Traffic-event Unification System Highlighting Arterial Roads

Anand Rangarajan, Sanjay Ranka

anand@cise.ufl.edu, sranka@ufl.edu

Contract: BDV31-977-97

Date: May 2020

Deliverable 7: Final Report

Disclaimer

The work was supported in part by the Florida Department of Transportation. The opinions, findings, and conclusions expressed in this publication are those of the author(s) and not necessarily those of the Florida Department of Transportation or the U.S. Department of Transportation.

Technical Report Documentation Page

<table>
<tr><td colspan="2">1. Report No.</td><td>2. Government Accession No.</td><td colspan="2">3. Recipient's Catalog No.</td></tr>
<tr><td colspan="3" rowspan="2">4. Title and Subtitle
Traffic-event Unification System Highlighting Arterial Roads
Final Report</td><td colspan="2">5. Report Date
March 2020</td></tr>
<tr><td colspan="2">6. Performing Organization Code</td></tr>
<tr><td colspan="3">7. Author(s)
Yashaswi Karnati, Dhruv Mahajan, Anand Rangarajan, Sanjay Ranka</td><td colspan="2">8. Performing Organization Report No.</td></tr>
<tr><td colspan="3" rowspan="2">9. Performing Organization Name and Address,
Department of Computer, Information Science and Engineering, University of Florida
Gainesville, FL 32611</td><td colspan="2">10. Work Unit No. (TRAIS)</td></tr>
<tr><td colspan="2">11. Contract or Grant No.
TWO # BDV31 977-97</td></tr>
<tr><td colspan="3" rowspan="2">12. Sponsoring Agency Name and Address
Florida Department of Transportation
605 Suwannee Street, MS 30
Tallahassee, FL 32399</td><td colspan="2">13. Type of Report and Period Covered
Final Report
03/15/2018 - 6/15/2020</td></tr>
<tr><td colspan="2">14. Sponsoring Agency Code</td></tr>
<tr><td colspan="5">15. Supplementary Notes</td></tr>
<tr><td colspan="5">16. Abstract
Detection of traffic interruptions (caused by vehicular breakdowns, road accidents, etc.) is a critical aspect of managing traffic on urban road networks. This work outlines a semi-supervised strategy to automatically detect traffic interruptions occurring on arteries in urban road networks using high resolution data from widely deployed fixed point sensors (inductive loop detectors).</td></tr>
<tr><td colspan="3">17. Key Word
Interruption Detection; Traffic Interruptions; Data Mining; Machine learning; Semi supervised;</td><td colspan="2">18. Distribution Statement
No restrictions.</td></tr>
<tr><td>19. Security Classif. (of this report)
Unclassified.</td><td>20. Security Classif. (of this page)
Unclassified.</td><td>21. No. of Pages
76</td><td colspan="2">22. Price.</td></tr>
</table>

Form DOT F 1700.7 Reproduction of completed page authorized

Acknowledgments

The authors would like to thank the Florida Department of Transportation (FDOT) for providing high resolution datasets and for insightful discussions during the course of the project.

Contents

List of Figures

List of Tables

1 Introduction

Managing traffic incidents is one of the crucial activities for any traffic management center. These incidents are non-recurrent, may be caused due to different reasons like traffic accidents, vehicle breakdowns, debris, etc., and cause congestion. It is worth noting that not all accidents (e.g., a fender-bender) result in interruptions. From a traffic management perspective, it is thus more important to detect interruptions rather than accidents. Further, this should be done in real time so that preemptive actions can be used for mitigation. Broadly, we define an interruption to be any time period where the amount of traffic is significantly lower than normal traffic for a significant period of time. There has been significant related work in incident detection using social media data (e.g., Waze) and probe-based systems (e.g., GPS or Bluetooth tracking data). While direct and manual traffic management center (TMC) monitoring has been adequate in previous years, many TMCs have had limited operational use of automatic incident detection techniques. This is due to these techniques' high rates of false alarms, complex calibration, and low detection rates. In fact, many automatic incident detection algorithms perform poorly in the real world, compared to simulated traffic environments (Parkany and Xie, 2005) (see Chapter 2 for more details). Our goal in this work was to use loop detector and other traffic-related data to detect traffic interruptions. Loop detector data are now widely available to traffic management personnel. Additionally, with new systems, such data are available at high frequency (10 Hz) with low latency. Hence, the utilization of these data for determining traffic interruptions can have wide applicability and can be used in conjunction with other systems based on human reporting or probe-based systems. In this study, we focus on the following datasets:

1. High resolution (10 Hz) detector data
2. Signal timing information
3. HERE.com average speeds for road segments
4. Accident reports from Signal 4.

The focus of this work was on development of data mining and machine learning techniques that use historical and real-time data for fusion of information and detection of incidents and accidents on arterial networks. Given the lack of information availability of actual interruptions, one key goal of this work was to design algorithms to detect traffic interruptions in a relatively unsupervised fashion. The key contributions of our work are as follows:

1. Literature Survey: We performed a comprehensive survey of related work in this area (Chapter 2). Previous work using loop detector data was generally limited to simulation or small datasets. As a major differentiator, we utilized six months of real data for 300 intersections (each with an average of 10 detectors) to demonstrate the usefulness of our method. This dataset is roughly 700 GB in size. We believe that this is the first study that uses fine grain (10 Hz) data for a large geographical region and over a long duration of time.

2. Data Processing and Fusion: The first step is to analyze and preprocess raw data gathered from fixed point sensors or detectors (Section 3). Traffic data are full of noise due to various known and unknown factors. Cleaning and verifying the data to prepare them for analytics is a difficult and expensive task. Datasets must be parsed, and any missing data or inaccurate data have to be addressed. Data reduction and fusion techniques match, aggregate, and consolidate several heterogeneous data sets into representations that can then be used for data mining and machine learning. This is an important step in using inputs from multiple data streams, in particular for datasets that contain spatial and temporal information that has to be used in the fusion process. A crucial step in data fusion is extraction, transformation, and loading (ETL) of the data. The ETL process generally converts the source format into a relational model that can be queried. We have developed a time series-based analysis system for processing this information using Elixir and Postgres. The former allows use of multicore processing in parallel, while the latter allows for specialized processing of time series data.

3. Defining Interruptions: Labeling data is a major challenge for big data applications. Ground truth for when an interruption has occurred is not available or humanly possible. We provide a rigorous definition and mechanism for automatically labeling interruptions or events of interest (EOIs, i.e., large traffic reductions for long periods) from historical data (Section 4). We define traffic interruption as a significant, contextual, and non-periodic change observed in a combination of the following parameters: the amount of deviation of traffic volumes from predicted volumes and the duration for which actual traffic volume deviated from predicted volume.

4. Predicting Interruptions: We have developed algorithms for predicting the labeled interruptions (Section 5) assuming that the data were available in real time with as low a latency as practical. Our approach uses traffic information from recent time periods as well as historical data (from similar time periods on a previous day or week) to predict if an event of interest has occurred (i.e., traffic interruption will last for a long time). A key parameter of interest is the duration of time to wait after some reduction in traffic to declare whether an EOI has occurred at a single detector. This has an impact on the overall accuracy (in terms of false positives and false negatives). In particular, we find that waiting 60 to 90 seconds after a significant reduction in traffic is sufficient to determine EOIs with high accuracy.

5. Spatiotemporal Analysis of Traffic Interruptions: We perform a spatiotemporal analysis of all EOIs to determine if there are hotspots (i.e., intersections with a large number of EOIs consistently at a monthly basis) and spatial relationships (two EOIs occur at nearby intersections within a small time frame). This analysis is done to determine if (1) there are subsets of intersections where traffic interruptions are occurring regularly and (2) if an interruption at a intersection causes interruption at any nearby intersection.

6. Spatiotemporal Analysis of Accidents and Relationship to Traffic Interruptions: Signal Four Analytics (SFA) is a statewide, interactive, Web-based, geospatial crash analytical tool developed by and hosted at the University of Florida, Geoplan Center. SFA contains all traffic crashes reported

in the state of Florida since 2006. For this study, we extract the crashes for Seminole County for the months of Nov. 2018 to Apr. 2019. These reports have the following important attributes: location of crash (geocoordinates), time of crash, and severity of crash. For each crash, we obtained nearby intersections which could have been possibly affected by the crash based on spatial proximity (intersections within 600 m of the crash). We analyzed the interruptions caused by these accidents. A key finding is that many of these reported accidents do not lead to sizeable interruptions (in terms of amount of traffic reduction and duration) as observed in detector data (Section 8).

2 Background

Automated incident detection (AID) is of significant interest in modern transportation networks because traffic incidents and congestion can impede commercial activities and hurt the economic growth of a region. By detecting incidents quickly, the incident-response times can be reduced, along with the negative impacts of the resultant congestion and secondary incidents.

While direct and manual traffic management center (TMC) monitoring has been adequate in previous years, many TMCs have had limited operational use of automatic incident detection techniques. This is due to these techniques' high rates of false alarms, complex calibration, and low detection rates. In fact, many automatic incident detection algorithms perform poorly in the real world, compared to simulated traffic environments (Parkany and Xie, 2005). However, the steady growth and development of transportation networks has motivated a need to discover highly reliable automated methods for detecting incidents on urban transportation networks.

AID algorithms using computational statistics or machine learning have been the research community's most recent offering on this subject. Significant efforts have been made to apply techniques such as genetic algorithms (Roy and Abdulhai, 2003), Bayes classifiers (Liu et al., 2014), support vector machines (Chen et al., 2009; Liang, 2015; Xiao and Liu, 2012), neural networks (Dia and Rose, 1997; Cheu and Ritchie, 1995; Cheu et al., 2004), deep learning (Zhang and Ni, 2018; El Hatri and Boumhidi, 2018) , fuzzy logic (El Hatri and Boumhidi, 2018; Hawas, 2007), wavelet transformation (Samant and Adeli, 2000; Jeong et al., 2011; Teng and Qi, 2003) and other artificial intelligence techniques (D'Andrea and Marcelloni, 2017) for incident detection. These modern techniques are data-centric, using historic and real-time data to detect various types of traffic incidents. Consequently, these advanced solutions are only as good as the data they are built on.

Arterial roads have different characteristics from freeways because arteries feature more dynamic traffic due to the influence of several factors: closely-spaced traffic signals and intersections, dynamic queues at the intersections, pedestrian crossings and jaywalking, roadside parking, exit/entry into/from collector lanes, public transport bus stops, arterial road work, etc. These factors of the physical environment make incident detection on arteries potentially more difficult. At the same time, to account for these factors, software solutions for arterial roads must consider data from heterogeneous sources. In this survey, existing machine learning algorithms for incident detection in traffic control were reviewed and evaluated. While data from FDOT detectors, probe cars, Twitter, and human reporting or event verification systems each have a different impact on the accuracy of incident detection, modern data fusion techniques can enhance the overall quality of combined traffic information. We summarize the successful techniques in this area along with the corresponding performance criteria, and we document possible new approaches.

A traffic incident is a non-recurring and unexpected event that has a noticeable, undesirable effect which could temporarily disrupt traffic on some segment of the transportation network (Sethi et al., 1995). An incident detection, on the other hand, corresponds to generation of an incident report (Sethi

et al., 1995). Examples of traffic incidents are traffic crashes, non-recurrent congestion, abandoned or stalled vehicles, spillage of oil or debris, structural fires, etc. Timely incident detection is important because it allows emergency responders to address the incident promptly and enables TMCs to redistribute and reroute the oncoming traffic into alternate lanes in the transportation network. This reduces the impact of the primary incident and avoids creating secondary incidents or follow-on congestion.

Incident detection solutions may be manual or automatic. While manual solutions are based on human reporting systems, automatic incident detection (AID) algorithms attempt to automatically give an alarm using anomaly detection techniques with respect to traffic conditions. The following list (Parkany and Xie, 2005) shows the developments in AID algorithms research with respect to statistics and pattern recognition:

1. Standard normal deviate algorithm (Dudek et al., 1974)
2. California family of algorithms (Payne, 1975; Payne and Tignor, 1978)
3. Bayesian algorithms (Levin and Krause, 1978; Tsai and Case, 1979)
4. Time series algorithms [(Ahmed and Cook, 1980)]
5. Smoothing/filtering algorithms (Cook and Cleveland, 1974; Stephanedes et al., 1992)
6. Traffic modeling algorithms [(Willsky et al., 1980)]
7. McMaster catastrophe theory-based algorithm (Gall and Hall, 1989)
8. Image processing algorithms (Michalopoulos, 1991; Michalopoulos et al., 1993)
9. MIT algorithms (Michalopoulos et al., 1993)
10. ADVANCE algorithms (Sethi et al., 1995; Sermons and Koppelman, 1996)
11. TTI algorithms (Sermons and Koppelman, 1996)
12. UCB algorithms (Petty et al., 1997)
13. TRANSMIT algorithms (Mouskos et al., 1999; Niver et al., 2000)
14. Waterloo algorithms (Hellinga and Knapp, 2000)

In the last two decades, data-centric algorithms based on computational statistics and machine learning have been developed to produce advanced incident detection algorithms. These advanced algorithms are superior in terms of their performance, as measured by the standard performance metrics. Three performance criteria for incident detection are as follows:

1. Detection rate (DR): Fraction of actual incidents that are correctly categorized as incidents

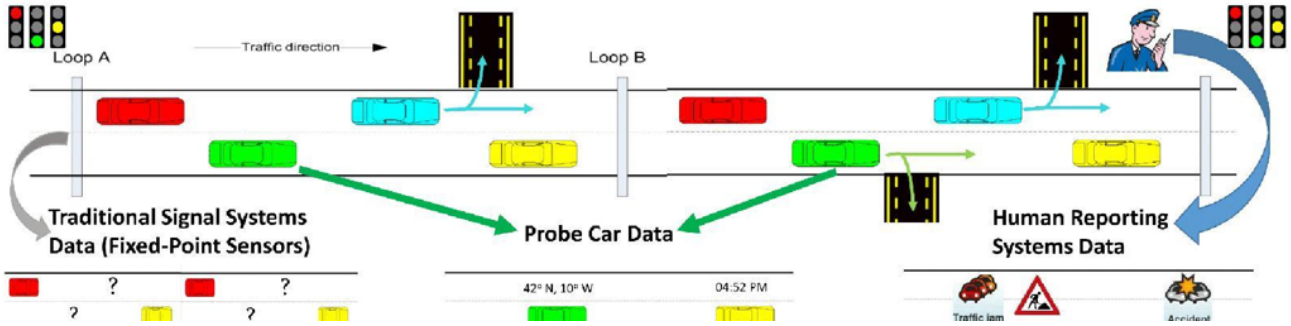

Figure 1: *Traffic data sources and their respective data of interest*

2. False alarm rate (FAR): Fraction of actual incidents that are incorrectly categorized as incidents. FAR corresponds to the fraction of non-incident periods that the algorithm incorrectly categorizes as incidents (Sethi et al., 1995).

3. Mean time to detection (MTTD): Average response time of the algorithm.

There is usually a trade-off, as well as a compromise, between the above performance factors. While Ren et al. (Ren et al., 2012) demonstrate trade-offs between FAR and MTTD, several TMCs acknowledge that early detection is more important than a better detection rate. Lastly, the algorithm's ease of calibration is another important factor taken into consideration by TMCs when adopting an AID solution.

2.1 Data Sources

The performance of the algorithms depends on the quality of the traffic data collected. Reliable sensor technologies and data feeds provide data with high accuracy, granularity, and wide spatiotemporal coverage. There are three major sources of data used for automatic incident detection:

1. Traditional signal systems (roadway-based fixed point/section)

2. Probe-based (floating car)

3. Human reporting systems, event verification, driver based

2.2 Traditional Signal Systems

Traditional signal systems sensors are fixed and measure traffic at a single point or section of a roadway. These systems exist in all urban areas and most suburban areas, but are limited in rural areas. These include inductive loop detectors (ILD), loop emulators, sensors (microwave, radar, infrared, ultrasonic, acoustic), pneumatic tubes, cameras, and video image processors. Data collected include traffic volume, occupancy, spot speed, vehicle passage or presence, queue length, turning ratios, vehicle classification, travel speed, travel time, etc. AID algorithms using signal systems roadway data rely heavily on adjacent fixed detectors. Apart from an incident possibly reducing traffic flow at nearby detectors, the basic assumption is that an incident would cause a substantial increase in occupancy at

the upstream detector, while producing a corresponding decrease in occupancy at the adjacent downstream detectors (Payne, 1976; Lin and Daganzo, 1997; Thancanamootoo and Bell, 1988; Han and May, 1989; Lee and Taylor, 1999; Chen and Chang, 1993). Existing works compare the space-time occupancy to a particular threshold value. However, selection of different threshold values produces very unique results. The California algorithm (Payne, 1976) estimates this threshold value using historical traffic data and detects incidents by taking into account the absolute difference in occupancy of adjacent detectors as well as the relative difference with respect to the upstream or downstream detector. Fluctuations in occupancy can occur due to loop detector errors and statistical variations. Hence, a range of values can be defined in order to identify the start (out of range) or the termination (back in range) of an incident (Lin and Daganzo, 1997). Older methods for estimating the threshold, such as smoothed averages and exponential smoothing, will mostly be effective for incidents that occurred close to a detector (Thancanamootoo and Bell, 1988), (Han and May, 1989). Kalman filtering was another popular technique to estimate traffic flow, speed, and occupancy (Lee and Taylor, 1999), (Chen and Chang, 1993), but it would produce algorithms whose calibration was quite complex. There was a need for new approaches to accurately estimate the occupancy threshold value. While conventional solutions, which apply fixed threshold values for traffic occupancy, speed, and volume, suffer from high false alarm rates, modern solutions (Jeong et al., 2011; Teng and Qi, 2003; Asare et al., 2013), adapt their threshold values based on real-time traffic data. The multiresolution property of wavelet transforms can be used to vary the threshold values for incident detection (Jeong et al., 2011; Teng and Qi, 2003). Discrete wavelet transforms would be used to decompose traffic data into different resolution time elements in order to extract occupancy and speed data. This would produce threshold values that fluctuate in relation to the traffic measurements. The threshold can also be based on the level of demand on a roadway segment. Asare et al. (Asare et al., 2013) used the Hilbert-Huang transform to decompose traffic data, exploiting empirical mode decomposition (EMD) and Hilbert spectral analysis (HSA) to adaptively change the threshold to correspond to the current demand level. Both wavelet transforms and Hilbert-Huang transforms produce algorithms with the highest detection rate, lowest false alarm rate, and shortest mean time to detection, compared to existing fixed point solutions. Fixed point detectors are restricted to providing traffic data from limited fixed points in the roadway. Hence, it is difficult to use these data to realistically model the traffic conditions that are spatially continuous.

2.3 Probe-Based Systems

Probe-based sensors are usually mounted on individual vehicles that are mobile, and they provide data for a single point, pair of points, and sections. The reliability of probe data depends on the penetration rate, but it can measure a wider area of a roadway compared to fixed sensors. Floating cars can provide traffic measurements in real time with good spatial and temporal coverage at a very inexpensive cost. Probe data are provided by several third parties like Here.com, TrafficCast, etc. These sensors include GPS receivers, Bluetooth readers, toll transponders, e-tag readers, on-board units, smartphones, and any other in-vehicle telematics. Data collected include origin-destination data, timestamp, location, speed, direction, density, queue length, turning ratios, route preferences, mode sharing, etc.

The main assumption behind using probe car data is that probe vehicles that pass an incident would have a relatively higher total time, coefficient of speed variation, and travel time (Sermons and Koppelman, 1996; Balke et al., 1996; Mouskos et al., 1999). The travel time during an incident is very high when compared to non-incident conditions from comparable times of day and days of the week (Balke et al., 1996; Mouskos et al., 1999; Hellinga and Knapp, 2000; Li and McDonald, 2005). Furthermore, it has been shown that the travel time for a road section increases more quickly due to the capacity decrease caused by an incident as compared to increased demand (Hellinga and Knapp, 2000). Hence, it is believed that travel times before and after an incident should be viewed as belonging to two different populations. Probe car data comprising traffic measurements for a link are very useful if the incident occurs downstream from that same link, but not upstream (Sethi et al., 1995), and AID algorithms perform better when these probe car data measurements are more detailed (Sermons and Koppelman, 1996).

A high penetration rate is crucial for probe-data–based algorithms, which may perform worse than fixed-detector–based algorithms when the penetration rate is low [(Balke et al., 1996), (Mouskos et al., 1999)]. If the algorithm is based on mean travel time, then a 50% penetration is required for good performance (Cheu et al., 2002). Petty et al. (Petty et al., 1997) provide a model that can be useful in estimating an upper bound for the detection rate for probe vehicle density in a defined cell. They claim that only a single probe vehicle is required in a defined cell in order to detect the incident occurrence within that cell.

Probe data cover longer sections of the urban roadway than fixed detectors and are especially useful for detecting secondary incidents or non-recurrent congestion, which are likely to occur away from the primary incident (Yang et al., 2017), (Park and Haghani, 2016).

2.4 Human Reporting Systems

Human reporting systems provide anecdotal information from incident reports made directly by drivers, road crews, patrol units, and other travelers on the road. This type of incident detection is manual rather than automated, but it provides a very rich source of information because the incident can be described very clearly with regard to its location, time, type, and severity (Skabardonis et al., 1998; Walters et al., 1999). Driver-based reports over the cellular phone are usually the fastest mode of incident detection and require much less investment for operation and maintenance. However, these reports are difficult to process and could contain false reports due to mistakes or malice. (Mussa and Upchurch, 1999; Mussa and Upchurch, 2000) showed that an incident detection system using cell phone calls (or tag-based digital messaging) from drivers could have lower detection times and higher detection rates. However, they do not completely consider malicious incident reports. Sources include Twitter tweets, wireless or cellular phone reports, roadside call boxes, patrol crew, road crew, Waze, etc.

Twitter is a very inexpensive method for incident detection. However, it is very brittle due to dependence on the social media organization allowing access to the data. (Gu et al., 2016) presented methods to classify tweets. Tweets can be acquired in real time using Twitter's streaming REST API and using a

set of important keywords that could imply traffic incidents. They used this information to classify each tweet as an incident or non-incident and then categorized them by location. They claimed that there is more incident information on Twitter during the weekends and during the daytime. Between 60% and 70% is posted by influential users, and the tweets are usually about incidents that are close to the center of a city. Studies have been performed to determine the value of tweets in incident detection (Zhang and Ni, 2018) where selected features were used to extract both the individual and paired token features from Twitter's social media site. Gu et al. address several issues for detecting incidents from tweets, such as the language customs that Twitter users would frequently use to describe a traffic incident, the time, location, and user influence bias.

