2-13-2020

Simulating Large-Scale Microscopic Traffic Data

Vrinda Khirwadkar

SIMULATING LARGE-SCALE MICROSCOPIC TRAFFIC DATA

By

Vrinda Khirwadkar

A Thesis

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Major: Computer Engineering

The University of Memphis

December 2019

Dedicated to my parents, husband, teachers, family, and friends who are all responsible for the little knowledge and wisdom I have.

Acknowledgment

My heartfelt gratitude to my thesis advisor, Dr. Bonny Banerjee, Principle Investigator, Computational Intelligence Lab, the University of Memphis, whose vision and constant support has helped me to proceed in the right direction, both academic and non-academic. His unfailing support and assistance have helped me understand a new domain of research. He has always monitored my progress periodically, which has helped me in meeting my deadlines. He has always encouraged and appreciated my work which gives a sense of satisfaction and motivation to work more towards the research.

I want to express my gratitude to Dr. Sabyasachee Mishra of the Department of Civil Engineering. It was a great experience to work with him, his tremendous knowledge in the domain of transportation helped me to make progress with the research. I want to thank Dr. Madhusudhanan Balasubramanian of the Department of Electrical and Computer Engineering for accepting to be a part of my Thesis Committee and for his valuable suggestions.

I would also like to add appreciation to my friends and colleagues at the CIL lab, who has made the work and study environment so lively, with constant encouragement and help throughout my stay. I would never forget the unconditional support I received from my parents, my in-laws and my supportive husband in the nitty-gritty, and for the unfailing encouragement, they gave me in the moments of difficulty. I would also like to thank my family, friends, and teachers who all backed me at the hour of need and without whom it was not possible to achieve where I am.

Simulating large-scale microscopic traffic data

Abstract

Traffic situations are continuous, uncertain, highly dynamic and partially observable, and they affect the day-to-day lives of people in a society. A worthwhile endeavor is to develop algorithms that can predict abnormal traffic situations by exploiting data from the myriad of sensors on the streets, in vehicles and in smartphones, leading to smoother flow of traffic. Unfortunately, the large volumes of microscopic (i.e. individual vehicle-level) data required for developing statistical/machine learning algorithms cannot be collected from the field by the public. The data collected by transportation agencies is either macroscopic or not widely available.

In this thesis, a framework is developed for simulating large-scale traffic data using a microscopic simulation model and limited real-world data. Five kinds of sensors are simulated: inductor loop detector, lane area detector, multi-entry multi-exit detector, Bluetooth, and edge-based traffic measure. Data is simulated using this framework from multiple sensors over an area covering Montgomery County and Prince George County in Washington DC for 720 hours (30 days). The synthesized data is validated with respect to real-world data for volume and speed. Widely-used classifiers are used to recognize eight traffic events, namely Collision, Disabled Vehicle, Emergency Roadwork, Injuries Involved, Obstructions, Road Maintenance Operations, Traffic Signal Not Working and with no events in the synthesized dataset with high accuracy. Given limited real-world microscopic traffic data from a particular area, this framework is the first of its kind that can simulate data from multiple kinds of sensors over a very long duration with high-fidelity to the given data.

Table of Contents

List of Figures

List of Tables

Introduction

The purpose of traffic model development, calibration and validation have raised the issue of data requirements. The flexibility and scalability of interactive simulations on road networks are of increasing interest, and it helps to demonstrate the applications of complex traffic scenarios. Road networks in urban environments can be complicated and extensive, and traffic flows on these roads can be enormous, making it a daunting task to model, simulate, and visualize at interactive rates. This thesis introduces a hybrid simulation technique that combines the strength of real-world data and synthetic data generated by the Simulation. There are conventional technologies to measure traffic data. The development of Intelligent Transportation Systems (ITS) requires high quality traffic information in real-time. For improving traffic management, collecting traffic data methods have been evolving considerably.

The use of traditional on-road sensors like Inductive loops, Lane area detectors for collecting data is necessary but not sufficient because of their limited coverage and expensive costs of implementation and maintenance. In the last few years, we have been noticing the emergence of various data sources. The effective method of data collection based on the vehicle location is Floating Car Data. This solution copes with some limitations from the fixed detectors. This method would not only improve traffic management but also helps the drivers to have access to the relevant real-time information.

This requires traffic data to be accurate, reliable and as complete as possible.

1.1 Why simulate traffic data?

The increasing traffic levels and developing transportation infrastructure has prompted research towards Intelligent Transportation Systems (ITS). The lack of widely available data from public sources from various modalities and the need for coordinated and adaptive road systems has expanded the capabilities of Simulation systems to model such challenges. Current road infrastructure needs to be used to its maximum capacity before making changes. This can be done only by thoroughly understanding the traffic mobility of an urban city. The best method is to build a monitoring infrastructure that will track the versatility of the entire vehicle population and then analyze that data. This can either be done by monitoring individuals, or by monitoring the movement of vehicles through cameras/sensors on roads. These methods have many roadblocks. Tracking an individual's progress has high privacy and security risks associated with it, and hence most citizens would not opt-in to this program. Furthermore, building the infrastructure to monitor the movement will be very expensive. This is where the synthetic generation of traffic data is useful.

The broad objective of traffic management systems is to improve the safety, efficiency, capacity and system reliability. Building an infrastructure to generate traffic data is very expensive and is not an option. Several software applications support these initiatives. Methodologies used for creating traffic simulations need examination in the context of real-world big traffic data. Such systems simulate road network performance at various levels of detail, estimate and predict real-time conditions and generate extensive scale data for microscopic analysis [1, 2, 3, 4]. This data can be used to create models for predicting the state of traffic flow, vehicle arrivals and driver behavior, and traffic flow. Simulation models also have a significant contribution to modeling the system for self-driving cars.

Our work focuses mainly on generating models for these concepts and using them to drive microscopic traffic simulations built upon real-world data. The large-scale deployment of traffic surveillance technologies has motivated advanced traffic optimization software. There is an increase in the installation of sophisticated sensor networks that widely collect time-varying traffic data. These sensors vary in their operating principles resulting in substantial potential data. Traffic management applications may exploit the strengths associated with each sensor. The idea of this research is to classify the various traffic patterns/events with each sensor. The classification based on the type of data each sensor collects. The focus is not only limited to the technologies and simulation systems used but also on the use of widely collected data from Sensors for traffic management purposes.

1.2 Literature Review

The emergence of traffic system optimizations has a significant role in the development of sensor technologies and traffic simulation tools. In the literature, vast research has done on generating real-world traffic data based on the two methods.

Urban traffic congestion and control is a big challenge in Intelligent Transportation Systems (ITS). It also has a significant impact on society, and it increases the travel times for citizens. Additionally, it affects various environmental factors as it is directly associated with air pollution and on economic factors like fuel consumption. It has proved that a significant rate of congestions produced due to unusual events taking place on the roads. These events interrupt the smooth traffic flow. Examples of such incidents include congestion caused by disabled vehicles on the road, emergency road work construction, and some obstruction or car accidents. The discomfort and financial cost caused by incidents are so significant that urge the need for traffic flow detection and immediate report of events.

In the literature, there have been many efforts at creating models and simulations of traffic systems for macroscopic and microscopic methods paradigms. Generally, macroscopic models assume a continuous flow of traffic from nodes and do not consider individual vehicle behavior. This approach has a lesser amount of details than other methods in the results of the simulation.

A traffic model was created using applied dynamic network loading by Tamp`ere et al. [5]. They used the simple merge and diverged models to represent the different types of connections roads can make inside a traffic system. The overall goal was to optimize the network flow over the entire network system. These models consider a network of nodes and edges, with vehicle volumes equating to network flow. Their focus was mainly on deriving constraints and generic requirements that such models must fulfill. This work was extended by Fl¨otter¨od et al. [6] to build a more robust model for representing traffic flow. In the first step, their model based on an incremental node model for road intersections.

A few of the limitations of this model are that it was not able to capture situations where the increase of one flow decreases another flow. Their new model augmented with the capability to describe such conditions. Car-following models have been used widely to model traffic flow and the behavior of vehicles for a long time. Gipps (1989) proposed a model that computes velocities and accelerations based on the differences between successive vehicle locations. He updated his models for his work on the MULTSIM traffic simulation system [7]. Other car-following models include the Optimal Velocity Model [8], the Generalized Force Model [9] and the Intelligent Driver Model [10]. Somewhat recently Li et al. [11] formulated a car-following model based on the headways, velocities, and accelerations of multiple preceding vehicles.

Microscopic simulation models provide a much more considerable amount of detail than macroscopic models since individual cars and their behavior are represented with much more sophisticated algorithms to control their movement and decisions. The obvious trade-off is that this requires a much higher computational cost, as simulations will usually contain hundreds or even thousands of vehicles in the system at the same time. The open-source traffic simulation platform SUMO-Simulation of Urban Mobility introduced in 2011 by Behrisch et al. [12], which has been used by many researchers to validate their models, and to optimize characteristics of traffic systems. Another microscopic simulation platform is VISSIM, which is time step based and was used by researchers at the Georgia Institute of Technology by Hunter et al. [13] to create traffic simulations based on real-world data. They generate vehicles using a Poisson counting process to produce random inter-arrival times. Many simulation systems, including SUMO and VISSIM, are capable of using Open Street Maps to generate road networks. This feature makes modeling real-world traffic systems much more comfortable, and it lends more credibility to the traffic simulations themselves. Traffic flow modeling and forecasting is an essential paradigm of transportation research. Lippi et al. [14] used time series analysis and support vector regression to forecast traffic flows for short-term periods. There has been an effort to optimize traffic light timings. Ezzat et al. [15] used the third party simulation software ExtendSim to create, execute, and optimize their traffic models. The software uses an evolutionary approach to optimization. They based their system performance on both queue lengths and vehicle waiting times. Osorio and Chong [16] used metamodels to optimize simulations of transportation systems. Their metamodel is based on a system of linear and nonlinear equations, which they test for suitability in reducing traffic congestion in a large-scale traffic system.

Institute of Transportation Systems at the German Aerospace Center created the TAPASCologne dataset and used by Uppoor S. et al. [17] in their research. The dataset describes the car traffic movement in the greater urban area of Koln, Germany. They presented the model based on vehicular mobility trace. Some of the features include compassing vast regions, focusing on the integrity of the road traffic, high time granularity, and realistic representation of microscopic behavior and also from a macroscopic point of view. Xia F. et al. [18] use a dataset available on the Shanghai Open Data Innovation Application Contest platform provided by Shanghai Qiang

Sheng Holding Co., Ltd. in their work. Their dataset describes detailed routes of 13750 taxis in the Shanghai region for one month. The data is highly accurate as it came from a sound source. They focus on small details such as the status of the taxi (occupied/empty), pick up time etc. which makes the data highly valuable with high dimensionality. The vehicular mobility trace described previously, is seen to hold good for the Shanghai dataset as well. Luca Bedogni et al. [19] introduced an original Bologna Ringway dataset. Their dataset describes the movement of more than 22000 vehicles in a 25-km area that covers the center and outskirts of Bologna, Italy. To generate the Bologna Ringway dataset, they employed an original version of the OSM-to-SUMO road network conversion tool.

The previous research on data generation has utilized some real world information to simulate synthetic data. One such practical world travel time information is provided by INRIX [20]. INRIX is a private party that collects information about roadway conditions. It accomplishes this mission with its smart drive network that aggregates nearly 400 sources of data. Sources of data with regards to flow and traffic incidents include: road sensors, traffic cameras, commercial vehicle GPS probes, consumer vehicle GPS probes, cellular network probes, road crashes, and road construction. Once the source-aggregated traffic data is collected, it then gets processed using a proprietary data fusion engine.

1.3 Contributions

The significant contributions of this thesis are:

- A new model for traffic data simulated using real-world traffic data with information from five different sensors Aggregated mobile data, Inductor Loop, Lane area detectors, Multi-Entry Multi-Exit detectors, and Bluetooth.

- A hybrid traffic generation model is simulated using sampled travel time data from INRIX and O.D. trips, which are based on six different purposes of passenger trips as well as on their income levels.
- Machine learning models are studied and developed which can differentiate between normal and abnormal events. Abnormal events include Collision, Disabled Vehicle, Emergency Roadwork, Injuries Involved, Obstruction, Road Maintenance Operation, and Traffic Signal Not Working. Department of Transportation Traffic Operations collected these anomalous traffic events data and was simulated using the Simulation Tool SUMO.
- The traffic data is collected at each traffic light of the area to get information about most of the abnormal events.
- Performance comparison carried between Decision Tree, Ensemble Boosted Tree, Ensemble Bagged Tree, KNN and Random Forest.

2. Models and Methods for Traffic Data Simulation

2.1 Sensors

1. **Aggregated Mobile GPS data**

   This data is collected based on Floating Car Data (FCD). The fundamental principle of FCD is to gather information about all vehicles throughout the entire road network using mobile phones or GPS. This accounts that every car that acts as a sensor with GPS equipped in it. As stated in [23]: FCD is an alternative or rather complement source of high-quality data to existing technologies. They will help improve the safety, efficiency, and reliability of the transportation system. They are becoming crucial in the development of new Intelligent Transportation Systems (ITS) and are involved in multiple applications worldwide dealing with real-time traffic information and traffic management. Data collected from this sensor includes vehicle location, speed, and direction of travel in the form of angle and position of the vehicle at every timestamp [24]. The FCD output analyzed as an aggregated edge-based traffic measure for this research. The number of vehicles that are present on the edge/lane in each second summed up over the measurement interval is termed as sampledSeconds [25]. The aggregation period is the time taken by the user to aggregate the data. It is taken as 60 seconds meaning it will aggregate the values from 1-60 sec, 61-120 sec and so on.

Features derived from this sensor :

Feature	Description
Begin	The first time step the values were collected in.
End	The last time step + DELTA_T in which the reported values were collected.
sampledSeconds	The number of vehicles that are present on the edge/lane in each second summed up over the measurement interval (may be subseconds if a vehicle enters/leaves the edge/lane).
Traveltime	Time needed to pass the edge/lane, note that this is just an estimation based on the mean speed, not the exact time the vehicles needed. The value is based on the time needed for the front of the vehicle to pass the edge.
Density	Vehicle density on the lane/edge
Speed	The mean speed on the edge/lane within the reported interval.

