves our thought that the feature augmentation could facilitate the deep representation learning in extracting latent travel behaviour along the parking. However, compared with our deep clustering model, both of these two deep learning baselines caused significant confusion between category 2 (one of direct parking) and category 4 (cruising type), and category 1 and 2 for the LSTM-Kmeans* model.

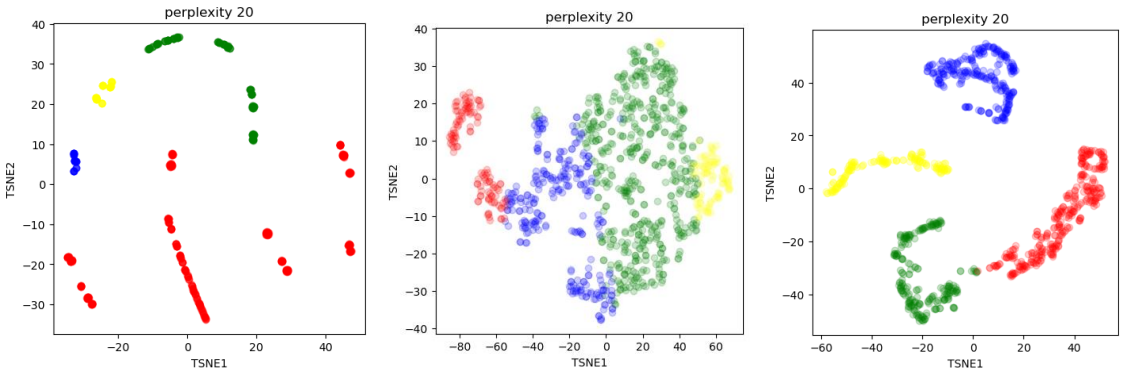
Table 3. 4 Comparison between LSTM-Kmeans categories and manual labelled categories

Manual Labeled Categories / LSTM-Kmeans	Category 1	Category 2	Category 3	Category 4
Category 1	130	12	48	0
Category 2	1	223	38	168
Category 3	3	30	131	4
Category 4	0	110	72	28

Table 3. 5 Comparison between LSTM-Kmeans categories and manual labelled categories*

Manual Labeled Categories / LSTM-Kmeans*	Category 1	Category 2	Category 3	Category 4
Category 1	130	102	55	23
Category 2	1	237	36	123
Category 3	3	33	197	7
Category 4	0	3	1	47

To further investigate the impact of joint optimization in the deep clustering model, we used a two-dimensional t-SNE plot over embeddings of LSTM-Kmeans, LSTM-Kmeans*, and our model for comparison. From t-SNE visualization figures (fig. 3. 13), we could clearly see that via the joint optimization process, the clusters separated more clearly and more balanced, which also proved the efficiency of our model in parking classification.



(a) *(b)* *(c)*

Figure 3. 13 Clustering results visualization of (a) LSTM-Kmeans, (b) LSTM-Kmeans, and (c) LSTM autoencoder based deep embedded clustering using t-SNE with 1000 parking trajectories and perplexity equals to 20.*

3.7 Conclusions and future work

Big GPS trajectory data brings new possibilities for quantitative driving behaviour analysis and can be used at larger spatial scales (e.g., regional scales) and finer granularities (e.g., at the individual level), which are useful for the foundation of the Intelligent Transportation System (ITS). In this study, we designed a computing framework consisting of a preprocessing model, a distributed parking trajectory extraction model, and an LSTM autoencoder deep clustering model that can extract and classify parking trajectories from passively collected data that is without labels. Taking advantage of SRDDs supported by Apache Sedona, the parking trajectory extraction model is suitable for trajectories with large data volumes and broader spatial coverage. The LSTM autoencoder deep clustering model used driving features with trajectories of varying lengths as input, and without labels, classified parking trajectories into direct parking or cruising behaviour categories as well as refinements of these subcategories. This analysis offers insights into the different steps in the parking process and how they occur in an urban environment and presents a method for cruising and direct parking trajectory extraction.

Our results showed that the average time-to-park durations were longer in the City of Washington, DC, than in the surrounding Washington DC metropolitan area. Inside of the city of Washington, DC, the cruising for parking is more significant in the commercial areas. Among the cruising categories classified by the deep clustering model, we identified a cruising category with little to no circling (Category 3) and one with circling properties that represented vehicles circling city blocks, parking lots, or parking garages (Category 4). We believe the Category 3 associated with the parking processes that the drivers were searching for parking but not involve much excess travel were involved. In this case, the driver might take the first available parking spot in the acceptable distance even the destination has not yet reached or passed a little bit (Weinberger et al., 2020). Trajectories of the Category 4 (the cruising type with the most circling), accounted for 22% of all the parking trajectories in our study area. We believe that these circling and turning in the city blocks correspond to excess travel due to not enough affordable parking resources that were discussed in previous research (Van Ommeren et al., 2012).

Searching or cruising for parking might show different searching radii and time duration in different urban areas due to the factors such as design/ density of road network and available parking spots. However, based on previous research in different study areas and our own examination, the cruising behaviours are always shown as "excess of travel". Therefore, we believe the framework (with different parameter settings such as radii in parking trajectory extraction) and the deep clustering model are generalizable for passively collected trajectory data collected in other urban areas. In addition, although the dataset we used in this study was collected for the Washington DC

metropolitan area, the study area contains a variety of different parking contexts from the perspective of residential density and traffic complexity. By applying the framework, locations subject to a higher frequency of cruising or searching for parking could be considered for further analysis, which will provide useful information for transportation management and facilitate the construction of the ITS.

Considering limitations associated with this research, one limitation is brought by the big GPS trajectory data where different sampling intervals exist within the data collection. Although the trajectory dataset we used had mostly a high sampling frequency and the imbalance is not significant especially after the preporcessing, it still has an impact on the classification (i.e., category 1 and 2). We didn't use a unified sampling interval or a random drop during training in this study since it would result in removing waypoints from the trips and may lose important travel behaviours captured a long the trips. Future work could consider including the random drop during training while finding methods to not lost important travel behaviour along parking. In addition, while we include a data preprocessing model in our framework to exclude anomalous trips that were not likely to include parking, there could still be some anomalous trips that were failed to be excluded (e.g., the in-vehicle GPS sensor lost connection while a vehicle was at a stopping or low-speed process). But we believe these trips would only count for a very small proportion of our trajectory dataset, and the big data helps to offset these issues to some extent. Another important issue was that circling on the street and in a parking-garage shared a very similar pattern and could be hard to distinguish from one another. Highly accurate POI/AOI dataset could be helpful to better distinguish these two different parking types, but since highly accurate POI/AOI dataset is inaccessible for the

authors, we believe this could also be explored in future studies. The data provider of the passively collected in-vehicle trajectory data may also bring biases since location-aware devices may not be evenly sampled across all regions and/or among all populations. Traffic conditions such as congestion during the trajectory data collection might also impact the accuracy of the parking classification since traffic speed was an important driving feature for parking classification by the deep learning model and this could be studied further in future research. Although the classification results are in line with the manual labeled datasets, it is noteworthy that we don't have real ground truth labels for parking categories to verify our classification results. We will try to collect some ground truth labels on cruising for parking trips if we have resource in the future works.

Chapter 4: Detecting propagation of non-recurrent congestion over road networks with an unsupervised deep segmentation model

4.1 Abstract

Compared to recurrent congestion (RC) caused by e.g., commuting, the impact area and propagation of non-recurrent congestion (NRC) induced by traffic incidents are more complex and unpredictable and therefore have the potential to bring more significant negative impact to the traffic flow. Although there has been research on the NRC impact area, most studies focused on only a small number of freeway segments and neglected the propagation of NRC over complex real-world road networks. In this research, taking advantage of passively collected trajectory data, we propose a research framework to investigate spatio-temporal propagation of NRC over complex road networks and disentangle the joint impact of RC and NRC. After preprocessing, the framework based on distributed computing first generates speed reduction index indicators and contour plots that represent difference between current speed and free-flow or historic average speed over the road networks. To overcome the challenges brought by data noise and a lack of labels, a W-Net-based unsupervised semantic segmentation model was developed to segment congestion impact ranges in the contour plots automatically. In addition to rules following the law of shockwave that are commonly applied, in postprocessing, we introduce an additional innovative step that separates NRC that were more likely to be caused by RC fluctuation. After experiments using passively collected trajectory data, contour plots for each congestion indicator were built for 7

different travel routes. Results show that our framework could identify the spatio-temporal impact area of NRC and distinguish them from RC. Moreover, our results revealed scenarios of NRC propagating between freeways via ramps and captured differences in propagation speed and impact area range.

4.2 Introduction

Non-recurrent congestion (NRC) refers to traffic congestion caused by unexpected incidents such as accidents and roadwork, leading to delays, increased driver frustration, and excess greenhouse gas emissions (An et al., 2016; Ou et al., 2020; Z. Zheng et al., 2023). Unlike recurrent congestion (RC), which typically associated with high traffic demand and follows predictable spatio-temporal patterns (e.g., rush hour congestion at specific locations), NRC can appear any time at any place, often having a more severe impact (An et al., 2016). Therefore, accurately estimating the spatio-temporal range of NRC's impact area (i.e., a section of the connected road network affected by congestion) and modeling its propagation (i.e., how congestion spreads across connected road sections) is crucial for predicting congestion duration, identifying contributing factors, and developing strategies to mitigate traffic delays.

Several approaches have been proposed to determine NRC's spatio-temporal impact, which can be categorized into four main categories. Static methods define impact areas using fixed thresholds but risk over- or underestimation due to the dynamic nature of traffic flows (Karlaftis et al., 1999; Moore et al., 2004; Raub, 1997; Z. Wang et al., 2024; H. Yang et al., 2018). Queue model-based methods estimate traffic queue lengths using statistical models, but their applicability to large-scale road networks is limited by the difficulty of obtaining reliable data on key factors such as lane capacity and incident

duration (Z. Wang & Jiang, 2020). Shockwave models used triangular shapes in a spatial and temporal speed contour to form NRC's impact area but have the limitation due to the restrictive triangular shape assumption and does not accommodate the shape of the impact area other than triangle (Z. Wang & Jiang, 2020). Finally, contour plot-based methods leverage traffic monitoring data (e.g., loop detectors, probe vehicles) to represent traffic statues across time and mileposts, with empirical thresholds or clustering techniques used to define congestion's impact areas (Z. Wang et al., 2024). However, most of these studies focus on a limited stretch of freeway (i.e., a few segments along one or multiple freeways) or use simulated road networks, neglecting NRC identification and propagation in actual road networks with the more complex topological (i.e., interchanges among freeways) and structural features (i.e., different number lanes) (Z. Wang et al., 2024).

To address these limitations and enable NRC analysis over large-scale real-world road networks, we propose a novel research framework based on passively collected trajectory data. The trajectory provides the advantages of broad spatio-temporal coverage and high sampling frequency, forming a solid foundation for examining congestion propagation in actual road networks. Unlike traditional traffic monitoring systems that capture traffic status only at specific locations, trajectory data allows for continuous tracking of traffic conditions across connected roads. Specifically, trajectory data enables the estimation of traffic conditions on multiple road types, including freeways and ramps, facilitating a more comprehensive examination of NRC propagation between different road types, including its spread from one freeway to another via ramps.

The proposed research framework consists of three main components: a distributed contour plot-building module, a W-Net-based unsupervised semantic segmentation model, and a postprocessing module that ensures consistency with shockwave propagation laws. The contour plot-building module first estimates two Speed Reduction Index (SRI) indicators that measure speed changes over time for each road segment, one relative to speed at a free-flow condition and the other to historical average speed. These indicators are then used to generate spatio-temporal contour plots for each designated route. By this means, clusters with high SRI values in the contour plots represent the impact areas of all types of congestion (both RC and NRC) and congestion that are likely to be NRC, which provides a foundation for distinguishing NRC from general congestion patterns. This module is designed based on Apache Spark and Sedona, enabling efficient processing of large-scale trajectory datasets across extensive road networks. To determine the spatio-temporal range of the congested areas from the contour plots and to overcome the challenges brought by data noise, a W-Net-based unsupervised semantic segmentation model is used. The segmentation model takes contour plots as inputs and generates pixel-level classifications for the spatio-temporal range of all congestion including NRC in the contour plots without any labels or predefined thresholds. After segmentation, in addition to the rules that make sure segmented NRC follow the law of shockwaves, we apply a new rule that identified segmented NRC that are more likely to be caused by the fluctuation of RC for further investigation. Neglecting this step might result in an overestimation of NRC and so our approach has developed this additional check to improve our detection of.

We validated the proposed framework using a passively collected trajectory dataset, consisting of approximately 67 million trips and 3.4 billion waypoints collected over six months (April–September 2018) in the Washington, DC, metropolitan area. The study analyzed seven travel routes covering both freeways and ramps, generating over 800 contour plots per SRI indicator. On average, 20 NRC per plot were detected, totaling nearly 17,000 NRC were detected. We also investigated scenarios where NRC propagated from one freeway to another, revealing variations in spatial impact range and propagation speed. The results demonstrate that our framework enables a detailed examination of NRC over complex real-world road networks and therefore facilitate our understanding on congestion in real-world. This research also laid the foundation for future research on NRC propagation including predicting the congestion duration and maximum impact range over actual road networks.

The rest of the paper is organized as follows: Section 2 reviews related work on congestion range estimation and propagation modeling; Section 3 describes the study area and datasets; Section 4 details our research framework, including a distributed contour plots building module, a W-Net-based segmentation model, and a post processing module; Section 5 presents our experimental results; and Section 6 presents our conclusions and ideas for future work relating to this research.

4.3 Related works

In previous research on NRC impact area estimation and propagation modeling, static methods (i.e., methods that use fixed spatio-temporal thresholds to estimate the range of impact areas) have been firstly applied in earlier research. Among these studies, Raub, (1997) first used 1 mile from the incident location as the spatial range and 15

minutes plus the incident duration as the temporal range for the incident impact area estimation. Later, there have been many other research that used a similar method to estimate the impact area of the incidents with different spatial and/or temporal thresholds (Karlaftis et al., 1999; Khattak et al., 2009; Moore et al., 2004). However, methods with a fixed spatial and temporal range could easily over or underestimate the range of the incidents since they do not consider the variance of the traffic situation (Z. Wang et al., 2024; H. Yang et al., 2018).