2.5 Incident Detection Algorithms

2.5.1 Threshold-Based Systems

Conventional incident detection algorithms use threshold values to determine if an incident has occurred. A system that uses overly precise threshold values cannot consistently classify incident vs. non-incident conditions in an accurate manner. This is due to the lack of quality (loss or approximation of information) in the available traffic data or even an overlapping membership in multiple sets, such as an incident spread spatiotemporally over multiple links (Yang et al., 2017). As a result, this ambiguity may cause lower detection rates and higher false alarm rates. Fuzzy logic can minimize error due to uncertainty in parameters by providing approximate reasoning instead of precise reasoning. It provides membership functions, with a range between 0 and 1, that represent the odds of membership in a particular set. After all calculations and computations are completed, if the cumulative membership value is greater than some threshold, the membership in that particular set holds true.

AID solutions using fuzzy logic are simpler to calibrate, maintain, and debug when compared to other AID algorithms (Lee et al., 1998; Chang and Wang, 1994; Hawas, 2007). In machine learning solutions (El Hatri and Boumhidi, 2018), fuzzy logic can be used to reduce errors, speed up convergence, and control the parameters or learning rate of the model in order to avoid overshooting during the training period. Chang and Wang (Chang and Wang, 1994) stress the necessity of automated learning techniques in incident detection. Hence, marrying machine learning with fuzzy logic would undoubtedly benefit AID algorithms by minimizing error due to uncertainty in data points.

2.5.2 Neural Networks

Artificial neural networks (ANNs) are computing systems that learn to make decisions in the absence of specific rules by considering examples instead. With respect to incident detection, Cheu and Ritchie (Cheu and Ritchie, 1995) found that multilayer feed-forward neural networks performed better than self-organizing feature maps (SOFM) and adaptive resonance theory model 2 (ART2). Neural networks (NN) have been used to estimate traffic volumes (Lingras and Adamo, 1996) and incident detection (Ivan and Sethi, 1998). However, constructive probabilistic neural networks (CPNN) can use a smaller network to achieve the same results (Jin et al., 2002). (Ivan and Sethi, 1998) focused on identifying

the factors that would predict the occurrence of traffic incidents. They showed that by adjusting the algorithm's inputs to avoid false alarms, neural networks can perform better than discriminant analysis models. Furthermore, their results indicated that additional inputs, like historical input-output data of the algorithm, along with conditions of adjacent links, would further improve detection rates and false alarms. Hence, they trained and connected three more neural networks providing these data. They found significant performance improvement with the addition of each extra type of input data, with the full network demonstrating the best performance. Their results suggest that a time series is beneficial to the solution. The performance of neural networks can also be improved by careful selection of parameters. For example, (Roy and Abdulhai, 2003) use genetic algorithms to optimize the selection of parameters in a probabilistic neural network (PNN) and achieve lower detection rates while experiencing fewer false alarms. Similarly, fuzzy logic (El Hatri and Boumhidi, 2018) has also been used for controlling the parameters in neural networks. Dia and Rose (Dia and Rose, 1997) used a multilayer perceptron neural network to detect incidents. The authors also evaluated the relevance of data quality with respect to the model's performance. For detecting traffic incidents or even arterial operational problems, neural networks generally perform better than other types of classifiers (Ivan and Sethi, 1998; Khan and Ritchie, 1998) except possibly support vector machines(SVM).

SVM is suitable for finding the global best solution when dealing with small sample data and non-linear but high-dimensional problems and has been found to be a good machine learning technique for detecting traffic incidents (Yuan and Cheu, 2003). SVM for incident detection has a higher detection rate, lower false alarm rate, and shorter average detection time than multilayer feed-forward neural networks (Hawas, 2007). The Ant Colony Algorithm (ACA) has been used for parameter selection that is crucial for its learning accuracy and generalization ability (Liang, 2015; Ivan and Sethi, 1998). (Chen et al., 2009) combined an ensemble of SVM classifiers for traffic incident detection, and in doing so, they improved on the basic SVM and overcame the crucial task of SVM parameter selection.

2.5.3 Unsupervised Learning

Unsupervised learning is proving useful to detect incidents that are not well defined, such as arterial operational problems (Han and May, 1989), and secondary crashes or congestion (Anbaroglu et al., 2014). (Yang et al., 2017) used probe car data to identify secondary crashes or non-recurrent congestion. They reduced duplicate reports, or false alarms, by using unsupervised learning via fuzzy C-means clustering (i.e., K-means clustering with fuzzy logic) to detect the impact area of a primary incident. Finally, they use meta-heuristics optimization (genetic algorithms or ant colony optimization) to minimize the boundary of an impact area that could contain secondary crashes or non-recurrent congestion. (Anbaroglu et al., 2014) use spatiotemporal clustering to identify significantly large link-journey times across single or adjacent links. Clustering of spatiotemporally overlapping episodes can be used to detect non-recurrent congestion that spans multiple links. The cumulative sum (CUSUM) algorithm detects incidents in arterial roads by applying the CUSUM chart in order to identify incident-based changes in traffic measurements that tend to linger for some time under incident-free conditions. This is an early form of unsupervised learning-based anomaly detection for traffic incidents.

Modern forms of unsupervised anomaly detection (Kinoshita et al., 2014; Zhu et al., 2009) of incidents use probe car data. (Kinoshita et al., 2014) claimed that traffic congestion may be a chronic condition in some roadways and may not always be due to incidents. Therefore, they attempted to distinguish between incidents and transient congestion in the transportation network by using a probability model with maximum-likelihood parameters and an expectation maximization algorithm. (Zhu et al., 2009) applied a distance-based outlier mining method in order to identify incidents in arterial roads. They used fluctuations in the travel speed and chose speed differences in nearby sections and adjacent intervals for spatiotemporal analysis.

Arterial operational problems provide yet another form of incident (Han and May, 1989; Khan and Ritchie, 1998). Operational problems include detector malfunctioning, signal malfunctioning, lane-blocking incidents, etc. Due to the lack of a well-defined training set for these types of problems, unsupervised learning or other innovative schemes must be used. (Khan and Ritchie, 1998) used neural network classifiers in a modular architecture.

Lastly, Ren et al. (Ren et al., 2012) attempted to improve AID by enhancing the feature representation of incidents via an unsupervised feature-learning algorithm which produced higher level features to describe incidents. This allowed the use of unlabeled data or a reduced amount of required labeled data in case of a semi-supervised approach.

2.6 Secondary Incidents and Non-Recurrent Congestion

Primary incidents are usually assumed to cause secondary incidents. Hence, detection of an incident could necessitate detection of such incidents (Yang et al., 2017; Park and Haghani, 2016). (Anbaroglu et al., 2014) presented a non-recurrent congestion (NRC) event detection method using spatiotemporal clustering to detect excessive link-journey times across single or multiple-adjacent link(s). Adjacent links with non-recurrent congestion are clustered together in one episode if they contain at least one common time interval in their duration. A spatiotemporal clustering using overlapping episodes can be used to detect secondary incidents that span multiple links. The frequency of secondary incidents is directly related to the length of primary incidents (Park and Haghani, 2016). (Park and Haghani, 2016) used GPS probe car data to infer the existence of secondary incidents and then utilized a Gaussian mixture model (GMM) to determine the reference speed for classification of such events.

2.7 Data Fusion Techniques

Data fusion is an interdisciplinary method that provides techniques to combine data from several sources in order to achieve synergy with respect to the resultant information derived. The JDL model developed by the U.S. Department of Defense is the most popular work on data fusion in general, providing five levels for processing system-level information. In Intelligent Transportation Systems (ITS), each data source brings with it a new challenge. Fixed detectors have low area coverage and accuracy, while probe cars have a low penetration rate, and driver reports are very sparse. However, each new data source also provides us with a new advantage. Since the advancement of road and vehicle

telematics, multiple sources of data provide multiple views into traffic conditions. This helps reduce the vagueness and incompleteness characteristic of individual data sources and, thereby allows us to use data fusion in order to augment our interpretation of observed traffic measurements. It is hypothesized that this fusion would result in performance improvement of the incident detection system.

Existing data fusion techniques (El Faouzi and Klein, 2016) for ITS applications are listed here:

1. Bayesian inference
2. Dempster-Shafer evidential theory
3. Artificial neural networks
4. Fuzzy logic
5. Knowledge-based expert systems
6. Particle filtering and Kalman or Extended-Kalman filtering
7. Monte Carlo techniques

The earliest works on data fusion for ITS each proposed data fusion frameworks to solve issues in traffic or transportation using travel time estimation (Sethi et al., 1995) or incident detection (El Faouzi and Lesort, 1995). In the case of incomplete data, which is insufficient for automatic incident detection, it is recommended to fuse the sensor data or fuse the outputs from incident detection algorithms (Mahmassani et al., 2014).

Sethi et al. (Sethi et al., 1995) proposed using fixed detector data and probe car data separately in two different algorithms, as part of the ADVANCE ITS, and then using discriminant analysis to estimate incidents. They intended to study how each type of data could complement the other, and they concluded that traffic measurements at a particular location were useful for detecting incidents that occurred downstream. They inferred that the fixed detector algorithm performed better than the probe vehicle algorithm and further observed that the benefit from fixed detector data was dependent on the number of instrumented links in the transportation network, while the benefits of probe car data were limited by reporting rate. Subsequently, Ivan (Ivan et al., 1995) combined the outputs of the two algorithms into a fusion algorithm and developed an AID using surveillance data. Neural networks were used in two ways: (1) algorithm output fusion, which uses a neural network to fuse the separate incident-likelihood scores that were output from the fixed detector algorithm and probe vehicle algorithm (Figure 2) and (2) integrated fusion, which fuses the raw data from both sources and determines the occurrence of an incident. It accomplishes this by combining the functions of fusion and single-source algorithms into a single feed-forward neural network. In effect, this integrated fusion attempts to detect incidents by using both data sources simultaneously, but not separately. The raw data consists of (1) volume and occupancy (from fixed detectors) and (2) travel time (from probe cars). The authors show that using data from both sources improved the AID compared to using data from any

single source alone. Consequently, (Ivan, 1997) experimented with this idea on the algorithm output fusion module and demonstrated considerable improvements in incident detection. Lastly, (Ivan and Sethi, 1998) compared the performance of automatic incident detection on signalized arterial streets by fusing data via discriminant analysis and neural networks. Although Ivan conceded to using driver-based reports as an anecdotal data source for manual manipulation of the algorithm, he concluded that neural networks performed better than discriminant analysis.

With respect to the ADVANCE ITS, (Bhandari et al., 1995) also proposed an automatic incident detection system for arterial streets, leveraging three separate sources: fixed detector data, probe vehicle data, and anecdotal reports. They used three independent modules to preprocess the data from each source separately. Their fusion process first uses discriminant analysis to fuse the pre-processed results of fixed detector data and probe car data. Next, the process fuses the resulting output with the output of the anecdotal algorithm. (Thomas and Dia, 2005; Dia and Thomas, 2011) used neural networks to combine simulated loop detector data and simulated probe car data for incident detection on arterials. They compared several neural network architectures by varying probe vehicle penetration rates and detector configurations. A multilayer feed-forward neural network allowed inclusion of historical data, which further improved the results, as did the addition of speed data. (Yin et al., 2006) used toll data for automatic incident detection.

Dempster-Shafer evidential theory is used for data fusion in order to detect incidents as well as other traffic and transportation events (Klein, 2000; Klein et al., 2002). (Byun et al., 1999) attempted to solve traffic congestion problems in links having limited capacity by assigning weights to incident reports, thereby identifying true incidents and false incidents, i.e., possible false alarms. They ran an incident detection algorithm on three data sources separately: loop detectors, CCD cameras, and probe vehicles. Dempster-Shafer's algorithm was used to fuse the three results and increase the accuracy of traffic-condition reports. (Klein, 2000; Klein et al., 2002) used the Dempster-Shafer algorithm to combine data from three different sources in order to detect and verify incidents and other traffic events, especially when the individual data sources cannot determine the occurrence of an event with full confidence. The author provided an example where all available data are combined using Dempster-Shafer's rule, allowing them to distinguish the most probable event. (Zeng et al., 2008) claimed that raw data from individual sources may be corrupted with unexpected missing values. They demonstrate creating an SVM classifier for each of three different sources of traffic data: loop inductive detector, AVI (automated vehicle identification) observation, and floating car data (FCD). They use Dempster-Shafer's evidential theory to combine the multiple SVM classifiers.

Bayesian inference has been used to detect incidents via a multiple attributes decision-making view. (Thomas, 1996) combined occupancies and volumes reported by induction loop detectors along with probe car data. Observations showed that algorithms based on fixed-detector data performed well. While probe car data alone suffered from an excess of overlaps in class distributions, they complemented fixed-detector data quite well. Next, (Thomas, 1998) proposed an incident detection approach based on two characteristics: (1) the ability to classify arterial traffic into several states based on the type of sensor (fixed detectors or probe-cars) and (2) use of a multivariable vector (as opposed to a

scalar) as the discrimination criterion for an incident. The algorithm was tested using data from a modified INTRAS simulation. The types of data included probe car travel time, number of probe reports, lane-specific detector occupancies, and vehicle counts. This approach uses multivariate classifiers to distinguish multiple traffic states on arterials.

Cohen [76] used three schemes to fuse the output of multiple incident detection algorithms. He used logical aggregation, neural network fusion, and a veto procedure. His validation step used real-world data and demonstrated that the logical aggregation and veto procedure resulted in considerable performance improvement when compared to any single-source detection algorithm alone. (Westerman et al., 1996) performed incident detection and estimates travel time by allowing a loop detector algorithm and a probe data algorithm to operate separately, while simultaneously complementing each other.

The two stages of the algorithms (Westerman et al., 1996) is as follows:

1. a trigger stage which suggests the occurrence of an incident

2. a verification stage that automatically verifies the occurrence.

The probabilities of the incident's occurrence from each algorithm are combined using a weight-averaging fusion method in order to reach a final classification decision in the verification stage. The weight of each component is determined by the number of verification steps that were executed. DubSTAR (Daly et al., 2013)] merges the traditional data sources, like loop detectors and traffic cameras, with social media and SMS messages in order to provide insights into real-time traffic conditions and incidents.

2.8 Summary

We have leveraged the following observations from the literature analysis in our work:

1. Minimizing the boundary of the region possibly containing the incident before executing the detection algorithm could reduce performance time and may increase accuracy. It could also reduce false alarms (Yang et al., 2017; Kamran and Haas, 2007).

2. Spatiotemporal analysis, i.e., considering space and time dimensions, could lower false alarm rates (FAR) in AID algorithms (Sermons and Koppelman, 1996; El Hatri and Boumhidi, 2018; Zhu et al., 2009; Anbaroglu et al., 2014; D'Andrea and Marcelloni, 2017).

3. For anomaly detection, model a stable state for the road's traffic before detecting anomalies against that stable state (Kinoshita et al., 2014).

4. Selection of the machine-learning model's parameters should be prioritized as one of the critical steps in the design of the solution (Liang, 2015; El Hatri and Boumhidi, 2018; Jeong et al., 2011; Chen et al., 2009).

5. Unsupervised learning is valuable in detecting events that have no concrete definitions, such as arterial operation problems (Han and May, 1989; Khan and Ritchie, 1998), congestion and secondary incidents (Yang et al., 2017; Anbaroglu et al., 2014) etc.

6. No fusion method always performs better than the other. Instead, each fusion method excels with data from a particular set of sensor types.

3 Datasets and Preprocessing

The main objective of this section is to determine the appropriate data, acquire it, and develop methods for preprocessing it. Based on discussion with the project managers, we decided to focus on three datasets:

1. High resolution sensor data
2. HERE.COM
3. Signal Four Analytics (SFA)

The choice was based on three criteria: availability of large amount of data, high frequency of update for potential use in real-time, and their potential effectiveness in predicting incidents. Traffic data are full of noise due to various known and unknown factors. Cleaning and verifying the data were performed and stored in a database running machine learning simulations. Based on availability of data and input from the sponsor, we decided to focus on 329 signalized intersections from Seminole County, Florida„ in FDOT District 5 (Figure 1). The initial data ranges for dates between May 1, 2016, and Sept. 16, 2016, for high resolution sensor data and Here.com. The total storage space occupied is 400 GB and consists of several comma-separated value files. We also received these data for the late 2018 and early 2019 (700 GB).

3.1 Datasets

3.1.1 High Resolution Loop Detector Data

Automated traffic signal performance measures (ATSPM) [1] is used to process data from high resolution sensor data provided at intersection signal controllers. Induction loop detectors attached to the intersection collect data at 10 Hz, indicating whether a vehicle passed over it or not. Signal behavior is also captured. These data allow traffic engineers to analyze the performance of traffic intersections and improve safety and efficiency while cutting costs and congestion. The data consist of the following:

1. Raw data files
2. Intersection metadata (additional tables)
3. Data logging requirements file

Raw Data Files The raw data table has nine months of raw data (700 GB). This table is indexed by signal ID and timestamp for fast retrieval. The table has 4 columns: SignalID, Timestamp,EventCode: What event at the signal was captured and EventParam: What was the value of the event or attribute at that timestamp. Table 1 shows sample data from the raw table. The data consist of four columns:

1. SignalID: Intersection identifier

Table 1: *Raw event logs from signal controllers. Most modern controllers generate this data at a frequency of 10 Hz.*

SignalID	Timestamp	EventCode	EventParam
1490	2018-08-01 00:00:00.000100	82	3
1490	2018-08-01 00:00:00.000300	82	8
1490	2018-08-01 00:00:00.000300	0	2
1490	2018-08-01 00:00:00.000300	0	6
1490	2018-08-01 00:00:00.000300	46	1
1490	2018-08-01 00:00:00.000300	46	2
1490	2018-08-01 00:00:00.000300	46	3
1490	2018-08-01 00:00:00.000300	46	4
1490	2018-08-01 00:00:00.000300	46	5
1490	2018-08-01 00:00:00.000300	46	6
1490	2018-08-01 00:00:00.000300	46	7
1490	2018-08-01 00:00:00.000300	46	8

2. Timestamp: Time at which event was logged (deci second resolution)

3. EventCode: What event at the signal was captured

4. EventParam: What was the value of the event or attribute at that timestamp

Intersection Metadata Intersection metadata are available in high resolution sensor data consisting of the following tables. All the additional tables are joined into a single table that gives compact representation of all the metadata. Table 2 shows a sample of a joined table with select attributes, obtained after combining all the metadata tables

1. “Approaches” table describes the various approaches per signal.

2. “Signals” table gives physical coordinates of the signal.

3. Detectors” table describes individual detectors, and corresponding signals.

Data Logging Requirements File This file contains event codes and their descriptive data (Table 3). Tables explaining all the different event codes, event parameters is included in the appendix.

Table 2: *Join table with selected, key attributes after combining all the intersection metadata tables*

Detector ID	Approach ID	Primary Name	Signal ID	Latitude	Longitude
100504	8438	US17-92	1005	28.833835	-81.323649
100505	8438	US17-92	1005	28.833835	-81.323649
100510	8439	US17-92	1005	28.833835	-81.323649
100511	8439	US17-92	1005	28.833835	-81.323649
101502	8440	US17-92	1015	28.811685	-81.273174
101507	8441	US17-92	1015	28.811685	-81.273174
101508	8441	US17-92	1015	28.811685	-81.273174
102002	8442	US17-92	1020	28.809877	-81.273161
102003	8442	US17-92	1020	28.809877	-81.273161
102008	8443	US17-92	1020	28.809877	-81.273161

Table 3: *Sample data-logging requirements table*

Code Explanation	Parameter	Code
Phase On	Phase # (1-16)	0
Phase Begin Green	Phase # (1-16)	1
Phase Check	Phase # (1-16)	2
Phase Gap Out	Phase # (1-16)	4
Phase Max Out	Phase # (1-16)	5
Detector Off	Det Channel # (1-64)	81
Detector On	Det Channel # (1-64)	82

3.1.2 HERE.COM

HERE.com [2] is a company based in Europe that provides mapping and location services for road traffic. HERE.com captures both static content such as road networks as well as dynamic content such as traffic flows. HERE.com collects average speed of a part of the road network (called TMC, based on OpenStreetMap designations of road portions) by using streaming data from vehicles using HERE.com GPS and routing services. By aggregating this data from various vehicles, it can calculate an average speed for that link at a one-minute resolution. The HERE.com dataset consists of data for Seminole County, Greater Orlando Metropolitan Area, and tracks 1,009 TMCs. Data since 2016 are available and can be queried for, by a Web API. The dataset consists of:

TMC Identification File This file has details about the various TMCs, including their OpenStreetMap identification code and their start and end positions. This file helps in identifying the location of the TMC.

Seminole County Data File This file contains the actual data. For every TMC, at a one-minute resolution, the average reported speed is recorded. The reference speed is the desired speed that should be expected at the link (and not the maximum speed). Travel time indicates how long the vehicles took to cross the link. Confidence indicates the quality of data collection.

Table 4: *Sample HERE.COM raw data table which has speed, reference speed, and travel time for a particular link*

Timestamp	speed	reference _speed	travel_time	confidence	tmc_code
2018-08-01 00:00:00	67.5	55.0	83.235	0.765	102+04807
2018-08-01 00:02:00	66.0	55.0	85.120	0.740	102+04807
2018-08-01 00:04:00	66.0	55.0	85.120	0.730	102+04807
2018-08-01 00:06:00	65.0	55.0	86.430	0.750	102+04807
2018-08-01 00:08:00	65.0	55.0	86.430	0.730	102+04807
2018-08-01 00:10:00	68.0	55.0	82.690	0.835	102+04807
2018-08-01 00:12:00	74.5	55.0	75.415	0.870	102+04807
2018-08-01 00:14:00	71.0	55.0	79.190	0.855	102+04807
2018-08-01 00:16:00	68.0	55.0	82.620	0.845	102+04807
2018-08-01 00:18:00	61.0	55.0	92.100	0.885	102+04807

3.1.3 Signal Four Analytics

The data on crashes are collected from digitized police crash reports which are housed at Signal Four Analytics (SFA), which contains all traffic crashes reported in the state of Florida since 2006. For this study, we extract the crashes for Seminole County for the months of Nov. 2018 to Apr. 2019. Figure 2 shows a table which has a sample of crash reports. These reports have the following important attributes: location of crash (geocoordinates), time of crash, and severity of crash.