	This is an average over time and space (space-mean-speed), rather than a local average over the vehicles (time-mean-speed). Since slow vehicles spend more time on the edge they will have a proportionally bigger influence on average speed.
Departed	The number of vehicles that have been emitted onto the edge/lane within the described interval.
Arrived	The number of vehicles that have finished their route on the edge lane.
Entered	The number of vehicles that have entered the edge/lane by moving from upstream.
Left	The number of vehicles that have left the edge/lane by moving downstream.

Average number of vehicles on the edge (#) = sampledSeconds / period

Average traffic volume (#/h) = speed * 3.6 * density

Traffic volume at the begin of the lane / edge (#/h) = 3600 * entered / period

Traffic volume at the end of the lane / edge (#/h) = 3600 * left / period

Total distance travelled (m) = speed * sampledSeconds.

The total number of vehicles is the sum of the number of vehicles departed and number of vehicles entered on that edge.

2. Bluetooth

SUMO supports simulation of wireless services to facilitate radio signals emitted by the vehicle. Bluetooth is a short range and low power standard for wireless networks. Bluetooth devices are available in a number of vehicles and depict an easy way of detecting motions of persons. It is also easy to equip small devices such as smart phones to act as a detector making Bluetooth a universally accessible data source [26]. The detection parameters can be configured and the detected events can be retrieved. The functionality of Bluetooth can be enabled using two devices- Bluetooth sender and Bluetooth receiver [27]. Since every Bluetooth device is uniquely identifiable due to its MAC address, new applications regarding traffic monitoring arose during the last 10 years. Bluetooth devices are available in several vehicles (e.g. in terms of mobile devices such as smartphones and headsets as well as in-vehicle systems like satnav or car radio) and thus allows detecting motions of persons and goods [28]. The device discovery process is modelled as an exponential distribution, that is the number of detections based on Bluetooth is a sequence of independent respectively seen or not seen trials, each of which occurs with a certain probability. This follows from the assumption that the number of vehicles equipped with Bluetooth devices (10% – 15%) and the number of observer vehicles (< 3%) within the network is small, so that the chances to encounter are stochastically independent events [29]. The monitoring process can be described as a Poisson process with λ being the average amount of these stochastical incidents and t being the time on the interval [0, t]:

$$P_n(t) = e^{-\lambda t} \frac{(\lambda t^n)}{n!}$$

In our research, we have assigned 10% of the vehicles with Bluetooth and it will collect the information whenever it gets a Bluetooth vehicle within its range.

Data collected from Bluetooth includes the speed, location and Bluetooth id carried in travelling vehicles for both sender and receiver.

Features derived from this sensor:

Features	Description
id@seen	The id of the detected vehicle (sender)
tBeg	The time the sender entered the detection range.
seenPosBeg	Cartesian coordinates of the sender when it entered the range.
seenSpeedBeg	Speed of the sender when it entered the range

3. Inductor loop detector

Inductor Loop is a detection system which uses electromagnetic communication. These acts as a vehicle presence indicator. Inductor loops are placed at a point when approaching the traffic light. The functioning of these detectors is based on induction of eddy currents in the wire loops. When a vehicle comes within the radius of the loop, inductance of wire loops is decreased and it actuates the electronic unit output relay which will detect the presence of a vehicle by sending a pulse to the traffic signal controller [30]. It provides a common standard for obtaining accurate occupancy measurements. The major strength of inductor loop detector is for high frequency excitation models, it provides a good classification data . This sensor is insensitive to inclement weather such as rain, fog, and snow. SUMO provides a way to define Inductor loop using additional files. In large scale simulation, installation of inductor loops can be cumbersome. Hence, SUMO tool "generateTLSE1Detectors.py" can detect all the traffic lights and install the inductor loops at each lane where the traffic lights are located [31]. It also provides the information about the number of vehicles that are passed within the integrated interval and mean velocity of the vehicles that are collected during that interval. In our research, the data is collected at every 60 seconds. Inductor loop sensor provides basic traffic parameters e.g., volume, presence of the vehicles, occupancy, speed and gap. Features derived from this sensor :

Features	Description
Begin	The first time step the values were collected in
End	The last time step + DELTA_T the values were collected in
Id	The id of the detector
nVehContrib	The number of vehicles that have completely passed the detector within the interval
Occupancy	The percentage (0-100%) of the time a vehicle was at the detector.
Speed	The arithmetic mean of the velocities of all completely collected vehicles (-1 indicates that no vehicles were collected). This gives the time mean speed.

4. Lane area detectors

Lane area detectors are quite similar to inductor loop detectors. They cover an area instead of a cross section. In real-world scenario, it will be similar to looking at a section of road using cameras. This sensor is customized for measuring queued vehicles. They are less specific as compared to inductor loop in regard of temporal precision at the entering and leaving. Data from this sensor is helpful when finding how much time, speed and jam has to pass until a vehicle is recognized as halting [32]. The mean speed given by this detector is rather the length divided by the mean travel time, so even if all vehicles drive with constant speed the result will differ from the measurements of an inductor loop detector. Features from this sensor :

Features	Description
nVehEntered	The number of vehicles that entered the detector in the corresponding interval.
nVehLeft	The number of vehicles that left the detector in the corresponding interval.
nVehSeen	The number of vehicles that were on the detector in the corresponding interval (were "seen" by the detector).
meanSpeed	The mean velocity over all collected data samples.
meanOccupancy	The percentage (0-100%) of the detector's place that was occupied by vehicles, summed up for each time step and averaged by the interval duration.

5. Multi-Entry Multi-Exit detectors

It is basically an extension of an Inductor loop which is used to count the number of vehicles entering or exiting a closed area. This detector can be used to get the average time spent by vehicles on an edge [33]. This detector is placed for evaluation of Level of Service (LOS) at signalized intersection. LOS is a performance metric for qualitative measures used to analyze the quality of the traffic flow. The Level of Service can be evaluated on the basis of density of traffic flow at highways using volume of flow per speed [34]. This sensor is placed at the start and end of the freeways as well as at the intersections. Along with it, in order to map the simulation output with real-world traffic events data, we have also placed these detectors at every location of occurred events. Features from this sensor:

Features	Description
meanTravelTime	The time vehicles needed to pass the area (the crossing of the vehicle front counts). Averaged over all vehicles which left the detector completely during the interval duration.
meanSpeed	The mean speed of vehicles that have passed the area. Averaged over the interval and vehicles.
vehicleSum	The number of vehicles that have left the area during the interval.
meanSpeedWithin	The mean speed of those vehicles that have entered, but not yet left the area. Averaged over the time each vehicle was in the area and vehicles.
vehicleSumWithin	The number of vehicles that have entered but not yet

	left the area.

2.2 Tools

SUMO

SUMO [12] is an open-source tool for traffic simulation, mainly used for network import, demand-based modeling, and supports dynamic user assignment routing. It is a microscopic, multi-modal to support the research community to implement their algorithms. It is widely used traffic surveillance and a new approach for traffic guidance. An external interface called Traci introduced for external communication. The simulation uses car-following models for vehicle modeling, where the driver's behavior depends on the distance between the vehicle leading him. SUMO follows an extension of the stochastic car following the model developed by Krauss. It tells about each vehicle's speed, and the simulated routes for a large number of vehicles like cars, trucks, bicycles, and pedestrians. It can import many network formats like VISUM, OpenStreetMap (OSM), VISSIM, shapefiles, and XML-files. It also includes several applications that require simulation like a Net convert, Polyconvert, OD2trips, and Duarouter. It also supports command-usage as most of the application tools in SUMO can be run through the command line.

ArcGIS

ArcGIS Geostatistical Analyst is an extension for advanced surface modeling using deterministic and geostatistical methods.

ArcGIS is a geographic information system (GIS) for working with maps and geographic information. It is used for creating and using maps, compiling geographic data, analyzing mapped information, sharing and discovering geographic information, using maps and geographic information in a range of applications, and managing geographic information in a database.

Openstreetmap

OpenStreetMap (OSM) is a collaborative project to create a free editable map of the world. The creation and growth of OSM have motivated by restrictions on the use or availability of map information across much of the world, and the advent of inexpensive portable satellite navigation devices.

2.3 Data

A number of data sources are collected including -

(1) Household travel survey by The Maryland Statewide Transportation Model

(2) INRIX path travel times

(3) Traffic events data by Department of traffic Operations.

Each of these data sets explained below.

(1) Household travel survey

This survey contains four types of information, which include individual characteristics, household characteristics, trip characteristics, and vehicle characteristics. The socio-economic and demographic characteristics obtained from the person, household, and vehicle characteristics of the household travel survey. In a conventional transport modeling exercise, the study area is divided into zones that are considered the generators and attractors of trips. The modeling process usually proceeds in a sequence of four sub-models:

- The trip generation which uses zonal data to model the number of trips generated from and attracted to each zone.

- The trip distribution, which synthesizes the origin-destination matrix that is the number of trips from each zone to each zone.

- The modal split in which the choice of mode made by each traveler is simulated.

- An assignment in which the route followed by each trip is modelled. The output from this stage includes link flows and a revised measure of the costs of traveling between each pair of zones.

At the Statewide Level, there are The 1588 Statewide Model-level Zones (SMZs) that cover Maryland, Delaware, Washington DC, and parts of New Jersey, Pennsylvania, Virginia, and West Virginia (Figure 1-2). The 151 Regional Model Zones (RMZs) cover the full US, Canada, and Mexico. RMZs are used for the multi-state commodity flow model and the long-distance passenger model only and are eventually translated into flows assigned to networks and zones at the Maryland-focused (SMZ) level. Travel demand is derived from economic and demographic activities—primarily households by type and employment by industry. Socioeconomic data by SMZ were developed for the entire statewide model area with consistent categories and definitions to the extent practical given the availability of source data.

In the traffic assignment, Origin and Destination for each mode assigned to the traffic network. For most kinds of analysis, there is always a need for Origin-Destination (O.D.) matrices, which specify the travel demands between Origin and Destination nodes of the network. The volume of the traffic determined by O.D. matrices and provided to the simulation system for the generation of trips.

The following trip purposes are identified :

- HBW = Home Based Work
- HBS=Home Based Shop
- HBO=Home Based Other
- HBSCH = Home-Based School
- NHBW = Non-Home Based Work
- NHBO = Non-Home Based Other

Trip productions for work-related purposes are based on trip rates cross-classified by income and number of workers. Each passenger trip from Home is sub-categorized into five income levels. Each income level has three categories- Single Occupancy Vehicles, Heavy Occupancy Vehicles with two passengers and Heavy Occupancy Vehicles with three passengers. Similarly, Non-Home based Workers and Others are also sub-divided according to Vehicle Occupancy. Travel Demand has categorized into Regional trips, which include Commercial Vehicles, long-distance Autos, long-distance trucks, short distance multi-unit trucks, and short distance single-unit trucks. These Origin-Destination Matrices are combined to get a single matrix that has the total travel demand from one traffic assignment zone to another.

(2) INRIX travel time data

Travel time data for various paths obtained from INRIX. Traffic Message Channels (TMCs) are the spatial units of INRIX data. In this study, INRIX historical data is gathered for four months in five-minute increments, for specific paths and aggregated for every hour. The particular ways provide geographical information at each associated waypoint data with trip provider details. It also includes information about the driving class represented by the provider and vehicle weight class. INRIX is a global firm that collects data for car services and detailed transportation analysis. It is leading in the transportation industry to provide the best solution for urban mobility. The current studies of INRIX show that it has analyzed over 100,000 traffic spots in North America. INRIX does not cover all the functional classes of roadways, but it contains most of the major and minor arterials, along with a full representation of freeways, interstates, and expressways.

Data Generation

Traffic Data Simulation in SUMO involves building the network, generating traffic demand, and running the simulation. The complete framework for Data Generation Process is as shown in Fig. 1.

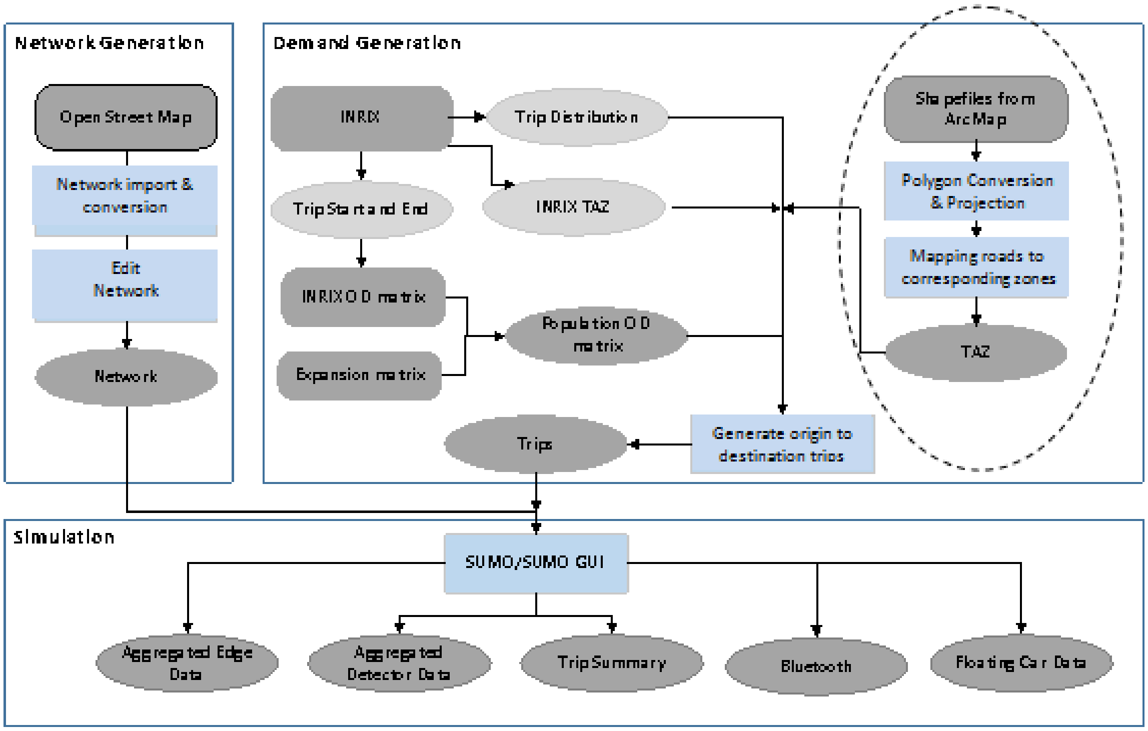


Figure. 1 Framework for Data Generation Process

Network Generation:

The road network is a system of interconnected links that are designed to carry different modes of transportation. Generally, it consists of roads, intersections (controlled/uncontrolled), roundabouts, traffic lights, junctions, pathways, etc. For microscopic simulation, simulated network should be an exact replica of the real world network, such as its geometry, lanes, ramps, number of vehicles passing through each lane, exact locations of sensors, etc. Some urban road networks also contain detectors, variable message signs, and dynamic road information panels. There are many options to generate this network which can be used in SUMO. One option is by using the OSM Web Wizard provided by SUMO. Based on the area displayed in the wizard, an entire road network can be imported, which can be loaded in SUMO. It also provides options to generate demand by giving several vehicles entering the system. The required map can also be created by importing areas from other sources like VISUM, OpenStreetMap and shapefiles and convert it using the NETCONVERT tool. The network importer NETCONVERT is an in-built application tool to make the network (from other sources) compatible with SUMO.