To overcome the shortcomings of the static model, queuing model-based methods and shockwaves-based modes have then been proposed for NRC impact area estimation (Ou et al., 2020). The queuing model-based methods use a varying queue to model the dynamic NRC impact area along the road segments and use multiple types of statistical model to estimate the queue length by considering a series of contributing variables, such as traffic speed information, lane information, traffic arrival rate, departure rate (C. C. Sun & Chilukuri, 2010; Z. Wang et al., 2024; H. Yang et al., 2018). However, an importation limitation of the queuing model-based methods is that detailed data for the contributing variable is challenging but necessary for a reliable model, which makes this type of method hard to be applied for the NRC propagation in the real-world large-scale road network (Z. Wang et al., 2024). Studies that used shockwave models generally assume the impact area of the NRC is a triangular shape in a spatial and temporal speed contour formed by two simplified straight shockwave lines which represent the "forming" and “discharging” of the congestion respectively (Sarker et al., 2015; Zheng et al., 2014). However, although this type of methods considers the dynamic of the NRC propagation,

it shows a high degree of restrictiveness since it only accommodates a triangular shape for the NRC impact area (Ou et al., 2020; Z. Wang et al., 2024; H. Yang et al., 2018).

With the increasing types and amount of data on traffic status monitoring that has become available, empirical methods based on contour plots have been proposed to examine the dynamics of the NRC impact area. Contour plots-based methods would first establish a contour plot using one or multiple types of traffic indicators (e.g., speed, traffic volume, speed change ratio). Then, each cell (i.e., traffic indicator for a specific road segment at a given timestamp) will be determined to be congested or not using different methods. For example, Chen et al., (2016) used a series of empirical thresholds including 80 percentile of historic delay and twice the traffic density as the upper stream traffic station to estimate the impact region of the incidents. The binary integer models have also been used in determining the spatio-temporal impact area based on contour plots and considered the shape of the propagation as the shockwave (Chung, 2013; Chung & Recker, 2012). However, these methods include empirical thresholds, which could lead to over or underestimation of the impact area for the variation of the traffic condition. Also, the determination of empirical thresholds is time and labor consuming. In recent years, clustering models have also been applied in estimating the range of incident impact area based on contour plots in which empirical thresholds are not needed. Ou et al., (2020) first applied a revised fuzzy C-Means (FCM) model to estimate the range of incident impact area from the spatio-temporal contour plots and then used a series of postprocessing steps to segment individual incident and also make sure the propagation of the NRC follows the shockwave theory. In addition, the researchers used historic traffic speed and volume instead of free-flow values to build their indicators in order to

disentangle the compound impact of RC and NRC. Zheng et al., (2023) also applied an improved FCM model and innovatively introduced three constraints in the optimization process to ensure the propagation of the NRC follows the law of shockwaves. However, the aforementioned research focused the NRC and their propagation on a stretch of freeway and neglected the complex topological structure of road networks (Z. Wang et al., 2024). Moreover, although some researchers considered the joint impact of RC and NRC, solely using historic speed to extract NRC impact area may lead to an overestimation since the spatio-temporal impact area of RC could also show fluctuations (e.g., longer than usual RC could also be shown as an impact area in a contour plot based historic speed or volume).

In addition to the 4 main categories introduced above, Wang et al., (2024) proposed an optimization model with a set of novel constraints to formulate the propagation of the NRC over road networks using simulation data. However, congestion propagation over the real-world road network is more complicated than the simulation data. For example, (Z. Wang et al., 2024) set a scenario of all road segments contain 3 lanes, but the real-world road networks are far more complicated (e.g., two freeways connect by ramps). In addition, these researchers didn't consider the joint impact of RC and NRC in their research.

In our research, we introduced a new research framework based on contour plots generated using trajectory data to examine the spatio-temporal propagation of the NRC over complex real-world road network. To the best of our knowledge, we are the first to examine the propagation of NRC from one freeway via the ramp and investigate the different propagation ranges and speeds. In addition, we innovatively introduced a

postprocessing rule that could exclude the impact of RC fluctuation on NRC identification, which otherwise might cause an overestimation.

4.4 Data and study area

The passively collected trajectory data used in this research was collected in the Washington, DC metropolitan area, which includes the City of Washington, DC and part of Maryland and Virginia, for 6 months from April to September 2018. The trajectory data was collected through in-vehicle GPS sensors by INRIX and therefore the dataset contains only vehicle trips.

In the dataset, trips were predefined by the data vendor and uniquely identified by trip IDs. Each trip in the dataset is comprised of a series of temporally consecutive GPS waypoints. Each waypoint contains attributes including: longitude, latitude, device ID, trip ID, timestamp, raw speed, the time difference from the previous point of the same trip, and distance from the previous point of the same trip.

Table 4. 1 Number of trips and waypoints for different month in the trajectory dataset

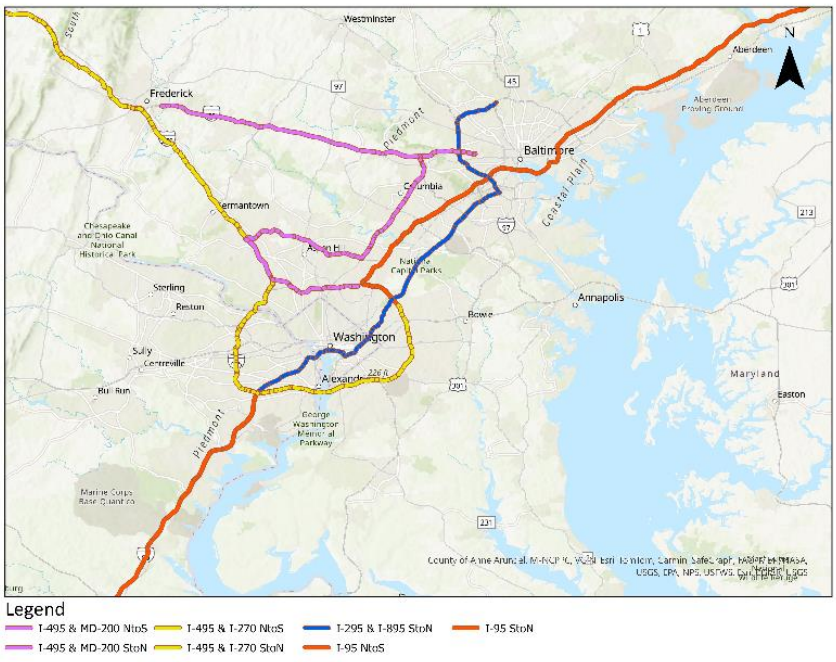
Month	Num of Trips (million)	Num of Waypoints (million)
April	11	870
May	12	584
June	12	529
July	11	509
August	11	520
September	10	438

The raw dataset contains 67 million trips and 3.4 billion waypoints and has an average of 51 waypoints per trip. Overall, the dataset shows a high sampling frequency where approximately 49% of waypoints had a sampling interval within 10 seconds. The

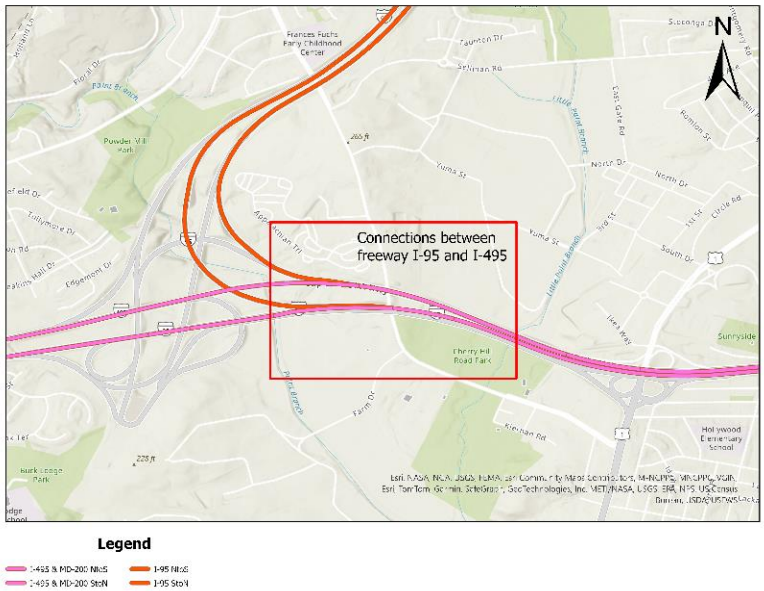
higher frequencies could help improve the capture of traffic information along all road segments, especially for those roads that are not covered by monitoring stations such as ramp roads.

In this research, we designed 7 travel routes for NRC impact area assessment and propagation investigation study (fig 4.1 a). The routes are formed along major freeways and ramp roads in the Washington DC metropolitan area, including: I-95, I-495, I-295, I-895, I-270, and MD-200. Each route contains 224 road segments for deep learning training purposes, which is discussed in detail in the Methods section. The total length of the routes range from approximately 60 miles (e.g., route I-295 to I-895 from south to north) to 130 miles (e.g., I-95 from north to south), with an average of 105 miles.

Specifically, we included 8 connections between different freeways in order to examine the propagation of NRC from one freeway to another (fig 4.1 b). The road network dataset for this research is from OpenStreetMap.



(a)



(b)

Figure 4. 1 (a) 10 routes designed based on freeways and link roads in the DC metropolitan area; (b) an example of the connections between two freeway routes.

4.5 Methods

To estimate the spatio-temporal range of NRC impact areas using trajectory data, and to examine the propagation of congestion on real-world complex road networks, a research workflow has been designed for this purpose. The workflow consists of a distributed congestion plot building module, a W-Net-based congestion impact area segmentation model, and a series of postprocessing steps to identify the NRC from the RC and to ensure the propagation of NRC follows the law of the shockwaves.

4.5.1 Distributed contour plots building module

The module is designed to take trajectory data as input, generate congestion indicators for the road segments for different timestamps, and build the contour plots for the different travel routes. The framework is designed based on the distributed computing environment of Apache Spark and Sedona to overcome the challenges brought by large

data volumes and broad spatial coverage. The module consists of 3 sub-modules, including a preprocessing module, a map matching and gap filling module, and a congestion plot generating module.

4.5.1.1 Preprocessing

The preprocessing module is designed to detect and exclude trip and waypoint anomalies in the trajectory dataset. Trip anomalies refer to trips with less than 3 waypoints, which could be caused by reasons such as recording ends during a trip, and all trip anomalies are excluded from the dataset. Waypoint anomalies refer to data noise, which shows as one or several waypoints suddenly "float away" from the trip. This type of anomaly could be caused by GPS sampling errors or other types of errors (Merry & Bettinger, 2019). We included an empirical threshold θ_{speed} on the speed at waypoints calculated from their GPS location and sampling interval (equation 1).

All waypoints ($\mathrm{Waypoint}_i$) with GPS speed $V(\mathrm{Waypoint}_i)$ exceed the θ_{speed} are excluded.

$$V(\mathrm{Waypoint}_i) = \frac{\Delta(\mathrm{Waypoint}_i, \mathrm{Waypoint}_{i+1})}{\Delta(\mathrm{t}_i, \mathrm{t}_{i+1})} \quad (1)$$

Where $\Delta(\cdot)$ refers to the difference method and t_i refers to the timestamp of $\mathrm{Waypoint}_i$.

4.5.1.2 Map matching and gap filling

Map matching is a common step in trajectory analysis and its main purpose is to align each of the waypoints to the road segment that the vehicle was traveling on at the time of data collection. By aligning the waypoints to the road segments, traffic status of

the corresponding road segment such as traffic speed can be estimated. However, due to GPS sampling errors and the dense road network, it is likely that the recorded coordinates of waypoints are closer to other segments that those that the waypoints were actually located on (Merry & Bettinger, 2019). In this research, we applied a distributed map matching module proposed in Zhang et al., (2023) that is designed to consider both the distance between the waypoints and road segments and the angle between the traveling direction and the direction of waypoints in the matching for better matching accuracy. Taking advantage of SRDDs supported by Apache Sedona, the KDB-Tree spatial partitioning and R-Tree index were used, which makes it suitable for datasets with large spatial coverage (J. Yu et al., 2019a).

Due to the irregular sampling interval in the trajectory dataset, GPS sensors only record certain locations along a trip and leave gaps between waypoints. By closing the gaps between the road segments that are matched to two consecutive waypoints and estimating the traffic status of road segments in the gap using these consecutive waypoints, the traffic status of the road segments along the entire trip trajectory would be able to be estimated. Therefore, in this study, we applied distributed gap filling that is designed using Apache Spark and iGraph (Armbrust et al., 2015; Ju et al., 2016; Zaharia et al., 2010).

4.5.1.3 Congestion indicators estimation and contour plot building

After traffic status was estimated for the road networks, congestion indicators for the road segments were then calculated. Congestion indicators can generally be classified into 5 categories, including indicators based on speed, travel time, delay, level of services, and congestion indices (Afrin & Yodo, 2020). Based on research finding in

Afrin & Yodo, (2020), except for indicators based on traffic volume and road capacity, the other 4 types of indicators generate similar results on traffic congestion measurement since they are essentially estimated by traffic speed. Therefore, in this research we introduce two indicators for road segments based on speed since traffic volume data at fine spatio-temporal granularity was inaccessible (ground truth is still necessary to scale and correct the volume estimates even if we use trajectory data to estimate traffic volume (Fan et al., 2019)). The two indicators were designed based on a speed reduction index (SRI), which measured the relative difference between current speed and comparative speed for a corresponding road segment.

The first indicator (SRI_{ff}) is designed to measure the relative difference between current speed of each road segment at a certain time step (v_{ac}) and the speed of the road segment at its free-flow condition (equation 2).

$$SRI_{ff} = \left(1 - {v_{ac}}/{v_{ff}}\right) * 10 \quad (2)$$

Following Federal Highway Administration (FHWA), (2019), the free-flow speed (v_{ff}) is calculated as the 85th percentile of the off-peak (weekdays from 9 am to 4 pm, and weekends from 6 am to 10 pm) speed. This indicator captures all types of congestion (both RC and NRC).