3.2 Processing and Storage Requirements

The high resolution sensor dataset and HERE.com dataset contain data at one decisecond and one-minute resolutions, respectively. This leads to a very large dataset. While the technology allowed for the collection and storage of such high-resolution data, such resolution is not required for our project. Hence, it is important to resample the data at useful timescales. Python multiprocessing packages and Pandas Python data processing library are used to process data for several intersections simultaneously in a multicore machine. Time requirement is approximately 5 min for one week of data, 300 intersections on a machine with 54 cores, and 256-GB memory. We use the following libraries and software for data processing:

- **Numpy:** NumPy is a library for the Python programming language, adding support for large, multidimensional arrays and matrices, along with a large collection of high level mathematical functions to operate on these arrays.

- **Pandas:** Pandas is an open source, BSD-licensed library providing high performance, easy-to-

HSMV_Report_Number	Agency_Report_Number	Reporting_Agency	Form_Type	Crash_Date	Crash_Time	City	County	Crash_Street	Intersecting_Street
88801272	2019TA001249	Oviedo PD	Long	01/30/2019	10:34 AM	Oviedo	Seminole	ALAFAYA TRL	
88801271	2019TA001218	Oviedo PD	Short	01/29/2019	02:52 PM	Oviedo	Seminole	LOCKWOOD BLVD	
88801270	2019TA001209	Oviedo PD	Long	01/29/2019	12:24 PM	Oviedo	Seminole	LOCKWOOD BLVD	BIG OAKS BLVD
88801269	2019TA001207	Oviedo PD	Short	01/29/2019	11:37 AM	Oviedo	Seminole	W MITCHELL HAMMOCK RD	SHARON CT
87773259	2019TA001258	Longwood PD	Long	01/30/2019	01:59 PM	Longwood	Seminole	N RONALD REAGAN BLVD	E CHURCH AVE
87773255	2019TA000917	Longwood PD	Long	01/22/2019	03:04 PM	Longwood	Seminole	E SR-434	MYRTLE ST

Vehicle_Dmg_Amt	S4_Mapping	S4_Decimal_Degree_Longitude	S4_Decimal_Degree_Latitude	S4_Albers_X	S4_Albers_Y	S4_Mapping_Date
7000	Mapped, Not On Network	-81.2092920699405	28.652563758939	672252.6964969510	519347.7409632880	1/31/2019 4:59:44 PM
2500	Mapped, Not On Network	-81.1797225507302	28.6580076214682	675123.0799309910	520017.7784222180	1/31/2019 4:59:44 PM
15000	Mapped, On Network	-81.1793178522321	28.6534562013973	675174.1128556370	519513.3571527360	1/31/2019 4:59:44 PM
2000	Mapped, On Network	-81.2103461487269	28.6556622156796	672142.0978081270	519689.42433331	1/31/2019 4:59:44 PM
500	Mapped, On Network	-81.3462549448013	28.7005400019544	658778.4948583440	524379.5140340440	1/31/2019 4:59:44 PM
1000	Mapped, On Network	-81.3439794501554	28.6978582256081	659006.7683515040	524086.5380577910	1/31/2019 4:59:45 PM

Estimated_Damages	Weather_Condition	Light_Condition	Street_Number	Crash_Type_Detailed	Crash_Type_Dir	Crash_Severity	Within_City_Limits	Manner_of_Collision
$7,000.00	Cloudy	Daylight	1268	Backed Into	W	Property Damage Only	Y	Angle
$2,500.00	Clear	Daylight	1016	Backed Into	E	Property Damage Only	Y	Rear to Side
$15,000.00	Clear	Daylight		Rear End	S	Property Damage Only	Y	Front to Rear
$2,000.00	Clear	Daylight		Rear End	E	Property Damage Only	Y	Front to Rear
$800.00	Cloudy	Daylight		Pedestrian		Property Damage Only	Y	
$1,000.00	Clear	Daylight		Rear End	W	Property Damage Only	Y	Front to Rear

Figure 2: *SFA crash reports showing selected key attributes*

use data structures and data analysis tools for the Python programming language.

- **Postgres:** PostgreSQL, also known as Postgres, is a free and open source relational database management system (RDBMS) emphasizing extensibility and technical standards compliance. It is designed to handle a range of workloads, from single machines to data warehouses or Web services with many concurrent users.

- **Python Multiprocessing:** This package supports spawning processes using an API similar to a threading module. We use this package to make use of each core in a multicore machine to process data for an intersection.

- **Tensorflow, Keras**: These packages are open source neural network libraries written in Python. These provide high level APIs for building deep learning models.

- **Scikit-Learn:** This is a Python library used for data mining and data analysis.

We cleaned the dataset by

1. Removing intervals of data where detectors are broken or not reporting any data for a significant amount of time on some days

2. Removing intervals of data where cycle length is less than a second

The high resolution sensor dataset was resampled at fixed buckets or based on cycle length. For 8 am to 8 pm, the intervals correspond to a cycle or a fixed size interval. For 8 pm to 8 am, the interval size is fixed to the median of cycle lengths from 8 am to 8 pm (because cycle lengths during the overnight period are significantly large and erratic). There are multiple phases in each cycle. The duration of each phase as well as the cycle vary slightly from one cycle to the next. We expect a cycle-based approach to be more predictive as it maintains the consistency of phases over time. The HERE.com

Table 5: *Processed representation of the raw data from Figure 1. We compute arrival volumes using Algorithm 1. Different attributes include (a)Timestamp: start of green time, (b) Count: number of arrivals in each cycle, (c) CL: cycle length in seconds, (d) AR: arrival rate, (e) D-ID: detector ID, (f) S-ID: signal, ID, and (g) Ph: phase.*

Timestamp	Count	CL	AR	D-ID	S-ID	A-ID	Ph
2018-08-01 00:00:00	5.0	210.0	0.0238	149016	1490	8612	6
2018-08-01 00:00:00	6.0	210.0	0.0285	149006	1490	8611	2
2018-08-01 00:00:00	4.0	210.0	0.0190	149007	1490	8611	2
2018-08-01 00:00:00	6.0	210.0	0.0285	149008	1490	8611	2
2018-08-01 00:00:00	7.0	210.0	0.0333	149018	1490	8612	6
2018-08-01 00:00:00	8.0	210.0	0.0380	149017	1490	8612	6
2018-08-01 00:00:00	6.0	210.0	0.0285	149009	1490	8611	2
2018-08-01 00:03:30	9.0	210.0	0.0428	149018	1490	8612	6
2018-08-01 00:03:30	9.0	210.0	0.0428	149006	1490	8611	2
2018-08-01 00:03:30	2.0	210.0	0.0095	149016	1490	8612	6
2018-08-01 00:03:30	5.0	210.0	0.0238	149017	1490	8612	6

dataset was resampled at 1-min and 2-min buckets. The controller log data are used to construct arrival volume time series aggregated over each cycle. Flow counts per lane were calculated for every approach direction of the intersection. Along with arrival volumes, a time series of cycle lengths for every approach direction of the intersection is calculated. Data are stored for intervals that are of fixed duration or based on cycles. For HERE.com data, travel times and average speed over 2-min bins are calculated. Congestion ratio is also calculated, which is the ratio of recorded speed to the reference speed for a link. A link with congestion ratio of 0.70 or below was deemed to be congested. The processed data are written out and stored on the MySQL database for further use. We use these time series data to label the data to find potential incident intervals.

Algorithm 1 Construct time series of arrival volumes

```
1: function CONSTRUCTTIMESERIES(rawdata , I, ph, time)
2: Require: rawdata - High resolution controller logs of the intersections being studied.
3: I - Intersection ID ph -phase time - time interval
4:     for every intersection in I do
5:         Fetch high resolution sensor data from database for I, time, ph
6:         Filter Event Code 81,82,2 and get the corresponding time stamps
7:         Aggregate volumes between two successive timestamps where EventCode==2.
8:         Push arrival volumes time series to the database.
9:     end for
10: end function
```

Algorithm 1 presents our approach for constructing arrival volume time series from controller logs. Table 5 shows a sample of a processed time series data with the following attributes: Timestamp (timing pattern start time), Count (number of arrivals in this cycle), and CL (cycle length in seconds). Figure 8 shows a plot of arrival volume time series for five days and highlights the periodic nature of the data.

Parallel and Distributed Data Processing System *"Elixir is a dynamic, functional language designed for building scalable and maintainable applications"*. Elixir builds on the the Erlang VM, which

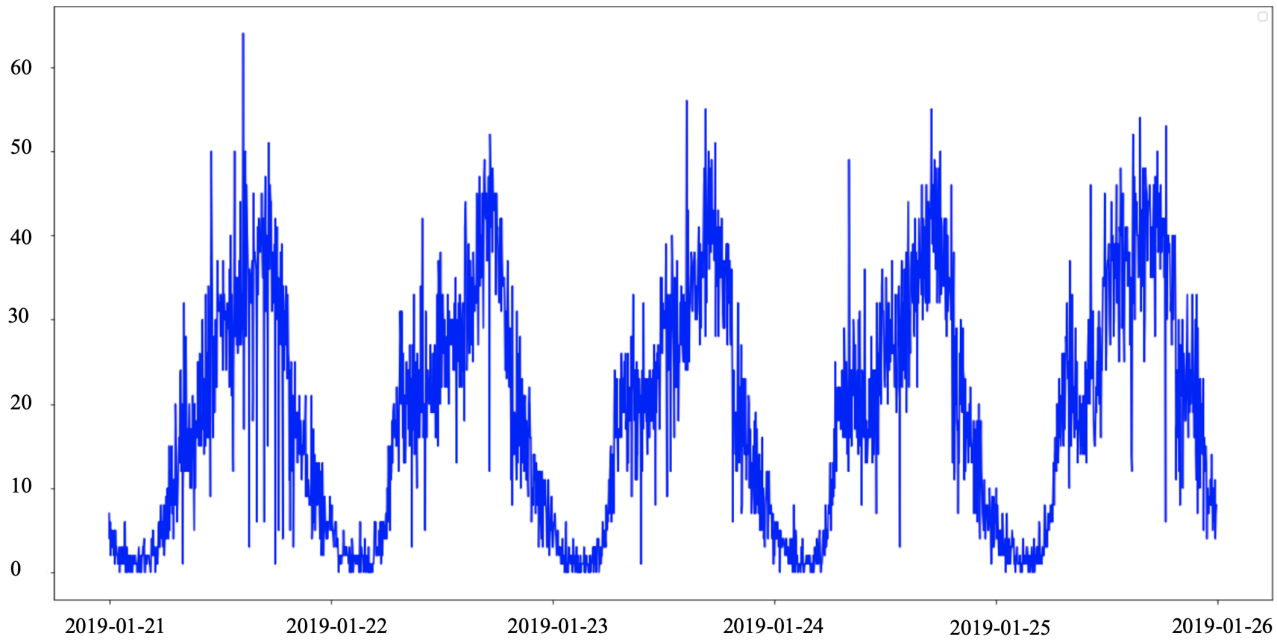

Figure 3: *Time series plot of vehicle arrival volumes for 5 days, computed using Algorithm 1. Note the periodic nature of the data.*

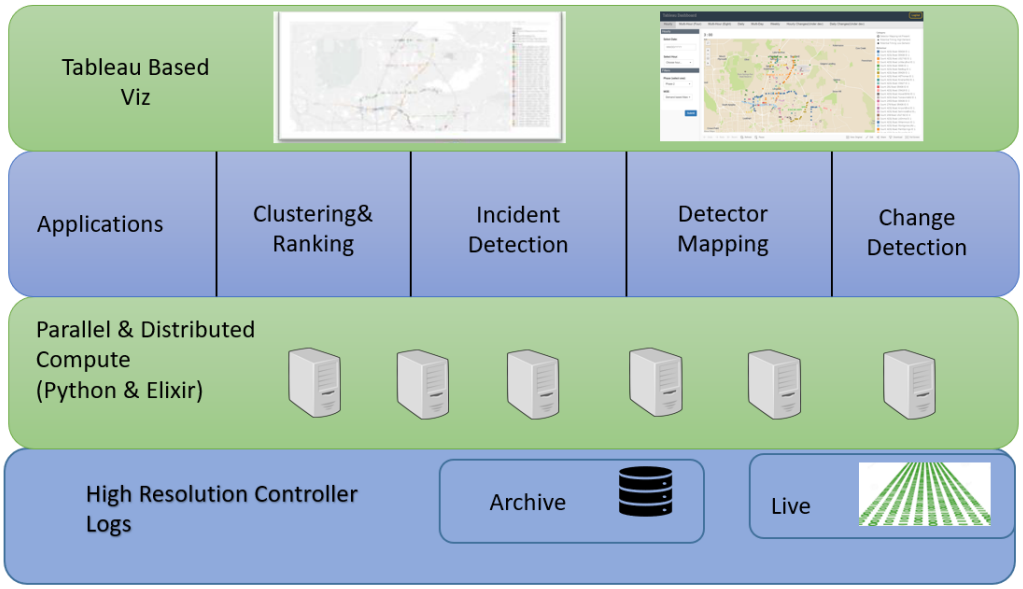

Figure 4: *Overall architecture for processing controller logs in a parallel and distributed manner*

Table 6: *Observed performance for different applications observed on a 50-core machine, varying the number of intersections*

Number of intersections	Time duration of data	MOE calculations	Classification	Incident detection
20	1 week (10,000 min)	1.5 min	<1 min	<2 min
300	1 week (10,000 min)	25 min	<15 min	<15 min

is known for three key reasons: low-latency execution, distributed computing, and fault-tolerant computing. The Erlang VM (and Elixir) is efficient for applications in particular domains and has been widely used in mission-critical networking equipment, but it doesn't perform well for operations like data processing and computation-intensive processes. Python, on the other hand, has a rich set of packages for tackling data processing. The Erlang VM allows interoperability with other programming languages using the Erlang port protocol. Hence, we use a combination of Python and Elixir to build our distributed and fault-tolerant computation architecture, which is shown in Figure 4.

3.3 Data Fusion

3.3.1 Linking Sensor Data and HERE.com Dataset

The key difference between the two datasets is that the high resolution sensor dataset captures readings from the point of view of an intersection, whereas the HERE.com dataset captures readings from the point of view of unidirectional road segments. They use different naming and ordering schemes. However, both datasets contain geolocation coordinates (latitude and longitude): high resolution sensor data contain the lat-long for the center point of the intersection, whereas the HERE.com data contain start and end lat-long coordinates (with traffic flowing from start to end). We used this fact to link the two datasets together.

We start by filtering out road segments that are too small (e.g., below 0.25 miles) in the HERE.com data. Then for the remaining, we find intersection coordinates that are the closest to the start points. Similarly, we find intersection coordinates that are the closest to the end points. We also keep a track of the distance errors, as the start and end points of the segments may not lie directly on the intersections. We then remove all segments that have a large error (e.g., above 0.2 miles). In this manner, we have an approximate but usable linking of datasets. Using this approach, we can obtain an adjacency matrix for the intersections, with the names of the road links that are connected to each other. We apply the same algorithms for road links. Further, we can also build a derivable adjacency list in which we determine which road links could be a part of a coherent vehicle path, without any U-turns.

Table 7: *Mapping of TMC link ID of HERE.COM link to start and end intersection IDs of ground sensor data. Different attributes include (a) tmc: TMC link ID of HERE.COM, (b) start_Nearest: intersection at start point of the road segment, and (c) end_Nearest: intersection at end point of the road segment.*

tmc	start_Coord	end_Coord	start_Nearest	end_Nearest
102+21238	POINT (28.7797 - 81.29693)	POINT (28.7708 - 81.2832)	2730	1065
102+11013	POINT (28.7564 - 81.3220)	POINT (28.78642 -81.3181)	2055	1985
102+21122	POINT (28.81171 -81.3225)	POINT (28.82829 -81.3228)	1685	2700
102+21121	POINT (28.80391 -81.32272)	POINT (28.81145 -81.3227)	2835	1685
102+21120	POINT (28.7870 - 81.3164)	POINT (28.80391 -81.3227)	1985	2835
102-21119	POINT (28.8039 - 81.3227)	POINT (28.78709 -81.3164)	2835	1985
102-21121	POINT (28.8282 - 81.3228)	POINT (28.81171 -81.3227)	2700	1685
102-21120	POINT (28.8114 - 81.3227)	POINT (28.80391 -81.3227)	1685	2835
102-11012	POINT (28.7864 - 81.3181)	POINT (28.7564 - 81.3220)	1985	2055
102+10992	POINT (28.6978 - 81.3462)	POINT (28.70085 -81.3463)	1295	1925

Figure 5: *Figure showing location of crash along with locations of derived nearby intersections. Red marker: location of crash; Blue marker: location of nearby intersections.*

3.3.2 Linking Sensor Data with Signal Four Analytics Crash Data

Crash reports from Signal Four Analytics (SFA) does not provide any information of nearby intersections at which a crash has occurred. We use the following attributes from these reports to link crashes to intersection metadata of ground sensor data, i.e., (a) Location of the crash (S4_Decimal_Degree_Latitude, S4_Decimal_Degree_Longitude) and (b) Crash_Street – Street name where the crash was reported.

For every crash, we calculate distance from location of crash to all the intersections and obtain four nearby intersections. We can further filter this to only get intersections within some distance of the location of crash. Table 8 shows this mapping for some of the crashes where we get intersections only within 600 m of the crash. Figure 5 shows an example of location of crash and locations of derived nearby intersections.

In some cases, as shown in Figure 6, a crash can happen on a side street, which may not affect the traffic on nearby intersections (even though the intersections are close by) and therby not impact the traffic significantly. So we use the street name from the SFA reports and intersection metadata to filter these types of crashes. The two key challenges we need to address is that the street names from these two data sources differ, and determining minor streets on major streets is not straightforward.

Figure 7 shows a brief outline of heuristic approach we use for getting crashes that happen on major streets. For each crash, we compare the street name with all the unique street names from intersection metadata of high resolution controller logs and get the street name with which it shares the longest common substring. Table 9 shows mappings of street names from crash reports to high resolution sensor data street names when length of maximum common substring is greater than 5.

3.4 Data Visualization

We visualize the parsed data by using Matplotlib [7] Python library. Matplotlib is a Python 2D plotting library that supports a variety of hardcopy formats and interactive environments. The aggregated vehicle count from high resolution sensor data data is visualized as a time series as shown in Figure 8. We developed a plotting tool which plots all the arrivals on one detector (or a subset of detectors)

Table 8: *Table showing mapping of each crash to the nearby intersections. Different attributes include (a) HSMV_No: unique identifier of crash from SFA reports, (b) location: location of the crash, (c) C_Date: crash date, (d) C_Time: crash time, (e) Crash_Severity: severity level of crash, and (f) near_sig: intersections within 600 m of crash.*

HSMV_No	location	C_Date	C_Time	C_Severity	near_sig
88801272	POINT (-81.20929 28.65256)	01/30/2019	10:34 AM	Property Damage Only	[1395, 1400, 2627]
88801271	POINT (-81.17972 28.65801)	01/29/2019	02:52 PM	Property Damage Only	[1870]
88801270	POINT (-81.17932 28.65346)	01/29/2019	12:24 PM	Property Damage Only	[1870]
88958477	POINT (-81.32735 28.69639)	02/11/2019	02:15 PM	Injury	[1130, 1127]
87773274	POINT (-81.35851 28.71469)	02/08/2019	05:43 PM	Injury	[1285]
88702737	POINT (-81.36583 28.64555)	02/08/2019	03:20 PM	Injury	[1555]
88614479	POINT (-81.32808 28.65128)	01/07/2019	09:15 AM	Property Damage Only	[1595, 1600]
88907150	POINT (-81.33461 28.75636)	01/08/2019	05:45 PM	Property Damage Only	[2065, 2125]
88907146	POINT (-81.34737 28.77745)	01/07/2019	06:45 PM	Injury	[2180, 1995, 2000]

Figure 6: *Location of crash on side street along with locations of derived nearby intersections. Red marker: location of crash; Blue marker: location of nearby intersections.*

Table 9: *Mapping of street names from crash reports to ATSPM street names when length of maximum common substring is greater than 5. This table has the following attributes: (a) HSMV_No: HSMV_Report_Number from crash reports, (b) C_Date: crash date, (c) C_Time: crash time, (d) C_Street: crash street reported in SFA, (e) S_name: mapped street name from ATSPM, and (f) LCS: longest common substring shared by crash street and street name from ATSPM.*

HSMV_No	C_Date	C_Time	C_Street	S_name	LCS
88801272	01/30/2019	10:34 AM	ALAFAYATRL	alafayawoods	alafaya
88801271	01/29/2019	02:52 PM	LOCKWOODBLVD	lockwoodadt	lockwood
88801270	01/29/2019	12:24 PM	LOCKWOODBLVD	lockwoodadt	lockwood
88801282	02/06/2019	08:26 AM	LOCKWOODBLVD	lockwoodadt	lockwood
88702663	01/25/2019	06:55 PM	MAITLANDBLVD	maitlandave	maitland
88614466	01/01/2019	02:28 PM	LAURAST	laurastadt	laurast
87773274	02/08/2019	05:43 PM	LAKEEMMARD	lkemmard	emmard
88702737	02/08/2019	03:20 PM	NMAITLANDAVE	maitlandave	maitlandave
88927573	02/12/2019	02:28 PM	RINEHARTRD	rinehartrd	rinehartrd
88614610	02/08/2019	06:45 PM	OXFORDRDN	oxford	oxford
88770185	02/12/2019	11:12 AM	REDBUGLAKERD	redbuglkrd	redbugl
88770133	02/08/2019	12:58 AM	SSYLVANLAKEDR	lakedradt	lakedr
88927352	01/02/2019	08:59 AM	COUNTRYCLUBRD	countryclub	countryclub
88927348	01/01/2019	03:58 PM	WSEMINOLEBLVD	seminol ablvdadt	seminol
87291537	01/07/2019	06:50 AM	US1792	us1792	us1792
87291536	01/06/2019	09:45 AM	WTOWNPKWY	west town pkwyadt	townpkwy

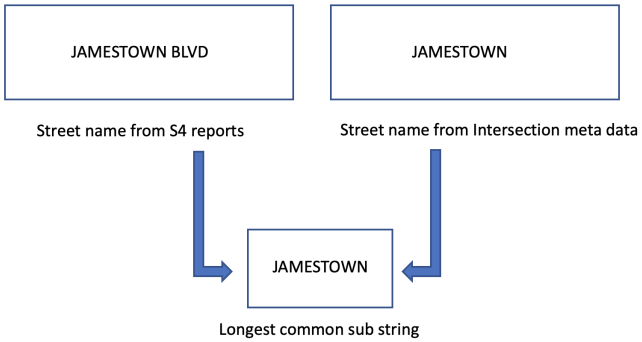

Figure 7: *Brief outline of the heuristic approach for filtering crashes that happened on major streets*

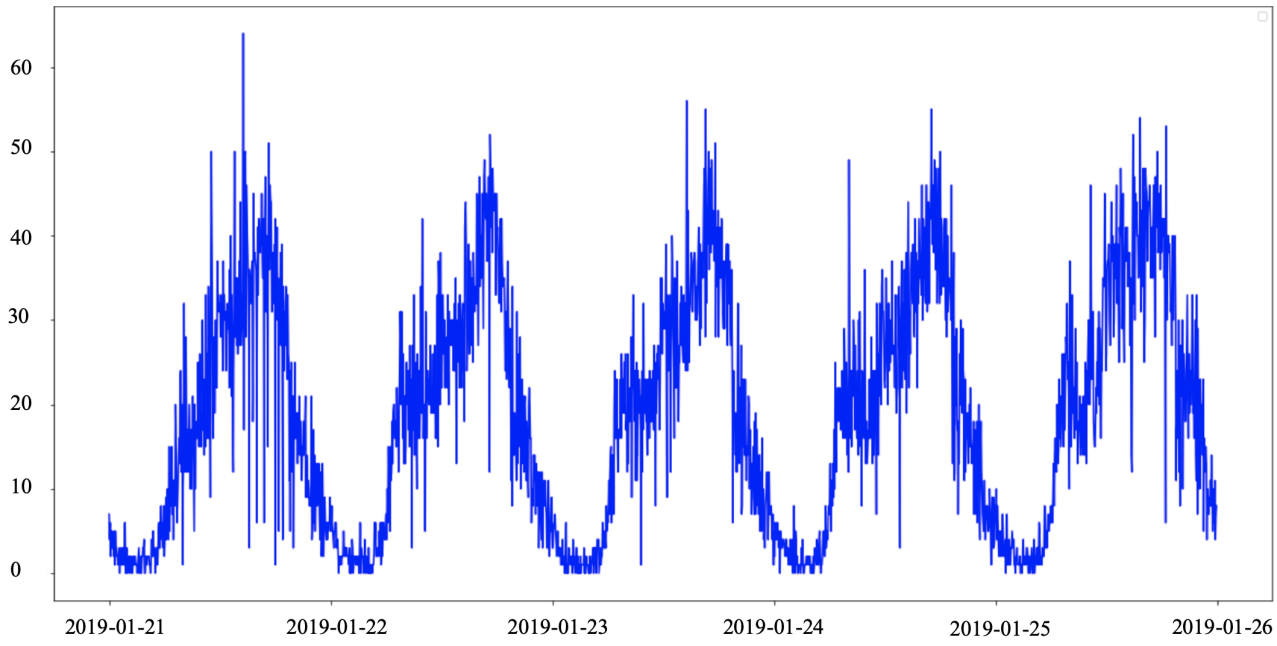

Figure 8: *Arrival volumes visualized as time series*

for an intersection for a given time interval. This tool helps us to visualize arrival volumes before and after a potential incident point (Figure 9).