In this research, we are using OpenStreetMap (OSM) to generate a network topology. This is a valuable source for real-world map data. The benefit of using a map from OSM is that we can modify the map data. The network file imported from OSM needs to be converted to the SUMO network file. It is a directed graph that contains network coordinates, edges (roads), traffic light logics, intersections (junctions), including right-of-way definitions, connections, and roundabouts. This was done using NETCONVERT. It extracts the map information from the OSM file and converts it into SUMO network file. Figure 2a and 2b shows network map from OSM and SUMO.

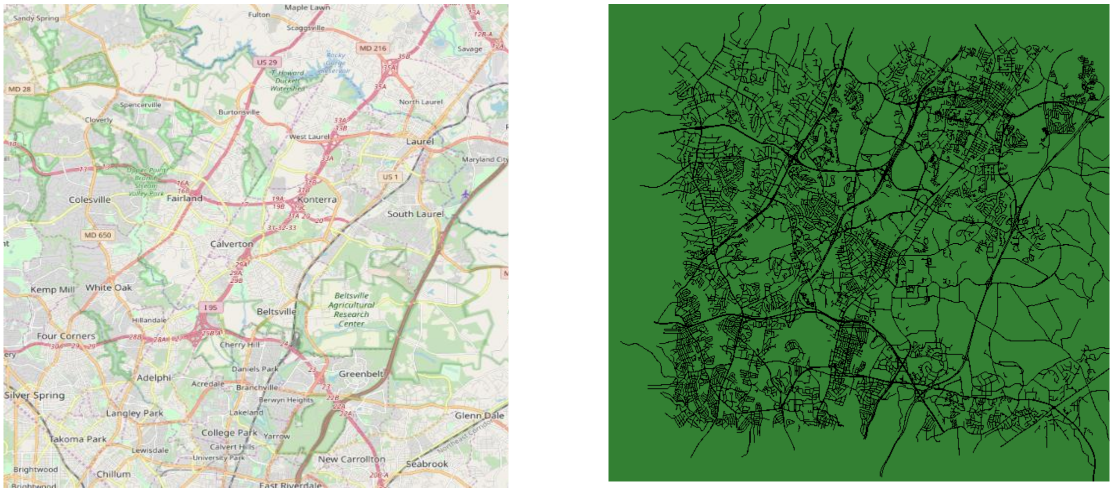


Figure. 2a) Network from OpenStreetMap

Figure. 2b) Network converted using NETCONVERT

Figure 2. Maps from OpenStreetMap and NETCONVERT

Converted map data has deviations compared with the actual data due to its open attributes and isolated links. To construct a precise road network for simulation, we have to correct the road topology so that it can match to the real world. The generated network can be edited using the graphical network editor NETEDIT provided by SUMO. It can be used to modify certain aspects of a network like broken and isolated links, loops, intersection, junctions, etc. In our functional area of Washington DC, we have removed a few edges at the periphery, removed some isolated links, and corrected junctions at some traffic lights. The reference was taken from Google Maps to alter the network. This edited network is used further to run the simulation. The network in SUMO GUI before and after using NETEDIT is shown in Figure. 3a and 3b. The road topology information is summarized in below Table 1. This map covers an area of approximately 2140 miles of road network, which includes interstate, freeways, arterials, and collectors.

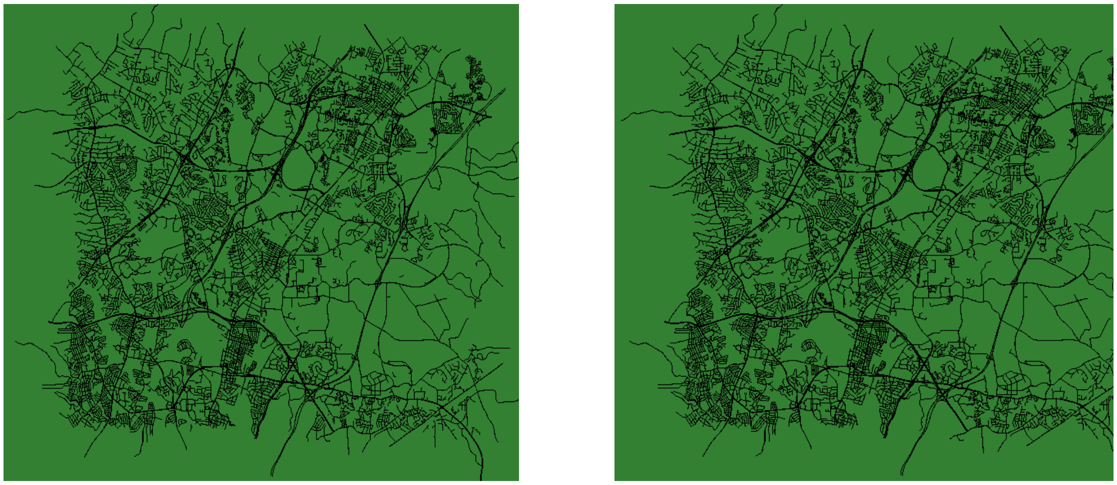


Figure.3a) Network before editing

Figure. 3b) Network after editing

Figure 3. Road Network before and after editing

Table 1. Network Information

Network Information	
Total Edges	23,498
Total Nodes	10,003
Total length Edges	2140.33 mi
Total length lanes	2659.46 mi

Demand Generation:

Microsimulation tools have become progressively crucial in traffic demand modeling. The numerous advantage over traditional assignment models lies within the fact that data is simulated/generated at the individual vehicle level. The main challenge is to create this demand for a large variety of inputs. These inputs can differ in quality, spatial resolution, and purpose. Traffic demand can be generated either by specifying individual vehicle routes, using flow definition and turning ratios, importing existing roads, and importing OD matrices. In our research, we are getting the inputs to generate Demand using shapefile. Population OD matrices and real-world INRIX data are defining Traffic Assignment Zones (TAZ) and trip distribution percentage at every half hour. In this section, we have discussed the process of generating a significant demand generation using the above three sources.

A shapefile is a simple set of related files, data storage format for storing the geometric location and attribute information of geographic features. Geographic features in a shapefile can be represented by points, lines, or polygons (areas). The idea is to use the network file and shapefile to get the polygons for our desired functional area using the inbuilt SUMO tool POLYCONVERT, which imports points of interest and polygons from different formats and translates them into a description to be visualized in SUMO-GUI. The generated polygon file is divided into numerous traffic assignment zones (TAZ) / districts in the network map. Using spatial join, we have mapped the edges in the network to these traffic assignment zones. This creates a plan between available boundaries and their corresponding traffic assignment zone. This entire process of converting shapefile to relevant zones is shown in above Figure 1 under the blue dotted circle.

The generated network needs traffic load and trips for traffic simulation. In our research, we have created trip information using the inbuilt SUMO tool OD2TRIPS. This tool primarily requires three inputs – a) OD Matrices (FMA files), b) traffic assignment zone details from INRIX and c) trip distribution percentage with respect to particular time of the day from INRIX. All three inputs are discussed in detail below.

a) OD Matrix obtained by Maryland Statewide Transportation Model is Population matrix which gives regular traffic information for all vehicle types from one zone to another. INRIX provides travel time data for each location and time stamp. As in this research, we need to generate continuous 30 days of simulated data, we would need demand (trip information) for 30 separate days. So, in order to achieve this, we have fused INRIX and population matrix data to generate different OD Matrix for separate days. This would serve our purpose of replicating real world demand for generating simulated data.

For each individual trip from INRIX data, geographical coordinates for start and end location stamps. This information is then converted to point shapefile in order to map with urban infrastructure data, such as road networks and Traffic Assignment zones, to better understand vehicle mobility patterns and subsequently, improve urban planning, traffic control, and infrastructure maintenance.

INRIX provides travel time data at aggregate Level at each waypoint. This information is used to generate a sample O.D. matrix. For each day and at every hour, we need to locate the Start and end locations of the Trips generated by GPS data corresponding to the desired area. The Start and End locations containing information about Longitude and Latitude are converted to point shapefile using the ArcMAP tool. This data needs to be joined with Traffic Assignment zones (polygon file created earlier) using spatial join in order to identify origin and destination zones for each trip id. Geographical Coordinate Systems (GCS) of both the shapefiles should be the same to avoid any error. Below Figure 4a & 4b shows the traffic assignment zone file (polygon) and INRIX OD file joined with TAZ file from ArcMap.

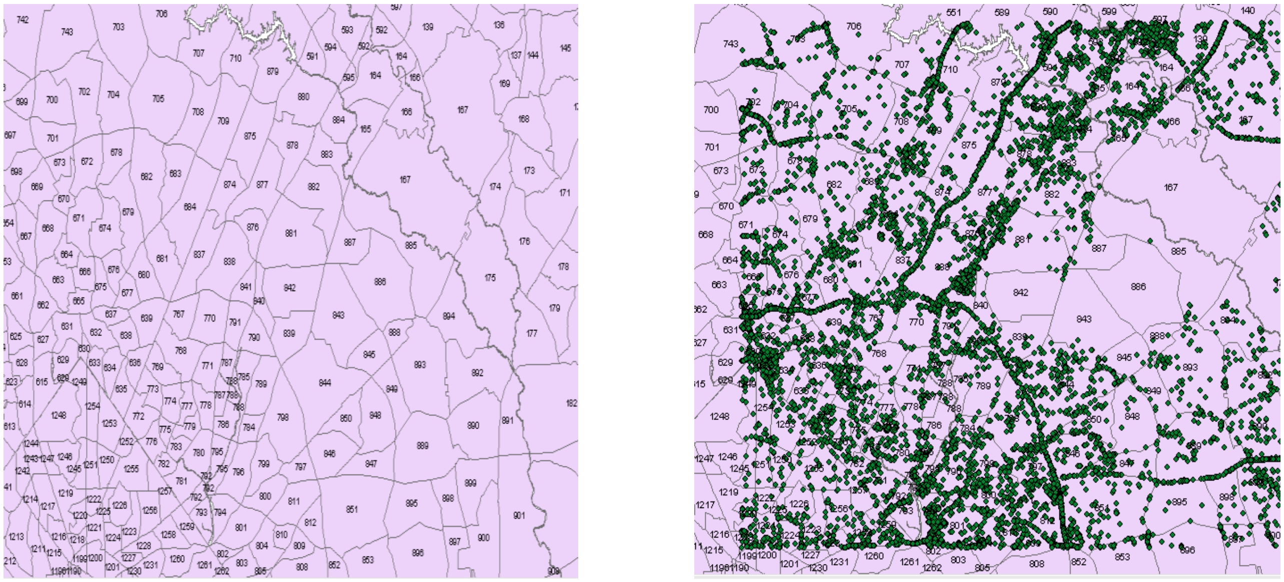
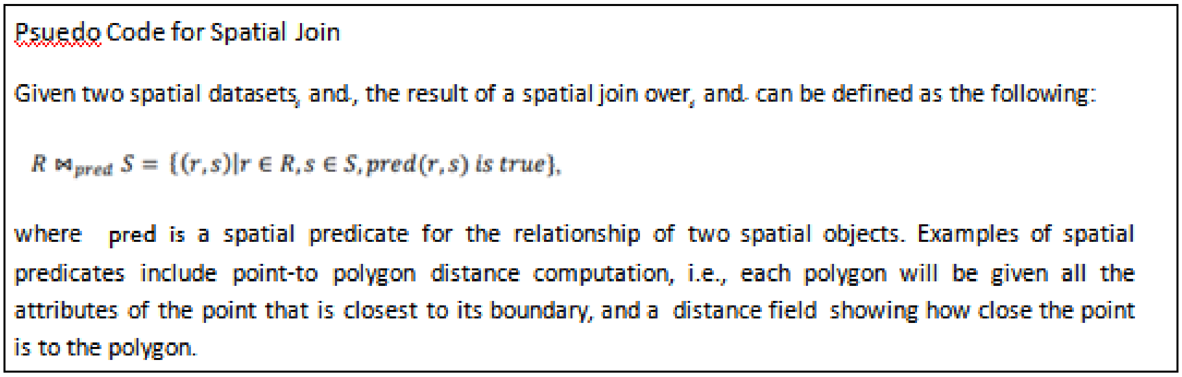


Figure. 4a) Polygon Shapefile.