The second indicator (SRI_{hs}) is designed to separate the compound impact of NRC and RC by measuring the relative change between current speed and historical average of traffic speed (v_{hs}) of road segments (equation 3) (Ou et al., 2020).

$$SRI_{hs} = \left(1 - {v_{ac}}/{v_{hs}}\right) * 10 \quad (3)$$

For each road segment at a given time interval, its v_{hs} is calculated as the average value of the speed value for that time of the day, day of the week in that month. Unlike v_{ff} that does not consider RC, the v_{hs} captures the traffic speed of road segments during RC (if there is one at that time) and therefore SRI_{hs} could be used to distinguish the separate impact of NRC from a RC (Ou et al., 2020).

After the two indicators are computed for the road network for all times, contour plots are built (Fig. 2 a). For a contour plot, the X-axis represents time flow, and the Y-axis represents a route in the order of traffic flow. Each cell in the contour plot represents an SRI indicator (e.g., SRI_{ff} or SRI_{hs}) for a given road segment Seg_i at time interval t_j. The spatio-temporal range of congestion and their propagation can be depicted as clusters of cells that have indicators higher than the normal condition (i.e., free-flow condition or historic average condition). The SRI_{ff} contour plots depict both RC and NRC but SRI_{hs} contour plots will only show congestion with speeds different from historic average values, which make them more likely the incident induced NRC (Fig. 2 b). Linear interpolation is used for cells that do not have enough waypoints to estimate SRI indicators.

To keep the balance of the fine temporal granularity and the accuracy of the traffic speed indicators (i.e., longer temporal intervals will mean that road segments have more waypoints to be matched to them and therefore offer higher accuracy), we used 10 minutes as the temporal interval in this research.

		Time flow →				
		t_1	t_2	…	…	t_n
Traffic flow direction ↓	Seg_1	SRI	SRI	…	…	SRI
	Seg_2	SRI	…	…	…	…
	…	…	…	…	…	…
	…	…	…	…	…	…
	Seg_n	SRI	…	…	…	SRI

(a)

	Time flow →				
Traffic flow direction ↓	SRI_{hs}	SRI_{hs}	…	…	SRI_{hs}
	SRI_{hs}	…	NRC	…	…
	…	…	NRC	NRC	…
	…	NRC	NRC	NRC	NRC
	SRI_{hs}	…	…	…	SRI_{hs}

(b)

Figure 4. 2 (a) Diagram of a contour plot generated using SRI indicators; (b) an example of NRC propagation in SRIhs contour plots that follows the law of shockwaves.

4.5.2 W-Net based segmentation model

After building contour plots for routes using the SRI indicators, the next step was to identify and segment the spatio-temporal range of each congestion. Traditionally, empirical thresholds are used for this step, but this could easily lead to an over or underestimation (Z. Wang et al., 2024). In recent research, clustering methods (i.e., FCM ++) have been used to automatically identify congested areas in contour plots generated using simulation data and monitoring station data (Ou et al., 2020; Z. Zheng et al., 2023). The need to handle data that is uncertain and vague make data-driven methods a good choice for the congestion range segmentation task (Ou et al., 2020; Stetco et al., 2015). However, classic clustering-based image segmentation methods have their limitations such as considering only pixel-level information and can work poorly in real-world scenarios where the data source contains noise (Jiang et al., 2021; Kaur et al., 2023).

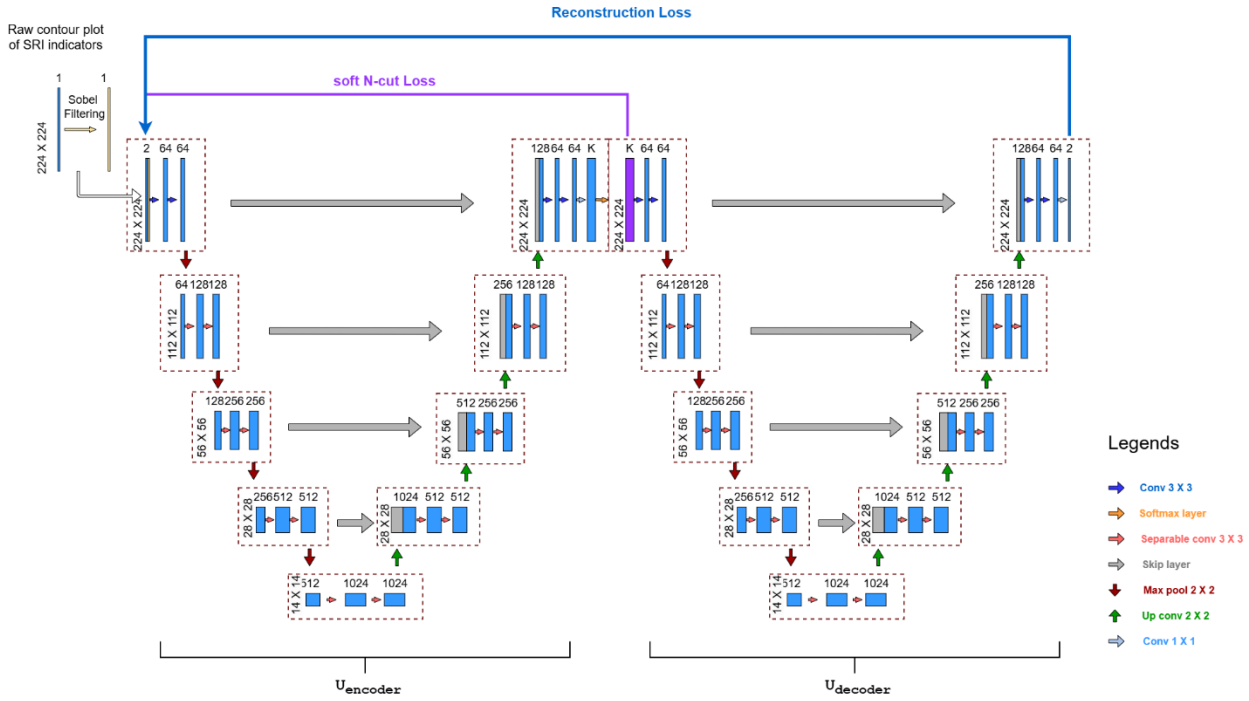


Figure 4. 3 The W-Net based unsupervised semantic segmentation

For these reasons, a W-Net based fully unsupervised semantic segmentation model is proposed to overcome the challenges of data noise and a lack of ground truth data for training. The model is a fully convolutional autoencoder (CAE) composed of two concatenated FCN architectures that are similar to the U-Net architecture and form a W-shaped architecture together (Xia & Kulis, 2017) (fig. 4.3). The first FCN architecture works as an encoder by compressing the latent information in the input image into a K-way soft segmentation and the second architecture reverses this process by reconstructing the segmentations into a reconstructed image. Similar to the original U-Net structure, the U-shaped encoder consists of a contracting path (the first half of the U-shape) that is designed to capture the complex spatial context (i.e., learn the essential features in early layers and complex patterns in later layers) and an expansive path (the second half of the U-shape) to localize the spatial features. The architecture of the U-shaped decoder is similar to the encoder, except for the shape of the input and output. Both the U-shaped

encoder and decoder contain 9 modules and each module consists of two 3 * 3 convolutional layers, followed by a ReLU and batch normalization process. In the contracting path in both the encoder and decoder, modules are connected using 2 * 2 max-pooling layers and the number of features doubles in these steps. For the expansive path, however, the 2*2 up convolution layers are used to connect the modules and features are halved in these steps. Specifically, in the last step of the encoder, a 1 * 1 convolution is used to map the 64-dimensional feature to K-classes. Similarly, a 1 * 1 convolution is used in the last step of the decoder to reconstruct the 64-dimensional feature to the image of the input shape. The U-shaped encoder and decoder are connected with a SoftMax layer that rescales the K-dimensional classes so that they lie in the range of 0 and 1 and sum to 1.

There are two losses generated by this W-Net architecture, namely a reconstruction loss of the autoencoder and a soft normalized cut (N-cut) loss generated in the encoding layers. The reconstruction loss is used to measure the difference between the input and output layers and the soft N-cut generates a soft vector to evaluate the consistency of pixels of the images. These two losses are jointly minimized in the training process.

Although the CNN-based models perform well with the task of image segmentation, a lack of smoothness constraints may cause an issue where the boundaries of the segmented objects are not accurate. Therefore, after the initial pixel-level classification from the W-Net-based model, following the postprocessing step in Xia & Kulis, (2017), we applied a fully-connected conditional random fields (CRF) for more accurate boundary recovery. In addition, to further capture the boundary of the

congestion impact area, Sobel filtering is used to generate an edge-emphasized image as a layer of input in addition to the contour plots. The shape of the input for the W-Net-based model is 224 * 224 * 2, which means each input contour plot will cover 224 * 10 minutes, which is 37 hours and 20 minutes, and 224 road segments (it is noteworthy that the trained model could be used to segment contour plots with any shape and size).

The W-Net-based segmentation model was implemented using *PyTorch* version 2.3 and Python 3.8, and the CRF was implemented using the *PyDenseCRF* library. In addition, the Python *imgaug* library was used for the image augmentation process before the training and 16 distinct images were generated for each original contour plot using strategies including random rotation, flip, blurring, and cropping processes.

4.5.3 Postprocessing

After segmentation, preliminary NRC and overall congestion areas can be identified using contour plots generated with SRI_{hs} and SRI_{ff}. However, due to data noise in the contour plots caused by noise in the trajectory data or traffic fluctuations, the preliminary NRC area segmented by the deep learning model might not be the final version of an incident-induced NRC impact area. Also, the boundary of any preliminary NRC impact area might not be accurate.

	Time flow →				
Traffic flow direction ↓	Normal traffic	Normal traffic	…	…	Normal traffic
	SRI_{hs}	…	NRC	…	…
	…	…	NRC	NRC	…
	…	NRC	NRC	NRC	NRC
	Normal traffic	…	NRC/remove	…	NRC/remove

(a)

	Time flow →				
Traffic flow direction ↓	NRC	NRC	NRC	…	Normal traffic
	NRC	NRC	NRC	NRC	Normal traffic
	…	…	NRC	NRC	…
	…	…	NRC	NRC	…
	Normal traffic	…	…	…	Normal traffic

(b)

	Time flow →				
Traffic flow direction ↓	Normal traffic	Normal traffic	…	…	Normal traffic
	Normal traffic	…	NRC	NRC	NRC
	…	…	Normal traffic /fill	NRC	…
	…	NRC	NRC	…	…
	Normal traffic	…	…	…	Normal traffic

(c)

	Time flow →				
Traffic flow direction ↓	Normal traffic	Normal traffic	NRC	…	Normal traffic
	Normal traffic	NRC	NRC	NRC	…
	…	NRC	RC	RC	…
	NRC	RC	RC	RC	RC
	Normal traffic	…	…	…	Normal traffic

(d)

Figure 4. 4 Diagram of SRIhs contour plots where preliminary NRC areas identified by the segmentation model are shown in red. (a) Propagation of NRC violates rule No. 2, cells bounded by green will be removed; (b) preliminary NRC violates rule No. 2 and will need further investigation; (c) propagation of NRC violates rule No. 1, cells bounded by green will be filled; (d) preliminary NRC only show at the boundary of the corresponding RC. This type of NRC is more likely to be caused by fluctuation of RC areas and need further investigation.

In previous research, it has been widely accepted that the shape of NRC impact areas should follow the propagation of shockwaves. This law of shockwaves can be described using 3 topological rules:

1. The propagation shockwave of the NRC areas must be uninterrupted
2. The spatial propagation of the shockwave must move from upper stream to the lower stream of the traffic
3. The spatio-temporal boundary of the NRC shockwave must be contiguous.

Based on these 3 rules, postprocessing steps were applied:

For each preliminary NRC_i in all NRC segmented from the SRI_{hs} contour plot:

a. Find the road segment with the highest stream (i.e., Seg_{max}) that have continuous cells over θ_{seg} along the temporal axis as the start road segment of the NRC_i. The cell with the minimum timestamp in Seg_{max} is defined as $Cell_{start}$, and the timestamp of the $Cell_{start}$ is defined as $t_{Cellstart}$;
b. Label all cells with Seg_i larger than Seg_{max} as normal traffic since they violate rule No. 2 (fig. 4. 4 a);
c. For the NRC cell with the minimum timestamp t_{min}, if $t_{Cellstart} - t_{min} > \sigma_{time}$, label current NRC_i as further investigation needed since they violate rule No. 2 (fig. 4. 4 b), otherwise, label all cells with timestamp smaller than $t_{Cellstart}$ as normal traffic, since NRC only propagate along the time flow;
d. For each time step, fill all gaps among cells that are labeled as NRC, following rule No. 1 (fig. 4. 4 c);
e. For each road segment, fill all the gaps among cells that are labeled as NRC, following rule No. 1;

It is noteworthy that most if not all previous research, to the best of our knowledge, applied an assumption that the propagation of NRC followed a constant speed. However, this is not necessarily the case in actual traffic since the propagation might be impacted by multiple factors such as the number of lanes and traffic volume. Therefore, we didn't include this rule as a part of our postprocessing workflow.

In addition to these rules, as we introduced earlier, fluctuation of RC impact areas (e.g., due to RC starting earlier, and/or lasting longer, and/or causing longer congestion over the roads) can also generate preliminary NRC impact areas in the SRI_{hs} contour plots and follow the law of shockwaves (fig. 4. 4 d). This type of congestion area is not incident-induced however and might cause an overestimation if we only used the rules introduced above. Therefore, as part of this research, we have introduced a novel rule to identify this type of congestion area for further inspection.

f. If the spatio-temporal impact area of NRC_i showed only at the boundary area of a corresponding congestion detected in the SRI_{ff} contour plot, the NRC_i may be due to RC fluctuation and is labeled as needing further investigation.

In implementation, an empirical threshold of a quarter of the width or length of the congestion area were used to define the boundary area of a congestion.

In addition to aforementioned steps, there is also one splitting step to separate congested areas if they are only connected by data noises (i.e., several cells in contour plots). The splitting step is implemented based on empirical thresholds that for any row or column if it contains less than of a quarter congested cells of the length or width of the

congestion area and the two neighbor rows or columns have more than two-thirds of the length or width of the congestion area, label this row or column as normal traffic and split current congestion area into two. This step is implemented as the first step of the postprocessing.

All postprocessing steps are implemented using Python *Numpy* library.