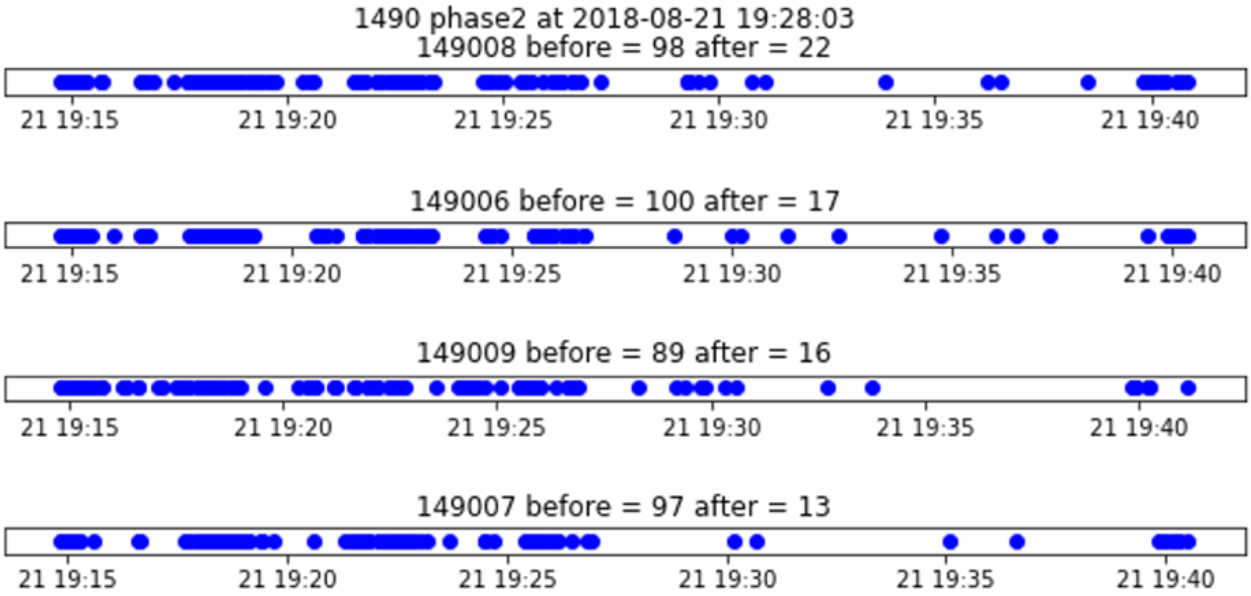

Figure 9: *Plot of arrivals on phase 2 of Signal 1490 from 19:20 to 19:40 on Aug. 21, 2018*

4 Labeling Interruptions

In order to be able to reliably predict events of interest, we first need a method for labeling such events, and furthermore, the labeling mechanism should be separate from the event prediction algorithm. We now describe the labelling mechanism.

Traffic interruptions are registered in the controller logs—as long as we know where to look. But, not every interruption is a major one. A *large* traffic interruption—and to be clear, we are not interested in small interruptions—is defined based on two parameters:

1. The magnitude of deviation (percentage reduction) of observed traffic volumes from predicted volumes. This is measured in terms of the percentage dip of the actual traffic volume vs. the predicted value. Common sense dictates that the greater the deviation, the larger the interruption.

2. The duration (in seconds) for which the actual traffic volume is less than a *baseline* predicted volume. Again, a big duration heralds a large interruption.

Let Y denote the percentage reduction of volume and T denote the time (in seconds) of the interruption. Events are therefore characterized in this two-dimensional summarization space.

Recall that we need a baseline prediction method that gives us *normal* expected volumes of traffic. To achieve this, we merely look at the differences between arrival volumes (in recent cycles) and historical arrival volumes from similar time periods from previous days and/or weeks. The baseline predictor uses a simple method to generate traffic interruptions based on the time series data generated from arrival volumes. We found that a simple baseline predictor works well in practice and that our approach is not terribly sensitive to the choice of method. In other words, any common sense approach that yields large traffic interruptions will work in this setup.

We use a variation of non-local means as the baseline predictor. Since arrival volumes at any given time are highly dependent on cycle length and immediately preceding traffic volumes, we use the *arrivals rate* rather than arrival volumes in the predictor. Let V_i and T_i correspond to the number of arrivals and the duration of cycle i, respectively. Then the arrival rate, X_i, is defined as $X_i = \frac{V_i}{T_i}$. The prediction algorithm finds a (linear) function that computes the arrival rate for the current cycle using previous cycles from the same day and historically relevant cycles from previous periods. Our model for f is

$$X_t = f(X_{t-1}, X_{t-2}, \ldots, X_{t-k}, Y_{t-k}, \ldots, Y_{t+k}),$$

where X_i and Y_i corresponds to arrival rates from the current day and historical data respectively. Expected arrival volumes (baseline) can now be computed using the arrival rate multiplied by cycle length.

Figure 10 shows that predicted volumes are in line with actual traffic volumes. Figure 11 shows example of an event where there is a significant deviation of traffic volumes from predicted volumes (amount

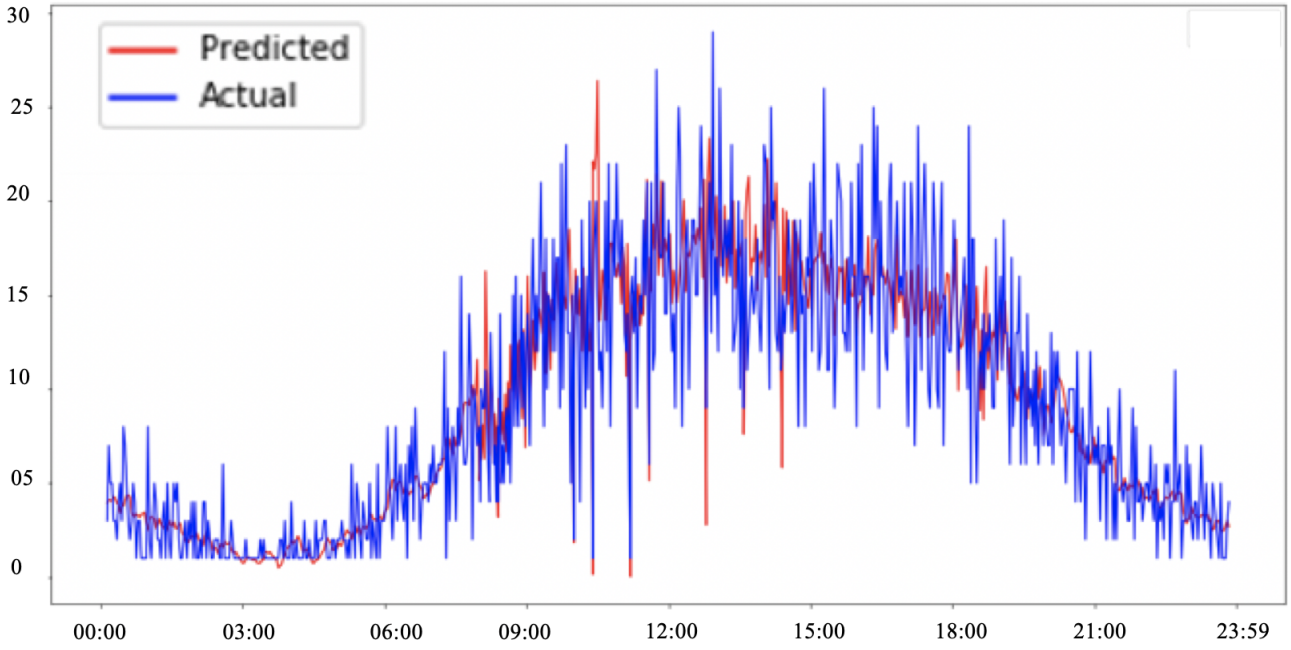

Figure 10: *Actual and predicted volumes vs. time for a period of 24 hours on a single detector shows that predicted volumes are largely consistent with actual traffic patterns.*

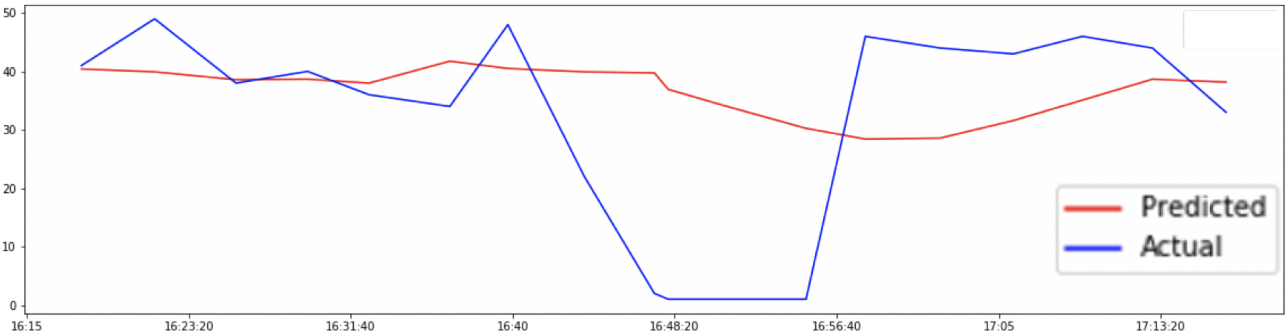

Figure 11: *Example of an event of interest which shows significant deviation of traffic volumes from predicted volumes (amount and duration)*

and duration). In the figure 11 we can see that actual traffic volume deviated from predicted volume for a long period of time. We are interested in interruptions where the mean dip and length of dip are significant, as shown in Figure 11.

It is worth noting that we are only interested (in this paper) in periods when the volume in the cycle is less than the baseline, with such events henceforth referred to as dips. Each dip, as mentioned previously, is parameterized by its amount (Y) and duration (T). The scatterplot of these dips is a 2D event space whose probability distribution can be estimated from a simple 2D histogram. We generated all the traffic interruptions using the algorithm in Algorithm 2 to compute this two-dimensional histogram.

The matrix in Figure 12 shows frequencies for different dips based on average reduction in volume (along the rows) and duration (along the columns). We are not interested in events where there are no significant interruptions. The distribution of dips suggests a tripartite distinction, which we adopt: central (green), borderline (yellow), and discards (red), as shown in Figure 12. The 2D histogram also

Algorithm 2 Label interruptions

function GENERATE EVENT(arrivalvols , predf)
Require: arrivalvols - Time series of arrival volumes .
predf - predictor function
    $listofevents$ = []
    while c< total no of cycles **do**
        predvol = predf($X_{c-1}, ..X_{c-k}, Y_{c-k}, ..Y_{c+k}$).
        differences = []
        Set starttime equal to the cycle time
        while cycle volume is < the $predvol$ **do**
            $reduction = \frac{predvol - c.Vol}{predvol}$*100
            append $reduction$ to differences
            increment c
            predvol =
                predf($X_{c-1}, .X_{c-k}, Y_{c-k}, .Y_{c+k}$).
        end while
        Set endtime equal to the cycle time
        generate $event$ Y = *average*(differences),
          T = endtime-starttime
        append $event$ to $listofevents$
    end while
    Return $listofevents$
end function

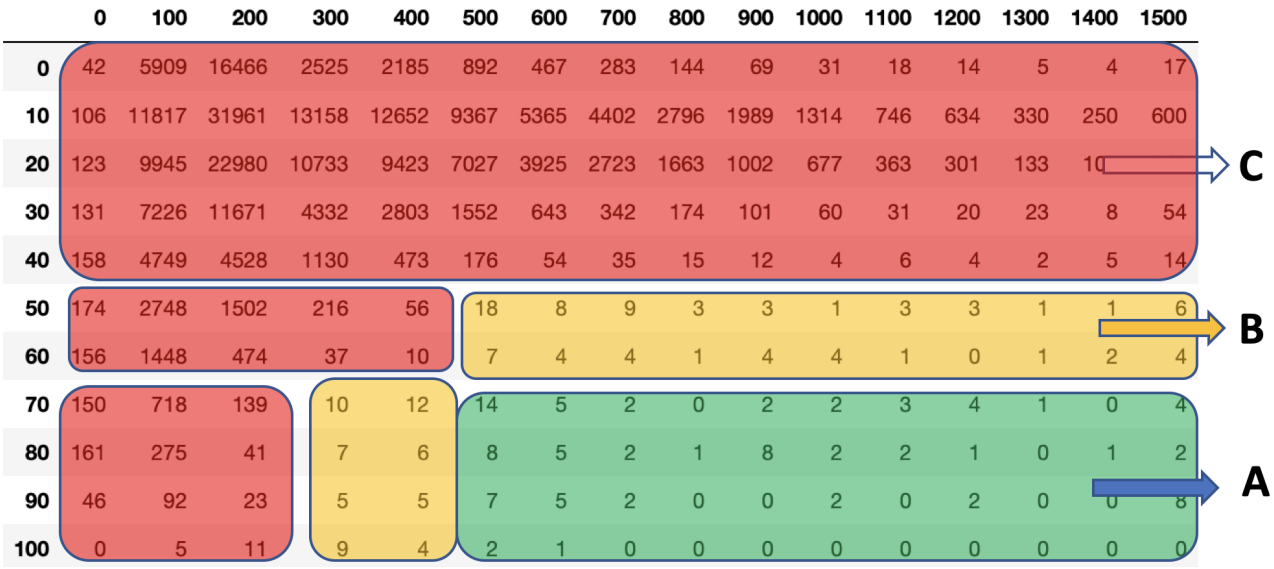

Figure 12: *Frequencies for different dips based on average reduction in volume percentage (along the rows) and duration in seconds (along the columns) showing the number of events in each bin: (a) Events of interest – these are events with long interruptions with significant reductions in traffic volume; (b) Border events – these are events that are not desirable but acceptable to catch; (c) Events that are not of interest.*

suggests natural thresholds on volume reduction and duration which can be adopted to discard normal behavior while only keeping the central and borderline behaviors.

In the remainder of the report, we use thresholds of $\mathbf{70}\%$ for volume reduction and **500 seconds** for duration, with these choices vetted by traffic engineers as being reasonable for this study. These result in traffic interruptions of reasonably long duration while being relatively infrequent but severe enough to require addressing. Clearly, such thresholds can be fine-tuned by traffic engineers based on their requirements.

In the next section, we present our methodology to predict events of interest (EOIs) with the goal of the predictive algorithm being to capture most of the events within the bounding box and perhaps some borderline events while ignoring most, if not all, the non-events outside the borders.

5 Predicting Events of Interest

In this study, we assume that the data are being streamed in real time for all the detectors on each intersection. The preprocessing algorithm described in Section 3 is used to compute per-cycle volumes in real time. This information and previous cycles (and historical data) are then used to determine if an EOI has occurred. Clearly, for the approach to be useful, this determination has to be done as soon as possible while minimizing the false positives and false negatives. There is a trade-off between latency of this determination and the false positive and false negative rates.

We present a brief outline of the approach. Since arrival volumes in cycles considerably vary, we use *cumulative volumes* instead. Then cumulative volumes from the present cycle are compared to previous and historically relevant cumulatives. The comparison in turn leads to a decision criterion which is scored in terms of true and false positives (using the 'central' and 'borderline' labels from the previous section). The first step of this process is to only consider a cycle if the current volume is smaller than the predicted volume. If this condition is met, we say that a trigger has happened, and we construct the following cumulative arrival curves.

- Curve 1 (in red) starting from start of previous cycle Figure 14).
- Curve 2 (in blue) corresponds to recent normal cycles.
- Curves 3, 4, and 5 (in green) are cumulative arrival curves for the same time interval as curve 1 from cycles based on historical data from the same time of the day and day of the week.

The goal is to compare the current cumulative curve from normal cumulative curves for that particular time interval(constructed based on recent and past history). For example, in Figure 13a for an event of interest, the current cumulative curve is different from those of normal cumulative curves. Whereas, for an event not of interest, the current cumulative curve is similar to some normal cumulative curves Figure 13b. Based on our analysis of many of such cases, we found that the following features are highly predictive of an EOI:

- **Feature 1: Slope of current cumulative curve(m1)** Slope of the current curve from crossover point. Crossover point as shown in Figure 14 is the point where curve 1 and curve 2 intersect (the point from where the current cumulative curve starts deviating from normal cumulative curves for that time interval).
- **Feature 2: Angle between current cumulative curve and historical curves(m2)** This is taken as second max (to eliminate a potentially extreme case) of angle between curve 1 and curves 2, 3, 4, and 5 (from the vertex at the crossover point).

In order to determine these two features, the duration of the curves to be considered after the trigger is also an important parameter. As seen in Figure 14, after the crossover point, the degree of dissimilarity between current cumulative curve and normal cumulative curves increases with time for an event of interest. So with increase in wait time after the trigger, the slope of current cumulative curve decreases

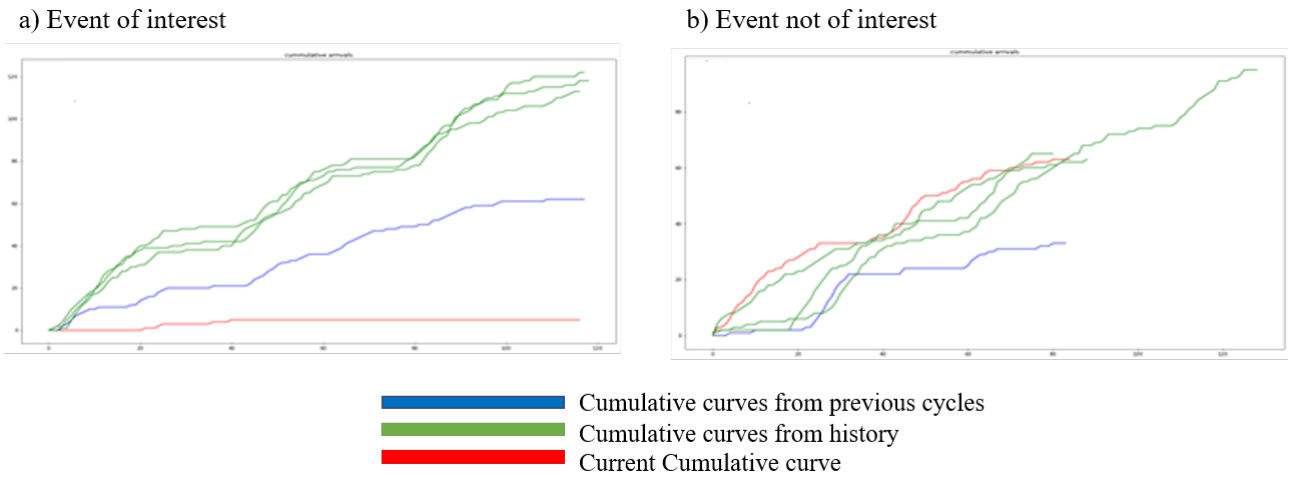

Figure 13: *This figure shows the difference between an event of interest and event not of interest. We plot cumulative arrival volumes vs. time.*

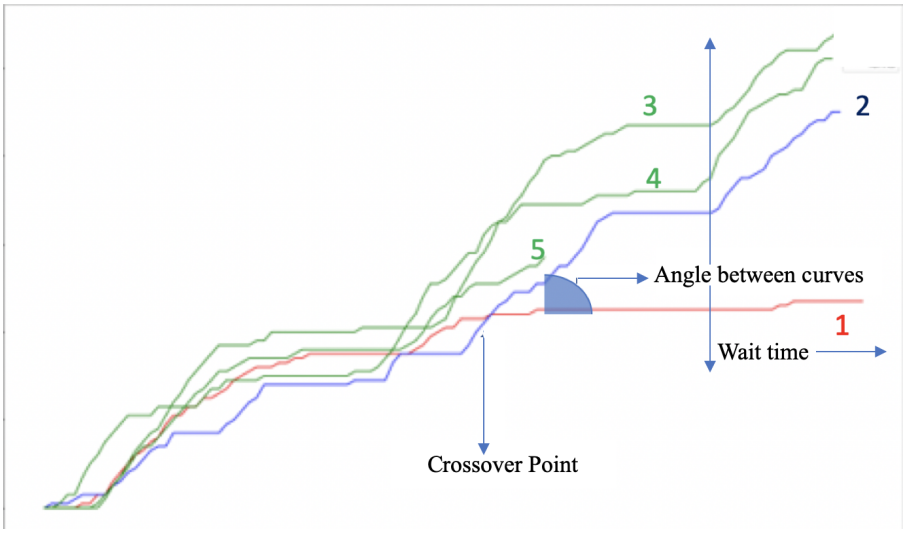

Figure 14: *Cumulative curves of arrival volumes vs. time. We use these curves to construct our feature set.*

while the angle between current cumulative and normal curves increases, thus making an EOI easier to detect.