Figure. 4b) Spatial Join of Point and Polygon file

Figure 4. Polygon Shapefile and Spatial join



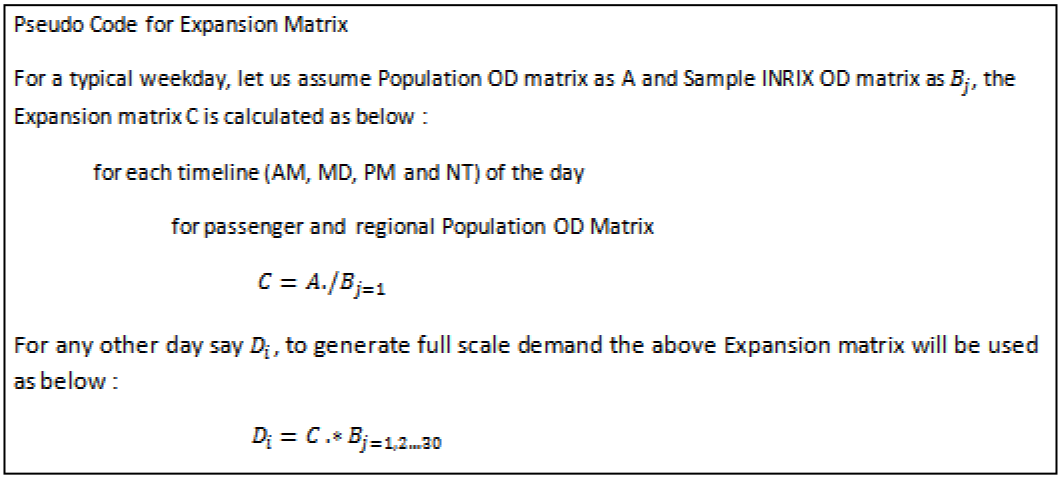
Psuedo Code for Spatial Join

Given two spatial datasets, and, the result of a spatial join over, and can be defined as the following:

$$R \bowtie_{pred} S = \{(r,s) | r \in R, s \in S, pred(r,s) \text{ is true}\},$$

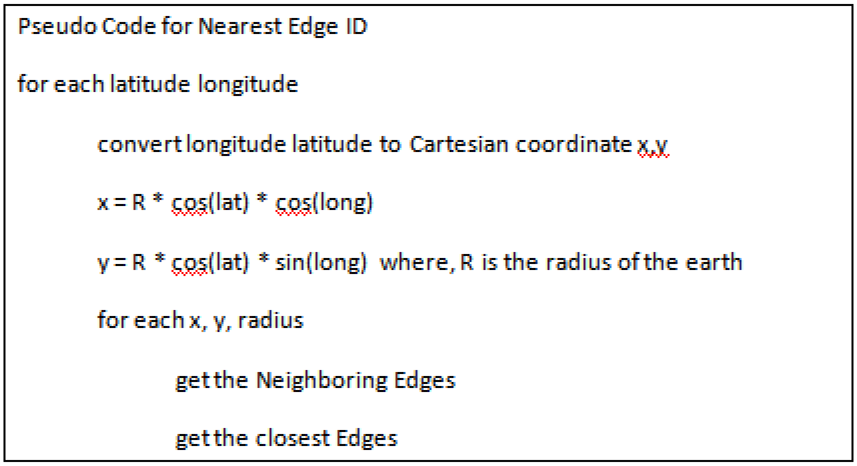
where pred is a spatial predicate for the relationship of two spatial objects. Examples of spatial predicates include point-to polygon distance computation, i.e., each polygon will be given all the attributes of the point that is closest to its boundary, and a distance field showing how close the point is to the polygon.

The population O.D. matrix and the sample O.D. matrix generated from INRIX data provide an Expansion Matrix. Then this expansion matrix is used to generate full demand (OD Matrices) for any given day from INRIX. This data is used to create an FMA file for each day.



```
Pseudo Code for Expansion Matrix

For a typical weekday, let us assume Population OD matrix as A and Sample INRIX OD matrix as $B_j$, the
Expansion matrix C is calculated as below :

        for each timeline (AM, MD, PM and NT) of the day

                for passenger and regional Population OD Matrix

                        $C = A./B_{j=1}$

For any other day say $D_i$, to generate full scale demand the above Expansion matrix will be used
as below :

                        $D_i = C .* B_{j=1,2\ldots30}$
```

b) Using the geographical location stamps for each individual trip from INRIX, information about each zone and its corresponding edges are retrieved. This has defined the INRIX traffic assignment zones. The Pseudo Code is as shown in the below-



```
Pseudo Code for Nearest Edge ID
for each latitude longitude
    convert longitude latitude to Cartesian coordinate x,y
    x = R * cos(lat) * cos(long)
    y = R * cos(lat) * sin(long)  where, R is the radius of the earth
    for each x, y, radius
        get the Neighboring Edges
        get the closest Edges
```

c) Further trip distribution percentage is also retrieved from INRIX for every 30 minutes. This information is then given to the OD2TRIPS tool to generate traffic demand. This process is iterated to create demand data for all 30 days.

Detector setup: In order to collect the trip/road information after running the simulations, we have placed inductive loops, lane area and multi-entry-multi-exit detectors at various points in network map. Details such as detector type, id, location and sampling frequency are defined in additional configuration file. In network map, these detectors are positioned at each junction with traffic light and entry and exit ramps. We have also placed these detectors at those locations extracted from Department of traffic Operations. This would result in collecting data for further research purpose. We fixed the location of each inductive loop close to the intersection to allow dynamic adjustments of the traffic light system using the information provided by the detectors as a feasible extension of the simulation. In case of the inductive loops situated on the highway, one of the possible usage is the monitoring of traffic flows on the peripheral roads [35]. All these are static detectors. We have also simulated dynamic detector Bluetooth. SUMO enables the simulation of wireless devices on vehicles. Every vehicle can be configured to either send or receive the radio signals. We can control the percentage of Bluetooth enable vehicles in simulation. In this research, we have limited the percentage of such vehicles to 5%.

Table 2. Number of Static Detectors

Detector	**Total Number**
Inductive Loop (E1)	1914
Lane Area (E2)	1914
Multi-entry-multi-exit (E3)	2030

Event Data Setup: Along with real world demand, we have also simulated historic traffic event for our network map. These traffic events data is provided by Department of traffic operations which contains information as event type, event location and time stamp and duration of each event. In order to simulate each of these events in our model some assumptions have been made. These events are triggered by mainly changing 3 parameters – lane capacity, driver behaviour and duration of event. Below table 3.1 and 3.2 shows different event scenarios and assumptions. Based on event types, corresponding actions are taken in simulation. For e.g. if event type is road maintenance operations, then Variable speed sign is placed as per the location and guidelines defined in [4] and particular section of road/lane is closed for entire duration of events. In the model, this scenario was implemented by stopping the vehicles at event location and duration.

Table 3.1 Different Event Scenarios

Serial #	**Event**	**Length of Lane Blocked(in meters)**	**Warning Signs/Obstruction**	**Variable Speed Sign Required**	**Buffer distance**	**Cars involved to block the lane**	**Roadway Acessible**

1	Collision	12	30	Yes	12	4	Yes
2	Disabled Vehicle	6	0	No	12	3	Yes
3	Emergency Roadwork	122	460	Yes	0	20	Yes
6	Incident	18	30		12	5	Yes
4	Injuries involved	24	30	Yes	12	6	Yes
5	Obstruction	30	0	No	12	7	Yes
6	Road Maintenance Operations	122	300	Yes	0	21	Yes
7	Traffic signal not working	0	0	No	0	0	No

Table 3.2 Event Scenarios Assumptions

Assumptions		
	Number of cars involved in collision	2 passenger cars
	Number of cars involved in collision with injuries involved	4 passenger cars
	Average Car length	20 feet (6 meters)

	Road Maintenance Operations (lane closures are considered)	Maximum length for activity area for one way traffic is determined by the capacity required to handle peak hour demand. Practical maximum length is 400 feet (122 meters) Advance warning sign at 1000 feet (300m) total = 122+300 = 422 m
	Emergency road work	Maximum length for activity area for one way traffic is determined by the capacity required to handle peak hour demand. Practical maximum length is 400 feet (122 meters) Advance warning sign at 1500 feet (460 m) total = 122+460 = 582 m
	Lane section length	1000 meter
	% of vehicle drive below or at maximum speed	80%

Table 3.2 Continued

	% of vehicle drive above maximum speed	20%
	Obstruction length	30 meters (100 feet) Reference from CHAPTER 2C. WARNING SIGNS AND OBJECT MARKERS - Table 2C-4. Guidelines for Advance Placement of Warning Signs (https://mutcd.fhwa.dot.gov/pdfs/2009r1r2/pdf_index.htm)

	Number of cars involved in Incident	3 passenger cars
	Incident + Collision	Incident is considered more severe than collision
	Ignore Flood, vehicle on fire, Alert	These are ignored because of less data points

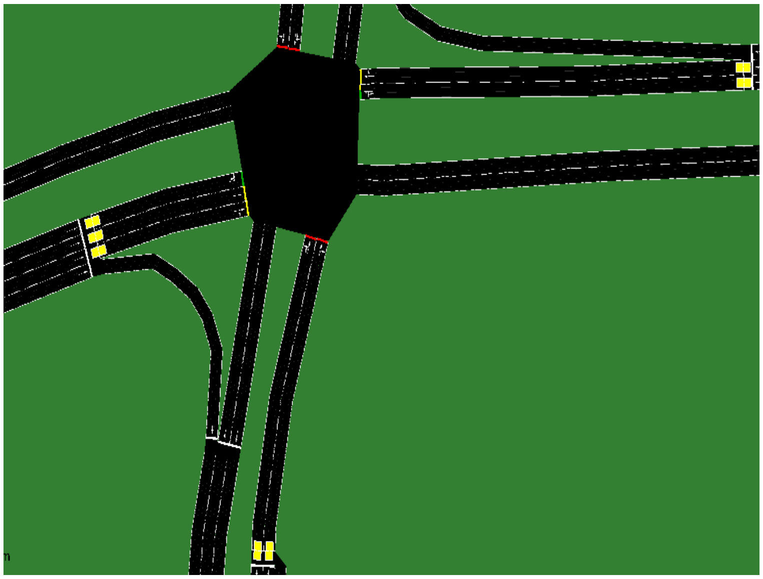


Figure 5a. Placement of Inductive loop on intersection

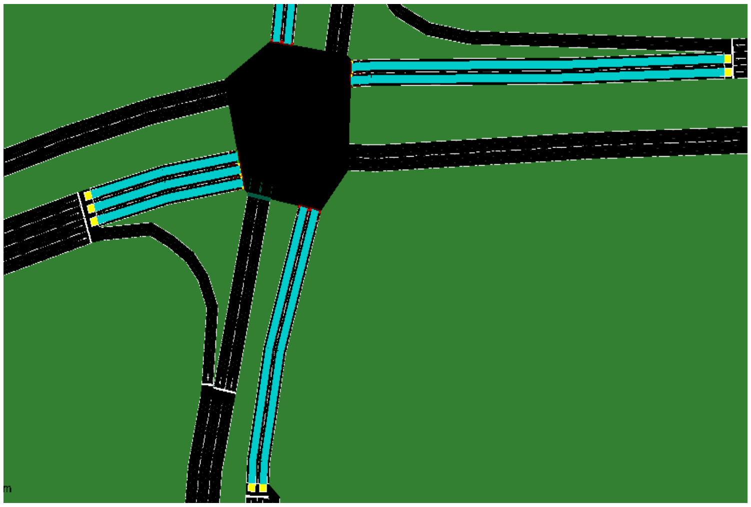


Figure 5b. Placement of Lane area detector

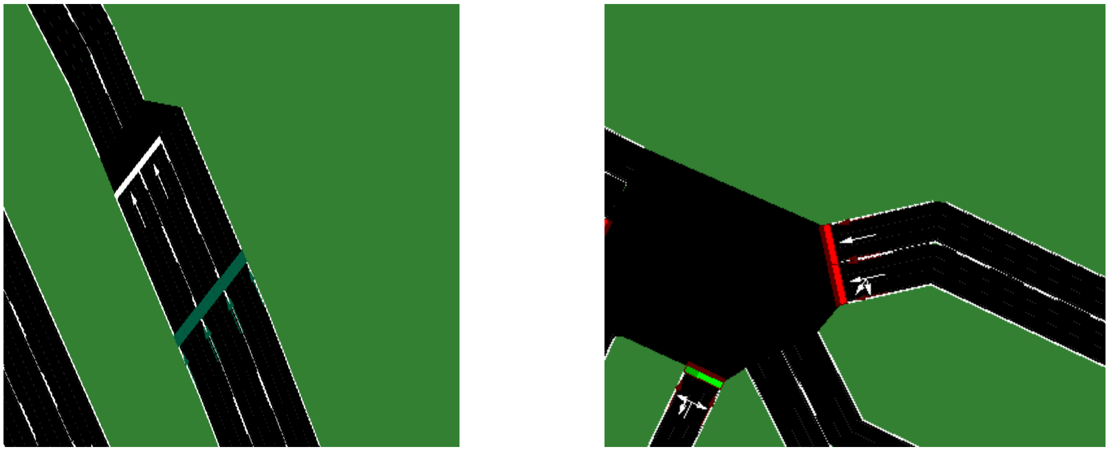


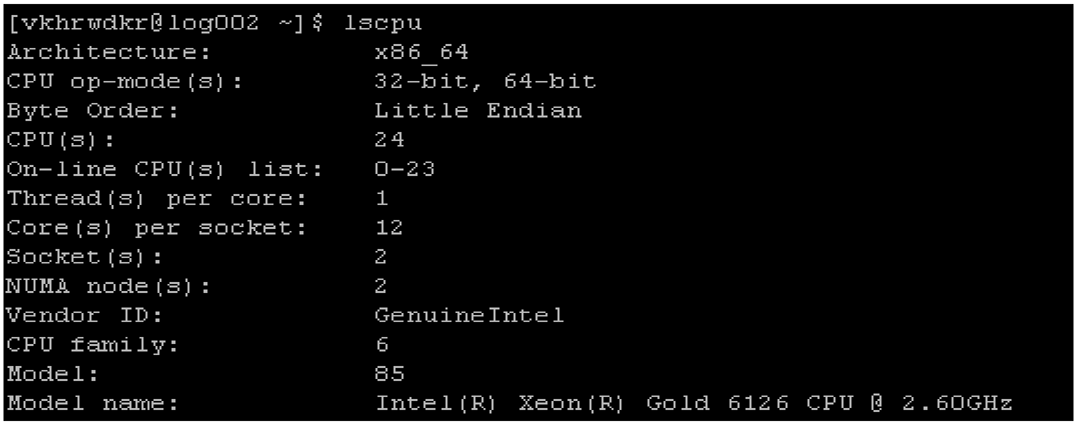
Figure 5c(a) Multi entry-multi-exit detector