4.6 Results

Using trajectory data collected in the Washington, DC metropolitan area for 6 months, 19 or 20 plots were generated for each month (depending on the number of days in each month), for each of the SRI indicators, and for each route, resulting in 833 plots in total for all SR indicators. For the last contour plot of each month, part of the data from the next month was borrowed and duplicated to generate a full plot and to make full use of the data we have. After train-validation splitting, we got 23984 training data and 2672 validation data for the model training.

We tested the parameter number of category (K) with the range from 2 to 32, and after manually examining the output, K equals to 14 generated the clearest segmentation results for contour plots generated using both SRI_{ff} and SRI_{hs}. The model converged after 12 epochs of training on an NVIDIA Tesla V100 GPU.

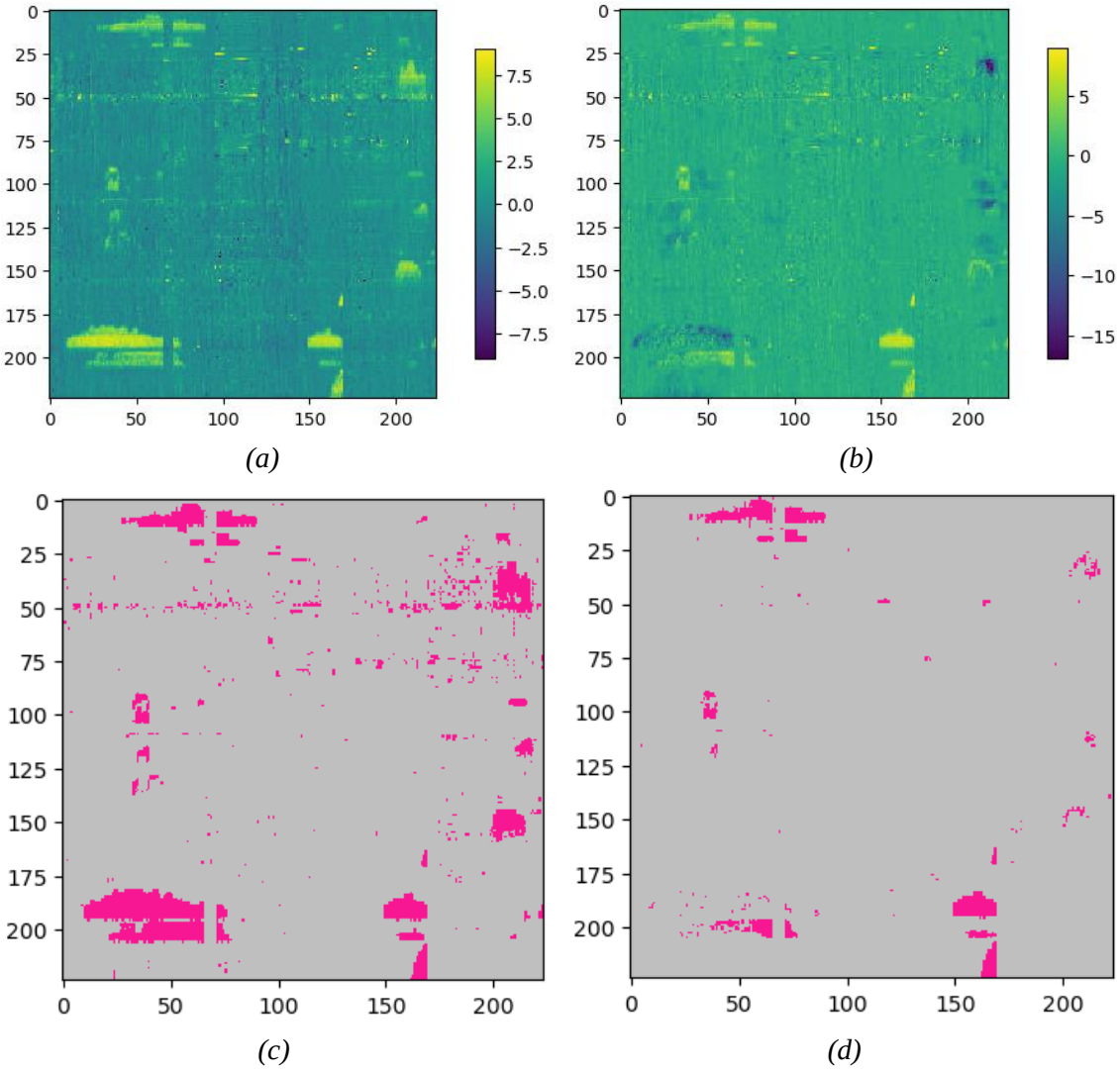
4.6.1 Segmentation of NRC impact area and post processing

After training, we had contour plots generated using SRI_{ff} (Fig. 5 a) and SRI_{hs} (fig. 4. 5 b) as input and achieved preliminary segmentations for all types of congestion impact areas (fig. 4. 5 c) and NRC impact areas (fig. 4. 5 d). Using contour plots generated for the travel route I-95 from north to south on midnight April 1st to noon April

2nd as examples, it can be seen that all preliminary NRC areas from SRI_{hs} contour plot also showed (as whole or part of congestion area) in the corresponding SRI_{ff} contour plot, and the difference between two contour plots were most likely RC areas. Using these results, NRC were preliminarily distinguished from the RC. However, it also was found that some of the preliminary NRC only represented a small area and were found on the upper boundary of congestion areas in the corresponding SRI_{hs} contour plot (e.g., the congestion with x from 200 – 210 and y from 146 – 153 in Fig. 5 a and b), are likely caused by fluctuation of RC areas instead of representing outflows from incidents. Also, although clusters with high SRI in both contour plots were properly identified and segmented by the W-Net based segmentation model, some *salt and pepper* noise were identified as congestion areas with small spatio-temporal coverage. In addition, the shape of part of the preliminary NRC detected did not follow the rule of shockwaves, which meant that further postprocessing was necessary.

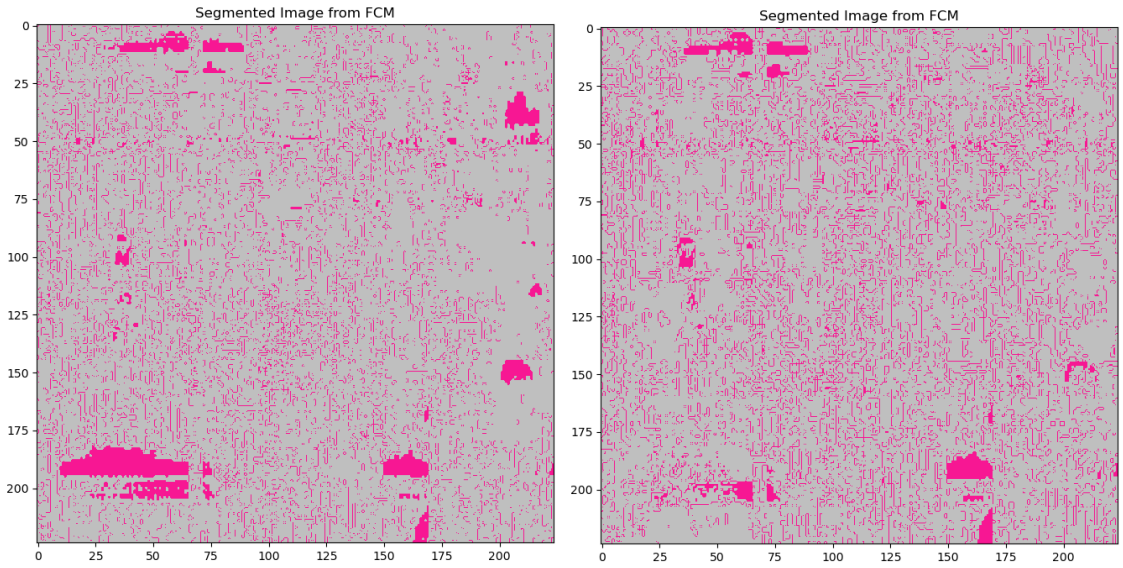
Before the postprocessing, for comparison, we also tested the FCM ++ clustering model using the same contour plots with SRI_{ff} (fig. 4. 5 e) and SRI_{hs} (fig. 4. 5 f) as input and the same parameter as used in previous research as a baseline (m = 2, p = 1.8) (Ou et al., 2020). The FCM ++ model generated lots of data noise in the shape of connected lines, which could be due to FCM considering only pixel-level information and neglecting the context information. Over the segmentation outcomes for all the contour plots that were constructed, around half of the outputs from FCM ++ showed this type of data noise. Therefore, although FCM ++ based models have been shown to be effective for extracting NRC using stationary-based contour plots in previous research, they were

not suitable for NRC impact area estimation using our real-world road network and trajectory data.



(a) *(b)*

(c) *(d)*



(e) *(f)*

Figure 4. 5 For route I-95 from north to south that covered from 12 am, April 1st to around 1 pm, April 2nd (a) SRI_{ff} contour plot, the highlighted cells represent high speed change ratio compared with the free-flow speed, the color bar represent the value of SRI_{ff} indicators; (b) SRI_{hs} contour plot, the highlighted cells represent high speed change ratio compared with the historic average speed, the color bar represent the value of SRI_{hs} indicators; (c) preliminary segmentation results from the W-Net based model, the preliminary congested areas were shown in pink; (c) preliminary segmentation results from the W-Net based model with SRI_{ff} contour plot, where the preliminary congestion impact areas were shown in pink; (d) preliminary segmentation results from the W-Net based model with SRI_{hs} contour plot, the preliminary NRC impact areas were shown in pink; (e) preliminary segmentation results from the FCM ++ with SRI_{ff} contour plot, the preliminary congestion impact areas were shown in pink; (f) preliminary segmentation results from the FCM ++ with SRI_{hs} contour plot, the preliminary NRC impact areas were shown in pink.

Before postprocessing, we first used the Python *scipy* library to label each individual congestion area (fig. 4. 6 a) and preliminary NRC (fig. 4. 6 b) and split the congestion area if they show two or several congestions linked by some data noises. In the next step, we performed strategy f to detect potential preliminary NRC caused by RC fluctuation (fig. 4. 6 c).

Strategy a and b were then performed with θ_{seg} setting as a quarter of the width of the current NRC to achieve the starting road segment and $Cell_{start}$. After $t_{Cellstart}$ and t_{min} were estimated, strategy c was performed to detect preliminary NRC that violate the

shockwave law that assumes NRC propagate along the time flow with σ_{time} setting as a quarter of the total preliminary duration. As the last steps, strategies d and e were performed to fill all the gaps along the NRC propagation along the time flow and counter the traffic flow (fig. 4. 6 d).

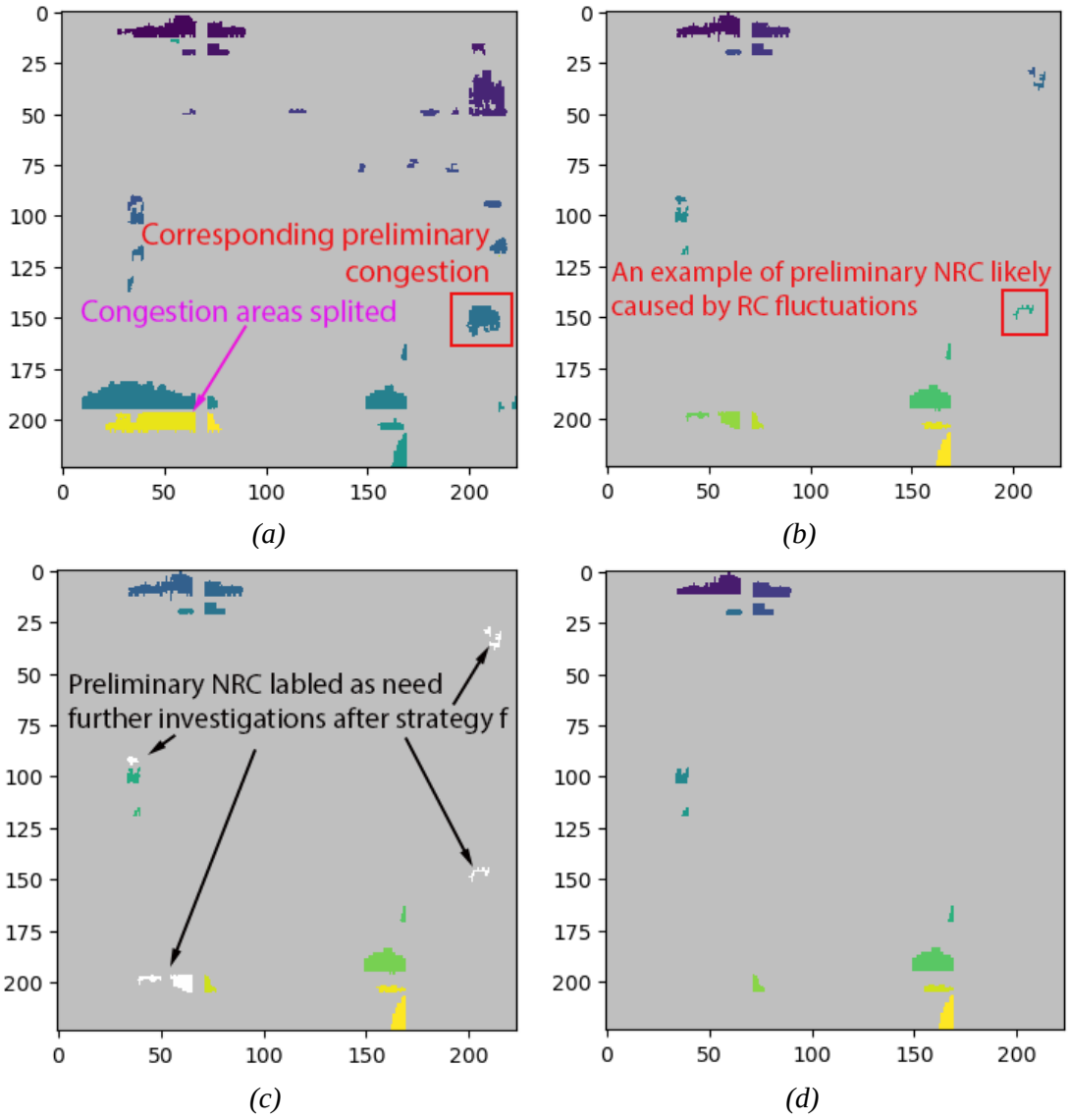


Figure 4. 6 Congestion areas after postprocessing (a) preliminary congestion area identified after splitting, each individual impact areas showed in different color, congestion impact areas are shown in different color; (b) preliminary NRC areas identified after splitting, each individual impact area showed in different color; (c) preliminary NRC labeled as need further investigation are shown in white; (d) final NRC impact areas after the postprocessing