Arrival volumes per unit time are variable across all the detectors so we divide the detectors into three sets: high volume detectors, medium volume detectors, and low volume detectors. The thresholds for capturing events of interest will be different for each set because the magnitude of a feature depends on arrival rates. For each detector, we compute arrivals per unit time during daytime for 30 days and we divide the detectors based on this ratio. We perform the following experiments to see the distribution of events by using different thresholds for feature 1, feature 2, and wait time. For each set of detectors, we compute the distribution of events captured using different thresholds for feature 1 and feature 2 and for different wait times after the trigger. This way, we try to find thresholds for the features to capture events of interest.

Figure 15 shows the distribution of events captured for different thresholds and wait times: 60 sec, 90

a) 60 sec after the trigger

m2 / m1	B 0.50	B 0.60	B 0.70	B 0.80	B 0.85	A 0.50	A 0.60	A 0.70	A 0.80	A 0.85	C 0.50	C 0.60	C 0.70	C 0.80	C 0.85
0.05	62	61	59	56	55	91	91	91	89	88	21	18	18	15	14
0.07	65	64	62	60	57	93	93	93	91	90	25	22	20	17	16
0.09	70	70	67	65	62	94	94	94	92	91	31	27	25	20	19
0.10	70	70	67	65	62	94	94	94	92	91	36	31	26	20	19
0.13	75	74	70	68	65	95	95	95	93	92	54	40	35	25	24
0.15	81	80	76	73	70	96	96	96	94	93	66	49	43	30	28
0.20	88	87	83	80	76	96	96	96	94	93	116	85	72	50	41
0.25	98	96	91	87	83	98	98	98	96	95	169	130	103	72	56
0.30	107	105	99	94	90	98	98	98	96	95	262	203	156	108	82

b) 90 sec after the trigger

m2 / m1	B 0.50	B 0.60	B 0.70	B 0.80	B 0.85	A 0.50	A 0.60	A 0.70	A 0.80	A 0.85	C 0.50	C 0.60	C 0.70	C 0.80	C 0.85
0.05	47	47	46	45	44	89	89	89	86	86	11	10	10	8	7
0.07	51	50	49	49	47	90	90	90	87	87	12	11	11	9	8
0.09	55	53	52	52	50	91	91	91	88	88	12	11	11	9	8
0.10	56	54	53	53	51	92	92	92	89	89	16	15	14	12	10
0.13	65	63	61	60	58	92	92	92	90	90	23	22	18	15	13
0.15	71	69	66	64	62	95	95	95	93	93	29	26	20	17	15
0.20	81	78	74	73	70	95	95	95	93	93	48	41	30	23	20
0.25	92	87	82	81	78	95	95	95	93	93	77	59	43	33	26
0.30	96	91	86	85	83	98	98	98	96	96	139	103	74	54	46

c) 120 sec after the trigger

m2 / m1	B 0.50	B 0.60	B 0.70	B 0.80	B 0.85	A 0.50	A 0.60	A 0.70	A 0.80	A 0.85	C 0.50	C 0.60	C 0.70	C 0.80	C 0.85
0.05	36	36	35	33	33	87	87	87	86	84	6	6	6	5	5
0.07	41	40	39	38	37	88	88	88	87	85	7	7	7	6	6
0.09	42	41	40	39	38	88	88	88	87	85	9	9	9	8	8
0.10	46	45	44	43	42	88	88	88	87	86	12	12	12	11	10
0.13	55	53	52	50	49	89	89	89	88	87	16	14	13	12	10
0.15	58	56	54	52	51	89	89	89	88	87	16	14	13	12	10
0.20	73	70	67	65	63	91	91	91	90	89	27	24	20	17	14
0.25	86	82	78	77	72	94	94	94	93	92	45	37	30	24	19
0.30	97	93	87	87	80	97	97	97	96	95	93	71	51	40	32

Figure 15: *Distribution of events captured for different thresholds (high volume detectors) 60 sec, 90 sec, and 120 sec after the trigger. This justifies that by waiting more time after the trigger, we capture fewer events that are not of interest.*

Algorithm 3 Detection algorithm

function DETECTION ALGORITHM(volume reduction, time)
Require: volume reduction – percentage reduction of arrival volume in the cycle. Detector ID – detector at which interruption happened. ph – phase time – Time at which reduction happened
  for different wait times after the trigger **do**
    Construct cumulative arrival curves based using current arrivals, history
    Construct feature 1, feature 2 from cumulative
      arrivals for the wait time as described.
    Decision = thresholds(feature1, feature2, wait time)
  end for
  Return Decision
end function

sec, and 120 sec after the trigger for high volume detectors. We can see that the number of captured events that are not of interest decreases with an increase in wait time for the same set of thresholds. This suggests that by waiting more time after the trigger, we capture fewer events that are not of interest. Also, the number of events captured that are not of interest decreases with increase in m2 and decrease in m1, but the trade-off is that we miss some of the events of interest.

Figure 16 shows a comparison of the overall distribution of events vs. events captured when feature 1 (m1) < 0.09 and feature 2 (m2) > 0.7 for wait times 90 sec after the trigger for high volume detectors. For this particular set of thresholds (m1, m2, wait time), 92% of the events of interest, 35% of border events, and a very small percentage of events not of interest were captured.

Figure 17 shows comparison of overall distribution of events vs. events captured when feature 1 (m1) < 0.07 and feature 2 (m2) > 0.5 for wait time 90 sec after the trigger for medium volume detectors (average reduction in volume percentage (along the rows) and duration in seconds (along the columns)). For this particular set of m1, m2, and wait time, 99% of events of interest were captured along with 8% border events and less than 1% percentage of events not of interest.

a) Overall Distribution of Events

Total Number of Events of Interest - 98

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	42	5909	16466	2525	2185	892	467	283	144	69	31	18	14	5	4	17
10	106	11817	31961	13158	12652	9367	5365	4402	2796	1989	1314	746	634	330	250	600
20	123	9945	22980	10733	9423	7027	3925	2723	1663	1002	677	363	301	133	107	340
30	131	7226	11671	4332	2803	1552	643	342	174	101	60	31	20	23	8	54
40	158	4749	4528	1130	473	176	54	35	15	12	4	6	4	2	5	14
50	174	2748	1502	216	56	18	8	9	3	3	1	3	3	1	1	6
60	156	1448	474	37	10	7	4	4	1	4	4	1	0	1	2	4
70	150	718	139	10	12	14	5	2	0	2	2	3	4	1	0	4
80	161	275	41	7	6	8	5	2	1	8	2	2	1	0	1	2
90	46	92	23	5	5	7	5	2	0	0	2	0	2	0	0	8
100	0	5	11	9	4	2	1	0	0	0	0	0	0	0	0	0

b) Events with m1<0.09 and m2>0.7

91 Events of Interest , 52 Border Events, 11 Events that are not of interest

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0
40	0	0	0	0	0	0	0	1	1	1	0	0	0	0	1	1
50	0	0	0	1	0	0	1	3	1	1	0	1	0	0	1	2
60	0	0	0	0	1	2	1	4	1	1	2	0	0	1	2	3
70	0	0	0	2	2	9	4	2	0	2	2	3	4	0	0	4
80	0	0	0	4	3	8	5	2	1	8	2	2	1	0	1	2
90	0	0	1	2	4	7	5	2	0	0	2	0	2	0	0	8
100	0	0	1	5	3	2	1	0	0	0	0	0	0	0	0	0

Figure 16: *Distribution of events captured for m1 < 0.09 and m2 > 0.7 for wait time 90 sec after the trigger (average reduction in volume percentage (along the rows); duration in seconds (along the columns). Ninety-two percent of the events of interest, 35% of border events, and a very small percentage of events not of interest were captured (high volume detectors).*

a) Overall Distribution of Events

Total Number of Events of Interest - 24

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	51	6755	12704	2297	1473	846	284	283	89	75	24	23	10	6	7	9
10	114	13961	26782	12159	10068	8058	4055	4192	2025	1940	1032	774	478	386	282	716
20	131	12622	21905	12620	10134	8439	4790	3702	2155	1506	1058	649	486	297	235	619
30	144	10023	13214	6755	4475	2878	1405	846	458	230	168	95	81	34	35	115
40	123	6868	6051	2257	997	487	168	65	41	16	22	15	7	10	3	21
50	133	4387	2267	469	169	43	7	10	5	3	2	4	2	0	2	5
60	128	2403	639	77	18	6	1	1	0	1	2	0	0	1	1	6
70	144	1154	209	11	2	0	1	0	1	0	0	0	1	1	0	6
80	101	511	70	4	1	0	0	0	1	2	0	0	1	0	0	5
90	6	124	19	2	0	1	1	0	0	1	0	0	0	0	0	1
100	0	2	3	0	0	0	1	0	0	0	0	0	0	0	0	0

b) Events with m1<0.07 and m2>0.5

23 Events of Interest , 10 Border Events, 12 Events that are not of interest

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
40	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	2
50	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	2
60	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	5
70	0	1	0	0	0	0	1	0	0	0	0	0	1	1	0	6
80	0	0	1	0	1	0	0	0	1	2	0	0	1	0	0	5
90	0	0	0	0	0	1	1	0	0	1	0	0	0	0	0	1
100	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

Figure 17: *Events captured for m1 < 0.07 and m2 > 0.5 for wait time 90 sec after the trigger (medium volume detectors) (average reduction in volume percentage (along the rows), and duration in seconds (along the columns)). 99% of events of interest were captured with 8% border events and less than 1% of events not of interest*

a) Overall Distribution of Events

Total Number of Events of Interest - 34

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	11	1068	2205	277	200	84	24	16	13	1	2	3	0	2	0	0
10	14	2264	4690	1839	1345	1046	548	496	242	205	103	79	40	35	24	60
20	18	2105	4301	2312	1937	1756	931	908	498	369	272	161	121	80	65	164
30	18	1684	3205	1610	1209	1068	492	421	240	168	112	55	50	32	30	52
40	23	1234	1829	704	460	307	149	92	50	22	16	20	7	5	4	12
50	13	816	824	227	94	49	21	8	8	0	3	2	3	0	0	2
60	15	470	328	54	19	7	2	4	2	1	2	0	2	1	1	2
70	20	251	103	10	3	2	0	2	0	1	3	3	4	1	2	6
80	9	91	28	2	0	0	2	0	2	0	1	1	0	0	0	2
90	0	21	9	0	0	0	0	0	1	1	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

b) Events with m1<0.09and m2>0.4

26 Events of Interest , 4 Border Events, 8 Events that are not of interest

	0	100	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0
40	0	0	0	0	1	0	2	0	0	0	1	0	0	0	0	0
50	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1
60	0	0	0	0	1	0	1	0	0	0	0	0	0	0	1	0
70	0	0	0	0	0	1	0	2	0	0	3	2	3	0	2	3
80	0	0	0	0	0	0	2	0	2	0	1	1	0	0	0	2
90	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 18: *Events captured for m1 < 0.09 and m2 > 0.4 for wait time 90 sec after the trigger (low volume detectors). Seventy-five percent of events of interest were captured.*

6 Discriminator for Detecting EOI

We design a discriminator for detecting Events of Interest (EOI), i.e., a two-class classifier separating Event of Interest from all other events. We have defined the problem as a binary time series classification task. Time series classification is one of more challenging problems in machine learning. The problem can be stated as training a model which when fed with a series of data points indexed in temporal order, outputs the probability distribution over different classes. We use a neural network with one hidden layer as our model for the classification. Input to our classifier is the current cumulative curve of length L concatenated with an historic (*"normal"*) cumulative curve of the same length L. The output is a prediction about the current cumulative curve belonging to EOI or not, one hot label encoding (either 1 or 0). It is well known that neural networks in the training phase converge faster and that the network learns better if the input features are not correlated. Therefore, we transform the input time series to a low dimensional subspace where most of the variability of the data is captured. Also, given that we have relatively few training examples for the positive class (EOI), dimensionality reduction helps in reducing the size of the network to a large extent, thereby reducing the number of parameters that need to be learned. This makes the model more robust and less likely to over-fit. We use principal component analysis as our tool for dimensionality reduction. (Section 6.2).

6.1 Neural Networks

Fundamentally, all neural networks consist of an input layer, hidden layers, and an output layer. The basic unit of a neural network is called a neuron (very loosely related to biological neurons). The following is some of the terminology used:

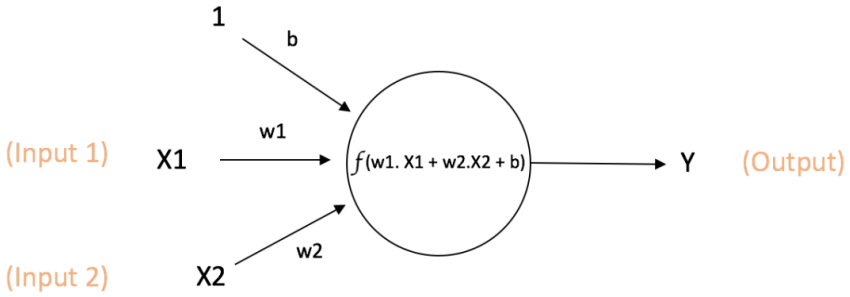

Figure 19: *Single neuron with input and output*

- **Input Layer**. This layer takes the input in the required shape and passes it to the hidden layer. No computations are done here.
- **Activation Function**. The output of each neuron is a function of its input. This function is called activation function of a neuron. Some activation functions that are commonly used in practice

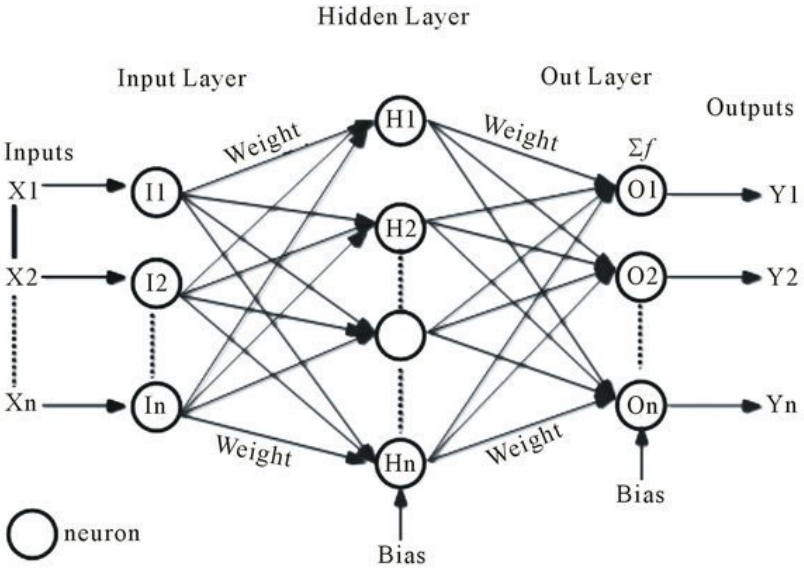

Figure 20: *Example of a neural network with one input layer, one hidden layer, and one output layer*

are Sigmoid, ReLu, Tanh.

- **Hidden Layer**. They receive input from the previous layers, perform computations and then transfer their output to the succeeding layer. The first hidden layer receives it's input from the *"input"* layer and the last hidden layer transfers it's output to the *"output"* layer.

- **Output Layer**. With the use of specific activation functions, this layer is used to map output to the desired format.

- **Connections**. Output of each neuron is transferred to a neuron in the succeeding layer with a connection. A weight is associated with each connection. If output of i is connected to input of j, then corresponding weight is W_{ij}

Figure 19 shows a single neuron and the input, output mechanism. It takes two inputs X1, X2 and they are connected with weights W1, W2. So, the input to the neuron becomes W1.X1 + W2.X2. Output of this neuron is a function of this input that is fed to it. These functions are called activation functions. This is important because it allows the model to learn non-linear relations between input and output.

Figure 20 shows a three layer network with all the labelling. The inputs X1, X2,...Xn when fed to the network outputs probability over different classes Y1, Y2...Yn.

6.2 Principal Component Analysis

Principal Component Analysis(PCA) is a well known technique for dimensionality reduction - reduce the dimensionality of data while modeling and preserving as much variability as possible. PCA finds new variables that are linear functions of the original variables, that successively maximize variance and are linearly independent with each other. Finding new variables (principal components) essentially reduces

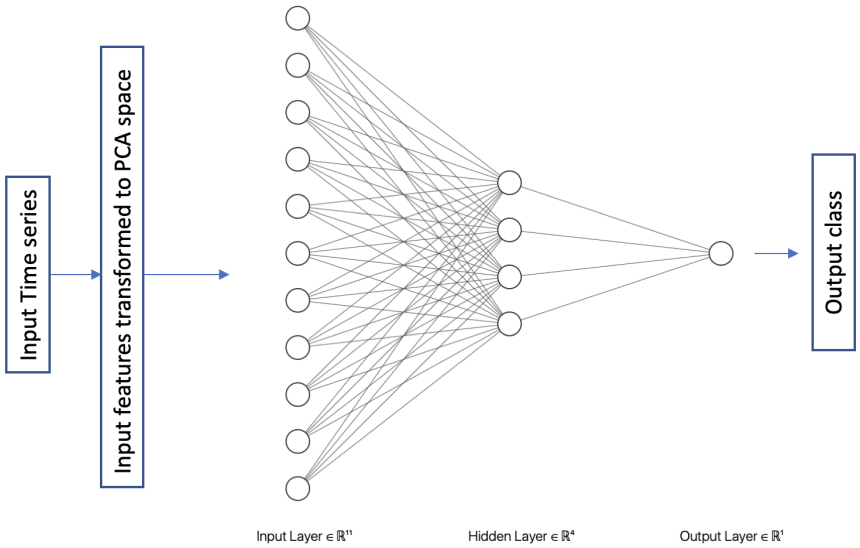

Figure 21: *Example architecture of the classifier*

to solving for eigenvectors/eigenvalues of the covariance matrix of the original data. The eigenvectors are ranked according to the variance of data along them (decreasing order of eigenvalues). The axis along top eigenvector (eigenvector with max eigenvalue) tries to capture as much information as possible from the data, then the maximum possible remaining information is captured in the second component and so on. The original data(in D dimensional space) is projected onto top d eigen vectors (which act as a basis for the transformed space) to get d dimensional representation of the original data. The value of d is chosen such that most of the variance in the data is captured. The steps for projecting D dimensional data to a d dimensional space are as follows:

- Compute covariance matrix of the data points (Input dimension *D*).
- Compute eigenvectors and corresponding eigenvalues. Rank eigenvectors in decreasing order of eigenvalues.
- Choose top n eigen vectors as new basis for the projected space.
- Construct projection matrix *W* from the choosen eigen vectors
- Use projection matrix to transform original data in D dimensional space to new d dimensional subspace.

  Transform the original D dimensional data points to this new d dimensional space.

Let $\{\vec{x}_i\}$ be a set of N column vectors of dimension D. Define the covariance matrix S_x of the original data as

$$\mathrm{S}_x = \sum_{i=1}^{N} (\vec{x}_i - \vec{\mu}_x)(\vec{x}_i - \vec{\mu}_x)^T$$

where $\vec{\mu}_x$ is the mean of the data

$$\vec{\mu}_x = \frac{1}{N}\sum_{i=1}^{N}\vec{x}_i$$

The d largest principle components are the eigenvectors $\vec{w}_i$ corresponding to the d largest eigenvalues. d can be chosen to have a value $d < D$, such that most of the variance in the data is captured. The eigenvectors of **S** can usually be found by using singular value decomposition.

The dominant eigenvectors describe the main directions of variation of the data.

The d eigenvectors can also be used to project the data into a d dimensional space. Define

$$\mathrm{W} = [\vec{\mu}_1, \vec{\mu}_2, \ldots, \vec{\mu}_d]$$

The projection of vector $\vec{x}$ is $\vec{y} = \mathrm{W}^T\vec{x}$. The corresponding scatter matrix S_y of the vectors $\{\vec{y}_i\}$ is:

$$\mathrm{S}_y = \mathrm{W}^T\mathrm{S}_x\mathrm{W}$$

The matrix W maximizes the determinant of S_y for a given d. Another important aspect is how many principal components(cardinality of new space) to choose for the new space. One useful measure is explained variance - how much variance is captured by each principal component. By looking at the cumulative of explained variance with respect to number of components, we can make a decision on how many components to choose. Figure 22 shows example of a two dimensional data which we intend to project on to a 1 dimensional space. One could select that single dimension in many possible ways. Suppose we select the line $y = 0$ as in Figure 22(a) or the magenta line as in Figure 22(a) and project the 2 dimensional data on to this line, the red dots shows the projection in 1 dimensional space. We can clearly see that the red dots in Figure 22(b) are more widely spread out than Figure 22(a). The axis along magenta line is direction of the most dominant eigenvector (eigenvector with highest eigenvalue). The original 2 dimensional data is most spread along this direction and projecting onto this line results in least reconstruction error or most explained variance.

Figure 23 shows plot of cumulative explained variance vs no of components when PCA is applied to the input time series. This plot suggests that taking top 20 components captures 99% of the variance of the data. So we can project 194 dimensional input vector on to 20 dimensional PCA space. Figure 21 shows the architecture of the neural network. One hidden layer with four neurons and an output node with sigmoid activation is used. This neural network is trained with cumulative curves transformed into PCA space as input features and binary cross entropy as the loss function. Adam optimizer is used for back propagation of error and updating the weights of network. Figure 24 shows the confusion matrix for a hold out test set. Figure 25 shows the plot of log of model loss versus number of epochs for training and validation sets.