Figure 5 c(b) Multi entry-multi exit detector

Figure 5c. Placement of Multi-Entry Multi-Exit detector

2.3 Data Simulation:

Once network, demand and other configuration files (detector, variable speed signs etc.) are corrected, simulation was run for continuous 30 days of INRIX data from June 1 to June 30 2015. We have provided demand in form of trip details which has origin and destination location and depart time for each vehicle. For completing the trips, SUMO [37] needs to have the entire route information which will be travelled by each vehicle. To accomplish this, SUMO provides various ways to generate route given the trip information. One approach is to generate the routing information beforehand using SUMO DUAROUTER tool. Network map and trip details are required in order to generate routes. DUAROUTER computes vehicle routes using fastest possible route computation using Dijkstra algorithm. However, the main disadvantage in using this approach is that route for each vehicle is generated in a network separately which leads to traffic congestion when all the vehicles are inserted. Another option is to assign travel time for each road which is unknown before running the simulations as it is directly dependent on number of vehicles. This problem can be overcome by running the simulation and DUAROUTER iteratively which computes the dynamic user assignment formulated by C. Gawron [5]. However, in our research as simulation is required to run for 30 days of source data, we cannot generate routes beforehand as it would be time consuming. So, we decided to use this approach at run time while running the simulation. To accomplish this, we have enabled all the vehicles with embedded routing device whose function is to recalculate travel time for each road after every given interval and provide it to DUAROUTER for computing new routes. This interval is set to 2 minutes in our model. For triggering the historic events, we have used SUMO inbuilt TraCI (Traffic Control Interface) tool which gives network access during simulation and allows to alter the roads/vehicle behaviour at run-time. For e.g. for triggering the disabled vehicle event, we are forcibly stopping the vehicle at desired location and time stamp for a given duration for corresponding event. For computing routes and running simulation at run time for this amount of huge demand is time consuming. In our research, to minimize this time, we have ran all the simulations on HPC systems. Configuration for HPC:



```
[vkhrwdkr@log002 ~]$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Byte Order:            Little Endian
CPU(s):                24
On-line CPU(s) list:   0-23
Thread(s) per core:    1
Core(s) per socket:    12
Socket(s):             2
NUMA node(s):          2
Vendor ID:             GenuineIntel
CPU family:            6
Model:                 85
Model name:            Intel(R) Xeon(R) Gold 6126 CPU @ 2.60GHz
```

SUMO provides number of output for simulation runs. All the outputs are aggregated for each road in the network and for sampling frequency of 60 seconds. We have mainly focused on floating car data, inductive loop, lane, multi-entry-multi-exit and Bluetooth detector data.

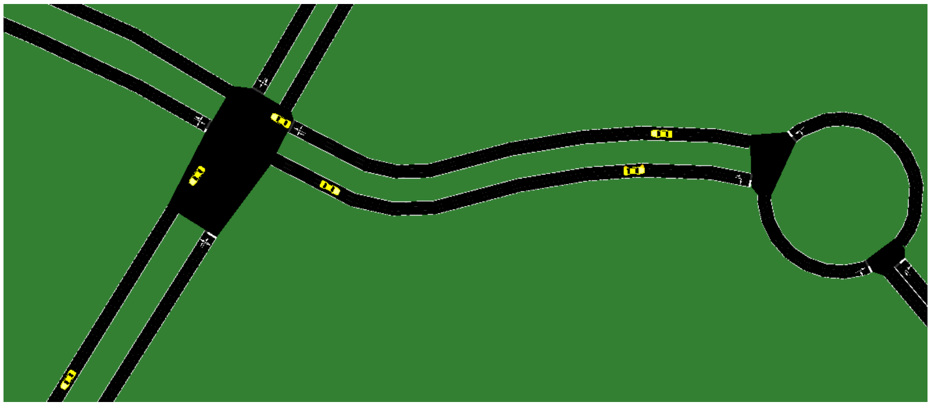


Figure 6a. Passenger vehicles on uncontrolled intersection

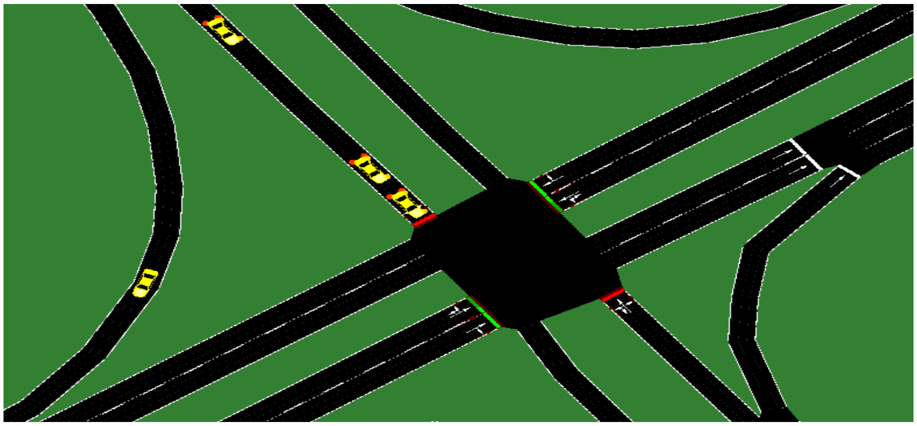


Figure 6b. Passenger vehicles on traffic lights

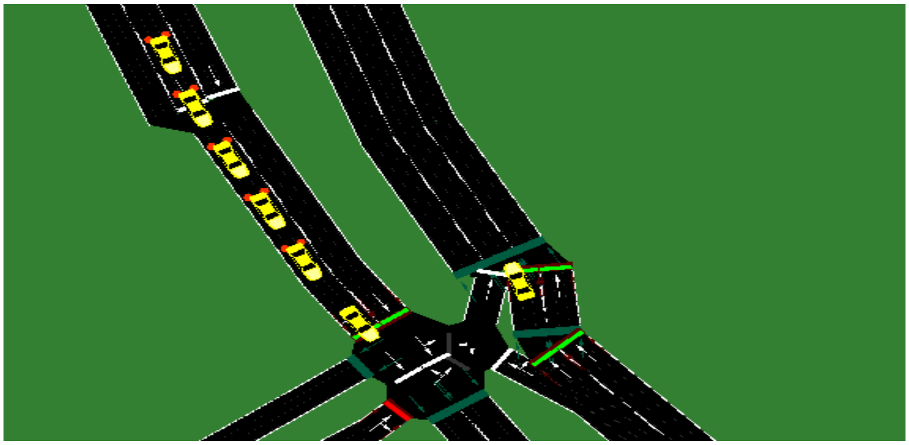


Figure 6c. Passenger vehicles queued on intersection

3. Data Validation

To study the accuracy of the simulated data, actual data was collected from Traffic Management Centers (TMCs) in Washington D.C area for a typical weekday. A typical weekday includes Tuesday, Wednesday & Thursday. The geographic location of various locations for validation and roadways are as shown in Figure 4.

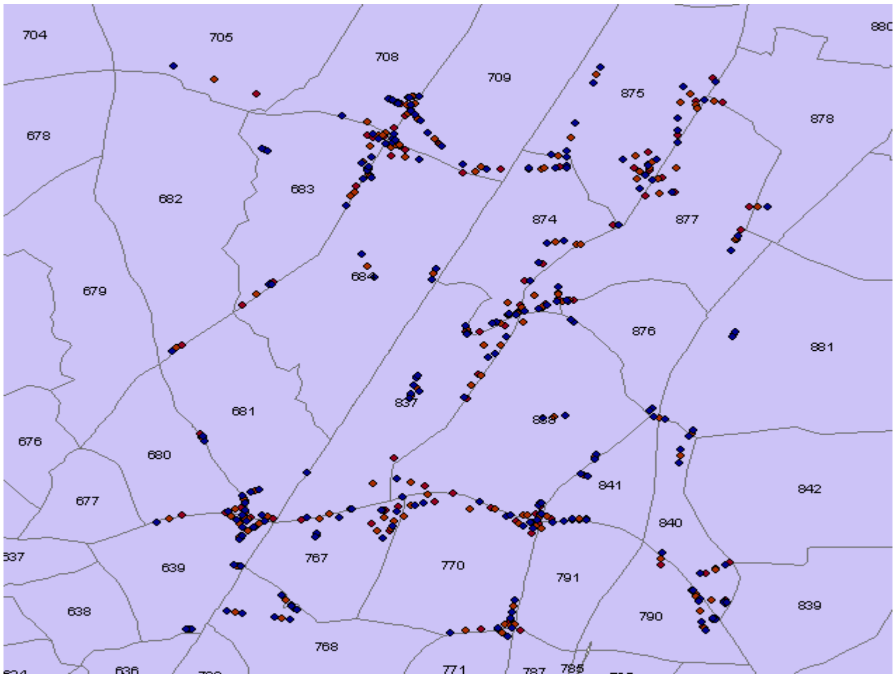


Figure 7. Location of station IDs for validation

For Validation purpose, each individual Station ID is observed and its volume count is validated with SUMO simulated validation count data. Both the datasets are aggregated for each hour of the day for all Station IDs. The Validation results are evaluated based on Volume and Speed for all station id's.

3.1 Volume based Validation

Traffic volume (the number of vehicles departing from the traffic region) and traffic absorptive volume (the number of vehicles arriving at the traffic region) of each region plays an important role to compare our results with the observed validation count at each Station ID. For our functional area, Annual average daily traffic (AADT) for a particular weekday at frequency of 60 minutes is available at various Station IDs across different roadways. Refer to details for the same in below Table no 4.

Simulated data from SUMO is generated at every minute for all the station id's. This data was aggregated for every 60 minute and averaged out for a timespan of 3 typical weekdays.

$$\text{For all t=1 to 24, } VT_j = \sum_{j=0}^{N} \mathrm{VS_j}$$

VT_j represent the total number of vehicles in functional area which is divided into "N" Station IDs. $\mathrm{VS_j}$ represent the total number of vehicles at Station ID j.

Machine learning algorithms perform better or converge faster when features are on a relatively similar scale and/or close to normally distributed. Scaling and standardizing can help features arrive in more digestible form for these algorithms. SKLEARN.preprocessing.scale() method is helpful in standardization of data points. It divides the data points by the standard deviation and subtract the mean for each data point.

Table No. 4. Station IDs corresponding to roadways

Roadways	**# Station Ids**
Interstate	32
Major Collector	16
Minor Arterial	55
Minor Collector	2
Principal Arterial Other	14
Principal Arterial Other Freeway & Expressways	14

Table No. 5 (68 Station IDs)

Roadways	**Station Ids**
Interstate	5
Major Collector	15
Minor Arterial	34
Minor Collector	2
Principal Arterial Other	3
Principal Arterial Other Freeways and Expressways	9
Total	**68**

Simulation and actual data volume count was evaluated using 3 performance measures - correlation coefficient (R), r-square and Root mean square (RMSE). We have observed that as more number of station id's are included in evaluation, RMSE value was decreased.

Out of total 133 stations, for 68 station id's, when simulated and actual data is compared, value of correlation coefficient is ~ 0.93, R-square is ~ 87.5% and RMSE is ~ 0.22. Whereas when data is evaluated at 97 different stations id's , value of correlation coefficient is ~ 0.88, R-square is ~77.5% and RMSE is ~ 0.35.Table No. 2 & 4 shows details for these measures for 24 hours. Table No. 3 & 5 shows the no. of station id's for each of the roadways respectively. Main reason for this increase in RMSE value when more number of station id's are included is routing of vechiles which can be improved by running multiple iterations for identifying road capacity at run time which would require more time for running simulations.