4.6.2 Spatio-temporal propagation of the NRC

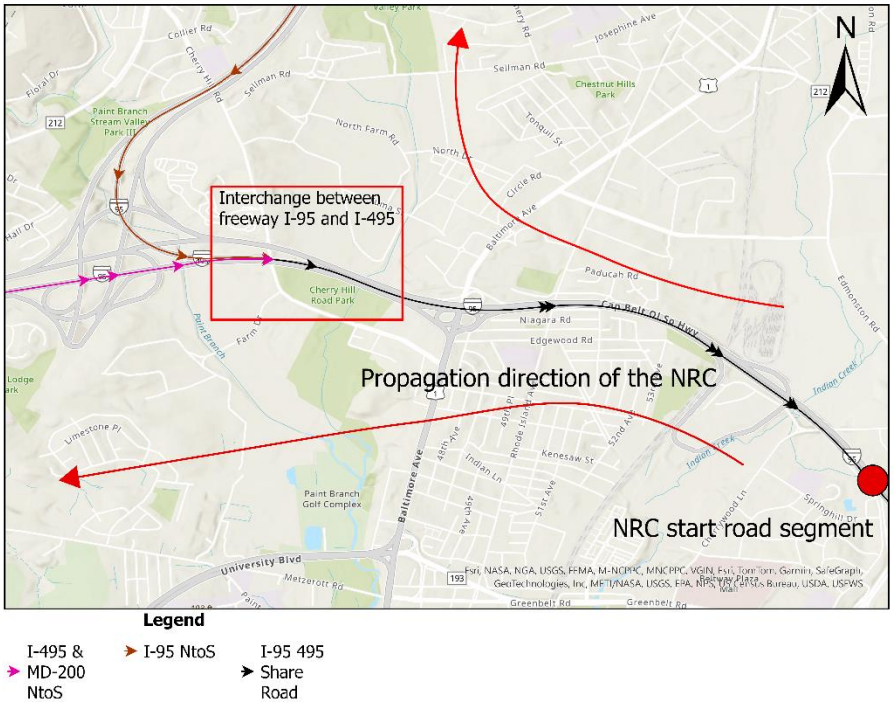
After postprocessing, we identified just less than 17,000 NRC in total for the 6-month period and for the 7 routes (duplicated NRC caused by data duplication at the end of each month were excluded). Approximately 4,500 preliminary NRC were labeled as needing further investigation since they were likely to be associated with traffic fluctuations involved with RC, and just over 6,700 preliminary NRC violated the law of shockwaves. On average, approximately 20 NRC were detected from each SR_{hs} contour plot, with an average temporal duration of NRC2 hours. If fact all routes showed a similar average temporal duration for NRC. This relatively long temporal duration could be due to there being many NRC with shorter time durations (e.g., less than 30 minutes) and where spatial coverage may have been treated as data noise during the segmentation and postprocessing process. From the spatial coverage of NRC, the average distance was 3.4 miles for the maximum length of each NRC, with a range of 2 miles to 4.5 miles (Table 1). In addition, we found that it took around 35 to 45 minutes for NRC to reach their maximum length of their impact area for all the routes.

Table 4. 2 Statistics of NRC impact areas by travel route

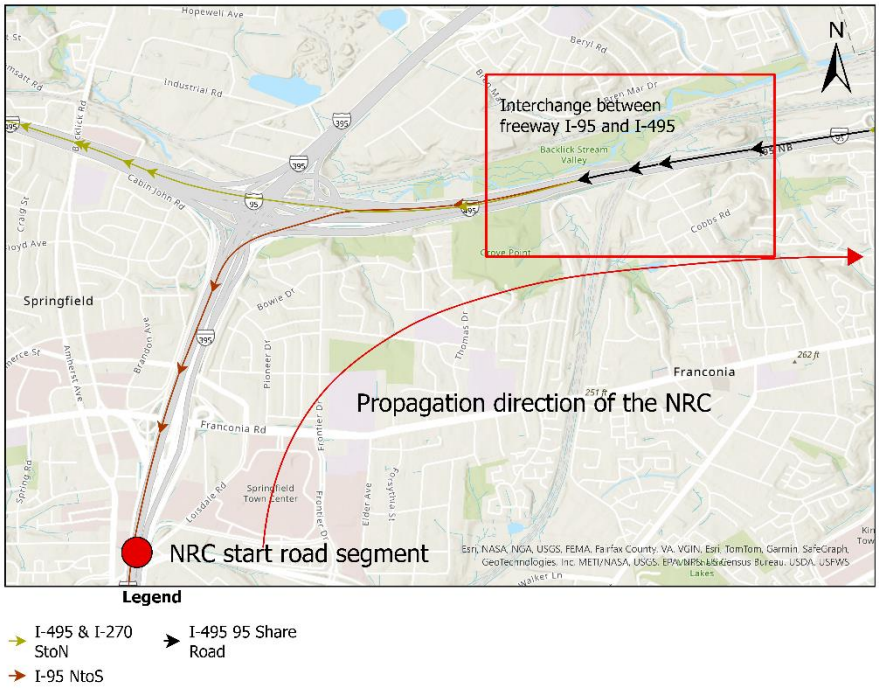
Route Name	Avg no. of segments	Avg length (mile)	Avg temporal duration (minutes)	AVG time reach maximum spatial coverage (minutes)	Total NRC
I-295 & I-895 StoN	7.63	2.08	128.20	36.05	3724
I-495 & MD-200 NtoS	6.16	2.54	128.53	38.50	2536
I-495 & MD-200 StoN	5.87	2.48	131.57	37.21	2707
I-495 & I-270 NtoS	9.38	4.03	124.16	43.97	1574
I-495 & I-270 StoN	9.29	3.78	123.65	43.03	1902

I-95 NtoS	8.79	4.58	120.36	40.16	2229
I-95 StoN	8.59	4.25	123.61	42.74	2224

A novel aspect of this study was that we examined the propagation of NRC in actual complex road networks. We investigated two scenarios, namely how an NRC propagates from one freeway to two connected freeways (i.e., two different NRC shapes were generated) (Fig. 7 a); and how an NRC propagates along one freeway, while another freeway is connected but is not impacted by any NRC (i.e., essentially only one NRC shape was generated) (Fig. 7 b).



(a)



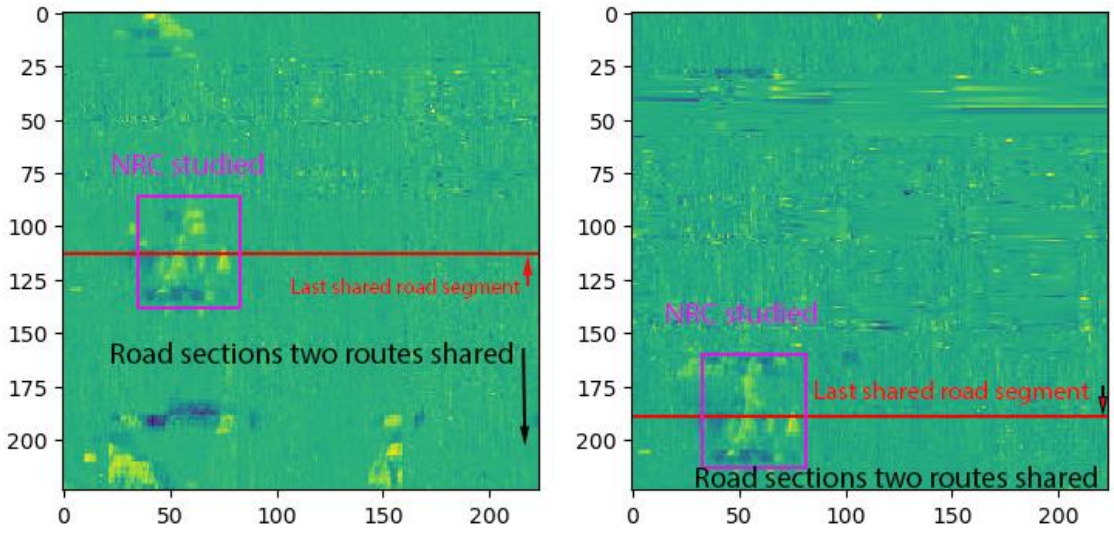
(b)

Figure 4. 7 Road maps of interchanges between I-95 and I-495 freeways, direction of traffic flow is shown as arrows along the road, propagation direction of the NRC are counter to the direction of traffic flow and is shown with red arrows. (a) a scenario showing how an NRC propagates from one freeway to two connected freeways; (b) a scenario where an NRC propagates along a single freeway, although another freeway is connected; NRC starting road segment is estimated based on $Cell_{start}$.

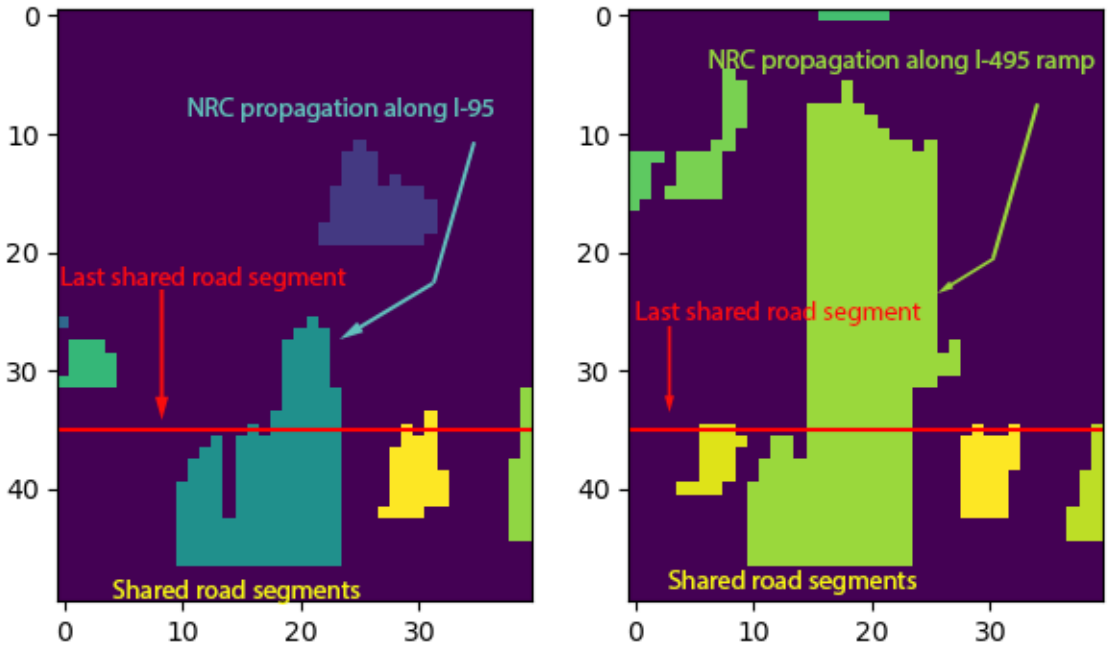
To examine the first case where an NRC propagation from one freeway to two connected freeways, we examined NRC that have spatial coverage on the freeway interchanges (bounded in red in fig. 4. 7 a) in both contour plots generated for both route I-495 & MD-200 N to S (fig. 4. 8 a) and I-95 N to S (fig. 4. 8 b). For road segments after the connection segments (i.e., road segments below the red line in fig. 4. 7 a and b), the contour plots of the two routes shared the same road segments and the detected NRC impact areas were therefore also the same. However, the upper stream of the traffic flow showed the difference between the two routes. From the detected spatio-temporal NRC impact areas, the NRC detected in I-95 showed a larger spatial coverage along the route

counter to the traffic flow and also showed a faster propagation speed. This could be due to the fact that I-95 is connected to the road via a ramp road, and therefore possibly more vulnerable to incidents since it has fewer lanes.

For the second case, we also examined NRC that covered the freeway interchange (bounded in red in fig. 4. 6 b) for both contour plots I-495 & I-270 S to N (fig. 4. 9 a) and I-95 N to S (fig. 4. 9 b). The road before the interchange in the two contour plots shared the same road segments (i.e., road segments above the red line in fig. 4. 9 a and b), and therefore the NRC detected in the common area were the same. However, for the sections in the two contour plots generated using different road segments (i.e., below the red line in fig. 4. 9 c and d) the NRC only showed in the I-95 contour plot since the I-495 (route showed in yellow in fig. 4. 7 b) was not impacted by the NRC since NRC only propagate counter the direction of traffic flow.

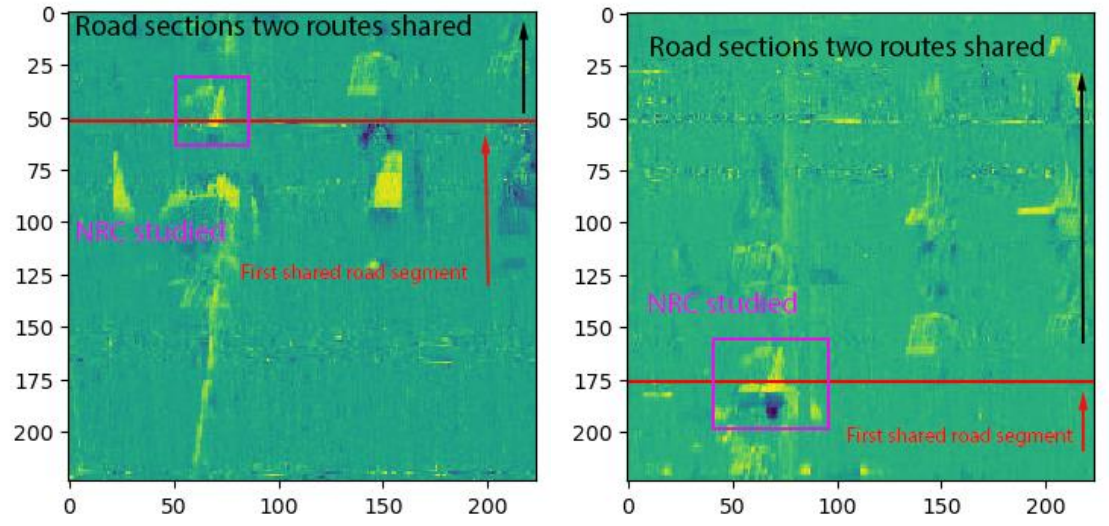


(a) *(b)*



(c) *(d)*

Figure 4. 8 Congestion propagation via interchanges Case 1 (a) SRIhs contour plot for route I-95 from north to south for time started in May 25th; (b) SRIhs contour plot for route I-495 & MD-200 from north to south for time started in May 25th; (c) NRC identified from (a) after post processing that covered the interchange; (d) NRC identified from (b) after post processing that covered the interchange; the red horizontal line in figures represents the road segment at the interchange, all cells below the line are from the same data.



