DEVELOPMENT OF A REAL-TIME CONNECTED CORRIDOR DATA-DRIVEN DIGITAL TWIN AND DATA IMPUTATION METHODS

A Dissertation
Presented to
The Academic Faculty

by

Abhilasha Jairam Saroj

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
Civil and Environmental Engineering

Georgia Institute of Technology
August 2020

DEVELOPMENT OF A REAL-TIME CONNECTED CORRDOR DATA-DRIVEN DIGITAL TWIN AND DATA IMPUTATION METHODS

Approved by:

Dr. Michael Hunter, Advisor
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Michael O. Rodgers
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Angshuman Guin
School of Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Richard Fujimoto
School of Computer Science and Engineering
Georgia Institute of Technology

Dr. Randall Guensler
School of Civil and Environmental Engineering
Georgia Institute of Technology

Date Approved: July 7, 2020

ACKNOWLEDGEMENTS

I would like to thank my professors, colleagues, family, and friends, without whom I would have not been able to complete this research. I am deeply thankful to my adviser Dr. Michael Hunter, who invested time, attention, and care to listen, provide feedback, and insights, from which I have learned a lot. I am incredibly thankful for all the opportunities and mentoring I received from him. His encouragement and drive for pursuit of excellence motivated me to push my limits on a daily basis. You provided an environment that encouraged freedom in research, is accepting of mistakes and failures and focuses on moving forward, for which I will be always grateful. Your advice on professional and personal front, had an invaluable role in building my persistence during the Ph.D. program. I will strive to keep the lessons and values I learned from you in future.

I am deeply thankful to Dr. Angshuman Guin for the guidance in several research projects and for the research opportunities provided to me. Your several practical insights and recommendations helped me to improve the quality of research product. I am deeply grateful to Dr. Michael Rodgers and Dr. Randall Guensler for providing me opportunities, knowledge and insight needed to broaden my horizon to think holistically about the impacts on the transportation system. I also want to thank you for your insights on MOVES emissions model, which allows potential feasibility for the developed model to study environmental impacts of traffic. I am deeply thankful to Dr. Richard Fujimoto for agreeing to be on my dissertation committee and pushing me to think about the impacts and contribution of the research from a practical viewpoint.

I am deeply thankful to my fellow Georgia Tech colleagues – Somdut Roy, Nishu Choudhary, Han Gyol Kim, Dr. Daejin Kim, Yingping Zhao, Ron Boodhoo, Dr. Xiaodan Xu, and Atiyya Shaw for their support, advice, and friendship during this journey. I want to thank wonderful friends – Marisha Pena, Maria Palmer, Sonal Jain, Nikhil Raj, Deepak Chaudhary, Mrunal Dehankar, Tanvi Singh, Neeraj Rao, Tejas Khandekar, Visak Kumar, Jackie Boodhoo, Kriti Bhargava, Harshit Banthia, and several others, who provided a slice of home when I missed it. I would like to express my deep gratitude to longtime friends back home for their unwavering encouragement and support – Suhani Mohan and Priyanka Nayak.

I am deeply grateful to my father – Jairam Shree Saroj, who was determined to provide my brother and me education, and made several sacrifices towards it, to do best in his capacity. I am thankful for your strong encouragement towards pursuing STEM since young age and your belief in me to achieve educational goals. I am deeply grateful to my sister-in-law – Dhuguru Saritha Saroj for her love and support during this journey. I am very thankful to my brother – Amar Kumar Saroj, for his deep understanding and support towards my pursuits.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ABBREVIATIONS

ITS	Intelligent Transportation Systems
CV	Connected Vehicle
CAV	Connected and Autonomous Vehicle
V2V	Vehicle-to-Vehicle
V2I	Vehicle-to-Infrastructure
GHG	Green House Gas
GDP	Gross Domestic Product
BSM	Basic Safety Message
SPaT	Signal Phasing and Timing
DSRC	Dedicated Short-Range Communications
OBU	On-board unit
USDOT	United States Department of Transportation
FHWA	Federal Highway Administration
COM	Component Object Module
DTW	Dynamic Time Warp
HAC	Hierarchical Agglomerative Clustering
LR	Learning Rate
SGD	Stochastic Gradient Descent
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
TT	Travel Time

EB	Eastbound
WB	Westbound
NB	Northbound
SB	Southbound
RBC	Ring Barrier Controller
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
ARIMA	Auto-regressive Integrated Moving Average
PCA	Principal Component Analysis
MOVES	Motor Vehicle Emissions Simulator
USEPA	United States Environmental Protection Agency
SARIMA	Seasonal Auto-regressive Integrated Moving Average

SUMMARY

Smart city "Connected Corridor" initiatives are a reality taking place in cities across the world. A connected corridor is expected to be deployed with sensors and intelligent transportation systems that enable communication between infrastructure and vehicles. The presented research aims to advance the understanding of leveraging high frequency traffic data streams from connected corridors to provide meaningful insights on the traffic and environmental performance of the corridor. The overarching goals of this dissertation are 1) to develop a connected corridor real-time data-driven simulation model capable of generating traffic and performance measures at a (near) real-time rate, 2) to investigate data gap patterns in the real-time volume data streams, 3) to measure sensitivity of volume data imputations on simulation model generated performance measures, and 4) to investigate a suitable volume data imputation methodology. Such a simulation tool can be used for testing and studying the traffic and environmental impacts of the connected technologies being deployed on a smart corridor.

The built simulation model is driven using real-time high frequency volume count and signal state data streams from a connected corridor testbed. The model architecture performs three primary tasks dynamically: 1) Raw Data Stream Processing, 2) Dynamic Data-Driven Traffic Simulation, 3) Dynamic Performance Measures Estimation and Visualization, and 4) Data Request Management. The traffic simulation model is built in PTV Vissim 9.00-08 simulation software and is enabled to take the real-time data as inputs, emulate the input data information in simulation runtime, and produce results for the

simulated vehicles dynamically, using Vissim's COM module and python programming language.

In testing of the simulation architecture, the real-time data streams are investigated. Significant presence of data gaps are found in both volume and signal data streams. Thus, this effort also focused on studying the impact of volume data imputation on simulation generated performance measures. A sensitivity analysis of volume data imputation on simulation generated travel-time is conducted. The data imputation approach initially tested in this study is a simple replacement approach. The results suggest a higher sensitivity of the model to a subset of intersection approaches, revealing a methodology to prioritize approaches most critical to such real-time applications. In addition, the sensitivity analysis results indicated a need to develop a more accurate data imputation methodology.

Thus, the final aspect of this dissertation included the development and performance investigation of an imputation model with Long Short Term Memory (LSTM) Recurrent Neural Network (RNN) layers. Univariate time series and multivariate time series LSTM RNN predictive traffic volume models are developed. Experiments are conducted to investigate the performance of the univariate and multivariate models to provide imputations for typical day traffic and atypical day traffic. Key results indicate 1) the potential advantages of using a multivariate model over an univariate model for predicting longer consecutive units, 2) the multivariate models superior performance over the univariate model in obtaining volume imputations for atypical traffic, and 3) the better performance of the multivariate model over the univariate model in providing accurate simulation generated travel times.

CHAPTER 1. INTRODUCTION

During the last two decades, city populations have grown sharply across the world. The World Bank estimates that the world urban population has exceeded the rural population since 2007 [1]. In 2017, the total number of people living in urban areas exceeded that of rural areas by 21% [1]. The 2018 Revision of World Urbanization Prospects by United Nations Department of Economic and Social Affairs estimates an increase on the order of 28% in the global population by 2050 [2]. This rise is expected to contribute to the increase in the global urban population. By 2050 the urban population is projected to be double that of rural [2]. The primary reason for the rise in urbanization is migration from rural areas, from poor countries or from countries under military conflicts, to gain access to better job opportunities, education, healthcare, etc. [3] A well-managed city fosters an environment that encourages economic activity, enables innovation, creates employment, encourages social tolerance, and contributes to world GDP. However, if not well managed, cities can contribute to economic and social problems such as environmental pollution, congestion, urban poverty, etc. [4] To function, a city intakes resources and produces waste. To be able to cater to needs of rising urbanization it's crucial to adapt resource-efficient, innovative, and sustainable city management practices [4-6]. One such innovative perspective – "Smart Cities" – focuses on the idea of leveraging technology to advance urban services to be more resource-efficient and sustainable [5, 7]. This thesis 1) focuses on development of a real-time data-driven simulation model of a Smart City Connected Corridor to provide performance evaluations, 2) explores the impact of data loss

on simulation results, and 3) investigates application of deep learning algorithm for connected corridor data imputations.

This includes development of methodology for real-time modeling and performance prediction of a smart corridor and as well as exploring the impact of data loss and methods of real-time imputations.

1.1 Background

At first, the term "Smart City" was used to encourage urban development for economic growth with a focus on technology, innovation, and globalization. However, in the last decade, the term has been applied under a wide variety of meanings. There is no longer 'one' definition, if there ever was, for Smart City, with the term evolving as per the city's and country's visions. Studying the different interpretations of this term, Greco et.al. categorized the concept of Smart City as: 1) a *technology-centered approach* – where the emphasis is on technological advances (hardware), 2) a *human-centered approach* – where the emphasis is on advancing social and human resources, and 3) an *integrated approach* – where the emphasis is on effective collaboration of technology, social, and human resources to grow and innovate [8]. In the Report of the European Smart Cities, a Smart City is defined as a city that performs well in terms of mobility, environment, people, living, and governance, using the "smart combination of the endowments and activities of self-decisive, independent and aware citizens" [9], Figure 1.

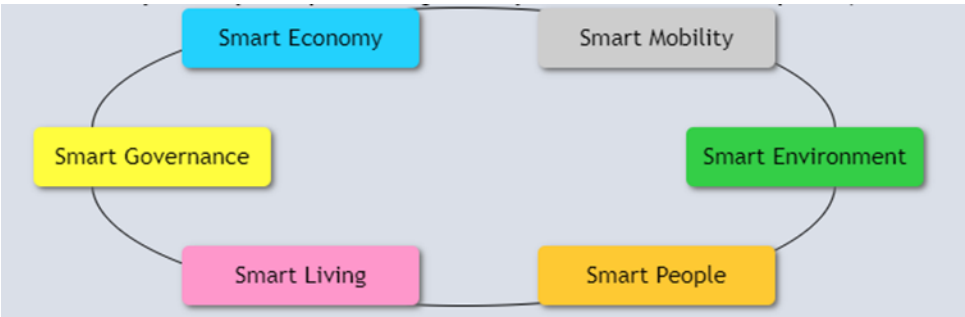


Figure 1 – The Smart City model [9].

Smart City initiatives are being carried out in cities across the world such as London, New York, Singapore, Hangzhou, Wolfsburg, etc. [10-14]. Smart city initiatives in transportation engineering consider providing solutions to transportation related issues such as congestion, fuel consumption, air pollution, crash occurrences, etc., using real-time traffic data available through advances in Intelligent Transportation System (ITS) technologies. In December 2015, USDOT's Smart City Challenge called city mayors to provide their vision for solving city's most crucial transportation problems [14]. The responses from 78 different cities across the U.S. outlined mobility challenges that included "Optimizing traffic flow on congested freeways and arterial streets" and "Limiting the impacts of climate change and reducing carbon emissions". Several city's demonstrated interest in implementing connected vehicles (CV) to study their potential for alleviating transportation challenges such as safety, congestion, and environmental pollution. Fifty-three cities proposed using Dedicated Short-Range Communication (DSRC) to facilitate communication between CVs and infrastructure to counter mobility challenges. Challenge finalists, Denver and San Francisco, proposed to deploy and use connected vehicle technology to improve traffic mobility and safety [14]. Connected vehicle corridors are expected to connect infrastructure, vehicles, and traveler mobile devices to make travel safer, smarter, and greener [15]. The Smart City "Connected Corridors" transportation

initiative is being tested in several cities in the US [16-19]. Connected corridors will allow access to high frequency near real-time traffic data such as vehicle count, signal phasing and timing, vehicle speed, etc. This dissertation takes a plunge into exploring applicability of the connected corridor generated high frequency data to study traffic performance.

1.2 Motivation and Problem Statement

On-road CV technologies provide access to granular real-time infrastructure and vehicle data. In a connected corridor, along with detailed vehicle data, signal state data is also expected to be available. With this emerging high frequency traffic data availability, it is crucial to explore applications, advantages, and challenges of such data streams to meet transportation related objectives such as congestion reduction, improved signal coordination, reduced air pollution, etc. This motivated the development of the (near) real-time connected infrastructure data-driven traffic simulation model, presented in this dissertation. In addition, an investigation of the real-time data streams that are inputs to the simulation model revealed the presence of data gaps, indicating a loss in data retrieval. Presence of data gaps in such real-time data streams can be expected in the complicated and environmentally harsh environment presented by field conditions. This motivated an investigative study to understand real-time data loss patterns, to find a suitable imputation methodology, and to measure the impact of data imputations on the real-time simulation model performance measures.

1.2.1 Connected Infrastructure Data-Driven Real-Time Traffic Simulation Model

The (near) real-time connected infrastructure data-driven traffic simulation model highlighted in this dissertation uses high frequency signal state data along with aggregate

volume count data as input, integrating infrastructure data into an online simulation. The developed architecture for the data-driven traffic simulation model enables near real-time visualization of the key performance indices of the connected corridor, such as travel time, queue lengths, emissions, energy consumption, etc. Such a dynamic data-driven simulation model can be crucial in testing the performance of different ITS technologies in varying traffic scenarios.

1.2.2 Real-Time Streaming Data Losses

High frequency data retrieval continuity from a connected corridor can be interrupted due to communication issues or other challenges. Such data gaps in the data streams can lead to erroneous simulation results. The error can be more pronounced for longer gap durations or when the loss occurs at a crucial corridor location. In the case of volume data loss this can result in the underestimate of true in-field demands. In case of signal state data, a data gap in the data stream can lead to a misinterpretation of signal phase timings, resulting in a significant deviation of the modeled vehicle behavior to that in the field. As seen, the presence of such gaps in the data streams that drive the simulation model can lead to a significantly different emulation of the traffic state in the simulation versus that in the real world. Thus, it is crucial to investigate the feasibility of using the developed connected corridor data-driven simulation model to derive insights from the simulated performance measures when data gaps are present. In addition, the development of imputation methodologies becomes crucial to the ability to utilize the deployed systems.

1.3 Research Objectives and Scope

Considering the onset of connected corridor testbeds and initiatives across cities in the U.S. and the world, **this research seeks to leverage the advanced ITS technologies in a connected corridor to drive real-time traffic simulations**. That is, to develop a **"Digital Twin" of the corridor**. The digital twin utilizes emerging and traditional data sources. For instance, connected corridors typically use wireless communications such as DSRC or Cellular V2V for connected vehicles and hardline (e.g. fiber) or cellular for connected infrastructure. In addition, such corridors often have traditional sensor technology, such as video, in-pavement detection, Bluetooth sensors, etc., and newer technologies, e.g. environmental sensors. To effectively maximize benefits from these existing and emerging technologies it is critical to develop the capability to simulate the connected infrastructure environment [20]. Consider, as of 2016, 74.6% of total Green House Gas (GHG) emissions produced by U.S. transportation system were from on-road vehicles [21]. Clearly, leveraging connected technology to aid in the reduction of GHG emissions on roads in the US is an important step to sustainability of future transportation systems.

The presented research explores the feasibility of the application of a real-time data driven simulation model of a connected corridor to provide dynamic traffic and environmental performance measures. Such a simulation tool can be vital to progress towards the smart city vision of safe, efficient, and green arterials. The research in this thesis is conducted on the North Avenue Smart Corridor in Atlanta, US.

The key research objectives are: 1) to develop the architecture and test the workability and the feasibility of the connected infrastructure driven real-time traffic simulation model, 2) to understand data gap patterns in the real-time volume data streams, 3) to measure and understand sensitivity of volume data imputations on simulation model generated performance measures such as travel time, energy consumption, and emissions, and 4) to investigate a suitable volume data imputation methodology. This research is funded by City of Atlanta and the National Center for Sustainable Transportation dissertation fund.

1.4 Document Organization

The research document contains an additional six chapters and appendices. Chapter 2 – Literature Review, provides a literature review on Smart & Connected Corridor initiatives and tests ongoing across the world, real-time data-driven traffic simulation models to measure traffic performance, and traffic data imputation and prediction methodologies. Chapter 3 – Connected Corridor Digital Twin Model Architecture, provides details of the key components of the developed connected corridor data-driven simulation model architecture and describes the connected corridor data streams used to drive the near real-time simulation model. Chapter 4 – Data Issues and Model Performance Sensitivity to Volume Imputations Experiment, investigates real-time volume data streams for data loss and presents a sensitivity analysis study conducted to measure sensitivity of simulation model generated performance measures to volume imputation values. Chapter 5 – Long Short Term Memory Recurrent Neural Network Layers for Connected Corridor Data Imputations, presents development of univariate and multivariate time series model for data imputations using Long Short Term Memory (LSTM) Recurrent Neural Network

(RNN) layers and experimental findings from comparing performance of multivariate and univariate models. Chapter 6 – Conclusions, Limitations, and Future Work, provides key takeaways from the studies conducted on development and performance of the real-time data-driven simulation model, Sensitivity Analysis Experiment, and LSTM RNN data imputation models, along with the challenges and limitations. It also presents a discussion on future research directions identified from the study needed to utilize connected corridor real-time data-streams to drive simulations and provide performance measures. Chapter 7 – Contributions, lists the contributions made through this dissertation research.

CHAPTER 2. LITERATURE REVIEW

This chapter reviews the prior work in development and application of real-time data driven traffic simulation models. The review focuses on three main aspects: 1) connected corridor deployments and studies, 2) real-time traffic data driven simulations, and 3) traffic data imputation methodologies. **Connected Corridor Deployments and Testbed Studies**

Cities across the world are undertaking smart city initiatives to improve resident and visitor quality of life, increase sustainability, and efficiently utilize resources. While there is no one smart city definition [22], smart cities concepts generally focus on advancing urban services by leveraging technology [23, 24]. A common smart city transportation initiatives is the deployment of sensors and advanced technologies to create Smart Connected Corridors. For example, the Pennsylvania Department of Transportation aims to use a coordinated smart corridor system to reduce congestion and crashes, and to provide real-time traffic information [25]. The Tennessee Department of Transportation is focusing on upgrading signals to optimize intersection operations on the Interstate 24 smart corridor [26]. In London, the A2M2 connected corridor is being used as pilot testbed to learn aspects of information transference between vehicles and infrastructure. The corridor is expected to provide enhanced mobility services such as real-time traffic management, and incident response through real-time personalized connectivity with the drivers [27]. The Minneapolis Department on Transportation (MnDOT) has selected part of the Highway 55 corridor for connected vehicle technology deployment. Using this corridor as a testbed, MnDOT aims to test connected infrastructure and develop a data management

system to support information sharing across connected environment stakeholders. The MnDOT connected corridor project aims to aid traffic safety by providing real-time assistive information to drivers about work zones, lane merges, pedestrian conflicts, etc. [28] Similar connected corridor projects are in progress in several other U.S. states, such as New York, California, Virginia, Wyoming, and Georgia [29-34].

To achieve the vision of leveraging information sharing between vehicles and the infrastructure, to enhance safety, mobility, and sustainability of transportation services, it is imperative to test different aspects of Connected Vehicle (CV) technologies in the real-world. To better understand and prepare for the "technical, institutional, and financial challenges" that emerging CV technology deployment will bring, the United States Department of Transportation (USDOT) is supporting the on-road deployment and testing of these technologies [35]. As part of this program, sites in New York, Wyoming, and Tampa, are testing CV applications that cater to local transportation needs. Since fall 2018, the connected vehicle systems at these sites have been operational. The impact on key performance measures at these locations are to be studied at least until fall 2021. The New York City Department of Transportation pilot is using a selected number of vehicles fitted with CV technology and Roadside Units (RSUs) deployed at selected intersections to test Vehicle-to-Infrastructure (V2I) safety applications. In addition, at this site in-vehicle applications to avoid vehicle-pedestrian conflicts will also be tested. The Tampa site focuses on testing V2I and vehicle-to-vehicle (V2V) safety and mobility applications, such as an intelligent traffic signal system. The key focus of the Wyoming Department of Transportation (WYDOT) is to use V2I and V2V applications to address challenges associated with weather and freight movement. For this, WYDOT is developing advisories

to provide safe transportation and efficient incident management systems, such as roadside alerts, parking notifications, and dynamic travel guidance [35]. Availability of high frequency data in such a smart corridor can be used to provide more accurate real-time dynamic information to road users and to transportation organizations.

Research on developing the necessary tools to test connected applications to quantify the impacts of this new technology is ongoing. For example, to evaluate the performance of a connected application - cooperative adaptive cruise control (CACC), Zulkerfli et al. (2016) recognized the need for a tool to test connected applications in a safe environment. The study built a simulation model to conduct an experiment with hypothetical input values for a number of vehicles in a network. The experiment was designed considering a mixed fleet of connected and non-connected vehicles [36]. Virginia DOT derived an algorithm for real-time prediction of vehicle locations in a connected environment using a traffic simulation model. The algorithm estimated locations of non-connected vehicles in the network using locations of the connected vehicles and other traffic parameters. The study also tested a connected application to optimize signal timing [37]. Doecke et al., studied a V2V safety application of market available connected vehicle technologies. In this study vehicle trajectory data from simulated crash scenarios are input to an On-Board Unit (OBU) that generated Basic Safety Messages [38].

The use of such hardware-in-the-loop (HITL) simulation platforms, are expected to advance the modeling of complexity found in connected environment systems [39, 40]. Chowdhary et al., studied connected applications to provide insights on handling real-time connected data streams and big data management tools [41]. The USDOT program Applications for the Environment: Real-Time Information Synthesis (AERIS) studied

connected environmental applications. Applications such as eco-approach and departure at signalized intersections, eco-traffic signal timing, and eco-traffic signal priority, etc., are tested on a simulation model of a 27 intersection network. The study utilizes historic volumes and signal data and different penetration rate of connected vehicles in the fleet. In this study, connected technology data such as Signal Phasing and Timing (SPaT) data, distributed by the road side units (RSUs) and on-road vehicle information such as location, speed, etc., provided by Basic Safety Messages (BSM) are simulated assuming a 100% penetration rate of RSUs [42].

2.2 Real-Time Traffic Data-Driven Simulation Modelling

The integration of smart technologies such as sensors, networked communications, and hardware and software computing with the physical infrastructure is central in creating a Smart City [43-45]. One area of significant focus is the integration of the new technologies enabling an improved estimation of the traffic state and real-time traveler and traffic information [46, 47]. Given the inherent challenges in field experimentation, traffic simulation models driven by real-time input data to emulate the real-world environment are utilized in numerous studies [43, 48]. Various traffic simulation tools have been used by researchers, depending on the modeling requirements. A University of Leeds report in 2000 compared the capability of several macroscopic and microscopic simulation models in developing real-time traffic management solutions [49]. More recently, a similar study looked into 17 simulation software tools and noted “a lack of online traffic simulation software applications specially designed for heterogeneous road transportation networks” [50].

The concept of using real-time data to drive a traffic simulation model is not new. Previous studies, such as Henclewood et al. (2010), injected real-time vehicle detection data into microscopic simulation models to simulate the current traffic state, which was then used to generate predictions of future traffic states [51]. An early example of real-time simulation includes Maroto et al., who developed a microscopic model that could simulate driving simulator traffic scenarios in real-time, where vehicle behavior model is based on car following theory [52]. An effort, tested in the Dutch City of Assen, built a real-time traffic model that used traffic flow and travel-time data from different sensor technologies such as cameras, highway loop detectors, and Bluetooth® sensors, to predict the short-term traffic state. The model architecture connected the real-time traffic measurements with the macroscopic dynamic traffic assignment model "StreamLine". Traffic counts were used for model calibration and forecasting [53]. A study by Sturari, Catani et al. presented the use of in-field mobile and fixed sensor data to drive a real-time microscopic traffic simulation model built using the Simulation of Urban Mobility (SUMO) simulation package. Model input included the real-time traffic count and vehicle location data obtained from different sources such as induction loop, camera counter, radar counter, automatic vehicles location (AVL) systems, etc. [54]. In these studies, the real-time data comprised of vehicle detection or vehicle position data; however, infrastructure information such as the traffic signal state, state of ramp meters, information from variable speed limit signs, etc., were assumed to be pre-encoded in the simulation model based on known logic or field calibration. Today, with the richness of information in a Smart City CV environment, where the vehicle sensor data may result in real-time changes to the

signal control, ramp meter rates, etc., it is imperative that the state of the infrastructure is updated in the simulation to ensure the accuracy of the simulation results.

Anthony et al. studied computational benefits of using parallel processing for simulating traffic flows in a real-time traffic simulation system and concluded that parallel system can be crucial for real-time traffic simulations [55]. USDOT FHWA is using the hardware in loop approach, where the connected infrastructure and connected vehicle data is fed as input to the simulation model, to study the application of CV technologies. The research is expected to provide engineers with a technology that will allow CV applications engineers to test various simulation scenarios and obtain meaningful results [56].

The availability of these various data streams also provide interesting challenges. The volume, velocity, and wide variety of these data streams naturally suggest the use of big data technologies for extracting useful information. Previous studies such as Amini et al. (2017) have used tools such as Kafka to address the issues of volume and variety of big data [57]. Lv et al. (2015) used deep learning techniques for predicting traffic flow [58]. The current study takes a hybrid approach, where the architecture allows for the use of big data concepts in the extract-transform-load (ETL) stages preceding the injection of the data into a simulation model.

2.3 Traffic Data Imputation Methodologies

The presence of data gaps in the traffic data collected is common and so, traffic data imputation methodologies have long been of interest to traffic engineers. Collected traffic data can be volume count data, vehicle speed data, or one of many other multitudes of data. Heuristic imputing methods such as replacing missing values by another day's data

directly, or by using an average, moving average, or weighted moving average of historical data are used [59, 60]. In 2006, Zhong et al. applied a pattern matching based method which choose the day for replacement values based on the match of values obtained in previous selected stretch of hours [61]. Auto-regressive Integrated Moving Average (ARIMA) models where missing data is predicted based on available preceding values have been used and studied in several efforts [62, 63]. ARIMA based models have been frequently used with modifications to account for long-term seasonal variations, that is, Seasonal ARIMA [64-66], and to account for spatial variations, that is, Space-Time ARIMA [67]. In the learning methods, non-parametric learning algorithm K nearest neighbor (knn) algorithm for imputations has shown high accuracy [68-70]. A recent study by Zhuang et al. applied a convolutional neural-networks (CNN) based image inpainting approach to find values for missing volume data imputation. In comparison to two other methods (Bayesian Principal Component Analysis and Denoising Stacked Auto Encoder – a deep learning based approach), CNN-based approach performed better [71]. For sequence predictions, Recurrent Neural Networks (RNNs) have shown success in several fields such as speech recognition and language modelling, particularly because of its' chain-like connected architecture [72]. While RNNs show success in predicting future values based on recent past values, they do not learn long term dependencies [72]. To address this, LSTM units in RNN architecture are preferred [72]. Detailed description of working of RNN, LSTM, and bidirectional LSTM along with literature review on its application for time series predictions is presented in Chapter 5.

The review of literature in first subsection of this chapter "Connected Corridor Deployments and Testbed Studies" suggests an increasing focus in cities across the world

to deploy connected corridor testbeds to study their potential for improving traffic characteristics such as mobility, safety, and information connectivity. The next subsection "Connected Corridor Deployments and Testbed Studies" presented a summary of previous work where transportation data from different data sources are leveraged to develop real-time transportation simulation models. It is seen that with increasing deployments of connected corridors it is imperative to explore leveraging connected corridor data to improve traffic mobility, safety, etc.

Leveraging what has been learned previously this research develops a real-time traffic simulation model of a connected corridor, i.e., Digital Twin, driven using data derived from the corridor. The developed model and data imputation methods are capable of providing traffic and environmental performance measures at a near real-time, which can be used to improve traffic mobility and reduce air quality impact.

CHAPTER 3. CONNECTED CORRIDOR DIGITAL TWIN MODEL ARCHITECTURE

In a Smart City, equipped with connected infrastructure, traffic data, such as vehicle detections and intersection signal indications, are expected to be received in (near) real-time. Utilizing such data the objective of this chapter is the creation of a connected infrastructure dynamic data-driven simulation that leverages high frequency connected data streams and may be used to derive meaningful insights about the current traffic state and real-time corridor environmental measures. An earlier version of the model that utilized hybrid data (a mix of real-time data and preset data) to simulate traffic and provide performance measure estimations at a near real-time rate is presented in the IEEE proceeding, proceeding of Winter Simulation Conference [73].

3.1 Background

The Connected Corridor Digital Twin model developed and studied in this research simulates 2.3 miles of the North Avenue Smart Corridor in Atlanta, Georgia, USA. This simulated section of the corridor consists of fifteen signalized intersections (Figure 2).

Figure 2 – Study corridor-2.3 miles of North Avenue Smart Corridor, including 15 signalized intersections.

The developed model is capable of being driven in near-real-time with high frequency connected infrastructure traffic volume and signal controller data streams. The model visualizes key traffic and environmental performance measures at near-real time, providing dynamic feedback to users and transportation stakeholders. Such a data driven simulation platform can be crucial in providing helpful insights on the effectiveness of new technology deployments. In this context, the built simulation model can be used to test impacts of smart/connected technologies, such as smart signal control or traveler information systems, in real-time.

The simulation model architecture feasibility and robustness were initially explored by driving two intersections with near-real-time data streams and the remaining intersections using preset data. In this initial experiment the overall architecture was found to be capable of inserting the data into the simulation platform, maintaining the faster than real-time processing necessary for such a platform to maintain real-time capabilities, and able to provide meaningful traffic operations and vehicle emission estimates along the corridor [73]. For this chapter the ability to stream the volume and signal real-time has been expanded to all fifteen signalized intersections and the overall architecture has been improve to address deficiencies in the original effort. For instance, an investigation of the

data streams has revealed the presence of data gaps. Given the challenges of maintaining data streams in harsh field environments, such data loses should be expected. Such data gaps can impact the performance measure results and insights generated from the model. Thus, the model architecture has been enhanced to handle such data losses.

However, development of an imputation methodology should be informed by an understanding of the impact of data loss, and errors in data imputation, on generated performance measures. This chapter provides an overview of the improved architecture, utilizing a simple data imputation methodology. Chapter 4 utilizes sensitivity analysis to explore the impact of the volume data gap imputations on the key performance measures produced by the real-time data-driven simulation and Chapter 5 will present an advance data imputation methodology.

3.2 Digital Twin Model Architecture

Figure 3 provides the real-time data-driven simulation architecture. The architecture is enabled to perform four primary tasks: 1) injection of real-time signal control and volume data streams into traffic simulation model through the *Real-Time Raw Data Stream Processing Module*, 2) model execution through the *Dynamic Data-Driven Simulation Module*, 3) the *Dynamic Performance Metric Evaluation and Visualization Module*, and finally 4) efficient handling of transactions between modules using a *Data Request Management Module – Flask Web Server*.

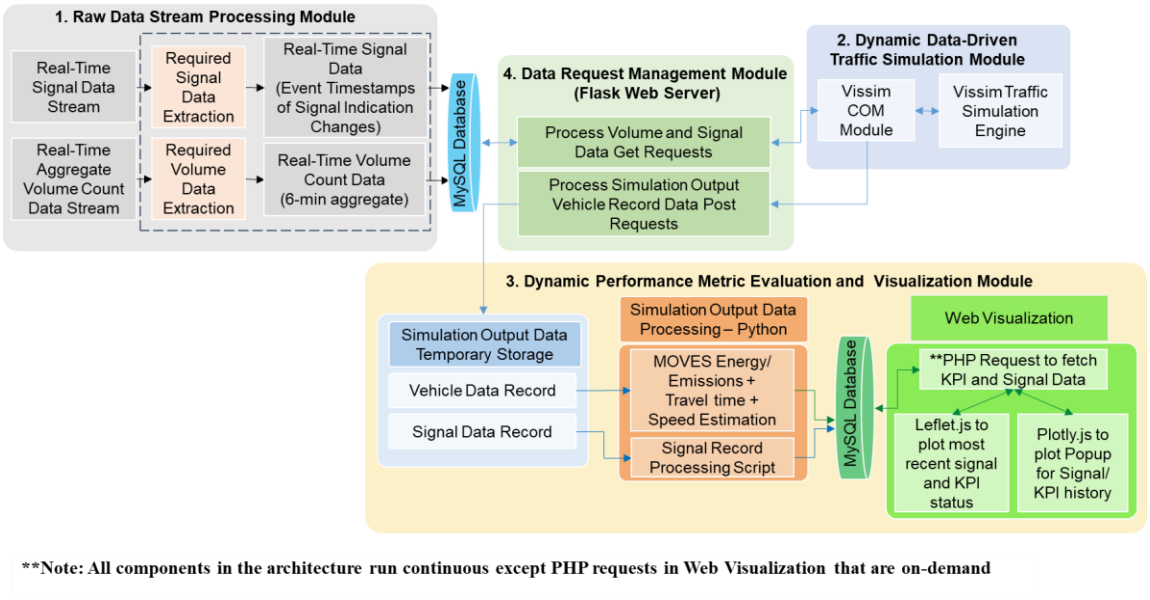
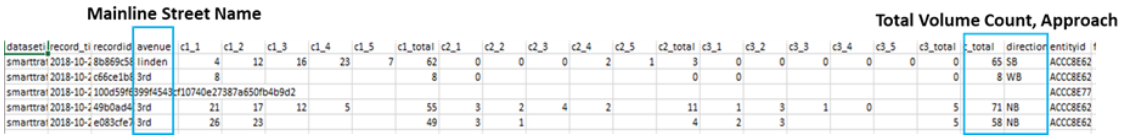


Figure 3 – Real-time data-driven simulation model architecture overview.

3.2.1 Real-Time Raw Data Stream Processing Module

For each intersection real-time vehicle per-lane counts are collected in the field using video detection and processing. Data is received (*Real-Time Aggregate Volume Count Data Stream*, Figure 3) as 6 minute, per-lane aggregate counts. The raw volume data stream, a sample shown in Figure 4, is reduced to include only the approaches of the fifteen studied intersections. Volume count data contains the timestamp of the start of interval and the distribution of volume count per lane, for each of three vehicle classes. The 3 classes are listed as c1, c2, c3 in Figure 4, where c1_1 refers to count of vehicle class 1 in lane 1 and similarly c2_5 refers to count of vehicle class 2 on lane 5. The lane ID (1, 2, and 3 etc.) follows the serial order from rightmost lane to the median. In the *Real-Time Raw Data Stream Processing Module* lane level raw volume data counts are aggregated across classes and lanes to obtain approach level by movement aggregate counts, to be used as input for the traffic simulation model.



dataset	record_ti	recordid	avenue	c1_1	c1_2	c1_3	c1_4	c1_5	c1_total	c2_1	c2_2	c2_3	c2_4	c2_5	c2_total	c3_1	c3_2	c3_3	c3_4	c3_5	c3_total	_total	direction	entityid
smarttra	2018-10-2	8b869c58	linden	4	12	16	23	7	62	0	0	0	2	1	3	0	0	0	0	0	0	65	SB	ACCC8E62
smarttra	2018-10-2	c66ce1b6	3rd	8					8	0					0	0					0	8	WB	ACCC8E62
smarttra	2018-10-2	100d59f6	399f4543	cf10740e27387a650fb4b9d2																				ACCC8E77
smarttra	2018-10-2	49b0ad4	3rd	21	17	12	5		55	3	2	4	2		11	1	3	1	0		5	71	NB	ACCC8E62
smarttra	2018-10-2	e083cfe7	3rd	26	23				49	3	1				4	2	3				5	58	NB	ACCC8E62

Figure 4 – Snapshot of raw real-time volume count data.

The signal controller data streams are received (*Real-Time Signal Data Stream*, Figure 3) at a higher frequency than the volume data, ranging between 0.1 Hz to 10 Hz. The received data stream contains the signal color indication status of all signal heads for all intersections. Separate messages are sent for each intersection and each message contains the state of all signal indications for that intersection. A message is sent whenever any indication changes at the intersection. However, the raw signal data stream, may include repeated redundant messages. For several intersections update messages were sent approximately every 2.5 seconds to 5 second even when there was no change in indications. From this data stream, messages that reflect a change in the current state of the signal indications (i.e. a light changes on any signal head associated with the intersection) are identified to be used as dynamic input to the simulation model. Figure 5 provides a sample signal data, where each signal message record contains associate date, timestamp, intersection ID, position of the intersection in latitude longitude format, and Hexcode ID. The Hexcode ID is converted to binary data string that contains signal status of all (max of 8) phases of the intersection at the corresponding timestamp.

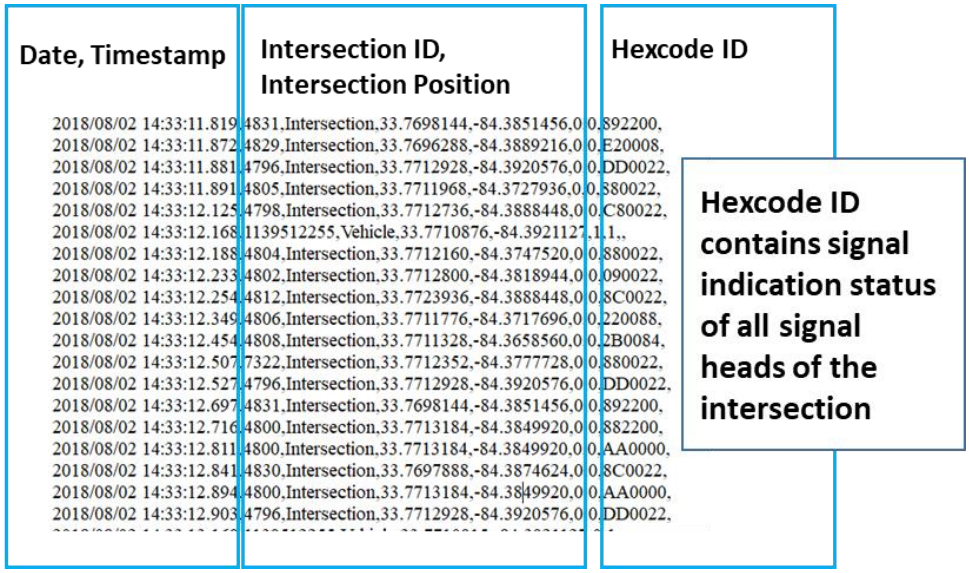


Figure 5 – Snapshot of raw real-time signal event data.

3.2.2 Dynamic Data-Driven Traffic Simulation Module

The dynamic data-driven traffic simulation model is developed in PTV's Vissim 9.00-08. Vissim's Component Object Model (COM) interface is used to the feed real-time data into the simulation model during runtime and to send simulation output data to the performance measure evaluation and visualization module dynamically. Python 3.7 programming language is used to access Vissim objects dynamically via its' COM interface. The simulation resolution is 10 Hz. The *dynamic-data driven traffic simulation module* consist of the following components: Volume Update Logic, Signal Update Logic, Dynamic Switch between Actuated Signal Timing and Real-Time Signal Data, and Dynamic Simulation Output.

3.2.2.1 Volume Update Logic

The vehicle volume simulation inputs for all the boundary links of the fifteen signalized intersections are updated every aggregate count period to match the fluctuations

in real-time vehicle volume. The turn movement ratios for internal links are enabled to change dynamically in the architecture base on streaming detector data. Where turn movements and through movements share a lane, turn movement splits are set based on historic data. Currently the aggregate count period is six-minutes, i.e. the volume count bin size from the field equipment. This represents the minimum lag between the simulation model and the field. In earlier versions of the architecture, for a different volume aggregate data stream, the time interval of volume update has been tested to work successfully for one-minute and five-minute time intervals as well.

The built architecture simulates the last interval for which data is obtained from the real-time data stream. The simulation resolution is 10 hz (although signal change events occur at a 1 hz rate as will be discussed in the next section). The volume at each entry point is entered according to a shifted Poisson inter-arrival headway distribution across the six-minute period. For dynamic control of volume inputs Vissim COM objects container "IVehicleInputContainer" and "ITimeIntervalSetContainer" and "ItemByKeys" and "SetAttValue" methods are used to access and generate the next six minute time interval and add the corresponding volume value at each of the entry links simulated [74]. Similarly, to dynamically adjust turn movement ratios (distribution of vehicle across different route paths) in different time intervals, Vissim COM object container "IVehicleRoutingDecisionStaticContainer", "ITimeIntervalSetContainer" and "IVehicleRouteStaticContainer", with "ItemByKeys" and "SetAttValue" methods are used [74]. The ID for each of the individual object "IVehicleInput", "IVehicleRoutingDecisionsStatic", and "IVehicleRouteStatic" in object containers are taken as input by the "ItemByKeys". For code brevity, dictionaries with list of the IDs for

these objects, for all intersection entry links in the network, are utilized to list each object item and update volume and turn movement split. Figure 6 provides sample code utilized to access Vissim objects "IVehicleInputs" to update volume count values and "IVehicleRoutingDecisionsStatic" and "IVehicleRouteStatic" to update turn movement ratio values dynamically.

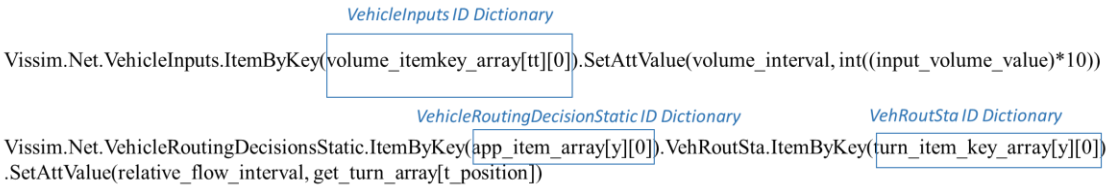


Figure 6 – Sample code to access Vissim objects dynamically via COM interface.

3.2.2.2 Signal Update Logic

The signal data stream is received in the form of hexcode ID that contains the signal indication status of all phases, for each of the fifteen intersections. The hexcode ID is compared with last received hexcode ID to check for any indication changes. The updated hexcode IDs for the fifteen intersections are identified and fed to Vissim every second. Thus, the signal update resolution to Vissim is 1 Hz. The hexcode IDs can be received at a 0.1 Hz frequency. In that case, the most recent hexcode ID identified in the complete one second interval is fed to Vissim. Considering signal indication length for an given phase always exceeds one second a phase change indication will not be missed with the with the 1 Hz signal update resolution utilized in updating Vissim. Through COM the updated hexcode IDs are decoded to identify the status of every phase at the intersection to implement the new status in the next simulation step in Vissim.

The signal data drives all signal changes in the VISSIM model unless a data gap is identified, in which case, the signal control will be switched to the inbuilt RBC controller to emulate actuated signal timing plan, as discussed in the next section. For dynamic control of signal indications at the fifteen signalized intersections, Vissim object container "ISignalControllerContainer" and "SignalGroupContainer", are used to access intersection signal controller and the signal groups (i.e. phases) at each intersection utilizing the "ItemByKeys" method via COM interface. To switch signal indication control from COM to RBC the "ContrByCOM" attribute of the signal group object is set to "False." Figure 7 provides utilization of COM method to access Vissim signal head dynamically.

```
Vissim.Net.SignalControllers.ItemByKey(11).SGs.ItemByKey(2).SetAttValue("SigState","GREEN")
```

Figure 7 – Sample code to access Vissim signal objects dynamically via COM interface.

The signal (and volume) data preprocessing methodology before feeding into traffic simulation is subject to modification based on the received data format. In an earlier version, a different signal data stream - Maxtime Signal Controller output - was utilized. In this version, the data entry received included event ID number (indicating start of an event such as start of Green indication) for an associated phase. However, regardless of the data source, when implementing an operational real-time Digital Twin model, data storage and organization of data streams can be crucial to efficiency, for example if the data is polled in separate files for each intersection or data for a group of intersections is stored in one file.

3.2.2.3 Dynamic Switch between Actuated Signal Timing and Real-Time Signal Data

The visualization enabled by the previous logic facilitated the inspection of signalized intersections, allowing for the ready recognition of errors in the simulated signal control, i.e., skipping or very long indications. Errors could be due to manual error in decoding hexcode messages, a gap in signal data stream due to communication or field equipment failure, or other error sources. A more detailed study of data gaps in the real-time connected corridor data streams is presented in Chapter 4. While application of advanced imputation models to infill volume data gaps is tested in this dissertation (presented in chapter 5), signal data imputation methodologies remains a future research topic. However, to address signal data gaps this effort utilizes COM's capability to implement preset actuated signal timing data using its Ring Barrier Controller (RBC) module. The RBC is utilized to keep the real-time data-driven digital twin intersection running continuously, even in the absence of streaming signal data for one or more intersections. For this effort a signal data gap is defined as is when the hexcode ID for an intersection is not updated for more than 4 minutes. At such time the digital twin architecture switches signal control of that intersection to RBC mode. On receiving updates in hexcode the intersection control is switched back to the dynamic control, where in signal status emulates the status in the hexcode ID. While this could result in momentary unrealistic signal changes within the digital twin, validity of this methodology is reserved to be part of future real-time signal data imputation methodology development.

It is highlighted that for the built simulation model of the real-time connected corridor data driven digital twin architecture the capability to dynamically switch between the RBC and external data stream to drive signal indications in simulation has been successfully implemented. However, the experiments conducted in following research do

not utilize this methodology. As the experiments were able to be run offline using historic data streams these streams were manually validated and updated to provide more realistic indication changes. This is a temporary measure to allow testing the robustness and efficiency of different configurations of the complete digital twin architecture without the confounding of signal control transition errors.

3.2.2.4 Dynamic Simulation Output

The dynamic link to transfer simulation output data from the *Dynamic Data Driven Simulation Module* to the *Performance Measure Evaluation and Visualization Module* is facilitated through COM utilizing Flask Web servers' post requests. For energy consumption and emissions estimation, a detailed record of the simulated vehicles is sent as key-value pairs through these posts into JSON files. After each simulation step, a vehicle record set for all vehicles in the network is obtained. The record set includes: Record Simulation Second, Vehicle Network Entry Time, Vehicle ID, Vehicle Front Coordinate, Vehicle Rear Coordinate, Speed, Acceleration, Vehicle Type, Vehicle Width, Vehicle Length, Direction of Current Lane Change, and Headway. COM transfers the vehicle record information to *Performance Measure Evaluation and Visualization Module* at a resolution of 1 Hz.

Additionally, for web visualization of the signal indication data, a separate Flask webserver post request is utilized. Similar to vehicle record transfer, the signal status information is sent as key-value pair (key: intersection id, value: signal status) to JSON files at a resolution of 1 Hz. Both signal status records and simulated vehicle records for

every 1 simulation second are stored in a separate JSON files. This allows a flush of the previous signal and vehicle record JSON files after they are utilized for visualization.

3.2.2.5 Vissim COM Logic

Figure 8 provides an overview of the dynamic data driven simulation initialization logic and Figure 9 provides an overview of the simulation runtime logic. Initial tests for robustness and feasibility of the built simulation model to run for 24 hour is tested for a selected day of 14 January 2019.

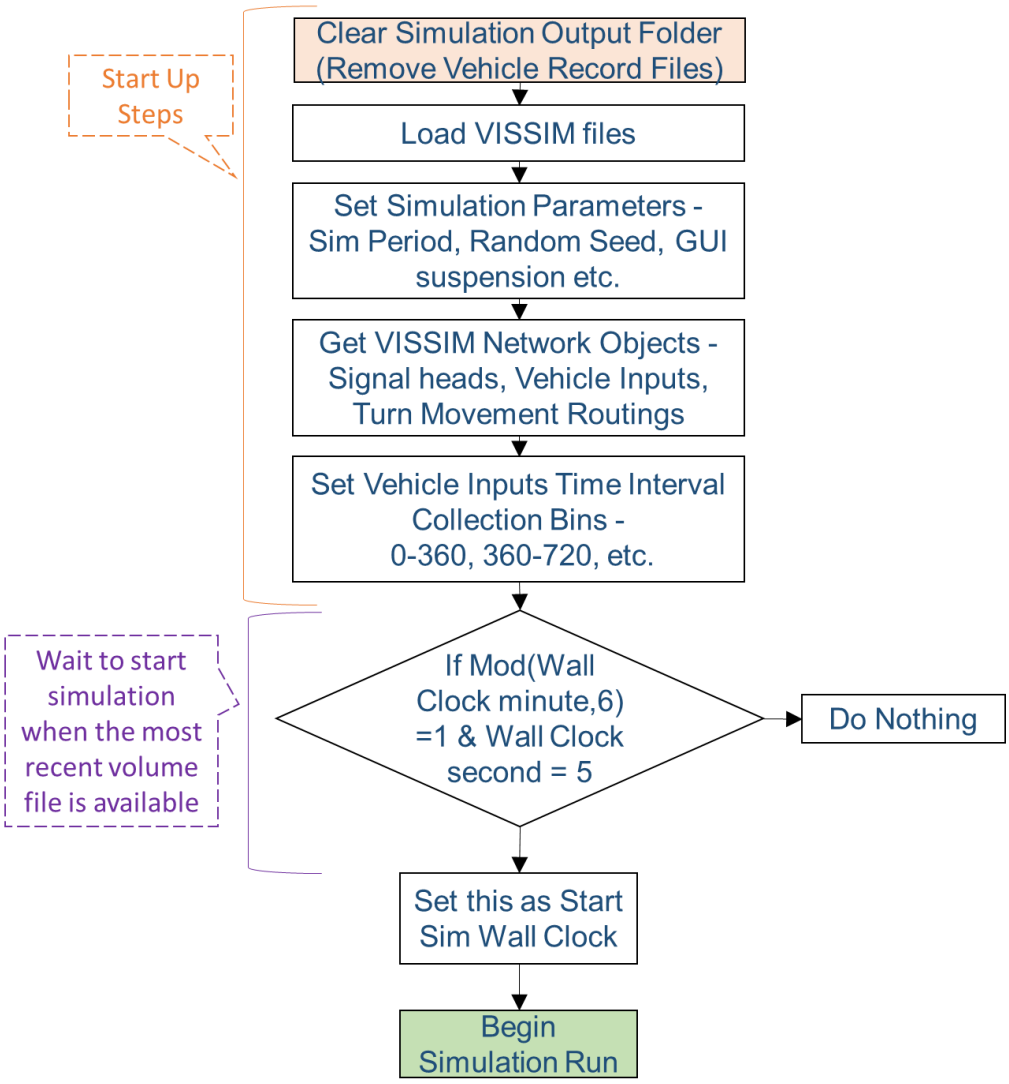


Figure 8 – Overview of dynamic data-driven traffic simulation initialization logic.

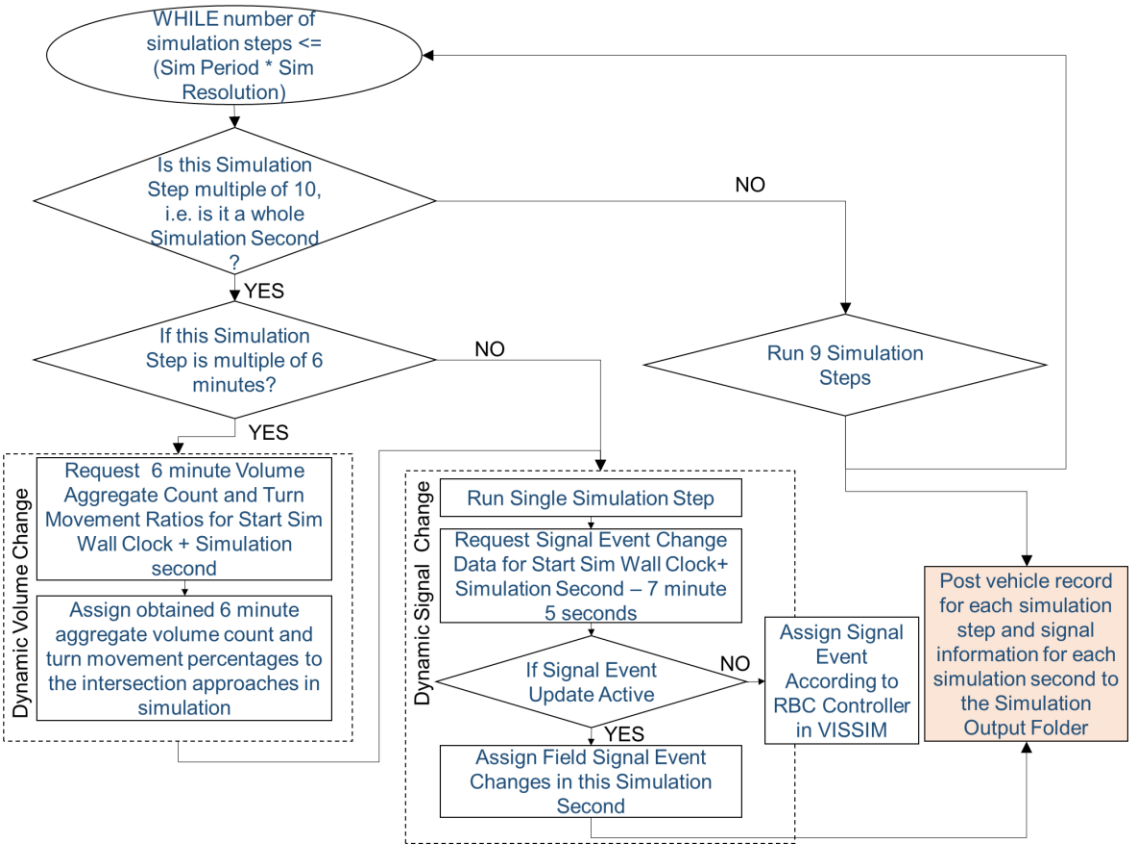


Figure 9 – Overview of dynamic data-driven traffic simulation runtime logic.