Table No. 6. Validation metricswith 68 station IDs

Hour	**Correlation**	**R_Square**	**Root_Mean_Square**	**Mean_Simulated**	**Std_Dev_Simulated**	**Mean_Val**	**Std_Dev_Val**
1	0.96	92.27	0.05	72.31	117.46	151.36	269.57
2	0.92	84.84	0.04	65.18	92.83	60.99	105.50
3	0.91	82.10	0.04	52.38	69.14	45.08	71.19
4	0.95	89.52	0.05	40.73	64.89	58.55	122.49
5	0.94	87.50	0.05	33.98	51.77	116.11	238.34
6	0.96	92.31	0.04	33.02	53.39	427.29	876.20
7	0.96	92.30	0.04	53.30	80.04	712.41	1118.12
8	0.95	90.01	0.05	157.81	218.52	1021.0	1248.15
9	0.94	87.96	0.05	266.45	385.71	890.37	1030.81
10	0.90	80.64	0.05	238.16	352.70	594.80	728.06
11	0.88	77.27	0.05	525.69	651.21	573.48	717.90
12	0.92	85.18	0.05	702.03	1034.30	615.96	802.82
13	0.95	90.25	0.05	741.31	1171.86	725.66	1002.80
14	0.92	85.06	0.04	588.15	832.38	635.44	820.89
15	0.93	86.68	0.04	552.25	827.58	737.78	998.53
16	0.95	90.87	0.04	465.73	767.84	885.93	1175.10
17	0.97	93.21	0.05	745.97	1229.71	1032.8	1444.04
18	0.96	91.82	0.05	709.30	1141.66	1019.15	1248.05

Table No. 6 Continued

19	0.96	91.55	0.05	323.27	478.34	1001.68	1308.76
20	0.94	88.89	0.04	320.11	520.95	673.02	864.07
21	0.89	79.10	0.04	224.41	306.04	453.83	525.48
22	0.89	79.26	0.04	198.53	260.17	401.44	494.07
23	0.90	80.45	0.04	157.01	208.47	287.52	379.62
24	0.96	91.57	0.04	87.18	150.62	197.43	331.18

Table No. 7 (97 station Id's)

Hour	**Correlation**	**R_Square**	**Root_Mean_Square**	**Mean_Simulated**	**Std_Dev_Simulated**	**Mean_Val**	**Std_Dev_Val**
1	0.91	83.27	0.34	85.80	117.99	151.14	257.38
2	0.89	78.53	0.34	103.06	135.28	86.63	163.08
3	0.89	79.91	0.35	87.77	111.77	71.90	140.18
4	0.87	76.13	0.38	63.21	81.42	80.54	172.65
5	0.85	72.39	0.39	49.74	63.20	148.41	314.93
6	0.86	74.78	0.38	44.94	58.30	457.65	907.61
7	0.91	82.05	0.36	79.92	97.96	985.88	1636.63
8	0.90	81.73	0.35	190.18	242.32	1256.2	1588.33
9	0.89	78.49	0.39	329.30	430.39	1214.8	1541.79
10	0.91	83.38	0.31	344.80	534.15	979.01	1342.67
11	0.89	79.50	0.33	770.60	1131.72	792.41	1085.08
12	0.87	76.45	0.34	817.99	1148.68	780.54	1038.26

Table No. 7 Continued.

13	0.87	74.88	0.35	825.63	1057.99	855.53	1121.09
14	0.87	75.96	0.32	771.75	1012.53	836.38	1090.64
15	0.86	73.45	0.33	707.19	926.15	923.76	1218.27
16	0.82	67.89	0.33	445.94	572.54	992.61	1155.82
17	0.90	81.80	0.34	845.01	1228.64	1222.97	1523.83
18	0.89	79.58	0.35	798.60	1163.45	1301.05	1566.84
19	0.89	79.57	0.37	384.26	513.53	1225.10	1599.64
20	0.85	72.46	0.36	371.13	507.06	904.74	1113.69
21	0.87	76.22	0.37	319.75	435.97	735.58	1026.48
22	0.87	76.19	0.37	287.54	378.29	617.00	886.90
23	0.87	75.74	0.37	231.01	305.70	428.94	654.81
24	0.89	79.26	0.36	116.82	160.03	281.37	440.57

Table No. 8. Station IDs corresponding to roadways

Roadways	**Station Ids**
Interstate	10
Major Collector	16
Minor Arterial	49
Minor Collector	2
Principal Arterial Other	10
Principal Arterial Other Freeways and Expressways	10

Figure 8 (a & b) shows the linear relation between simulated and actual datapoints for volume based validations for a complete day.

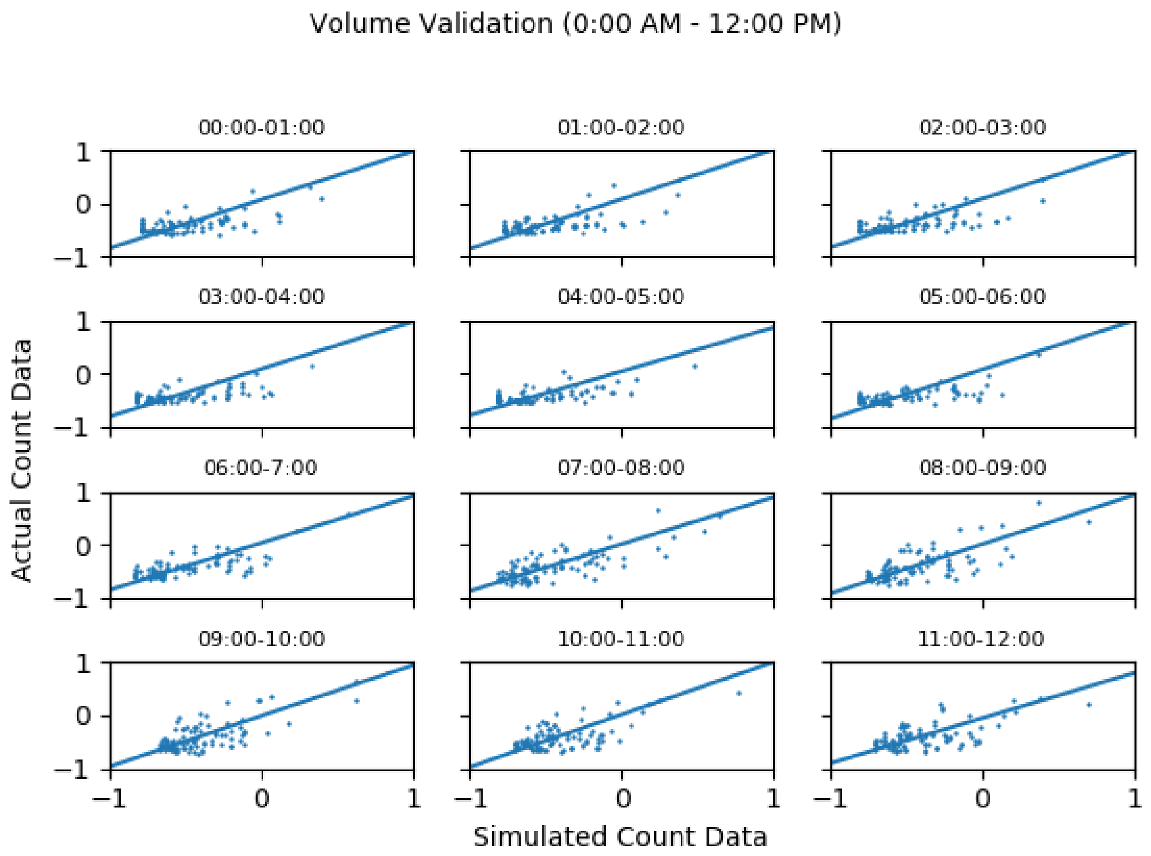


Figure 8 (a) Volume Validation for 68 Station Ids

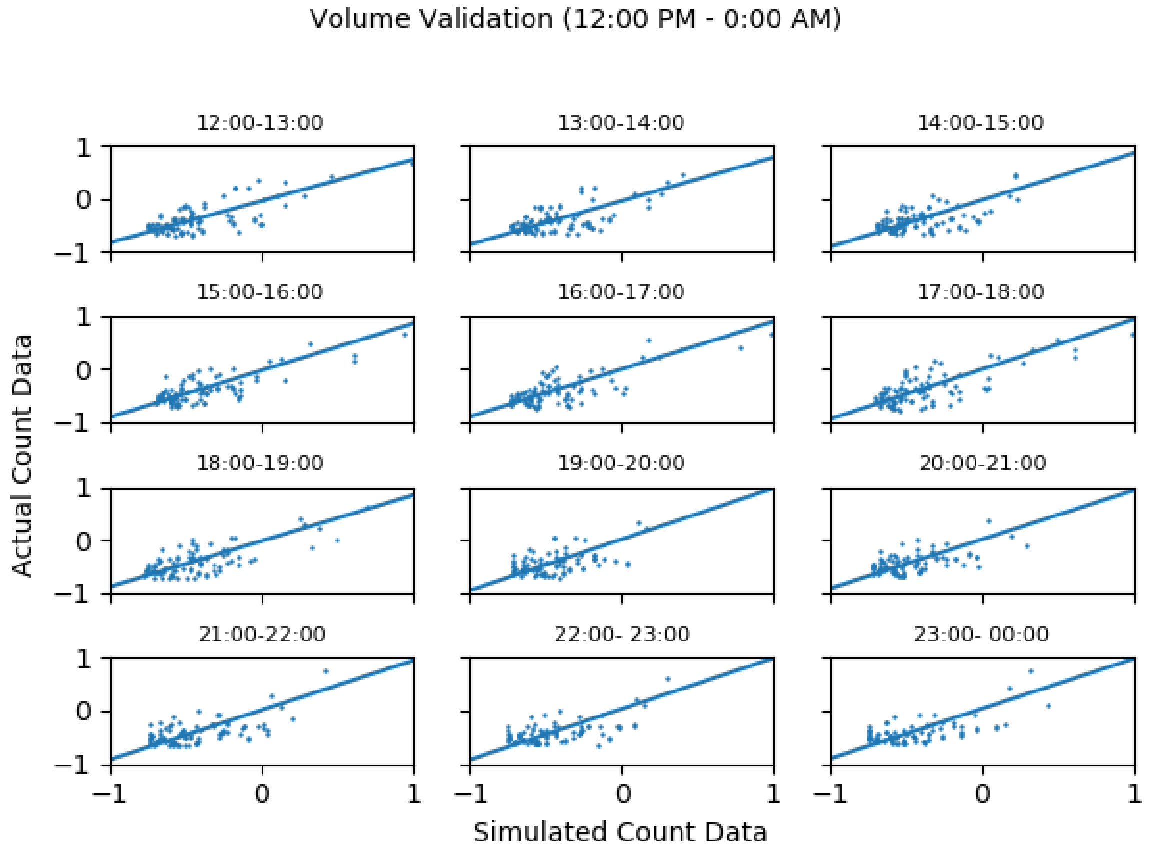


Figure 8(b) Volume based Validation for 68 Station Id's

3.2 Speed based Validation

INRIX data gives the information at individual trip level with intermediate waypoints for each timestamp and location stamp. Many real-time traffic-monitoring applications only require speed or travel time. In recent years INRIX Traffic has started collecting and selling real-time speed data collected from ‘‘a variety of sources". In order to compare INRIX speed feeds with the simulated output for every station id, average speed is determined by using difference between 2 geographical locations and corrosponding timestamps between 2 waypoints. At first, for each trip, speed is calculated at every waypoint.

$\forall t \in \{1, 2 \ldots n\}$

$v^t = d^t/T^t$

This uses the ‘haversine’ formula to calculate the great-circle distance between two points - that is, the shortest distance over the earth’s surface.

$a = \sin^2(\Delta\varphi/2) + \cos \varphi 1 * \cos \varphi 2 * \sin^2(\Delta\lambda/2)$

where,

R = 6371; // metres

φ1 = lat1.toRadians();

φ2 = lat2.toRadians();

Δφ = (lat2-lat1).toRadians();

Δλ = (lon2-lon1).toRadians();

c=2 arcsin(sqrt(a))

d = R * c * 1000 (in meters)

where n is total number of trips, φ is latitude, λ is longitude, R is earth’s radius (mean radius = 6,371km); note that angles need to be in radians to pass to trig functions!

T is calculated by subtracting 2 timestamp for respective waypoints.

Once speed and time is obtained for each trip id at everytime stamp, data is averaged for each station id and an hour of the day in INRIX data.

$\forall a \in \{1, 2 \ldots n\}$

where, a = station ID

n = total number of Station IDs.

$\forall t \in \{1, 2 \ldots 24\}$

$v(INRIX) \ = \ \sum_{t=1}^{n} s^t$ /n

……………………………………………………………………………………….(1)

Similarly, average speed is calculated from simulated output by taking mean of speed data for each station id and an hour of the day.

∀date ∊ {1, 2 . . . 30}

∀Station ID ∊ {1, 2 . . . s}

∀a ∊ {1, 2 . . . n}

$v(SUMO) \ = \ \sum_{t=1}^{n} s^t$ /n

……………………………………………………………(2)

Similar to volume validation, simulation and actual data for average road speed was evaluated using 3 performance measures - correlation coefficient (R), r-square and Root mean square (RMSE). Below table shows the all 3 performance measures for 24 hours for a typical weekday with value of correlation coefficient is ~ 0.96, R-square is ~ 93.0% and RMSE is ~ 0.25.

Table No 9. Speed valdation metrics

Hour	**Correlation**	**R_Square**	**Root_Mean_Square**	**Mean_Simulated**	**Std_Dev_Simulated**	**Mean_Val**	**Std_Dev_Val**
1	0.94	0.88	0.35	23.21	3.03	24.07	2.96
2	0.96	0.92	0.28	22.00	4.29	22.47	4.54
3	0.98	0.96	0.20	20.71	4.70	21.50	4.75
4	0.98	0.96	0.19	20.14	5.02	20.38	5.00

Table No 9. Continued.

5	0.97	0.93	0.26	22.63	3.52	23.69	3.46
6	0.88	0.78	0.49	23.46	2.27	24.00	2.57
7	0.90	0.80	0.46	23.28	2.26	23.94	2.59
8	0.94	0.89	0.33	23.27	2.79	23.60	2.87
9	0.99	0.98	0.16	18.41	6.06	18.53	6.35
10	0.98	0.96	0.20	15.06	4.80	15.17	4.92
11	0.99	0.97	0.17	14.13	5.75	14.00	5.74
12	0.98	0.97	0.18	12.19	6.57	12.57	6.32
13	0.98	0.97	0.18	13.45	6.83	14.00	6.99
14	0.99	0.97	0.16	15.90	7.62	16.75	7.79
15	0.98	0.95	0.22	19.22	5.84	19.67	5.70
16	0.99	0.97	0.16	16.00	6.62	16.94	6.83
17	0.99	0.98	0.16	18.90	5.80	19.55	6.44
18	0.91	0.83	0.42	22.89	2.81	23.50	2.87
19	0.98	0.96	0.21	20.71	5.11	21.43	5.21
20	0.99	0.97	0.17	19.45	5.38	19.91	5.68
21	0.99	0.98	0.15	16.29	7.76	16.79	7.87
22	0.98	0.97	0.19	18.73	6.01	19.27	6.17
23	0.85	0.72	0.55	22.89	1.97	23.00	2.00
24	0.99	0.97	0.16	18.86	5.23	19.79	5.32

Figure 9 shows the linear relation between simulated and actual datapoints for volume based validations for a complete day.