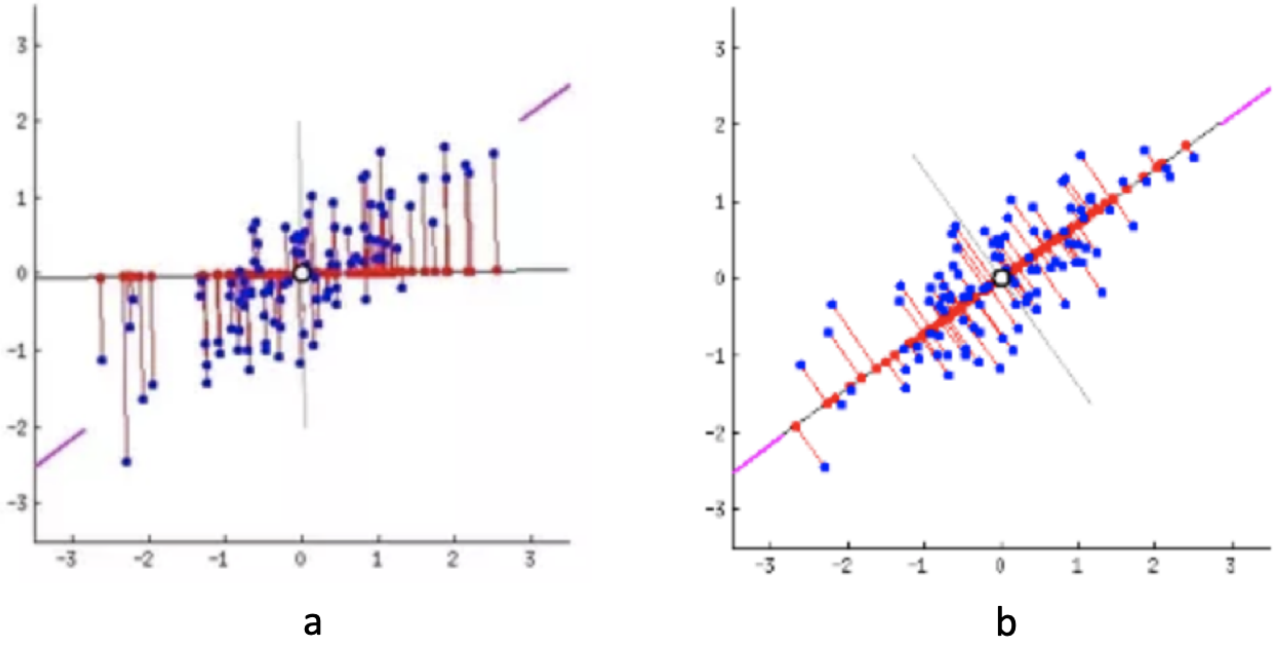

Figure 22: *Example of two-dimensional data projected onto a one-dimensional space*

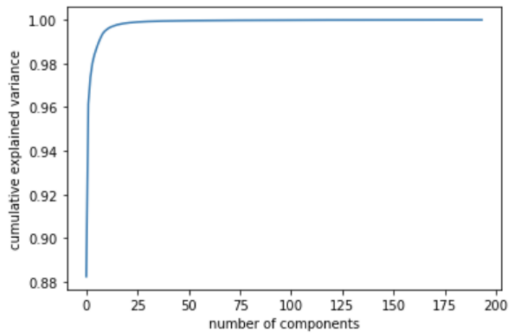

Figure 23: *Plot of cumulative explained variance vs. no. of components. This plot suggests that taking the top 20 components captures 99% of the variance of the data.*

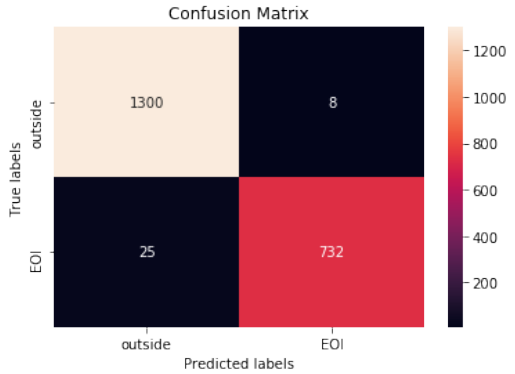

Figure 24: *Confusion matrix for the binary classification task*

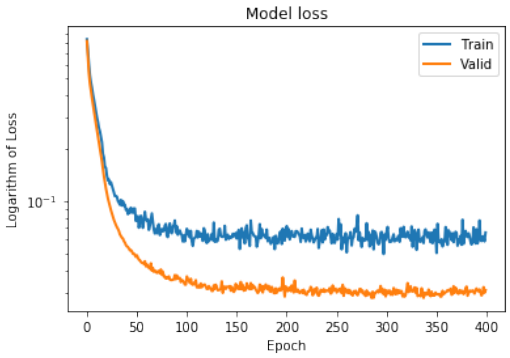

Figure 25: *Plot of logarithm of model loss vs. number of epochs for training and validation sets*

7 Spatiotemporal Analysis of Events

We also analyze the events of interest with the aim of finding spatial or temporal patterns. This is done to highlight

- **Temporal patterns:** To see if there is any particular intersection where traffic interruptions are occurring regularly
- **Spatial patterns:** To see if interruption at a intersection causes interruption at any nearby intersection.

Figure 26 shows a plot of locations of all the events of interest. Each red marker on the map indicates the location of an intersection where an event occurred. From the plot, we can see that events are occurring more frequently at some intersections.

7.1 Temporal Patterns

The onus is on temporal pattern discovery to determine coherent activities at intersections over the period of a month. This requires the visualization of locations of EOIs separately for each month. In these plots, the size of the marker is directly proportional to the number of events occurring at that particular intersection. We can see from Figure 27 that at some intersections, EOIs occur consistently every month. Figure 28 shows an example of an intersection where the EOIs are occurring consistently over the months of January, February, March, and April. Figure 29 shows another example of an intersection where the events are occurring consistently over the months of December, January, February, March, and April. The payoff from temporal pattern discovery is the highlighting of problematic intersections in this manner.

Table 10 shows a set of intersections where interruptions are occurring consistently every month. At these intersections, the total number of EOIs occurred is greater than 20, and an EOI occurred at least 5 out of 6 months.

Table 10: *Table showing list of intersections where total number of EOIs > 20 and an EOI occurred at least 5 out of 6 months. Attributes include (a) signalid: signal ID, (b) c_nov: no. of EOIs in Nov. (similarly for other months), (c) total: total no. of EOIs, and (d) N_months: no. of months in which at least one EOI occurred.*

signalid	c_nov	c_dec	c_jan	c_feb	c_mar	c_apr	total	N_months
2220	2	8	9	7	7	3	36	6
2615	2	6	11	1	6	7	33	6
1925	1	10	2	4	1	7	33	6
1235	2	0	6	21	22	18	69	5
1470	12	51	1	1	1	0	66	5
2145	0	6	12	8	9	6	41	5
1115	0	1	5	5	7	3	21	5

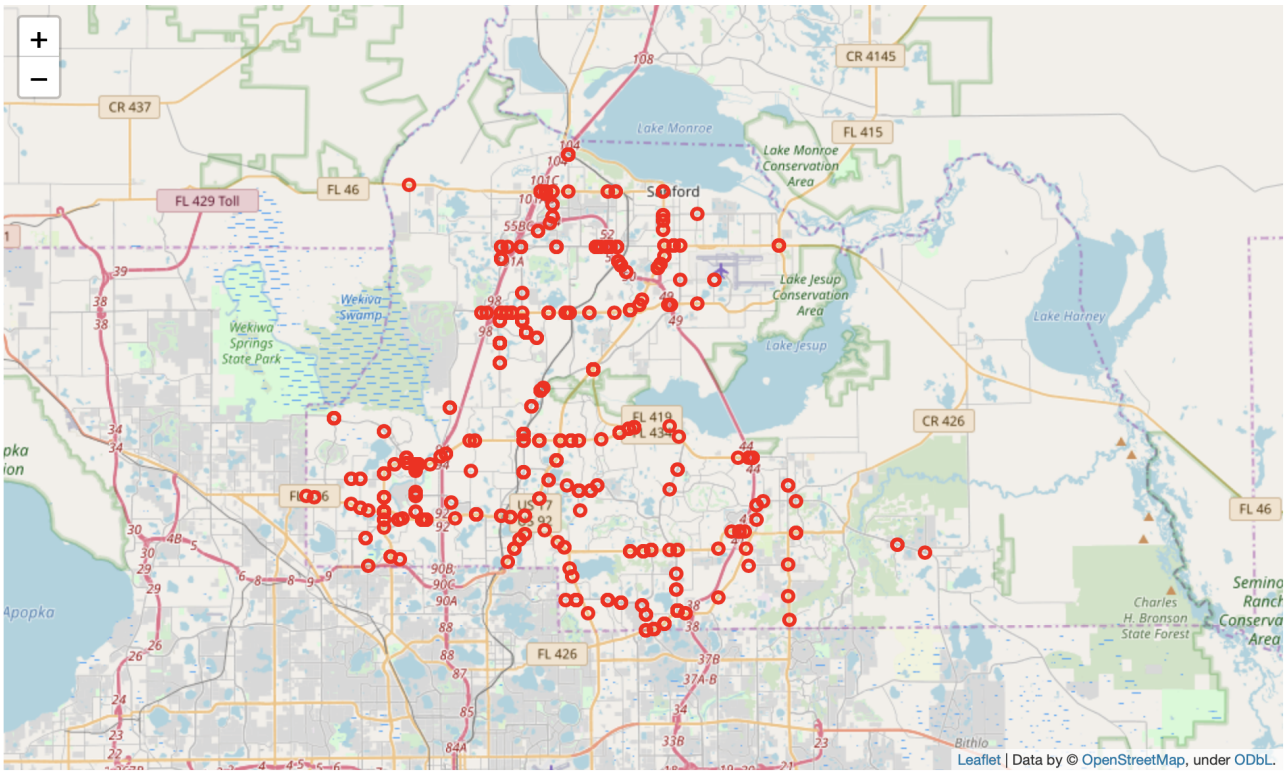

Figure 26: *Locations of all the events of interest generated from Nov. 2018 to Apr. 2019. This plot shows that events occur more frequently at some intersections when compared to others (the number of events at any particular intersection is proportional to number of red markers).*

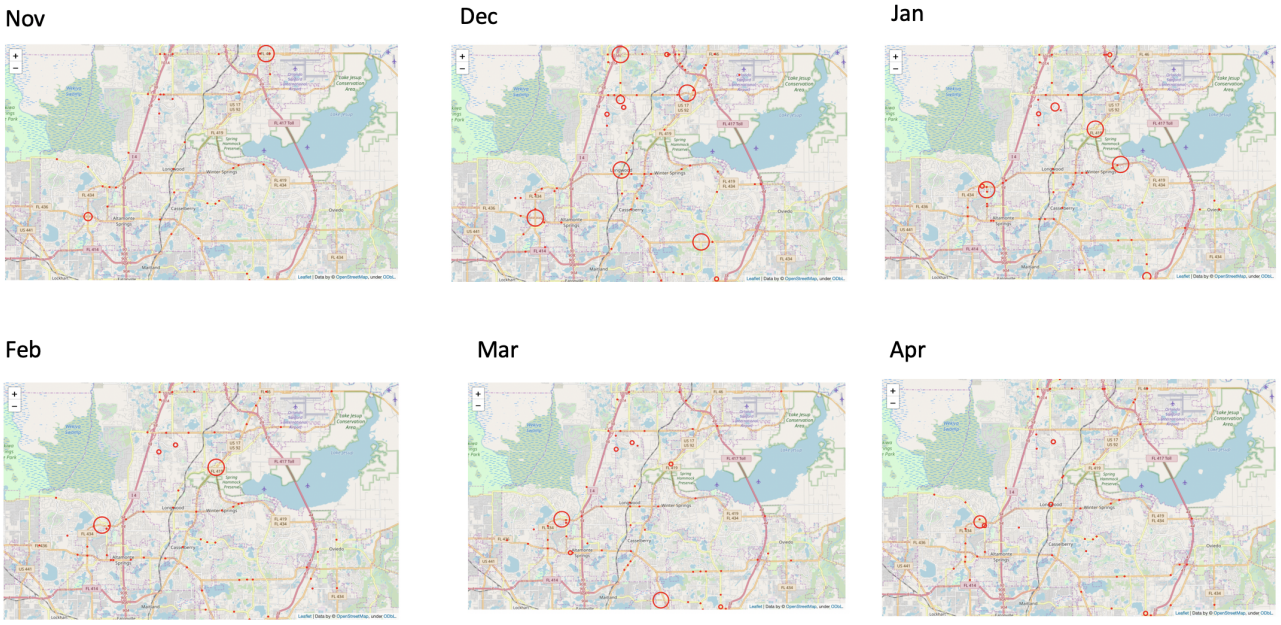

Figure 27: *Locations of all the events of interest generated for each month separately from Nov. 2018 to Apr. 2019. This plot shows temporal consistency of some intersections.*

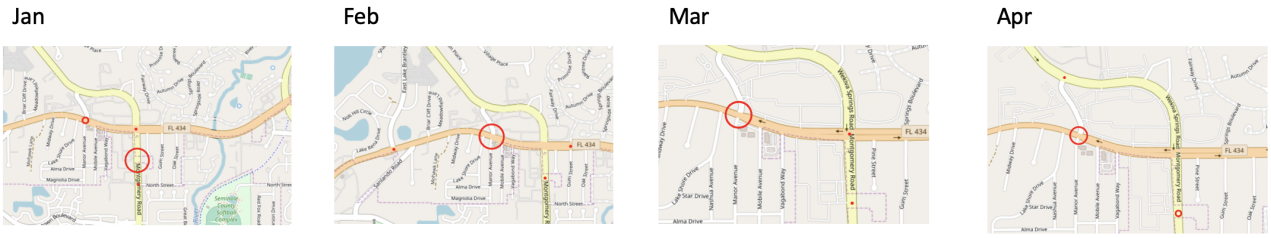

Figure 28: *Intersection where the EOIs are occurring consistently (at least 5 events occurring each month) over the months of January, February, March, and April*

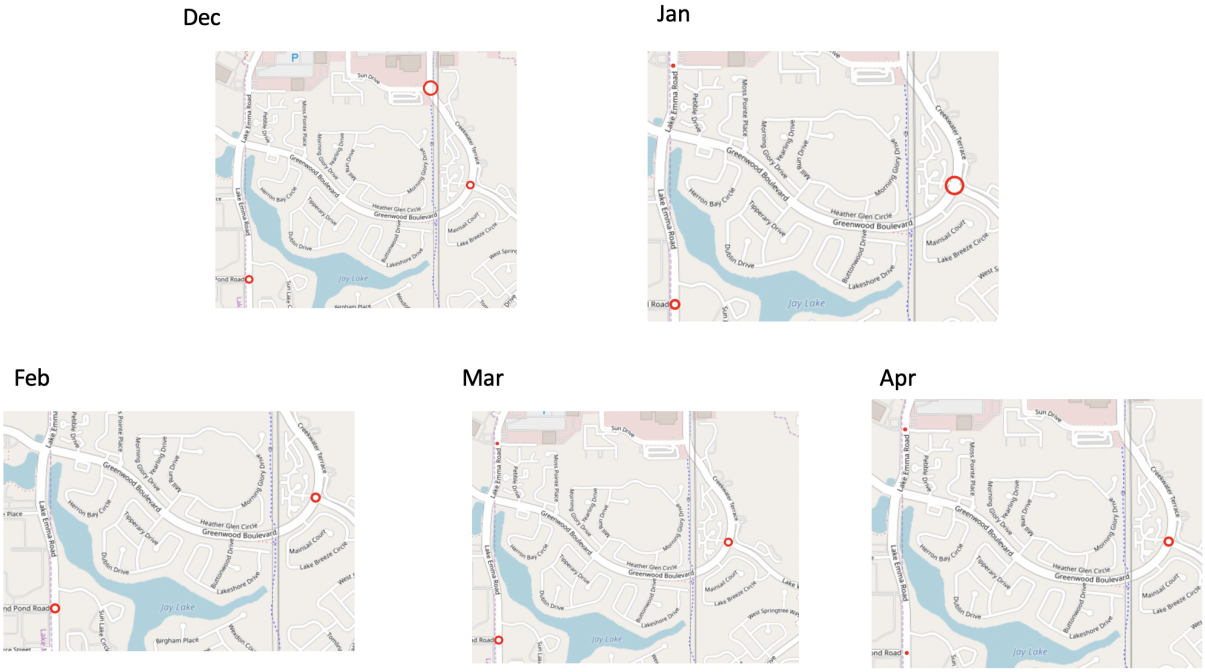

Figure 29: *Intersection where the EOIs are occurring consistently over the months of December, January, February, March, and April*

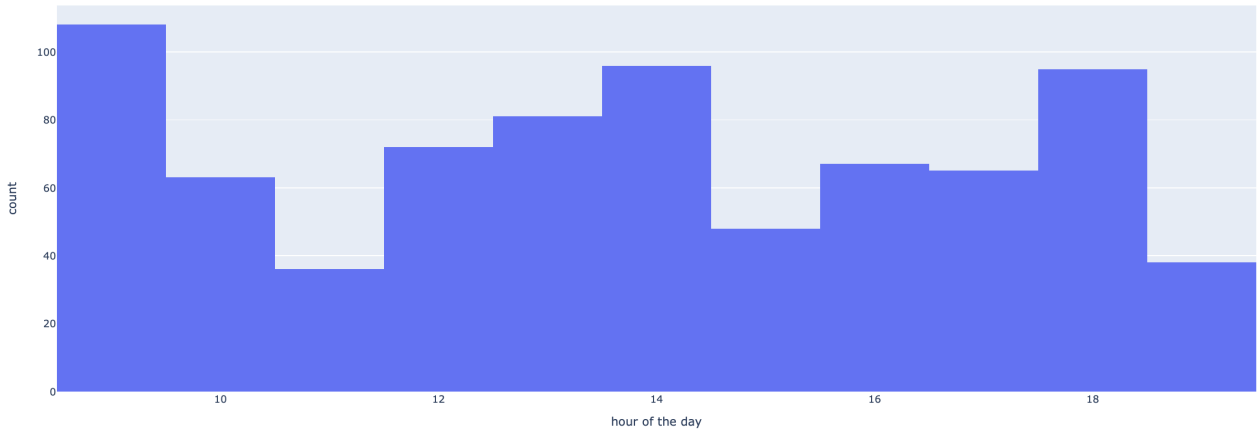

Figure 30: *Histogram of number of EOI based on time of day. More EOI are occurring at times 9-10 AM and 5-6 PM*

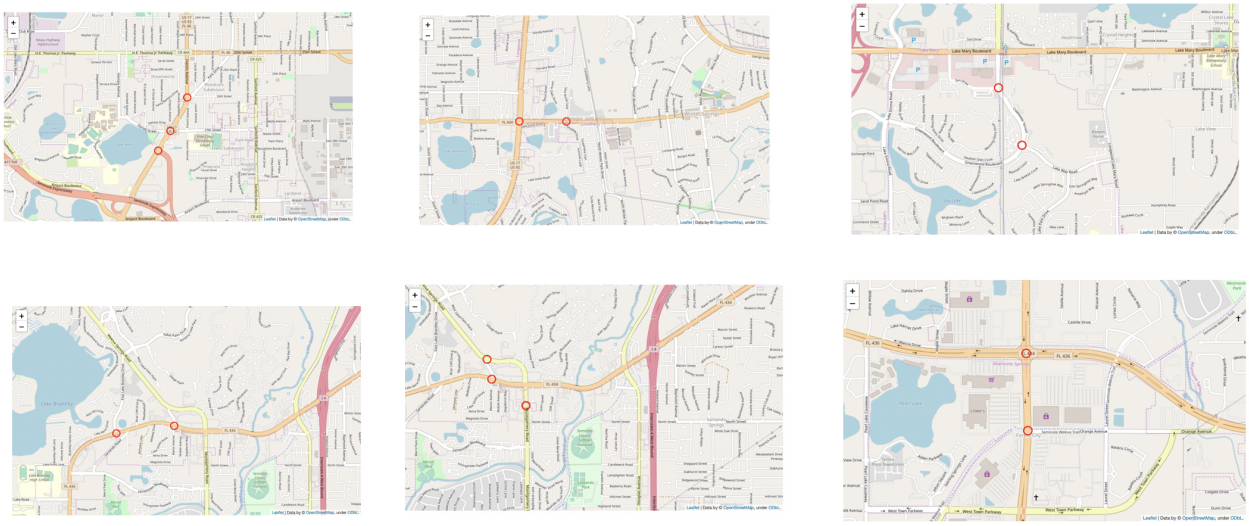

Figure 31: *Co-occurrence of EOIs at some intersections within a 10-min window. This shows spatial correlation at some of the intersections.*

7.2 Spatial Patterns

Spatial pattern discovery complements the earlier case of temporal patterns. Here, we seek to determine the impact of EOIs at one intersection on a nearby one. The first step in this process is to derive the network topology. For each intersection, we derived set of intersections which are nearby based on spatial proximity. The second step is to analyze co-occurrence of interruptions in the nearby intersections (or secondary interruptions).

Algorithm 4 Spatial analysis

function FIND SPATIAL PATTERNS(events, spatial info)
Require: events – list of events with signal IDs and time of occurrence . spatial info – Geo-coordinates for each intersection
  for each signal, time in event **do**
    Obtain nearby intersections for signal using spatial info
    Search for events for in the nearby intersections occurred in a 10 min window
    Store pairs of co occurred events
  end for
end function

Algorithm 4 describes steps to find pairs of co-occurring EOIs given a list of events and geocoordinates for intersections. With this analysis, we find that 52 out of 900 events were coincident with, and thus possibly caused, an interruption in nearby intersections. Figure 31 shows some examples where events co-occurred at nearby intersections within a 10-min window.

8 Accident Analysis

The data on crashes are collected from digitized police crash reports which are housed at Signal Four Analytics (SFA), which contains all traffic crashes reported in the state of Florida since 2006. For this study, we extract the crashes for Seminole County for the months of Nov. 2018 to Apr. 2019. Figure 32 shows a table which has a sample of crash reports. These reports have the following important attributes: location of crash (geocoordinates), time of crash, and severity of crash. For each crash, we obtained nearby intersections which could have been possibly affected by the crash based on spatial proximity (intersections within 600 m of the crash). Table 11 shows a table which has the mappings of crashes to nearby intersections.

8.1 Distribution of Approach-Based Events

We define an approach-based event using the following parameters (X,T): At least X% average reduction in volume on each detector (for an approach) and at least T sec interruption on each detector (for an approach). Figure 33 shows the distribution of approach-based events for different X and T.

8.2 Analyzing Interruptions Caused by Crashes

For analyzing interruptions caused by accidents or crashes, we divide crashes based on severity level. The data from SFA has the following severity levels: property damage only (level 0), injury (level 1), and fatality (level 2). We analyze interruptions caused by these accidents on single-lane approaches, two-lane approaches, and three-lane approaches separately.