3.2.3 Dynamic Performance Measures Visualization Module

The Dynamic Performance Measures Visualization Module was completed by team members outside of the efforts reported in this dissertation [75]. However, for completeness the module may be described as "The simulation provides a record of position, speed, acceleration, etc. for each vehicle in the network, at a 1 Hz rate. These per-second vehicle record data are used in runtime to estimate performance measures such as travel time on selected routes, queue lengths, energy consumed per vehicle, and emissions generated per vehicle. To estimate energy and CO2 emissions profile the Motor Vehicle Emission Simulator (MOVES) matrix is utilized. MOVES matrix is a computationally improved version of the energy estimation tool MOVES, developed and mandated by the

US Environmental Protection Agency (USEPA) [76]. It estimates energy and emissions for off-road and on-road vehicles based on the vehicle type, weather conditions and vehicle model, make, and year. The estimated performance measures are then visualized on Openstreet maps" [75].

3.2.4 Data Request Transactions Management Module – Flask Web Server

During a run dynamic data requests between the three other modules are handled using python's Flask webserver. A Flask webserver is used to generate urls that contain raw data fetched from the *real-time raw data stream processing module* requested by the *traffic simulation module*. Figure 10 includes the two primary functions used to fetch real-time data dynamically by simulation using Flask webserver hosted urls.

Figure 10 – Flask webserver is used to fetch real-time data during simulation.

Further, the flask webserver is used to process post requests in the Vissim simulation logic, creating a dynamic log of the simulation output data that contains a record of position, speed, acceleration, etc. for each vehicle in the network. Lastly, the flask webserver is also used to facilitate a similar dynamic data request link between the simulation output data log and the dynamic performance metric visualization module.

While not presented for brevity an assessment of the feasibility aspects of using a real-time data-driven transportation simulation model to evaluate and visualize network performance indices to provide dynamic operational feedback in a real world environment was conducted using an earlier version of the architecture [75]. Within this early

experiment it was seen that not only was the method feasible but that it likely provided a prediction improvement over the simulation platform run offline with historical data. Results for the current architecture will be seen in Chapter 4 and Chapter 5.

3.3 Conclusions and Limitations

A real-time data-driven simulation model that drives signals and volumes of all fifteen signalized intersections utilizing real-time data is developed. The four primary components of the developed Digital Twin are 1) *Real-Time Raw Data Stream Processing Module,* 2) *Dynamic Data-Driven Simulation Module*, 3) *Dynamic Performance Metric Evaluation and Visualization Module*, and 4) *Data Request Management Module – Flask Web Server.*

Vissim's COM interface is utilized for dynamic control of simulation inputs and outputs. Inputs to the simulation model, i.e. volume, signal status, and turning movement ratios at the fifteen intersections, are provided dynamically utilizing the COM interface. Dynamic simulation updates are made to input volumes and turning movement ratios every six-simulation minutes and signal indications every one-simulation second. Further, simulation output, the vehicle record of all vehicles in the simulation network at a rate of 10 Hz and signal record of signal status of all phases at the fifteen intersections are dynamically provided at a resolution of 1 Hz by COM to the *Performance Metric Evaluation and Visualization Module*. Flask webserver is utilized to 1) transfer the real-time data from the raw data streams to the simulation COM interface, and 2) transfer simulation output data provided by COM to external JSON files for performance metric evaluation and visualization. Thus, the dynamic handle to Vissim simulation objects

provided by COM interface and Flask webserver requests are central to the dynamic operations of the built data-driven traffic simulation model. The dynamic attribute of the developed simulation model to update volume inputs, turning movement percentages, and signal status at different frequencies, can be changed depending on the real-time data stream granularity.

The handling of real-time data streams to drive the developed Digital Twin addresses the seven big data attributes: 1) Volume – handles large volumes of data; 2) Velocity – data is streamed in real-time; 3) Variety – a number of data formats are integrated; 4) Veracity – addresses data quality (missing data); 5) Variability – data varies over time, following typical and atypical patterns; 6) Visualization – the architecture includes performance metric visualization; and 7) Value – critically the method converts the data into useful information that may be used to evaluate the corridor operations. Further, the developed real-time simulation platform can be integrated with different smart technologies as the model relies only on the data from smart technologies and not the imbedded logic of the smart technology to drive the simulation.

One limitation of the current approach is runtime. While for initial test version, the traffic simulation model is observed to operate 1.3x rate compared to wall clock [77], for the current real-time data-driven traffic simulation model, the traffic simulation rate varies between 0.8x to 1.3x, depending on network traffic density. Thus, a network with continuous high traffic density can result in the simulation falling behind real-time. The Digital Twin with all fifteen intersections driven using real-time data and the dynamic simulation outputs sent at a high resolution, tends to lag real-time rate, particularly during peak periods. This may be partially addressed by reducing the output resolution, although

this is identified as a limitation and an area of future work. It is also noted that to keep up with the real-time rate for a larger network, the simulation will need to either incorporate distributed architecture or utilize multi-processing or multi-threading for efficient computing. Achieving a real-time simulation can lead to future research to devise a methodology to simulate the prediction state duplicating the current simulation state.

In the current architecture, the real-time volume data is received with six-minute aggregate. The simulation model updates the volume for every six-minute interval. A more accurate version can be created with a lesser aggregate real-time volume data. For example, with one-minute aggregate volume data, the volume in simulation model could be updated every one-minute. This low level aggregation will also allow the simulation to perform more closer to real-time than the current version since it will simulate and provide performance measure of traffic state observed in last minute instead of last six-minute interval for which data is received. It is important to note that this will contribute to higher number of dynamic changes in the simulation model and could impact simulation rate. Hence, efficiency of the simulation model might need to be enhanced as well. Further, if instead of aggregate volume data, per vehicle record data is obtained, the lag of one-minute can also be reduced significantly to achieve a more accurate real-time Digital Twin. However, this change might increase the frequency of communications between the real-time data stream database and the simulation model. Hence, the dynamic link between the two components will need to be tested for robustness.

Another limitation of the model is that the traffic simulation model has not been calibrated for driving behavior and validated for the traffic and environmental performance measures. The model in future will need to be calibrated and validated. For a successful

calibration and validation of the simulation model, it is also crucial to investigate fidelity of the real-time data received in the connected corridor. As a first step, in this effort the real-time connected corridor data is investigated for presence of data gaps, which is presented in Chapter 4. However, a study to verify the accuracy of data received from sensors will also be needed to develop a validated and calibrated model.

Further, in the built Digital Twin architecture, the dynamic turning movement ratios from historic turning movement counts are used. The connected corridor data received from the studied currently did not provide turning movement counts at approach before intersections. In addition, it also lacks information that could come from on exit detectors which could be used to estimate turn movement counts more accurately. Potential solutions to this could be to facilitate obtaining turn count data directly from the connected corridor sensors at approach detectors, or to leverage data from exit detectors as well if present or to develop a methodology to estimate turn movement count from the counts data and signal phase timing data of the neighboring intersections. It is noted that using an external turn movement count estimation methodology might cause additional delay in execution of the real-time aspect of the Digital Twin. Hence, it will be also important investigate the execution time of the developed estimation methodology after it is embedded in the architecture. A model that has learned from the turn count patterns from historic data could be used.

It is also noted that to keep the Digital Twin running 24 x 7 for practical purposes, a robust and reliable network communication between raw data stream database, traffic simulation model, and simulation output storage database that is used for estimating performance measures is needed. The dynamic link between the three main components of

the architecture makes this crucial. In practice, interruptions have been observed due to network fluctuations and communication loss to Vissim license. The architecture in future can be enabled to provide a notification when simulation model does not run.

A related challenge identified in building the real-time digital twin transportation simulation model is in integrating the multiple components underlying the complete model and ensuring they work in synchrony with real-time. However, through the architecture developed and test the performance results indicated that with the presented architecture it is plausible to work with the high velocity data while ensuring sufficient responsiveness of the model to input changes. However, ensuring synchronicity can impact the model efficiency contribution to runtime challenges.

Investigation of raw data streams during model development revealed the presence of data gaps. This led to next research efforts: 1) to study sensitivity of the error in data imputations on Digital Twin generated performance measures, presented in Chapter 4, and 2) to investigate potential benefits of deep RNN models for data imputations, presented in Chapter 5. A pseudo-real-time data-driven version of the Digital Twin is utilized to conduct experiments in Chapter 4 and Chapter 5, where the raw data for a fixed day is already available as opposed to real-time. The performance measure evaluation is conducted on the obtained result of multiple trials of the simulation as opposed to dynamically in the Digital Twin.

CHAPTER 4. DATA ISSUES AND MODEL PERFORMANCE SENSITIVITY TO VOLUME IMPUTATIONS EXPERIMENT

4.1 Introduction

While the architecture has been shown to be feasible an investigation of the real-time data streams used as input to the simulation revealed data loss in the data streams, which if left absent or incorrectly imputed may impact the simulation results. Investigation of the streams revealed data gaps are likely related to communications (dropped or highly latent messages), equipment failure, or data message processing. The volume and signal data were transmitted through separate systems, albeit both cellular based. Thus, data loss was not correlated between the two data streams. The key characteristics of the observed data loss events are presented in the next sections.

4.2 Investigation of Real-Time Data Streams

A description of raw data streams of volume count and signal indications is provided in Chapter 3 section *3.2.1 – Real-Time Raw Data Stream Processing Module*. In this chapter, the real-time raw volume data streams are investigated. Investigation of volume data streams for 115 days across months of February, March, April, May, and June (2019) revealed presence of data gaps.

4.2.1 Volume Data Streams

The raw real-time volume data stream contains information such as the volume interval starting timestamp, volume count distribution per vehicle class per lane, speed,

occupancy, etc. for each record, where a record is created for each intersection approach. To study data gaps in the volume data stream, the raw data is reduced to contain only that information required for each record to clearly identify the presence and absence of volume data for each interval of the day, i.e. identify data gaps.

4.2.1.1 Raw Volume Data Processing to Identify Missing Volume Intervals

Using the four-step filter process shown in Figure 11, starting with the database table containing the raw volume data, a final *Standardized Name Table* is developed. This table implements a standardized data and lane number scheme (i.e. lane number ID increments from right most approach lane) allowing standardized input into the simulation model. This table filters unused data, converts all roadway naming and lane assignments to a standardized, consistent format, and modifies the timestamps, converting from Greenwich Mean Time (GMT) to Eastern Standard Time (EST). In generating the *Standardized Name Table,* Lane ID modification is completed for the approaches where incomplete or incorrect lane IDs are identified. For example, if an approach with three lanes (two through lanes and a right turn lane) has data for only 2 lanes, listed as c1_1 and c1_2, it is assumed that the detections are present on the through lanes. With this assumption and according to the lane ID numbering scheme, the IDs for these lane are converted to c1_2 and c1_3 with c1_1 being the right missing detector ID name. Every intersection was reviewed and manually verified for lane numbering.

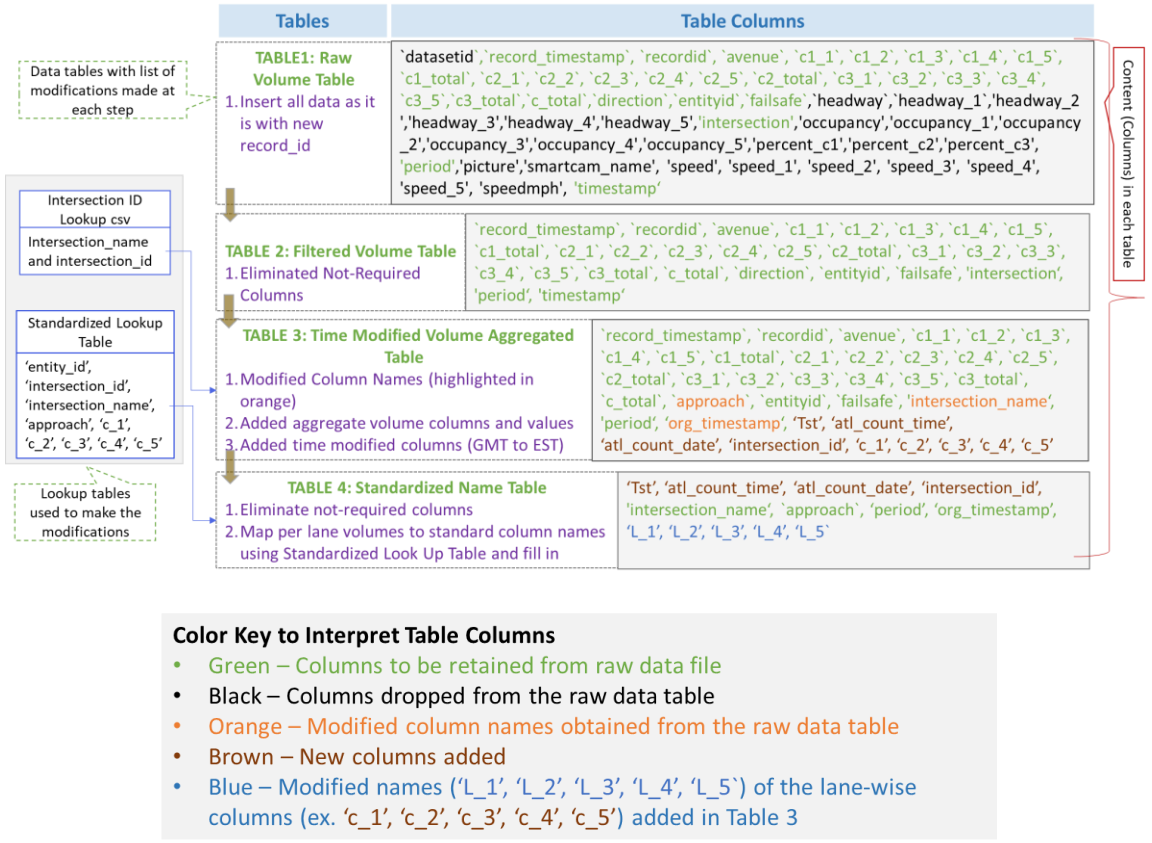


Figure 11 – Four-steps of data processing to obtain the Standardized Name Table from Raw Volume Table.

4.2.1.2 Volume Data Gaps Analysis Aided with Interactive 2D and 3D Visualizations

Analysis of volume data streams received for 115 days across the months of February, March, April, May, and June (2019) revealed gaps in the volume data. On plotting the missing data intervals on 2-D and 3-D dynamic visualizations and heat maps, different missing volume patterns were observed. Three of 115 days were found to have incomplete retrieval of data for more than 50% of total hours, hence, for further investigation of characteristics of data gaps, remaining 112 days are used. Figure 12 show example 2-D representation of missing volume pattern for 112 days. The three days for which more than 50% of data was not received is not included, as it the large continuous

absence on these days might create an obstacle to view the missing volume patterns on the days data was received. Figure 12 shows the aggregation of data loss over 28 day periods. Within this figure it can be seem that data loss is not a completely random process, with volume outages more likely across certain detectors and times of day.

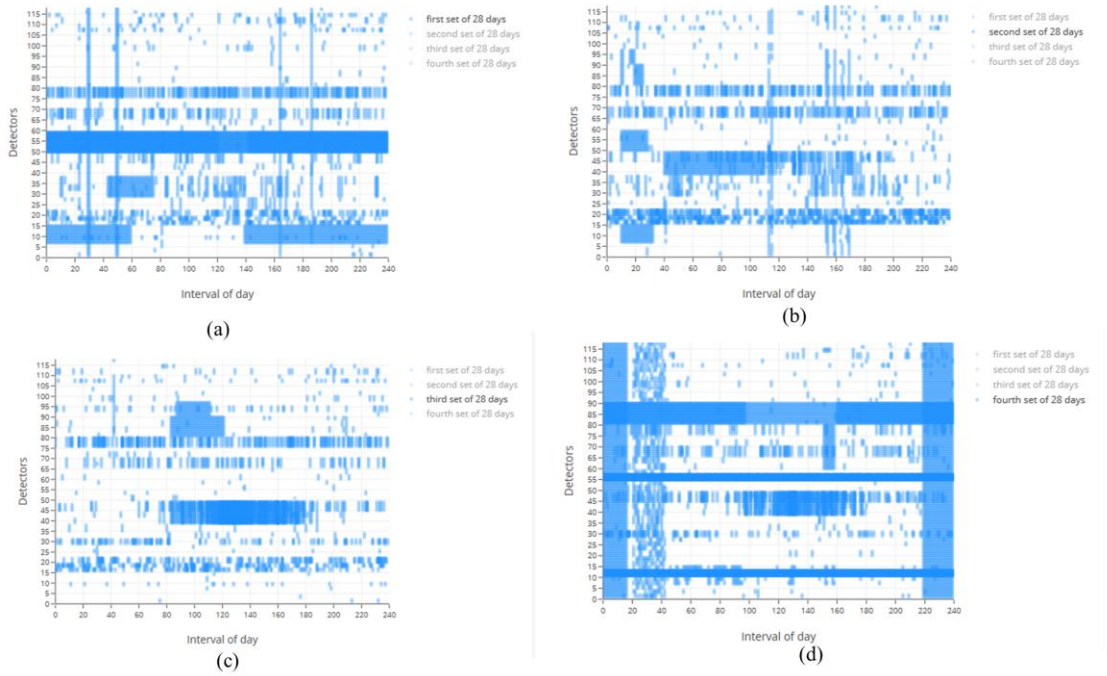


Figure 12 – Interactive 2D visualizations showing intermittent missing data patterns for 112 days. (a) Aggregate data loss, days 1-28, (b) Aggregate data loss, days 29-56, (c) Aggregate data loss, days 56-84, and (d) Aggregate data loss, days 85-112.

Figure 13 shows 3-D representation of missing volume pattern for 115 days. The 3D plots, Figure 13, show the hours with (a) no missing data, (b) one missing six-minute intervals, (c) two-missing six minute intervals, etc. It is seen that there is a significant level of missing data, with one missing interval or all intervals missing within an hour the most frequently occurring. Figures (a-k) show the number of hours with 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10 intervals, where each hour has a maximum of 10 intervals.

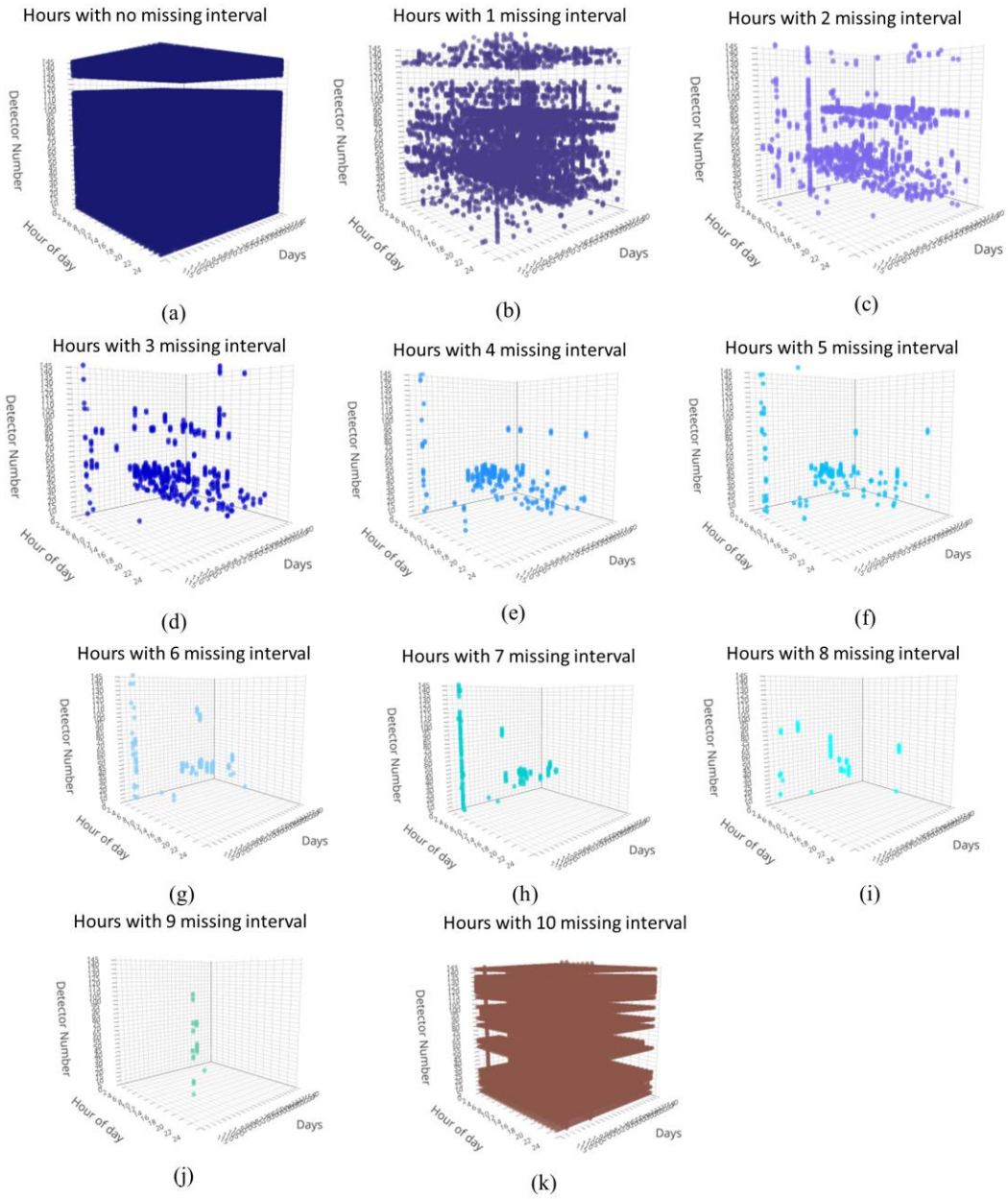


Figure 13 – 3D visualizations of number of missing data intervals, per hour, with 24 hours of a day on X axis, 115 days of Y axis, and detectors on Z axis.

Figure 14 is a heat map showing the presence (in light blue) and absence (in dark blue) of data availability for a typical day. The x-axis is divided into 240, 6-minute intervals (i.e. the number of 6-minute intervals in a 24 hour period), from midnight-to-midnight. The y-axis represents 147 detectors spread across the 15 intersections included in the model.

While at some detectors no data was obtained implying complete data loss, at other detectors intermittent data loss, with varying patterns from day-to-day was observed.

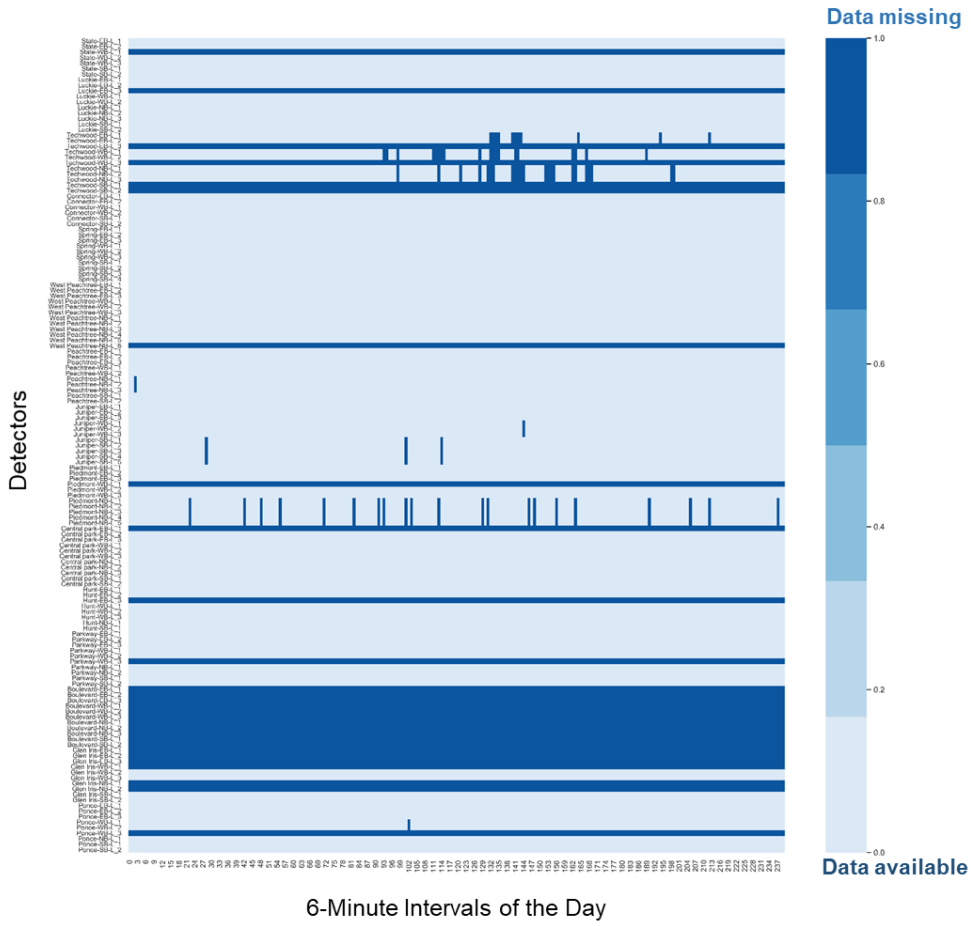
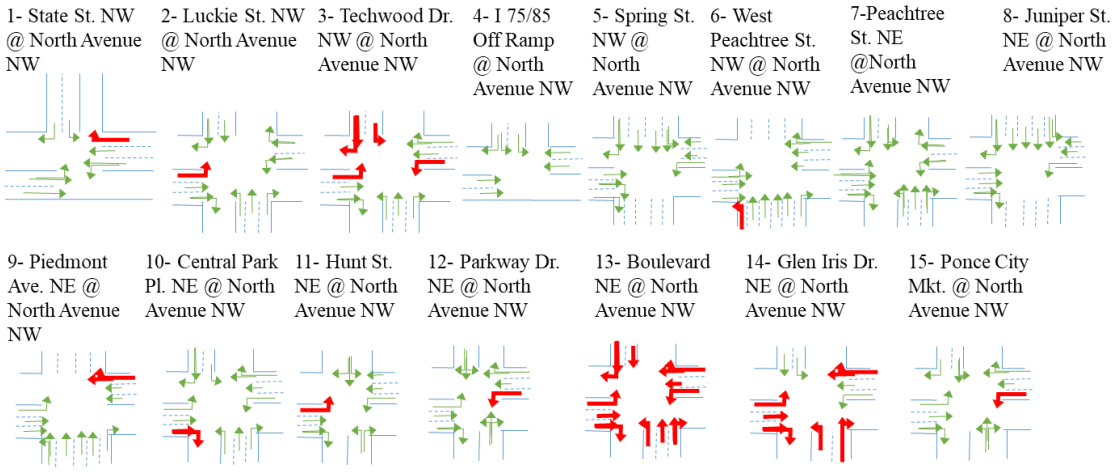


Figure 14 – Missing volume pattern for 147 detectors over 24 hours on February 15 2019. Missing data represents 20.46% of the total day's detections.

Inspection of the missing value patterns throughout the study period revealed that intermittent data loss nearly always includes all lanes of an approach, that is, data is rarely received from some lanes on an approach while not on others. However, where data loss was permanent (i.e. no data was collected over the entire 112 days), it may incorporate all lanes on an approach or be isolated to a single lane on an approach, likely indicating either

equipment failure or that the given lane was not detectorized. Figure 15 summarizes availability of detectors based on studying volume data gap patterns of the 112 days.



Note: The numbers present in the intersection names are associated with the intersection numbering provided in the Google Maps image shown in Figure 2 (Chapter 3).

Figure 15 – Volume detection availability at the study intersections, per lane, per movement. Permanently missing patterns are identified in bold and red.

4.2.2 Signal Data Streams

The current simulation architecture assumes that all messages are received and that the state of a signal does not change between messages. However, in reviewing the signal data stream message loss was observed. The lost or delayed messages lead to longer (potentially significantly longer) GREEN, AMBER, or RED indications than occurred in the field, as well as the skipping of indications. This effect resulted where messages to change to the next indication in the cycle were not received. The outages could be on the order of a few seconds to many hours. Dropped messages could also result in unrealistic timing pattern transitions as a missed change message could result in VISSIM transitioning between non-sequential signal states, i.e., going directly from the GREEN of one phase to the GREEN of the next.

4.3 Sensitivity Analysis Experiment Methodology

Gaps in the data stream feeds to the simulation can result in underestimates in traffic volumes or unrealistic congested due to incorrect signal control. While imputations on data gaps can be used to emulate a more realistic traffic scenario, it is also crucial to understand the potential impact of imputation errors on the simulation results. While data imputations are needed in both volume and signal data streams, to begin with, this chapter focuses on understanding the impact of volume data imputations on model generated performance measures. For this effort signal messages were generated to infill missing data based on historic timings; however, future efforts will explore impacts in potential signal errors.

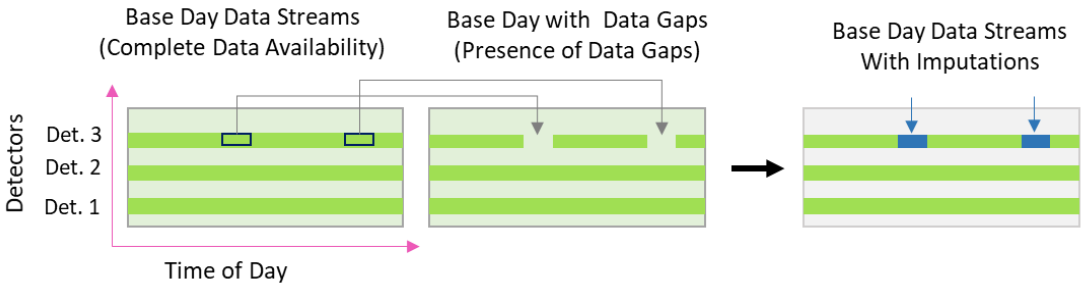
4.3.1 Experiment Design

4.3.1.1 Overview

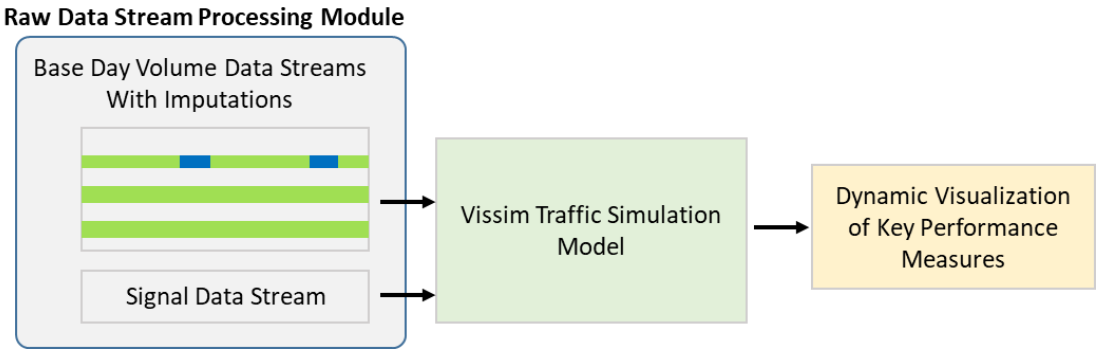
The current data imputation method assumes that imputed data is drawn from historic data, and as such, may differ from actual field conditions. As the current field data has no day with complete data, that is, all 112 days in the data set had both volume and signal timing outages, a composite typical day with complete data was generated. Monday, March 18th 2019, was chosen as the *Base Day*, with missing signal and volume gaps imputed based on historic data, existing signal timing plans, alternate data sources (e.g. count data from a corridor development report), and any other available data sources. This *Base Day* is then considered to be "accurate" field conditions for the sensitivity analysis. In addition, 24-hour data loss patterns are generated based on the 112 days of field data. A random selection of five of the data loss patterns are applied to the *Base Day* to create five *Base Day with Data Gaps* scenarios, inserting errors into the detection data streams for

those detectors and times identified as having data loss in the given data loss pattern. For this effort, potential errors in imputed volumes of 20%, 50%, and 80% are considered. These values are representative of the difference observed in volumes, on the corridor in several locations, over the 112 days. Ten replicate trials are completed for each *Base Day with Data Gaps* gap pattern at each error level, resulting in 160 total simulation trials (including 10 replicate trials of the *Base Day*). Key performance measures are then generated from the simulation tool for a *Base Day* and the *Base Day with Data Gaps*, allowing for an evaluation of the impact of volume data imputation errors. As the data is prepared *a priori* for this experiment the simulation was run faster than real-time to allow reduced processing time.

Figure 16 below schematically shows the design of volume data stream imputations and the logic position in the real-time simulation model architecture. In the real-time simulation model architecture, data imputations are applied at the data input level, prior to injecting the data into the simulation model. Thus, the imputed data will be formatted as the field data and the simulation model will have no dependency on the imputation method.



(a)



(b)

Figure 16 – (a) *Base Day* raw volume data stream, *Base Day with Data Gaps* pattern applied, *Base Day with Data Gaps* with imputed data, (b) Volume imputations made in the raw volume data streams in the *Raw Data Streams Processing Module* of the data-driven simulation model architecture shown in Figure 3.

4.3.1.2 Modelling Likelihood of Missing Data on a Simulated Day Using Unsupervised Learning Method

One of the key items in the experiment is the generation of data gap patterns. Data gap patterns are based on the 24 hour volume data streams received over 112 days spread across February, March, April, May, and June 2019, at 147 detector locations in the study corridor. As discussed, volume data for each detector is received in six-minute aggregate vehicle counts. Over the 112 days, 29 detectors failed to provide any detection data. The remaining 118 detectors experienced at least one an intermittent data loss, ranging from six-minutes (a single bin) to an entire day. To analyze the data loss patterns a binary

representation of the volume data was generated, where 0 represents volume data presence within a six-minute bin and one represents volume data absence. Each detector, per day, was considered a sample for the cluster analysis. Thus, for the detectors that received at least some data there are 13,216 detector samples (i.e. 118 detectors x 112 days). These samples are cluster into groups to allow for a determination of the likelihood of different failure patterns. K-means unsupervised learning is used to cluster the 13,216 samples into groups based on selected key characteristics of the data gap pattern.

Missing Volume Pattern Grouping using K-means Clustering Algorithm: Each detector sample contains 240 binary values (number of six-minute intervals in a day) representing the presence or absence of detection per interval. To find clusters with different presence and absence patterns, features are extracted that describe the primary characteristics of potential patterns. This feature engineering reduces the dimension of each sample from 240 to 7.

The following 7 features were selected based on multiple clustering trials:

- Feature 1 – Total count of intervals without data over 24 hours: captures variation in number of absences
- Feature 2 - Average separateness between intervals without data, i.e. average separateness between absences: captures variation in spread between absence occurrences
- Feature 3 – Total number of clusters intervals (i.e. more than 1 consecutive interval without data) without data: captures number of groups with absence intervals

- Feature 4 – Maximum consecutive string of intervals without data in a 24-hour period: captures variation in maximum cluster of absences group size
- Feature 5 – Median consecutive string of intervals without data in a 24-hour period: captures variation in median cluster of absences group size
- Feature 6 – Sum of intervals without data included in a string with 24-hour period: captures variation in total number of intervals with a group
- Feature 7 – Sum of positions of first interval with missing data point relative to start of day and last missing data point from end of day: captures variation in range in which absence occurrences are distributed within the 24 hr. period

The 24 hour data samples with 0 absences (no data loss) and 240 absences (complete data loss) are separated from the data set based on the feature Total Count of absence occurrences. K-means clustering is applied on remaining instances. To reduce dimensions and to reduce correlation between variables, Principal Component Analysis (PCA) is performed on the clustering dataset. The dataset contains 13216 instances, where each instance has seven features. The dataset is standardized over each of the seven features before conducting PCA by applying Equation 1 on each data point of the seven features. Thus, each feature in the standardized data set has mean 0 and variance 1.

$$x_{standardized} = \frac{x_{original} - \mu_{feature}}{\sigma_{feature}} \quad (1)$$

PCA can provide a low dimension orthogonal subspace in which the variance of the projected data is maximized [78]. To find this subspace, PCA involves eigen-decomposition of the covariance matrix of the data. The eigenvalues provide variance of the projected data on the principal component along the corresponding eigenvector. Thus,

the projected data has highest variance along the eigenvectors that corresponds to highest eigenvalues. Results from PCA conducted on the dataset are summarized next [79].

Eigenvalues from PCA

$$[\lambda_1 \quad \lambda_2 \quad \lambda_3 \quad \lambda_4 \quad \lambda_5 \quad \lambda_6 \quad \lambda_7] = [\,4.39 \quad 1.66 \quad 0.67 \quad 0.21 \quad 0.04 \quad 0.01 \quad 0.001\,]$$

Percent of variance explained by the 7 eigenvectors

$$[0.6271 \quad 0.2372 \quad 0.0960 \quad 0.0304 \quad 0.0061 \quad 0.0026 \quad 0.0002]$$

The percent of variance explained by first three eigenvalues are approximately 63%, 24%, and 10%. PCA results show that ~ 97% of variance in data is captured by the first three eigenvectors space, that is, the first three Principal Components (PCs). Eigenvector matrix provides vector representation of the three principal components along with weights of each feature on respective principal components. Table 1 lists the weights.

Table 1 – Weights on the seven feature for the 3 principal components.

W1	W2	W3	W4	W5	W6	W7
0.47	-0.33	0.25	0.44	0.42	0.46	-0.18
0.04	0.48	-0.44	0.29	0.34	0.18	0.59
0.06	0.14	0.77	-0.08	-0.20	0.10	0.58

The clustering dataset is transformed into the three PCs space, thus, each data instance has only three features, its value on PC1, PC2, and PC3. Each PC is a weighted linear combination of features. Thus, projecting data points in feature space of the three principal components the contribution of variation in data point by all features will be captured. It can be observed that some features that are highly weighted on principal

component 1 have a lower weight value contribution to the other two principal components, such as feature 1, feature 4, and feature 6. The clusters primary reflect the number of intervals without data, indicating the first principle component is reflecting a general likelihood of missing data. Feature 1, 4, 5, and 6 contribute highly to principal component 1, which explains 63% of variance in data. Principle component 2, with significant weighting on feature 2, 3, and 7, appears to reflect the spread of the intervals missing data. And finally, principle component 3 appears to reflect the size of the cluster of intervals missing data. By transforming standardized data points into feature space of the three principal components, the contribution to variation in data points by all features is captured to a total of 97%.

The K-means clustering algorithm is applied on the transformed dataset, excluding detectors that always failed or never failed, as these will a priori be treated as standalone clusters. In this algorithm, the value of K, the number of clusters is set a priori. First, K random data points are chosen as the centers of the K clusters. Next, each remaining data points is allotted to the cluster with the nearest chosen cluster center. After all data points are allotted to a cluster, new cluster centers are calculated. Based on the new cluster centers the data assignment process is then repeated, with all data points newly assigned to the nearest new cluster center. This process is repeated until the cluster centers stabilize, not changing between repetitions. The objective function being minimized through this iterative algorithm is presented in Equation 2, where, $n = 1, 2, 3, \dots, N$, N is the total number of data points, x_n represents the n^{th} data point, $k = 1, 2, 3, \dots, K$, where K is the number of clusters, μ_k represents center (mean) of cluster k, and $r_{nk} =$

$$\begin{cases} 1 & if\ x_n \in k \\ 0 & otherwise \end{cases}$$

This value is the sum of within cluster sum of squared distances. That is, the objective function is the sum of squared Euclidean distances of data points from the cluster center to which they are assigned.

$$J_{min} = min\left(\sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}||x_n - \mu_k||^2\right) \quad (2)$$

The two steps in the iterative process to minimize the objective function are 1) to find cluster assignment for each data point with fixed cluster centers that minimizes J, and 2) to find cluster center that minimizes J keeping the assignment of data points to clusters fixed. The K-means algorithm converges to local minimum for the objective function [79].

This procedure is repeated for increasing values of K to select number of clusters K based on elbow method. The sum of squared distances of data points to their closest cluster center (inertia) is evaluated for varying values of the number of clusters (K). The point of inflection suggests a value for K. Figure 17 shows the graph obtained from the elbow method. The K value after which the sum of squared distance values does not drop significantly is chosen in the graph to be six or seven. After looking at cluster results, seven is chosen.

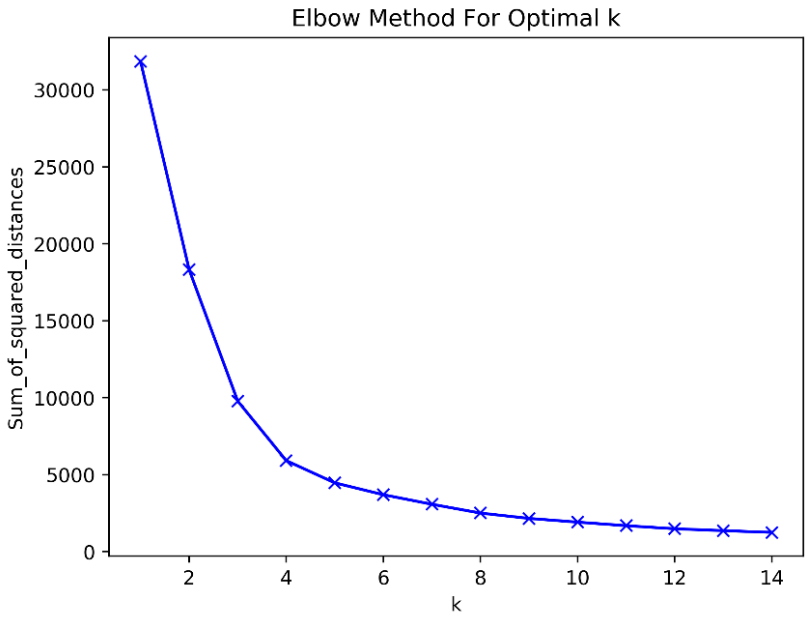


Figure 17 – Elbow method to determine K value. Sum of squared distances versus number of clusters.

Seven clusters are initialized and formed based on the K-means clustering algorithm. Similarity in two of the 7 clusters is noticed and hence, they are merged into one single cluster. Thus, a total of 8 clusters are created, where one cluster includes detector samples with no data loss, another cluster includes detector samples with permanent data loss pattern, and the remaining 6 clusters, formed using clustering analysis. Figure 18 shows two of the clusters.

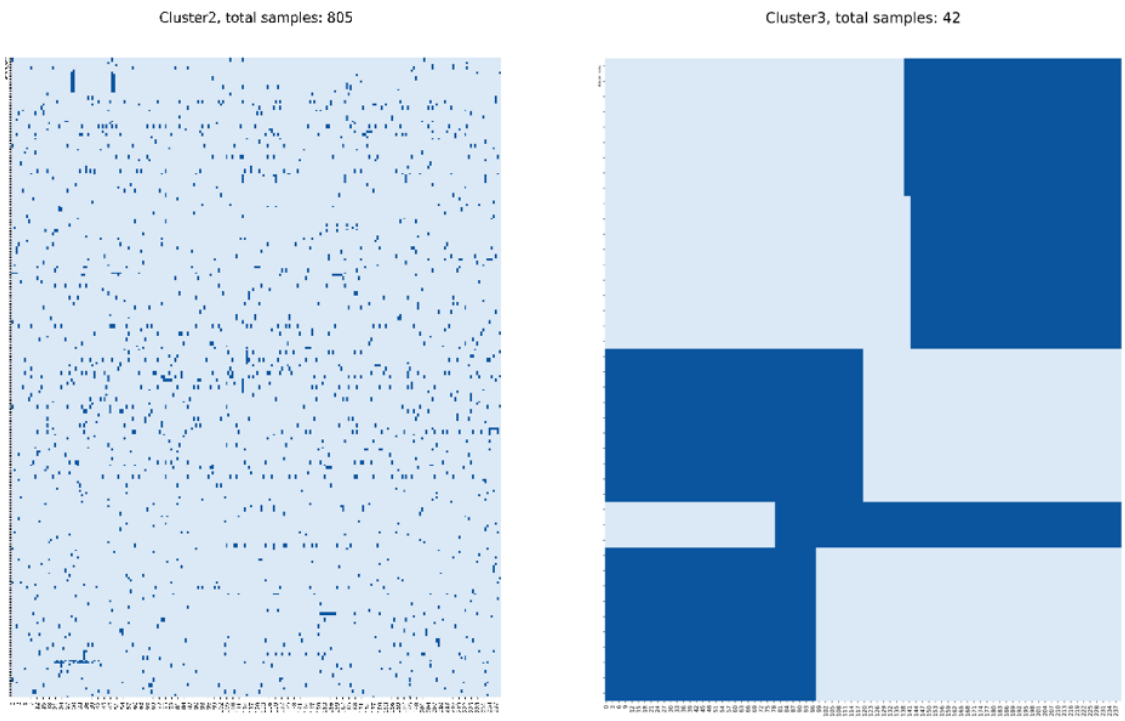


Figure 18 – Two clusters of 24 hour data gap patterns with dark blue blocks as data loss intervals.

Figure 19 shows the K-means cluster assignment on the data points projected in the 3 PC space.

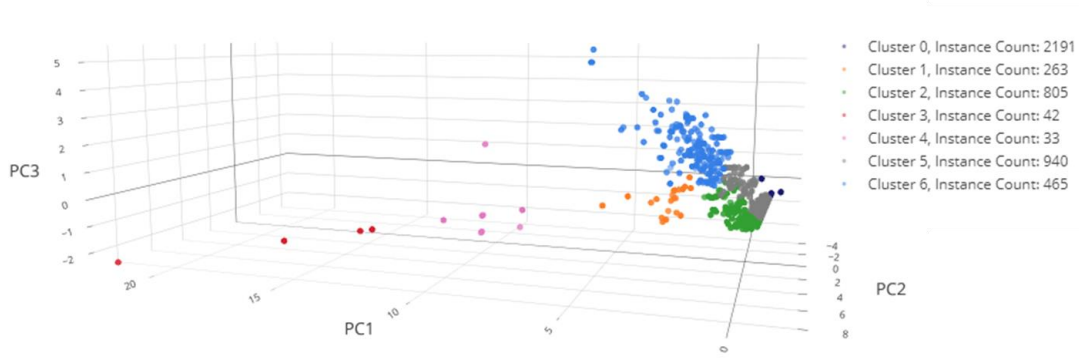


Figure 19 – Clusters of data points projected in the three PC space.

Missing Volume Pattern Generation: The missing volume pattern for a 24 hour day is generated by assigning each of the 147 detectors to a cluster, sampling a data loss pattern

from the assigned cluster, and assigning that pattern to the detector. Detectors are randomly assigned to clusters with a likelihood based on the percentage of the 13,216 samples within each cluster. In addition, when assigning detectors to an intermittent data loss pattern all detectors on an approach are consider together, as observations from the field data showed that in nearly all cases detectors on an approach would exhibit the same intermittent data loss pattern. It was further observed that permanent data loss could be observed at a single detector or at the intersection approach level. To implement assignment of detectors to a data loss pattern (Figure 20), first, a set of approaches with complete data loss are randomly assigned from all available approaches in the corridor, based on the likelihood of an approach having complete data loss, observed to be 0.13 in the 112 day sample. Then, from the remaining approaches, the approaches with complete data loss at a turn lane are randomly assigned, again based on field observation of 0.23 in the 112 day sample. Finally, all remaining detectors are randomly assigned a cluster and a missing volume pattern.

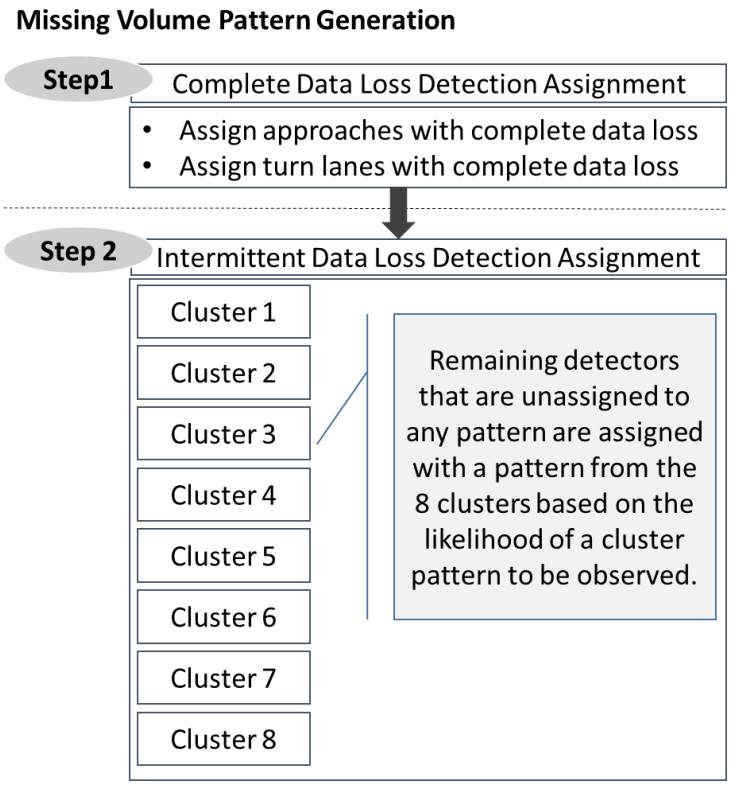


Figure 20 – Missing volume data pattern generation methodology.

4.3.1.3 Simulation Experiment Implementation

The simulation experiment involves three primary steps: 1) generate missing volume pattern, 2) generate *Base Day with Data Gaps* using the missing volume pattern generated in the previous step, and 3) generate *Base Day with Data Gaps* with imputed data.

Simulation runs are conducted for a 3 hour PM peak period (3 PM to 6 PM). For five different missing volume patterns, the average travel time of a vehicle on a route for three values of error in data imputation, i.e., 20%, 50%, and 80%, is compared with that of 0% error in data imputation, i.e., complete data availability. Thus, for each of five missing

value patterns, runs are conducted for 0%, 20%, 50%, and 80% data imputation errors, for 10 random seeds.

4.3.1.4 Data Collection Routes and Detector Outages

The effect of 20%, 50%, 80% error in data imputation (*base day with data gaps*) vs the *base day* on travel time is studied by conducting ten different replicate trials for five different data loss patterns. Six routes along the mainline of the study corridor are selected for travel time comparison. These are:

1. Route 57: Eastbound State St. NW to Ponce City Mkt. (full corridor length)
2. Route 58: Westbound Ponce City Mkt. to State St NW (full corridor length)
3. Route 59: Eastbound State St. NW to Spring St. NW
4. Route 60: Eastbound West Peachtree St. to Hunt St. NW
5. Route 61: Westbound Hunt St. NW to West Peachtree St. NW
6. Route 62: Westbound Spring St. NW to State St. NW

Along with these selected six routes, the effect of data loss on network entry approaches is also studied. For example, missing volume Pattern 1, shown in Figure 21, has three boundary intersection approaches and six internal intersection approaches with permanent data loss on all lanes, and six boundary approaches and seven internal approaches with intermittent data loss on all lanes. The approaches with detector outages for Pattern 1 are shown in Figure 21.