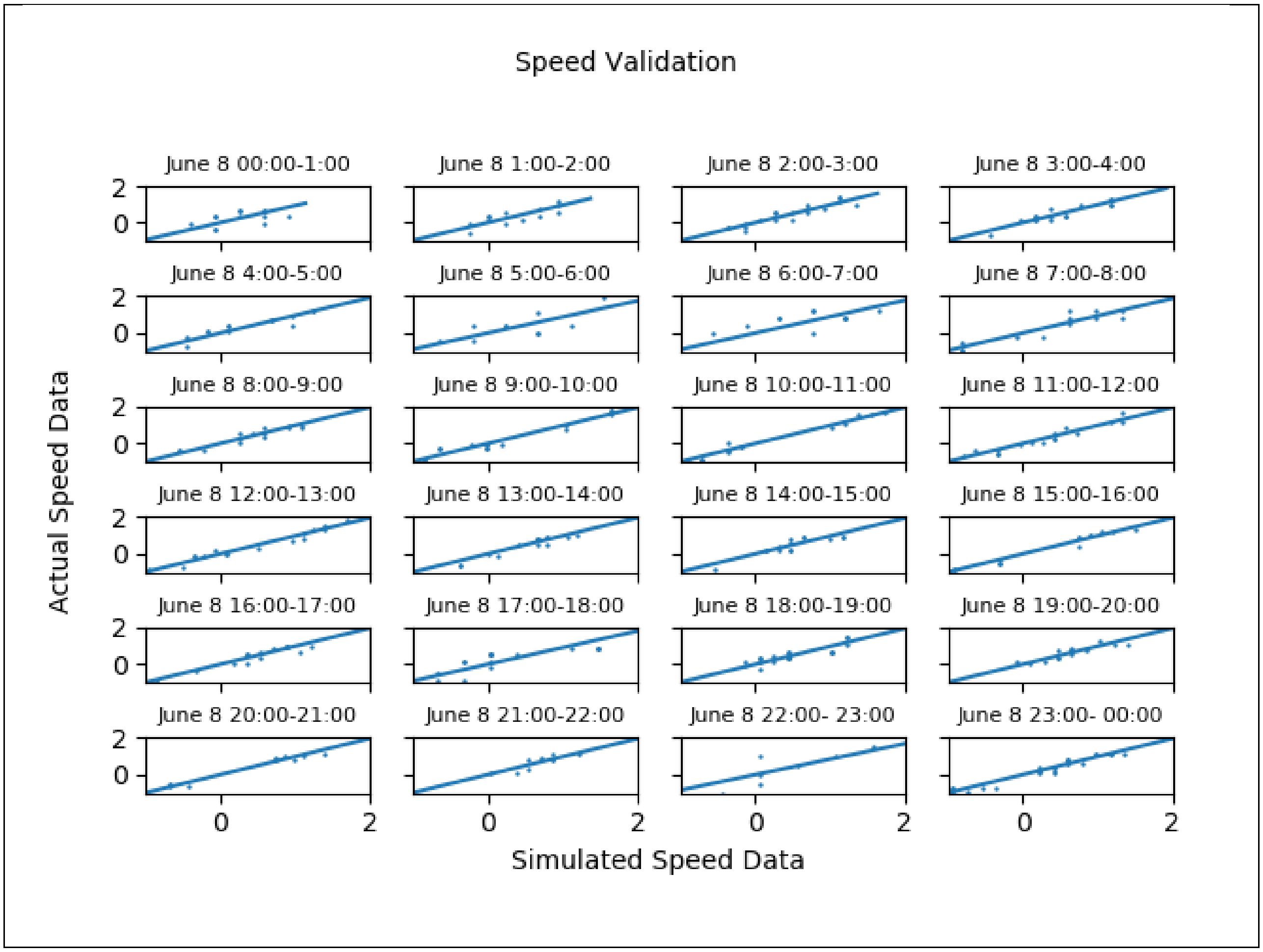


Figure 9. Speed Validation

4. Recognition of Traffic Events

4.1 Classification problem

Let $\{X, Y\}$ denoted the data pair (sample, label), θ the parameters.

A classification algorithm is a decision function $f(x\,;\,\theta)$ together with a cost/risk function $C(f(\cdot\,;\,\theta),X,Y)$ specifying the functional form of f and C completely specifies the classification algorithm. $\hat{\theta} = \arg min_{\theta}\, C(\theta)$ defines the classifier.

In order to classify the events in generated simulated data from different sensors, we need to have the information for the location stamp and time stamp for the exact events. For this purpose, historical events data is provided by Department of Transportation Operations. This data contains information for different traffic events and corresponding location and time stamp. Below is summary for these events. With respect to our functional area, total 386 events are determined with 7 different classes for the month of June 2015.

Table No. 10.Number of occurrences of events

Events	**No. of Occurrences**
Collision	30
Disabled Vehicle	240
Emergency Roadwork	5
Injuries Involved	20
Obstructions	34
Road Maintenance Operations	45
Traffic Signal Not Working	12
Grand Total	**386**

4.2 Experimental setup

Data Balancing:

Most of the data in the real-world are imbalance in nature. This situation occurs when the distribution of the target class is not uniform among the different class levels. Classification of this type of data is one of the most challenging problems in the field of machine learning and has recently gained a great deal of interest [38]. This is because most of the known machines learning algorithms were developed with an optimal goal of maximizing the overall accuracy, which is the percentage of correct predictions made by a classifier. This results in classifiers with a high accuracy but very low sensitivity towards the positive class [39]. Therefore, the optimal goal needs to be shifted toward maximizing the sensitivity of positive class and negative class separately rather than focusing on the overall accuracy. Several methods were developed to overcome this problem; these methods include methods based on sampling techniques, cost-sensitive learning, Ensemble learning, Feature selection and algorithmic modification [40].

Mathematical Definition of Imbalanced class classification-

Let Y denotes the initial data set , with $Y^1 = \{ y_1^1 , y_2^1 , \ldots y_n^1 \} \subset Z$ $\imath$s a subset of n_1 positive class records denoting 1 while $Y^0 = \{ y_1^0 , y_2^0 , \ldots y_n^0 \} \subset Z$ is a subset of n_0 negative class records that denote 0. In case of an imbalanced class dataset, we have $n_1 < n_0$ which if left unhandled can negatively affect the efficiency of a classifier [41].

Feature Extraction:

The features in the data will directly influence the predictive models and the results achieved. Feature extraction is a process of automatically reducing the dimensionality of these types of observations into a much smaller set that can be modelled. Feature importance and selection can inform you about the objective utility of features, but those features have to come from somewhere.

You need to manually create them. This requires spending a lot of time with actual sample data (not aggregates) and thinking about the underlying form of the problem, structures in the data and how best to expose them to predictive modeling algorithms.

With tabular data, it often means a mixture of aggregating or combining features to create new features, and decomposing or splitting features to create new features. With more features, the data becomes high dimensional and decision boundaries can be easily created.

For all the five different sensors , these features were extracted :

1. CaptureSecond into Day, Hour, Minute
2. Day into weekday to weekend
3. Hour into Peak-NonPeak
4. Each feature was time shifted by T+1 and T-1

Data Pre-processing :

The pre-processing involves joining events data with all sensor outputs based on location and time stamp.

Train – Test Split:

Before applying over sampling methods, base dataset was split into training and testing dataset in the ratio of 70%:30%. Later on 70% training samples, SMOTE method is applied to balance the entire training dataset. This dataset is used to train the different models as Decision Tree, KNN, Ensemble Boosted tress, Ensemble Bagged trees and Random Forest algorithm.

Data Balancing:

The synthesized data from all the five sensors was mapped to the respective events by referring to their location and time stamp. Like most of the real world datasets, these simulated datasets are also highly imbalanced in nature where most of the samples belong to category where no events were observed. For our classification problem, we need to build the model which can identify the correct events. If model is created using samples without adjustment in class distribution, then overall accuracy would be higher but it might not be able to correctly classify the events and hence poor traffic forecast. The reason is very less samples are available for some classes of the data.

In order to deal with this, under sampling or over sampling methods are used to adjust the class distribution of the training data and either one or both methods can be used to deal with class imbalance. Under sampling methods will tend to adjust the distribution of the majority class and over sampling methods would adjust the distribution of minority class. In some oversampling techniques, samples are exactly replicated, thus leading to overfitting. Also, oversampling would increase the number of training examples, thus increasing the total learning time. On the other hand, some undersampleing techniques could discards potentially useful data.

Re-sampling methods can be classified into basic sampling techniques and advanced sampling techniques. Basic sampling techniques include methods such as Random under-sampling (RUS) of majority class, Random over-sampling (ROS) of minority class, and a hybrid of both. On the other hand, advanced sampling techniques are basically based on the idea of a guided sampling approach which has been utilized using special methods. These methods include Tomek Link (T-

Link) [42], Synthetic Minority Oversampling Technique (SMOTE) [43], Neighborhood Cleaning Rule (NCR) [44], Edited Nearest Neighbor Rule (ENN) [45] etc.

Ensemble learning is a machine learning method that uses multiple learners-called base learners-to learn from multiple bootstrap samples generated from the training data set. It has a strong generalizability as compared to machine learners that use a single learner because of its ability to boost weak learners in to stronger learners and finally aggregate the results and make the predictions based on the majority of votes. Ensemble learning method is based on the work done by Breiman [46]. It includes methods such as bagging and boosting.

In this thesis we have balanced all the datasets using Synthetic Minority Ovesampling Technique (SMOTE). SMOTE is an advance method of over-sampling developed by Chawala [43]. It aims to enrich the minority class boundaries by creating artificial examples in the minority class rather replicating the existing examples to avoid the problem of

overfitting.

The algorithm works as follows:

SMOTE-

A matrix defining the distance between corresponding feature values for all feature vectors is created. The distance δ between two corresponding feature values is defined as follows.

$$\delta(V_1, V_2) = \sum_{i=1}^{n} \frac{C_{1i}}{C_1} - \frac{C_{2i}}{C_2}^{\;k} \tag{1}$$

In the above equation, V_1 $and\, V_2$ are the two corresponding feature values. C_i is the total number of occurrences of feature value V_1, and C_{1i} is the number of occurrences of feature value V_1 for class i. A similar convention can also be applied to C_{2i} and C_2. k is a constant, usually set to 1. This equation is used to compute the matrix of value differences for each nominal feature in the given set of feature vectors. Equation 1 gives a geometric distance on a fixed, finite set of values. The distance Δ between two feature vectors is given by:

$$\Delta(X,Y) = w_x\ w_y \sum_{i=1}^{N} \delta(x_i\ , y_i\ )^r$$

r = 1 yields the Manhattan distance, and r = 2 yields the Euclidean distance. w_x and w_y are the exemplar weights. w_y = 1 for a new example (feature vector), and w_x is the bias towards more reliable examples (feature vectors) and is computed as the ratio of the number of uses of a feature vector to the number of correct uses of the feature vector; thus, more accurate feature vectors will have w_x ≈ 1 [43].

4.3 Performance evaluation

Classification accuracy was evaluated using different measures : Accuracy, Precision, Recall and F-statistics.

Sensitivity (Recall): The True Positive rate (TP) = $\mathrm{TP}/(\mathrm{TP}+\mathrm{FN})$

Specificity: The True Negative rate (TN) = $\mathrm{TN}/(\mathrm{FP}+\mathrm{TN})$

Precision: Positive predictive value (PPV) = $\mathrm{TP}/(\mathrm{FP}+\mathrm{TP})$

Negative Predictive Value (NPV)= $\mathrm{TN}/(\mathrm{FN}+\mathrm{TN})$

F1 score (Fi): $2 * \mathrm{Precision} * \mathrm{Sensitivity}/(\mathrm{Precision} + \mathrm{Sensitivity})$

Weighted accuracy = 0.5* Sensitivity + 0.5* Specificity

Classification Algorithms:

Fine Decision Tree:

Decision tree learning is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. Learned trees can also be re-represented as sets of if-then rules to improve human readability. Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute [46]. An instance is classified by starting at the root node of the tree, testing the attribute specified by this node, then moving down the tree branch corresponding to the value of the attribute in the given example. Basic trees may differ based on their splits that could be controlled to produce diversity of results. The performance of each type of tree is assessed on the entire data set. Fine Tree is defined by increasing the maximum splits allowed in the generation process.

Ensemble Methods: Ensemble learning is a concept of combining several decision trees to predict better results than single decision tree. Main idea behind ensemble tree is to increase accuracy of predictions by grouping week learners to build strong learner. Advantage of using this is that it overcomes the problem of overfitting and underfitting. We have used below 3 ensemble methods.

Bagged Trees: It is a technique to reduce the variance of a decision tree. It creates several subsets of data from training sample chosen randomly with replacement. Each collection of subset data is used to train their decision trees. As a result, we end up with an ensemble of different models. Average of all the predictions from different trees are used which is more robust than a single decision tree [47].

Boosted Trees: It is another ensemble technique to create a collection of predictors. In this technique, learners are learned sequentially with early learners fitting simple models to the data and then analyzing data for errors. In other words, we fit consecutive trees (random sample) and at every step, the goal is to solve for net error from the prior tree [47].

Random Forest: Random Forest classification belongs to supervised machine learning algorithm. is an extension over bagging. It takes one extra step where in addition to taking the random subset of data, it also takes the random selection of features rather than using all features to grow trees. Accuracy of model can be improved by tuning hyperparameter using k-fold cross-validation method along with grid search method. In this, different set of parameter values are provided in a grid and model is trained will all possible combinations. Here, for each set of parameter values, scores are compared and keeps the best one. Along with K-fold cross validation, model is run on different folds for each set of hyperparameter to get more accurate performance. In our research, we have tuned model with below grid parameters. Based on these, model ran for a set of 500 iterations and which performed best was then used on testing dataset.