(a) *(b)*

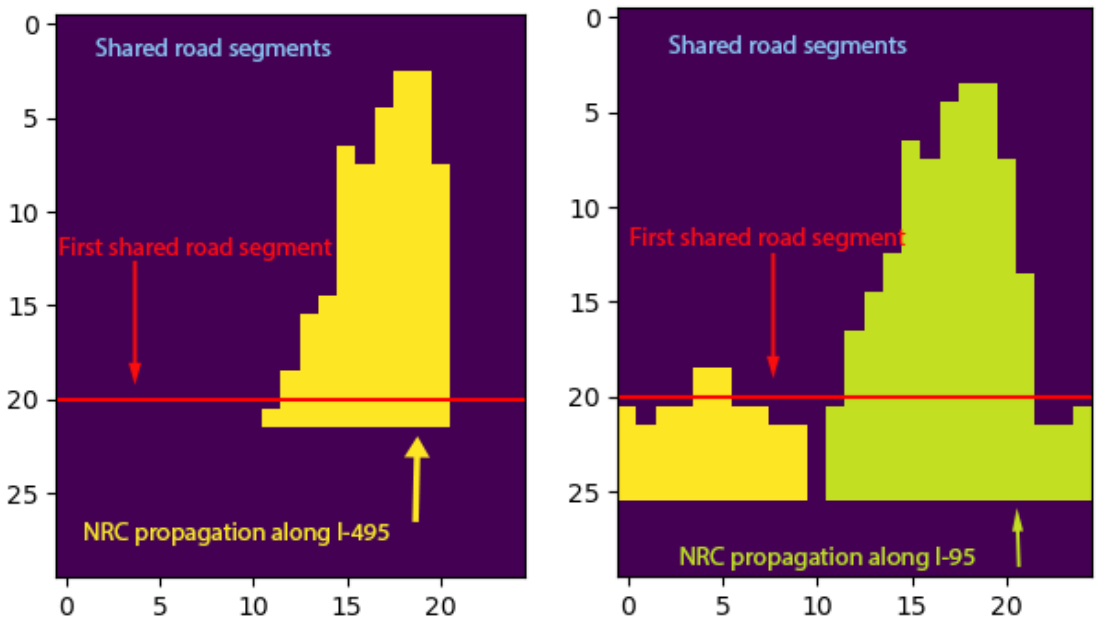


Figure 4. 9 Congestion propagation via interchanges Case 2 (a) SRI_{hs} contour plot for route I-495 & I-270 from south to north for time started in September 25th; (b) SRI_{hs} contour plot for route I-95 from north to south for time started in September 25th; (c) NRC identified from (a) after post processing that covered the interchange; (d) NRC identified from (b) after post processing that covered the interchange; the red horizontal line in figures represents the road segment at the interchange, all cells above the line are from the same data.

4.7 Conclusions and future work

Identification of NRC and modeling how the impacts of these events are propagated along a transportation route is of great importance due to their negative impact on traffic. An important first step is to mitigate the uncertainty these events bring, such as the range of the impact area and the potential duration. In this research, we present a research framework useful for identifying spatio-temporal impact areas of NRC and how these areas are propagated along actual road networks using trajectory data. The framework contains a congestion plot module that generates SRI indicators to measure speed change radios for road segments at given times with their free-flow speed and historic average speed and builds spatio-temporal contour plots based on these indicators. A W-Net-based semantic segmentation module was designed to segment congestion

identified using the contour plots and to overcome the challenges brought by data noise and a lack of ground truth labels. In the last step, a postprocessing module was created to make sure the propagation of NRC follows the law of shockwaves and separate congestion that were not likely caused by incidents for further investigations. Results from the experiments using trajectory data collected for the Washington DC metropolitan area over a 6-month period showed that the research framework could correctly identify the spatio-temporal range of NRC impact areas in actual road networks. The results also examined the different propagation ranges and speeds over different travel routes when NRC propagate from one freeway to another via ramp roads.

The research framework we have designed fills a research gap in identifying and modeling the NRC propagation over complex actual road networks. The framework also properly distinguishes the impacts of RC from NRC and introduces an innovative postprocessing rule that excludes the impact of RC fluctuations on NRC identifications. Our research has laid a foundation for future research on, for example, predicting the duration and length of NRC.

There are some limitations with this research, for example, due to bias in trajectory data coverage, some road segments do not have enough waypoints for speed and SR indicator estimation, especially during the night, and therefore might result in a bias in our NRC findings. In addition, we used some empirical parameters in the postprocessing steps (e.g., defining boundary areas and time thresholds). The selection of these parameters was based on our examination of the contour plots and preliminary NRC detected, but the parameters might cause the spatio-temporal range of some of the NRC to be either over or underestimated. In future work, we plan to introduce

postprocessing rules as constraints instead of empirical rules, which could further increase the generalizability of our research framework. Future work could also consider using graphs to represent spatio-temporal contour plots, which could capture the NRC propagation along the road networks.

Chapter 5: Conclusions

5.1 Review of dissertation

This dissertation comprises three studies that explore how advanced research methods, such as distributed computing and unsupervised deep learning, can be extended and applied to the analysis of human travel behaviors using passively collected trajectory data. Each study develops a research framework to address key challenges, including large data volumes, broad spatio-temporal coverage, and the absence of predefined labels in the trajectory data for the studied travel behaviors. By leveraging the broad spatio-temporal coverage provided by trajectory data, this dissertation examines travel behaviors across large-scale road networks, such as the entire state of California (Chapter 2) and the Washington, DC metropolitan area (Chapters 3 and 4), enhancing the representativeness of the findings concerning mobility. At the same time, the fine granularity of the data allows for detailed, individual-level analyses, for example, the investigation of parking trajectories in Chapter 3, offering insights that are not possible with traditional aggregated data sources such as stationary sensors. The integration of passively collected trajectory data with advanced computational methods improves the interpretability and applicability of the findings by providing large-scale, high-resolution exploration of travel behaviors.

The first study of the dissertation (Chapter 2) developed a distributed framework for traffic speed and speeding estimation across state-wide road networks that encompassed multiple road types using passively collected mobile device data. Leveraging RDDs and SRDDs supported by Spark and Sedona, the framework provided a high computing efficiency for speed and speeding estimates across a broad spatial scale

(e.g., at the scale of large urban agglomerations or at state level) that is better than traditional spatial database management systems, Map Reduce, and modules based only on Spark (i.e., the map matching module is 4-times faster than a map-matching module based on Spark). Using mobile device data passively collected over a 24-hour period on February 1st, 2020, for the State of California, with approximately 126 million waypoints, the study estimated traffic speeds for different road types in California. Incorporating speed limit data from OpenStreetMap (OSM) further allowed for an examination of the spatial heterogeneity of speeding behaviors where road with higher percentage of speeding trips was found. This work not only demonstrated the framework's computational efficiency but also expanded the understanding of traffic dynamics to residential roads, service roads, and ramps, i.e., road types that go beyond freeways.

The second study of the dissertation (Chapter 3) developed a research framework for extracting parking trajectories from massive passively collected trajectory data and distinguishing cruising for parking from direct parking (where vehicles simply drive into an available parking space). It combined a rule-based distributed computing approach using Spark and Sedona for trajectory extraction with an LSTM-autoencoder deep clustering model to differentiate between different types of parking behaviors including cruising and direct parking. The deep clustering model, which processes variable-length driving trajectories and associated features (e.g., driving speed, turning behavior, and proximity to road networks), was able to classify parking trips without relying on predefined labels. Using in-vehicle GPS data collected over one month in July 2018 for the Washington, DC metropolitan area, the framework identified four distinct parking

behavior categories—two corresponding to direct parking and two for cruising. The results revealed that cruising trajectories exhibited longer search durations (e.g., over five minutes of searching), larger spatial ranges, and more frequent turns, offering valuable insights into areas where parking availability may not meet demand.

The third study of the dissertation (Chapter 4) developed a research framework to identify the range of non-recurrent congestion (NRC) impact areas in road networks and model their propagation onto nearby roads using passively collected trajectory data. The framework consisted of a distributed contour plot-building module (designed to capture speed changes compared with free-flow and historical averages), a W-net-based semantic segmentation model to segment the NRC impact areas from these plots, and a postprocessing module to ensure that the propagation patterns follow the law of shockwaves. By analyzing trajectory data collected over six months (April–September 2018) for the Washington, DC metropolitan area, the framework examined NRC propagation along seven routes that included freeways and ramps, effectively distinguishing the overlapping impacts of RC and NRC. This work provided novel insights into how NRC spreads in complex, real-world road networks.

While trajectory data holds much promise, biases associated with trajectory data used in these studies must be acknowledged. One possible source of bias was the non-uniform sampling of the trajectory data used in the dissertation research (S. Hu et al., 2021). The passively collected mobile device data used in Chapter 2 was collected from location-based Apps, but App users may not have been uniformly sampled across all populations, which may have impacted the representativeness of the data for vehicle drivers in the study area. The trajectory data used in Chapters 3 and 4 was collected from

in-vehicle location sensors by INRIX, which has a sampling rate of approximately 2% of the total vehicle trips in the study area and so trips may not have been uniformly sampled. Therefore, the spatio-temporal distribution of travel behaviors (i.e., cruising for parking and NRC) captured and examined in the studies might miss certain groups and their behaviors. Future work could investigate these aspects further to reveal mobility characteristics for understudied groups and locations.

5.2 Significant contributions

Contribution 1: An Apache Spark and Sedona-based computing framework for estimating traffic speed and speeding for a state-wide road network using passively collect mobility data was developed. Taking advantage of RDD and SRDD supported by Spark and Sedona, this framework is suitable for traffic status estimation for road networks at large spatial scales (e.g., state level) and provides fast computing speeds (i.e., the map matching module is 4-times faster than a map-matching module proposed in pervious study based only on Spark).

Contribution 2: The speed estimates and traffic speeding behaviors analyzed in Study 1 provide insights about traffic status on road types other than freeways (e.g. residential roads and service roads), which are generally unavailable or based only on data from traffic monitoring stations. Traveling on these road types are an important component of people's daily travel, which make the examinations of road types such as residential roads of equal importance to freeways, if not perhaps more important.

Contribution 3: A computing framework that comprised a preprocessing model, a distributed parking trajectory extraction model, and an LSTM autoencoder deep

clustering model was proposed to extract and classify unlabeled driving trajectories into different categories of parking behaviors relating to cruising or direct parking. The extraction and classification tasks were based on driving behaviors being automatically retrieved using passively collected data.

Contribution 4: Analysis of spatial and temporal distribution of cruising for parking trajectories in the Washington DC metropolitan area revealed the locations and times of the day that parking supply may not meet demand and offer useful information for local drivers and city planning officials.

Contribution 5: A distributed contour plot building module is designed to first estimate speed reduction index indicators by calculating change of traffic status compared with the free-flow status and historic average status for real-world road networks using passively collected trajectory data. The module then generates the contour plots to represent the indicator over travel routes and time using the index indicators. The designation of contour plots over large-scale complex road networks allows NRC propagation to be analyzed for different routes (e.g., how NRC propagate from one freeway to another via ramps) instead of focusing only on stretches of freeways.

Contribution 6: A W-net based semantic segmentation model is applied to identify congestion impact areas (i.e., a section of the connected road network affected by congestion) from the contour plots and overcome challenges brought by data noise caused by raw trajectory data and traffic fluctuations.

Contribution 7: In the postprocessing module, in addition to making sure the propagation of NRC follows the law of shockwaves, the impact of fluctuations of RC is also excluded to properly distinguished the impact area of NRC from RC.

5.3 Future work

The three studies in the dissertation extended current cutting-edge methods including distributed computing and deep learning, to use with passively collected trajectory data, examining three distinct types of travel behaviors.

While the work in this dissertation used data that had already been collected, a potential future research direction is to predict travel behaviors in real-time, forecasting future states based on past observations. For instance, leveraging the NRC impact areas using the methods described in Chapter 4, a future study could predict the maximum congestion duration and potential expansion areas at earlier stages, enabling more proactive and real-time traffic management strategies.

Another potential direction for future research is to introduce Graph Neural Networks to travel behavior research by treating road networks as graphs and using a set of edges to represent a trip trajectory. By applying graph neural networks, raw trajectories could be transformed and compressed into low-dimensional representation vectors while maintaining the spatio-temporal connective information. For example, a parking process such as that described in Chapter 3 could be represented as a series of connected edges and this would overcome the limitation of varying temporal intervals in the raw trajectory data. Also, for the congestion propagation modeling task in Chapter 4,

the road networks for the study area could be represented as one graph, and propagation among different routes could be captured using this approach.

Bibliography

Afrin, T., & Yodo, N. (2020). A survey of road traffic congestion measures towards a sustainable and resilient transportation system. In *Sustainability (Switzerland)* (Vol. 12, Issue 11). MDPI. https://doi.org/10.3390/su12114660

Agarap, A. F. (2018). Deep learning using rectified linear units (relu). *ArXiv Preprint ArXiv:1803.08375*.

Aji, A., Wang, F., Vo, H., Lee, R., Liu, Q., Zhang, X., & Saltz, J. (2013). *Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce*.

An, S., Yang, H., Wang, J., Cui, N., & Cui, J. (2016). Mining urban recurrent congestion evolution patterns from GPS-equipped vehicle mobility data. *Information Sciences*, *373*, 515–526. https://doi.org/10.1016/j.ins.2016.06.033

Anuradha, J. (2015). A brief introduction on Big Data 5Vs characteristics and Hadoop technology. *Procedia Computer Science*, *48*, 319–324.