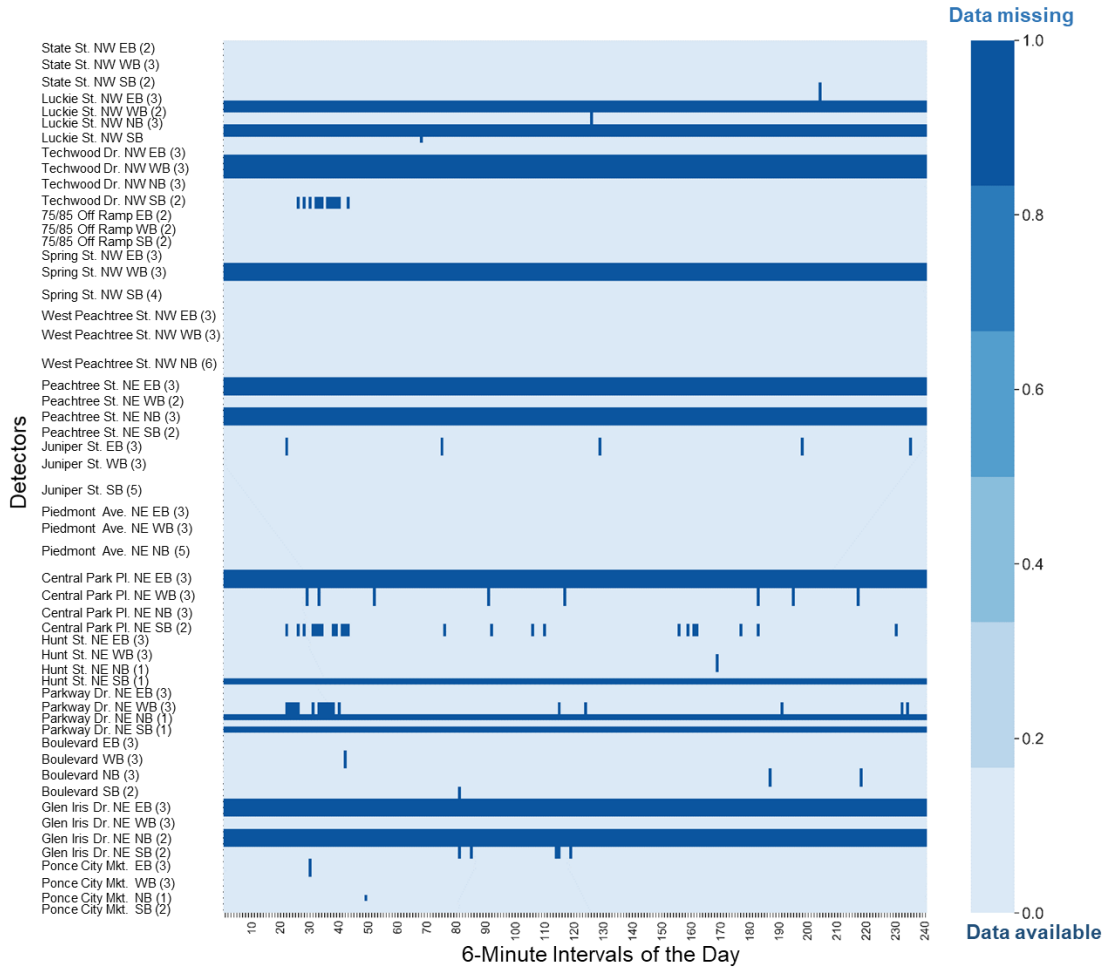


Figure 21 – Missing volume Pattern 1 generated for the sensitivity experiment.

In these experiments, potential error in volume imputation values for the data loss on boundary approach lanes is inserted by adjusting the base volume by 20%, 50%, or 80%. As the simulation is providing vehicle movement once a vehicle enters a boundary link, similar errors may not be applied directly to internal links with detection outages. However, to reflect the potential impact of volume imputation values on internal links the turn movement percentages for the missing lane configurations are increased by 5%, 10%, and 15% for the data imputation error scenarios of 20%, 50%, and 80%, respectively. This reflects that while internal detector data outages do not directly impact the absolute volume through an intersection, they may impact the assigned "volume splits". For example, if the

left only turn lane is missing at an internal approach, the left turn percentage at this approach is increased by 5%, 10%, or 15% for the respective error in volume data imputation scenario of 20%, 50%, and 80%. Table 2 lists the boundary and internal approaches with permanent or intermittent data loss for each of the five patterns.

Table 2 – Summary of Boundary and Internal Approach Lanes with Permanent or Intermittent Data Loss for the Five Detector Outage Patterns.

	Permanent Data Loss				Intermittent Data Loss			
	Boundary Approach/lane		Internal Approach/ Lane		Boundary Approach/Lane		Internal Approach/Lane	
Pattern	Boundary Approaches with Permanent Data Loss on All Lanes	Boundary Approaches with Permanent Data Loss on at least One Lane	Internal Approaches with Permanent Data Loss on All Lanes	Internal Lane Approaches with Permanent Data Loss on at least One Lane	Complete Entry Approaches with Intermittent Data Loss on All Lanes	Entry Lane Approaches with Intermittent Data Loss on at least One Lane	Complete Internal Approaches with Intermittent Data Loss on All Lanes	Internal Lane Approaches with Intermittent Data Loss on at least One Lane
Pattern 1	PTST (NB), PWDR (SB), GIDR (NB), HNST (SB)	LKST (NB), LKST (SB),	LKST (WB), TWDR (WB), SPST (WB), PTST (EB), CPPL (EB), GIDR (EB)	TWDR (EB), PWDR (WB), GIDR (WB)	TWDR (SB), BLVD (NB), BLVD (SB), GIDR (SB), PCM (NB), CPPL (SB)	LKST (NB)	LKST (EB), JNST (EB), CPPL (WB), HNST (WB), PWDR (WB), BLVD (WB), PCM (EB)	PWDR (WB)
Pattern 2	CPPL (SB), HNST (NB), HNST (SB), PWDR (NB), PWDR (SB), GIDR (NB)	PTST (SB), PCM (SB)	SPST (WB), PDAV (WB), CPPL (EB), BLVD (EB), BLVD (WB)	LKST (EB), JNST (EB), PWDR (WB)	LKST (NB), LKST (SB), PDAV (NB), BLVD (SB)	PTST (SB), PCM (SB)	STST (EB), STST (WB), LKST (WB), TWDR (WB), OFRP (SB), SPST (EB), GIDR (WB), PCM (EB)	JNST (EB)

Table 2 continued.

Pattern 3	SPST (SB), CPPL (NB), GIDR (SB), PCM (SB)	STST (SB), TWDR (NB), OFRP (SB), PTST (NB), JNST (SB), CPPL (SB) STST (EB)	PTST (EB)	LKST (EB), TWDR (EB), CPPL (EB), PWDR (EB), GIDR (WB)	TWDR (SB), PTST (SB), PDAV (NB), HNST (NB), HNST (SB), BLVD (SB), PCM (NB)	STST (SB), TWDR (NB), JNST (SB)	SPST (WB), PDAV (WB), PWDR (EB), BLVD (EB), GIDR (EB)	CPPL (EB), GIDR (WB)
Pattern 4	SPST (SB), PWDR (NB), GIDR (SB) STST (EB)	TWDR (SB), WPTST (NB), PTST (NB), PCM (SB)	PDAV (WB), CPPL (EB), CPPL (WB) PCM (WB)	SPST (EB), JNST (WB), HNST (WB), PCM (EB),	STST (SB), TWDR (NB), PTST (SB), CPPL (SB), BLVD (NB), BLVD (SB), GIDR (NB),	TWDR (SB), PTST (NB)	OFRP (WB), PTST (WB), PWDR (WB), BLVD (WB), GIDR (WB)	JNST (WB), HNST (WB), PCM (WB)
Pattern 5	LKST (NB), LKST (SB), BLVD (SB), PCM (NB), HNST (NB), HNST (SB)	OFRP (SB), PTST (SB), JNST (SB), SPST (SB) STST (EB)	SPST (EB), JNST (WB)	STST (WB), LKST (EB), TWDR (EB), TWDR (WB), WPTST (WB), HNST (WB)	STST (SB), PTST (NB), PDAV (NB), BLVD (NB), PCM (SB)	_	OFRP (EB), OFRP (WB), WPTST (EB), PTST (EB), PWDR (WB), BLVD (EB), GIDR (EB), GIDR (SB), PCM (EB)	TWDR (EB), WPTST (WB)

Table 2 continued.

Key for Intersection Abbreviations Used in this Table	1. State St. NW @ Northe Avenue NW	STST	
	2. Luckie St. NW @ North Avenue NW	LKST	
	3. Techwood Dr. NW @ North Avenue NW	TWDR	
	4. I 75/85 Off Ramp @ North Avenue NW	OFRP	
	5. Spring St. NW @ North Avenue NW	SPST	
	6. West Peachtree St. NW @ North Avenue NW		WPTST
	7. Peachtree St. NE @ North Avenue NW	PTST	
	8. Juniper St. NE @ North Avenue NW	JNST	
	9. Piedmont Ave NE @ North Avenue NW	PDAV	
	10. Central Park Pl. NE @ North Avenue NW	CPPL	
	11. Hunt St. NE @ North Avenue NW HNST		
	12. Parkway Dr. NE @ North Avenue NW	PWDR	
	13. Boulevard NE @ North Avenue NW	BLVD	
	14. Glen Iris Dr. NE @ North Avenue NW	GIDR	
	15. Ponce City Mkt. @ North Avenue NW	PCM	

In Figure 22, the mainline and side street routes studied for all patterns are shown on a network schematic. For all patterns, side street through vehicle routes associated with the boundary approaches that observe permanent volume data loss for one or all lanes are shown in Figure 22. (For Pattern 1, Hunt St. NE (SB), which experiences detector outage, has no through vehicles, hence a side street route is not included). The side street route numbers associated with Pattern 1 are:

1. Route 43: Luckie St. NW SB (left turn only)
2. Route 78: Luckie St. NW NB (shared thru-right)
3. Route 86: Peachtree St. NE NB (all approach lanes)
4. Route 67: Parkway Dr. NE SB (all approach lanes)
5. Route 65: Glen Iris Dr. NE NB (all approach lanes)

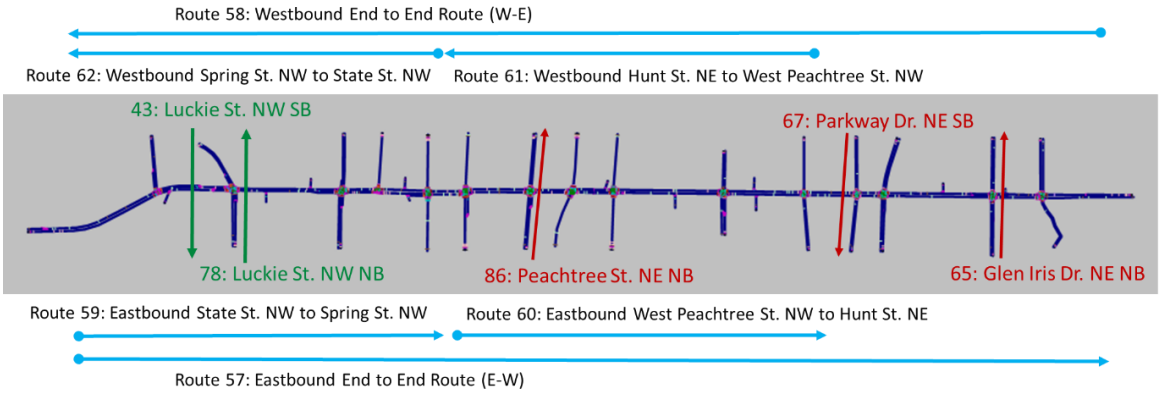


Figure 22 – Routes on which vehicle travel times are studied for Pattern 1 detector outages.

4.4 Results

For missing volume Pattern 1, the variation in mean vehicle travel times for the ten replicate simulation runs for the *base case* (0%), and the *base case with gaps* (20%, 50%,

and 80%), are presented in the Figure 23 boxplot. For each of the six mainline routes, the boxplot presents the variation in the mean travel times of all vehicles that complete the given route, for each of the 10 replicate runs. An upward trend in travel times is observed from 0% to 20%, 50%, and 80% error, on most routes studied. The effect of error in data imputation on travel time is greater on the Eastbound end-to-end route than the Westbound end-to-end route. Route 59 and 60 are subparts of the eastbound end-to-end route 57. Both the western half (Route 59) and eastern half (Route 60) contribute to the over increase from west-to-east (Route 57). An upward trend in the mean travel time is observed on the side street routes 67, 78, 86. While Routes 43 and 65 do not have an increasing travel time with increasing errors. The lack of impact on Route 43 (Luckie St. NW (SB)) is likely as result of base traffic volumes close to saturated condition, thus additional travel time was experienced outside the travel time trap (vehicles were not able to enter the link), while Route 65 (Glen Iris Dr. NE (NB)) observed a very low hourly volume (approximately 110 vph) for the base case, resulting in a minimal travel time increase even given the increase percentage. Figure 24 visualizes the mean simulation vehicle input count at the side streets for the error in data imputation cases – 20%, 50%, and 80% in comparison to the base case. Except for Luckie St. NW (SB) which is at near-saturation state in base case, the volume input counts at other approaches show an increase.

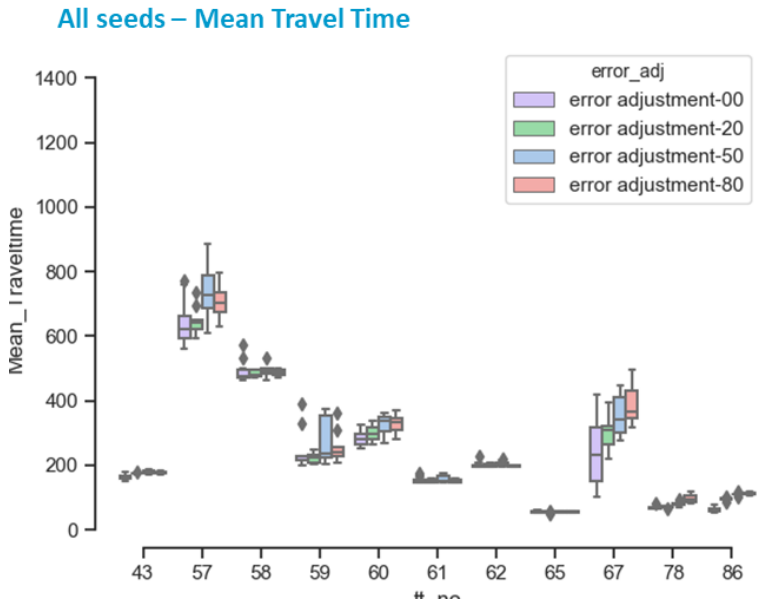


Figure 23 – Boxplot of mean vehicle travel times for data imputation error cases, for the six mainline routes and the side street routes in Pattern 1.

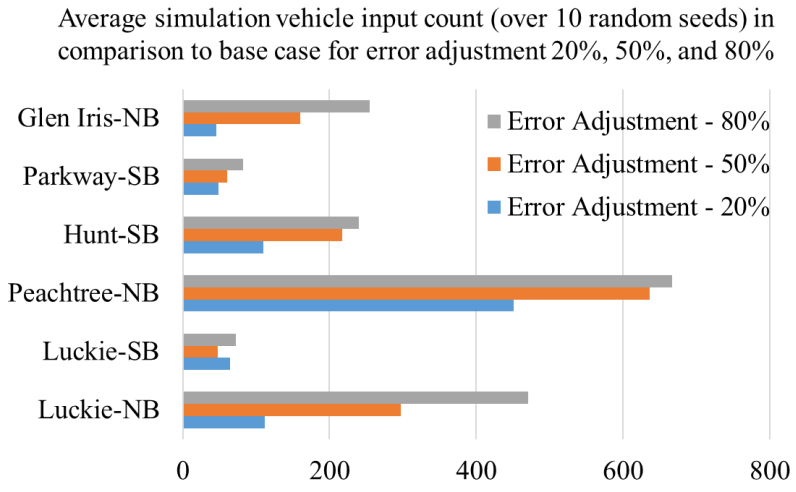


Figure 24 – Mean simulation vehicle input count for data imputation error cases in comparison to the base case on the side street routes that observed permanent data loss in Pattern 1.

The results obtained for 5 different missing volume patterns are summarized in Figure 25.

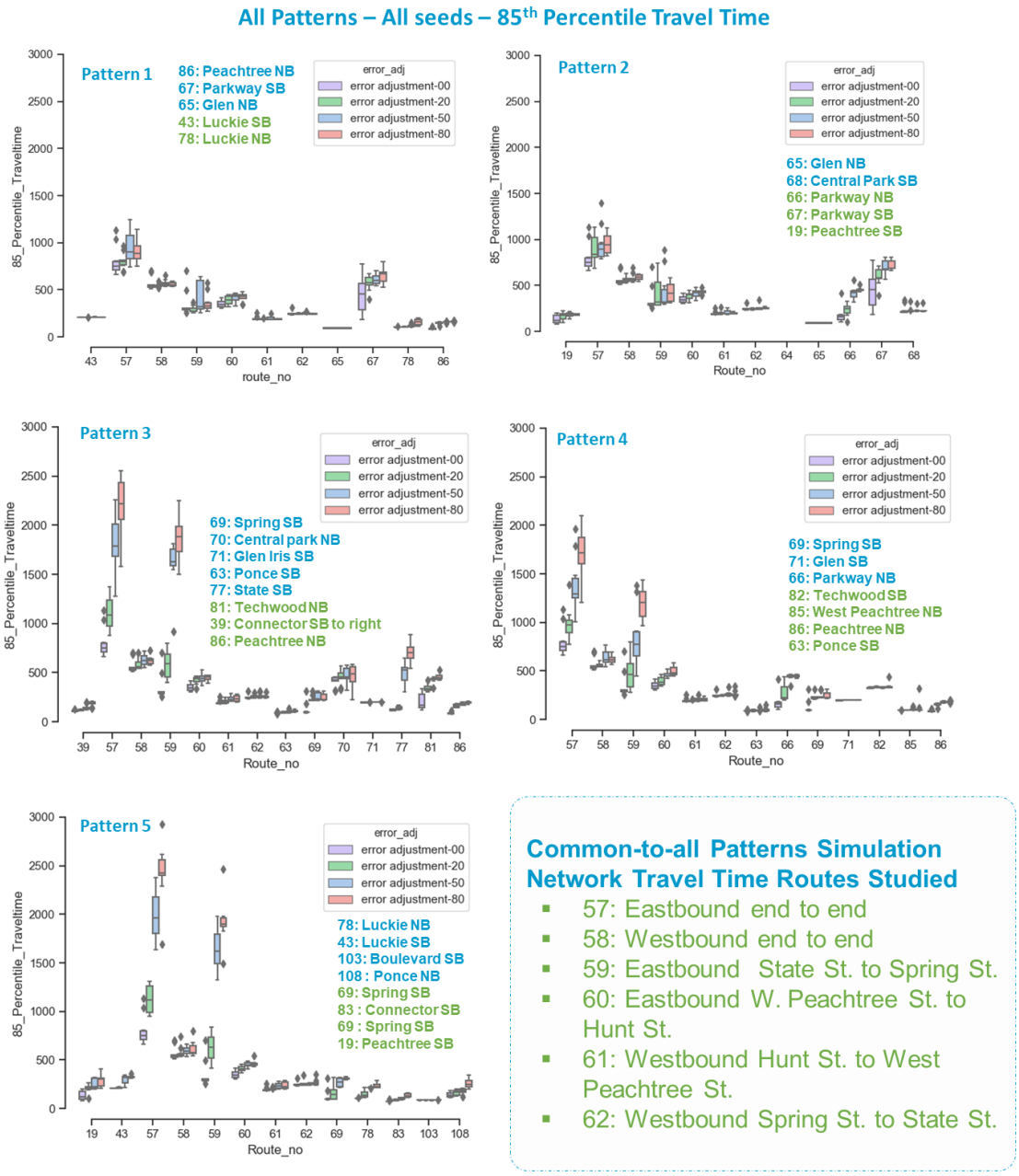


Figure 25 – Variation in 85th percentile travel time values at the mainline and side street routes for the five missing volume patterns.

Figure 25 shows the travel time impact of imputed volume data on approaches that observe permanent data loss as well as the mainline routes.

4.4.1 Intermittent Data Loss

The impact of intermittent data loss on travel time is studied using a statistical test for a selected case. Among the five missing volume patterns simulated, for the PM peak hour period (3 PM to 6 PM), Pattern 3 contained three entry approaches with intermittent volume data loss: Techwood Dr. NW (SB), Piedmont Ave NE (NB), and Boulevard (SB), making this a good scenario to investigate the impact of intermittent data loss. Techwood Dr. NW (SB) volume data loss pattern had more frequent distributed data gaps, hence was selected for studying the impact of intermittent data gap imputations on side street travel time.

4.4.1.1 Case Study: Impact of Intermittent Volume Data Gap Imputations at Techwood Dr. NW

To study the impact of intermittent volume data gap imputations on travel time, result from two simulation scenarios are compared: 1) Pattern 3 with intermittent and permanent data loss - *with intermittent loss scenario*, and 2) Pattern 3 with only permanent data loss - *without intermittent loss scenario*. Pattern 3 with only permanent data loss comprises an additional 10 simulation runs (10 replicates of the permanent detection loss only pattern) in addition to the 160 already discussed. The comparison of these two scenarios allows for an isolation of the impact of intermittent data loss. Results from ten random seed simulation runs for the two scenarios are compared for the three imputations error cases: 20%, 50%, and 80%.

A simulation time of three hours (3 PM to 6 PM) is binned into six-minute intervals. For each detector error scenario, for each replicated run, for each time bin, the 85th percentile travel time for is calculated. This results in a series of 30 (i.e. the number of bins)

85^{th} percentile differences across the three hours, for each replicate trial. These 30 values are then averaged, resulting in a single 85^{th} percentile value for each replicate. Finally, the difference between the replicate value for *with intermittent loss scenario* and *without intermittent loss scenario* is calculated, where the paired replicates have the same seed. Equation 3 provides the described 85^{th} percentile difference between paired replicated trials. The results for all scenarios are shown in Figure 26 and Table 3.

$$Sample\ set\ for\ error\ adjustment\ 20\%\ case\ =$$

$$\frac{\sum_{\forall timebins}\left(85th\ percentile\ tt_{seed,int,20\%}^{timebin}\right)}{No.\ of\ timebins} - \frac{\sum_{\forall timebins}\left(85th\ percentile\ tt_{seed,woint,20\%}^{timebin}\right)}{No.\ of\ timebins}\ \forall\ seeds \quad (3)$$

Here, tt = travel time, timebin = simulation time bin such as (0, 360), (360, 720), etc., seed= simulation trail seed, such as seed 21, seed 22, seed 23, etc., int = result from *with intermittent loss scenario* run, and woint = result from *without intermittent loss scenario* run.

A t-statistics hypothesis test is conducted for each error adjustment in volume data imputation case to test if imputations on intermittent data loss statistically increase the travel time. A one-sample t-test for the mean is conducted to test the alternate hypothesis that the mean 85^{th} percentile travel time difference for ten samples will be greater than 0, with the null hypothesis that the mean 85^{th} percentile difference is 0. In this test, the sample set includes values of all 30 time bins.

Table 3 – Average 85th percentile difference of all time bins for 10 random seed runs for the three error imputation scenarios.

Seed	Error adjustment-20	Error adjustment-50	Error adjustment-80
Seed 1	24.22	20.01	7.05
Seed 2	27.50	61.82	24.78
Seed 3	17.08	44.46	3.90
Seed 4	-33.71	13.14	32.67
Seed 5	0.54	14.22	11.49
Seed 6	14.95	-3.75	27.34
Seed 7	-23.24	-0.53	10.49
Seed 8	-21.13	8.84	24.27
Seed 9	10.47	17.76	60.35
Seed 10	2.82	32.67	75.22
Mean	*1.95*	*20.86*	*27.75*
t-test P-value	*0.291*	*0.005*	*0.002*
Significant level:0.05	*P value > 0.05*	*P value < 0.05*	*P value < 0.05*
	Fails to reject null hypothesis	Reject null hypothesis	Reject null hypothesis

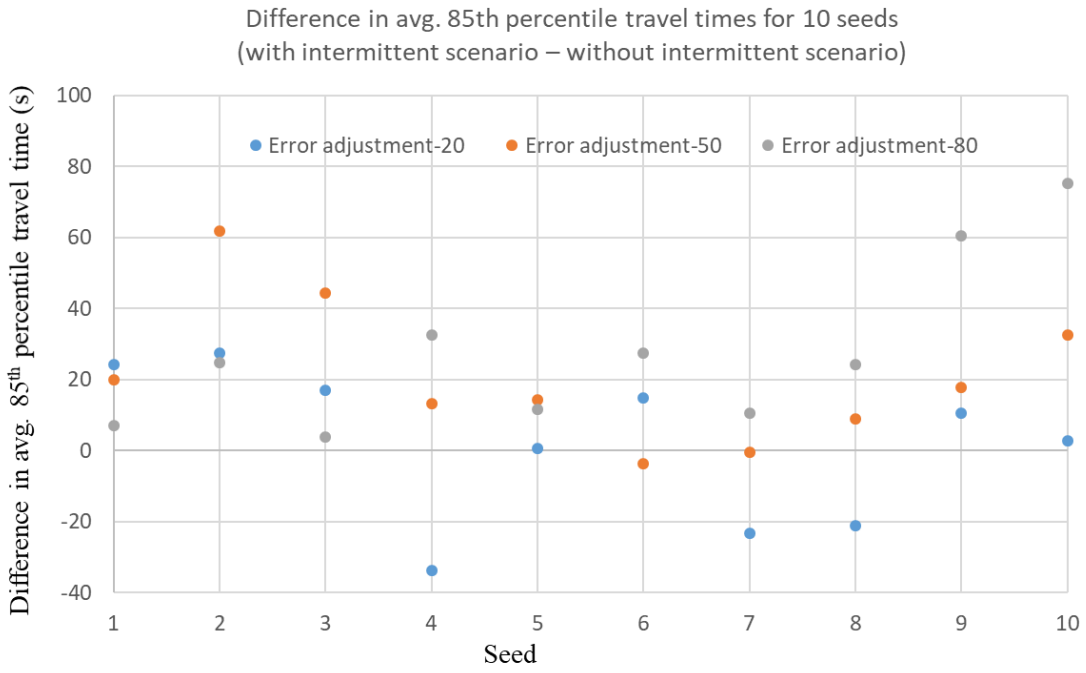


Figure 26 – Variation in average 85 percentile travel time differences with intermittent loss scenario and without intermittent loss scenario of all time bins for different error in data imputation cases.

For an error adjustment of 20%, for a 0.05 significance level, the t-test fails to reject the null hypothesis that there is no effect on travel times due to imputations in intermittent data loss. While for the higher error adjustment, 50% and 80%, the t-test results reject the null hypothesis in favor of the alternate hypothesis that travel times for the *with intermittent loss scenario* is higher than that of the *without intermittent loss scenario.* The hypothesis test results indicate that higher error in data imputation values can impact the simulation generated travel time results at approaches with intermittent data loss patterns when the sample difference values for all time bins are considered. The data plotted in Figure 26 provides a visual confirmation of this finding.

A similar hypothesis test is conducted to test if the travel times are significantly higher for time bins that that follow the time bins with imputed values for *with intermittent*

loss scenario in comparison to that of *without intermittent loss scenario,* that is, does the intermittent error potentially increase as the number of consecutive intervals with missing data increases. For this hypothesis test the average 85^{th} percentile difference in travel time is evaluated only for the 6 time bins that follow the time bins with intermittent data loss. Similar to the prior results the hypothesis test t-test fails to reject the null hypothesis for error adjustment 20% case while for error adjustment 50% and 80% case rejects the null hypothesis in favor of alternate hypothesis that higher travel times are observed in time bins that follow after the imputed time bins. Unfortunately, the small sample size (6 samples) likely limits the ability to statistically distinguish different. Future analysis will expand the study period to all for additional data samples.

4.5 Discussion

The results plotted in Figure 25 for the five missing volume patterns indicate higher sensitivity of travel time on the eastbound routes in comparison to that of the westbound routes, likely an indication of the eastbound direction of travel operating closer to capacity. This is seen in the simulation where at several intersections the simulation is unable to process the full traffic load created by the imputations errors. Mainline eastbound travel time increases are spread throughout the corridor as seen by increases on both sub routes 59 (EB, west of Spring St.) and 60 (EB, east of Spring St.). Although, one section of roadway may be more impacted, for instance, the western portion of North Ave. (Route 59) is the source of most of the eastbound increases in Patterns 3, 4, and 5. Further, it is seen that the travel time increases on eastbound end-to-end (Route 57) and westbound end-to-end (Route 58) differ substantially across patterns, with Patterns 1 and 2 having less impact than Patterns 3, 4, and 5. The pattern travel time differences are due to the

underlying data gaps and turn movement errors specific to each pattern. For Patterns 1 and 2, the less dramatic eastbound travel time increases indicate that approaches experiencing data gaps within these patterns have a lesser impact relative to the missing data in Patterns 3, 4, and 5. For example, the higher impacts seen with Pattern 3 is hypothesized to be a result of data imputation on the Spring St. NW (SB) and 75/85 Off-ramp/Connector approaches. While other approaches in the approach set for Pattern 3 (Central Park Pl. (NB), Glen Iris Dr. (SB), and Ponce City Mkt. (SB)) may contribute towards higher traffic in the eastbound direction, their contribution to the total higher traffic entering the simulation network is substantially less. It is not only total volumes that impacts performance but also position of the intersection. For instance, while Ponce City Mkt. (SB) does contribute traffic, its position as the east most intersection on the corridor lessens its impact on eastbound traffic flow. Figure 27 shows significantly higher increases in entering volumes at 75/85 Off-ramp (SB), Ponce City Mkt. (SB), and State St. (SB) approaches in comparison with that of other missing volume value approaches, for Pattern 3, for the ten different simulation seed runs. The vehicle volume entering at Spring St. NW SB for base case is 4,344 (avg. for all seeds) as opposed to 4,861 (avg. for all seeds) of 80% error imputation case, indicating that much of the increase in demand was not even able to enter the network. If the time period had been extended this traffic queued outside the model would have increase as underlying demand decreased, resulting in even more severe impact due to the imputation error. For the 75/85 Off-ramp SB, for the *base case* the entering volume is 2,461 in comparison to 3,672 for 80% *base case with data gaps*. Similarity an increase in vehicle volume input is observed at Ponce City Mkt. (SB). At State St. (SB), although an increase is observed, it is limited from 50% to 80% error adjustment, indicating

a near-saturation state close to 50% error adjustment scenario. It is likely that that routes that did not process the full increase were already operating close to saturation performance, or that the increase exceed the amount necessary to exceed approach capacity. This leads to the hypothesis that Ponce City Mkt. (SB), State St. (SB), and 75/85 Off-ramp (SB) right turning vehicles contributed primarily to the increase in travel time values on the Eastbound mainline routes.

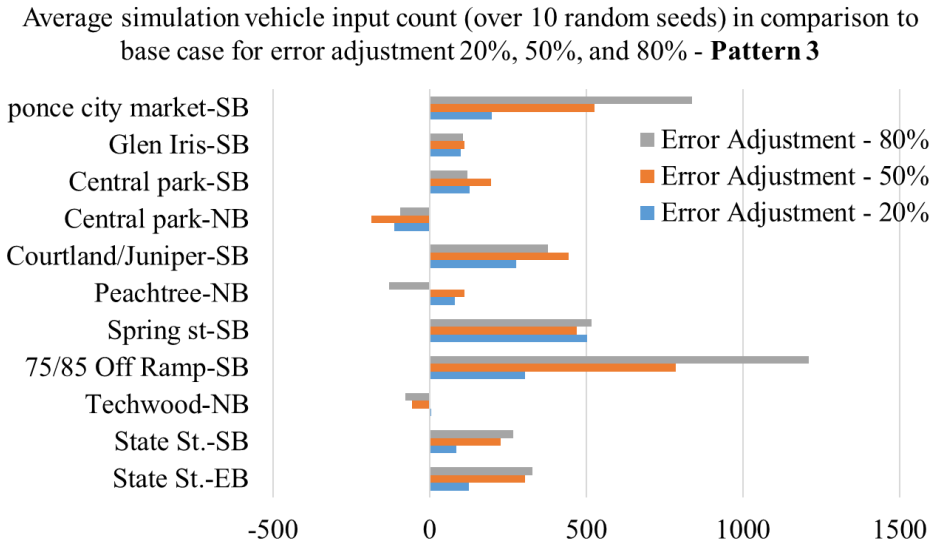


Figure 27 – Mean simulation vehicle input count for data imputation error cases in comparison to the base case on the side street routes that observed permanent data loss in Pattern 3.

In addition, the impact of travel times on most of the side street routes is not as magnified as on the mainline routes. A more sensitive increasing travel time trend for higher error adjustment in data imputation cases is observed on Routes 69, 70, 77, 81, and 86 (Spring St. (SB), Central Park Pl. (NB), State St. (SB), Techwood Dr. (NB), and Peachtree St. (NB)). While this trend is seen for Routes 63 and 39 (Ponce City Mkt. (SB) and 75/85 Off-ramp (SB) turning right), they are less sensitive in comparison. For Route 71 (Glen Iris Dr. (SB)), this trend is not seen. As seen in Figure 27, the average entering

volume value for all random seeds did not show an increase from the base case to the full 80% error adjustment case, although the simulation input volume is increased at Glen Iris Dr. (SB). While for Routes 63 and 39, although the increasing effect exists it is not significant because of very low *base case* volumes, thus even an 80% increase in traffic is a relatively limited number of new vehicles. However, for Routes 69, 81, and 86 (Spring St. (SB), Techwood Dr. (NB), and Peachtree St. (NB), respectively) increasing trends are seen. For Route 70 (Central Park Pl. (NB)), despite the saturated condition at *base case*, the sensitivity in through vehicle travel times could be due to the variation in left turn operations over the 10 seeds. Thus, in Pattern 3 it is observed that some side street route travel times are more sensitive to data loss than other side streets and some side street approaches contribute to increase in total number of vehicles in network with increasing error in data imputation cases in comparison to others.

For Pattern 4, both Glen Iris Dr. NE (SB) and Techwood Dr. NW (SB) are close to saturated condition for the *base case* and hence, do not directly contribute in terms of increased vehicle volume. Here, Ponce City Mkt. (SB), Ponce City Mkt. (WB), Parkway Dr. (NB), West Peachtree St. (NB), and State St. (EB) contribute to the increased entering traffic consistently in the simulation network for different cases of data imputation error. Thus, indicating volume increases at these approach dominate the traffic performance in the network in comparison to other approaches for Pattern 4. Figure 28 shows average increase or decrease in entering volume count for these approaches for 20%, 50%, and 80% error in data imputation levels in comparison to base for Pattern 4.

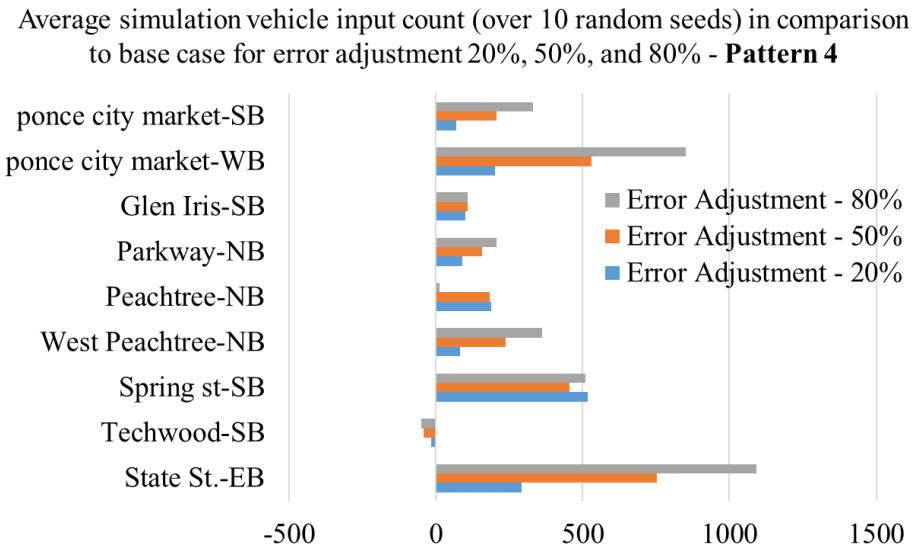


Figure 28 – Mean simulation vehicle input count for data imputation error cases in comparison to the base case on the side street routes that observed permanent data loss in Pattern 4.

Similar analysis of Pattern 5 shows that approaches Spring St. (SB) and Peachtree St. (NB) are not expected to significantly contribute to increased demand in the eastbound traffic or travel, due to their initial low vehicle volume values for the base case. While, others show consistent increase in total vehicle volumes entered in the simulation network for the three error levels. Among this, 75/85 Off Ramp (SB) and Juniper St. (SB) add highest number of vehicles in network in comparison to base level. For Pattern 1, the approaches that contribute highest increase in vehicles in the network are Peachtree St. (NB), Luckie St. (NB) and Glen Iris Dr. (NB), while that for Pattern 2 these approaches are Peachtree St. (SB) and Ponce City Mkt. (NB). Table 4 summarizes the group of intersection approaches that when imputed are most likely to impact the travel times on the eastbound routes in the PM peak hour for the five missing volume patterns studied. In addition, the sensitivity of volume imputation on eastbound routes is categorized as High, Medium, or Low. For patterns where the difference between median of 85% percentile

values for base case, 20%, 50%, and 80% error adjustment cases are greater than 300s the sensitivity is categorized as High. For pattern where these values are less than 200s, the sensitivity is categorized as Low and for patterns where these values are in between 200s to 300s, the sensitivity is categorized as Medium.

Table 4 – Summary of key complete missing lane intersection approaches identified to cause higher impact to travel times on the eastbound routes.

Pattern	Boundary Approaches with Permanent Data Loss at All Lanes	Boundary Approaches with Permanent Data Loss at At least One Lane	Highest Volume Contributing Dominant Approaches That Showed Consistent Increase in Volumes in Simulation	Sensitivity to Travel Time on Eastbound Routes
Pattern 1	Peachtree St. NB, Parkway Dr. SB, Glen Iris Dr. NB	Luckie St. NB, Luckie St. SB, Hunt St. SB	Peachtree St. NB, Hunt St. SB, Luckie St. NB, Glen Iris Dr. NB	Low
Pattern 2	Central Park Pl. SB, Hunt St. NB, Hunt St. SB, Parkway Dr. NB, Parkway Dr. SB, Glen Iris Dr. NB	Peachtree St. SB, Ponce City Mkt. SB	Peachtree St. SB, Ponce City Mkt. SB, Glen Iris Dr. NB	Low
Pattern 3	Spring St SB, Central Park Pl. NB, Glen Iris Dr. SB, Ponce City Mkt. SB	State St. SB, Techwood Dr. NB, 75/85 Off Ramp SB, Peachtree St. NB, Juniper St. SB, Central Park Pl. SB, State St. EB	75/85 Off Ramp SB, Ponce City Mkt. SB	High
Pattern 4	Spring St.SB, Parkway Dr. NB, Glen Iris Dr. SB, State St. EB	Techwood Dr. SB, West Peachtree NB, Peachtree St. NB, Ponce City Mkt. SB, Ponce City Mkt. WB	West Peachtree WB, Ponce City Mkt. NB, Ponce City Mkt. WB, State St. EB	Medium
Pattern 5	Luckie St. NB, Luckie St. SB, Boulevard SB, Ponce City Mkt. NB, Hunt St. NB, Hunt St. SB	75/85 Off Ramp SB, Peachtree St. SB, Juniper St. SB, Spring St. SB, State St. EB	75/85 Off Ramp SB, Juniper St. SB, Ponce City Mkt. NB, State St. EB	High

For Pattern 1 although the combination of Peachtree St. (NB), Luckie St. (NB), and Glen Iris Dr. (NB) together increase the entering volume this value is lower in comparison to that contributed due to the combination of Spring St. (SB), Connector (SB), and Ponce City Mkt. (SB) of Pattern 3 or similarly to that of Pattern 5. This is also because Luckie St. (NB) in Pattern 1 observed permanent data loss on one turning lane as opposed to all lanes. However, it can be deduced that among the studied intersection approaches, Spring St. (SB), Connecter (SB), and Luckie St. (NB) are some of the crucial approaches that impact travel time sensitivities on main line routes.

4.6 Conclusions

This chapter investigates sensitivity of data imputations in volume data streams on the travel times generated from the data-driven simulation model. The experiment reveals the feasibility of the approach and provides insight on the imputation methodology of using historic volume data (*represented by the base day*) to impute another day's missing data streams to drive the simulation model to generate meaningful performance measures. It is readily recognized that this historic approach is simplified and with increasing data more robust historic data approaches could be developed.

From the study, it is noted that the different combinations of the intersection approaches can affect travel times differently. Applying an understanding of missing value patterns to the volume detection data streams of a connected corridor is crucial in understanding the impact of data imputation errors in the generated traffic performance measures. This combination effect can be an attribute of vehicle volumes observed at these approaches in the *base case* and the capacity of these approaches to process additional

vehicles. The effect of different approach combinations is also a result of the location of intersections on the corridor. For instance, in Patterns 3 and 5, the close proximity of Spring St. (SB) and Connector (SB) is expected to have contributed the higher traffic impacts in that roadway section.

The experiment also revealed that imputations at certain intersection approaches are more crucial in affecting travel times of selected routes. For example, from the results obtained, it is interpreted that the combination or individual effect of imputations in volume data streams of intersection approaches such as State St. (EB), Connector (SB), Juniper St. (SB) have higher effect on travel time values on the end-to-end eastbound travel (Route 57) and the eastbound western half of the corridor (Route 59). While it is found that the travel times are more sensitive to data imputations at few selected intersection approaches, it is also seen that data imputation at some intersection approaches do not significantly impact travel times. For example, Parkway Dr. (NB), Central Park Pl. (NB), Central Park Pl. (SB), Hunt St. (SB) etc.

This difference in impacts among intersections and approaches may be used to identify selected intersection approaches to prioritize maintenance and monitoring, particularly were resources are limited. This identification and prioritization of data streams in a corridor can help provide a more robust system and a more efficient application and usage of Smart technologies.

The impact of intermittent data gap imputations on travel times is studied for the test case of Pattern 3, Techwood Dr. (SB). A t-test is conducted on 85^{th} percentile travel time difference of "*with intermittent loss scenario"* and "*without intermittent loss*

scenario" for 10 replicate trials, for the three error in data imputation cases: 20%, 50%, and 80%. The results indicated no significant difference in travel times at the Techwood Dr. (SB) approach for the 20% error in data imputation case. For the 50% and 80% error cases, the t-test results rejected the null hypothesis in favor of alternate hypothesis that travel times for "*with intermittent loss scenario"* is higher than that of "*without intermittent loss scenario".* Similar results were obtained for a t-test conducted for the three error adjustment cases on six time bins that immediately follow the imputed time bins. The test results indicated that a higher error in data imputation values (50% and 80%) can increase travel times of vehicles in time bins that follow the imputed time bins.

Investigation of the real-time volume data streams in this effort highlights potential challenges of using connected corridor data. As seen intermittent data gaps and data aggregation are significant issues. Clearly, as an initial step a successful, i.e., reliable, digital twin requires an investigation and understanding of the connected corridor real-time data stream quality so that potential data issue may be addressed. Future research identified is to develop data imputation methodologies and to investigate reasons for data gaps to mitigate presence of such data gaps. The next efforts in the research will begin the development of methods to address these issues.

4.7 Future Work and Limitations

A primary limitation of this study is that the results obtained may only be directly applied to the North Avenue Smart Corridor, Atlanta, GA, case study, although some generalizations are possible. Further understanding may be gained by investigative studies

at other connected corridor locations. Results from this study indicate that different pattern of intermittent data losses can affect travel times differently. This implies that not all detectors, or detector outages, are likely to have equivalent impacts on the digital twin performance. In additional, while the data streams are investigated for outages they accuracy of the received data has not been thoroughly explored. Similarly, the simulation model has not been field verified.

The study highlights that high frequency connected corridor data streams can have discontinuities. For smart, real-time applications of these data streams it is imperative to develop imputation techniques and understand their effectiveness. The data imputation methodology tested in this chapter is simple. From this study, it is observed that higher errors in the data imputations may impact the travel time performance measures generated by the simulation model. Thus, predictive data imputation methodologies considering weekday and seasonal variations should be developed. In this direction, the development and performance of a methodology that uses LSTM RNN layers to learn from historic traffic volume patterns to impute missing data is studied and presented in Chapter 5. Future work in this direction also entails measuring the reliability of travel time values provided by an imputation enabled simulation model.

CHAPTER 5. DEEP RECURRENT NEURAL NETWORKS FOR CONNECTED CORRIDOR TIME SERIES DATA IMPUTATIONS

5.1 Introduction

Across the world, Smart Cities seek to improve the quality of life for city residents and visitors. A key component of many smart cities is a focus on the transportation system, including connectivity, mobility, safety, air quality, sustainability, etc. Often smart cities utilize smart corridor test beds to explore technology implementations. For example, active studies are ongoing in cities such as New York City, New York; Tampa, Florida; Singapore City, Singapore; Palo Alto, California; Austin, Texas; and London, England; to name a few [80-84]. Typically, a smart corridor is equipped with 'vehicle-to-infrastructure' communications technologies [82], enabling the transfer of significant data between vehicles and the infrastructure. Critical to the success of smart corridor applications are data reliability and the ability to convert the data into actionable information. However, as seen in Chapter 4, a challenge in the use of smart corridor applications can be data streams outages, resulting in data loss ranging from seconds to days. Thus, to successfully implement smart corridor applications it is imperative to find imputation methodologies to fill these data gaps and to understand the application's sensitivity to the imputed data. Given this challenge, the two primary objectives of the research in this chapter are 1) to investigate data imputation through Long Short Term Memory (LSTM) Recurrent Neural Networks (RNNs), and 2) to measure the sensitivity of a given application, a digital twin of a smart corridor in this case, to data imputations error.

The effort reported in this chapter leverages the smart corridor implementation previously described, i.e., the North Avenue Smart Corridor in Atlanta, Georgia, and the developed real-time, data driven, corridor traffic simulation model, subsequently referred to as the North Avenue digital twin. The North Avenue digital twin provides traffic and environmental performance measures, at a near real-time rate [77, 85]. However, investigation of the real-time volume and signal indication data streams obtained from the North Avenue Smart Corridor revealed the presence of losses in some data streams, as seen in Chapter 4. The data losses in volume data streams were observed to range from a few minutes to multiple days. The effort reported in this chapter will focus on the imputation for data loss in the vehicle detector data streams.

Thus, the first research objective, performance of LSTM RNNs to infill data gaps, considers data loss imputation at selected corridor vehicle detectors. For this effort LSTM RNN is investigated using univariate and multivariate time series models. A univariate time series model generates predictions for the data gaps based on parameters estimated using historic time series patterns of the data stream. A multivariate time series model estimates model parameters based on historic time series patterns of the given data stream as well as similar data streams. In this research, bivariate time series models are developed, where the model is trained on two detector data streams from the corridor with similar time series data patterns. Similarity in the time series data in the corridor is measured using Dynamic Time Warping and Hierarchical Clustering algorithms. In the multivariate time series approach predictions to infill instances of data loss use the most recent data available from similar time series. Of specific interest is exploring if utilizing the most recent data from similar data streams provides superior predications over historic data from the data

stream experiencing loss. For the second research objective a simulation experiment is conducted using the North Avenue digital twin corridor to investigate the impact of the univariate and multivariate models data imputation on select generated corridor travel times.

The remainder of this chapter is organized as follows: first, a Literature Review provides a survey of time series similarity measurement techniques and time series data prediction and imputation methodologies; this is followed by the Methodology which describes, 1) Time Series Clustering which is utilized to identify similar detection data streams, and 2) Time Series Imputation which is used to develop the LSTM RNN models used for imputation; next in Model Development and Experimental Design the model development process and three subsequent experiments are described; and, finally presented are the Results and Discussion, Conclusion, and Future Work.

5.2 Literature Review

5.2.1 Time Series Similarity Measures

Time series data, i.e. observations for a variable collected over a period of time, is a common type of empirical data in numerous fields, such as, meteorology, geophysics, astrophysics, financial data, motion data, etc. [86-88]. Time series data is most often processed for one or more of several potential purposes, to: 1) *index*, i.e., to find the most similar time series to a given time series based on a similarity measure; 2) *cluster*, i.e., to find groupings of similar time series; 3) *classify*, i.e., to identify the time series group to which a given time series belongs; or 4) *segment*, i.e., to use time series segments to create a new similar time series [89]. Underling each of these is the ability to identify similarity

characteristics that may be used to describe time series. However, finding a time series similarity measure is considered a complex task, primarily due to the high-dimensional nature of time series data [90], as well as the need to perform efficient and accurate computations. [86, 89]. To evaluate and compare the performance of different time series similarity measures, Serrà et al. grouped similarity measures into four basic categories, namely: 1) *lock-step measures*, where the i^{th} point on one time series is compared with i^{th} point of another time series, e.g., Euclidean distance measure; 2) *feature-based measures*, where a few features (or characteristics) from raw time series are selected for similarity measurement, e.g., Fourier coefficients; 3) *model-based measures*, where similarity in parameters of the time series models are used; and 4) *elastic measures*, where a comparison of one-to-many points between two time series is used [91],e.g., Dynamic Time Warp (DTW) [86].

Lock step measures can include different Minkowski distances (L_p-norm), e.g., Euclidean distance for $p = 2$, Manhattan distance for $p = 1$, and Chebyshev or Maximum distance for $p = \infty$ [91, 92]. Several studies mention lock step distance measures such as the Euclidean distance as being least complex time series similarity measure [86, 91, 93] and common [92]. Wang et al. states that the Euclidean distance similarity measure performs at par with complex similarity measures when the dataset is large [91]. However, Serrà et al. expressed a need for a more quantitative experimental proof on performance of Euclidean distance for measuring time series similarity [86]. With Euclidean distance's computational simplicity considered as its primary advantage as a time series similarity measure [86] the most commonly mentioned drawbacks are its' inflexibility to handle different sequence lengths, varying data frequency, or series data shifted in time [91, 94].

In addition, the Euclidean distance measure does not consider linear correlation, as its value remains the same if the data order is changed [95].

Lock step measures may also include correlation based distance measures, for example, Pearson's correlation distance measures the similarity in linear association between two sequences [92] and is not impacted by data shift in time or scaling [96]. Some other correlation based similarity measure include Spearman's Rank and Kendall's Tau correlation coefficients, that are more flexible to accommodate data noise [97]; however, they have a higher time complexity in comparison to Pearson's coefficient distance [98].

Another approach to time series similarity measure is to extract features from the time series and measure the similarity in the extracted feature of the two time series using lock step measures, for example, using Fourier coefficients [86]. A similar approach is to obtain time series models, for example, auto regressive models, for the two time series and measure similarity in the model parameters [86].

Elastic measures allow for one-to-one or one-to-many point comparisons, making them more flexible than lock-step measures; however, increasing the time complexity of algorithm implementation [92]. An example is Dynamic Time Warping (DTW) [91]. With DTW [99, 100] it is possible to a perform a non-linear comparison of time series with differences in lengths, data shifts in time, data frequency differences, etc. [92]. DTW has been used in several studies [86, 101-103] for similarity measurements and is considered as a definitive approach to measure the dissimilarity in time series [86]. Figure 29 visually highlights the advantage of DTW over a Euclidean distance measure in accommodating time shifts when measuring time series similarities [104]. In the figure, different time series

segments are shown in blue, green, yellow, and red colors. Time series segments with similar patterns warped in time have same color.

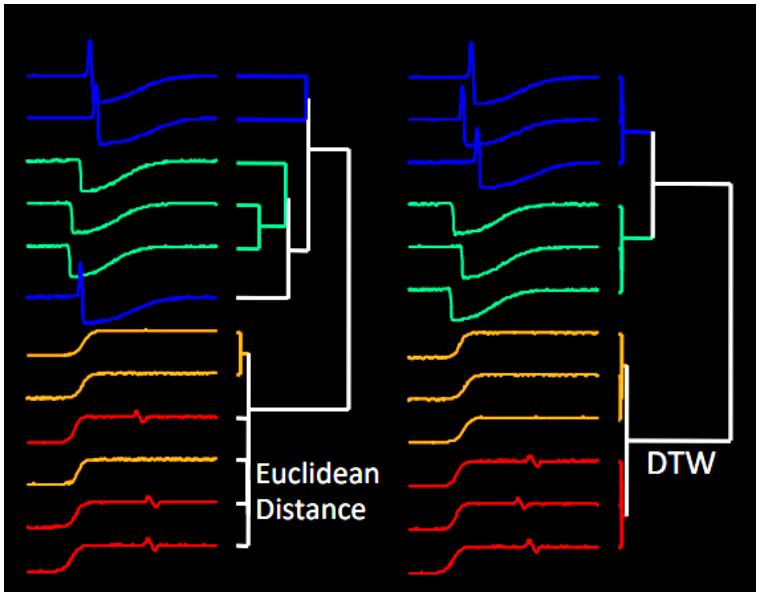


Figure 29 – DTW versus Euclidean measure for determining time series similarity [104].