Number of trees in forest: [5, 10, 25, 50, 100]

Maximum depth: [2, 5, 10, 15, 20]

Criterion: [‘gini’, ‘entropy’]

Bootstrap: [True, False]

K-fold: 5

Best hyperparameters:

Table 11. Best hyperparameters for Random Forest

Best Parameters	Edge State	E1	E2	E3	BT
Number of trees in forest	100	100	50	10	100
Maximum Depth	20	20	20	15	20
Criterion	entropy	entropy	entropy	gini	entropy
Bootstrap	TRUE	TRUE	TRUE	FALSE	TRUE

Results of Classification models for different sensors :

Aggregated Mobile GPS data-

Table 12. Classification Results for Aggregated Mobile GPS Data

Aggregated Mobile GPS data	Overall Accuracy	Overall Precision	Overall Recall	Overall F1 Score	Average Precision	Average Recall	Average F1 Score
Fine Tree	91.90%	90.68%	84.79%	87.41%	47.41%	79.84%	51.63%
Fine KNN	99.90%	99.70%	99.70%	99.70%	96.76%	96.71%	96.70%
Bagged Tree	99.90%	99.70%	99.70%	99.70%	97.44%	97.28%	97.35%
Boosted Tree	85.10%	83.40%	73.28%	77.69%	32.90%	75.30%	36.45%
Random Forest	100.00%	99.90%	99.90%	99.90%	97.50%	97.50%	97.62%

Inductor Loop-

Table 13. Classification Results for Inductor Loop

Inductor Loop	Overall Accuracy	Overall Precision	Overall Recall	Overall F1 Score	Average Precision	Average Recall	Average F1 Score
Fine Tree	98.60%	98.18%	97.47%	97.79%	50.88%	70.92%	54.13%
Fine KNN	99.40%	98.89%	98.85%	98.87%	73.73%	74.82%	74.19%
Bagged Tree	99.80%	99.40%	99.40%	99.40%	86.42%	84.56%	85.30%
Boosted Tree	97.80%	97.20%	96.30%	96.70%	43.06%	65.76%	45.50%
Random Forest	100.00%	99.78%	99.75%	99.76%	85.50%	82.00%	83.37%

Lane area detector-

Table 14. Classiication Results for Lane area detector

Lane Area Detector	Overall Accuracy	Overall Precision	Overall Recall	Overall F1 Score	Average Precision	Average Recall	Average F1 Score
Fine Tree	83.10%	79.17%	69.53%	73.30%	43.15%	79.00%	50.00%

Fine KNN	99.30%	98.90%	98.90%	98.90%	93.75%	93.48%	93.61%
Bagged Tree	99.40%	98.90%	98.80%	98.80%	94.01%	96.80%	95.06%
Boosted Tree	79.20%	76.02%	65.43%	69.40%	37.68%	78.01%	39.06%
Random Forest	100.00%	99.70%	99.70%	99.70%	94.25%	97.25%	95.50%

Multi-Entry Multi-Exit detector-

Table 15. Classification Results for Multi-Entry Multi-Exit detector

Multi Entry Multi Exit Detector	Overall Accuracy	Overall Precision	Overall Recall	Overall F1 Score	Average Precision	Average Recall	Average F1 Score
Fine Tree	99.80%	99.50%	99.50%	99.50%	90.65%	96.35%	92.98%
Fine KNN	100.00%	99.80%	99.80%	99.80%	95.83%	95.73%	95.77%
Bagged Tree	100.00%	99.80%	99.80%	99.80%	99.55%	94.83%	96.94%
Boosted Tree	100.00%	99.90%	99.90%	99.90%	100.00%	100.00%	100.00%
Random Forest	100.00%	100.00%	99.90%	99.90%	100.00%	98.50%	99.25%

Bluetooth-

Table 16. Classification Results for Bluetooth

Bluetooth	Overall Accuracy	Overall Precision	Overall Recall	Overall F1 Score	Average Precision	Average Recall	Average F1 Score
Fine Tree	98.20%	97.90%	96.40%	97.15%	41.53%	74.21%	49.14%
Fine KNN	99.90%	99.70%	99.60%	99.60%	81.59%	81.95%	81.70%
Bagged Tree	99.50%	99.10%	98.80%	98.90%	80.26%	85.71%	81.22%
Boosted Tree	97.70%	97.40%	95.40%	96.30%	31.58%	73.60%	37.99%
Random Forest	100.00%	99.80%	99.90%	99.90%	84.50%	84.38%	84.13%

5. Conclusions and Future Work

- The classification model was able to predict between the normal and abnormal traffic events. The abnormal traffic events include 7 different traffic events and performance was analyzed across Precision, Recall and F1 Score.
- Large-scale traffic data was simulated for a long duration using real-world data and data from household survey based on their incomes.
- The high fidelity simulated dataset can be further used to predict driver behavior in the future work.
- This dataset can be used for solving traffic problems which can be used by traffic management operations.
- Classification can be done by taking into account the dependencies between spatial and temporal neighborhoods, and data from multiple sensors taking them together.

References

[1] Ben-Akiva M, Bierlaire M, Koutsopoulos HN, Mishalani R (2002) Real-time simulation of traffic demand-supply interactions within DynaMIT. In Gendreau M, Marcotte P (Eds.), Transportation and network analysis: current trends, pages 19–36. Kluwer Academic Publishers. Miscellenea in honor of Michael Florian

[2] Ben-Akiva M, Koutsopoulos HN, Antoniou C, Balakrishna R (2010) Traffic simulation with DynaMIT. In: Barcelo J (ed) Fundamentals of traffic simulation. Springer, pp. 363–398

[3] Mahmassani HS (2001) Dynamic network traffic assignment and simulation methodology for advanced system management applications. Netw Spacial Econ 1(3):267–292

[4] Yang Q, Koutsopoulos HN, Ben-Akiva ME (2000) A simulation model for evaluating dynamic traffic management systems. Transp Res Rec 1710:122–130

[5] Tamp`ere, C. M., R. Corthout, D. Cattrysse, and L. H. Immers. 2011. "A Generic Class of First Order Node Models for Dynamic Macroscopic Simulation of Traffic Flows". Transportation Research Part B: Methodological 45 (1): 289–309.

[6] Fl¨otter¨od, G., and J. Rohde. 2011. "Operational Macroscopic Modeling of Complex Urban Road Intersections". Transportation Research Part B: Methodological 45 (6): 903–922.

[7] Gipps, P. 1986. "Multsim: a Model for Simulating Vehicular Traffic on Multi-Lane Arterial Roads". Mathematics and Computers in Simulation 28 (4): 291–295.

[8] Bando, M., K. Hasebe, A. Nakayama, A. Shibata, and Y. Sugiyama. 1995. "Dynamical Model of Traffic Congestion and Numerical Simulation". Physical Review E 51 (2): 1035.

[9] Helbing, D., and B. Tilch. 1998. "Generalized Force Model of Traffic Dynamics". Physical Review E 58 (1): 133.

[10] Treiber, M., A. Hennecke, and D. Helbing. 2000. "Congested Traffic States in Empirical Observations and Microscopic Simulations". Physical Review E 62 (2): 1805.

[11] Li, Y., D. Sun, W. Liu, M. Zhang, M. Zhao, X. Liao, and L. Tang. 2011. "Modeling and Simulation for Microscopic Traffic Flow Based on Multiple Headway, Velocity and Acceleration Difference". Nonlinear Dynamics 66 (1-2): 15–28.

[12] Behrisch, M., L. Bieker, J. Erdmann, and D. Krajzewicz. 2011. "SUMO–Simulation of Urban MObility". In The Third International Conference on Advances in System Simulation (SIMUL 2011), Barcelona, Spain.

[13]Hunter,M.P.,R.M.Fujimoto,W.Suh,andH.K.Kim.2006."AnInvestigationofReal-TimeDynamicData Driven Transportation Simulation". In Proceedings of the 2006 Winter Simulation Conference, edited by L. F. Perrone, F. P. Wieland, J. Liu, B. G. Lawson, D. M. Nicol, and R. M. Fujimoto, 1414–1421. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

[14] Lippi, M., M. Bertini, and P. Frasconi. 2013. "Short-Term Traffic Flow Forecasting: An Experimental Comparison of Time-Series Analysis and Supervised Learning". IEEE Transactions on Intelligent Transportation Systems 14 (2): 871–882.

[15] Ezzat, A. A., H. A. Farouk, K. S. El-Kilany, and A. F. A. Moneim. 2014. "Optimization Using Simulation of Traffic Light Signal Timings". In Proceedings of the 2014 International Conference on Industrial Engineering and Operations Management Bali, Indonesia.

[16] Osorio, C., and L. Chong. 2012. "An Efficient Simulation-Based Optimization Algorithm for LargeScale Transportation Problems". In Proceedings of the 2012 Winter Simulation Conference, edited by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, 423. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, Inc.

[17] S. Uppoor, O. Trullols-Cruces, M. Fiore, and J. M. Barcelo-Ordinas, "Generation and analysis of a large-scale urban vehicular mobility dataset," IEEE Transactions on Mobile Computing, vol. 13, no. 5, pp. 1061–1075, 2014.

[18] F. Xia, A. Rahim, X. Kong, M. Wang, Y. Cai, and J. Wang, "Modeling and analysis of large-scale urban mobility for green transportation," IEEE Transactions on Industrial Informatics, vol. 14, no. 4, pp. 1469– 1481, 2018.

[19] L. Bedogni, M. Gramaglia, A. Vesco, M. Fiore, J. Härri and F. Ferrero, "The Bologna Ringway Dataset: Improving Road Network Conversion in SUMO and Validating Urban

Mobility via Navigation Services," in *IEEE Transactions on Vehicular Technology*, vol. 64, no. 12, pp. 5464-5476,Dec.2015. doi: 10.1109/TVT.2015.2475608

[20] INRIX Incorporated. INRIX Total Fusion. http://www.inrix.com/pdf/INRIX%20Total%20Fusion.pdf. Accessed July 20, 2014.

[21] https://www.civil.iitb.ac.in/~vmtom/SiMTraM_Web/html/docs/sumo-user.pdf

[22] OpenStreetMap contributors. (2015) Planet dump Retrieved from https://planet.openstreetmap.org

[23] Travel Time Data Collection Handbook, FHWA report, chapter 5, ITS Probe Vehicle Techniques, 1998. http://www.fhwa.dot.gov/ohim/handbook/chap5.pdf

[24] https://sumo.dlr.de/docs/Simulation/Output/FCDOutput.html

[25] https://sumo.dlr.de/docs/Simulation/Output/Lane-_or_Edge-based_Traffic_Measures.html

[26] Gurczik, Gaby, and Michael Behrisch. "Modelling and simulating Bluetooth-based moving observers." In 2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS), pp. 223-228. IEEE, 2015.

[27] http://sumo.sourceforge.net/userdoc/Simulation/Bluetooth.html#output

[28] Kasten, O., and M. Langheinrich. First Experiences with Bluetooth in the Smart-ITS Distributed Sensor Network. In Proceedings of 2001 International Conference on Parallel Architectures and Compilation Techniques (PACT'01), Barcelona, Spain, 2001.

[29] Klein, L.A., Mills, M.K. and Gibson, D.R. 2006. Traffic detector handbook Volume I., 3rd ed, McLean, VA: Federal Highway Administration.

[30] https://sumo.dlr.de/docs/Simulation/Output/Induction_Loops_Detectors_(E1).html

[31] https://sumo.dlr.de/docs/Simulation/Output/Lanearea_Detectors_(E2).html

[32] https://sumo.dlr.de/docs/Simulation/Output/Multi-Entry-Exit_Detectors_(E3).html

[33] Sutaria, T. C., and J. J. Haynes. "Level of service at signalized intersections." Transportation Research Record 644 (1977).

[34] Gurczik, Gaby. "Performance Measurement of a Bluetooth-based Floating Car Observer." Transportation research procedia 25 (2017): 1839-1850.

[35] https://orbilu.uni.lu/bitstream/10993/23011/1/LC_LuSTScenario.pdf

[36] https://osha.gov/doc/highway_workzones/mutcd/figures.html

[37] "Microscopic Traffic Simulation using SUMO"; Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. IEEE Intelligent Transportation Systems Conference (ITSC), 2018.

[38] Qiang Yang, Xindong Wu (2006) 10 Challenging Problems in Data Mining Research. International Journal of Information Technology & Decision Making 4: 597-604.

[39] Gu J, Zhou Y, Zuo X (2007) Making Class Bias Useful: A Strategy of Learning from Imbalanced Data. Lecture Notes in Computer Science, Intelligent Data Engineering and Automated Learning - IDEAL.

[40] Choi MJ (2010) A Selective Sampling Method for Imbalanced Data Learning on Support Vector Machines, Graduate Theses, Iowa State University.

[41] Mirza, Amaad & Asghar, Sohail & Manzoor, Awais & Noor, Muhammad. (2019). A Classification Model For Class Imbalance Dataset Using Genetic Programming. IEEE Access. 7. 71013-71037. 10.1109/ACCESS.2019.2915611.

[42] Tomek Ivan (1976) An Experiment with the Edited Nearest- Neighbor Rule. IEEE Transactions on Systems. Man, and Cybernetics 6: 448-452.

[43] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP (2002) SMOTE: Synthetic Minority Over-sampling Technique. Journal of Artificial Intelligence Research 16: 321-357.

[44] Laurikkala J (2001) Improving Identification of Difficult Small Classes by Balancing Class Distribution. AIME, LNAI 2101, pp: 63-66.

[45] Angiulli F, Bucci P (2005) Fast condensed nearest neighbor rule Appearing in Proceedings of the 22nd International Conference on Machine Learning, Bonn, Germany.

[46] Mitchell, Tom M. "Machine learning." (1997).

[47] Dietterich, Thomas G. "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization." Machine learning 40, no. 2 (2000): 139-157.

[48] Johnston, Kevin, Jay M. Ver Hoef, Konstantin Krivoruchko, and Neil Lucas. Using ArcGIS geostatistical analyst. Vol. 380. Redlands: Esri, 2001.

[49] https://en.wikipedia.org/wiki/OpenStreetMap