For each crash, we check if there is any event generated at the nearby intersections (Table 11) thus obtaining approach-based events that happen to occur within a 5-min interval of the crash time. Figure 34 shows a comparison of the overall distribution of single-lane events vs. the approach-based events that happen to occur within a 5-min interval of crash time. This is done for different levels of crash severity. Similarly, Figure 35, 36 shows the comparison for two-lane and three-lane approaches.

8.3 Analyzing Traffic Volumes at Nearby Intersections

Some of the crashes happen to occur on minor streets or small streets near an intersection. We assume that these crashes won't affect the arrivals on major streets and exclude these from our analysis (we only have detector-to-phase mappings for major streets). For this, we make use of street names from SFA reports and high resolution sensor dataset additional tables. As street names from the sources differ, determining whether a crash happened on a major or minor street is not straightforward. For each street name in the SFA reports, we compare it with all the unique street names from the high resolution sensor data (Additional Tables) and get the street name with which it shares the longest common substring. Table 12 also shows a table which compares the length of longest common substring with the number of accidents. Table 13 shows mappings of street names from crash reports to high resolution sensor data street names when the length of the maximum common substring is

HSMV_Report_Number	Agency_Report_Number	Reporting_Agency	Form_Type	Crash_Date	Crash_Time	City	County	Crash_Street	Intersecting_Street
88801272	2019TA001249	Oviedo PD	Long	01/30/2019	10:34 AM	Oviedo	Seminole	ALAFAYA TRL	
88801271	2019TA001218	Oviedo PD	Short	01/29/2019	02:52 PM	Oviedo	Seminole	LOCKWOOD BLVD	
88801270	2019TA001209	Oviedo PD	Long	01/29/2019	12:24 PM	Oviedo	Seminole	LOCKWOOD BLVD	BIG OAKS BLVD
88801269	2019TA001207	Oviedo PD	Short	01/29/2019	11:37 AM	Oviedo	Seminole	W MITCHELL HAMMOCK RD	SHARON CT
87773259	2019TA001258	Longwood PD	Long	01/30/2019	01:59 PM	Longwood	Seminole	N RONALD REAGAN BLVD	E CHURCH AVE
87773255	2019TA000917	Longwood PD	Long	01/22/2019	03:04 PM	Longwood	Seminole	E SR-434	MYRTLE ST

Vehicle_Dmg_Amt	S4_Mapping	S4_Decimal_Degree_Longitude	S4_Decimal_Degree_Latitude	S4_Albers_X	S4_Albers_Y	S4_Mapping_Date
7000	Mapped, Not On Network	-81.2092920699405	28.652563758939	672252.6964969510	519347.7409632880	1/31/2019 4:59:44 PM
2500	Mapped, Not On Network	-81.1797225507302	28.6580076214682	675123.0799309910	520017.7784222180	1/31/2019 4:59:44 PM
15000	Mapped, On Network	-81.1793178522321	28.6534562013973	675174.1128556370	519513.3571527360	1/31/2019 4:59:44 PM
2000	Mapped, On Network	-81.2103461487269	28.6556622156796	672142.0978081270	519689.42433331	1/31/2019 4:59:44 PM
500	Mapped, On Network	-81.3462549448013	28.7005400019544	658778.4948583440	524379.5140340440	1/31/2019 4:59:44 PM
1000	Mapped, On Network	-81.3439794501554	28.6978582256081	659006.7683515040	524086.5380577910	1/31/2019 4:59:45 PM

Estimated_Damages	Weather_Condition	Light_Condition	Street_Number	Crash_Type_Detailed	Crash_Type_Dir	Crash_Severity	Within_City_Limits	Manner_of_Collision
$7,000.00	Cloudy	Daylight	1268	Backed Into	W	Property Damage Only	Y	Angle
$2,500.00	Clear	Daylight	1016	Backed Into	E	Property Damage Only	Y	Rear to Side
$15,000.00	Clear	Daylight		Rear End	S	Property Damage Only	Y	Front to Rear
$2,000.00	Clear	Daylight		Rear End	E	Property Damage Only	Y	Front to Rear
$800.00	Cloudy	Daylight		Pedestrian		Property Damage Only	Y	
$1,000.00	Clear	Daylight		Rear End	W	Property Damage Only	Y	Front to Rear

Figure 32: *SFA crash reports showing selected, key attributes.*

Table 11: *Table showing mapping of each crash to the nearby intersections. Attributes include (a) HSMV_No: unique identifier of crash from SFA reports, (b) location: location of the crash, (c) C_Date: crash date, (d) C_Time: crash time, (e) Crash_Severity: severity level of crash, and (f) near_sig: intersections within 600 m of crash.*

HSMV_No	location	C_Date	C_Time	C_Severity	near_sig
88801272	POINT (-81.20929 28.65256)	01/30/2019	10:34 AM	Property Damage Only	[1395, 1400, 2627]
88801271	POINT (-81.17972 28.65801)	01/29/2019	02:52 PM	Property Damage Only	[1870]
88801270	POINT (-81.17932 28.65346)	01/29/2019	12:24 PM	Property Damage Only	[1870]
88958477	POINT (-81.32735 28.69639)	02/11/2019	02:15 PM	Injury	[1130, 1127]
87773274	POINT (-81.35851 28.71469)	02/08/2019	05:43 PM	Injury	[1285]
88702737	POINT (-81.36583 28.64555)	02/08/2019	03:20 PM	Injury	[1555]
88614479	POINT (-81.32808 28.65128)	01/07/2019	09:15 AM	Property Damage Only	[1595, 1600]
88907150	POINT (-81.33461 28.75636)	01/08/2019	05:45 PM	Property Damage Only	[2065, 2125]
88907146	POINT (-81.34737 28.77745)	01/07/2019	06:45 PM	Injury	[2180, 1995, 2000]

T →

X ↓	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	140452	75864	39179	22341	12767	8008	4416	3354	1962	1470	1014	878	2934
30	58416	28412	13059	6897	3841	2463	1434	1056	699	539	437	345	1705
40	17867	6818	2908	1514	810	487	370	266	185	136	121	114	748
50	4485	1476	549	303	162	103	75	68	48	32	22	27	225
60	974	316	120	105	43	41	20	11	6	5	4	11	50
70	217	119	28	28	17	9	6	11	8	9	4	2	30
80	97	60	10	14	12	8	10	12	9	2	0	0	17
90	63	39	7	10	2	4	5	7	1	0	1	1	15
100	22	14	1	4	0	1	2	1	0	0	0	0	3

X - At least X% Avg reduction in volume on each detector for an approach
T – At least T sec interruption on each detector for an approach

Figure 33: *Frequencies for different dips based on at least X% average reduction in volume on each detector and at least a T-second interruption on each detector (for an approach)*

Overall distribution – Single Lane approaches 117 approaches

X ↓	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	43245	27975	16490	10552	6636	4237	2487	1949	1201	857	604	526	1804
30	27062	14699	7504	4201	2482	1606	1016	734	470	366	279	223	1154
40	11587	4944	2175	1205	625	381	274	205	143	103	97	85	541
50	3416	1207	478	255	128	87	70	61	42	28	20	22	159
60	734	265	100	85	38	32	19	9	6	4	1	8	34
70	158	109	22	17	12	8	5	9	7	8	5	2	16
80	82	56	9	13	12	7	7	12	6	2	0	0	12
90	59	38	7	10	2	3	5	7	1	0	1	0	8
100	22	14	1	4	0	1	2	1	0	0	0	0	2

A

Property damage accidents (level 0)

X ↓	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	117	93	60	42	22	12	3	5	6	1	3	1	2
30	73	44	23	17	9	8	6	1	0	1	0	0	2
40	19	20	3	9	0	1	3	1	1	2	0	1	1
50	8	6	1	3	0	1	1	0	1	0	0	0	0
60	0	4	0	2	0	0	0	0	0	0	0	0	0
70	1	0	0	0	0	0	0	0	0	0	0	0	0
80	1	0	0	0	0	0	0	0	1	0	0	0	0
90	1	1	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

B

Injury accidents (level 1)

X ↓	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	24	27	18	7	9	5	4	1	5	8	2	1	5
30	24	18	4	2	0	0	0	2	0	0	0	2	5
40	9	7	0	1	4	1	1	0	0	0	0	2	0
50	0	1	0	0	0	0	0	0	0	0	0	0	6
60	0	2	0	0	0	0	0	0	2	0	0	0	0
70	1	0	0	0	0	0	0	0	2	1	2	0	3
80	0	0	0	0	0	0	0	0	0	0	0	0	2
90	0	1	0	0	0	1	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

C

Fatality accidents (level 2)

X ↓	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	2	0	0	0	0	0	0	0	0	0	0	0
40	0	0	2	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

D

X ↓ T →

Figure 34: *Comparing the overall distribution of single-lane events with distribution of events that may have been caused due to crashes (X: at least X% average reduction in volume on each detector for an approach; T: at least a T-sec interruption on each detector for an approach): (a) overall distribution of approach-based events for single-lane approaches; (b) distribution of approach-based events based for level 0 accidents; (c) distribution of approach-based events for level 1 accidents; (d) distribution of approach-based events for level 2 accidents.*

Overall distribution - Two Lane approaches 267 approaches

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	75984	39324	18311	9802	5152	3184	1673	1258	680	553	373	319	1097
30	27477	12276	4933	2453	1219	783	379	291	214	163	143	112	519
40	5651	1698	644	291	173	94	84	53	41	29	21	26	201
50	976	232	61	39	22	12	5	5	3	4	2	4	77
60	212	42	14	10	0	6	1	2	0	1	3	3	17
70	53	10	5	3	1	1	1	2	1	1	0	0	14
80	12	3	1	1	0	1	2	0	3	0	0	0	7
90	4	0	0	1	0	1	1	0	0	0	0	1	7
100	0	0	0	0	0	0	0	0	0	0	0	0	1

A

Property damage accidents (level 0)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	251	147	63	32	28	11	6	2	2	3	1	0	4
30	93	21	8	10	3	4	0	4	0	1	0	0	4
40	19	6	3	2	2	1	0	0	0	0	0	0	2
50	8	2	0	2	0	0	0	0	0	0	0	0	0
60	3	0	0	0	0	0	0	0	0	0	0	0	0
70	2	0	1	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	1
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	1

B

Injury accidents (level 1)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	79	33	9	5	4	4	0	2	2	0	0	0	0
30	24	9	10	6	0	0	2	0	0	1	0	1	0
40	7	0	1	0	0	0	1	0	0	0	0	0	4
50	0	0	0	0	0	0	0	0	0	0	0	0	1
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

C

Fatality accidents (level 2)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	0	0	0	0	0	0	0	0	0	0	0	0	0
30	4	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	4
100	0	0	0	0	0	0	0	0	0	0	0	0	0

D

X

T

Figure 35: *Comparing the overall distribution of two-lane events with distribution of events that may have been caused by crashes (X: at least X% average reduction in volume on each detector for an approach; T: at least a T-sec interruption on each detector for an approach): (a) overall distribution of approach-based events for two-lane approaches; (b) distribution of approach-based events for level 0 accidents; (c) distribution of approach-based events for level 1 accidents; (d) distribution of approach-based events for level 2 accidents.*

Overall distribution – Three Lane approaches (123 approaches)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	19939	8065	4216	1965	954	604	257	161	107	61	51	39	83
30	3479	1308	588	234	125	71	34	33	16	11	15	11	48
40	528	142	82	17	13	12	9	5	3	3	3	3	22
50	73	35	11	8	12	4	1	2	3	0	0	1	7
60	25	8	6	11	5	3	0	0	0	0	0	0	0
70	6	0	1	8	4	0	0	0	0	0	0	0	1
80	3	1	0	0	0	0	1	0	0	0	0	0	0
90	0	1	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

A

Property damage accidents (level 0)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	81	38	13	9	1	11	2	1	0	0	0	0	1
30	17	9	1	2	0	0	0	0	0	0	0	0	0
40	0	2	0	0	0	0	0	0	0	0	0	0	0
50	1	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	1
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

B

Injury accidents (level 1)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	30	20	2	1	4	0	0	0	0	0	0	0	1
30	10	4	2	0	0	0	0	0	0	0	0	0	0
40	0	0	2	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

C

Fatality accidents (level 2)

	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500
20	0	0	0	0	0	0	0	0	0	0	0	0	0
30	0	0	0	0	0	0	0	0	0	0	0	0	0
40	0	0	0	0	0	0	0	0	0	0	0	0	0
50	0	0	0	0	0	0	0	0	0	0	0	0	0
60	0	0	0	0	0	0	0	0	0	0	0	0	0
70	0	0	0	0	0	0	0	0	0	0	0	0	0
80	0	0	0	0	0	0	0	0	0	0	0	0	0
90	0	0	0	0	0	0	0	0	0	0	0	0	0
100	0	0	0	0	0	0	0	0	0	0	0	0	0

D

X

T

Figure 36: *Comparing the overall distribution of three-lane events with distribution of events that may have caused due to crashes (X: at least X% average reduction in volume on each detector for an approach; T: at least a T-sec interruption on each detector for an approach): (a) overall distribution of approach-based events for three-lane approaches; (b) distribution of approach-based events based for level 0 accidents; (c) distribution of approach-based events for level 1 accidents; (d) distribution of approach-based events for level 2 accidents.*

Table 12: *Length of common maximum substring (between street names in crash reports and high resolution sensor data street names) with number of accidents: (a) L_LCS_Crash_high resolution sensor data: length of common substring; (b) Number of accidents: filtered number of accidents*

L_LCS_Crash_high resolution sensor data	**Number of accidents**
>3	5768 out of 6708
>4	4359 out of 6708
>5	2904 out of 6708
>6	2036 out of 6708
>7	1597 out of 6708
>8	940 out of 6708

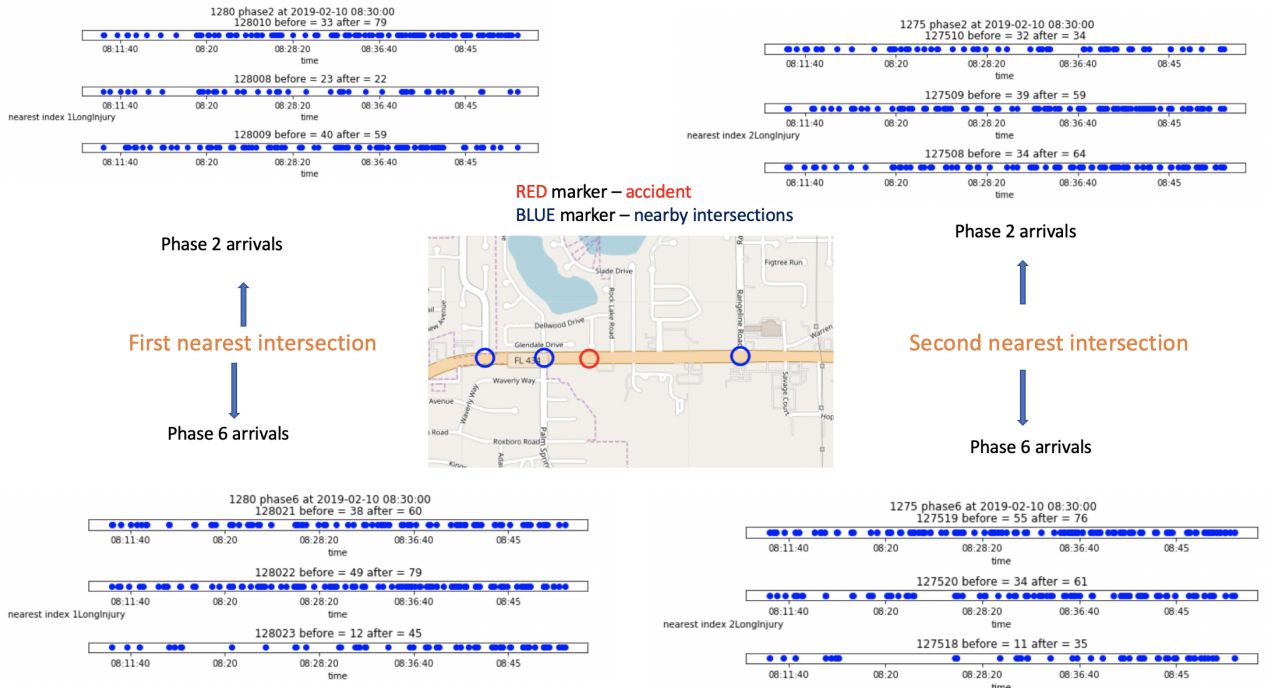

Figure 37: *Crash location along with locations of derived nearby intersections. Also, arrivals on each detector for first- and second-nearest intersections 20 min before and 20 min after the crash time are shown.*

greater than five. We plot arrivals on each detector for first- and second-nearest intersections 20 minutes before and 20 minutes after the crash time to see if an accident has caused an interruption at any nearby intersection. From Figures 37, 38,39,40, and 41, we can see that accidents didn't cause any interruption in arrivals at nearby intersections.

Based on our analysis presented in subsection 8.2 and subsection 8.3, we see that most of the accidents are not causing major interruptions in traffic.

Table 13: *Mappings of street names from crash reports to high resolution sensor data street names when length of maximum common substring is greater than 5. This table has the following attributes: (a) HSMV_No: HSMV_report_number from crash reports, (b) C_Date: crash date, (c) C_Time: crash time, (d) C_Street: crash street reported in SFA, and (e) S_name: mapped street name from high resolution sensor data, (f) LCS: longest common substring of crash street, street name from high resolution sensor data.*

HSMV_No	C_Date	C_Time	C_Street	S_name	LCS
88801272	01/30/2019	10:34 AM	ALAFAYATRL	alafayawoods	alafaya
88801271	01/29/2019	02:52 PM	LOCKWOODBLVD	lockwoodadt	lockwood
88801270	01/29/2019	12:24 PM	LOCKWOODBLVD	lockwoodadt	lockwood
88801282	02/06/2019	08:26 AM	LOCKWOODBLVD	lockwoodadt	lockwood
88702663	01/25/2019	06:55 PM	MAITLANDBLVD	maitlandave	maitland
88614466	01/01/2019	02:28 PM	LAURAST	laurastadt	laurast
87773274	02/08/2019	05:43 PM	LAKEEMMARD	lkemmard	emmard
88702737	02/08/2019	03:20 PM	NMAITLANDAVE	maitlandave	maitlandave
88927573	02/12/2019	02:28 PM	RINEHARTRD	rinehartrd	rinehartrd
88614610	02/08/2019	06:45 PM	OXFORDRDN	oxford	oxford
88770185	02/12/2019	11:12 AM	REDBUGLAKERD	redbuglkrd	redbugl
88770133	02/08/2019	12:58 AM	SSYLVANLAKEDR	lakedradt	lakedr
88927352	01/02/2019	08:59 AM	COUNTRYCLUBRD	countryclub	countryclub
88927348	01/01/2019	03:58 PM	WSEMINOLEBLVD	seminol ablvdadt	seminol
87291537	01/07/2019	06:50 AM	US1792	us1792	us1792
87291536	01/06/2019	09:45 AM	WTOWNPKWY	west town pkwyadt	townpkwy

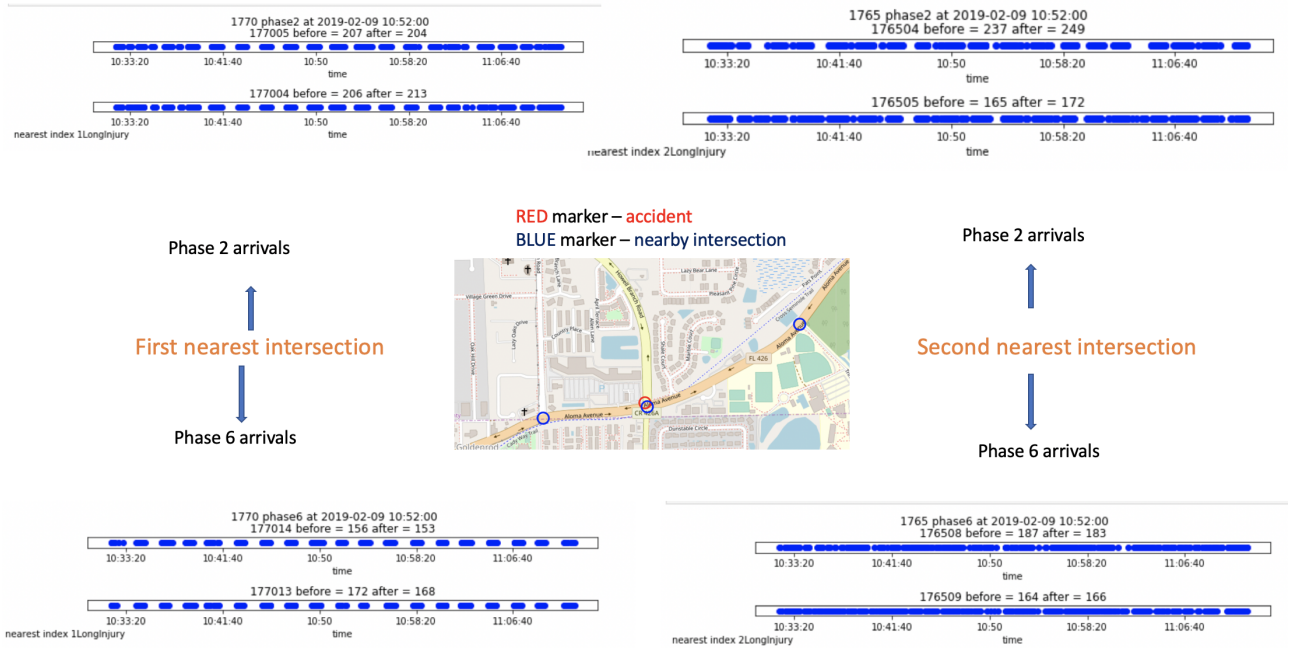

Figure 38: *Crash location along with locations of derived nearby intersections. Also, arrivals on each detector for first- and second-nearest intersections 20 min before and 20 min after the crash time are shown.*