For example, time series segments with red color have similar pattern and time series segments with blue color have similar pattern. The figure shows the results for grouping similar time series pattern using Euclidean distance measure (left side of figure) and Dynamic Time Warp distance measure (right side of figure). Euclidean distance measure is not able to differentiate between the red and yellow time series segments (bottom left in figure); however, DTW is able to group the time series in red differently than the yellow (bottom right in figure). Similarly, for time series segments in blue and green, the Euclidean distance measure isn't able to capture the similarity in green time series segments and blue time series segments that are shifted in time (mixed grouping seen

at top left in figure) despite their similarities in pattern. This similarity in pattern shifted in time is captured by DTW distance measure (top left in figure).

This is because Euclidean distance measure matches time series one-to-one point comparison while DTW distance measure uses one-to-many comparison, shown visually in Figure 30, allowing for more accurate pattern matching.

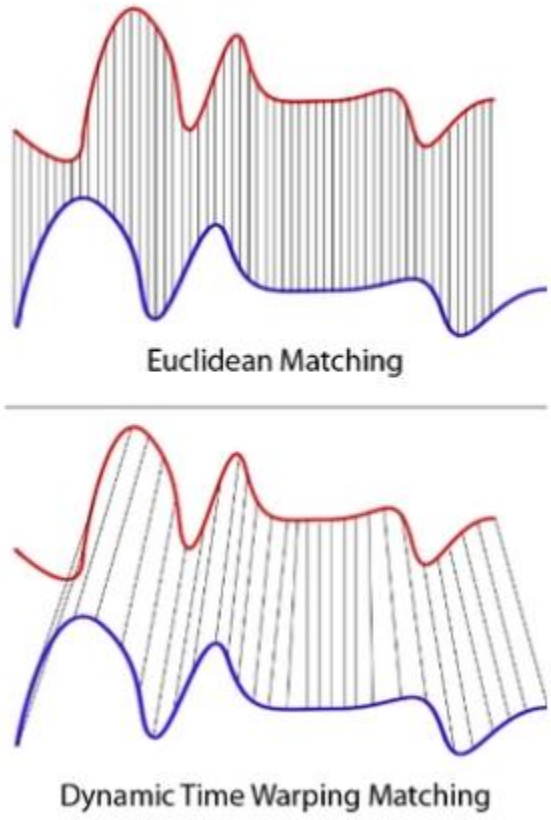


Figure 30 – Euclidean distance measure (one-to-one matching) and DTW distance measure (one-to-many matching) to estimate similarity in time series segments [105].

Several studies have found that DTW often outperforms other similarity measures [86]. For instance, for measuring similarity it was seen that DTW was superior to feature vector in a number of different time series data types [104]. Similarly, studies show that Nearest neighbor DTW time series classification exceeds most other similarity

classification techniques [104, 106]. However, a potential drawback to DTW is that a naïve implementation has quadratic time complexity - $O(n^2)$, making its efficiency a concern. To address this, several modified DTW algorithms have been proposed and tested [100, 107-111]. For example, FastDTW [112] states DTW with time complexity $O(\max\{n, m\})$, where n and m are the time series lengths. However, the approximation used to reduce complexity in FastDTW comes at the cost of computational efficiency, thus this implementation remains less efficient than Euclidean distance measures [92].

5.2.2 DTW in Transportation

One of the earliest applications of DTW in a transportation study was by Chandrashekar et al. in 2011 [113] to estimate the speed of a test vehicle utilizing DTW optimal path alignment between received signal strengths from a moving cell phone handset for the test vehicle and training vehicle traveling the same path. Hi-ri-o-tappa et al. leveraged DTW for pattern classification based automatic incident detection [114]. In this study, DTW is utilized to detect an incident given the upstream and downstream traffic patterns. The study concluded that the DTW based incident detection algorithm performed better than the combined detector evaluation (CODE) [115] incident detection algorithm. In a subsequent study Hi-ri-o-tappa et al. utilized DTW distance measures to study the variance pattern in microscopic traffic state variables such as headway, speed, and flow to predict traffic congestion [116]. In 2014, an incident detection study [117] compared the performance of DTW and the Support Vector Machine (SVM) algorithm, where the SVM algorithm is also used as pattern classifier for incident detection [118-120]. While the study stated both DTW and SVM to be applicable for real-time incident detections DTW was seen to be a simpler concept; however, it was stated that more exhaustive testing of these

algorithms is needed. Taylor et al. used DTW to calibrate car following model parameters by monitoring heterogeneity in driving behavior [121]. More recently, in 2018, Sun et al. applied DTW to identify and analyze pedestrian shockwaves in a bottleneck by studying crowd trajectory data [122]. In 2019, He et al. used DTW to classify time series Unique Cellular Counts (UCC) data obtained from several links of a study area and to investigate the relationship between the UCC time series and traffic volume count data obtained at nearby links [123].

In the present study are utilized the DTW distance measures (FastDTW algorithm) and agglomerative hierarchical clustering to identify groups of similar volume time series data obtained across detectors in the corridor. The results are utilized to obtain multivariate time series LSTM RNNs models. Details of DTW algorithm, the modification in FastDTW, the hierarchical algorithm, and the overall implementation procedure is found in the methodology section.

5.2.3 Traffic Data Imputation Methodologies

In Intelligent Transportation Systems missing data is a common prevailing problem [124]. There are numerous potential causes for detector or transmission network failures [125, 126] such as weather conditions, detector malfunction, machine error, [127], restricted power supply, scheduled maintenance [128], etc. The problem of missing data has been seen across the world [129-131]. Researchers often need accurate and complete traffic information [126] to create successful control mechanisms [130, 132], to predict traffic flows [133], or to estimate travel time [134]. When traffic data is missing data imputation may be a preferred solution over deletion of the entire data stream [135]. Based

on the study objective, missing data on a site can become an obstacle either in understanding characteristics of a certain site that is important for study analysis or in deriving statistical generalizations from the sample data obtained over multiple sites in the study [136]. Several imputation methodologies have been proposed [126, 137]. Most imputation methodologies fall in one of the three categories: prediction based methods, interpolation based methods, and statistical learning based methods [137, 138].

The first category, prediction based methods, seek to utilize potential relationships between historical and future data [71]. Examples of prediction based methods include: auto-regressive integrated moving average method (ARIMA) [139], support vector regression (SVR) [140], and Feed Forward Neural Network (FFNN) [141]. ARIMA, a parametric time-series prediction model, is the most commonly used methodology [142]. ARIMA model performance accuracy depends on three parameters: AR (Auto-regressive order; p), I (difference order; d), and MA (moving-average order; q) [143]. Difference order, d (0, 1, 2, 3 etc.) represents the number of times time series consecutive points are differenced. The three parameters are determined by conducting stationarity tests for different order (d) of the differenced time series (i.e. transformation of the times series into a time series of the difference between consecutive points), and utilizing the Partial Auto-Correlation Function (PACF) and Auto-correlation Function (ACF) of the time series to determine model values of p and q [143]. The model assumes stationarity in the time series data [142]. Several modified versions of ARIMA have been proposed for use in traffic prediction, such as Seasonal ARIMA [144], Vector Auto Regressive Moving Average (VARMA), and Space-Time ARIMA (STARIMA) [145]. While ARIMA is univariate time-series model, STARIMA and VARMA are multivariate models that can utilize

multiple input variables [145]. Kang et al. indicated that parametric models may not capture the underlying relationship in traffic time-series data, where the data is non-linear and stochastic [142]. Three drawbacks noted in the application of prediction based methods for traffic imputation are: 1) prediction methods do not use data after the gap for making estimated imputations [137], 2) in case of longer consecutive missing data, lack of data availability in last window may impact prediction accuracy [137], and 3) most prediction methods only consider temporal information for making predictions while spatial information can also be utilized [71]. Where the traffic volume imputation method is applied to fill gaps in real-time data stream (such as the present study) the first is not applicable.

The second category of imputation methods, interpolation based methods, may be subdivided into two types: 1) temporal-neighboring methods [146] where data from neighboring days for same time period or neighboring time periods of same day, from the same detector, is used for imputation, and 2) pattern-similar methods where data from historical data with a similar pattern, from the same detector, is used [71]. An example of temporal-neighboring interpolation is the historical average model [63] and of pattern-similar interpolation is the non-parametric K-Nearest Neighbor (KNN) model [142, 147]. A drawback of using interpolation-based imputation is its' reliance on traffic pattern similarity at neighboring days or sensors, which may or may not be available [137]. Interpolation based methods also include spline/linear interpolation where mathematical interpolation algorithms are used to infill the missing data using the available neighboring data. This method however does not accommodate the stochastic variation in traffic data [130].

The third category of imputation methods, statistical learning based methods, assume that the observed traffic data can be explained by a probability distribution. Thus, the missing data can be interpreted using the underlying distribution [137]. The methods assume that missing data occurs randomly [148]. Examples of statistical learning based imputation methods are Markov Chain Monto Carlo (MCMC) methods [149], Probabilistic Principal Component Analysis (PPCA) [130], Bayesian Principal Component Analysis (BPCA) [150], and Functional Principal Component Analysis (FPCA) [151]. Statistical learning based imputation methods often perform better than other traditional methods [137]. However, the high reliance on an accurate underlying probability distribution model can be a disadvantage [71].

Given the non-linear and stochastic nature of traffic flow significant effort has been placed in the development of non-parametric prediction methods [142]. Motivated by the success of deep learning (DL) data imputations methods in fields such as natural language processing, image classification, and object identification, researchers are applying DL methods to traffic flow predictions [71] [142, 152-156]. In a recent study, Zhuang et al. used convolutional neural networks (CNN) on traffic image representations where volumes correspond to the pixel values. CNN is then utilized to paint any missing portion of image [71]. Another deep learning method is Long Short Term Memory (LSTM) Recurrent Neural Networks (RNN) [142, 157]. LSTM RNNs are a neural network architecture that learn long-term time dependencies and find optimal time-lags in a sequence, such as a time-series [142]. This LSTM RNN property of maintaining long-term historical data in memory can allow it to effectively capture the non-linear and random nature of traffic flow data [157]. For example, in 2017, Kang et al. investigated the ability of LSTM RNNs to predict

traffic flow at upstream and downstream detector stations for given various combinations of input variables, including: traffic flow, speed, and occupancy [142]. This study found that traffic flow predictions improved with the use these of additional variables. Tian et al. applied LSTM RNN for traffic flow prediction and observed that it performed better than other non-parametric models such as random walk, support vector machines, and stacked auto encoder [157].

5.3 Methodology

From the connected-corridor real-time volume data investigation it was seen that data loss, or gaps, often occurred. Thus, this effort is seeking to fill these gaps, allowing for the execution of the digital twin. As the digital twin operates in (near) real-time the imputation methodology must also operate in (near) real-time. To accomplish this the study utilized prediction based methodologies that utilize recent and/or historic data to inform the prediction model. A challenge of performing real-time prediction based imputation is that data after the gap cannot be used, as would be possible in a post-hoc analysis. In this study, the performance of a bidirectional LSTM RNN layer architecture is tested to model univariate and multivariate volume time series predication models for selected detectors. The multivariate approach considers spatial and temporal correlation by leveraging the time series similarity in detectors at different locations to develop the multivariate LSTM RNN model for the detector missing data (i.e., subject detector).

The methodology includes two primary steps. First, a Dynamic Time Warp similarity measure and hierarchical clustering are used to identify detectors across the corridor with time series patterns similar to that of the subject detector. Second, the subject detector and

those detectors identified in the first step are utilized as input variables to the univariate (uses subject only detector) and multivariate LSTM RNN models. The performance of the univariate and multivariate LSTM RNN prediction models are tested for different consecutive missing data time periods. It is expected that multivariate model will provide superior performance over the univariate model as the length of the detection failure increases, increasing the importance data from similar detectors.

LSTM RNN model performance tests are conducted to address three research questions.

- Research Question 1: How does multivariate model perform in comparison to univariate model for long consecutive data gap predictions?
- Research Question 2: How does multivariate model perform in comparison to univariate model when an unusual traffic pattern is observed?
- Research Question 3: How does multivariate and univariate imputed data impact the digital twin generated travel time for selected routes?

In the next sections of the Methodology, a detailed description of the DTW algorithm, the LSTM RNN, and the experimental design are provided.

5.3.1 Dynamic Time Warp (DTW) - Time Series Clustering

For this study, the DTW distance measure with Hierarchical clustering is used to find similar time series data. The main advantages of DTW over Euclidean distance similarity measures is that DTW accommodates time shifts and different sampling rates between the time series being compared. [92]. A review of this method is presented in the following.

5.3.1.1 Review of Algorithm: Dynamic Time Warp - How it works? (the following explanation is based on [92]

To find the DTW distance between two time series, x and y, with sequence lengths, n and m, respectively, first, a local cost matrix (LCM) is computed. The elements of LCM (i, j) represent the distance between data points of sequence x_i and y_j where i varies from 1 to m and j varies from 1 to n. Generally, this distance is squared Euclidean distance described in Equation 4.

$$d(x_i, y_j) = (x_i - y_j)^2 \tag{4}$$

Next, to find the warping path, $W = w_1, w_2, w_3, \dots, w_K$ across the LCM, a $(n \times m)$ matrix, where K is the length of the warping path, which follows $\max(n, m) \leq K \leq m + n - 1$. There are three constraints for the warping path computation:

1. Boundary Condition: The end points of the warping path are fixed $w_1 = (1,1)$ and $w_K = (n, m)$
2. Continuity: A step from a point $w_q = (i, j)$ in the LCM can go to only adjacent cells (right, up, or up-right). That is, w_{q+1} can be $(i + 1, j), (i, j + 1), or\ (i + 1, j + 1)$ where $q = 1, \dots, K - 1$ and $i = 1, \dots n - 1$ and $j = 1, \dots m - 1$
3. Monotonicity: Each step in the warping path that is moving forward in time in LCM must be monotonically spaced, that is either rightward, upward, or diagonally upward (up-right). This is also seen in constraint 2.

The constraints are visualized on an example LCM in Figure 31.

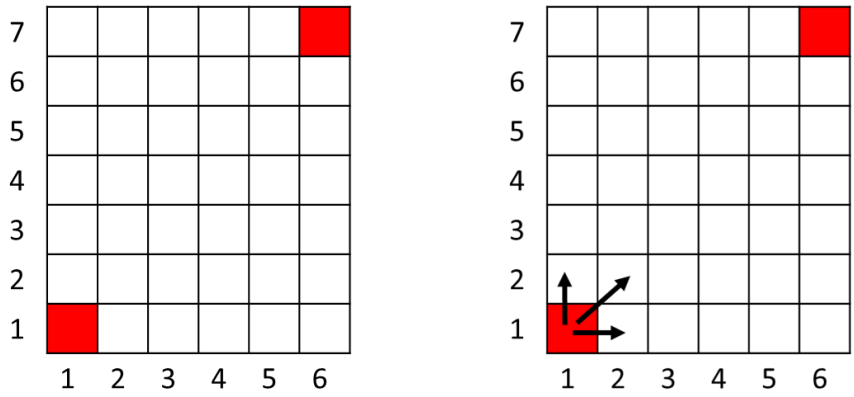


Figure 31 – Constraints for DTW path visualized on an example LCM matrix, (a) end point constraint (boundary conditions), (b) next allowable step (continuity and monotonicity)

Among many possible W paths that satisfy above constraints, warping path W has the minimum cumulative distance cost between the start and end points in the LCM. Dynamic programming is utilized to find the minimum cumulative distance path in the LCM, or the warping path, using the recursive computation in Equation 5.

$$d_{cum}(i,j) = d\left(x_i, y_j\right) + \min\{d_{cum}(i-1,j-1),\ d_{cum}(i-1,j),\ d_{cum}(i,j-1)\} \tag{5}$$

The cumulative total distance cost of the warping path W is the DTW distance between the two time series x and y presented in Equation 6, where, w_k is the distance of k^{th} element in the warping path.

$$DTW(x,y) = min\left\{\sqrt{\sum_{k=1}^{K} w_k}\right. \tag{6}$$

Example: For example [158], consider x and y two time series.

$$x = \{1,1,2,3,2,0\}$$

$$y = \{0,1,1,2,3,2,1\}$$

The LCM matrix for the two time series is obtained by computing the pairwise data point distances, in this case: $d(x_i, y_j) = (x_i - y_j)^2$

The LCM Matrix obtained for this examples is shown in Figure 32.

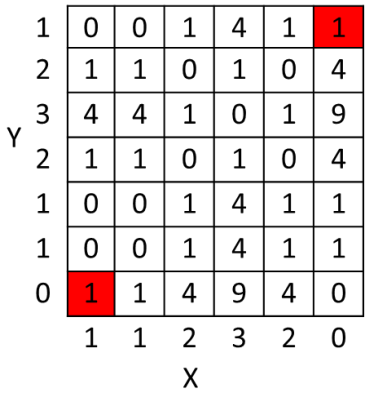


Figure 32 – Local cost matrix (LCM) for the example.

In the next step, "for each cell in the matrix, calculate the cost of arriving from a) below, b) left, and c) below-left, by adding the cell value to the lowest-cost way of arriving at the cell below, left, or below-left, respectively." (*Roelofsen 2018)* For example, the possible values for $w(2,1)$ it will be 1 (from down) with no other potential value, for $w(1,2)$ it will be 2 (from left) with no other potential value, and for $w(2,2)$ will be 1 (from left), 2 (from down), and 1 (from left down). With the potential distances, the warping path

is likely to move from $w(1,1)$ to $w(2,1)$ or $w(2,2)$, as these have the same cost and are both lower than moving from $w(1,2)$. This process is visually seen in Figure 33.

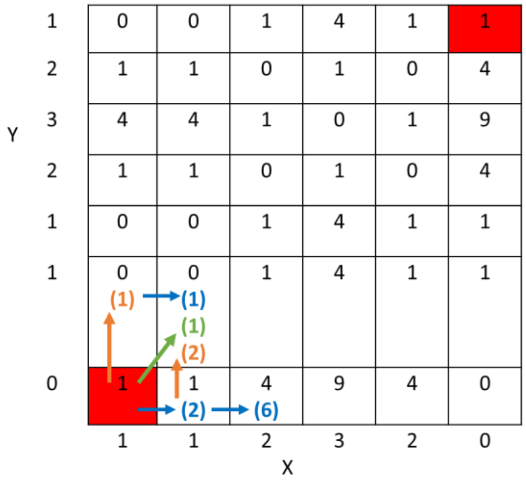


Figure 33 – Distance computation for cells – (2, 1), (1, 2), (2, 2), and (1, 3), where cost to move to the cell from bottom, left, and left bottom is shown in Orange, Blue and Green color arrow and values.

Similarly, for all cells potential values for $w(i,j)$ are computed. Then, the least cost warping path is obtained as the path that has the total minimum cost of travel from the fixed start point to the fixed end point. Figure 34(a) shows the LCM matrix with cost computation for each cell. Cost of each cell from adjacent bottom cell, adjacent left cell, and adjacent bottom left cell is shown in parenthesis in orange, blue, and green color respectively. For each cell, the minimum cost of the three costs from adjacent bottom cell, adjacent left cell, and adjacent bottom left cell is marked with an asterisk. Figure 34(b) shows the two possible DTW path alternates with same lowest cost.

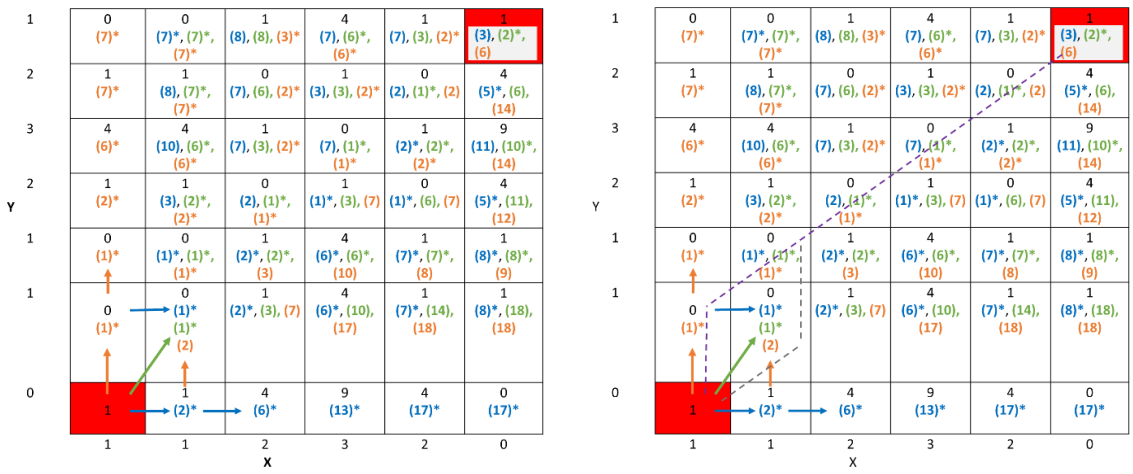


Figure 34 – (a) LCM matrix with cost computations for each cell, and (b) DTW path alternatives with minimum cost.

Figure 35 shows one such possible warping path for the example (DTW for this example visualized using Python, the smaller X and Y graphs represent the starting data).

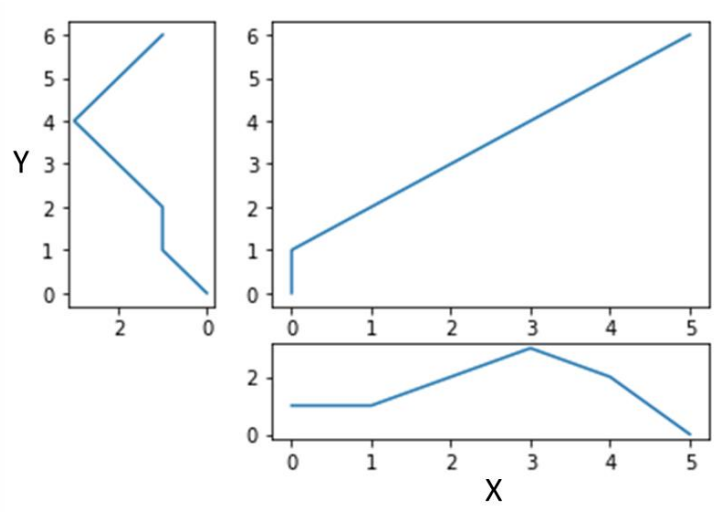
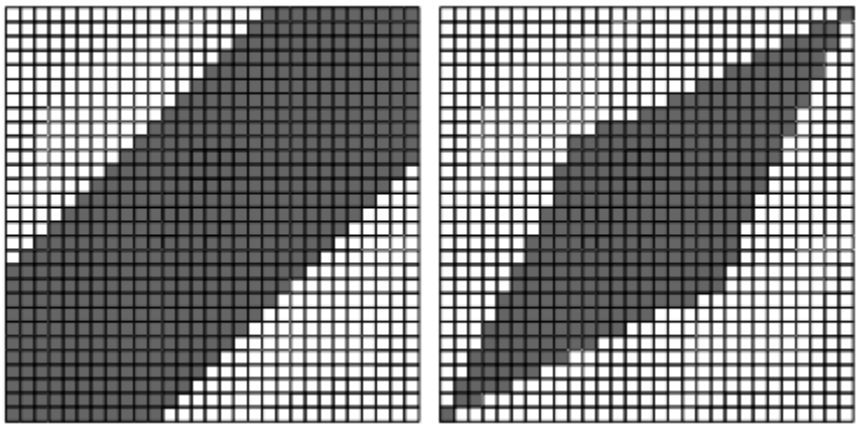


Figure 35 – DTW path between the two time series in the example.

Note that DTW has quadratic time complexity $O(N^2)$ when $|X| = |Y| = N$ where, $|X|$ denotes length of time series X and $|Y|$ denotes length of time series Y, due to the need to compute cost for all cells in the low cost matrix of size N by N. To increase the efficiency

of DTW most proposed modifications can broadly be categorized as: 1) adding constraints [100, 108], 2) abstracting the data [107, 110], and 3) indexing [109]. The modification of adding constraint in DTW algorithm includes constraining the boundary of the DTW path in the LCM. This reduces the algorithm time complexity by a large factor. Two variants of constraint modified DTW algorithm are – Sakoe and Chiba Band and Itakura parallelogram. Figure 36 below shows these two constraints.



(a) (b)

Figure 36 – Shaded portion showing the types of constraints used in (a) Sakoe-Chiba Band (left), and (b) Ikatura Parallelogram (right) to reduce computations needed to find the DTW path [100, 108, 112]

In the second modification of DTW – abstracting the data, to increase the efficiency DTW is conducted on reduced time series representations. In the third modification "indexing", is reducing the number of times DTW is to be run for time series classification or clustering process. Salvador et al. proposed FastDTW that utilizes "adding constraints" and "abstracting the data" to achieve a linear time complexity of $O(N)$ [112]. The approximations used in FastDTW does not assure finding the optimal warping path; however, it is stated to estimate a path quite near to the optimal path. FastDTW algorithm includes three steps: 1) Coarsening – reduce the size of the time series being compared, 2)

Projection – find the minimum cost DTW path for the reduced time series representation (low resolution) obtained in step 1 and project it on a higher resolution, and 3) Refinement – make local changes to refine the projected DTW path to find minimum cost path in high resolution. These steps are used iteratively to find the DTW for highest resolution. Figure 37 [112] explains working of the iterative process in FastDTW visually. The figure shows use of both constraints and data abstractions in FastDTW.

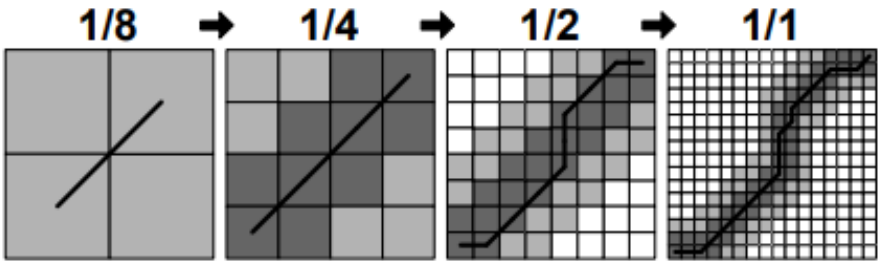


Figure 37 – FastDTW implementation to find optimal DTW path includes – coarsening, projection, and refinement iteratively. The number of cell costs computed in complete process are of the order N relative to N^2 in DTW algorithm [112].

Salvador et al. found FastDTW to perform better in accuracy in comparison to Sakoe and Chiba Band and Data Abstraction modified algorithms of DTW. In this study, FastDTW [159] is used to estimate DTW distances between time series.

5.3.1.2 Application to Study Data

In this study, volume time series data obtained as six minute aggregate counts from 118 detectors across the corridor are clustered based on DTW distances. This effort focuses only on data from Mondays, under the assumption that the same day of the week across multiple weeks should generally have consistent data. This assumption was supported through visual inspection of the data. A 24-hour volume time series, for each detector, was obtained for 15 Mondays over the study period: 4 Feb 2019, 18 Feb 2019, 25 Feb 2019, 4

Mar 2019, 11 Mar 2019, 18 Mar 2019, 25 Mar 2019, 1 Apr 2019, 8 Apr 2019, 15 Apr 2019, 22 Apr 2019, 6 May 2019, 13 May 2019, 20 May 2019, and 27 May 2019. As an example, Figure 38 shows the raw volume time series for two detectors – detector L_3 (lane 3) at Spring St. NW Westbound (WB) approach (Spring-WB-L_3) and detector L_1 (lane 1) at State St. Eastbound (EB) approach (State-EB-L-1) across 15 Mondays. Each Monday comprises of 240 six-minute bins. A similar volume pattern is observed across the different Mondays. Also, it is seen that that the daily patterns are different for the two time series.

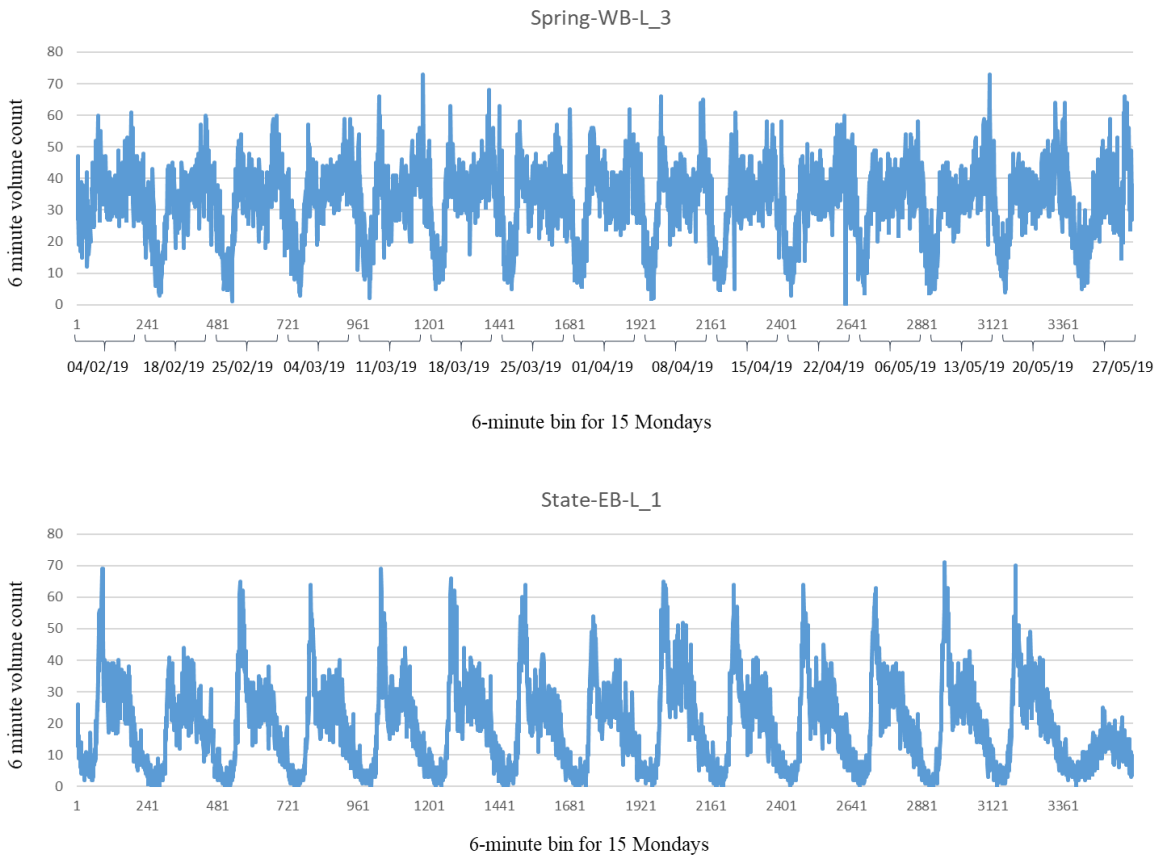


Figure 38 – Volume time series for 15 Mondays at detectors: Spring-WB-L_3 (Spring St., West Bound, Lane 3) and State-EB-L_1 (State Street, East Bound, Lane 1).

The clustering objective in this study is to determine which detectors volume time series data are correlated. Those detectors that are correlated with the subject detector (i.e.

detector with data gaps) will be used to create the multivariate LSTM RNN imputation model. To cluster the detectors, DTW distance based clustering is utilized to indicate correlation between detectors. Mueen et al. suggests z-normalizing the entire sequence as well as z-normalizing every subsequence of the time series sequences. [104]. This allows for a focus on correlated pattern, while significant differences may exist in the absolute values.

For this study, each subsequence is considered as each Monday. However, it is noted that z-normalizing every subsequence of 118 detector time series data adds to the computational expense needed to obtain the DTW measures. In addition, as the 15 Mondays fall within the same general season it is not expected to have significant variation between Mondays within a detector. Thus, z-normalization for every subsequence is not conducted in the pilot experiment, instead, the entire time series sequence is z-normalized.

DTW distance based clustering results of the z-normalized times series sequences were obtained and visualized to evaluate the clustering performance. Figure 39 shows the z-normalized time series for the Spring-WB-L_3 and State-EB-L_1 detectors shown in Figure 38.

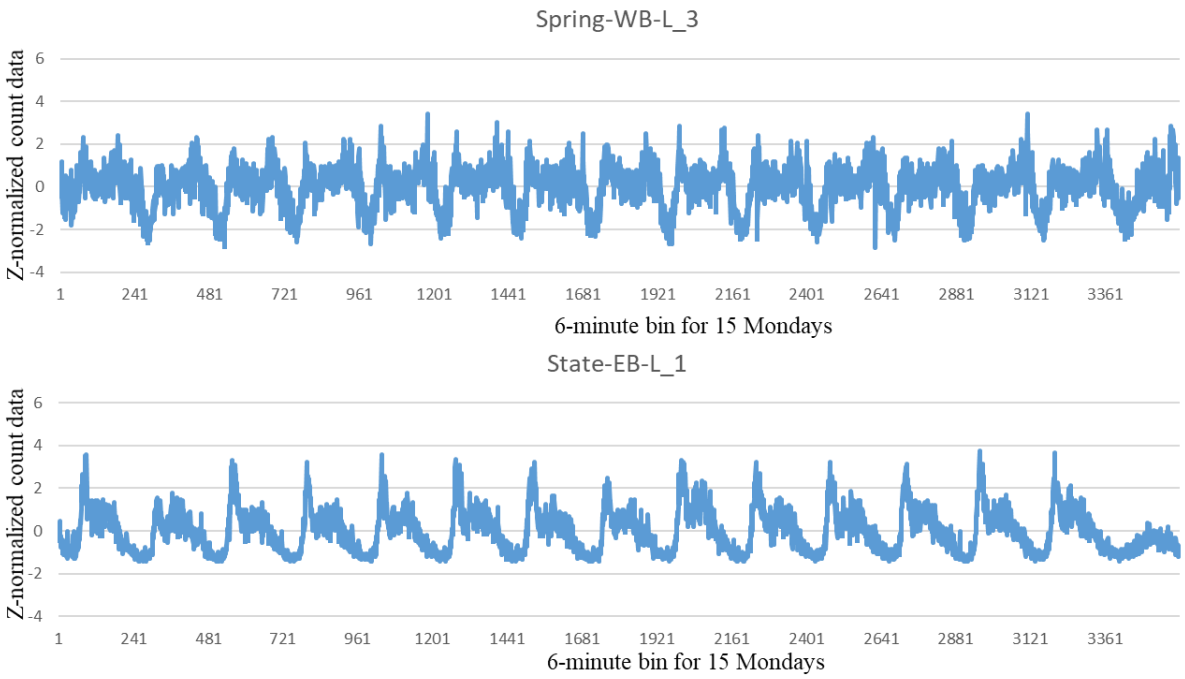


Figure 39 – Z-normalized volume time series for 15 Mondays at Spring-WB-L_3 and State-EB-L_1.

For clustering the z-normalized time series sequences, hierarchical agglomerative clustering is utilized. A short review of hierarchical clustering is provided in the next subsection.

5.3.2 Review of Algorithm: Hierarchical Agglomerative Clustering

Agglomerative Hierarchical Clustering aims to fuse clusters based on the distance between two clusters. When starting a hierarchical agglomerative clustering (hac) each data point is considered as a cluster. A distance matrix is obtained of the distance between every pair of the data points. The two clusters closest in distance are combined. The combined cluster is then considered as a single cluster and the pairwise distances between all clusters are updated in distance matrix. Then, the updated pairwise cluster distances are evaluated to find the closest cluster pair and fused into a single new cluster, which is again followed

by updating the pair wise distances between all clusters. This process is iterated until a single cluster including all data point remains. Thus, the iterative process includes two steps – 1) pairwise cluster distance computation, and 2) merging the clusters with minimum distance. At the end, a tree of the all cluster fusions executed during the process is created that is used to select the number of clusters. This tree displaying the hierarchical clustering analysis process is referred as "dendrogram". Several types of agglomerative hierarchical clustering algorithm exist based on the methodology to compute distance between two clusters such as single linkage, complete linkage, and average linkage [160, 161]. In this study, average linkage criteria is used to find distance between two clusters.

In this algorithm [123], at first, each starting data point is a detector time series, and thus is considered to be an individual cluster. Therefore, the starting distance matrix contains the DTW distance between every detector pair, creating a matrix of 118 by 118 (where 118 is the number of detectors with intermittent data loss). Next, the clusters with minimum pairwise DTW distance are grouped together. The similarity measure of the newly formed cluster is obtained as the average similarity measure between each point in the two original clusters. For example, assume a cluster with one time series (1) and another cluster with one time series (3) have been grouped to become a new cluster X (1, 3). Then, in the next step, the pairwise distance (similarity measure) of a cluster with one time series (9) from the new cluster X (1, 3) is evaluated as in Equation 7.

$$DTW(1 \cup 3, 9) = \frac{1}{2}\,[DTW(1,9) + DTW(3,9)] \tag{7}$$

The formula for the similarity measure between clusters for average linkage, derived from the generalized from *Lance-Williams dissimilarity update formula* [160, 162] is described in Equation 8, where, C_i, C_j, and C_k are cluster i, j, and k; $\alpha_i = \frac{|i|}{|i|+|j|}$, and $|j|$ is the number of data points in a cluster C_j.

$$d(C_i \cup C_j, C_k) = \alpha_i d(C_i, C_k) + \alpha_j d(C_j, C_k) \tag{8}$$

A simplified representation of methodology to estimate similarity between clusters (cluster G and cluster H with n_G and n_H data points) using average linkage criteria for hierarchical clustering is shown in Equation 9 [163].

$$d(G, H) = \frac{1}{n_G * n_H} \sum_{m \in G} \sum_{n \in H} d_{mn} \tag{9}$$

Results of the hierarchical analysis is obtained as a dendrogram diagram, which shows the hierarchical relationship between the clusters [164]. Complete dendrogram obtained for average linkage hierarchical clustering is shown in Figure 40. On Y-axis is the distance measure (DTW) and the X-axis is the detectors. The height of the bars of two points that are joined together provides the similarity measure.

To choose optimal number of clusters or to evaluate validity of clustering results, two visual methods commonly used are the elbow method and silhouette analysis [165, 166]. The elbow method involves performing clustering analysis for several choices of number of clusters and plotting total within sum of squares distance of the data points in

all clusters against the chosen number of clusters. The cluster number (k) that observes a bend in the plot is chosen as the number of clusters [165, 166].

Silhouette analysis includes computing the average silhouette coefficient for clusters for clustering results with different values of k and choosing the k that gives maximum average silhouette coefficient. Silhouette coefficient for a sample is obtained by calculating mean intra-cluster distance that is mean distance of the sample with all data points in the cluster (a) and nearest-cluster distance (b) that is mean distance of the sample with all data points in the next nearest cluster. Silhouette coefficient (s) for the sample is evaluated using Equation 10 [167].

$$s = \frac{b - a}{max(a, b)} \tag{10}$$

For this study, with a large dataset, the computational load to obtain clustering results for the different number of clusters required to conduct elbow method and silhouette analysis was high. In this pilot study, an approximate rule of thumb to cut the dendrogram to obtain clusters with longest branches is applied [166]. However, to automate clustering process silhouette analysis or elbow method would be preferred if computational efficiency is not an obstacle. The dotted line in Figure 40 shows the threshold cut off used in this study to obtain 11 clusters. Through visual observation it is seen in Figure 40 that 11 clusters are created over this section by cutting the dendrogram at the shown position with dotted line.

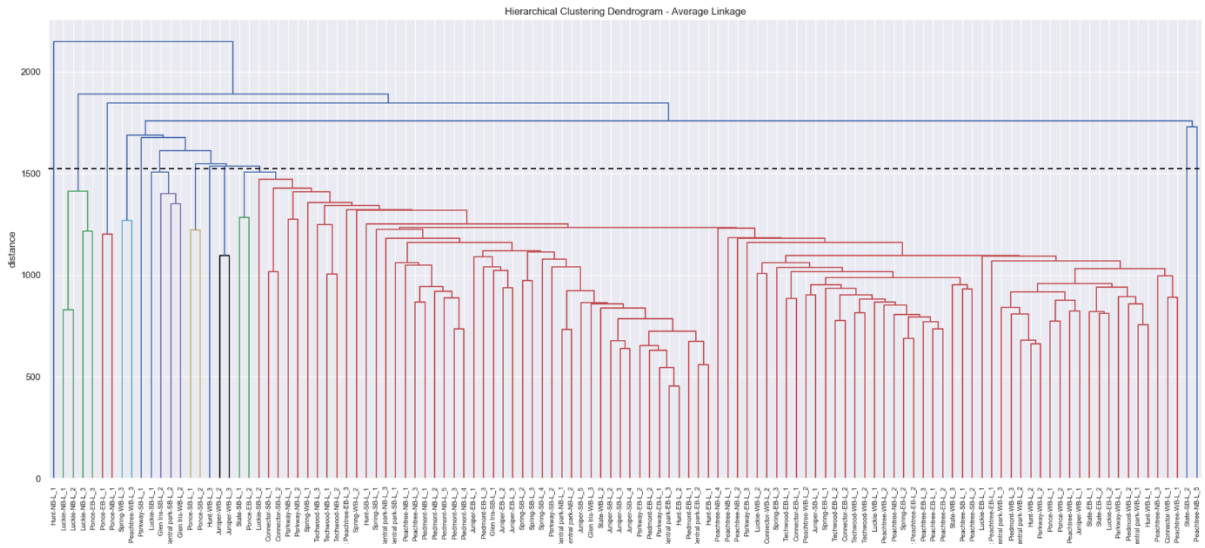


Figure 40 – Dendrogram of average linkage hierarchical agglomerative clustering result on 118 detectors with DTW similarity measure.

The next sub-section presents clustering results and its' implementation for building LSTM RNN imputation model.

5.3.3 Time Series Clustering Analysis and Results

The cluster analysis identified similar detector patterns across fifteen Mondays. This method helped in efficiently filtering the detectors with similar traffic. Observing the Dendrogram, when considering all 118 detectors, 11 clusters are chosen when cutting dendrogram tree. Figure 41 visualizes four of the obtained clusters. The pattern difference between cluster 3, 4, 5, and 6 is visible. Cluster 6 has the largest cluster size among all 11 clusters.

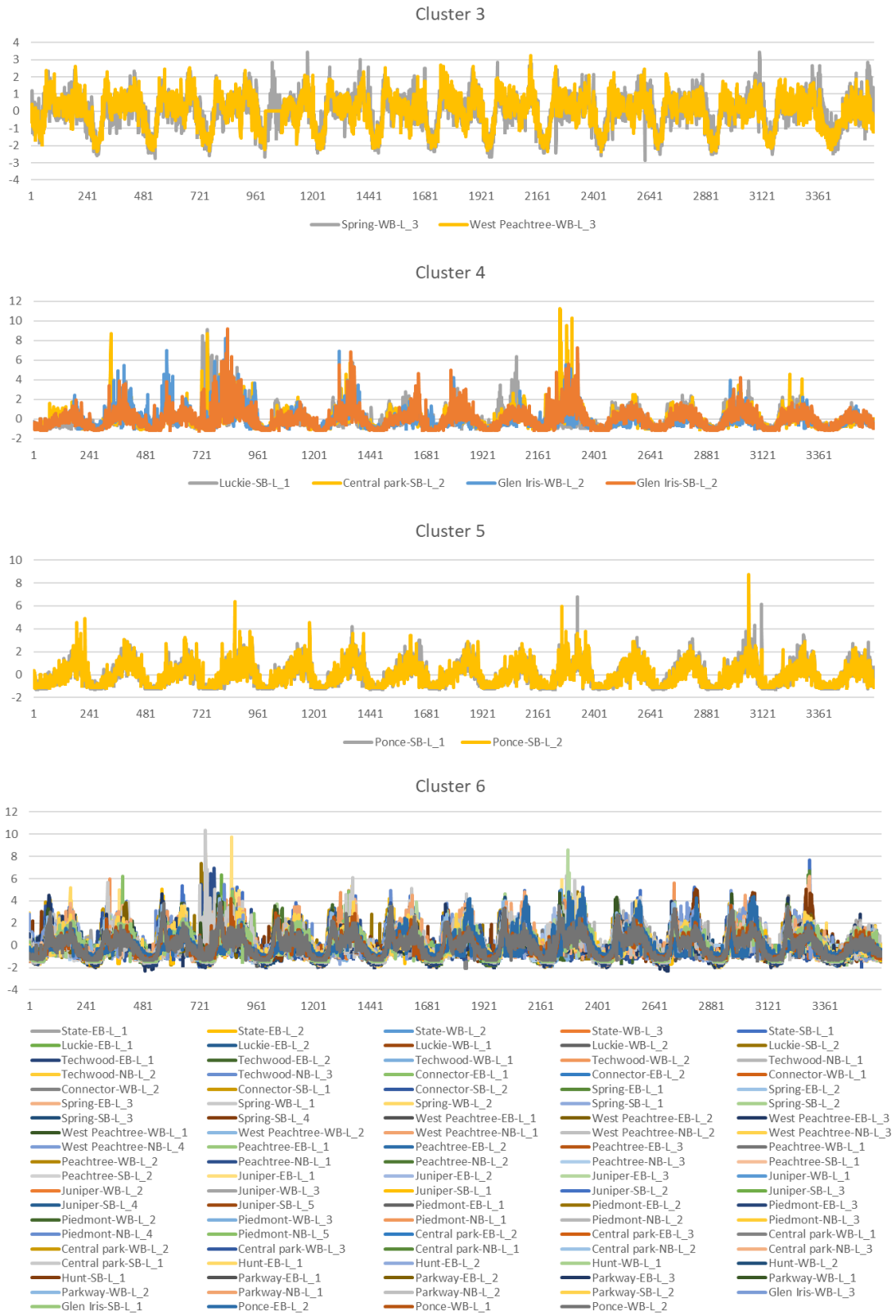


Figure 41 – Z-normalized time series cluster of detectors obtained from hierarchical clustering using DTW.

The similarity in detectors identified is utilized to develop the multivariate LSTM RNN model. The next section provides a review of the LSTM RNN algorithm and describes the experimental design implemented to compare the univariate and multivariate LSTM RNN model performance.

5.3.4 Time Series Imputation

For time series imputation, a type of deep learning algorithm, Long Short Term Memory (LSTM) Recurrent Neural Networks (RNN), is utilized in this effort. Deep learning refers to training Neural Networks to predict a value based on inputs provided [168]. RNNs fall under the category of Neural Networks (NN) [169] with LSTM RNN as a special type of RNN. The next sections include a brief review of Neural Networks, RNN, and LSTM RNN.

5.3.4.1 What are Neural Networks?

Neural networks assist in machine learning of a process by studying several training examples [170]. A neural network architecture includes multiple layers. Typically these are the Input, Hidden, and Output Layers, as shown in Figure 42. Each layer consists of interconnected neurons or 'nodes'. The input layer provides the initial variable information (for example - x_1, x_2, x_3, x_4 in Figure 42) which is then processed by the hidden layer utilizing connections. The hidden layer is connected to the output layer that provides the output values (y) [169].

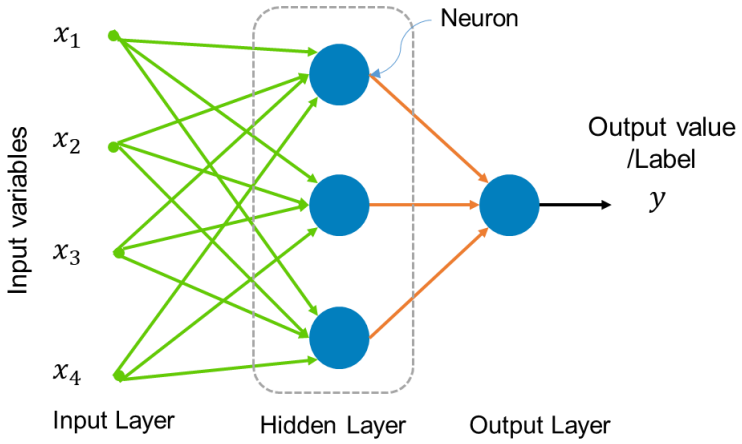


Figure 42 – Standard neural network architecture [171].

Each connection has a weight and each node has a bias value and weight. Each node is composed of a summation and an activation function. The summation function provides a linear combination of the input variables based on the weights and bias value. The activation function is then applied to this linear combination. The Activation function regulates the output behavior of the neuron [172]. Figure 43 shows the working of a single neuron where, w_1, w_2, w_3 are weights for input features x_1, x_2, x_3 and b is the bias for the single neuron or node shown in the figure, z is the linear combination of weighted input values and bias, and a is the output from activation function. The Activation function here is the sigmoid function (σ).

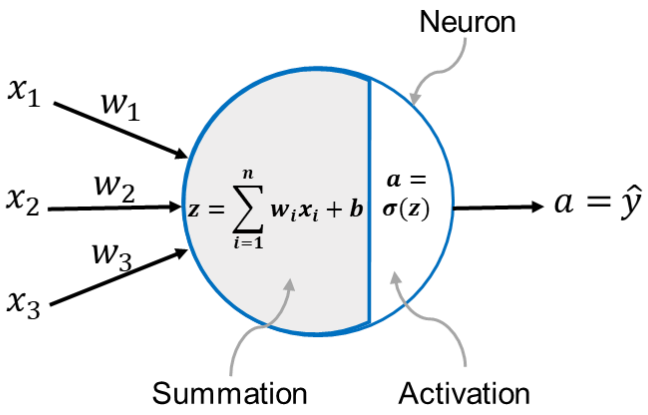


Figure 43 – Neuron consists of a summation function (linear combiner) and an activation function [173].

The Activation function is crucial to neural networks as it adds a non-linear component to the neuron output [174]. A sigmoid activation function is shown in Figure 43. In Figure 42 the NN architecture has one hidden layer. In practice, several NN layers can be used to increase the accuracy of the mapping of the input values with the output value.

5.3.4.2 Cost Function, Gradient Descent, Forward Propagation, and Backward Propagation [168][1]

Notations: Consider a neural network with n nodes and l layers that have sigmoid activation function and is trained using training sample of size m. A parameter, for example weight w associated with node n and layer l for training sample m is represented as $w_n^{[l](m)}$.

Cost Function: When all weights of nodes are stored in matrix w and vector b, output $\hat{y}$ can be represented as in Equation 11.

[1] *Theory interpretation and equations presented in this section are drawn from the Coursera course Neural Networks and Deep Learning – instructed by Andrew Ng.

$$\hat{y} = \sigma(w^T x + b) \tag{11}$$

Where, sigmoid function is $\sigma(z) = \frac{1}{1+e^{-z}}$

The convex loss function to measure error for logistic regression is in Equation 12. This provides the error for a single training sample.

$$L(\hat{y}, y) = -(y log\hat{y} + (1-y)\log(1-\hat{y})) \tag{12}$$

For a neural network training optimization, the objective is to find parameters w and b that minimize the cost function J (Equation 13), the average total loss for all the training samples.

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$
$$= -\frac{1}{m}\sum_{i=1}^{m}\left(y^{(i)} log\hat{y}^{(i)} + \left(1-y^{(i)}\right) log\left(1-\hat{y}^{(i)}\right)\right) \tag{13}$$

The Gradient Descent algorithm is used to conduct the optimization to find parameters w and b for the cost function J.

Gradient Descent in NN: To find w and b that minimizes $J(w,b)$ gradient descent is used to find an approximate value of global optimum for $J\ (w,b)$. In the gradient descent algorithm, at first, w and b are assigned initial values. Then, the value of the parameters w and b are updated iteratively until the global minimum for $J(w,b)$ is reached (algorithm converges). Each update step for the parameter value is shown in Equation 14 and 15. The

measure or update size (step size) in the parameter values depend on the learning rate represented by α.

$$w := w - \propto \frac{\partial J(w,b)}{\partial w} \tag{14}$$

$$b := b - \propto \frac{\partial J(w,b)}{\partial b} \tag{15}$$

Figure 44 shows the gradient descent step visually.