Armbrust, M., Xin, R. S., Lian, C., Huai, Y., Liu, D., Bradley, J. K., Meng, X., Kaftan, T., Frankliny, M. J., Ghodsi, A., & Zaharia, M. (2015). Spark SQL: Relational data processing in spark. *Proceedings of the ACM SIGMOD International Conference on Management of Data*, *2015-May*, 1383–1394. https://doi.org/10.1145/2723372.2742797

Axhausen, K. W., & Polak, J. W. (1991). *Choice of parking: Stated preference approach*.

Bagha, H., Choudhury, F. M., & Winter, S. (2023). Autonomous Vehicles Empty Cruising Impact on Parking Dynamics. *Proceedings of the 16th ACM SIGSPATIAL*

International Workshop on Computational Transportation Science, 52–61. https://doi.org/10.1145/3615895.3628171

Bakowski, A., Dekŷŝ, V., Radziszewski, L., Skrobacki, Z., & Świetlik, P. (2018). Estimation of uncertainty and variability of urban traffic volume measurements in Kielce. *11th International Science and Technical Conference Automotive Safety, AUTOMOTIVE SAFETY 2018*, *m*, 1–8. https://doi.org/10.1109/AUTOSAFE.2018.8373314

Baldi, P. (2012). *Autoencoders, Unsupervised Learning, and Deep Architectures* (Vol. 27).

Barbosa, H., Barthelemy, M., Ghoshal, G., James, C. R., Lenormand, M., Louail, T., Menezes, R., Ramasco, J. J., Simini, F., & Tomasini, M. (2018). Human mobility: Models and applications. *Physics Reports*, *734*, 1–74. https://doi.org/10.1016/j.physrep.2018.01.001

Bernstein, D., & Kornhauser, A. (1996). *An introduction to map matching for personal navigation assistants*.

Besse, P. C., Guillouet, B., Loubes, J. M., & Royer, F. (2016). Review and Perspective for Distance-Based Clustering of Vehicle Trajectories. In *IEEE Transactions on Intelligent Transportation Systems* (Vol. 17, Issue 11, pp. 3306–3317). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/TITS.2016.2547641

Cabanas-Tirapu, O., Danús, L., Moro, E., Sales-Pardo, M., & Guimerà, R. (2025). Human mobility is well described by closed-form gravity-like models learned

automatically from data. *Nature Communications*, *16*(1), 1336. https://doi.org/10.1038/s41467-025-56495-5

Cao, J., Menendez, M., & Waraich, R. (2019). Impacts of the urban parking system on cruising traffic and policy development: the case of Zurich downtown area, Switzerland. *Transportation*, *46*(3), 883–908. https://doi.org/10.1007/s11116-017-9832-9

Chai, H., Ma, R., & Zhang, H. M. (2019). Search for parking: A dynamic parking and route guidance system for efficient parking and traffic management. *Journal of Intelligent Transportation Systems*, *23*(6), 541–556. https://doi.org/10.1080/15472450.2018.1488218

Chaniotakis, E., & Pel, A. J. (2015). Drivers' parking location choice under uncertain parking availability and search times: A stated preference experiment. *Transportation Research Part A: Policy and Practice*, *82*, 228–239. https://doi.org/10.1016/j.tra.2015.10.004

Chen, C., Bian, L., & Ma, J. (2014). From traces to trajectories: How well can we guess activity locations from mobile phone traces? *Transportation Research Part C: Emerging Technologies*, *46*, 326–337. https://doi.org/10.1016/j.trc.2014.07.001

Chen, C., Ma, J., Susilo, Y., Liu, Y., & Wang, M. (2016). The promises of big data and small data for travel behavior (aka human mobility) analysis. *Transportation Research Part C: Emerging Technologies*, *68*, 285–299. https://doi.org/10.1016/j.trc.2016.04.005

Chen, C., Zhang, X., Dong, Y., Dong, H., & Rao, F. (2014). Map-matching based on driver behavior model and massive trajectories. *2014 17th IEEE International Conference on Intelligent Transportation Systems, ITSC 2014*, 2817–2822. https://doi.org/10.1109/ITSC.2014.6958141

Chen, Z., Liu, X. C., & Zhang, G. (2016). Non-recurrent congestion analysis using data-driven spatiotemporal approach for information construction. *Transportation Research Part C: Emerging Technologies*, *71*, 19–31. https://doi.org/10.1016/j.trc.2016.07.002

Choong, M. Y., Angeline, L., Chin, R. K. Y., Yeo, K. B., & Teo, K. T. K. (2017). Modeling of vehicle trajectory clustering based on LCSS for traffic pattern extraction. *2017 IEEE 2nd International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, 74–79.

Chung, Y. (2013). Identifying primary and secondary crashes from spatiotemporal crash impact analysis. *Transportation Research Record*, *2386*, 62–71. https://doi.org/10.3141/2386-08

Chung, Y., & Recker, W. W. (2012). A Methodological Approach for Estimating Temporal and Spatial Extent of Delays Caused by Freeway Accidents. *IEEE Transactions on Intelligent Transportation Systems*, *13*(3), 1454–1461. https://doi.org/10.1109/tits.2012.2190282

Dabiri, S., & Heaslip, K. (2018). Inferring transportation modes from GPS trajectories using a convolutional neural network. *Transportation Research Part C: Emerging Technologies*, *86*, 360–371. https://doi.org/10.1016/j.trc.2017.11.021

de Almeida, D. R., de Souza Baptista, C., de Andrade, F. G., & Soares, A. (2020). A survey on big data for trajectory analytics. *ISPRS International Journal of Geo-Information*, *9*(2). https://doi.org/10.3390/ijgi9020088

Eldawy, A., & Mokbel, M. F. (2015). SpatialHadoop: A MapReduce framework for spatial data. *Proceedings - International Conference on Data Engineering*, *2015-May*, 1352–1363. https://doi.org/10.1109/ICDE.2015.7113382

Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd*, *96*(34), 226–231.

Fang, Z., Du, Y., Chen, L., Hu, Y., Gao, Y., & Chen, G. (2021). E2DTC: An end to end deep trajectory clustering framework via self-training. *Proceedings - International Conference on Data Engineering*, *2021-April*, 696–707. https://doi.org/10.1109/ICDE51399.2021.00066

Fan, J., Fu, C., Stewart, K., & Zhang, L. (2019). Using big GPS trajectory data analytics for vehicle miles traveled estimation. *Transportation Research Part C: Emerging Technologies*, *103*(March), 298–307. https://doi.org/10.1016/j.trc.2019.04.019

Federal Highway Administration (FHWA). (2019). *The Urban Congestion Report*.

Graves, A., Fernández, S., & Schmidhuber, J. (2005). Bidirectional LSTM networks for improved phoneme classification and recognition. *International Conference on Artificial Neural Networks*, 799–804.

Guo, X., Zhu, E., Liu, X., & Yin, J. (2018). Deep Embedded Clustering with Data Augmentation. In *Proceedings of Machine Learning Research* (Vol. 95).

Gu, Z., Najmi, A., Saberi, M., Liu, W., & Rashidi, T. H. (2020). Macroscopic parking dynamics modeling and optimal real-time pricing considering cruising-for-parking. *Transportation Research Part C: Emerging Technologies*, *118*(July), 102714. https://doi.org/10.1016/j.trc.2020.102714

Hartigan, J. A., & Wong, M. A. (1979). Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series c (Applied Statistics)*, *28*(1), 100–108.

Herrera, J. C., Work, D. B., Herring, R., Ban, X. (Jeff), Jacobson, Q., & Bayen, A. M. (2010). Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment. *Transportation Research Part C: Emerging Technologies*, *18*(4), 568–583. https://doi.org/https://doi.org/10.1016/j.trc.2009.10.006

Hochmair, H. H. (2016). Spatiotemporal pattern analysis of taxi trips in New York city. *Transportation Research Record*, *2542*, 45–56. https://doi.org/10.3141/2542-06

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Hong, J., Shen, Q., & Zhang, L. (2014). How do built-environment factors affect travel behavior? A spatial analysis at different geographic scales. *Transportation*, *41*(3), 419–440. https://doi.org/10.1007/s11116-013-9462-9

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, *79*(8), 2554–2558.

Hsueh, Y. L., & Chen, H. C. (2018). Map matching for low-sampling-rate GPS trajectories by exploring real-time moving directions. *Information Sciences*, *433–434*, 55–69. https://doi.org/10.1016/j.ins.2017.12.031

Huang, X., Lu, J., Gao, S., Wang, S., Liu, Z., & Wei, H. (2022). Staying at Home Is a Privilege: Evidence from Fine-Grained Mobile Phone Location Data in the United States during the COVID-19 Pandemic. *Annals of the American Association of Geographers*, *112*(1), 286–305. https://doi.org/10.1080/24694452.2021.1904819

Huang, Y., Sun, D., & Tang, J. (2018). Taxi driver speeding: Who, when, where and how? A comparative study between Shanghai and New York City. *Traffic Injury Prevention*, *19*(3), 311–316.

Huang, Z., Qiao, S., Han, N., Yuan, C., Song, X., & Xiao, Y. (2021). Survey on vehicle map matching techniques. *CAAI Transactions on Intelligence Technology*, *6*(1), 55–71.

Hughes, J. N., Annex, A., Eichelberger, C. N., Fox, A., Hulbert, A., & Ronquest, M. (2015). Geomesa: a distributed architecture for spatio-temporal fusion. *Geospatial Informatics, Fusion, and Motion Video Analytics V*, *9473*, 128–140.

Hu, S., Xiong, C., Yang, M., Younes, H., Luo, W., & Zhang, L. (2021). A big-data driven approach to analyzing and modeling human mobility trend under non-pharmaceutical interventions during COVID-19 pandemic. *Transportation Research Part C: Emerging Technologies*, *124*, 102955.

Hu, T., Wang, S., She, B., Zhang, M., Huang, X., Cui, Y., Khuri, J., Hu, Y., Fu, X., Wang, X., Wang, P., Zhu, X., Bao, S., Guan, W., & Li, Z. (2021). Human mobility

data in the COVID-19 pandemic: characteristics, applications, and challenges. *International Journal of Digital Earth*, *14*(9), 1126–1147. https://doi.org/10.1080/17538947.2021.1952324

Jiang, Y., Gu, X., Wu, D., Hang, W., Xue, J., Qiu, S., & Lin, C. T. (2021). A Novel Negative-Transfer-Resistant Fuzzy Clustering Model with a Shared Cross-Domain Transfer Latent Space and its Application to Brain CT Image Segmentation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, *18*(1), 40–52. https://doi.org/10.1109/TCBB.2019.2963873

Ju, W., Li, J., Yu, W., & Zhang, R. (2016). iGraph: an incremental data processing system for dynamic graph. *Frontiers of Computer Science*, *10*(3), 462–476. https://doi.org/10.1007/s11704-016-5485-7

Kan, Z., Tang, L., Kwan, M. P., Ren, C., Liu, D., & Li, Q. (2019). Traffic congestion analysis at the turn level using Taxis' GPS trajectory data. *Computers, Environment and Urban Systems*, *74*(November 2018), 229–243. https://doi.org/10.1016/j.compenvurbsys.2018.11.007

Karlaftis, M. G., Latoski, S. P., Richards, N. J., & Sinha, K. C. (1999). ITS impacts on safety and traffic management: an investigation of secondary crash causes. *Journal of Intelligent Transportation Systems*, *5*(1), 39–52.

Kaur, R., Karmakar, G., & Imran, M. (2023). Impact of Traditional and Embedded Image Denoising on CNN-Based Deep Learning. *Applied Sciences (Switzerland)*, *13*(20). https://doi.org/10.3390/app132011560

Ke, R., Li, W., Cui, Z., & Wang, Y. (2020). Two-stream multi-channel convolutional neural network for multi-lane traffic speed prediction considering traffic volume impact. *Transportation Research Record*, *2674*(4), 459–470.

Key:highway. (n.d.). Https://Wiki.Openstreetmap.Org/Wiki/Key:Highway.

Khanna, A., & Anand, R. (2016). IoT based smart parking system. *2016 International Conference on Internet of Things and Applications (IOTA)*, 266–270.

Khattak, A., Wang, X., & Zhang, H. (2009). Are incident durations and secondary incidents interdependent? *Transportation Research Record*, *2099*(1), 39–49.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *ArXiv Preprint ArXiv:1412.6980*.

Klinger, T., & Lanzendorf, M. (2016). Moving between mobility cultures: what affects the travel behavior of new residents? *Transportation*, *43*(2), 243–271. https://doi.org/10.1007/s11116-014-9574-x

Kullback, S., & Leibler, R. A. (1951). On Information and Sufficiency. In *Source: The Annals of Mathematical Statistics* (Vol. 22, Issue 1).

Kwon, J., Varaiya, P., & Skabardonis, A. (2003). Estimation of Truck Traffic Volume from Single Loop Detectors with Lane-to-Lane Speed Correlation. *Transportation Research Record*, *1856*, 106–117. https://doi.org/10.3141/1856-11

Lave, C. A. (1985). Speeding, coordination, and the 55 mph limit. *The American Economic Review*, *75*(5), 1159–1164.

Lee, J. B., Agdas, D., & Baker, D. (2017). Cruising for parking: New empirical evidence and influential factors on cruising time. *Journal of Transport and Land Use*, *10*(1), 931–943. https://doi.org/10.5198/jtlu.2017.1142

Li, M., Gao, S., Lu, F., & Zhang, H. (2019). Reconstruction of human movement trajectories from large-scale low-frequency mobile phone data. *Computers, Environment and Urban Systems*, *77*(January), 101346. https://doi.org/10.1016/j.compenvurbsys.2019.101346

Lin, Y., Wang, P., & Ma, M. (2017). Intelligent transportation system (ITS): Concept, challenge and opportunity. *2017 Ieee 3rd International Conference on Big Data Security on Cloud (Bigdatasecurity), Ieee International Conference on High Performance and Smart Computing (Hpsc), and Ieee International Conference on Intelligent Data and Security (Ids)*, 167–172.