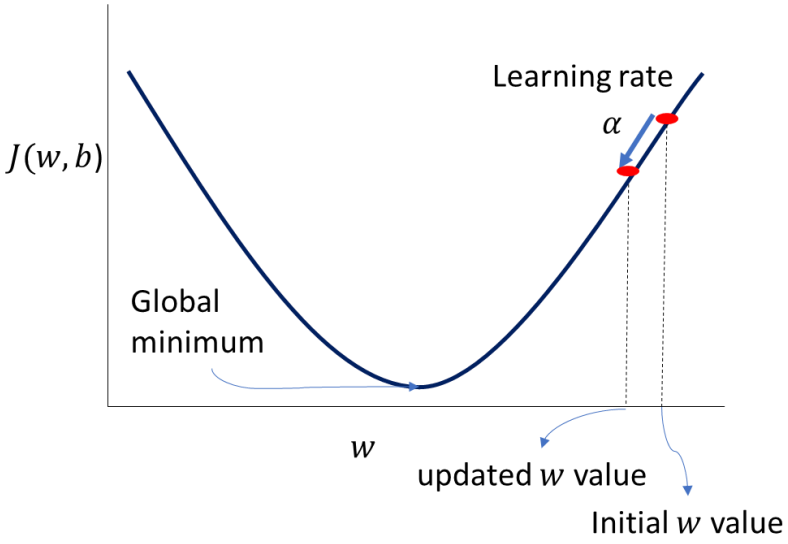


Figure 44 – A step in gradient descent to find w corresponding to global minimum (modified from diagram in Coursera course by Andrew Ng) [168].

Computation of derivatives of the cost function, $J(w,b)$ with respect to weights w and biases b is required for every step of gradient descent to update the parameter values and to decide the direction of the descent. Computations of neural network gradient descent process include two primary components: 1) forward propagation, where output value is computed based on current parameter values of w and b, and 2) backward propagation,

where the derivatives of cost function with respect to the parameters are computed to update parameter values w and b for next step of gradient descent.

A brief example of a gradient descent update using forward and backward propagation is presented using for a simple neural network that has one input layer – layer 0, one hidden layer – layer 1, and one output layer – layer 2. Representation for the number of nodes (n) in each layer is $n^{[0]}$, $n^{[1]}$, and $n^{[2]}$. A single output node is assumed, that is, $n^{[2]} = 1$. The parameters for this neural network are $W^{[1]}, b^{[1]}, W^{[2]}, and\ b^{[2]}$, where $W^{[1]}, W^{[2]}$ represent matrix that contains weights for layer 1 and 2 for m samples and $b^{[1]}$, $b^{[2]}$ represent matrix that contains biases for layer 1 and 2 for m samples. Gradient descent for cost function $J(W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]})$ includes iteration of following steps:

1. Step 1 – Compute prediction/output using forward propagation: $\hat{y}$
2. Step 2 – Compute derivatives using backward propagation - $\frac{\partial J}{\partial W^{[1]}}, \frac{\partial J}{\partial W^{[2]}}, \frac{\partial J}{\partial b^{[1]}}, \frac{\partial J}{\partial b^{[2]}}$
3. Step 3 – Update parameters using Equations 16, 17, 18, and 19.

$$W^{[1]} = W^{[1]} - \alpha \frac{\partial J}{\partial W^{[1]}} \tag{16}$$

$$W^{[2]} = W^{[2]} - \alpha \frac{\partial J}{\partial W^{[2]}} \tag{17}$$

$$b^{[1]} = b^{[1]} - \alpha \frac{\partial J}{\partial b^{[1]}} \tag{18}$$

$$b^{[2]} = b^{[2]} - \alpha \frac{\partial J}{\partial b^{[2]}} \tag{19}$$

Step 1: Forward propagation computation formulas for this example are described in Equations 20, 21, 22, and 23.

$$Z^{[1]} = W^{[1]}X + b^{[1]} \tag{20}$$

$$A^{[1]} = g^{[1]}(Z^{[1]}) \tag{21}$$

$$Z^{[2]} = W^{[2]}A^{[1]} + b^{[2]} \tag{22}$$

$$A^{[2]} = g^{[2]}\left(Z^{[2]}\right) = \sigma\left(Z^{[2]}\right) = \hat{y} \tag{23}$$

Here, $g^{[1]}$ and $g^{[2]}$ are the activation functions for layer 1 and layer 2. Output layer activation function $g^{[2]}$ in this example is sigmoid function.

Step 2: Backward propagation computation for this example

Backward propagation includes computation of derivatives of the cost function with respect to the parameters. For this the chain rule is utilized. Figure 45 provides the computational graph for forward and backward propagation for the example neural network for a single training sample inspired from Andrew Ng's Coursera course lectures [168]. Backward propagation derivative computation is shown in orange color and red arrows. The variables x represents the single training example, y represents the label for the single training sample, $z^{[1]}$ and $z^{[2]}$ represent linear combination computation output for the single training sample for layer 1 and layer 2, $a^{[1]}$ and $a^{[2]}$ represent the output

(activation output) for the single training sample from layer 1 and layer 2, and $L(a^{[2]}, y)$ represents the logistic regression loss value for the single training sample.

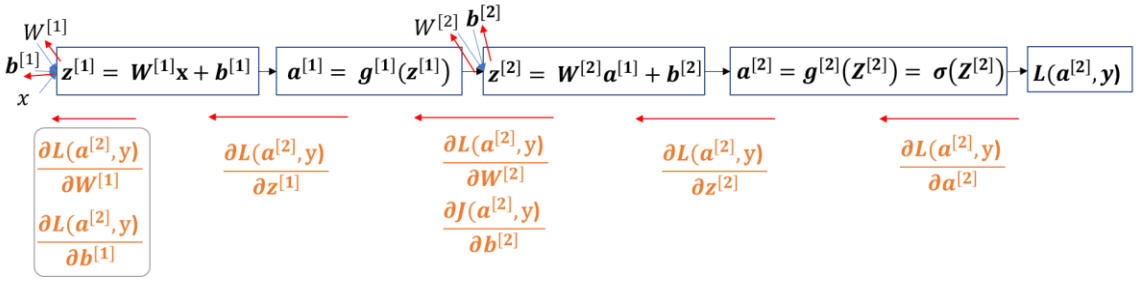


Figure 45 – Forward and backward propagation computation graph for the example neural network on a single training sample (modified from diagram in Coursera course by Andrew Ng) [168].

Note that for computation of $\frac{\partial L(a^{[2]}, y)}{\partial z^{[1]}}$, chain rule is utilized (Equation 24), that consists of derivative of activation functions.

$$\frac{\partial L(a^{[2]}, y)}{\partial z^{[1]}} = \frac{\partial L(a^{[2]}, y)}{\partial (a^{[2]}} \frac{da^{[2]}}{dz^{[2]}} \frac{dz^{[2]}}{da^{[1]}} \frac{da^{[1]}}{dz^{[1]}} \tag{24}$$

When large number of hidden layers are utilized the product of many partial derivatives of activation functions can lead to a very small value of the final partial derivative w.r.t parameters. This leads to very slow rate of update in parameters. This issue is called as the problem of Vanishing Gradient [175]. On the other hand, if an activation function is utilized that can take higher values, then when number of layer are high, product of multiple high value derivatives can lead to final derivative values that are very high. This will lead to very large step sizes and can miss the global optimum. This is called as the problem of exploding gradients [176]. Deep layer neural network architectures are more conventionally used in supervised learning to map output label y with input variables x_1,

x_2 etc. For sequences, mainly where dependency can exist between a previous data point back in time and current data (output), a "memory" of previous data is needed, thus a Recurrent Neural Network (RNN) is utilized. Another reason RNN is preferable to model sequences is that it allows for different lengths of input and output data mapping which is often the case in several areas, such as speech recognition, text translation, and music generation etc. [177]. Working of RNN and LSTM RNN is briefed next.

5.3.4.3 What are RNN and how do they work?[177][2]

Recurrent Neural Network (RNN) is a variant of Neural Network that is capable of utilizing "memory" of previous event data to predict the next values for sequence [178]. The notations used in this sub section to describe the architecture, forward, and backward propagation for RNN are: 1) Input data of the sequence x is represented based on its position in the sequence as $x^{<i>}$, 2) Output data of the RNN is represented as $\hat{y}^{<i>}$, where i is the position of the output data in the sequence, 3) T_x and T_y are lengths on the input and output sequence respectively, 4) Activation values for data in sequence with position i are represented as $a^{<i>}$, 5) the weights and biases associated with computation of activation value $a^{<i>}$ and predicted value $\hat{y}^{<i>}$ are represented as W_{aa}, W_{ax}, W_{ya}, b_a, and b_y. Figure 46 shows the RNN architecture unrolled and rolled to show the connectivity of previous data to current data prediction.

[2] *Theory interpretation and equations presented in this sub section draw from Coursera Course – Sequence Learning, instructed by Andrew Ng

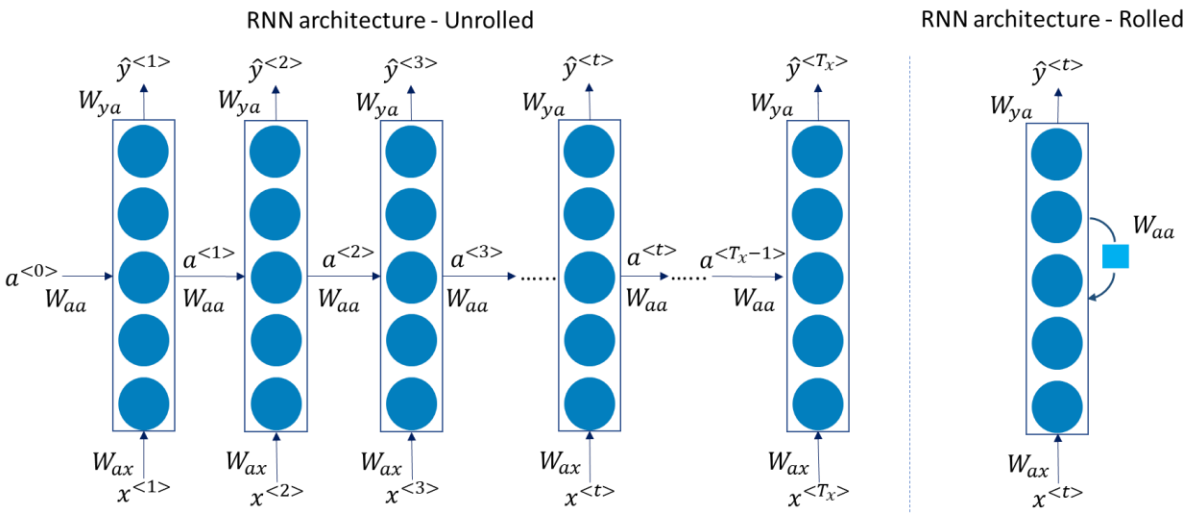


Figure 46 – RNN architecture unrolled and rolled schematic, when $T_x = T_y$ (modified from diagram in Coursera course by Andrew Ng) [177]

The following formulas represent forward propagation computations for RNN (Equation 25 and Equation 26), where, g_a and g_y are activation functions.

$$a^{<t>} = g_a(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a) \tag{25}$$

$$\hat{y}^{<t>} = g_y(W_{ya}a^{<t>} + b_y) \tag{26}$$

Note in equations, the dependence of predicted value $\hat{y}^{<t>}$ on previous input value $x^{<t-1>}$ through previous activation value computation $a^{<t-1>}$. This is also shown in Equation 27.

$$a^{<t-1>} = g_a(W_{aa}a^{<t-2>} + W_{ax}x^{<t-1>} + b_a \tag{27}$$

This allows RNN to update weights and biases based on previous input values. While Figure 46 shows RNN architecture when $T_x = T_y$, RNN architecture is flexible to accommodate different input and output lengths. These are categorized in four types: 1)

one-to-one, 2) many-to-one, 3) many-to-many, and 4) one-to-many. Figure 47 shows the generalized architecture for the four categories. Many-to-many architecture with different input and output lengths is shown last.

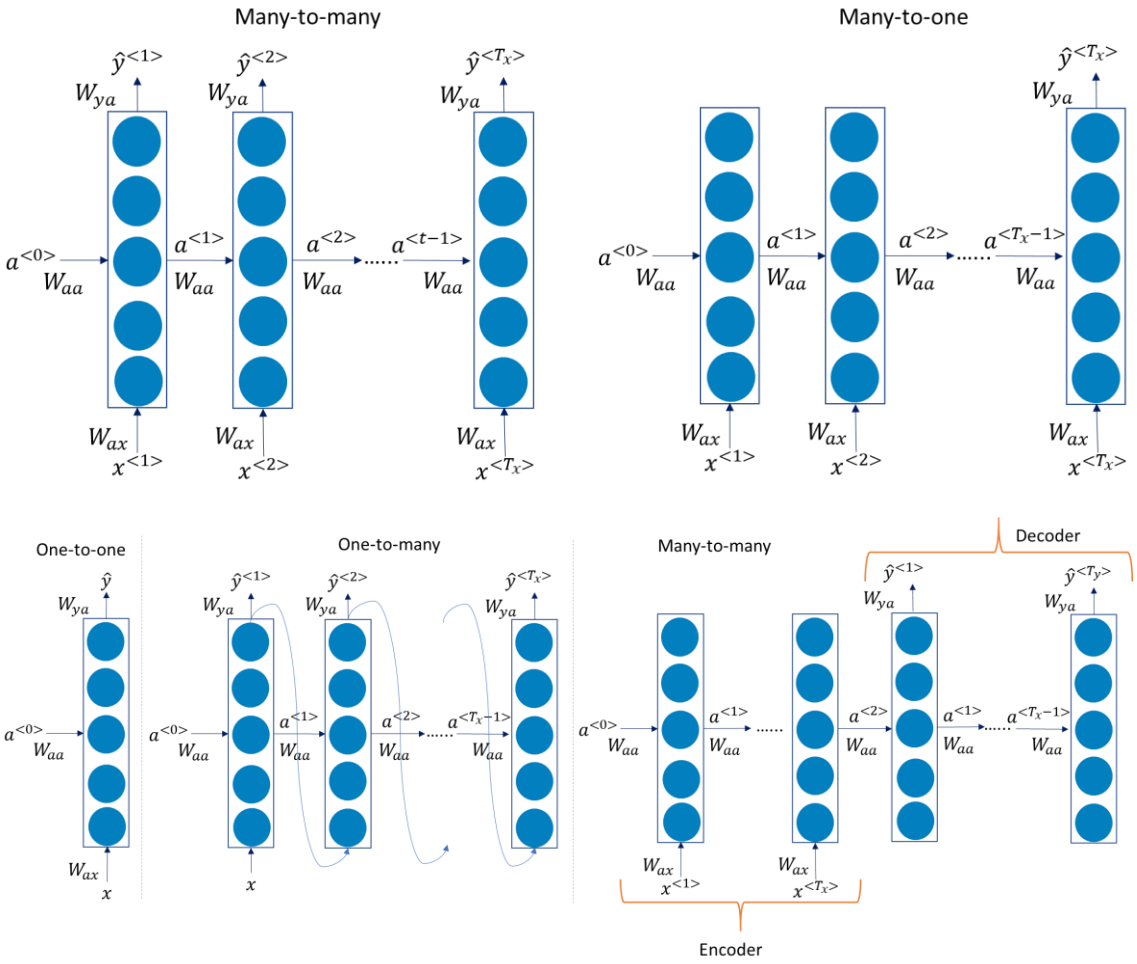


Figure 47 – Different type of RNN architectures (modified from diagram in Coursera course by Andrew Ng) [177].

The loss function for RNN is sum of loss for all input predictions. Equation 28 show the total loss computation formula, where, $L^{<t>}(\hat{y}^{<t>}, y^{<t>})$ is the loss in prediction at time step t.

$$L(\hat{y}, y) = \sum_{t=1}^{T_y} L^{<t>}(\hat{y}^{<t>}, y^{<t>}) \tag{28}$$

Backpropagation flow in RNN to find parameters W_{aa}, b_a, W_{ya}, and b_y is shown using red arrows in Figure 48. This is also called backpropagation through time.

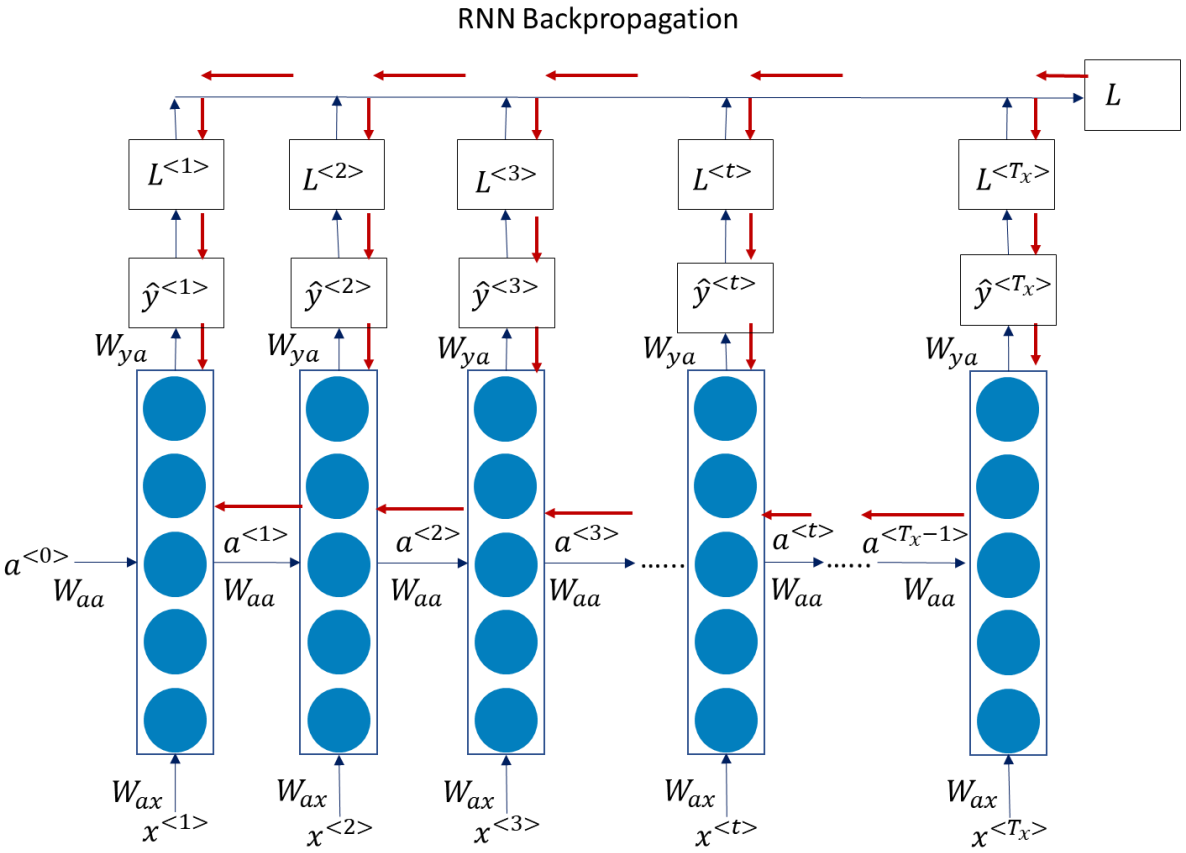


Figure 48 – Backpropagation through time for RNN (modified from diagram in Coursera course by Andrew Ng) [177].

RNN can suffer from vanishing gradient problem due to product of many small activation values in the chain rule in backpropagation implementation [179]. This leads to updates in weights and biases of the model based on nearby losses. In other words, the model parameters don't get well trained based on far away data prediction performance

[179]. This is a drawback of RNN especially when accounting for long-term dependency in a sequence is crucial to predictions. To tackle the issue of vanishing gradient in RNN, LSTM RNN [180] is utilized.

5.3.4.4 What are LSTM RNN?[177]

The architecture of LSTM RNN differs from traditional RNN mainly because of addition of a 'memory' component and gates to regulate the memory – forget gate, update gate, and output gate. A memory cell variable is added to LSTM to retain previous information. The components, notations and Equations (29 to 34) of LSTM computations are provided next. For simplicity, in these equations the weights associated with activations $a^{<t-1>}$ and input variables $x^{<t>}$ are combined together, for example in Equation 29, W_c represents weights matrix that contains weights for $a^{<t-1>}$ and $x^{<t>}$ for computation of $\tilde{c}^{<t>}$. In LSTM, at every time step:

1. A candidate for memory cell ($\tilde{c}^{<t>}$) is computed (Equation 29) that is dependent on previous time step activation value ($a^{<t-1>}$) and current input value ($x^{<t>}$).
    - Candidate memory cell value, ($\tilde{c}^{<t>}$)

$$\tilde{c}^{<t>} = tanh(W_c[a^{<t-1>} + x^{<t>}] + b_c) \quad (29)$$

2. Candidate memory cell value ($\tilde{c}^{<t>}$) for current time step and previous time step memory cell value ($c^{<t-1>}$) are regulated as per update gate (Γ_u) and forget gate (Γ_f) respectively to obtain memory cell value for the current time step ($c^{<t>}$) , shown in Equation 30.

- Memory cell value, ($c^{<t>}$)

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>} \quad (30)$$

Thus, forget and update gates are used to retain previous memory cell value and update memory cell value to candidate value.

3. Forget gate (Γ_f) and update gate (Γ_u) utilized for candidate memory cell computation in previous bullet are obtained using Equation 31 and Equation 32.

- Forget gate, (Γ_f):

$$\Gamma_f = \sigma(W_f[a^{<t-1>} + x^{<t>}] + b_f) \quad (31)$$

- Update gate, (Γ_u):

$$\Gamma_u = \sigma(W_u[a^{<t-1>} + x^{<t>}] + b_u) \quad (32)$$

4. Activation value for the time step ($a^{<t>}$) is computed using Output gate (Γ_o) value as shown in Equation 33 and Equation 34.

$$a^{<t>} = \Gamma_o * \tanh c^{<t>} \quad (33)$$

- Output gate (Γ_o):

$$\Gamma_o = \sigma(W_o[a^{<t-1>} + x^{<t>}] + b_o) \quad (34)$$

This process is visualized in schematically in Figure 49.

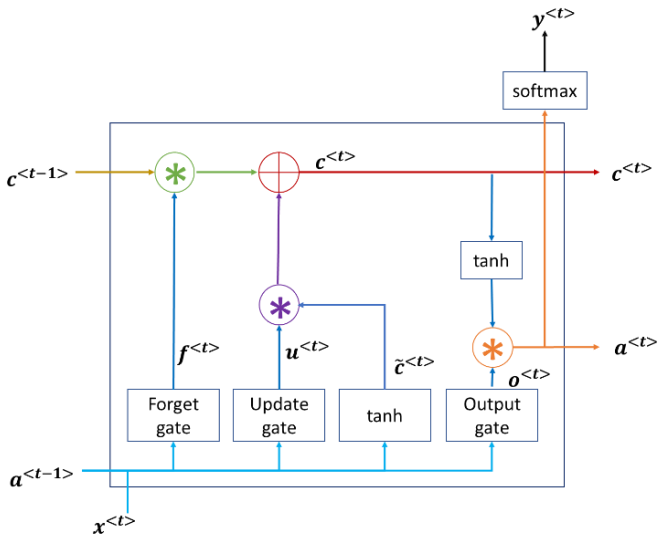


Figure 49 – LSTM cell architecture (modified from diagram in Coursera course by Andrew Ng [177] and Chris Olah's blogpost [72].

The value of gates and candidate memory cell value are regulated based on current input ($x^{<t>}$ and previous step activation value ($a^{<t-1>}$). This allows the LSTM to regulate the derivative values from getting too low to too high to prevent vanishing gradient and exploding gradient issues [181].

A variant of LSTM RNN, the bidirectional LSTM RNN (BLSTM) has an architecture that allows it to learn output mapping both from past and future information [182]. BLSTM has two layers to propagate in both forward and backward directions. The BLSTM RNN architecture is shown in Figure 50. The prediction values are computed utilizing both forward activation and backward activation values [177].

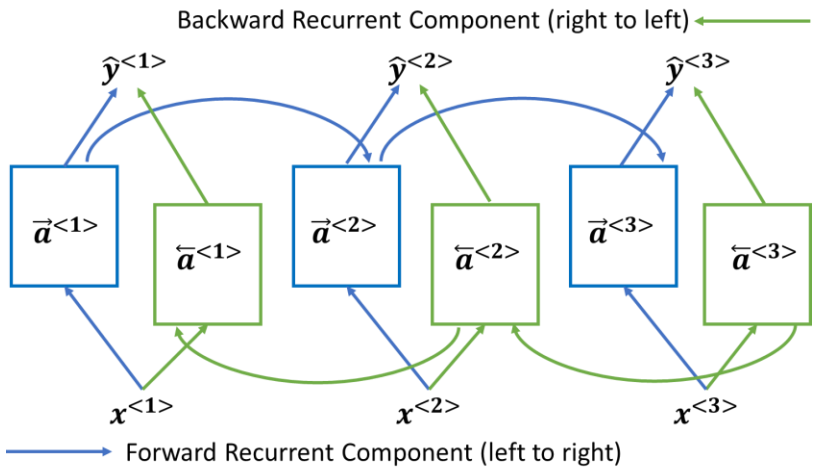


Figure 50 – Simplified representation of Bidirectional RNN architecture (modified from diagram in Coursera course by Andrew Ng) [177].

Deep RNN models include multiple RNN layers that can be traditional RNN, LSTM RNN, or LSTM layers. The architecture for deep RNNs is presented in Figure 51.

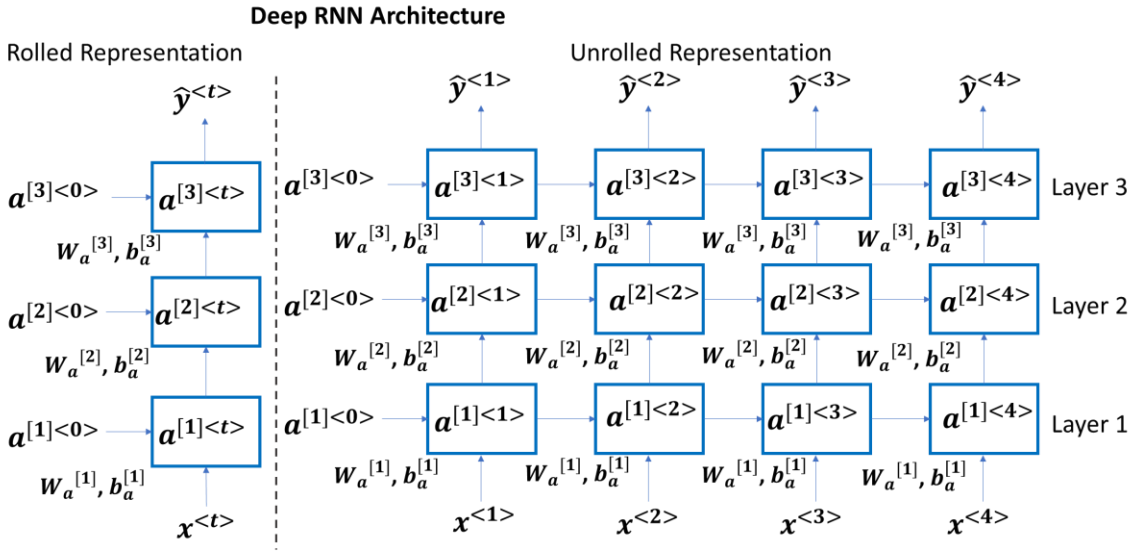


Figure 51 – Simplified representation of deep RNN architecture (modified from diagram in Coursera course by Andrew Ng) [177].

In this study, Deep RNN with BLSTM layers are utilized to develop univariate and multivariate prediction models. Experiment design and model parameters utilized to test the effectiveness of univariate and multivariate Deep RNN models is presented next.

5.3.5 Model Development and Experiment Design

Three set of experiments are designed to investigate the three research questions in this study:

- How does the multivariate model perform in comparison to univariate model for consecutive data gap predictions for typical day traffic?
- How does the multivariate model perform in comparison to univariate model when a non-typical traffic pattern is observed?
- How does multivariate and univariate imputed data impact the digital twin generated travel time at selected routes?

5.3.5.1 Model Development

The objective of first two experiments are: 1) to compare the accuracy of multivariate and univariate LSTM RNN model predictions, and 2) to compare ability of multivariate and univariate LSTM RNN models data gap predictions when a non-typical traffic pattern is observed, i.e., a special event, crash, holiday, etc. Three corridor approaches are selected for the study based, representing varied traffic demand conditions: State St. NW EB, Connector SB, and Peachtree St. NE SB. Data loss will be created for these approaches, the imputation algorithms applied, and then the results of the algorithms compared to the actual collected data.

The approach locations on the North Avenue Smart Corridor are shown in Figure 52. Each of these approaches has two lanes. Twenty-four hour 6-minute aggregate historic time series detector data for Mondays, for each lane, (State-EB-L_1, State-EB-L_2,

Connector-SB-L_1, Connector-SB-L_2, Peachtree-SB-L_1, and Peachtree-SB-L_2) are utilized for the first two sets of experiments.

Figure 52 – Experiment 1 and 2 North Ave. corridor approach locations.

For the developed models, time series data for 10 or 11 Mondays are utilized for model training, while the remaining Mondays are utilized for model validation. Monday March 18th is not utilized in the training set as it is utilized as the base day with typical traffic to study impact of univariate and multivariate model imputations on typical traffic on simulation model generated results.

The training data consists of a 24-hour time series for each Monday, synchronously placed, creating a 264 hour (240 for a 10 day training period) data set. If a day has more than 10 hours with missing data at a detector it is not included in the training set for that detector. The potential 11 Mondays available for training are February 4th, 18th, and 25th; March 4th, 11th, and 25th; April 1st, 8th, 15th, and 22nd; and May 6th; all in 2019. The validation set utilized in this experiment consists of 24-hour time series data for May 13th, 20th, and 27th, 2019, again arranged synchronously. For State-EB-L_1, State-EB-L_2, Connector-SB-L_1, and Connector-SB-L_2 the training set comprises of all 11 Mondays and validation set comprises of the three Mondays. For Peachtree-SB-L_1 and Peachtree-SB-L_2, the training set consists of 10 Mondays (February, 25th is excluded due to data loss).

Each data series is considered in six-minute bins, the fidelity of the data provided by the field collection equipment. Thus, a training data set contains a time-series of 2640 (or 2400) data points in length, i.e. the six-minute increments in 264 (or 240) hours. Table 5 lists the training data and validation data length used for the univariate and multivariate model development for the six detectors in this study.

Table 5 – Training set and validation set data description used for model development.

	Detector	Training Set		Missing Data Points in Training Set	Validation Set	Missing Data Point Count in Validation Set	
		Days	Data points			Typical Day	Atypical Day
1	State-EB-L_1	11	2640	11	Same for all 2 Typical Days 13 May 2019 20 May 2019 1 Atypical Day 27 May 2019	1	0
	Luckie-EB-L_1	11	2640	7		NA	NA
2	State-EB-L_2	11	2640	11		1	0
	Spring-WB-L_1	11	2640	16		NA	NA
3	Connector-SB-L_1	11	2640	12		2	1
	Juniper-WB-L_1	11	2640	11		NA	NA
4	Connector-SB-L_2	11	2640	12		2	1
	Luckie-EB-L_2	11	2640	7		NA	NA
5	Peachtree-SB-L_1	10	2400	8		0	0
	Luckie-WB-L_1	10	2400	10		NA	NA
6	Peachtree-SB-L_2	10	2400	8		0	0
	State-WB-L_3	10	2400	8		NA	NA

Detectors at Peachtree St. SB observed 120 missing data points on 25 February 2019. Hence, this day was not included in the training data set. The total missing data points in the training set studied at the detectors at State St. NW EB, Connector SB, and Peachtree St. SB are 11 out of 2640, 12 out of 2640, and 8 out of 2400, respectively (less than 0.005%

of the training set data series). Given the relatively small number of missing data points in the training set a [-10] is inserted at each point in the series to handle NaN (not a number) values and to mark the presence of an unexpected data point in series, as no data point is below 0 otherwise. Other methods for infilling training data such as replacing data gaps with the median or mean of the surrounding values or removing them could also be implemented [183]. However, in this study, the handful of missing values in training set are replaced by -10 assuming that the RNN deep layers will determine these as outliers. Although, this hypothesis has not been tested in this study and remains a future study area.

For the univariate time series model training, the historical data for the given detector is utilized. For multivariate time series model training, two time series are utilized, the given detector and a second detector with a similar time series. The results of the time series clustering using the DTW time similarity measure is utilized to select a detector with a similar time series at each of the six detectors under study. That is, for the multivariate time series models a similar detector is chosen from the same cluster (of the 11 clusters created) as the given detector. The selected time series for the multivariate time series models for the six detectors are:

1. State-EB-L_1: State-EB-L_1 and *Luckie-EB-L_1*
2. State-EB-L_2: State-EB-L_2 and *Spring-WB-L_1*
3. Connector-SB-L_1: Connector-SB-L_1 and *Juniper-WB-L_1*
4. Connector-SB-L_2: Connector-SB-L_2 and *Luckie-EB-L_2*
5. Peachtree-SB-L_1: Peachtree-SB-L_1 and *Luckie-WB-L_1*
6. Peachtree-SB-L_2: Peachtree-SB-L_2 and *State_WB_L_3*

The similar time series detectors chosen were also checked to ensure a low level of data gaps. Figure 53, Figure 54, and Figure 55 show the training and validation data for the univariate and multivariate time series model development for the three approaches under study and the identified similar detector. Here, the Y-axis is the volume count value and the X-axis is the time bin count, when all Mondays are arranged in synchronous manner. In these plots, the time series data for the detector for which model is trained is shown in blue and the detector with similar time series is shown in orange.

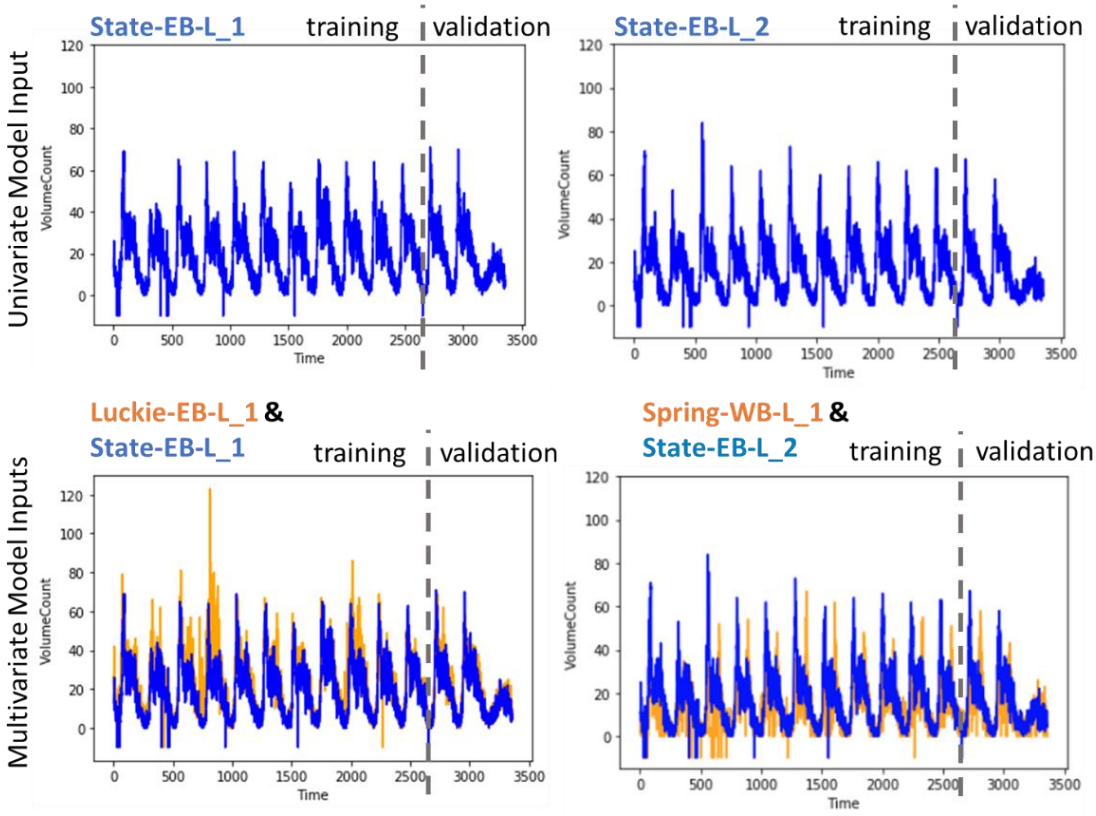


Figure 53 – Time series data utilized for modelling State-EB-L_1 and State-EB-L_2.

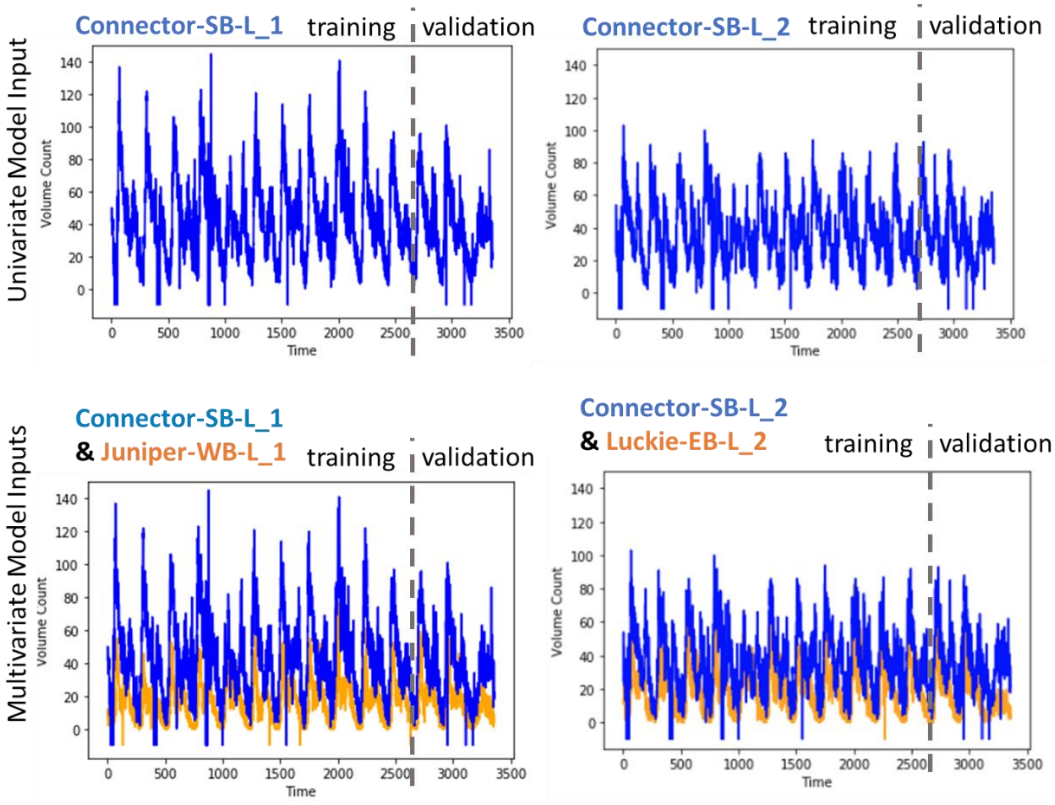


Figure 54 – Time series data utilized for modelling Connector-SB-L_1 and Connector-SB-L_2.

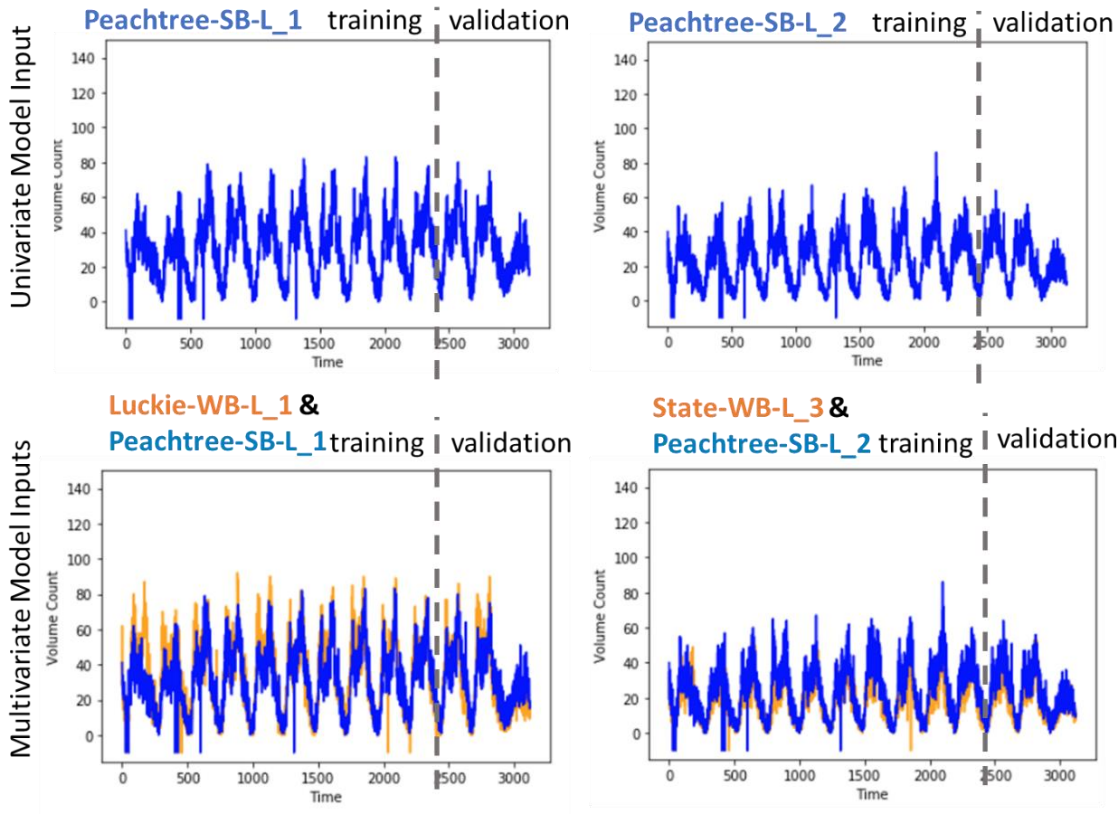


Figure 55 – Time series data utilized for modelling Peachtree-SB-L_1 and Peachtree-SB-L_2.

Data preparation [184][3]: For time series modeling, feature sets and feature set labels are created from the series where feature sets are sequences of values from the series and feature set labels are the next value in the series after the feature set. The feature set label represents the value to be predicted in the training. The length of sequence taken as a feature set is the window size. For example, for a model with window size ten, the feature set will be first ten data points and feature set label will be the eleventh data point. The order of these pairs of feature sets and set labels are shuffled to avoid sequence bias in selection of the pairs. Next, batching is used to create batches of feature sets and labels that

[3] Methodology of data preparation is drawn from video lectures in Coursera course "Sequences, Time Series and Prediction" instructed by Laurence Moroney.

are fed into RNN layers for training. Figure 56 shows the process of preparing data to feed into deep RNN layer model for an example series of length 10, window size 4, and batch size 2.

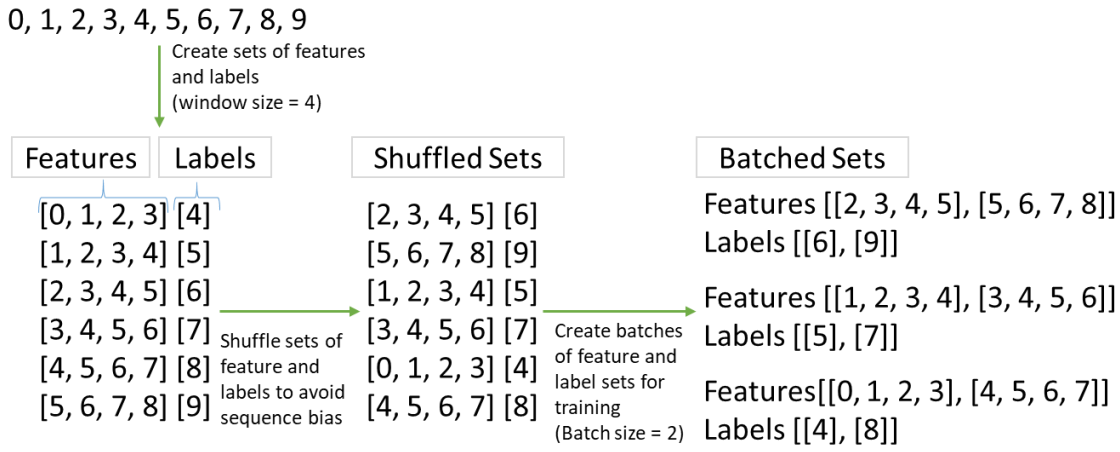
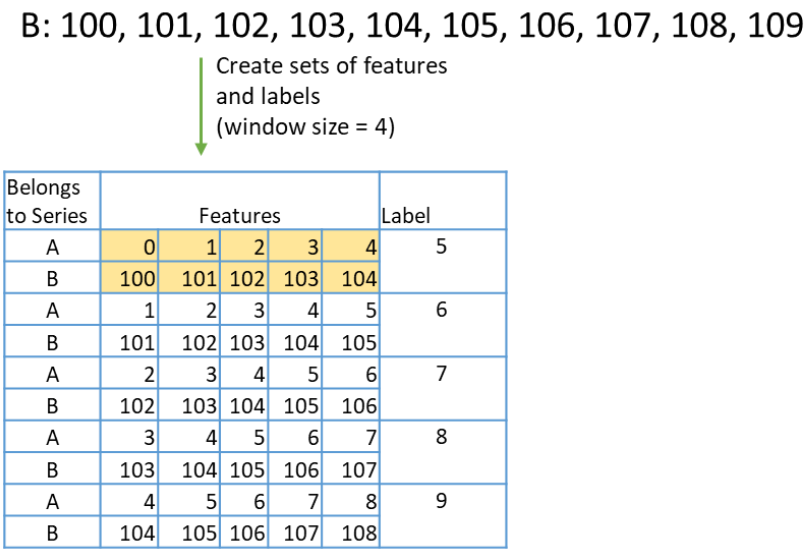


Figure 56 – Creating batches of features and labels from the time series for training the deep RNN model.

In a multivariate time series model it is recalled that multiple time series are utilized as input. In this case, the feature set includes sequences of the selected window size length for all of the input time series and the corresponding feature set label. After creating pairs of feature sets and feature set labels, the process of shuffling and batching is same as described previously. The dimension of features in this case is greater than 1. Figure 57 shows an example of feature and label set for window size 4 to train the model for predicting series A using inputs as series A and B.



Belongs to Series	Features					Label
A	0	1	2	3	4	5
B	100	101	102	103	104	
A	1	2	3	4	5	6
B	101	102	103	104	105	
A	2	3	4	5	6	7
B	102	103	104	105	106	
A	3	4	5	6	7	8
B	103	104	105	106	107	
A	4	5	6	7	8	9
B	104	105	106	107	108	

Figure 57 – Preparing feature and label sets to train deep RNN multivariate time series model.

Model training and validation [184][4]: To program deep RNN models Python version 3.7.2 [185] programming language, Keras [186] python library, and TensorFlow [187] python library are utilized. Deep RNN models trained for multivariate and univariate models for the six lanes consists of four BLSTM layers and a dense layer. The crucial step for univariate time series and multivariate time series modeling is to adjust the dimensionality using the parameter *input_shape* in the first keras BLSTM layer, which is fed with batches containing feature sets and feature set label pairs. The expected input to the RNN layers is 3 dimensional: batch size, window size, and series dimensionality. Series dimensionality is 1 for univariate time series modeling and greater than one for multivariate time series modelling. In this study, the multivariate models are tested with 2 time series inputs, hence the series dimensionality is 2 for the developed multivariate models. Figure

[4] Methodology of data preparation is drawn from video lectures in Coursera course "Sequences, Time Series and Prediction" instructed by Laurence Moroney.

58 shows shape of input to be fed to deep RNN model trained using Keras and TensorFlow library.

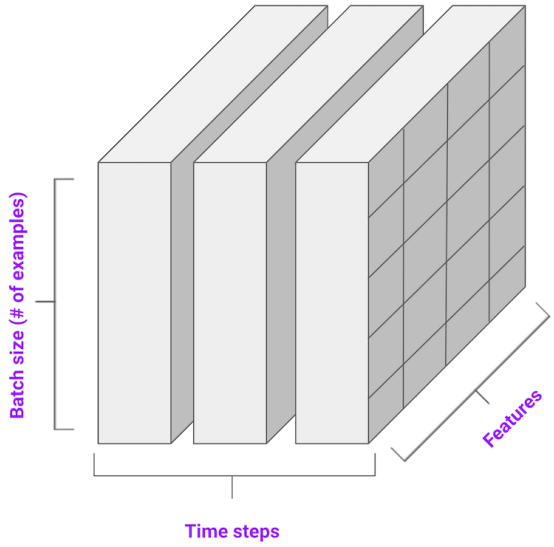


Figure 58 – Shape of input data for training deep RNN model [187].

To train the model, the loss function is set to the mean squared error value and the Stochastic Gradient Descent (SGD) optimizer is utilized. Lastly, the number of iterations (epochs) to train is also set. Note that there are several hyper parameters (i.e. pre-set parameters) of the deep RNN model, such as number of layers, nodes per layer, learning rate, number of epochs, window size, batch size, etc. The hyper parameter values for all the developed models are kept similar to facilitate comparison. The hyper parameter value utilized are: window size of 80 or 100, batch size of 30, SGD learning rate of $1e^{-5}$, $1e^{-6}$, $2e^{-6}$, and $5e^{-6}$, and number of epochs of 500. To optimize model training and performance the various hyper parameters such as window size, batch size, and learning rate could be tuned. For this initial effort of exploring the advantage of using multivariate

deep RNN versus univariate deep RNN modeling, the model parameters were selected (as explained in the following) and only slightly tuned for improved performance; however, future research will further explore the impact of these parameters. Most hyperparameters such as number of epochs, model layer configuration parameters (i.e., the number of layers and nodes in each layer), number of epochs, batch size, optimizer (SGD) are same for all models trained. Initial model performance tests with window sizes of 30, 60, and 80 were conducted, with 80 chosen to train the univariate models for State St. EB detectors. For the remaining detectors, both for univariate and multivariate models, a larger window size of 100 is used, further expanding the window size utilized to make predictions to 10 hours. Such a window size is expected to provide sufficient information about the historic traffic volume trend. The four learning rates used are all of the order $1e^{-6}$. In a previous study, for practical purposes, Yoshua suggested a learning rate range of 0.1 to 10^{-6} [188]. In model development process, the learning rates are tuned only slightly to avoid vanishing and exploding gradient issues and to improve model accuracy. For model validation (testing), the developed models are utilized to perform next step predictions on the validation length. The accuracy of model predictions is measured as closeness of predicted values with actual values in the validation set.

5.3.5.2 Experiment Designs

Experiment Design 1: The experiment design compares the performance of the univariate and multivariate time series prediction models to infill data gaps tests five prediction scenarios on the typical traffic pattern validation data set. The validation set includes two Mondays with typical traffic patterns. Univariate and multivariate model performance is tested on four prediction scenarios. The four scenarios vary in the number

of consecutive data units predicted by the model: Scenario 1 – 30 missing units, Scenario 2 – 80 missing units, Scenario 3 – 120 missing units, Scenario 4 – 380 missing units. The performance of models to predict the 4 scenarios starting at 100 units or 10 AM on the validation set is compared. Since the models require a window size of 100 to make new prediction, this time is selected for comparison. To obtain more evidence on the univariate and multivariate model performance to predict consecutive missing units, prediction accuracy results for Scenario 1, Scenario 2, and Scenario 3, starting at hours after 10 AM such as 11 AM, noon, 1 PM, 2 PM etc. are also compared using statistical test.

Experiment Design 2: For the second objective, testing the univariate and multivariate model prediction performance when traffic demand pattern differs from the typical traffic pattern, the prediction accuracy is measured for May 27th, 2019. The atypical traffic pattern is hypothesized to be a result of Federal holiday (Memorial Day). This can be observed in the complete time series plots shown previously in Figure 53, Figure 54, and Figure 55.

Experiment Design 3: The third objective, explores the impact simulation generated performance measures of univariate and multivariate time series predictions, for a regular day and a day with unexpected traffic conditions. The PM peak hours (3 PM to 6 PM) for two traffic day scenarios are simulated: typical Monday (March 18th, 2019) and holiday Monday (May 27th, 2019, Memorial Day). For each of the two traffic scenarios three sets of simulation runs are conducted, where at three approaches (State St. NW EB, Connector SB, and Peachtree St. SB) three different sets of traffic volume values are utilized: 1) base traffic condition – non-imputed/original traffic volume data, 2) univariate model imputations, and 3) multivariate model imputations traffic data. For each of the three traffic

data scenarios at the six detectors, for each of the traffic day scenarios (regular Monday and holiday Monday) ten replicate simulation trials are run. The impact of utilizing a multivariate model over univariate model for imputations on a regular Monday and holiday Monday on travel times at selected routes in the corridor is evaluated. The next section presents results of the three experiments.

5.4 Results and Discussion

Prior to showing the results for the three experiments, the behavior of the models is shown when predicting only the next step. This is done by executing the model to predict next volume data at every time step of the validation data set using actual historic data of window size length as inputs. For example, the first prediction is made at midnight (start of validation set) using last window size length of data observed on previous Monday. The process of using previous actual historic data as input is applied at every step of the validation set.

5.4.1 Model Validation - One- Step Prediction Accuracy Results

One-step prediction results for the developed multivariate and univariate time series models at the six detectors are obtained. Results for three detectors are presented in Figure 59. Predictions are plotted in Green and actual detector data in Blue. For multivariate prediction plots, the time series data from the similar detector is shown in Orange.

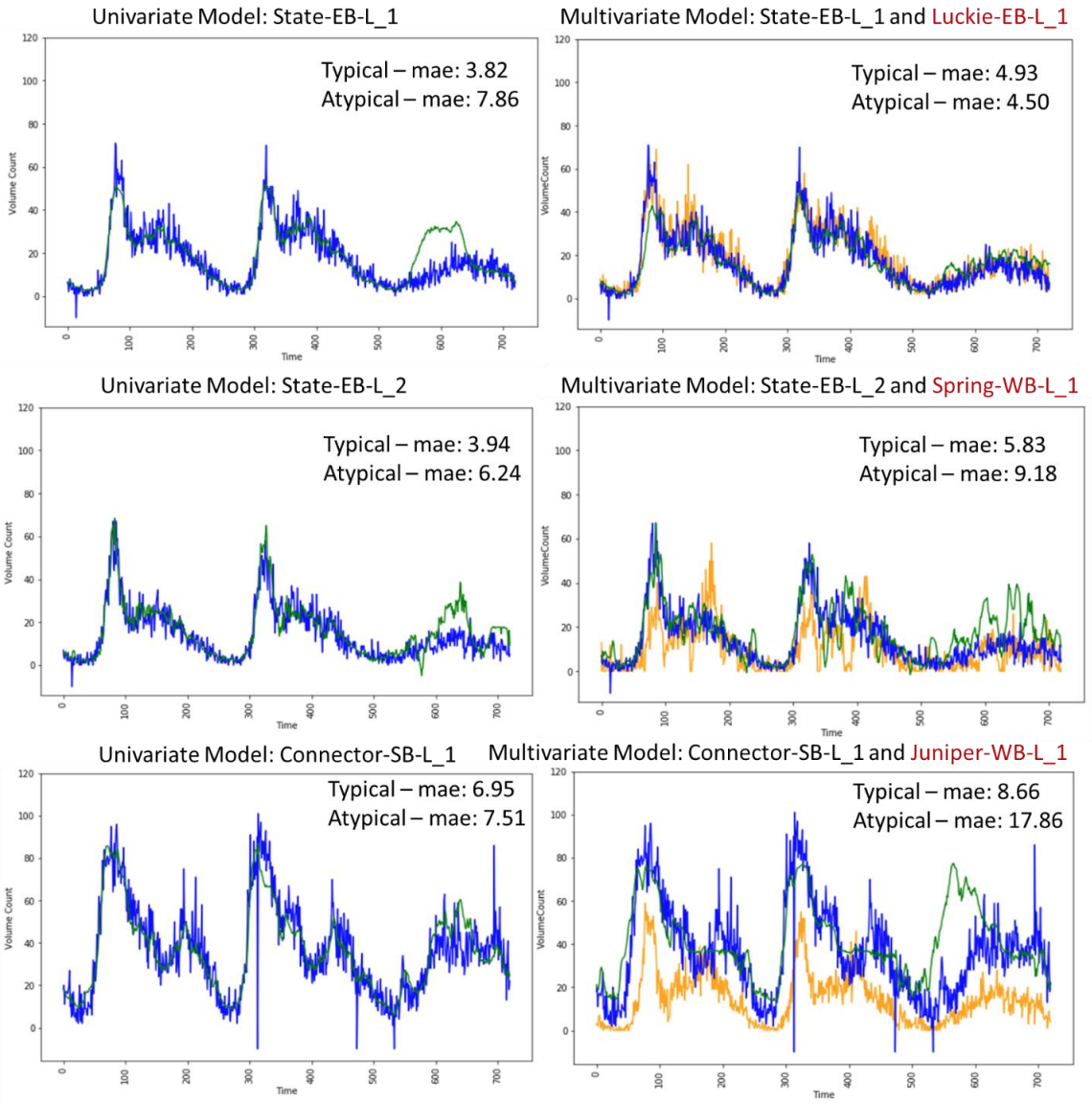


Figure 59 – Univariate and multivariate one-step prediction results for State-EB-L_1, State_EB-L_2, and Connector-SB-L_1. (mae refers to mean absolute error)

It is observed in the plots that both univariate and multivariate deep RNN models provide one-step predictions that captures the traffic volume pattern on typical days much more accurately than on atypical day. Three different measure of errors of the multivariate and univariate time series imputation predictions with respect to actual data are evaluated. These are: mean absolute error (mae), root mean squared error (rmse), and standard deviation of error (std. dev of errors). Table 6 summarizes the time series prediction

accuracy errors of univariate and multivariate model performance for the six detectors on typical day and atypical day.

Table 6 – Model Validation Results: One-step prediction accuracy measures.

Model Type	Approach Lane	Typical Days			Atypical Day		
		mae	rmse	Std. dev	mae	rmse	Std. dev
Univariate	State-EB-L_1	*3.82	*5.20	*5.18	7.86	11.04	9.48
Multivariate	State-EB-L_1	4.93	7.06	6.82	*4.50	*5.65	*4.76
Univariate	State-EB-L_2	*3.94	*5.47	*5.44	*6.24	*8.47	*7.26
Multivariate	State-EB-L_2	5.83	7.83	7.77	9.18	11.74	8.42
Univariate	Connector-SB-L_1	*6.95	*9.70	*9.60	*7.51	*9.96	*9.80
Multivariate	Connector-SB-L_1	8.66	11.40	11.39	17.86	24.61	20.76
Univariate	Connector-SB-L_2	*7.24	*10.20	*10.05	*7.26	*9.39	*9.36
Multivariate	Connector-SB-L_2	8.53	11.58	11.56	9.60	11.97	11.73
Univariate	Peachtree-SB-L_1	*5.25	*6.78	*6.78	*6.71	*8.08	*7.16
Multivariate	Peachtree-SB-L_1	6.02	7.88	7.81	7.22	9.51	9.28
Univariate	Peachtree-SB-L_2	*4.87	*6.25	*6.24	*3.98	*5.22	*4.96
Multivariate	Peachtree-SB-L_2	5.47	7.08	7.08	5.77	7.21	5.67

In Table 6, the lower accuracy measures among univariate and multivariate model one step prediction performance on typical days and atypical day is marked with an asterisk (*). Results obtained indicate more accurate predictions by trained univariate model than multivariate model on typical day traffic pattern and atypical day traffic pattern for all cases except for State-EB-L_1 where multivariate model provides more accurate predictions than univariate on atypical day. Univariate models are trained to predict next-step volumes using historic traffic volume data at the detector that is modeled while multivariate models are trained to predict volumes using historic data at the modeled detector and at another detector that has similar traffic pattern. Better performance of univariate model could be

due to its' closeness in mapping next step data with historic data at the detector. The additional input of historic data from another detector with similar pattern in multivariate model seems to be weakening the mapping. This could be because although the time series are similar they might not be as well matched to enhance the multivariate model performance over the univariate model for such a short gap. The lower prediction accuracy of the multivariate models may also be an indication of a weak correlation or noise between the given detector and selected similar detector. In particular, it is plausible on the atypical day the two detectors are not well correlated, while the correlation is stronger on a typical day. A need to inspect the relationship between the similarities between the inputs to the multivariate models and the model performance is noted for future investigation. Among model performances for all detectors on typical and atypical days, multivariate model performance at Connector-SB-L_1 on atypical day observes a much higher error than univariate model. Multivariate model performance at Connector-SB-L_1 is investigated further as part of Experiment 1 in the next subsection. However, it is also noted that the maximum difference in error measures for univariate and multivariate model performance for all detectors except Connector-SB-L_1 is 2.94 for mae and 3.27 for rmse, indicating the difference between univariate and multivariate model performances is practically minimal.

5.4.2 Experiment 1

The performance of the multivariate and univariate model predictions for four set of consecutive data gaps on typical day validation data set starting at 100^{th} time step (i.e., 10 AM) are tested at the six detectors. Figure 60, Figure 61, and Figure 62 show the five prediction sets for detector State-EB-L_1, Connector-SB-L_1, and Peachtree-SB-L_2.

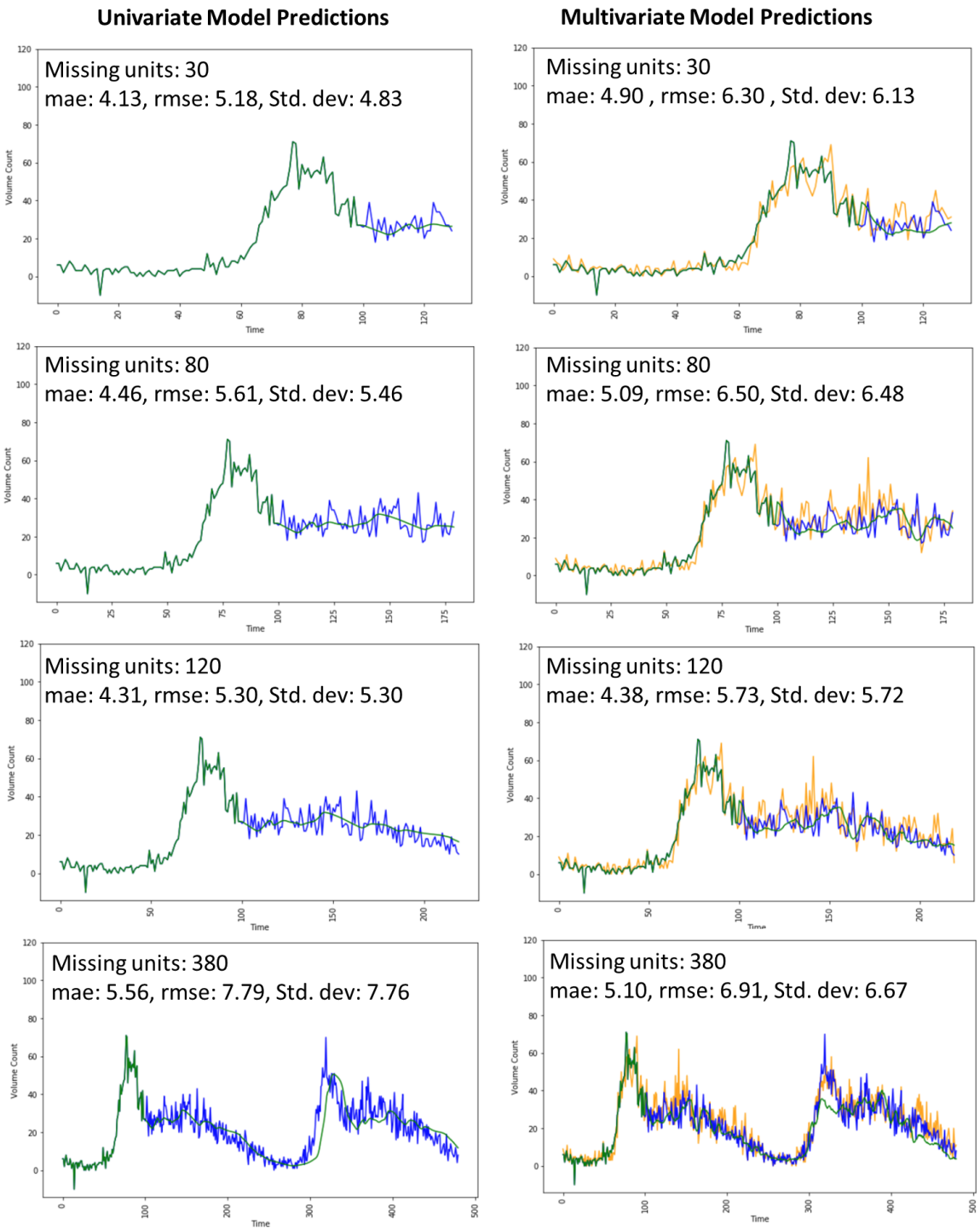


Figure 60 – Univariate and multivariate model predictions at State-EB-L_1 for five missing unit scenarios where, multivariate model inputs are State-EB-L_1 and Luckie-EB-L_1.

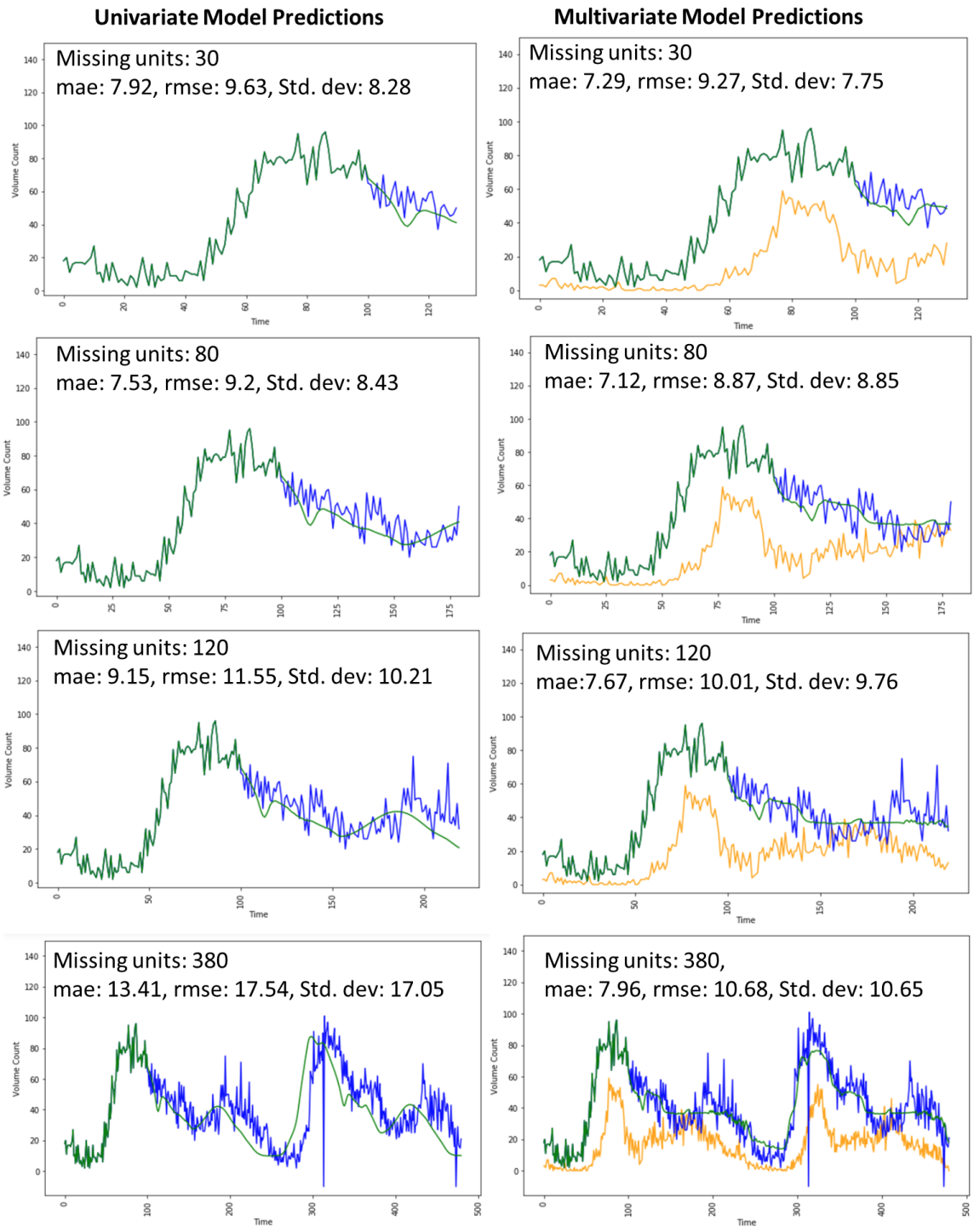


Figure 61 – Univariate and multivariate model predictions at Connector-SB-L_1 for five missing unit scenarios where, multivariate model inputs are Connector-SB-L_1 and Juniper-WB-L_1.

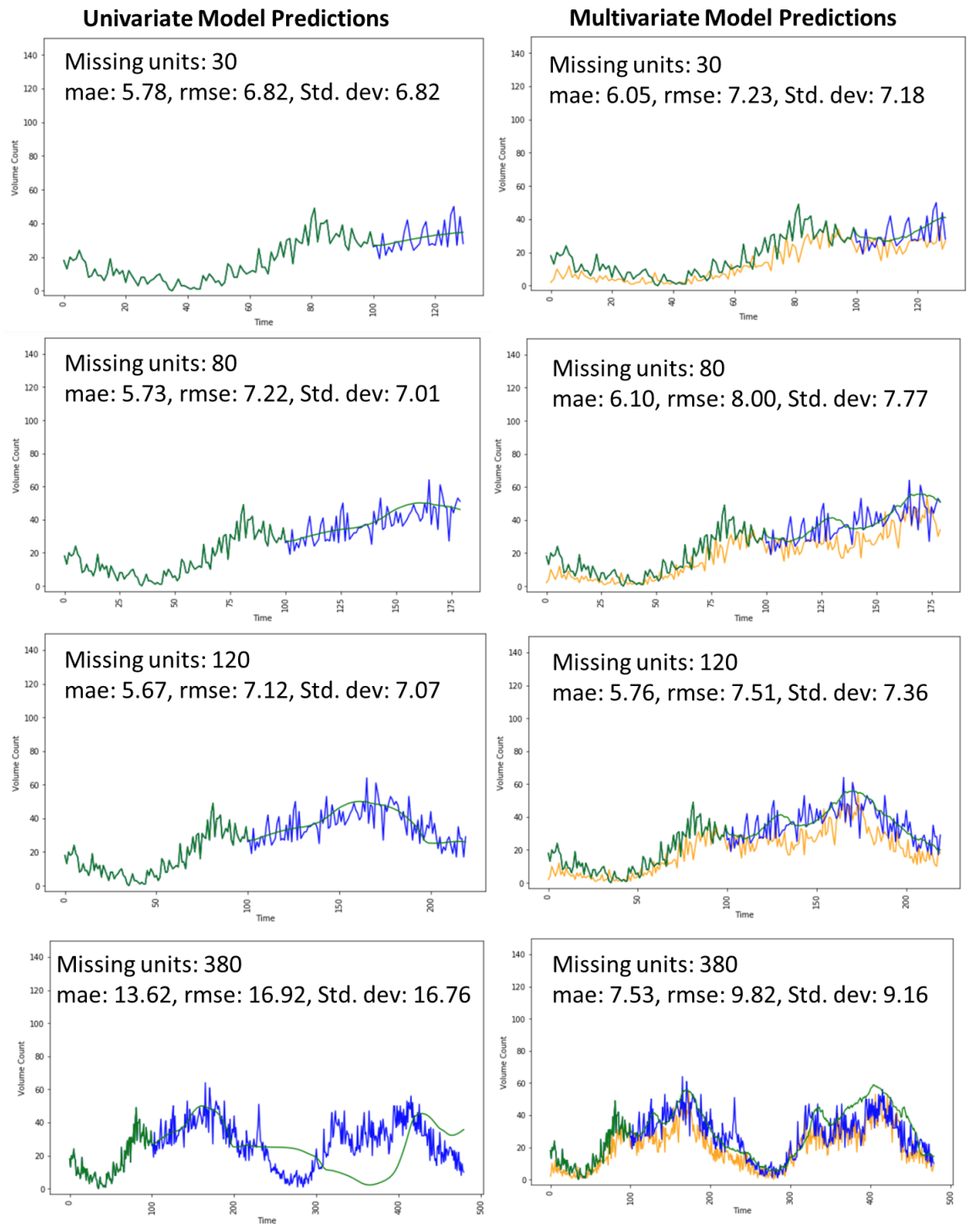


Figure 62 – Univariate and multivariate model predictions at Peachtree-SB-L_2 for five missing unit scenarios where multivariate model inputs are Peachtree-SB-L_2 and State-WB-L_3.

It is observed from the error measures for State-EB-L_1, Connector-SB-L_1 and Peachtree-SB-L_2 (in Figures 60, 61, and 62) that the multivariate model provides higher prediction accuracy for the longer missing units scenario of 380 (38 hours) of data gap in comparison to univariate model predictions. For shorter missing data period, i.e., 30, 80, and 120 (3, 8, and 12 hours), the univariate and multivariate models have similar performance. Similar results were also observed for models at other detectors. Table 7, Table 8, and Table 9 present a summary of the univariate and multivariate model prediction performance results for the four missing unit prediction scenarios at these same locations: State-EB-L_1, Connector-SB-L_1, and Peachtree-SB-L_2. In Tables 7, 8 and 9, the model with higher accuracy results is italicized and marked with an asterisk symbol (*).

Table 7 – Performance measures of multivariate and univariate model for State-EB-L_1. Model with higher accuracy results is italicized and marked with an asterisk symbol (*)

State-EB-L_1	Prediction Units	mae	rmse	std. dev
**Univariate*	*30*	*4.13*	*5.18*	*4.83*
Multivariate	30	4.90	6.30	6.13
**Univariate*	*80*	*4.46*	*5.61*	*5.46*
Multivariate	80	5.09	6.50	6.48
**Univariate*	*120*	*4.31*	*5.30*	*5.30*
Multivariate	120	4.38	5.73	5.72
Univariate	380	5.56	7.79	7.76
**Multivariate*	*380*	*5.10*	*6.91*	*6.67*

Table 8 – Performance measures of multivariate and univariate model for Connector-SB-L_1. Model with higher accuracy results is italicized and marked with an asterisk symbol (*)

Connector-SB-L_1	Prediction Units	mae	rmse	std. dev
Univariate	30	7.92	9.63	8.28
**Multivariate*	*30*	*7.29*	*9.27*	*7.75*
Univariate	80	7.53	9.23	8.43
**Multivariate*	*80*	*7.12*	*8.87*	*8.85*
Univariate	120	9.15	11.55	10.21
**Multivariate*	*120*	*7.67*	*10.01*	*9.76*
Univariate	380	13.41	17.54	17.05
**Multivariate*	*380*	*7.96*	*10.68*	*10.65*

Table 9 – Performance measures of multivariate and univariate model for Peachtree-SB-L_2. Model with higher accuracy results is italicized and marked with an asterisk symbol (*)

Peachtree-SB-L_2	Prediction Units	mae	rmse	std. dev
**Univariate*	*30*	*5.78*	*6.82*	*6.82*
Multivariate	30	6.05	7.23	7.18
**Univariate*	*80*	*5.73*	*7.22*	*7.01*
Multivariate	80	6.10	8.00	7.77
**Univariate*	*120*	*5.67*	*7.12*	*7.07*
Multivariate	120	5.76	7.51	7.36
Univariate	380	13.62	16.92	16.76
**Multivariate*	*380*	*7.53*	*9.82*	*9.16*

Table 7, 8 and 9 confirm the results from the visual inspection of the figures. Similar findings were obtained for univariate and multivariate model performances at State-EB-L_2 and Connector-SB-L_2. At Peachtree-SB-L_1 although a high difference in multivariate and univariate model prediction performance is not observed in the error measures of mae, rmse, and std. dev, the multivariate model prediction are visually

observed to capture the traffic trend better than univariate model predictions, shown in Figure 63.

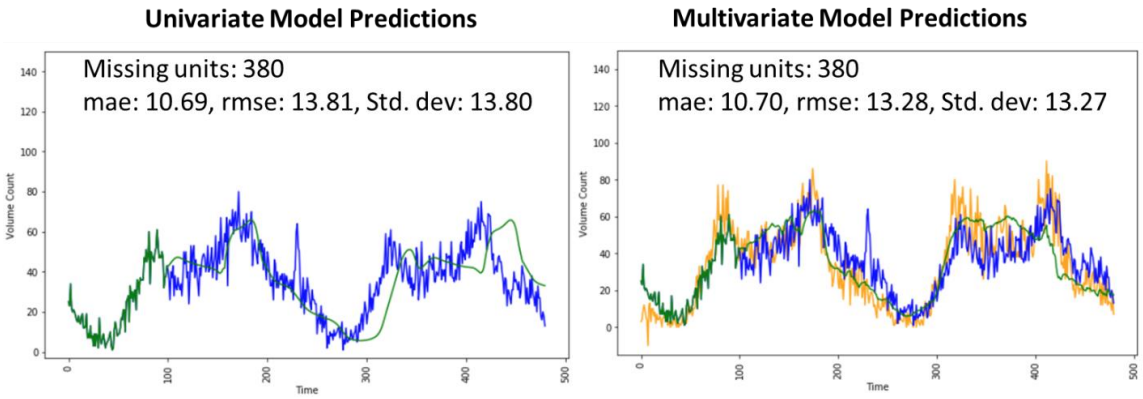


Figure 63 – Performance of univariate and multivariate model at Peachtree-SB-L_1 to predict 380 consecutive data units at 10 AM.

For longer consecutive data predictions (more than 80 or 100 units missing) the univariate model uses relies more, and then entirely, on predicted values as input, as the window size is less than the gap. Whereas in comparison, the multivariate model also relies on predicted values for the detector with missing data; however, it also receives field values from the additional detector input. The difference in the univariate and multivariate model prediction errors is less for shorter consecutive data loss scenarios as some field data will always be included in both. Thus, as seen, for typical traffic, for all six detectors, multivariate model predictions outperform univariate model predictions for the 380 missing units but neither approach is clearly dominate for shorter data gaps.

5.4.2.1 Univariate and Multivariate Model Performance Dependency on Data Gap Start Time

To obtain more evidence on univariate and multivariate model performance the models are run for different start times of the data gap. That is, all results thus far have

assumed the data gap started at 10AM. Performance is now observed for start times incremented over a 24 hour period, i.e., 10 AM, 11 AM, Noon, 1 PM, 2 PM etc. Error accuracy measures mae, rmse, and std. deviation of errors, are obtained for data gap scenarios of 30, 80, and 120. With two typical days (480 data points) in validation set, a similar set of values could not be obtained for the 380 missing data scenario as at 10 AM (100^{th} unit) only one set of prediction can be performed, obtaining values from the 100^{th} to 479^{th} unit. With more typical day validation data, a similar test for longer consecutive data prediction scenarios can be performed in future research to confirm the findings under longer data gap scenarios. Figure 64 shows the mae values obtained for the three prediction scenarios by the multivariate and univariate models, starting at different time unit of the typical day validation data for State-EB-L_2.

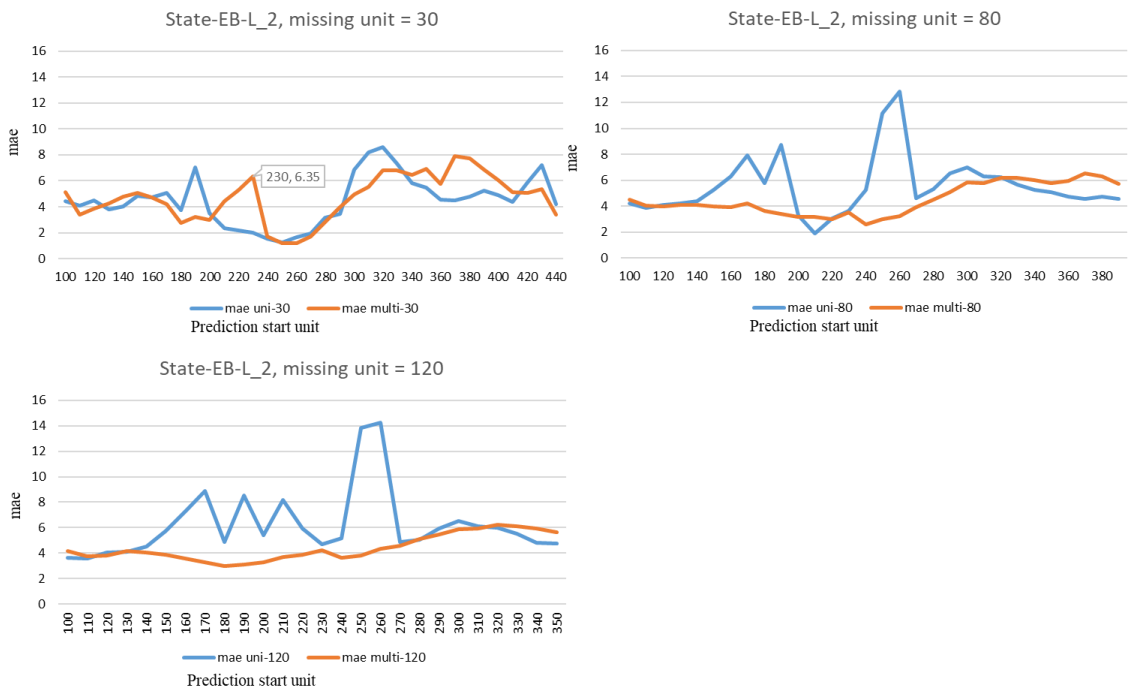


Figure 64 – Mean absolute error measures of univariate and multivariate prediction performance at State-EB-L_2 for 30, 80, and 120 data gaps, starting at different times of typical day validation set.

While there is some variability in absolute values of mae seen in the previously discussed 10AM scenarios, lesser or similar values are generally observed for the multivariate compared to the univariate model for 80 and 120 consecutive unit prediction scenarios, with some limited exceptions. The mae values are also mostly comparable for different start times for 30 consecutive unit prediction scenario. Figure 65 shows mae values obtained for Peachtree-SB-L_1. The highest error values are observed from the univariate model for prediction scenarios 80 and 120. For prediction scenario of 30 units the highest error value is from multivariate model at starting time 220 units or 8 PM. Except for this point, visually error measures for both univariate and multivariate seem comparable for 30 missing unit prediction scenarios.

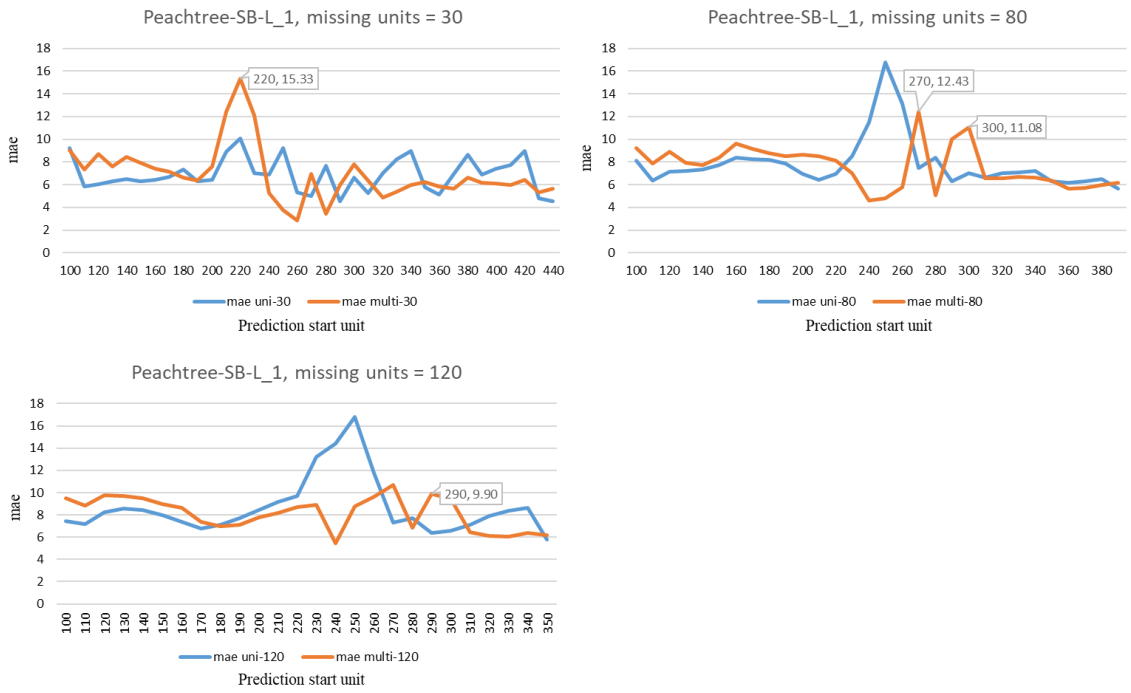


Figure 65 – Mean absolute error measures of univariate and multivariate prediction performance at Peachtree-SB-L_1 for 30, 80, and 120 data gaps, starting at different times of typical day validation set.

As seen above, and for those detectors not shown, except State-EB-L_1 (Figure 66), the mae values obtained using multivariate and univariate are mostly comparable with some limited exceptions. (A non-parametric hypothesis test is conducted in the next subsection that explores the significance of the differences between the approaches.) However, at State-EB-L_1 the values are also similar, except for two somewhat extended periods (in the 80 and 120 scenarios) where the univariate model mae is consistently lower than that from multivariate model, as shown in Figure 66. It is believed that this may result from a lack of similarity in pattern of the additional input time series to the actual detector, resulting in weaker performance than univariate model. In addition, other issues, such as enhanced by tuning hyper parameters, may be a contributing factor. These potential will be further explored as part of future research.

Figure 66 – Mean absolute error measures of univariate and multivariate prediction performance at State-EB-L_1 for 30, 80, and 120 data gaps, starting at different times of typical day validation set.

5.4.2.2 Wilcoxon's Signed Rank Sum Test Results

Wilcoxon's Signed Rank Sum Test is non-parametric test for paired samples [189]. Two sets of the Wilcoxon's Signed Rank Sum Test are conducted. First, a two-tailed Wilcoxon's Signed Rank Sum test of the hypothesis that the multivariate model predictions and univariate model predictions are comparable. Second a one-sided Wilcoxon's Signed Rank Sum test to further explore the hypotheses that the multivariate prediction errors are less than univariate prediction errors. The two tests are conducted for the six detectors for the three error measures (mae, rmse, and std. dev. of errors) for three scenarios – 30, 80, and 120, where the error value from univariate prediction and multivariate prediction for same starting unit are considered as the paired sample. Thus, a total of 54 tests (6x3x3) are conducted for each of the two hypothesis tests.

For first test (two-sided), the null hypothesis is that the median difference between pairs of observations (univariate and multivariate model prediction errors) is zero and alternate hypothesis is that the median difference between pairs of observation is not zero. The test is conducted for significance level 0.05. Hence, a p value less than 0.05 will imply sufficient evidence to reject null hypothesis. For second test (one-sided), alternate hypothesis is that the median difference between two samples (univariate model prediction error – multivariate model prediction error) is greater than 0 implying multivariate models performs better than univariate models. The null hypothesis for this test is that the median difference between the two samples is not greater than 0. Similar to first test, the significance level used for this test is also 0.05. Table 10 provides summary of p-values obtained on conducting the two tests.

Table 10 – P value from two-sided and one-sided Wilcoxon's Signed Rank Sum Test. P values less than 0.05 are marked using asterisk (*). (RN – Reject Null Hypothesis)

Detector	missing units	Two-sided test				One-sided test (greater)			
		mae	rmse	std. dev		mae	rmse	std. dev	
State-EB-L_1	30	*0.01	*0.00	*0.00	RN	1.00	1.00	1.00	
State-EB-L_1	80	*0.01	*0.00	*0.00	RN	0.99	1.00	1.00	
State-EB-L_1	120	*0.00	*0.00	*0.01	RN	1.00	1.00	1.00	
State-EB-L_2	30	0.71	0.47	0.01		0.64	0.76	0.99	
State-EB-L_2	80	0.09	0.11	0.57		0.05	**0.01	0.71	RN
State-EB-L_2	120	*0.01	*0.01	0.25	RN	**0.00	**0.00	0.12	RN
Connector-SB-L_1	30	0.39	0.77	0.52		0.80	0.62	0.74	
Connector-SB-L_1	80	0.14	*0.00	0.05	RN	0.07	**0.00	**0.03	RN
Connector-SB-L_1	120	0.37	*0.00	0.08	RN	0.18	**0.00	**0.04	RN
Connector-SB-L_2	30	0.36	0.53	0.07		0.82	0.73	0.97	
Connector-SB-L_2	80	0.15	0.14	0.42		0.08	0.07	0.21	
Connector-SB-L_2	120	*0.02	*0.01	*0.03	RN	0.01	**0.01	**0.04	RN
Peachtree-SB-L_1	30	1.00	0.87	0.57		0.50	0.43	0.28	
Peachtree-SB-L_1	80	0.25	0.48	0.36		0.88	0.76	0.18	
Peachtree-SB-L_1	120	0.77	0.55	0.65		0.39	0.28	0.33	
Peachtree-SB-L_2	30	0.59	0.45	0.77		0.71	0.77	0.62	
Peachtree-SB-L_2	80	0.73	0.49	0.13		0.63	0.75	0.06	
Peachtree-SB-L_2	120	0.39	0.32	*0.03	RN	0.20	0.16	**0.01	RN

The two-sided test indicates the rejection of null hypothesis for the three scenarios of State-EB-L_1. This was expected as the univariate model prediction errors were observed similar or lower than multivariate (shown in Figure 66). For remaining five detectors, the null hypothesis is rejected for State-EB-L_2 (120 unit only), Connector-SB-L_1 (80 and 120 units), Connector-SB-L_2 (120 unit only), and Peachtree-SB-L_2 (120

unit only). Rejection of null hypothesis favors the alternate hypothesis that for these cases, the median difference between univariate and multivariate errors are not 0. For the all remaining cases, the results show the mean difference between univariate and multivariate prediction errors are comparable. It is noted that, the statistical test results indicate a significant median differences in univariate and multivariate prediction errors for 80 or 120 unit prediction scenario and not the 30 unit prediction scenario (except State-EB-L_1). This supports that for shorter data gaps (i.e., 30), and many of the medium (i.e., 80 and 120), that the univariate and multivariate model performances are comparable.

Where the one-sided test results rejects the null hypothesis in favor of alternate hypothesis it implies a higher error measure from univariate model predictions compared to multivariate model predictions, indicating better performance of multivariate model for predictions. The null hypothesis is rejected for State-EB-L_2 (i.e., 80 and 120), Connector-SB-L_1 (i.e., 80 and 120), Connector-SB-L_2 (i.e., 120) unit prediction, and Peachtree-SB-L_2 (i.e. 120). The results confirm the hypothesis that multivariate model is often observed to perform better than univariate model for longer unit predictions.

5.4.3 Experiment 2

The performance of the multivariate and univariate model predictions to predict missing data when the traffic pattern is atypical is tested in this experiment. Multivariate and univariate model prediction results are given for May 27th, 2019. Table 11 presents a summary of the findings for all 6 detectors. Figure 67 and Figure 68 present plots for multivariate and univariate model predictions obtained for the entire day of 27th May 2019 at the six detectors. The blue data points show the actual detector data for the Monday,

May 27th atypical traffic pattern. Predictions are made for this day using both multivariate and univariate models.

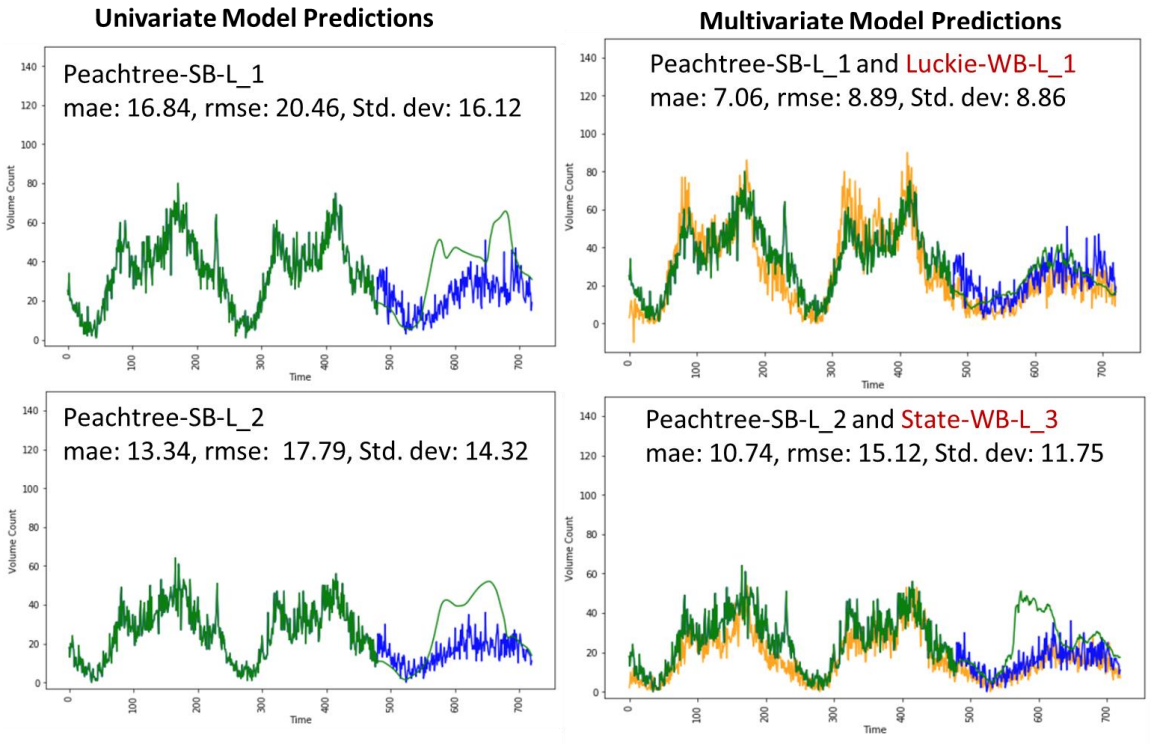


Figure 67 – Multivariate and univariate model prediction performance for a Monday with non-regular traffic pattern for the detectors at Peachtree St. SB approach.

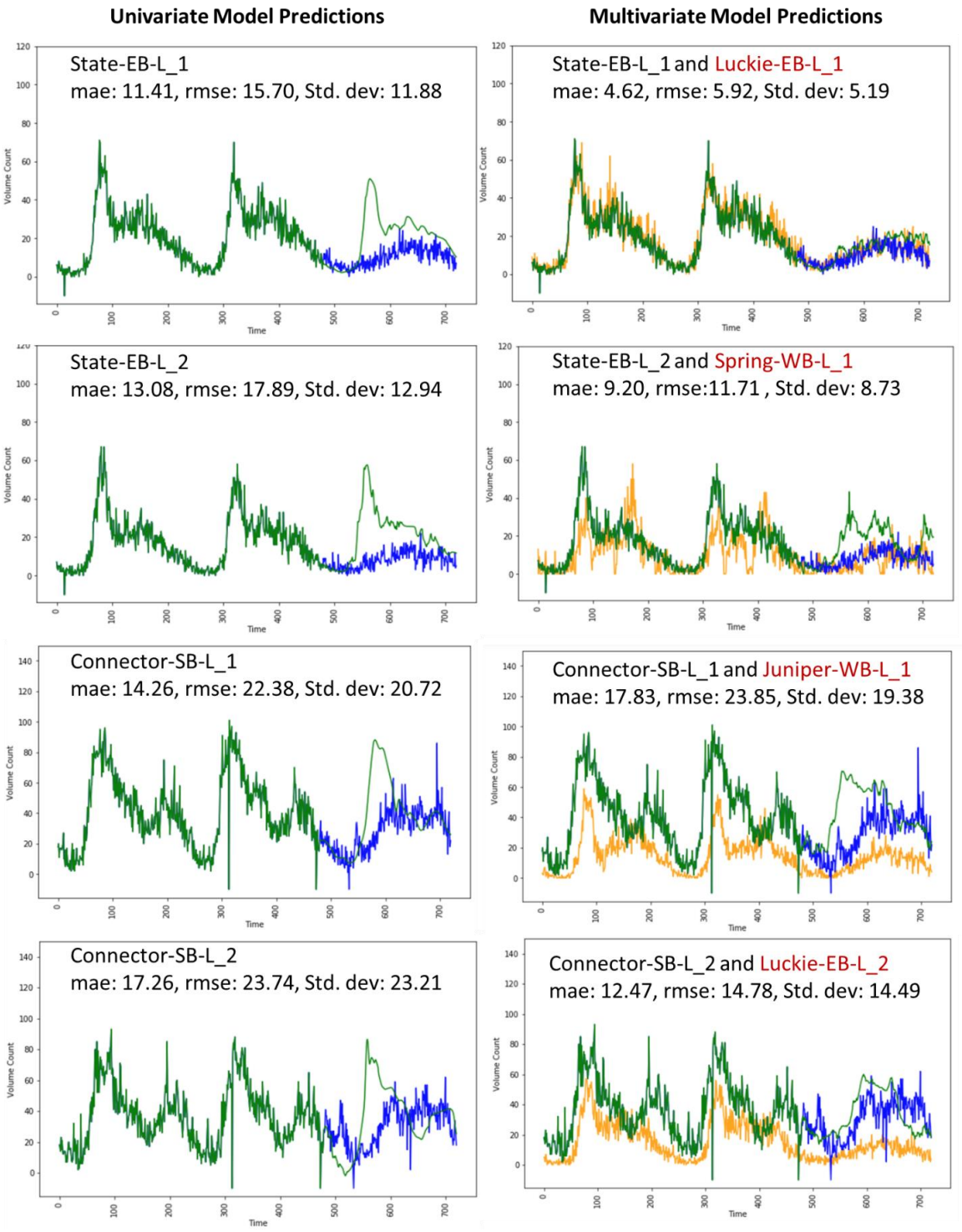


Figure 68 – Multivariate and univariate model prediction performance for a Monday with non-regular traffic pattern for the detectors at State St. NW EB and Connector SB approach.

Table 11 – Performance measures of multivariate and univariate model on atypical day for six approaches

Model Type	Approach	mae	rmse	Std. dev
Univariate	State-EB-L_1	11.41	15.70	11.88
**Multivariate*	*State-EB-L_1*	*4.62*	*5.92*	*5.19*
Univariate	State-EB-L_2	13.08	17.89	12.94
**Multivariate*	*State-EB-L_2*	*9.20*	*11.71*	*8.73*
**Univariate*	*Connector-SB-L_1*	*14.26*	*22.38*	*20.72*
Multivariate	Connector-SB-L_1	17.83	23.85	19.38
Univariate	Connector-SB-L_2	17.26	23.74	23.21
**Multivariate*	*Connector-SB-L_2*	*12.47*	*14.78*	*14.49*
Univariate	Peachtree-SB-L_1	16.84	20.46	16.12
**Multivariate*	*Peachtree-SB-L_1*	*7.06*	*8.89*	*8.86*
Univariate	Peachtree-SB-L_2	13.84	17.79	14.32
**Multivariate*	*Peachtree-SB-L_2*	*10.74*	*15.12*	*11.75*

Comparing the performance measures of prediction accuracies, it is observed that multivariate model provides better predictions than the univariate model at five of six studied detectors – State-EB-L_1, State-EB-L_2, Peachtree-SB-L_1, Peachtree-SB-L_2, and Connector-SB-L_2. At Connector-SB-L1 univariate model provides better prediction accuracy measure. Although based on observation of Connector-SB-L_1 plots in Figure 68, the predictions of univariate model doesn't seem significantly more accurate than multivariate model predictions. The margin by which error values for multivariate model predictions are greater than univariate model predictions are 3.57 (mae), 1.47 (rmse), and -1.34 (Std. dev.). It is hypothesized that the weak performance of multivariate model could be because of over fitting on typical day patterns. Another set of univariate and multivariate models are trained for this detector with a less deep RNN model (2 layers are reduced from original) with an aim to reduce overfitting. Figure 69 shows the results obtained for the new set of multivariate and univariate model performance on atypical day consecutive missing unit predictions.

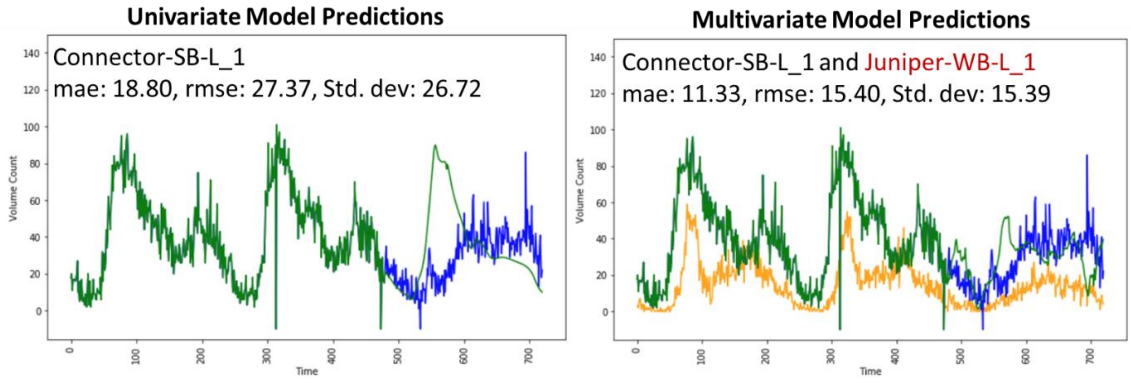


Figure 69 – Multivariate and univariate model prediction on atypical day for Connector-SB-L_1 when trained on less deep network.

The new multivariate model for Connector-SB-L_1 provides improved performance for predictions on the atypical day compared to the original multivariate model. The performance is also better compared to the new and old univariate model prediction results indicating that hyper parameter tuning can improve multivariate model performance. A better predication performance of multivariate model over univariate model at other five detectors could be due to the additional information input model obtains from the other active detector.

5.4.4 Experiment 3

This experiment is conducted to investigate the impact of utilizing multivariate and univariate volume imputations input on vehicle travel times generated by the North Avenue Digital Twin on a regular Monday and on a holiday Monday with an atypical traffic pattern. Nine routes in the simulated corridor are selected to evaluate the impact on vehicle travel times. Figure 70 shows the nine routes along with their assigned route numbers. Six of the nine routes are on mainline, three in eastbound direction (Routes: 59, 60, and 57) and three in westbound direction (Routes: 61, 62, and 58). The remaining three routes are on entry

points in the model: Route no. 75 at State St. NW EB, Route no. 84 at Connector SB, and Route no. 19 at Peachtree St. NE EB.

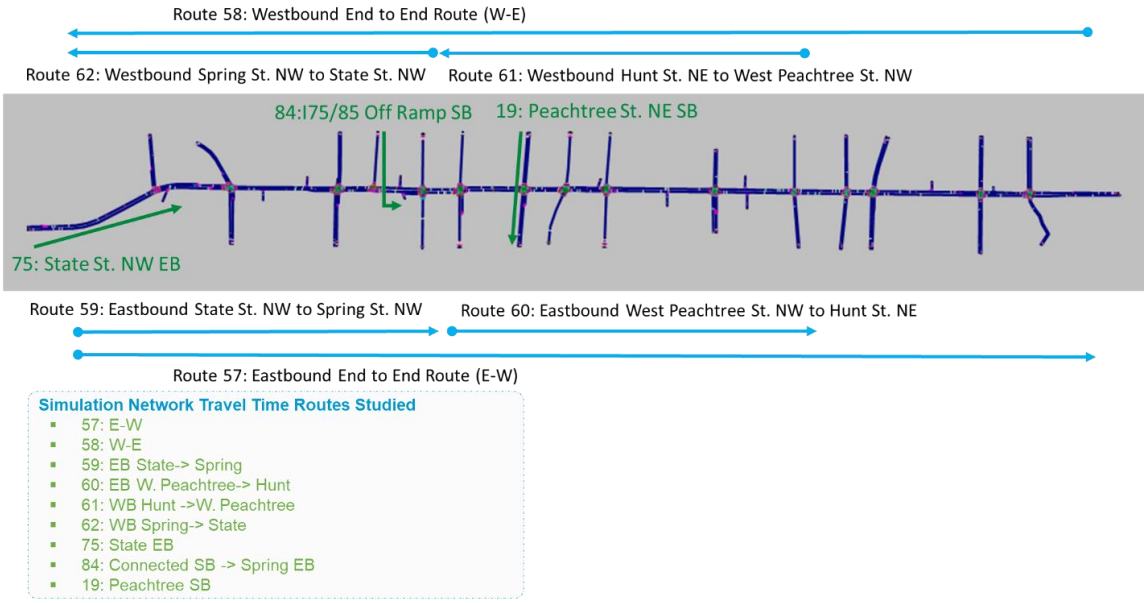


Figure 70 – Corridor Travel time routes.