Li, X., Zhao, K., Cong, G., Jensen, C. S., & Wei, W. (2018a). Deep representation learning for trajectory similarity computation. *Proceedings - IEEE 34th International Conference on Data Engineering, ICDE 2018*, 617–628. https://doi.org/10.1109/ICDE.2018.00062

Li, X., Zhao, K., Cong, G., Jensen, C. S., & Wei, W. (2018b). Deep representation learning for trajectory similarity computation. *Proceedings - IEEE 34th International Conference on Data Engineering, ICDE 2018*, 617–628. https://doi.org/10.1109/ICDE.2018.00062

Lu, E. H. C., & Liao, C. H. (2020). Prediction-based parking allocation framework in urban environments. *International Journal of Geographical Information Science*, *34*(9), 1873–1901. https://doi.org/10.1080/13658816.2020.1721503

Merry, K., & Bettinger, P. (2019). Smartphone GPS accuracy study in an urban environment. *PLoS ONE*, *14*(7). https://doi.org/10.1371/journal.pone.0219890

Millard-Ball, A., Hampshire, R. C., & Weinberger, R. (2020). Parking behaviour: The curious lack of cruising for parking in San Francisco. *Land Use Policy*, *91*. https://doi.org/10.1016/j.landusepol.2019.03.031

Millard-Ball, A., Weinberger, R. R., & Hampshire, R. C. (2014). Is the curb 80% full or 20% empty? Assessing the impacts of San Francisco's parking pricing experiment. *Transportation Research Part A: Policy and Practice*, *63*, 76–92. https://doi.org/10.1016/j.tra.2014.02.016

Moore, J. E., Giuliano, G., & Cho, S. (2004). Secondary accident rates on Los Angeles freeways. *Journal of Transportation Engineering*, *130*(3), 280–285.

Morris, B., & Trivedi, M. (2009). Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 312–319.

OpenStreetMap contributors. (2020). *Planet dump retrieved from https://planet.osm.org*. Https://Www.Openstreetmap.Org.

Ou, J., Xia, J., Wang, Y., Wang, C., & Lu, Z. (2020). A data-driven approach to determining freeway incident impact areas with fuzzy and graph theory-based

clustering. *Computer-Aided Civil and Infrastructure Engineering*, *35*(2), 178–199. https://doi.org/10.1111/mice.12484

Park, S. H., Kim, B., Kang, C. M., Chung, C. C., & Choi, J. W. (2018). Sequence-to-sequence prediction of vehicle trajectory via LSTM encoder-decoder architecture. *2018 IEEE Intelligent Vehicles Symposium (IV)*, 1672–1678.

Petitjean, F., Forestier, G., Webb, G. I., Nicholson, A. E., Chen, Y., & Keogh, E. (2014). Dynamic Time Warping Averaging of Time Series Allows Faster and More Accurate Classification. *Proceedings - IEEE International Conference on Data Mining, ICDM*, *2015-January*(January), 470–479. https://doi.org/10.1109/ICDM.2014.27

Petitjean, F., Ketterlin, A., & Gançarski, P. (2011). A global averaging method for dynamic time warping, with applications to clustering. *Pattern Recognition*, *44*(3), 678–693.

Raub, R. A. (1997). Occurrence of secondary crashes on urban arterial roadways. *Transportation Research Record*, *1581*(1), 53–58.

Robinson, J. T. (1981). The KDB-tree: a search structure for large multidimensional dynamic indexes. *Proceedings of the 1981 ACM SIGMOD International Conference on Management of Data*, 10–18.

Roy, S. sen, Perlman, J., & Balling, R. C. (2020). Analysis of urban mobility in South Florida using Uber Movement. *Case Studies on Transport Policy*, *8*(4), 1393–1400. https://doi.org/10.1016/j.cstp.2020.10.003

Sarker, A. A., Naimi, A., Mishra, S., Golias, M. M., & Freeze, P. B. (2015). Development of a secondary crash identification algorithm and occurrence pattern determination in large scale multi-facility transportation network. *Transportation Research Part C: Emerging Technologies*, *60*, 142–160.

Seo, T., Bayen, A. M., Kusakabe, T., & Asakura, Y. (2017). Traffic state estimation on highway: A comprehensive survey. *Annual Reviews in Control*, *43*, 128–151. https://doi.org/10.1016/j.arcontrol.2017.03.005

Shoup, D. C. (2006). Cruising for parking. *Transport Policy*, *13*(6), 479–486. https://doi.org/10.1016/j.tranpol.2006.05.005

Stetco, A., Zeng, X.-J., & Keane, J. (2015). Fuzzy C-means++: Fuzzy C-means with effective seeding initialization. *Expert Systems with Applications*, *42*(21), 7541–7548.

Stewart, K., Fan, J., Schwarz, C., & McGehee, D. V. (2018). Geospatial analysis of residential parking behaviors using a semantic modeling approach. *Travel Behaviour and Society*, *11*, 9–20. https://doi.org/10.1016/j.tbs.2017.12.004

Student, B. (n.d.). *BIOMETKIKA. THE PKOBABLE EKBOB OF A MEAN*. https://academic.oup.com/biomet/article/6/1/1/225634

Sun, C. C., & Chilukuri, V. (2010). Dynamic incident progression curve for classifying secondary traffic crashes. *Journal of Transportation Engineering*, *136*(12), 1153–1158.

Sun, Y., Ren, Y., & Sun, X. (2020). Uber movement data: A proxy for average one-way commuting times by car. *ISPRS International Journal of Geo-Information*, *9*(3). https://doi.org/10.3390/ijgi9030184

van der Waerden, P., Timmermans, H., & Van Hove, L. (2015). GPS data and car drivers' parking search behavior in the City of Turnhout, Belgium. *Lecture Notes in Geoinformation and Cartography*, *214*, 247–256. https://doi.org/10.1007/978-3-319-11463-7_18

Van Ommeren, J. N., Wentink, D., & Rietveld, P. (2012). Empirical evidence on cruising for parking. *Transportation Research Part A: Policy and Practice*, *46*(1), 123–130. https://doi.org/10.1016/j.tra.2011.09.011

Vlachos, M., Kollios, G., & Gunopulos, D. (2002). Discovering similar multidimensional trajectories. *Proceedings 18th International Conference on Data Engineering*, 673–684.

Wang, C., Huang, J., Wang, Y., Lin, Z., Jin, X., Jin, X., Weng, D., & Wu, Y. (2024). A Deep Spatiotemporal Trajectory Representation Learning Framework for Clustering. *IEEE Transactions on Intelligent Transportation Systems*, *25*(7), 7687–7700. https://doi.org/10.1109/TITS.2024.3350339

Wang, J., & Hu, Y. (2024). Unraveling hurricane Ian's Impact: A multiscale analysis of mobility networks in Florida. *Transportation Research Part D: Transport and Environment*. https://doi.org/10.1016/j.trd.2024.104482

Wang, S., Lu, C., Liu, C., Zhou, Y., Bi, J., & Zhao, X. (2020). Understanding the Energy Consumption of Battery Electric Buses in Urban Public Transport Systems. *Sustainability*, *12*(23), 10007.

Wang, Y., & Nihan, N. L. (2000). Freeway traffic speed estimation with single-loop outputs. *Transportation Research Record*, *1727*(1), 120–126.

Wang, Z., & Jiang, H. (2020). Identifying Secondary Crashes on Freeways by Leveraging the Spatiotemporal Evolution of Shockwaves in the Speed Contour Plot. *Journal of Transportation Engineering, Part A: Systems*, *146*(2), 04019072.

Wang, Z., Zheng, Z., Chen, X., Ma, W., & Yang, H. (2024). Modeling the evolution of incident impact in urban road networks by leveraging the spatiotemporal propagation of shockwaves. *Transportation Research Part C: Emerging Technologies*, *164*. https://doi.org/10.1016/j.trc.2024.104668

Weinberger, R. R., Millard-Ball, A., & Hampshire, R. C. (2020). Parking search caused congestion: Where's all the fuss? *Transportation Research Part C: Emerging Technologies*, *120*(July), 102781. https://doi.org/10.1016/j.trc.2020.102781

White, C. E., Bernstein, D., & Kornhauser, A. L. (2000). Some map matching algorithms for personal navigation assistants. *Transportation Research Part C: Emerging Technologies*, *8*(1), 91–108. https://doi.org/https://doi.org/10.1016/S0968-090X(00)00026-7

Wu, H. (2018). Comparing Google Maps and Uber Movement Travel Time Data. *Transport Findings*. https://doi.org/10.32866/5115

Xia, X., & Kulis, B. (2017). W-net: A deep model for fully unsupervised image segmentation. *ArXiv Preprint ArXiv:1711.08506*.

Xie, J., Girshick, R., & Farhadi, A. (2016). *Unsupervised Deep Embedding for Clustering Analysis*. https://github.com/piiswrong/dec.

Xiong, C., Hu, S., Yang, M., Luo, W., & Zhang, L. (2020). Mobile device data reveal the dynamics in a positive relationship between human mobility and COVID-19 infections. *Proceedings of the National Academy of Sciences of the United States of America*, *117*(44), 27087–27089. https://doi.org/10.1073/pnas.2010836117

Yabe, T., Rao, P. S. C., & Ukkusuri, S. V. (2021). Resilience of Interdependent Urban Socio-Physical Systems using Large-Scale Mobility Data: Modeling Recovery Dynamics. *Sustainable Cities and Society*, *75*. https://doi.org/10.1016/j.scs.2021.103237

Yang, H., Wang, Z., Xie, K., Ozbay, K., & Imprialou, M. (2018). Methodological evolution and frontiers of identifying, modeling and preventing secondary crashes on highways. *Accident Analysis and Prevention*, *117*, 40–54. https://doi.org/10.1016/j.aap.2018.04.001

Yang, X., Stewart, K., Tang, L., Xie, Z., & Li, Q. (2018). A review of GPS trajectories classification based on transportation mode. In *Sensors (Switzerland)* (Vol. 18, Issue 11). MDPI AG. https://doi.org/10.3390/s18113741

Yao, D., Zhang, C., Zhu, Z., Huang, J., & Bi, J. (2017). Trajectory clustering via deep representation learning. *2017 International Joint Conference on Neural Networks (IJCNN)*, 3880–3887.

Yi, B.-K., Jagadish, H. V, & Faloutsos, C. (1998). Efficient retrieval of similar time sequences under time warping. *Proceedings 14th International Conference on Data Engineering*, 201–208.

Yokoo, T., & Levinson, D. (2019). Measures of speeding from a GPS-based travel behavior survey. *Traffic Injury Prevention*, *20*(2), 158–163. https://doi.org/10.1080/15389588.2018.1543873

You, S., Zhang, J., & Gruenwald, L. (2015). Large-scale spatial join query processing in Cloud. *Proceedings - International Conference on Data Engineering*, *2015-June*, 34–41. https://doi.org/10.1109/ICDEW.2015.7129541

Yue, M., Li, Y., Yang, H., Ahuja, R., Chiang, Y.-Y., & Shahabi, C. (2019). Detect: Deep trajectory clustering for mobility-behavior analysis. *2019 IEEE International Conference on Big Data (Big Data)*, 988–997.

Yu, J. J. Q., & Gu, J. (2019). Real-Time Traffic Speed Estimation with Graph Convolutional Generative Autoencoder. *IEEE Transactions on Intelligent Transportation Systems*, *20*(10), 3940–3951. https://doi.org/10.1109/TITS.2019.2910560

Yu, J., Stettler, M. E. J., Angeloudis, P., Hu, S., & Chen, X. (Michael). (2020). Urban network-wide traffic speed estimation with massive ride-sourcing GPS traces. *Transportation Research Part C: Emerging Technologies*, *112*(May 2019), 136–152. https://doi.org/10.1016/j.trc.2020.01.023

Yu, J., Wu, J., & Sarwat, M. (2015). Geospark: A cluster computing framework for processing large-scale spatial data. *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 1–4.

Yu, J., Zhang, Z., & Sarwat, M. (2019a). Spatial data management in apache spark: the GeoSpark perspective and beyond. *GeoInformatica*, *23*(1), 37–78. https://doi.org/10.1007/s10707-018-0330-9

Yu, J., Zhang, Z., & Sarwat, M. (2019b). Spatial data management in apache spark: the GeoSpark perspective and beyond. *GeoInformatica*, *23*(1), 37–78. https://doi.org/10.1007/s10707-018-0330-9

Zaharia, M., Chowdhury, M., Franklin, M. J., Shenker, S., & Stoica, I. (2010). Spark: Cluster computing with working sets. *2nd USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 10).*

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., & Franklin, M. J. (2016). Apache spark: a unified engine for big data processing. *Communications of the ACM*, *59*(11), 56–65.

Zhang, J., He, S., Wang, W., & Zhan, F. (2015). Accuracy Analysis of Freeway Traffic Speed Estimation Based on the Integration of Cellular Probe System and Loop Detectors. *Journal of Intelligent Transportation Systems: Technology, Planning, and Operations*, *19*(4), 411–426. https://doi.org/10.1080/15472450.2014.1000456

Zhang, P., Stewart, K., & Li, Y. (2023). Estimating traffic speed and speeding using passively collected big mobility data and a distributed computing framework. *Transactions in GIS*, *27*(4), 1124–1144. https://doi.org/10.1111/tgis.13061

Zhao, J., Gao, Y., Yang, Z., Li, J., Feng, Y., Qin, Z., & Bai, Z. (2019). Truck traffic speed prediction under non-recurrent congestion: Based on optimized deep learning algorithms and GPS data. *IEEE Access*, *7*, 9116–9127.

Zheng, D., Chitturi, M. V, Bill, A. R., & Noyce, D. A. (2014). Secondary crash identification on a large-scale highway system. *Transportation Research Board 93rd Annual Meeting*.

Zheng, Y. (2015). Trajectory data mining: An overview. *ACM Transactions on Intelligent Systems and Technology*, *6*(3), 1–41. https://doi.org/10.1145/2743025

Zheng, Z., Wang, Z., Chen, X., Ma, W., & Ran, B. (2023). Spatiotemporal clustering for the impact region caused by a traffic incident: an improved fuzzy C-means approach with guaranteed consistency. *Transportmetrica A: Transport Science*. https://doi.org/10.1080/23249935.2023.2236719

Zhu, Y., Liu, Y., Yu, J. J. Q., & Yuan, X. (2022). Semi-Supervised Federated Learning for Travel Mode Identification from GPS Trajectories. *IEEE Transactions on Intelligent Transportation Systems*, *23*(3), 2380–2391. https://doi.org/10.1109/TITS.2021.3092015