For the two traffic scenarios, Monday, March 18th, 2019 (typical), and Monday, May 27th 2019 (atypical, i.e. holiday), the impact of data gap imputations during the three-hour PM peak (3PM to 6PM) are evaluated for the nine selected routes. The scenarios are run on the developed Digital Twin model. A three hour data gap is assumed at the three approach entry points listed above. A base day is also run with original data inputs for both days. The only difference between the base day and the multivariate and univariate imputation scenarios is the traffic imputation at the three selected approaches.

5.4.4.1 Base traffic data

The base day data for March 18th and May 27th, 2019 include volume data received from the 147 detectors and signal stream data received from the fifteen intersections.

However, as there were no days with 100% data the base data did require some infill prior to running the experiments. As previously discussed, 29 detectors are found to be inactive (Figure 15 in Chapter 4). Gaps in signal data streams were also observed. For March 18th, 2019, volume estimations at the 29 detectors were based on online traffic data sources: the Automated Traffic Signal Performance Measures (ATSPM) system and available field collected counts. Signal imputations were made to replicate the phase timings of previous available signal cycles. Additional details of the data gaps observed and estimation methodology to create the complete March 18th base day can be found in Chapter 4. To create the May 27th base day similar volume estimation methodologies were utilized. Due to unavailability of the signal data streams on the chosen study day of holiday Monday – 27th May 2019, actuated signal control data for PM peak is emulated in the simulation model utilizing PTV's Vissim 9.00-08 Ring Barrier Controller feature.

5.4.4.2 Data imputation

For the univariate and multivariate model imputed data scenarios the volume input data is predicted at the six detectors associated with the three approaches as discussed in section *Experimental Design.* For multivariate and univariate model imputed data scenarios, a data gap for the entire simulation period of three hours is assumed at the three approaches. Ten replicate trials are run for each of the three scenarios, for each of the two Mondays.

5.4.4.3 Results

Deep RNN univariate and multivariate models developed for each of the six detectors are used to predict volume values for 30 consecutive missing units starting at 3

PM for the typical Monday (March 18th) and atypical Monday (May 27th). Table 12 presents the summary of performance error measures for the univariate and multivariate model predictions on both Mondays.

Table 12 – Error measures for univariate and multivariate model predictions for 30 consecutive units on March 18th and May 27th starting at 3 PM. (An asterisk indicates lower error values among the two model types on typical day predictions and two asterisk indicate lower values among the two model types on atypical day predictions.)

Detector	Model Type	March 18th (Typical Day)			May 27th (Atypical Day)		
		mae	rmse	Std. dev	mae	rmse	Std. dev
State-EB-L_1	Univariate	5.2	6.3	6.3	20.6	22.7	9.6
State-EB-L_1	*Multivariate**	5.0	6.1	6.1	5.2	6.1	4.0
State-EB-L_2	*Univariate	4.8	6.0	6.0	32.0	33.1	8.4
State-EB-L_2	Multivariate**	5.4	7.24	7.2	16.4	18.2	7.8
Connector-SB-L_1	Univariate	32.1	38.3	21.5	40.5	42.9	16.1
Connector-SB-L_1	*Multivariate**	19.4	26.2	20.7	7.0	8.2	8.0
Connector-SB-L_2	*Univariate**	8.7	11.1	10.7	12.2	15.0	13.0
Connector-SB-L_2	Multivariate	10.0	12.8	12.8	23.1	25.1	9.8
Peachtree-SB-L_1	*Univariate**	6.4	8.6	7.4	9.2	11.1	8.1
Peachtree-SB-L_1	Multivariate	7.2	8.6	7.8	9.8	11.3	7.9
Peachtree-SB-L_2	*Univariate**	6.9	8.4	7.5	4.6	6.0	5.6
Peachtree-SB-L_2	Multivariate	8.3	10.6	8.0	12.5	13.3	5.1

From experiment 1, it is observed that that multivariate and univariate predictions tend to be similar on typical day, especially for short 30 consecutive units predictions. This observation is seen with March 18th 30 unit prediction performance in Table 12.

For the atypical day, on State-EB-L_1, State-EB_L_2 and Connector-SB_L_1, the multivariate model errors are clearly lower than univariate model prediction errors, as expected from experiment 2. At Peachtree-SB-L_1, the error values of univariate and

multivariate model predictions on the atypical day are comparable. Figure 71 and Figure 72 show the predictions by univariate and multivariate models of the typical and atypical day for 30 consecutive units starting at 3 PM for Connector-SB-L_1 and Connector-SB-L_2.

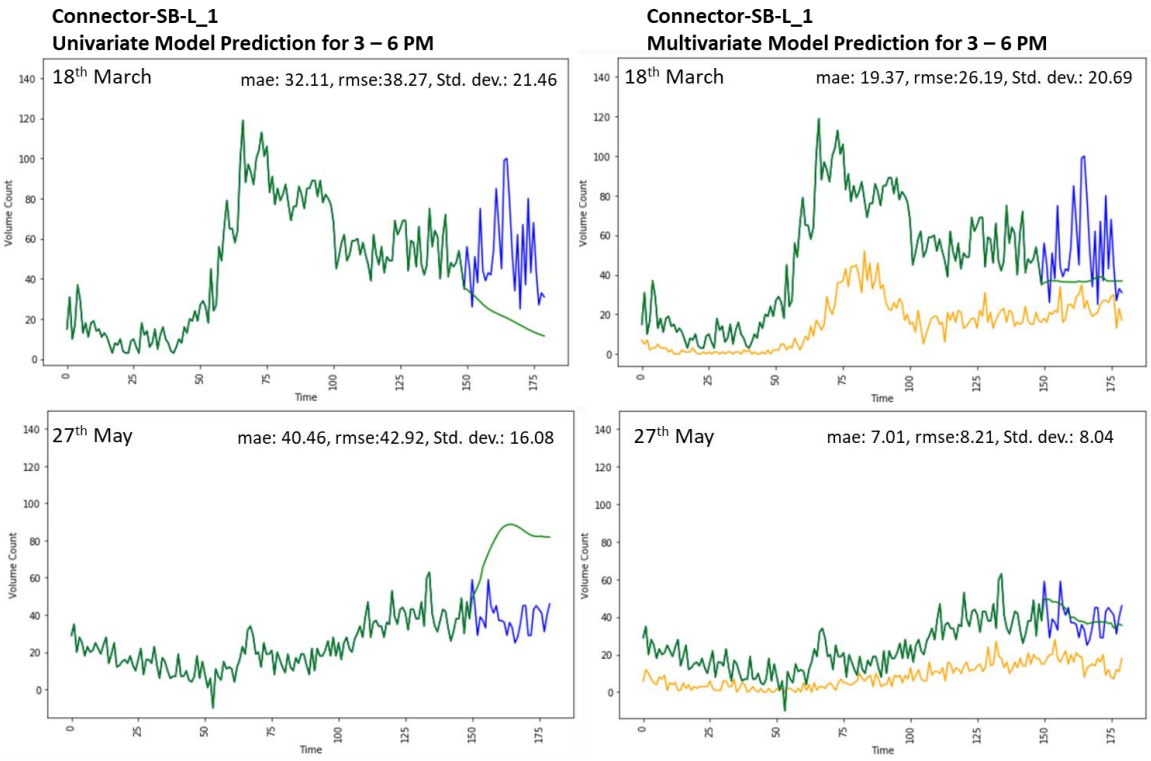


Figure 71 – Univariate and multivariate model predictions for Connector-SB-L_1 for 30 consecutive units starting at 3 PM on a typical Monday (March 18th) and an atypical Monday (May 27th).

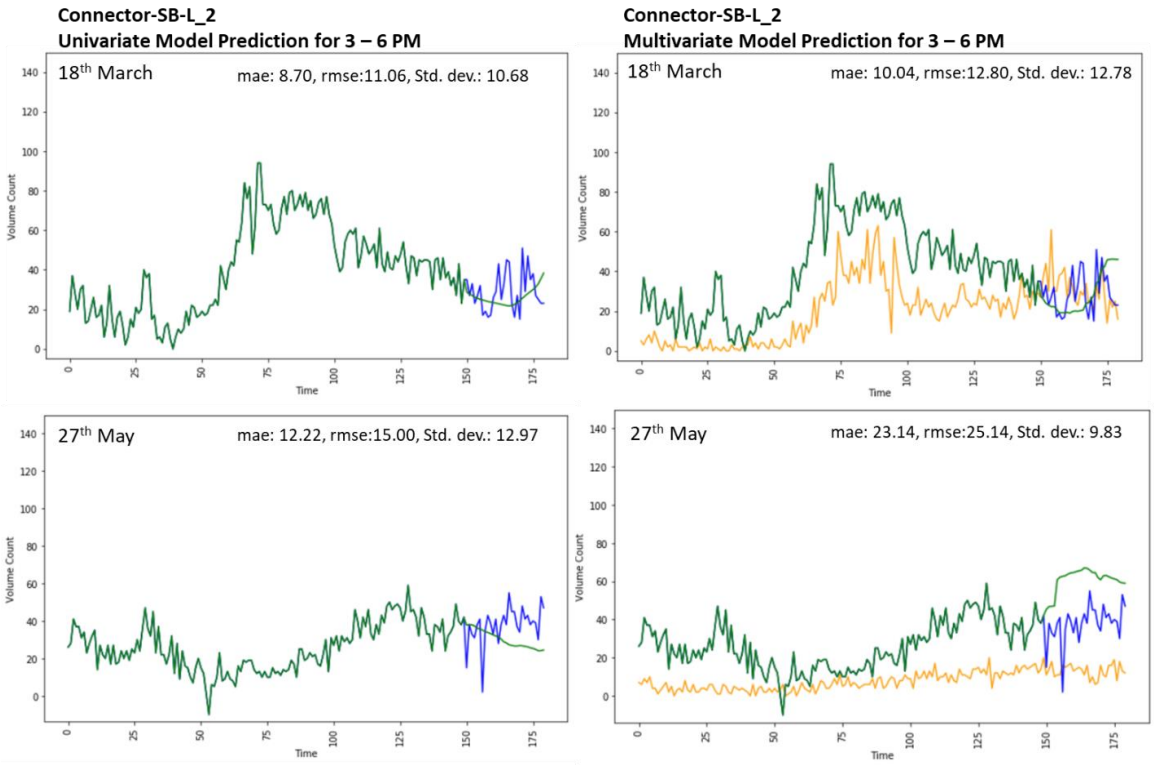


Figure 72 – Univariate and multivariate model predictions for Connector-SB-L_2 for 30 consecutive units starting at 3 PM on a typical Monday (March 18th) and an atypical Monday (May 27th).

At Connector-SB-L_2 and Peachtree-SB-L_2, for the atypical day univariate model prediction errors are observed to be lower than multivariate model prediction errors. This is partially due to the shorter data gap. For instance, for longer gaps, such as 90 units (i.e. 9 hours) the multivariate model would outperform the univariate model at Peachtree-SB-L-2. However, as seen in Figure 73, one reason for the weaker performance of the multivariate model in comparison to the univariate model is a poor correlation between the traffic pattern on the additional detector and the Connector-SB-L-2 detector. This raises an interesting possibility that detectors that are reasonably correlated under typical conditions may not be well correlated under atypical conditions. Future investigations will include, additional atypical training data, tuning hyper parameters to avoid overfitting and to

achieve better optimization, and performing tests on more atypical day validation data to obtain robust results.

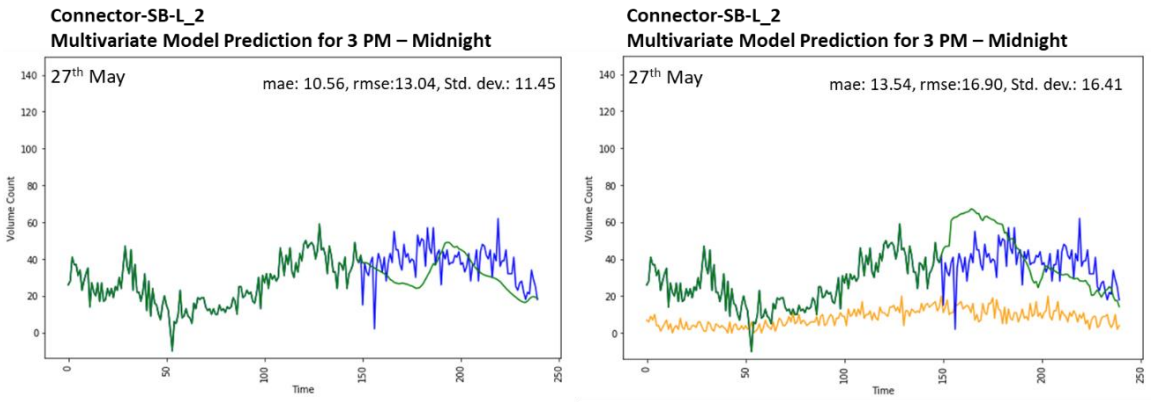


Figure 73 – Univariate and multivariate model predictions for Connector-SB-L_2 for 90 consecutive units (i.e., 9 hours) starting 3 PM on atypical Monday (27th May).

The implemented Vissim vehicle input allows for vehicle volume input at the approach rather than per lane level. Hence, to feed the detector volume data into the simulation model, the volume data at the lane level is aggregated to obtain bin wise volume input data for the three approaches: State St. EB, Connector SB, and Peachtree St. SB. Approach level volume input for 3-6 PM for the base case (actual data), multivariate model predictions case, and univariate model predictions case for the typical and atypical days for Connector SB and Peachtree St. SB are shown in Figure 74 and Figure 75.

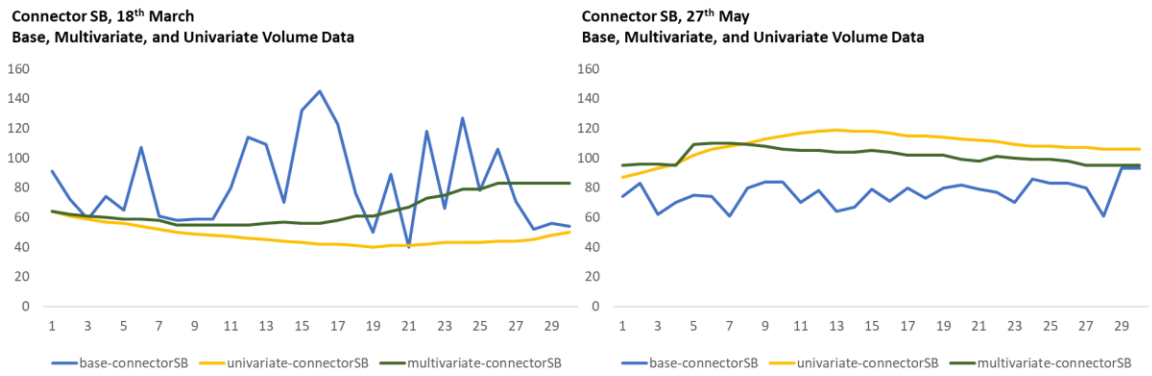


Figure 74 – Approach volume for base case, multivariate model, and univariate model for 3-6 PM on typical day and atypical day at Connector SB.

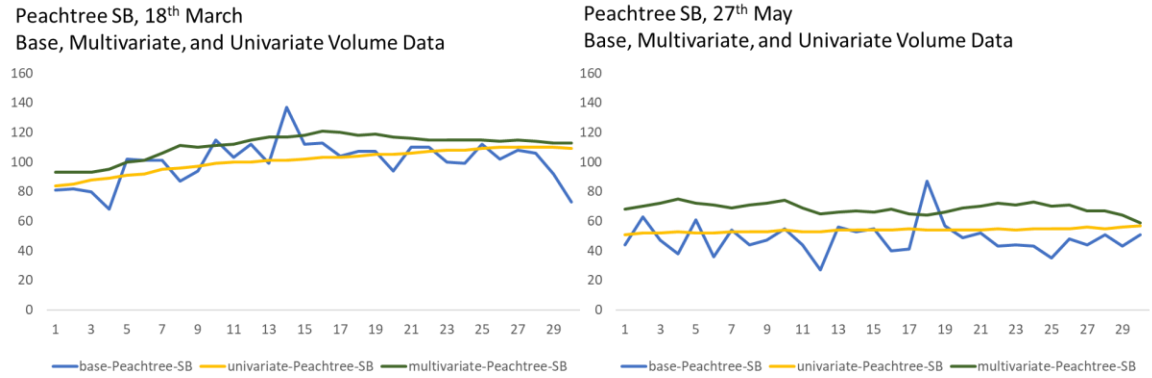


Figure 75 – Approach volume for base case, multivariate model, and univariate model for 3-6 PM on typical day and atypical day at Peachtree St. SB.

Figure 76 shows plots for the volumes generated in simulation model at Connector SB for the 10 replicate base case, multivariate, and univariate trials, for the typical and atypical scenarios. In the Vissim implementation, binned volumes (six minute bins) are provided as input at six minute intervals. Vissim generates the assigned count of vehicle input in the six minute interval, unless the link is oversaturated. The replicate trials differ in the generated distribution of vehicle arrival times over the six minute intervals. A very low difference between the number of vehicles generated and assigned vehicle inputs per six minute interval suggests under-saturated traffic conditions at this entry approach on the

typical and atypical days. Vehicle input volume for most six minute bins in multivariate and univariate predictions case for the atypical day is higher than the base case, as seen in Figure 74. However, this volume is processed in the simulation network as seen in Figure 76, suggesting under-saturated condition at this approach for multivariate and univariate predictions cases as well. Figure 74 and Figure 76 also highlight the smoothing effect of the univariate and multivariate models, with the base data having much more variability between six-minute intervals. Figure 76 also verifies the execution of the simulation scenarios to some extent. A similar trend is observed at the State St. EB approach suggesting under saturated conditions at this approach for all three cases of the two scenarios.

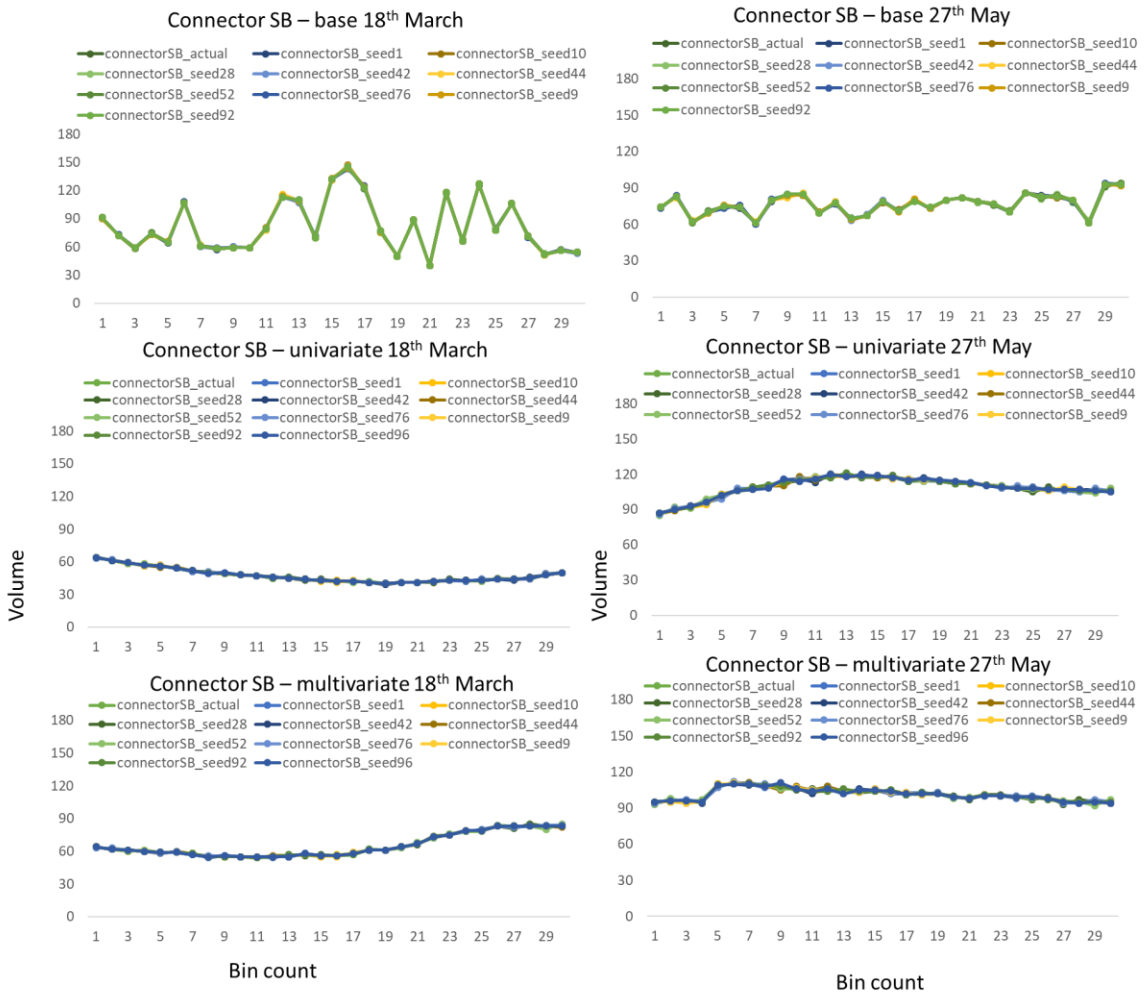


Figure 76 – Typical and atypical day volume count for each of 30 six-minute bins in the simulation period of three hours for 10 replicate trials at Connector SB on base case, model, and multivariate model scenarios.

At the Peachtree St. SB approach, this trend of a very low difference in vehicle volumes generated in Vissim and vehicle volume input assigned in all 10 replicate trials is observed in the atypical day scenario. However, for the typical day scenario volume counts generated in Vissim vary from the assigned input volume for most of the bins, as shown in Figure 77. Variation in six-minute bin volumes between different replicate trials in base case scenario suggests that this approach operates near saturation state in the PM peak period of the typical day. This trend is not observed on the atypical day due to lower

volumes (refer Figure 75). The near saturated condition on the side street approach of Peachtree St. SB could be due to high traffic volume on the mainline during the PM peak period. In Figure 75 it can also be seen that the multivariate predictions for volumes are higher than the base case on typical day. This leads to variations in volumes generated in Vissim for the multivariate predictions case (Figure 77). Much less variation in the generation of traffic volumes is observed in the univariate predictions scenario (Figure 77). This is likely due to lower volume predictions from the univariate model relative to the multivariate model and base case (Figure 75).

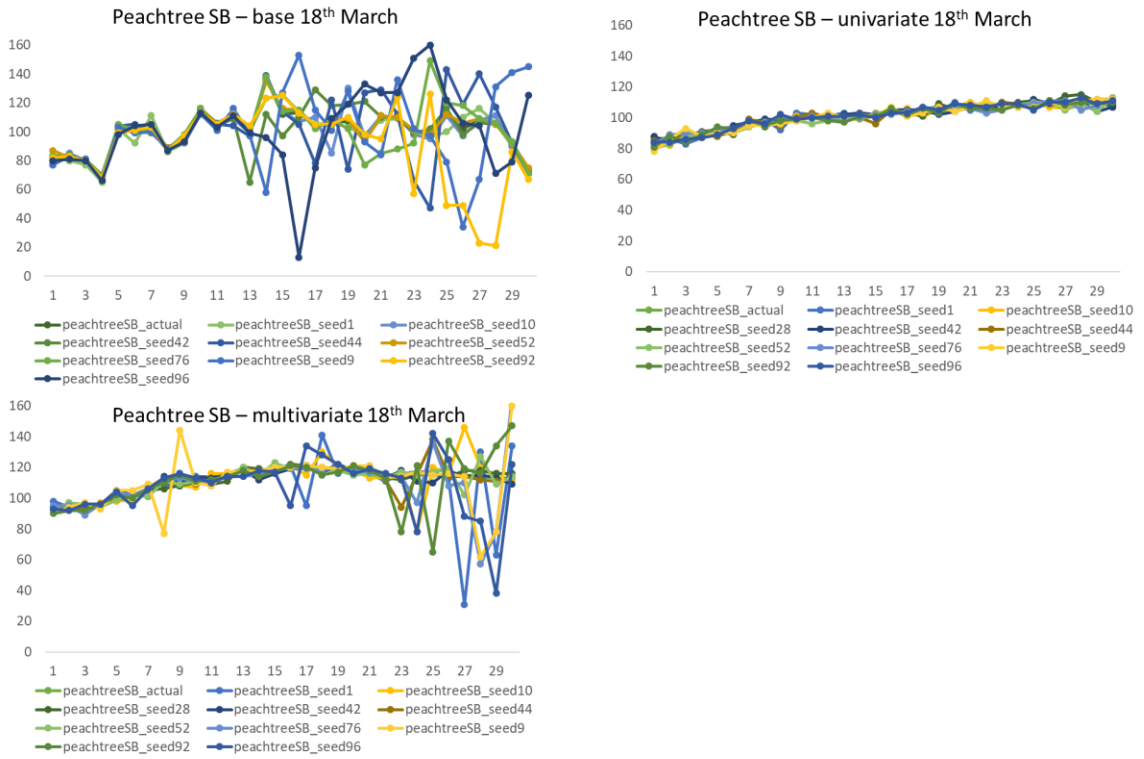
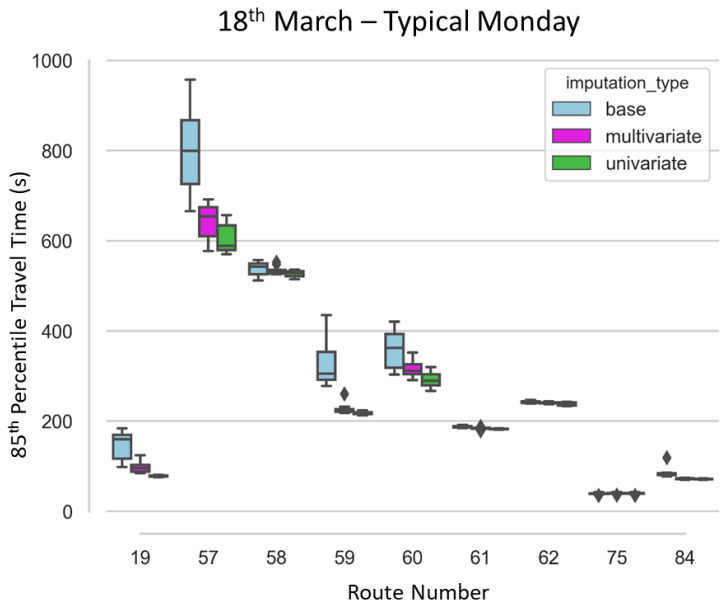
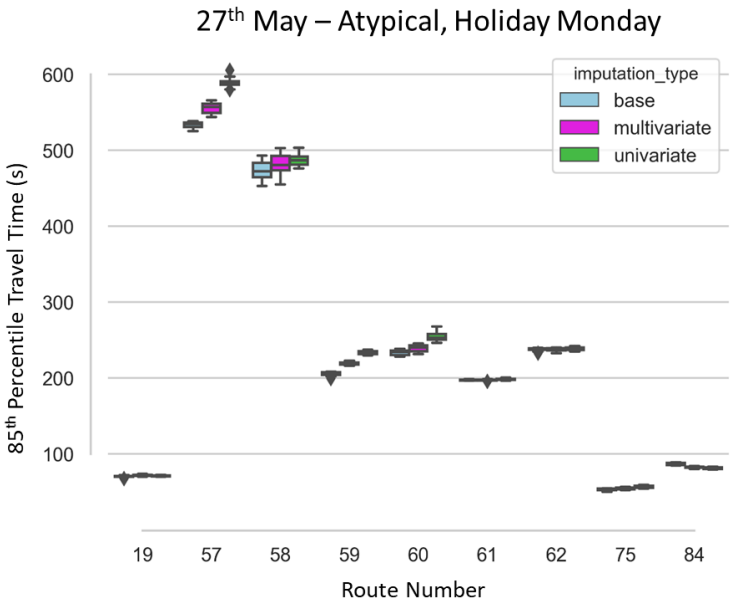


Figure 77 – Variation in simulation generated Peachtree St. SB approach typical day volumes in the six-minute bins between different replicate trails for the three cases: base, univariate model, and multivariate model.

Figure 78 presents box plots of the 85th percentile travel times obtained from the replicate trials at the nine routes for the three data input scenarios for Monday, March 18th (typical) and Monday, March 27th (atypical, holiday).



(a)



(b)

Figure 78 – Boxplots showing variation in 85th percentile travel time at the nine study routes for base data and univariate and multivariate model imputed data for (a) Monday March 18th (typical) and (b) Monday, March 27th (atypical, holiday).

It is observed that for both the typical and atypical scenarios, vehicle travel times generated at the studied routes for the multivariate imputed data scenario is closer to that of the base day data than the univariate imputed data scenario. A lesser variation in the travel time values is observed at the routes for the data input scenarios for the atypical holiday traffic than the typical traffic. Lesser overall traffic volume on base day of holiday Monday in comparison to regular Monday could have contributed to this lack of variation. On the atypical day, the univariate model prediction case has a higher volume for most six minute bins than base case and the multivariate prediction case at both State St. SB (Figure 79) and Connector SB (Figure 74). The increased volumes at Connector SB for the atypical univariate prediction scenario are comparable to the base case volumes on the typical Monday and the increased volume at State St. EB for the atypical univariate prediction scenario is higher than the typical Monday scenarios. However, this increase of volume at only two approaches on the atypical day might not be sufficient to create traffic volume that leads to as much traffic flow and travel time on mainline approach as it is observed for typical Monday.

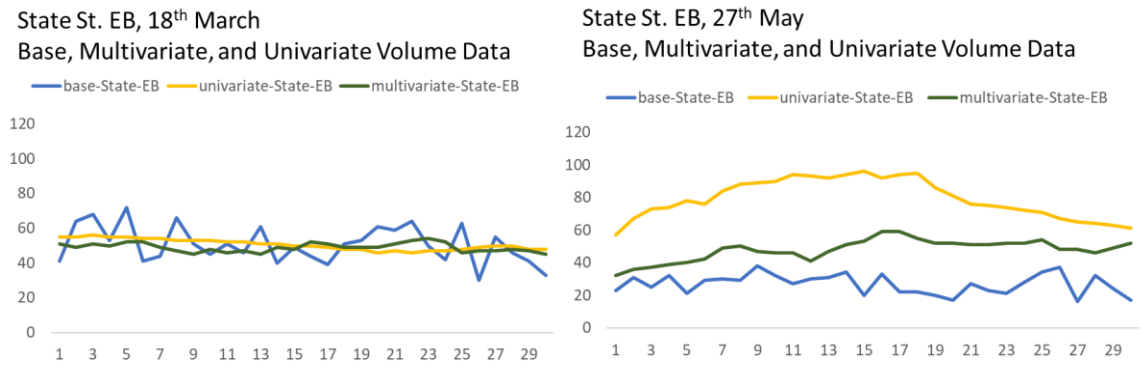


Figure 79 – Approach volume for base case, multivariate model predicted case, and univariate model predicted case for 3-6 PM on typical day and atypical day at State St. EB.

With increased volumes for univariate volume predictions on the atypical day it is expected to see more variation in the travel time or higher travel times on the State St. EB and Connector SB approach routes (Routes 75 and 84). However, only minor differences in travel time values or variation is observed (Figure 78). This could be because the increased vehicle volumes are processed on these side street routes, as suggested from the simulation volume plot for State St. EB in Figure 80 and for Connector SB in Figure 76. However, this increase might be a factor in the increased travel time for the univariate model prediction scenario compared to base volume on Eastbound and Westbound mainline Routes of 57, 58, 59, and 60 for the atypical day scenario (different from trend observed for the typical day case where the base case and multivariate travel time are more than the univariate prediction scenario travel times).

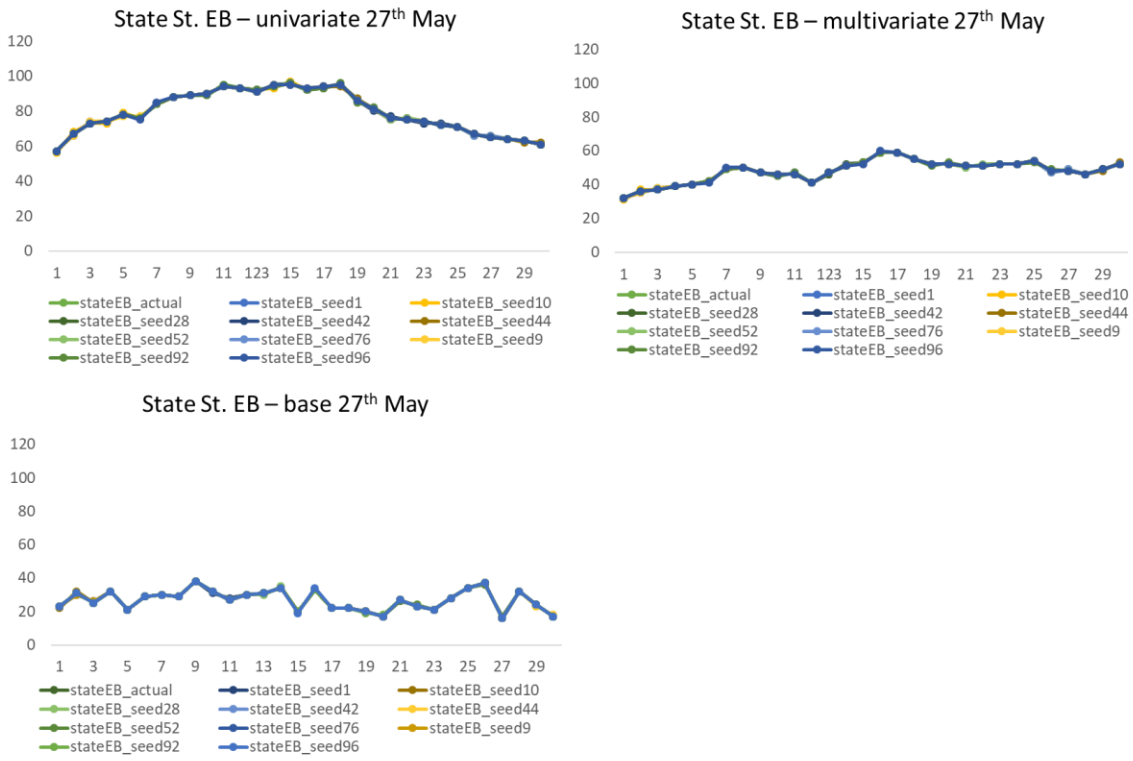


Figure 80 – State St. EB approach atypical holiday traffic variation in simulation generated volumes in the six minute bins between different replicate trails for the three cases: base, univariate model, and multivariate model.

For several routes the variation in travel time is low relative to the other routes. This is likely a result of demands sufficiently below capacity that errors in volume imputation have minimal impact on travel time. This implies that the likelihood of multivariate imputations resulting in more accurate travel time estimates that univariate imputations is more significant for some routes than others. For example, for Routes 57, 59, 60 (Eastbound routes) and 58 (Westbound) route, for both the typical and atypical traffic, the travel time differences are obvious through visual inspection. Here, the vehicle travel times for the multivariate model imputed data scenario are closer to the base day scenario than the univariate model imputed data scenario. While for Westbound routes (61, 62) and entry routes Connector SB (84) and State St. NW EB (75) the differences in vehicle

travel times appear quite small. At Route 19, Peachtree St. NE SB side-street, there are observable travel time differences for the three scenarios under typical conditions; however, minimal differences under the atypical holiday conditions. Investigation of the simulation volume counts at Peachtree St. NE SB for a typical Monday show that this street observes a saturated traffic state (operates at over capacity conditions) for the base day PM peak hour while under the atypical holiday conditions this approach is under capacity. While on the other entry approaches (Connector SB and State St. NW EB) both typical and atypical holiday traffic was under capacity. As expected, it is seen that as demand approaches or exceed capacity the impact of errors in volume imputations are magnified.

A higher variation in vehicle travel times for the base day on a typical Monday could also be due to higher fluctuations in volume counts across 6-minute bins compared to multivariate imputed and univariate imputed bin values which were smoothed, as shown in Figures 74, 75, and 79. The variation in volume across bins on base day is common for the typical and atypical day scenarios; however, higher values on typical day can make the model more sensitive to travel times on congested mainline routes.

The results indicate that the impact of error in volume predictions is more focused on congested routes, with travel time estimates on the atypical day is greater for the univariate predictions scenario compared to multivariate predictions. While the impact is not significant on the routes that are less congested and possess the capacity to process the increased volumes. However, the additional volume could impact travel time estimates for a downstream congested portion.

5.5 Conclusion

Multivariate models can use information from more than one detector time series to provide predictions, thus, indicating a potential of providing accurate estimations even when there are changes in traffic pattern. Towards this, this effort investigates two research objectives: 1) development of multivariate and univariate deep LSTM RNN models to impute data gaps in volume time series data and 2) impact of the multivariate and univariate model imputations on Digital Twin generated travel time values.

Univariate and multivariate predictive models are developed using deep Bi-directional LSTM RNN layers for the six studied detectors. In experiment 1, the ability of the developed multivariate and univariate deep RNN models to impute typical day volumes for four data gap scenarios is tested: 30 intervals (3 hours), 80 (8 hours), 120 intervals (12 hours), and 380 intervals (38 hours). Through visual inspection of the results it was seen that for the shorter gap scenarios (i.e., 30 and 80 intervals) neither model proved consistently superior; however, the multivariate model did provide generally better performance than the univariate model in the long gap scenarios (i.e., 120 and 380 intervals). The hypothesis that deep RNN multivariate model can provide more accurate predictions over the univariate model, under typical conditions, is tested using Wilcoxon's Signed Rank Sum non-parametric statistical test. The test is conducted for three consecutive prediction scenarios: 30, 80, and 120 intervals. A two-tailed test is conducted to test the null hypothesis that there is no significant difference in the median prediction errors provided by multivariate and univariate model and a one-sided test is conducted with the alternate hypothesis that median prediction errors provided by univariate model is greater than that provided by multivariate model. Results from the two-sided and one-sided

tests indicated no significant difference between median univariate and multivariate prediction errors for the 30 unit consecutive prediction scenario. However, the results indicated that the median univariate prediction errors were statistically greater than that of the multivariate model for the 80 and 120 interval consecutive prediction scenario at several detectors. Thus, implying an advantage by the multivariate deep RNN models in utilizing additional information from another detector over univariate deep RNN model to perform imputations of long consecutive data gaps. However, it was also seen that the multivariate model performance would be degraded if the second detector did not correlate well with the primary detector. As well, it is believe additional benefits could be achieve though fine tuning of model parameters. Further, a dependency of model performance on the prediction starting time is also observed from experiment 1, which will need to be investigated to increase robustness of model performance. Finally, additional tests on other corridors should be conducted to further confirm and define the benefit of utilizing multivariate deep RNN model over univariate deep RNN model for time series imputations.

The performance of the developed multivariate and univariate deep RNN models to impute data gap were further explored in experiment 2, focusing on an atypical day of traffic (holiday Monday, May 27th 2019). At five of the six detectors the multivariate model imputations are found to have higher accuracy measures than the univariate model imputations. The additional information from the other detector in the multivariate model likely enabling better imputation accuracy under atypical traffic conditions. Experiment 2 results further indicate the potential benefits of a multivariate over a univariate modeling approach to impute volumes during unprecedented traffic conditions. However, a need for

increased training and validation data for atypical traffic is identified to increase robustness of model performance. In addition, the correlation between the detector with the data gaps and the second detector is again a potential issue. In particular, it is possible that while the two detector are well correlated during typical traffic conditions that the correlation may not hold under atypical traffic. Further exploration of the selection correlated detectors, under varying conditions, is needed. It is also observed that one of the studied detector model's performance on atypical day improved when trained with lesser number of layers to avoid potential overfitting, thus, indicating, hyper parameter tuning can improve model performance, which needs to be studied in future.

In Experiment 3, the impact of multivariate and univariate model imputed data streams on the Digital Twin generated travel times at nine selected routes across the North Ave. Smart Corridor for a typical Monday (March 18th 2019) and holiday Monday (May 27th 2019) is evaluated. For the typical and holiday Monday ten replicate trails are run on the digital twin for three data input scenarios: base day, multivariate model, and univariate model. The results indicate that for both regular Monday and holiday Monday scenarios that the vehicle travel time values for multivariate model imputed data are closer to that of the base day data than the univariate model imputed data travel time values. In addition, during the holiday Monday the univariate model imputations result increased travel times over that of the base and multivariate data, for several of the studied routes. This is likely a result of the higher prediction values by the univariate model. This further indicates the potential benefits of utilizing multivariate imputations, particularly for atypical conditions. Next, it was seen that the difference in simulation generated travel times based on the univariate and multivariate model and univariate model imputations could vary widely by

route. That is, at some routes, the vehicle travel times were similar for both typical and atypical conditions; while on other routes (e.g., 57, 58, 59, and 60) the differences in vehicle travel times are higher. It is hypothesized that the input data variation impacts on performance, due to multivariate and univariate imputations, is greater on comparatively congested routes, as there is less leeway to absorb the additional volume into the traffic stream. Similarly, a higher variation in travel time is also observed when the path is operating at a near saturated traffic scenario. Finally, the observed variation in base day travel times under typical conditions is hypothesized to result from input volumes fluctuations between consecutive intervals; whereas, the multivariate and univariate imputed intervals have a tendency to smooth the input volume variation. This hypothesis is not fully studied in this research effort and remains to be investigated in future.

It is noted that for implementation, if there is a shift in traffic pattern due to an event, infrastructural changes, or travel behavior changes, then the deep RNN model will need to be re-trained on the recent data pattern to adapt to the changes. Re-training of the model might need some amount of hyperparameter tuning, it might still be faster than developing an accurate model using a traditional approach. This is not investigated however, a potential for such findings is identified as needed future work.

5.6 Discussion and Future Work

In this chapter, the ability of univariate and multivariate deep RNN models to impute volume time series data is evaluated under different conditions. However, additional similar comparative analysis is needed at other detectors and on other corridors to derive more generalized conclusions. Further, the developed multivariate deep RNN model

performance in this research utilized two input time series. Experiments to evaluate the performance of multivariate deep RNN models that utilize more than two detector time series data will provide additional insights on the potential benefits of multivariate models. In addition, for the multivariate model development the second detector is chosen based on hierarchical clustering using DTW distance measures. Several detector time series are classified in a cluster and a detector from this cluster is randomly chosen as the second input time series to the multivariate model. For future research, this selection of the second (or more) detectors should be further explored. This could include conditions under which detector correlation is considered (e.g., typical vs atypical traffic), key characteristics of the data (e.g. peak hour demand, relative rate of change of volume, etc.) and so on.

Further, the performance evaluation used to compare univariate and multivariate models in this study are performance error measures for different prediction scenarios. However, it is noted that for imputation implementation of deep RNN model, a model that provides less performance error more often on different typical and atypical traffic conditions, will be a resilient and preferred imputation model. Thus, in future, a performance evaluation parameter that measures the sensitivity of model performance error measures to different traffic conditions will need to be identified. Furthermore, to better evaluate the benefits of utilizing deep RNN based models for time series imputations a comparative study with other traditional time series model imputation methods should be undertaken.

In this study, the model performance is tested on one atypical day traffic pattern, which was holiday Monday. The atypical traffic pattern on the holiday is evident due to significantly less traffic volumes in comparison to the regular Monday traffic patterns.

However, it is also possible to see atypical traffic patterns due to changes in signal timing plans or shifts in route choice impacting traffic demand patterns. The different types of 24 hour time series patterns on regular days can be identified by performing a time series clustering on 24 hour time series of several days. If different groups of such traffic patterns are found, it will be important to include sufficient days from each traffic pattern group to train the model and to test model performance on different identified traffic patterns. Another approach could be to train different models for different identified traffic pattern groups and build an algorithm to choose the appropriate model based on the previous hour(s) traffic pattern window. In this study, the connected corridor studied was deployed with adaptive traffic signal control. Although the traffic signal timing plan is not preset, the volume pattern remains similar on most regular days. This might be due similar traffic demand patterns. However, there are a few days that observed changes in traffic pattern under "regular" conditions, for example in Figure 41 – Cluster 5. It is noted that availability of sufficient historical data is crucial to identify different traffic patterns. Availability of sufficient reliable data can be used to gain insight in understanding different traffic data patterns that could be due to changes in signal timing plans, seasonality, events, or holidays. Further, representative data with different traffic patterns can be used to train the model to provide predictions for different traffic pattern scenarios and to test model performance to develop roust models.

The time series data modeled in this effort was not long enough to observe seasonal variation, however, availability of long term historic data will provide seasonal time series data. The effectiveness of deep RNN over SARIMA to model seasonal time series data will need to be investigated. While LSTM RNN can perform better over ARIMA for time series

forecasting of financial and economic data [190], its' capability to model seasonal traffic volume time series data to provide forecasts needs to be investigated. Liu et al. in a study in 2019 compared ARIMA and LSTM for highway traffic flow prediction and found that LSTM better adjusted to sharp changes in time series patterns compared to ARIMA [191]. Results from an effort by Muzaffar et al. indicate that LSTM RNN can provide improved results in comparison to SARIMA on seasonal time series data for short term electric load forecasts [192]; however, a robust comparative analysis to study LSTM RNN univariate and multivariate models ability to provide forecasts for seasonal traffic time series data in comparison to SARIMA needs to be investigated. While for ARIMA models stationary time series data is required LSTM RNNs do not have this requirement and can learn from the historical fluctuations [191]. However, use of smoothened time series data with lesser deviations in time series data for training univariate and multivariate LSTM RNNs can have potential benefits, which can be evaluated in future.

In addition, to increase robustness of the multivariate time series model performance under atypical traffic, the following should be undertaken: 1) include atypical traffic data in time series similarity measure to choose the input time series for the model, 2) increase training data for the atypical day traffic pattern, and 3) use a systematic approach for hyperparameter tuning to enhance model performance. Finally, to improve the digital twin performance a methodology to impute data gaps in signal time series should be developed and the developed imputation models should be integrated with the built real-time data-driven Digital Twin model architecture.

In near future, increased numbers of high frequency data streams are expected. This study provides findings on using deep learning on such high frequency data to provide

accurate imputation values. Similar deep learning based model may also be useful in learning patterns in other types of traffic data, such as signal indication data.

CHAPTER 6. CONCLUSIONS, LIMITATIONS, AND FUTURE WORK

This chapter summarizes the key conclusions, limitations, and future research areas identified from the studies presented in previous chapters. Three studies are conducted with an overarching goal to investigate leveraging of high frequency data received from a connected corridor to drive a real-time traffic simulation model and generate corridor performance measures to provide dynamic feedback to users.

6.1 Research Study 1

Research Study 1, presented in Chapter 3, included development of a real-time connected corridor data-driven traffic simulation model.

6.1.1 Conclusions

The developed simulation model architecture consists of four primary components: 1) Raw Data Stream Processing Module, 2) Dynamic Data-Driven Traffic Simulation Module, 3) Dynamic Performance Measures Estimation and Visualization Module, and 4) Data Request Management Module. The dynamic link between modules 1, 2, and 3 is facilitated by module 4. This dynamically linked architecture allows the traffic simulation model to use connected corridor volume and signal indication data to drive the simulation and to provide simulated vehicle information (e.g., position, speed, acceleration, etc.) to the performance measure estimation and visualization module. The built framework can also be used to develop an understanding of the impact of data imputations performance

estimations generated from the model. The model architecture performance was found to be maintain the near real-time rate for the corridor. The conclusions from this study are:

6.1.1.1 On Model Architecture

The initial pilot study investigates the feasibility of driving the traffic simulation model with hybrid data, a mix of preset and real-time data is investigated. Performance measures obtained from simulation runs driven using only preset data and driven using a mix of preset and real-time data are found to be comparable and to follow similar trends with simulation time. However, there were some differences in performance measures calculated, potentially *indicating the importance of using real-time data where real-time feedback is desired*. Thus, using a hybrid data approach to drive a simulation model was found to be feasible. This implied the possibility of using hybrid data when real-time data is not available for all intersections.

Next, the model architecture is updated to drive fifteen intersections of the modeled network with the real-time data streams. Vissim COM code is used to enable all intersection entry volumes and signal timings, driven using the real-time data streams. The COM logic to update the signal status performs signal state changes by creating flag variables. Initial, to conduct long duration runs the architecture, the Vissim traffic simulation generated readable-only ".fzp" output file that contained all vehicle record information. However, this was found to create a memory obstacle. The COM logic was then modified to post vehicle record data for every simulation second in a separate lightweight JSON. This modification allowed for deletion of previous files for which performance measures were previously processed.

Initial investigation of the real-time volume and signal data obtained revealed the presence of data gaps. In this dissertation, an investigation of data gaps and the impact of imputations on the simulation model generated travel time measures is conducted for the volume data streams (Chapters 4 and 5). However, prior to this the feasibility of the simulation model using Vissim actuated signal timing control to address outages in the signal data stream is investigated. Vissim's COM and RBC (Ring-Barrier-Control) modules are used to create a fallback when a signal data gap is detected. This allows comparatively more reasonable performance measures from simulation model. The built architecture is found to perform at a near real-time rate. Thus, *this study demonstrates the feasibility and potential of using connected corridor data to drive a real-time traffic simulation model to provide and visualize traffic and environmental performance measures at a near real-time rate.*

Overall, the key challenges for the smooth functioning of a real-time data-driven traffic simulation model architecture are in maintaining synchronicity and efficiency in 1) the real-time raw data processing and the processed data injection process, 2) assimilating newly added data in the next simulation step of running the simulation model, and 3) using high frequency simulation output data to generate performance measures. Usage of database queries to access and process raw data, and the usage of an external data request manager (Flask micro web framework) to handle data requests were found to be crucial in ensuring (near) real-time rate of the architecture. However, additional architectural challenges that need to be address to further increase the model speed remains for future work.

6.1.1.2 On Vissim Traffic Simulation Model:

Vissim COM module's capabilities are successfully utilized to dynamically change the input volumes and signal states. The computational time required by Vissim each time step depends on number of changes being made in the model. For example, implementation of each signal change requires additional CPU time to reflect addition signal state change requests. Hence, the traffic simulation speed fluctuates. In addition, suspending the graphical user interface and activating Vissim's quick mode increased Vissim's simulation performance rate. The Vissim COM module is also used to create the fallback to preset actuated signal timing in the simulation model when data gap in signal data stream is observed.

6.1.1.3 Model Performance

The speed of built simulation model with fifteen intersections driven using real-time data ranges from 0.8x to 1.3x. Over time the simulation model is observed to lag real-time due to slower than wall clock rate. It is hypothesized that one of the reasons that the simulation might be slowing down is the increased number of volume and signal indication assignment operations executed in a simulation second.

6.1.2 Limitations and Future Research Direction

1. The architecture developed so far uses a single instance on a simulation model. In general, traffic microscopic simulation models do not inherently use a parallel architecture; hence, they do not scale well with increases in the size of the network. As the size of the network increases, the model will run slower and slower until a point is reached where the simulation clock will be slower than the wall clock. If wide area simulation is required, then the architecture

will need to integrate a distributed simulation structure. Alternately, the advantages of using multithreading and multiprocesssing to execute dynamic changes in the simulation model should be explored. Further, in future, a methodology to simulate the prediction state by creating a clone of current simulation state can be explored.

2. The Vissim traffic simulation model is not field validated and calibrated for Atlanta driving behavior. In future, it is imperative to validate the simulation model to increase the credibility of the environmental and traffic performance measures generated from it.
3. Currently the simulation model uses the last six minute volume aggregate and signal indication aggregate data to simulate, thus, operating at least six minutes behind real-time. One minute aggregate volume data would allow the simulation to run at a lag of at least one minute from wall clock. This lag can be further reduced if the real-time data available from connected corridor is per-vehicle-record (PVR) data. PVR data can allow for a real-time simulation and can potentially improve the accuracy of the simulation generated performance measures. Future research will seek to explore the use of PVR data to enhance simulation performance and accuracy.
4. As discussed, a fallback using Vissim's RBC module is created to address signal outages. This allows simulation to impute the signal data stream gap with preset actuated signal timing plan. However, impact of this imputation approach on traffic and environmental performance measures has not yet been investigated. Future research on measuring impact of this data imputation approach on simulation generated performance measures is suggested.

In this study the real-time data investigation during model development process revealed presence of data gaps. Deriving from this insight Research Study 2 focused on

investigating volume data streams for the presence of data gaps and measuring the sensitivity of volume data imputations on simulation generated travel time measures. The main conclusions from second study are in following subsections.

6.2 Research Study 2

A sensitivity analysis experiment is conducted to measure the impact of volume data imputations on simulation generated travel time measures (presented in Chapter 4). The main conclusions from this second study are in following subsections.

6.2.1 Conclusions

6.2.1.1 On Real-Time Data Streams

Investigation of the real-time volume and signal status data streams revealed data gaps. Two types of data gaps are identified in the volume data stream: permanent data loss and intermittent data loss. Similar data gaps are found in signal data streams.

6.2.1.2 On Sensitivity Analysis

The sensitivity analysis experiment is performed to study the impact of errors in the volume data imputation values on the simulation generated travel times. Results were obtained for three error percentage values in volume imputations (i.e., 20%, 50%, and 80%) for five missing volume patterns. The patterns were generated based on probability of observing a missing volume pattern in the studied 112 days. The results revealed that some routes are more sensitive to volume imputations in comparison to others. Also, the severity of the impact to mainline travel times was dependent on the approach that contained the

missing volume data. Study results from the experiment identified the set of approaches with the highest impact on the mainline route travel times, for the patterns studied.

In addition, the impact of volume imputations on intermittent data loss patterns is studied for one selected approach, Pattern 3, Techwood Dr. (SB). A hypothesis test (one-sided t-test for mean 0) is conducted on the differences in the 85th percentile travel time values, at the Techwood Dr. (SB) route for "*with intermittent data loss*" and "*without intermittent data loss*" volume data, for the three data imputation error cases: 20%, 50%, and 80%. For the 20% data imputation error case, the test result failed to reject the null hypothesis that there is no impact of intermittent loss data imputations on travel times. While for higher data imputation error cases of 50% and 80%, the test result rejected the null hypothesis in favor of the alternate hypothesis that the volume imputations for intermittent data loss increased the travel times on the Techwood Dr. (SB) route. This demonstrated that for approaches with intermittent data loss patterns higher errors in data imputation values can impact simulation generated travel times. The incorporation of historical detector data for improved imputation with intermittent data loss can be used to better predict the missing volume data.

6.2.2 Limitations and Future Work

1. First, it is recognized that the approach based results are limited to the studied corridor. However, this effort revealed the need to study the real-time data characteristics of any corridor, to understand and account for issues such as data gaps, data fidelity, etc. Future research is needed to better understand and generalize issues related to the accuracy and usability of connected corridor real-time data.

2. To improve the generalization of the experiment results, similar sensitivity experiments should be conducted on real-time data-driven simulation models of other connected corridors.
3. Another limitation of the results from the built model architecture performance is that the real-time data used to drive the simulation model has not been field verified. Hence, future work includes the verification of the data accuracy in the connected corridor real-time data stream. Overall, there exists a need to determine parameters that measure the integrity of connected corridor data.
4. While investigation of the studied real-time data streams revealed the presence of data gaps in both volume and signal indications data streams, this dissertation focused on researching aspects of volume data streams. Future efforts will investigate imputation methodologies for the signal indication data.
5. To improve the feasibility of using the developed model to obtain insights on environmental performance measures, the impact of data imputations on these measures also needs to be investigated in the future.
6. Lastly, realizing a need for data imputation in the volume data stream gaps to drive the real-time data-driven simulation model, the model architecture will need to include a real-time imputations component in future.

The sensitivity experiment conducted measured the impact of error in data imputations on simulation generated travel time measures. However, the data imputation algorithm tested is simple. A need to develop a more accurate data imputation methodology is identified. Towards this end an LSTM RNN model is developed and its' performance is investigated in Research Study 3 (Chapter 5).

6.3 Research Study 3

Deep RNN models using bidirectional LSTM RNN layers for univariate and multivariate time series are developed for six detectors in the third research study presented in Chapter 5. Three experiments are conducted that aim to investigate: 1) development of multivariate and univariate deep RNN models to impute consecutive data gaps in volume times series data, 2) performance of the developed multivariate and univariate deep RNN models under typical and atypical traffic conditions, and 3) impact of the multivariate and univariate model imputations on Digital Twin generated travel time values..

6.3.1 Conclusions

6.3.1.1 Experiment 1

Univariate and multivariate predictive models are developed using deep Bi-directional LSTM RNN layers for the six studied detectors. In experiment 1, the ability of the developed multivariate and univariate deep RNN models to impute typical day volumes for four data gap scenarios is tested. Through visual inspection of the results it was seen that for the shorter gap scenarios neither model proved consistently superior; however, the multivariate model did provide generally better performance than the univariate model in the long gap scenarios. The hypothesis that deep RNN multivariate model can provide more accurate predictions over the univariate model, under typical conditions, is tested using Wilcoxon's Rank Sum non-parametric statistical test. It is seen that there is no significant difference between median univariate and multivariate prediction errors for the shorter prediction scenario while the median univariate prediction errors were statistically greater at several detectors for the longer scenarios. Thus, an advantage is seen by the

multivariate deep RNN models in utilizing additional information from another detector to perform imputations of long consecutive data gaps. However, it was also seen that the multivariate model performance would be degraded if the second detector did not correlate well with the primary detector. As well, it is believe additional benefits could be achieve though fine tuning of model parameters. Finally, additional tests on other corridors should be conducted to further confirm and define the benefit of utilizing multivariate deep RNN model over univariate deep RNN model for time series imputations.

6.3.1.2 Experiment 2

The performance of the developed multivariate and univariate deep RNN models to impute data gap were further explored in experiment 2, focusing on atypical traffic. At five of the six detectors the multivariate model imputations are found to have higher accuracy measures than the univariate model imputations. The additional information from the other detector in the multivariate model likely enabling better imputation accuracy under atypical traffic conditions. Experiment 2 results further indicate the potential benefits of a multivariate over a univariate modelling approach to impute volumes during unprecedented traffic conditions. However, a need for increased training and validation data for atypical traffic is identified to increase robustness of model performance. In addition, the correlation between the detector with the data gaps and the second detector is again a potential issue. In particular, it is possible that while the two detector are well correlated during typical traffic conditions that the correlation may not hold under atypical traffic. Further exploration of the selection correlated detectors, under varying conditions, is needed.

6.3.1.3 Experiment 3

In Experiment 3, the impact of multivariate and univariate model imputed data streams on the Digital Twin generated travel times at nine selected routes across the North Ave. Smart Corridor for a typical and atypical day is evaluated. The results indicate that for both traffic conditions that the vehicle travel time values for multivariate model imputed data are closer to that of the base day data than the univariate model imputed data travel time values. In addition, during the atypical day the univariate model imputations result in increased travel times over that of the base and multivariate data, for several of the studied routes. This is likely a result of the higher prediction values by the univariate model. This further indicates the potential benefits of utilizing multivariate imputations, particularly for atypical conditions.

Next, it was seen that the difference in simulation generated travel times based on the univariate and multivariate model and univariate model imputations could vary widely by route. That is, at some routes, the vehicle travel times were similar for both typical and atypical conditions; while on other routes the differences in vehicle travel times are higher. It is hypothesized that the input data variation impacts on performance, due to multivariate and univariate imputations. Similarly, a higher variation in travel time is also observed when the path is operating at a near saturated traffic scenario. Finally, the observed variation in base day travel times under typical conditions is hypothesized to result from input volumes fluctuations between consecutive intervals; whereas, the multivariate and univariate imputed intervals have a tendency to smooth the input volume variation. This hypothesis is not fully studied in this research effort and remains to be investigated in future.

6.3.2 Limitations and Future Work

Conducting the three experiments, potential advantages of the multivariate model over the univariate model are identified. However, the results are limited. Presented next are the limitations of these results and future work identified to develop a robust understanding of multivariate deep RNN model development and performance.

1. Firstly, the results obtained are for models developed on six detectors of the one study corridor. In future, to increase generalizability of using deep RNN models a similar investigative study should be conducted for more detectors from other connected corridors. Further, the model in this study is developed using the connected corridor data that has not been field verified. Before imputing volume data values to drive Digital Twin field data should be verified.
2. Secondly, the results on potential advantages of using a multivariate model over a univariate model on a typical day and atypical traffic day are based on a pilot model architecture configuration. The model development and performance of the deep RNN multivariate and univariate models need to be further investigated by tuning model hyperparameters to develop a roust understanding of the comparative performances.
3. Thirdly, the studied deep RNN models are trained on typical days only. Advantages of training models on some atypical days is yet to be explored to improve model performance on atypical days. Further, to improve model robustness to impute traffic on atypical days, more atypical day patterns should be used. Next, the dependence of model performance on prediction start time on atypical day needs to be investigated in the future to improve the understanding of model performance and robustness.

4. Fourth, all the multivariate models developed and investigated in the study use two similar time series from the cluster created by hierarchical clustering using DTW. Research is needed to identify parameters to choose similar (correlated) time series and to measure the impact of correlation in multivariate input time series on multivariate model performance. Further, to improve generalizability of the model, exploring the performance of using more than two time series for multivariate models is suggested.
5. Lastly, while the simulation study results imply benefits and potential of using deep RNN based models for connected corridor volume data imputation to drive the Digital Twin, the imputation model is not plugged into the real-time architecture. In the future, the model will need to be plugged into the Digital twin architecture to operate at near real-time rate. Further, the impact of univariate and multivariate imputations on simulation generated environmental performance measures also needs to be studied.

CHAPTER 7. CONTRIBUTIONS

7.1 Development of Dynamic Real-Time Data-Driven Traffic Simulation Model – Digital Twin

Developed and implemented a methodology for a dynamically driven traffic simulation model, using PTV Vissim 9.00's COM module and programming language Python 3.7. This contribution involves a number of aspects.

1. The methodology involved linking the real-time aggregate volume and signal status data to a traffic simulation model, during run time.
2. The methodology creates a time interval approach for processing network input values. The current time bin interval is six-minutes, however this is changeable based on the data stream aggregation level.
3. The methodology also reflects the variability of real-time field data by dynamically routing vehicles through intersections based on current field conditions.
4. The methodology is capable of dynamically updating of signal status at the signal heads for all approaches, every one-simulation second, based on the real time data streams.
5. Critically, the approach is dynamic, responding to traffic in near-real time, and capable of handling volume inputs and signal status at different frequencies, which can be changed depending of granularity of the real-time data streams.

It is highlighted that these contributions cover the range of big data attributes: 1) Volume – handles large volumes of data; 2) Velocity – data is streamed in real-time; 3) Variety – a number of data format are integrated; 4) Veracity – addresses data quality

(missing data); 5) Variability – data varies over time, following typical and atypical patterns; 6) Visualization – the architecture includes performance metric visualization; and 7) Value – critically the method converts the data into useful information that may be used to evaluate the corridor operations.

7.1.1 Development of Platform for Testing Smart Connected Technology

A key feature of the developed real-time simulation platform is that additional smart technologies, implemented in the field, may be integrated into the model. For instance, during one period of the study the City of Atlanta was operating an adaptive control system on the corridor. Using this platform it is possible to simulate the adaptive signal control system without having the adaptive logic imbedded in the simulation. In many instances, this may be the only means to collect data on such smart technologies within a simulation environment.

7.2 Identification of Key Challenges in Utilizing Real-Time Connected Data Streams

Studying the real-time volume data streams revealed several challenges posed by real-time high-frequency data from connected infrastructure, such as intermittent data gaps, filtering requirements, aggregation of differing formats, etc. These are challenges that need to be tackled by the transportation researcher and engineering community to fully leverage the benefits of connected corridor data. The study also provided insight on the necessity of connected corridor real-time data stream investigation as one of the initial steps in the utilization of the data streams for driving a simulation model.

7.3 Identification of Key Limitations as Next Research Steps in the Development of a Connected Corridor Data-Driven Traffic Simulation Model – Digital Twin

This effort highlighted the key limitations in real-time connected corridor modeling. The key next research focus areas identified are: 1) Development of intermittent data imputation methodologies that leverage the presence of high frequency historical data, 2) Investigate the reasons for gaps in data stream and identify methods to minimize data loss, 3) Apply parallel computing techniques to distribute the simulation load and improve model architecture runtime, allowing for the simulation of a larger network at (near) real-time pace, and to 4) Devise a methodology to simulate the prediction state by creating a clone of current simulation state.

7.4 Developed a Methodology to Prioritize Locations that Need Data Imputation

The sensitivity analysis study conducted to measure the impact of imputed volume data on simulation generated travel time revealed that a subset of approaches are mainly responsible for the majority of impacts. The results also showed certain routes in the network to be more sensitive to volume imputations. Applying such an analysis allows for the prioritization of implementation and maintenance of smart technologies, which is particularly critical in today's resource constrained environments. This methodology can be utilized to prioritize connectivity of the volume data streams or developing more robust and accurate methodology for imputations.

7.5 Development and Performance Investigation of Deep RNN Univariate and Multivariate Time Series Models for Data Imputations

The presence of data gaps in connected corridor volume data streams can impact traffic and environmental performance measures produced from a Digital Twin. To facilitate meaningful insights from the simulation generated performance measures a need to develop a data imputation methodology is identified. Bidirectional RNN LSTM layers are used to model univariate and multivariate time series to provide imputation estimates for missing data. The performance of developed univariate and multivariate model to provide imputations from missing consecutive data is investigated on a typical traffic day and on an atypical traffic day. Experiment results indicated potential advantages of using information from additional input time series in the multivariate model to obtain more accurate estimates when imputing longer consecutive missing data and when traffic is atypical compared to imputation estimates by the univariate model. Further, the impact of the using imputations from the developed univariate and multivariate model on model generated travel time measures of nine selected routes is evaluated for typical day traffic and atypical day traffic. Results indicated more accurate travel times with multivariate model imputations versus univariate model imputations, on an atypical traffic pattern day.

The developed deep RNN models can be used to learn from previous traffic patterns and provide imputations for real-time data gaps. Further, the study demonstrates how developed multivariate deep RNN model leverages information from another detector of the corridor to provide better estimates for predicting atypical day traffic patterns. Thus, the developed methodology reveals a potential of providing more accurate estimations even when there are dynamic changes in the traffic pattern. If due to an event or infrastructural changes, there is a shift in traffic pattern, then the model can be re-trained on the most recent data to adapt to the changes. While re-training of the model when the traffic pattern

changes might need some amount of hyperparameter tuning, it might still be faster than developing a model using a traditional approach. While this is not investigated, this study reveals a potential for such findings, thus identifying needed future work. Such deep learning based model could be also used to learn patterns in other types of traffic data such as signal indication data. Considering the high likelihood of increasing numbers of high frequency data streams in near future, this study provides findings on leveraging high frequency data to learn patterns and provide accurate imputation values.

7.6 Identification of Future Work for Applying Deep RNNs to Model Traffic Time Series Data

Results obtained from the experiments conducted in the Deep RNN study (Chapter 5) revealed future work needed to improve performance of deep RNN models that can be used to model traffic time series data. For example, a dependency of model performance on the prediction starting time is observed, thus, revealing a need to investigate this dependency to increase robustness of model performance. A need to include more atypical traffic pattern data to improve model development and model testing is also identified. Further, the impact of hyperparameter tuning to improve model performance by increasing model generalizability and reducing model overfitting is also identified. Lastly, the impact of correlation in input time series data used to develop multivariate model on performance needs to be investigated. With rise in traffic data received, future research to investigate the application of deep RNNs to model traffic data is crucial.

REFERENCES

1. TheWorldBank. *The World Bank Data*. 2018 [cited 2019 14 August]; Available from: https://data.worldbank.org/indicator/SP.RUR.TOTL.

2. UnitedNations. *World Urbanization Prospects 2018*. United Nations DESA/Population Division 2018 [cited 2019 14 August]; Available from: https://population.un.org/wup/DataQuery/.

3. Eremia, M., L. Toma, and M. Sanduleac, *The Smart City Concept in the 21st Century*. Procedia Engineering, 2017. **181**: p. 12-19.

4. Charles, A. *Cities need to innovate to survive. Here are four ways they can do it*. 2017 10 Feb 2017 [cited 2019 10 Jan]; Available from: https://www.weforum.org/agenda/2017/02/cities-must-tirelessly-innovative-to-respond-to-their-challenges/.

5. Albino, V., U. Berardi, and R. Dangelico, *smart city*. 2015.

6. Stephan, A., A. Trundle, D. Kendal, H. Henderson, H. Kamalipour, and M. Lowe. *Our cities need to go on a resource diet*. 2016 30 November 2016 [cited 2019 10 Jan]; Available from: https://theconversation.com/our-cities-need-to-go-on-a-resource-diet-68984.

7. Choudhary, M. *Technology - The backbone of a smart city*. 2018 22 May 2018 [cited 2019 10 Jan]; Available from: https://www.geospatialworld.net/article/technology-the-backbone-of-a-smart-city/.

8. Greco, I. and A. Cresta, *A Smart Planning for Smart City: The Concept of Smart City as an Opportunity to Re-think the Planning Models of the Contemporary City*. 2015. 563-576.

9. ViennaUniversityOfTechnology, *europeansmartcities 4.0 (2015)*. 2015.

10. DeWit, A., *Japan's Smart Cities*. 2018, School of Economic Policy Studies: Intelligent Cities / Smart Cities. p. 35.

11. Seymour, T. *A2M2 connected corridor offers real-time, personalised travel information*. 17 September 2018 [cited 2019 24 Jan]; Available from: https://www.fleetnews.co.uk/news/fleet-industry-news/2018/09/17/a2m2-connected-corridor-offers-real-time-personalised-travel-information.

12. Staff, C.A., *Singapore to spend US$1 billion in smart city initiative during 2019*. 2019, CIO: www.cio.com.

13. Toh, M. and L. Erasmus, *Alibaba's 'City Brain' is slashing congestion in its hometown*. 2019, CNN Business: www.cnn.com.

14. U.S. DOT, I.T.S.J.P.O., *Smart City Challenge Lessons Learned*. n.d.: www.transportation.gov.

15. Sheehan, R., *Connected Corridors*, J.P.O. Intelligent Transportation Systems, USDOT, Editor. 2015: https://www.its.dot.gov.

16. MnDOT, *MnDOT Connected Corridor Project Overview*. www.dot.state.mn.us.

17. U.S. DOT, I.T.S.J.P.O. *Connected Vehicle Pilot Deployment Program*. n.d. [cited 2019 25 January]; Available from: https://www.its.dot.gov/pilots/index.htm.

18. UCBerkeley. *Connected Corridors Program*. [cited 2019 25 Jan]; Available from: https://connected-corridors.berkeley.edu/.

19. VirginiaTechTransportationInstitute. *Virginia Connected Corridors*. [cited 2019 21 Jan 2019]; Available from: https://www.vtti.vt.edu/vcc/map.html.

20. Saroj, A., S. Roy, R. Fujimoto, A. Guin, and M. Hunter. *Smart City Real-Time Data-Driven Transportation Simulation* in *Winter Simulation Conference*. 2018. Gothenburg, Sweden.

21. U.S.E.P.A, *U.S. Transportation Sector Greenhouse Gas Emissions 1990-2016*. 2018. p. 5.

22. Greco, H., Cresta A., *A SMART PLANNING for SMART CITY: the concept of smart city as an opportunity to re-think the planning models of the contemporary city*, in *International Conference on Computational Science and Its Applications*. 2015.

23. Williams, H. *What is a smart city? How to define a smart city?* 2018 10 September 2018 [cited 2019 29 July]; Available from: https://www.computerworld.com.au/article/646458/what-smart-city-how-define-smart-city/.

24. Glasmeier A, C.S., *Thinking about Smart Cities.* Cambridge Journal of Regions, Economy and Society, 2015. **8**(1).

25. PennDOT. *Smart Corridor Initiatives*. Transform76.com 2018 2018 [cited 2019 July 29]; Available from: http://transform76.com/smart-corridor-initiatives/.

26. TDOT. *Interstate 24 Smart Corridor: Davidson and Rutherford Counties*. n.d. [cited 2019 29 July]; Available from: https://www.tn.gov/tdot/projects/region-3/i-24-smart-corridor.html.

27. Seymour, T. *A2M2 connected corridor offers real-time, personalised travel information*. Fleet Industry News 2018 17 September 2018 [cited 2019 29 July];

Available from: https://www.fleetnews.co.uk/news/fleet-industry-news/2018/09/17/a2m2-connected-corridor-offers-real-time-personalised-travel-information.

28. MnDOT, *MnDOT Connected Corridor Project Overview*. n.d., MnDOT www.dot.state.mn.us.

29. Caltrans. *Connected Corridors*. [cited 2019 29 July]; Available from: https://dot.ca.gov/programs/traffic-operations/connected-corridors/overview.

30. CityOfAtlanta. *NORTH AVENUE SMART CORRIDOR*. 2017 [cited 2019 29 July]; Available from: https://renewatlantabond.com/project/north-avenue-smart-corridor/.

31. NYCDOT. *NYC Connected Vehicle Project*. 2019 [cited 2019 29 July]; Available from: https://www.cvp.nyc/.

32. UCBerkeley. *Connected Corridors Program*. n.d. [cited 2019 29 July]; Available from: https://connected-corridors.berkeley.edu/.

33. VTTI. *Connected Vehicle Environment*. n.d. [cited 2019 29 July]; Available from: https://www.vtti.vt.edu/vcc/communications.html.

34. WYDOT. *WYOMING DOT Connected Vehicle Pilot*. 2017 [cited 2019 29 July]; Available from: https://wydotcvp.wyoroad.info/.

35. USDOT. *Connected Vehicle Pilot Deployment Program*. Connected Vehicles n.d. [cited 2019 29 July]; Available from: https://www.its.dot.gov/pilots/index.htm.

36. Mohd Zulkefli, M.A., P. Mukherjee, Y. Shao, and Z. Sun, *Evaluating Connected Vehicles and Their Applications.* Mechanical Engineering Magazine Select Articles, 2016. **138**(12): p. S12-S17.

37. Goodall, N., *Real-Time Prediction of Vehicle Locations in a Connected Vehicle Environment*. 2013, Virgina Center for Transportation Innovation and Research. p. 55.

38. Doecke, S., A. Grant, and R.W.G. Anderson, *The Real-World Safety Potential of Connected Vehicle Technology*. Traffic Injury Prevention, 2015. **16**(sup1): p. S31-S35.

39. U.S. DOT, F.H.A. *New Approaches for Testing Connected Highway and Vehicle Systems*. 2016 [cited 2019 6 March]; Available from: https://highways.dot.gov/new-approaches-testing-connected-highway-and-vehicle-systems.

40. U.S. DOT, F.H.A. *In-the-Loop Simulations Provide Improved Methods for Testing of Connected Vehicle Technologies*. 2017 July 2017 [cited 2019 6 March];

Available from: https://www.fhwa.dot.gov/publications/research/ear/17044/index.cfm.

41. Chowdhury, M., M. Rahman, A. Rayamajhi, S.M. Khan, M. Islam, Z. Khan, and J. Martin, *Lessons Learned from the Real-World Deployment of a Connected Vehicle Testbed.* Transportation Research Record, 2018. **2672**(22): p. 10-23.

42. U.S. DOT, I.T.S.J.P.O., *AERIS—Applications for the Environment: Real-Time Information Synthesis, Eco-Signal Operations Modeling Report*. 2014. p. 428.

43. Brügmann, J., M. Schreckenberg, and W. Luther. *Real-Time Traffic Information System Using Microscopic Traffic Simulation*. in *2013 8th EUROSIM Congress on Modelling and Simulation*. 2013.

44. U.S. DOT, I.T.S.J.P.O., *The Smart/Connected City and Its Implications for Connected Transportation*. 2014, John A. Volpe National Transportation Systems Center, U.S. DOT, Intelligent Transportation Systems Joint Program Office.

45. Pop, M.-D. and O. Proștean, *A Comparison Between Smart City Approaches in Road Traffic Management.* Procedia - Social and Behavioral Sciences, 2018. **238**: p. 29-36.

46. Allström, A., J. Barcelo, J. Ekström, E. Grumert, D. Gundlegård, and C. Rydergren, *Traffic Management for Smart Cities*. 2017. 211-240.

47. Chen, S. and L. Du, *Simulation study of the impact of local real-time traffic information provision strategy in connected vehicle systems.* International Journal of Transportation Science and Technology, 2017. **6**(4): p. 229-239.

48. Xiaowen, D., M.A. Ferman, and R.P. Roesser. *A simulation evaluation of a real-time traffic information system using probe vehicles*. in *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*. 2003.

49. "SMARTEST". 2000, ITS, University of Leeds (GB).

50. Pell, A., A. Meingast, and O. Schauer, *Trends in Real-time Traffic Simulation*. Vol. 25. 2017. 1477-1484.

51. Henclewood, D., A. Guin, R. Guensler, M. Hunter, and R. Fujimoto. *Real-time data driven arterial simulation for performance measures estimation*. in *Proceedings of the 2010 Winter Simulation Conference*. 2010.

52. Maroto, J., E. Delso, J. Felez, and J.M. Cabanellas, *Real-Time Traffic Simulation With a Microscopic Model.* IEEE Transactions on Intelligent Transportation Systems, 2006. **7**(4): p. 513-527.

53. Wismans, L., E. de Romph, K. Friso, and K. Zantema, *Real Time Traffic Models, Decision Support for Traffic Management.* Procedia Environmental Sciences, 2014. **22**: p. 220-235.

54. Sturari, M., L. Catani, A. Mancini, and E. Frontoni. *An integrated mobility system using real-time data for traffic simulation*. in *2016 12th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA)*. 2016.

55. A Chronopoulos, A.T. and C.M. Johnston. *A real-time traffic simulation system*. 1998.

56. FHWA, U. *Simulating Connected Vehicle Technologies in Virtual Traffic Environments*. 2019 15 March 2019 [cited 2019 29 July]; Available from: https://highways.dot.gov/research-programs/exploratory-advanced-research/simulating-connected-vehicle-technologies-virtual.

57. Amini, S., I. Gerostathopoulos, and C. Prehofer. *Big data analytics architecture for real-time traffic control*. in *2017 5th IEEE International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. 2017.

58. Lv, Y., Y. Duan, W. Kang, Z. Li, and F.Y. Wang, *Traffic Flow Prediction With Big Data: A Deep Learning Approach.* IEEE Transactions on Intelligent Transportation Systems, 2015. **16**(2): p. 865-873.

59. Zhong, M., S. Sharma, and Z. Liu, *Assessing Robustness of Imputation Models Based on Data from Different Jurisdictions: Examples of Alberta and Saskatchewan, Canada.* Transportation Research Record, 2005. **1917**(1): p. 116-126.

60. Roh, H.-J., Canadá, and S. Sharma. *IMPUTATION OF MISSING CLASSIFIED TRAFFIC DATA DURING WINTER SEASON*. 2016.

61. Zhong, M. and S. Sharma, *Matching Hourly, Daily, and Monthly Traffic Patterns to Estimate Missing Volume Data.* Transportation Research Record, 2006. **1957**(1): p. 32-42.

62. Lee, S. and D.B. Fambro, *Application of Subset Autoregressive Integrated Moving Average Model for Short-Term Freeway Traffic Volume Forecasting.* Transportation Research Record, 1999. **1678**(1): p. 179-188.

63. Williams, B.M., *Multivariate Vehicular Traffic Flow Prediction: Evaluation of ARIMAX Modeling.* Transportation Research Record, 2001. **1776**(1): p. 194-200.

64. Spławińska, M., *The Problem Of Imputation Of The Missing Data From The Continuous Counts Of Road Traffic.* Archives of Civil Engineering, 2015. **61**.

65. Ghosh, B., B. Basu, and M. O'Mahony, *Time series modelling for forecasting vehicular traffic flow in Dublin*. 2005.

66. Pell, A., A. Meingast, and O. Schauer, *Trends in Real-time Traffic Simulation.* Transportation Research Procedia, 2017. **25**: p. 1477-1484.

67. Ding, Q., X. Wang, X. Zhang, and Z. Sun, *Forecasting traffic volume with space-time ARIMA model.* Advanced Materials Research, 2010. **156-157**: p. 979-983.

68. Tak, S., S. Woo, and H. Yeo, *Data-Driven Imputation Method for Traffic Data in Sectional Units of Road Links.* IEEE Transactions on Intelligent Transportation Systems, 2016. **17**(6): p. 1762-1771.

69. Sun, B., L. Ma, W. Cheng, W. Wen, P. Goswami, and G. Bai. *An improved k-nearest neighbours method for traffic time series imputation*. in *2017 Chinese Automation Congress (CAC)*. 2017.

70. Oehmcke, S., O. Zielinski, and O. Kramer. *kNN ensembles with penalized DTW for multivariate time series imputation*. in *2016 International Joint Conference on Neural Networks (IJCNN)*. 2016.

71. Zhuang, Y., R. Ke, and Y. Wang, *Innovative method for traffic data imputation based on convolutional neural network.* IET Intelligent Transport Systems, 2019. **13**(4): p. 605-613.

72. Olah, C., *Understanding LSTM Networks*, in *Colah's blog*, C. Olah, Editor. 2015: colah.github.io.

73. Saroj, A., S. Roy, A. Guin, M. Hunter, and R.M. Fujimoto, *SMART CITY REAL-TIME DATA-DRIVEN TRANSPORTATION SIMULATION*. 2018 Winter Simulation Conference (WSC), 2018: p. 857-868.

74. PTV Group, *PTV Vissim 9 Introduction to the COM API*. 2016, PTV Planung Transport Verkehr AG: Karlsruhe, Germany. p. 40.

75. Hunter, M., R. Guensler, A. Guin, A. Saroj, and S. Roy, *SMART CITIES ATLANTA - NORTH AVENUE* in *City of Atlanta Research Project*. 2019. p. 82.

76. Guensler, R., H. Liu, Y. Xu, A. Akanser, D. Kim, M.P. Hunter, and M.O. Rodgers, *Energy Consumption and Emissions Modeling of Individual Vehicles.* Transportation Research Record, 2017. **2627**(1): p. 93-102.

77. Saroj, A., S. Roy, A. Guin, M. Hunter, and R. Fujimoto, *Smart city real-time data-driven transportation simulation*, in *Proceedings of the 2018 Winter Simulation Conference*. 2018, IEEE Press: Gothenburg, Sweden. p. 857–868.

78. Hotelling, H., *Analysis of a complex of statistical variables into principal components.* Journal of Educational Psychology, 1933. **24**(6): p. 417-441.

79. Bishop, C.M., *Pattern Recognition and Machine Learning (Information Science and Statistics)*. 2006: Springer-Verlag.

80. USDOT. *Connected Vehicle Pilot Deployment Program*. n.d. [cited 2020 February 26]; Available from: https://www.its.dot.gov/pilots/.

81. Frost, A. *Singapore to develop first 5G C-V2X research testbed on NTU campus*. 2019 October 22, 2019 [cited 2020 February 26]; Available from: https://www.traffictechnologytoday.com/news/connected-vehicles-infrastructure/singapore-to-develop-first-5g-c-v2x-research-testbed-on-ntu-campus.html.

82. CenterForTransportationResearch. *City of Austin Connected Corridors*. 2018 [cited 2020 February 26]; Available from: https://ctr.utexas.edu/nmc/research-2/projects/current-and-ongoing-projects/city-of-austin-connected-corridors/.

83. HighwaysEngland, *Signs of the future: new technology testbed on the A2 and M2 in Kent*. 2018, Highways England: https://www.gov.uk/government/news.

84. *California CV Testbed*. n.d. [cited 2020 February 26]; Available from: http://www.caconnectedvehicletestbed.org/index.php/.

85. Miller, P., *Atlanta North Avenue Smart Corridorearns national recognition award*, in *AJC*. 2018, https://www.ajc.com/: Atlanta.

86. Serrà, J. and J.L. Arcos, *An empirical evaluation of similarity measures for time series classification.* Knowledge-Based Systems, 2014. **67**: p. 305-314.

87. Keogh, E., *Machine Learning in Time Series Databases (and Everything Is a Time Series!)*. 2011, Univeristy of California Riverside. p. 31.

88. Cassisi, C., P. Montalto, M. Aliotta, A. Cannata, and , and A. Pulvirenti, *Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining*, in *Advances in Data Mining Knowledge Discovery and Applications*. IntechOpen.

89. Keogh, E.J. and S. Kasetty, *On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration.* Data Min. Knowl. Discov., 2003. **7**(4): p. 349-371.

90. Han, J. and M. Kamber, *Data mining : concepts and techniques*. 3 edition ed. 2011.

91. Wang, X., A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, *Experimental comparison of representation methods and distance measures for time series data.* Data Mining and Knowledge Discovery, 2013. **26**(2): p. 275-309.

92. Roelofsen, P., *Time series clustering*, in *Business Analytics*. 2018, Vrije Universiteit in Amsterdam. p. 83.

93. Faloutsos, C., M. Ranganathan, and Y. Manolopoulos, *Fast Subsequence Matching in Time-Series Databases.* ACM SIGMOD Record, 2000. **23**.

94. Kianimajd, A., M.G. Ruano, P. Carvalho, J. Henriques, T. Rocha, S. Paredes, and A.E. Ruano, *Comparison of different methods of measuring similarity in physiologic time series.* IFAC-PapersOnLine, 2017. **50**(1): p. 11005-11010.

95. Iglesias, F. and W. Kastner, *Analysis of Similarity Measures in Times Series Clustering for the Discovery of Building Energy Patterns.* Energies, 2013. **6**(2).

96. Shasha, D. and Y. Zhu, *High Performance Discovery In Time Series: Techniques And Case Studies (Monographs in Computer Science)*. 2004: SpringerVerlag.

97. Fujita, A., J. Sato, M. Demasi, M. Sogayar, C. Ferreira, and S. Miyano, *Comparing Pearson, Spearman and Hoeffding's D measure for gene expression association analysis.* Journal of bioinformatics and computational biology, 2009. **7**: p. 663-84.

98. Xu, W., Y. Hou, Y.S. Hung, and Y. Zou, *A comparative analysis of Spearman's rho and Kendall's tau in normal and contaminated normal models.* Signal Process., 2013. **93**(1): p. 261–276.

99. Berndt, D.J. and J. Clifford. *Using Dynamic Time Warping to Find Patterns in Time Series*. in *KDD Workshop*. 1994.

100. Sakoe, H. and S. Chiba, *Dynamic programming algorithm optimization for spoken word recognition.* IEEE TRANSACTIONS ON ACOUSTICS, SPEECH, AND SIGNAL PROCESSING, 1978. **ASSP-26**(1): p. 43-49.

101. Bartolini, I., P. Ciaccia, and M. Patella, *WARP: accurate retrieval of shapes using phase of Fourier descriptors and time warping distance.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2005. **27**(1): p. 142-147.

102. Kholmatov, A. and B. Yanikoglu, *Identity authentication using improved online signature verification method.* Pattern Recognition Letters, 2005. **26**(15): p. 2400-2408.

103. Rodríguez, J.J. and C.J. Alonso, *Interval and dynamic time warping-based decision trees*, in *Proceedings of the 2004 ACM symposium on Applied computing*. 2004, Association for Computing Machinery: Nicosia, Cyprus. p. 548–552.

104. Mueen, A. and E. Keogh. *Extracting Optimal Performance from Dynamic Time Warping*. in *22nd ACM SIGKDD COnference on Knowledge Discovery and Data Mining*. 2016. San Francisco, California.

105. Tolpygo, A., *Dynamic Time Warping: Time Series Analysis II*, in *A Modern Approach To Time-Seriees Analysis*. 2016: sflscientific.com.

106. Xi, X., E. Keogh, C. Shelton, L. Wei, and C.A. Ratanamahatana, *Fast time series classification using numerosity reduction*, in *Proceedings of the 23rd international conference on Machine learning*. 2006, Association for Computing Machinery: Pittsburgh, Pennsylvania, USA. p. 1033–1040.

107. Chu, S., E. Keogh, D. Hart, and M. Pazzani, *Iterative Deepening Dynamic Time Warping for Time Series*. 2002.

108. Itakura, F., *Minimum prediction residual principle applied to speech recognition.* IEEE Transactions on Acoustics, Speech, and Signal Processing, 1975. **23**(1): p. 67-72.

109. Keogh, E. and C.A. Ratanamahatana, *Exact indexing of dynamic time warping.* Knowledge and Information Systems, 2005. **7**(3): p. 358-386.

110. Keogh, E.J. and M.J. Pazzani, *Scaling up dynamic time warping for datamining applications*, in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. 2000, Association for Computing Machinery: Boston, Massachusetts, USA. p. 285–289.

111. Sang-Wook, K., P. Sanghyun, and W.W. Chu. *An index-based approach for similarity search supporting time warping in large sequence databases*. in *Proceedings 17th International Conference on Data Engineering*. 2001.

112. Salvador, S. and P. Chan, *Toward accurate dynamic time warping in linear time and space.* Intell. Data Anal., 2007. **11**(5): p. 561–580.

113. Chandrasekaran, G., T. Vu, A. Varshavsky, M. Gruteser, R.P. Martin, J. Yang, and Y. Chen. *Tracking vehicular speed variations by warping mobile phone signal strengths*. in *2011 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 2011.

114. Hi-ri-o-tappa, K., C. Likitkhajorn, A. Poolsawat, and S. Thajchayapong. *Traffic incident detection system using series of point detectors*. in *2012 15th International IEEE Conference on Intelligent Transportation Systems*. 2012.

115. Mak, C.L. and H.S.L. Fan, *Development of Dual-Station Automated Expressway Incident Detection Algorithms.* IEEE Transactions on Intelligent Transportation Systems, 2007. **8**(3): p. 480-490.

116. Hiri-O-Tappa, K., S. Pan-Ngum, S. Narupiti, and W. Pattara-Atikom. *A Novel Approach of Dynamic Time Warping for Short-Term Traffic Congestion Prediction*. in *Transportation Research Board 90th Annual Meeting*. 2011. Washington DC, United States.

117. Motamed, M. and R. Machemehl, *Real Time Freeway Incident Detection*. 2014, Center for Transportation Research, University of Texas at Austin.

118. Ma, Y., M. Chowdhury, M. Jeihani, and R. Fries, *Accelerated incident detection across transportation networks using vehicle kinetics and support vector machine in cooperation with infrastructure agents.* IET Intelligent Transport Systems, 2010. **4**(4): p. 328-337.

119. Ruey Long, C., D. Srinivasan, and T. Eng Tian. *Support vector machine models for freeway incident detection*. in *Proceedings of the 2003 IEEE International Conference on Intelligent Transportation Systems*. 2003.

120. Xiao, J. and Y. Liu, *Traffic Incident Detection Using Multiple-Kernel Support Vector Machine.* Transportation Research Record, 2012. **2324**(1): p. 44-52.

121. Taylor, J., X. Zhou, N.M. Rouphail, and R.J. Porter, *Method for investigating intradriver heterogeneity using vehicle trajectory data: A Dynamic Time Warping approach.* Transportation Research Part B: Methodological, 2015. **73**: p. 59-80.

122. Sun, L., Q. Gong, L. Yao, W. Luo, and T. Zhang, *A Dynamic Time Warping Algorithm Based Analysis of Pedestrian Shockwaves at Bottleneck.* Journal of Advanced Transportation, 2018. **2018**: p. 1-8.

123. He, S., F. Ding, Y. Zhou, Y. Cheng, and B. Ran, *Investigating and modelling the relationship between traffic volume and extracts from cellphone activity data.* IET Intelligent Transport Systems, 2019. **13**(8): p. 1299-1308.

124. Bickel, P.J., C. Chen, J. Kwon, J. Rice, E. van Zwet, and P. Varaiya, *Measuring Traffic.* Statist. Sci., 2007. **22**(4): p. 581-597.

125. Ni, D., D. Leonard John, A. Guin, and C. Feng, *Multiple Imputation Scheme for Overcoming the Missing Values and Variability Issues in ITS Data.* Journal of Transportation Engineering, 2005. **131**(12): p. 931-938.

126. Zhao, N., Z. Li, and Y. Li. *Improving the Traffic Data Imputation Accuracy Using Temporal and Spatial Information*. in *2014 7th International Conference on Intelligent Computation Technology and Automation*. 2014.

127. Tang, J., X. Zhang, W. Yin, Y. Zou, and Y. Wang, *Missing data imputation for traffic flow based on combination of fuzzy neural network and rough set theory.* Journal of Intelligent Transportation Systems, 2020: p. 1-16.

128. Bae, B., H. Kim, H. Lim, Y. Liu, L.D. Han, and P.B. Freeze, *Missing data imputation for traffic flow speed using spatio-temporal cokriging.* Transportation Research Part C: Emerging Technologies, 2018. **88**: p. 124-139.

129. Al-Deek, H.M., C. Venkata, and S.R. Chandra, *New Algorithms for Filtering and Imputation of Real-Time and Archived Dual-Loop Detector Data in I-4 Data Warehouse.* Transportation Research Record, 2004. **1867**(1): p. 116-126.

130. Qu, L., L. Li, Y. Zhang, and J. Hu, *PPCA-based missing data imputation for traffic flow volume: a systematical approach.* Trans. Intell. Transport. Sys., 2009. **10**(3): p. 512–522.

131. Xu, J.-R., X.-Y. Li, and H.-J. Shi, *Short-term traffic flow forecasting model under missing data.* Journal of Computer Applications, 2010. **30**: p. 1117-1120.

132. Chen, C., J. Kwon, J. Rice, A. Skabardonis, and P. Varaiya, *Detecting Errors and Imputing Missing Data for Single-Loop Surveillance Systems.* Transportation Research Record, 2003. **1855**(1): p. 160-167.

133. Chen, C., Y. Wang, L. Li, J. Hu, and Z. Zhang, *The retrieval of intra-day trend and its influence on traffic prediction.* Transportation Research Part C: Emerging Technologies, 2012. **22**: p. 103-118.

134. van Lint, J.W.C., S.P. Hoogendoorn, and H.J. van Zuylen, *Accurate freeway travel time prediction with state-space neural networks under missing data.* Transportation Research Part C: Emerging Technologies, 2005. **13**(5): p. 347-369.

135. Gang, C. and G. Tongmin. *Comparison of missing data imputation methods for traffic flow*. in *Proceedings 2011 International Conference on Transportation, Mechanical, and Electrical Engineering (TMEE)*. 2011.

136. Bennett, R.J., R.P. Haining, and D.A. Griffith, *Review Article: The Problem of Missing Data on Spatial Surfaces.* Annals of the Association of American Geographers, 1984. **74**(1): p. 138-156.

137. Shang, Q., Z. Yang, S. Gao, and D. Tan, *An Imputation Method for Missing Traffic Data Based on FCM Optimized by PSO-SVR.* Journal of Advanced Transportation, 2018. **2018**: p. 1-21.

138. Li, Y., Z. Li, and L. Li, *Missing traffic data: comparison of imputation methods.* IET Intelligent Transport Systems, 2014. **8**(1): p. 51-57.

139. Zhong, M., S. Sharma, and P. Lingras, *Genetically Designed Models for Accurate Imputation of Missing Traffic Counts.* Transportation Research Record, 2004. **1879**(1): p. 71-79.

140. Castro-Neto, M., Y.-S. Jeong, M.-K. Jeong, and L.D. Han, *Online-SVR for short-term traffic flow prediction under typical and atypical traffic conditions.* Expert Systems with Applications, 2009. **36**(3, Part 2): p. 6164-6173.

141. Karlaftis, M.G. and E.I. Vlahogianni, *Statistical methods versus neural networks in transportation research: Differences, similarities and some insights.* Transportation Research Part C: Emerging Technologies, 2011. **19**(3): p. 387-399.

142. Kang, D., Y. Lv, and Y. Chen. *Short-term traffic flow prediction with LSTM recurrent neural network*. in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. 2017.

143. Alghamdi, T., K. Elgazzar, M. Bayoumi, T. Sharaf, and S. Shah, *Forecasting Traffic Congestion Using ARIMA Modeling*. 2019. 1227-1232.

144. Williams, B. and L. Hoel, *Modeling and Forecasting Vehicular Traffic Flow as a Seasonal ARIMA Process: Theoretical Basis and Empirical Results.* Journal of Transportation Engineering, 2003. **129**: p. 664-672.

145. Kamarianakis, Y. and P. Prastacos, *Forecasting Traffic Flow Conditions in an Urban Network: Comparison of Multivariate and Univariate Approaches.* Transportation Research Record, 2003. **1857**(1): p. 74-84.

146. Yin, W., P. Murray-Tuite, and H. Rakha, *Imputing Erroneous Data of Single-Station Loop Detectors for Nonincident Conditions: Comparison Between Temporal and Spatial Methods.* Journal of Intelligent Transportation Systems - J INTELL TRANSPORT SYST, 2012. **16**.

147. Liu, Z., S. Sharma, and S. Datla, *Imputation of Missing Traffic Data during Holiday Periods.* Transportation Planning and Technology, 2008. **31**(5): p. 525-544.

148. Yang, H., J. Yang, L.D. Han, X. Liu, L. Pu, S.-M. Chin, and H.-L. Hwang, *A Kriging based spatiotemporal approach for traffic volume data imputation.* PloS one, 2018. **13**(4): p. e0195957-e0195957.

149. Ni, D. and J.D. Leonard, *Markov Chain Monte Carlo Multiple Imputation Using Bayesian Networks for Incomplete Intelligent Transportation Systems Data.* Transportation Research Record, 2005. **1935**(1): p. 57-67.

150. Qu, L., Y. Zhang, J. Hu, L. Jia, and L. Li, *A BPCA based missing value imputing method for traffic flow volume data*. 2008. 985-990.

151. Chiou, J.-M., Y.-C. Zhang, W.-H. Chen, and C.-W. Chang, *A functional data approach to missing value imputation and outlier detection for traffic flow data.* Transportmetrica B: Transport Dynamics, 2014. **2**(2): p. 106-129.

152. Wang, K., C. Gou, Y. Duan, Y. Lin, X. Zheng, and F. Wang, *Generative adversarial networks: introduction and outlook.* IEEE/CAA Journal of Automatica Sinica, 2017. **4**(4): p. 588-598.

153. Yuan, Y. and F. Wang. *Towards blockchain-based intelligent transportation systems*. in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. 2016.

154. Lv, Y., Y. Duan, W. Kang, and Z. Li, *Traffic Flow Prediction With Big Data: A Deep Learning Approach.* IEEE Transactions on Intelligent Transportation Systems, 2014. **16**: p. 865-873.

155. Duan, Y., Y. Lv, and F. Wang. *Performance evaluation of the deep learning approach for traffic flow prediction at different times*. in *2016 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*. 2016.

156. Fu, R., Z. Zhang, and L. Li. *Using LSTM and GRU neural network methods for traffic flow prediction*. in *2016 31st Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. 2016.

157. Tian, Y. and L. Pan, *Predicting Short-Term Traffic Flow by Long Short-Term Memory Recurrent Neural Network*. 2015. 153-158.

158. Coleman, J. *Dynamic programming example (dynamic time warping)*. n.d. [cited 2020 19 February 2020]; Available from: http://www.phon.ox.ac.uk/jcoleman/old_SLP/Lecture_5/DTW_explanation.html.

159. PythonSoftwareFoundation. *fastdtw 0.3.4*. 2019 October 7 [cited 2020 17 Feruary]; Available from: https://pypi.org/project/fastdtw/.

160. Murtagh, F. and P. Contreras, *Methods of Hierarchical Clustering.* ArXiv, 2011. **abs/1105.0121**.

161. Shah, P. and R. Pahwa, *Hierarchical Clustering*. n.d., MIT: www.mit.edu. p. 8.

162. Ah-Pine, J. and X. Wang. *Similarity Based Hierarchical Clustering with an Application to Text Collections*. in *Advances in Intelligent Data Analysis XV*. 2016. Cham: Springer International Publishing.

163. Tibshirani, R., *Clustering 2: Hierarchical Clustering*, in *Data Mining: 36-462/36-662*. 2013: www.stat.cmu.edu/~ryantibs/datamining/.

164. Bock, T., *What is a Dendrogram?* n.d.: www.displayr.com.

165. Boehmke, B., *Hierarchical Cluster Analysis*. n.d.: https://uc-r.github.io/.

166. Kulma, K., *Determining Optimal Number Of Clusters In Your Data*. 2017: https://kkulma.github.io/.

167. Pedregosa, F., G.e. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, and others, *Scikit-learn: Machine learning in Python.* Journal of machine learning research, 2011. **12**(Oct): p. 2825--2830.

168. Coursera. *Neural Networks and Deep Learning*. n.d. [cited 2020 February 26]; Available from: https://www.coursera.org/learn/neural-networks-deep-learning/home/info.

169. Nigam, V. *Understanding Neural Networks. From neuron to RNN, CNN, and Deep Learning*. 2018 [cited 2020 February 22]; Available from: https://towardsdatascience.com/understanding-neural-networks-from-neuron-to-rnn-cnn-and-deep-learning-cd88e90e0a90.

170. Hardesty, L. *Explained: Neural networks*. 2017 April 14, 2017 [cited 2020 February 22]; Available from: http://news.mit.edu/2017/explained-neural-networks-deep-learning-0414.

171. Mayo, M. *Neural Network Foundations, Explained: Activation Function*. 2017 September 2017 [cited 2020 February 22]; Available from: https://www.kdnuggets.com/2017/09/neural-network-foundations-explained-activation-function.html.

172. DeepAI. *Activation Function* n.d. [cited 2020 February 22]; Available from: https://deepai.org/machine-learning-glossary-and-terms/activation-function.

173. Goyal, S. and A. Parashar, *Machine Learning Application to Improve COCOMO Model using Neural Networks*. Vol. 10. 2018.

174. Sharma, A. *Understanding Activation Functions in Neural Networks*. 2017 March 30, 2017 [cited 2020 February 22]; Available from: https://medium.com/the-theory-of-everything/understanding-activation-functions-in-neural-networks-9491262884e0.

175. Wang, C.-F. *The Vanishing Gradient Problem*. 2019 january 8 2019 [cited 2020 February 22]; Available from: https://towardsdatascience.com/the-vanishing-gradient-problem-69bf08b15484.

176. Nielson, M.A. *Why are deep neural networks hard to train?* 2019 December 2019 [cited 2020 February 22]; Available from: http://neuralnetworksanddeeplearning.com/chap5.html.

177. Coursera. *Sequence Models*. n.d. [cited 2020 May 20 2020]; Available from: https://www.coursera.org/learn/nlp-sequence-models/home/info.

178. Kostadinov, S. *How Recurrent Neural Networks work*. 2017 December 2017 [cited 2020 February 22]; Available from: https://towardsdatascience.com/learn-how-recurrent-neural-networks-work-84e975feaaf7.

179. See, A., *Vanishing Gradients and Fancy RNNs*, in *Natural Language Processing with Deep Learning*. 2019.

180. Hochreiter, S. and J. Schmidhuber, *Long Short-term Memory.* Neural computation, 1997. **9**: p. 1735-80.

181. Weber, N. *Why LSTMs Stop Your Gradients From Vanishing: A View from the Backwards Pass*. 2017 November 15 2017 [cited 2020 February 23]; Available from: https://weberna.github.io/blog/2017/11/15/LSTM-Vanishing-Gradients.html.

182. Yao, Y. and Z. Huang. *Bi-directional LSTM Recurrent Neural Network for Chinese Word Segmentation*. in *Neural Information Processing*. 2016. Cham: Springer International Publishing.

183. Brownlee, J. *How to Handle Missing Timesteps in Sequence Prediction Problems with Python*. Deep Learning for Time Series 2017 August 5 2019 [cited 2020 February 24]; Available from: https://machinelearningmastery.com/handle-missing-timesteps-sequence-prediction-problems-python/.

184. Coursera. *Sequences, Time Series and Prediction*. n.d. [cited 2020 February 26]; Available from: https://www.coursera.org/learn/tensorflow-sequences-time-series-and-prediction/home/info.

185. Rossum, V., Guido and Drake, and F. L., *Python 3 Reference Manual*. 2009, Scotts Valley, CA: CreateSpace.

186. Chollet, F.a.o. *Keras*. 2015 [cited 2020 May 20]; Available from: https://github.com/fchollet/keras.

187. Abadi, M., A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, and a. others, *TensorFlow: Large-scale machine learning on heterogeneous systems*. 2015.

188. Bengio, Y. *Practical recommendations for gradient-based training of deep architectures*. 2012.

189. Shier, R., *The Wilcoxon signed rank sum test*, in *Mathematics Learning Support Centre*. 2004: http://www.statstutor.ac.uk/resources/. p. 3.

190. Siami-Namini, S. and A.S. Namin, *Forecasting Economics and Financial Time Series: ARIMA vs. LSTM*. 2018, arXiv: https://arxiv.org/. p. 19.

191. Xingwei, L. and S. Kuniaki, *The Comparison Between ARIMA and Long Short-term Memory for Highway Traffic Flow Prediction.* Journal of the Eastern Asia Society for Transportation Studies, 2019. **13**: p. 1817-1834.

192. Muzaffar, S. and A. Afshari, *Short-Term Load Forecasts Using LSTM Networks.* Energy Procedia, 2019. **158**: p. 2922-2927.