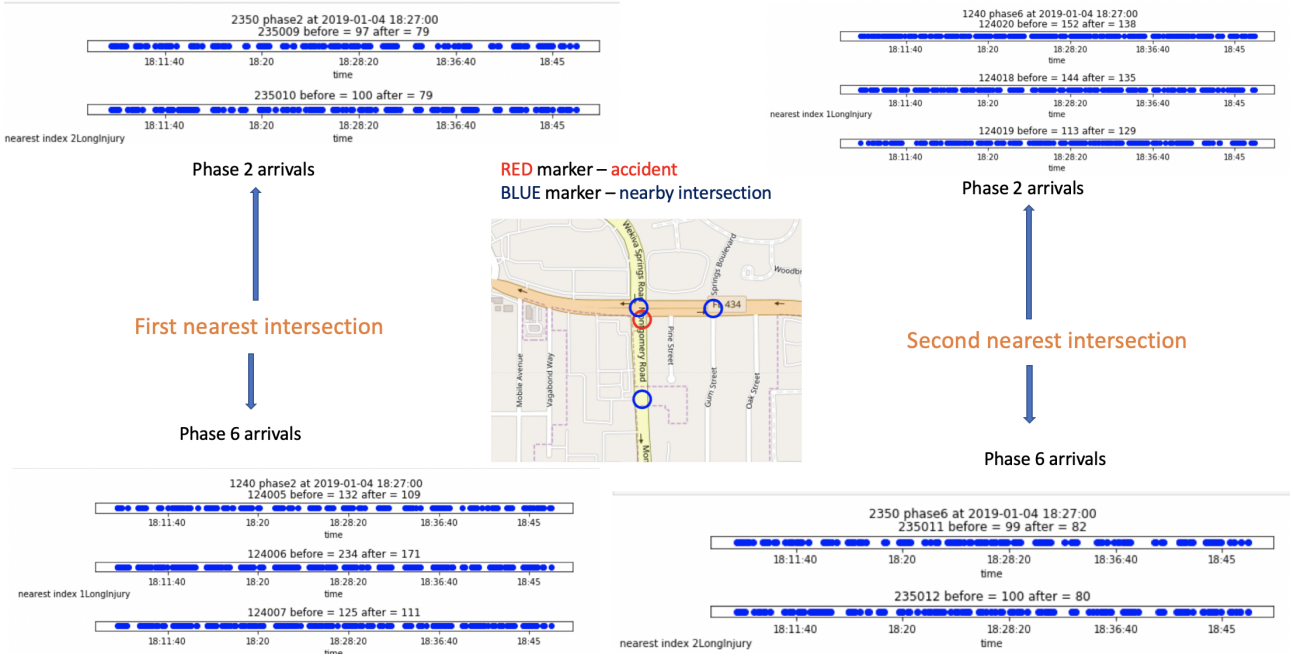

Figure 39: *Crash location along with locations of derived nearby intersections. Also, arrivals on each detector for first- and second-nearest intersections 20 min before and 20 min after the crash time are shown.*

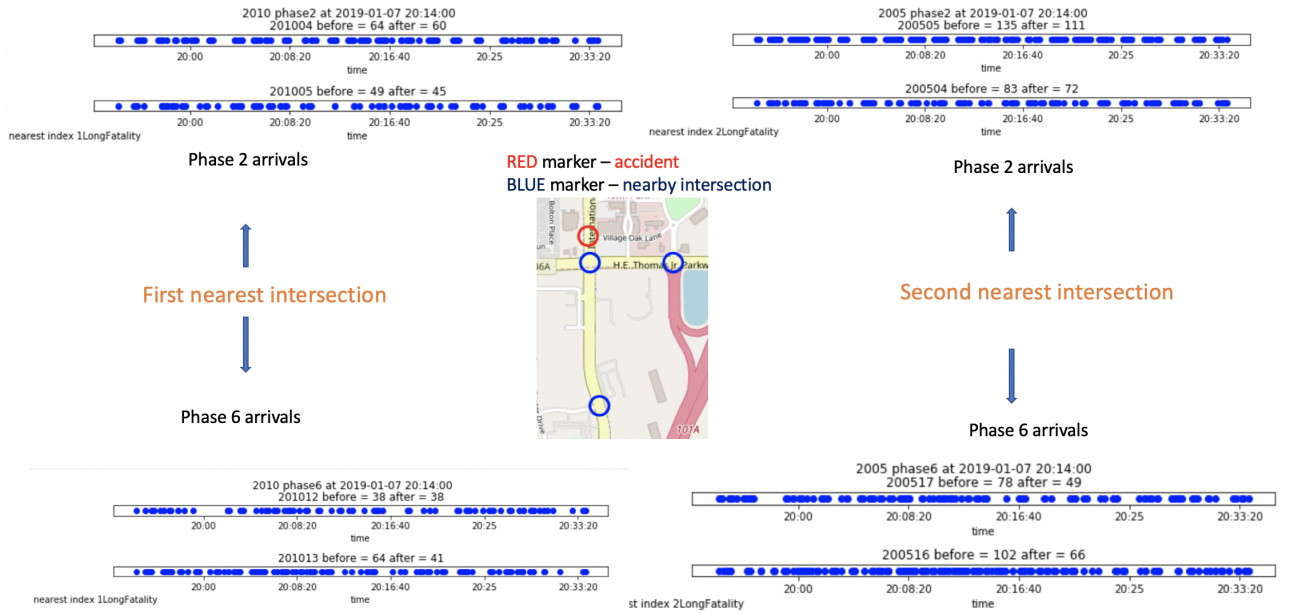

Figure 40: *Crash location along with locations of derived nearby intersections. Also, arrivals on each detector for first- and second-nearest intersections 20 min before and 20 min after the crash time are shown.*

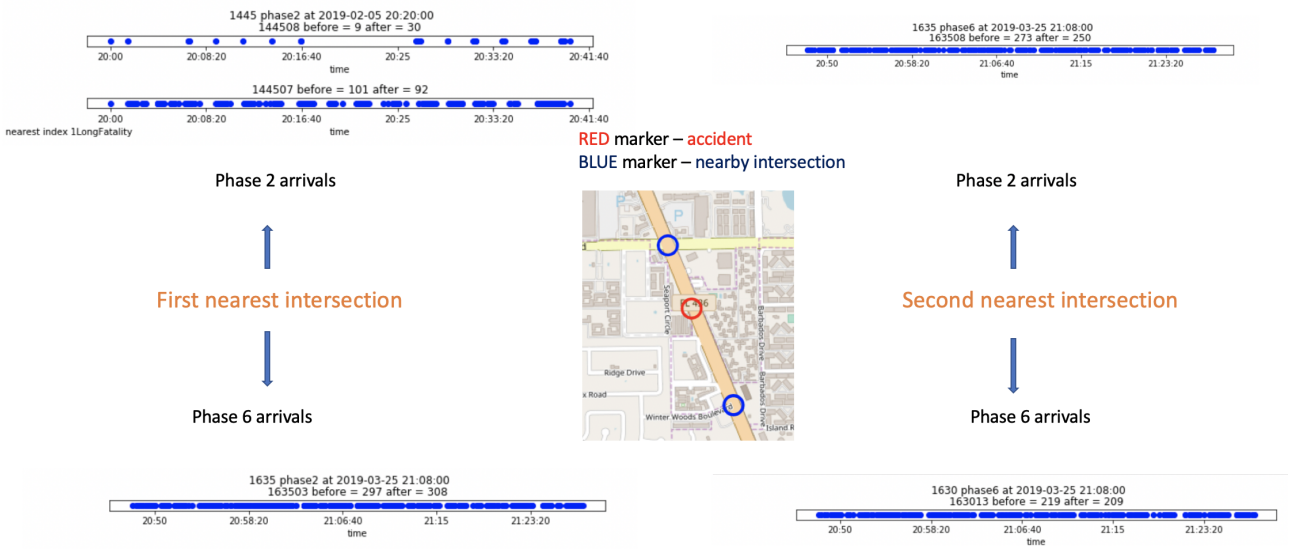

Figure 41: *Crash location along with locations of derived nearby intersections. Also, arrivals on each detector for first- and second-nearest intersections 20 min before and 20 min after the crash time are shown.*

9 Conclusion

The focus of this work is on the detection and prediction of traffic interruptions without the need for labeled data (from police reports and the like). After first defining events of interest corresponding to traffic interruptions using large deviations of traffic volumes and the build-up of significant delays, we constructed an event prediction approach to discover these interruptions from signalized intersection datasets. The approach—using time series analysis—examined (cumulative) approach volumes at intersections and at specific time points and compared them to historical (cumulative) approach volumes at similar time points (hour of the day and/or day of the week). This proved capable of predicting the occurrence of events of interest (EOIs) with high accuracy. We also performed a spatiotemporal analysis of the EOIs to find recurrent patterns in the said events in order to obtain human-understandable summarizations of traffic interruptions. Finally, we analyzed traffic crash reports from SFA to see if accidents cause interruptions at nearby intersections. Based on our analysis presented in subsection 8.2 and subsection 8.3, we see that most of the accidents are not causing major interruptions in traffic.

Overall, we were able to

1. Establish a problem-solving platform based on multiple, new data streams.
2. Establish a data-cleansing and data-fusing platform to prepare data sets
3. Utilize new data sources and techniques to solve currently solvable, observable, and verifiable problems, allowing us to evaluate the effectiveness of the innovative solution.
4. Establish confidence in a data-centric, problem-solving platform on which to solve future problems that cannot be easily observed or verified.

References

Ahmed, S. A. and Cook, A. R. (1980). Time series models for freeway incident detection. *Transp. J. Eng. ASCE*, 106(6):731–745.

Anbaroglu, B., Heydecker, B., and Cheng, T. (2014). Spatio-temporal clustering for non-recurrent traffic congestion detection on urban road networks. *Transp. Res., Part C*, 48:47–65.

Asare, S. K., Adu-Gyamfi, Y., Attoh-Okine, N., and Park, H. (2013). Adaptive freeway incident detection algorithm using the Hilbert-Huang transform. In *TRB 92nd Annual Meeting*, Washington, D.C. Transp. Res. Board.

Balke, K., Dudek, C. L., and Mountain, C. E. (1996). Using probe-measured travel time to detect major freeway incidents in Houston, Texas. *Transp. Res. Rec.*, 1554:213–220.

Bhandari, N., Koppelman, F. S., Schofer, J. L., Sethi, V., and Ivan, J. N. (1995). Arterial incident detection integrating data from multiple sources. *Transp. Res. Rec.*, 1510:60–69.

Byun, S. C., Choi, D. B., Ahn, B. H., and Ko., H. (1999). Traffic incident detection using evidential reasoning based data fusion. In *Proc. 6th World Cong. Intell. Transp. Sys.*, Washington, D.C. ITS America.

Chang, E. C. P. and Wang, S. H. (1994). Improved freeway incident detection using fuzzy set theory. *Transp. Res. Rec.*, 1453:75–82.

Chen, C.-H. and Chang, G.-L. (1993). A dynamic real-time incident detection system for urban arterials-system architecture and preliminary results. In *Pacific Rim TransTech Conference: Advanced Technologies*, New York, NY. ASCE. Pp. 98-104.

Chen, S., Wang, W., and Van Zuylen, H. (2009). Construct support vector machine ensemble to detect traffic incident. *Expert Sys. Appl.*, 36(8):10976–10986.

Cheu, L., Ruey, H. Q., and Lee, D.-H. (2002). Mobile sensor and sample-based algorithm for freeway incident detection. *Transp. Res. Rec.*, 1811:12–20.

Cheu, R. L. and Ritchie, S. G. (1995). Automated detection of lane-blocking freeway incidents using artificial neural networks. *Transp. Res., Part C*, 3(6):371–388.

Cheu, R. L., Srinivasan, D., and Loo, W. H. (2004). Training neural networks to detect freeway incidents by using particle swarm optimization. *Transp. Res. Rec.*, 1867:11–18.

Cook, A. R. and Cleveland, D. E. (1974). Detection of freeway capacity-reducing incidents by traffic stream measurements. *Transp. Res. Rec.*, 495:1–11.

Daly, E. M., Lecue, F., and Bicer, V. (2013). Westland row why so slow?: fusing social media and linked data sources for understanding real-time traffic conditions. In *In Proc. 2013 Int. Conf. Intell. User Interfaces*, New York, NY. ACM. Pp. 203-212.

D'Andrea, E. and Marcelloni, F. (2017). Detection of traffic congestion and incidents from GPS trace analysis. *Expert Sys. Appl.*, 73:43–56.

Dia, H. and Rose, G. (1997). Development and evaluation of neural network freeway incident detection models using field data. *Transp. Res., Part C*, 5(5):313–331.

Dia, H. and Thomas, K. (2011). Development and evaluation of arterial incident detection models using fusion of simulated probe vehicle and loop detector data. *Information Fusion*, 12(1):20–27.

Dudek, C. L., Messer, C. J., and Nuckles, N. B. (1974). Incident detection on urban freeways. *Transp. Res. Rec.*, 495:12–24.

El Faouzi, N.-E. and Klein, L. A. (2016). Data fusion for ITS: techniques and research needs. *Transp. Res. Proc.*, 15:495–512.

El Faouzi, N. E. and Lesort, J. B. (1995). Travel time estimation on urban networks from traffic data and on-board trip characteristics. In *Proc. 2nd World Congress Intell. Transp. Sys.*, Tokyo, Japan. Vehicle, Road and Traffic Intell. Soc. Pp. 88-93.

El Hatri, C. and Boumhidi, J. (2018). Fuzzy deep learning based urban traffic incident detection. *Cogn. Sys. Res.*, 50:206–213.

Gall, A. I. and Hall, F. L. (1989). Distinguishing between incident congestion and recurrent congestion: A proposed logic. *Transp. Res. Rec.*, 1232:1–8.

Gu, Y., Qian, Z. S., and Chen, F. (2016). From Twitter to detector: Real-time traffic incident detection using social media data. *Transp. Res., Part C*, 67:321–342.

Han, L. D. and May, A. D. (1989). *Automatic detection of traffic operational problems on urban arterials* (Calif. PATH Rpt. UCB-ITS-RR-89-15). Institute of Transportation Studies, U.C. Berkeley, Berkeley, CA.

Hawas, Y. E. (2007). A fuzzy-based system for incident detection in urban street networks. *Transp. Res., Part C*, 15(2):69–95.

Hellinga, B. and Knapp, G. (2000). Automatic vehicle identification technology-based freeway incident detection. *Transp. Res. Rec.*, 1727:142–153.

Ivan, J. N. (1997). Neural network representations for arterial street incident detection data fusion. *Transp. Res., Part C*, 5(3-4):245–254.

Ivan, J. N., Schofer, J. L., Koppelman, F. S., and Massone, L. L. E. (1995). Real-time data fusion for arterial street incident detection using neural networks. *Transp. Res. Rec.*, 1497:27–35.

Ivan, J. N. and Sethi, V. (1998). Data fusion of fixed detector and probe vehicle data for incident detection. *Comput.-Aided Civil Infrastruc. Eng.*, 13(5):329–337.

Jeong, Y. S., Castro-Neto, M., J. M. K., and Han, L. D. (2011). A wavelet-based freeway incident detection algorithm with adapting threshold parameters. *Transp. Res., Part C*, 19(1):1–19.

Jin, X., Cheu, R. L., and Srinivasan, D. (2002). Development and adaptation of constructive probabilistic neural network in freeway incident detection. *Transp. Res., Part C*, 10(2):121–147.

Kamran, S. and Haas, O. (2007). A multilevel traffic incidents detection approach: Identifying traffic patterns and vehicle behaviours using real-time GPS data. In *In 2007 IEEE Intelligent Vehicles Symposium*, Piscataway, NJ. IEEE. Pp. 912-917.

Khan, S. I. and Ritchie, S. G. (1998). Statistical and neural classifiers to detect traffic operational problems on urban arterials. *Transp. Res., Part C*, 6(5-6):291–314.

Kinoshita, A., Takasu, A., and Adachi, J. (2014). Traffic incident detection using probabilistic topic model. In *Proc. EDBT/ICDT 2014 Joint Workshop*, Aachen, Germany. RWTH Aachen Univ. Pp. 323-330.

Klein, L., Yi, P., and Teng, H. (2002). Decision support system for advanced traffic management through data fusion. *Transp. Res. Rec.*, 1804:173–178.

Klein, L. A. (2000). Dempster-Shafer data fusion at the traffic management center. In *TRB 79th Annual Meeting*, Washington, D.C. Transp. Res. Board. 20 pp., CD-ROM.

Lee, J.-T. and Taylor, W. C. (1999). Application of a dynamic model for arterial street incident detection. *Intell. Transp. Sys. J.*, 5(1):53–70.

Lee, S., Krammes, R. A., and Yen, J. (1998). Fuzzy-logic-based incident detection for signalized diamond interchanges. *Transp. Res., Part C*, 6(5-6):359–377.

Levin, M. and Krause, G. M. (1978). Incident detection: a Bayesian approach. *Transp. Res. Rec.*, 682:52–58.

Li, Y. and McDonald, M. (2005). Motorway incident detection using probe vehicles. *Proc. Inst. Civ. Eng.-Transp.*, 158(1):11–15.

Liang, G. (2015). Automatic traffic accident detection based on the Internet of Things and support vector machine. *Int. J. Smart Home*, 9(4):97–106.

Lin, W.-H. and Daganzo, C. F. (1997). A simple detection scheme for delay-inducing freeway incidents. *Transp. Res., Part A*, 31(2):141–155.

Lingras, P. and Adamo, M. (1996). Average and peak traffic volumes: Neural nets, regression, and factor approaches. *J. Comput. Civil Eng.*, 10(4):300–306.

Liu, Q., Lu, J., Chen, S., and Zhao, K. (2014). Multiple naïve Bayes classifiers ensemble for traffic incident detection. *Math. Prob. Eng.*, Article ID 383671.

Mahmassani, H. S., Haas, C., Zhou, S., and Peterman, J. (2014). *Evaluation of incident detection methodologies* (FHWA/TX-00/1795-1). FHWA, Washington, D.C.

Michalopoulos, P. G. (1991). Vehicle detection video through image processing: the autoscope system. *IEEE Trans. Veh. Tech.*, 40(1):21–29.

Michalopoulos, P. G., Jacobson, R. D., Anderson, C. A., and DeBruycker, T. B. (1993). Automatic incident detection through video image processing. *Traff. Eng. Cont.*, 34(2):66–75.

Mouskos, K. C., Niver, E., Lee, S., Batz, T., and Dwyer, P. (1999). Transportation operation coordinating committee system for managing incidents and traffic: evaluation of the incident detection system. *Transp. Res. Rec.*, 1679:50–57.

Mussa, R. and Upchurch, J. (2000). Modeling incident detection using vehicle-to-roadside communications systems. *J. Transp. Res. Forum*, 39(4):117–127.

Mussa, R. N. and Upchurch, J. E. (1999). Simulation assessment of incident detection by cellular phone call-in programs. *Transp.* (Neth.), 26(4):399–416.

Niver, E., Mouskos, K. C., Batz, T., and Dwyer, P. (2000). Evaluation of the TRANSCOM's system for managing incidents and traffic (TRANSMIT). *IEEE Trans. Intell. Transp. Sys.*, 1(1):15–31.

Park, H. and Haghani, A. (2016). Real-time prediction of secondary incident occurrences using vehicle probe data. *Transp. Res., Part C*, 70:69–85.

Parkany, E. and Xie, C. (2005). *A complete review of incident detection algorithms their deployment: what works and what doesn't*. New England Transportation Consortium, Storrs, CT.

Payne, H. J. (1975). Freeway incident detection based upon pattern classification. In *Proc. IEEE Conf. Decis. Control, incl. 14th Symp. Adaptive Processes*, Piscataway, NJ. IEEE. Pp. 688-692.

Payne, H. J. (1976). *Development and and testing of incident detection algorithms* (FHWA-RD-76-20). FHWA, Washington, D.C.

Payne, H. J. and Tignor, S. C. (1978). Freeway incident detection algorithms based on decision trees with states. *Transp. Res. Rec.*, 682:30–37.

Petty, K. F., Skabardonis, A., and Varaiya, P. P. (1997). Incident detection with probe vehicles: performance, infrastructure requirements and feasibility. *IFAC Proceedings Volumes*, 30(8):125–130.

Ren, J. S. J., Wang, W., Wang, J., and Liao, S. (2012). An unsupervised feature learning approach to improve automatic incident detection. In *Proc. 15th IEEE Conf. Intell. Transp. Sys.*, Piscataway, NJ. IEEE. Pp. 172-177.

Roy, P. and Abdulhai, B. (2003). GAID: Genetic adaptive incident detection for freeways. *Trans. Res. Rec.*, 1856:96–105.

Samant, A. and Adeli, H. (2000). Feature extraction for traffic incident detection using wavelet transform and linear discriminant analysis. *Comput.-Aided Civil Infrastruct. Eng.*, 15(4):241–250.

Sermons, M. W. and Koppelman, F. S. (1996). Use of vehicle positioning data for arterial incident detection. *Transp. Res., Part C*, 4(2):87–96.

Sethi, V., Bhandari, N., K. F. S., and Schofer, J. L. (1995). Arterial incident detection using fixed detector and probe vehicle data. *Transp. Res., Part C*, 3(2):99–112.

Skabardonis, A., Chira-Chavala, T., and Rydzewski, D. (1998). *The I-880 field experiment: effectiveness of incident detection using cellular phones* (Calif. PATH Rpt. UCB-ITS-PRR-98-1). Inst. of Transp. Studies, Univ. of Calif., Berkeley, CA.

Stephanedes, Y. J., Chassiakos, A. P., and Michalopoulos, P. G. (1992). Comparative performance evaluation of incident detection algorithms. *Transp. Res. Rec.*, 1360:50–57.

Teng, H. and Qi, Y. (2003). Application of wavelet technique to freeway incident detection. *Transp. Res., Part C*, 11(3-4):289–308.

Thancanamootoo, S. and Bell, M. G. H. (1988). *Automatic detection of traffic incidents on a signal-controlled road network (Res. Rpt. 76)*. Univ. of Newcastle upon Tyne, Transport Operations Research Group, Newcastle upon Tyne, U.K.

Thomas, K. and Dia, H. (2005). Neural network incident detection on arterials using fusion of simulated probe vehicle and loop detector data. In *Proc. 12th World Cong. Intell. Transp. Sys.*, Washington, D.C. ITS America. 12 pp.

Thomas, N. (1996). Multisensor, multivariate, and multi-class incident detection system for arterial streets. In *Transp. Traffic Theory: Proc. 13th Int. Symp. Transp. Traffic Theory*, pages 315–339, New York, NY. Pergamon.

Thomas, N. E. (1998). Multi-state and multi-sensor incident detection systems for arterial streets. *Transp. Res., Part C*, 6(5-6):337–357.

Tsai, J. and Case, E. R. (1979). Development of freeway incident-detection algorithms by using pattern recognition techniques. *Transp. Res. Rec.*, 722:113–116.

Walters, C. H., Wiles, P. B., and Cooner, S. A. (1999). Incident detection primarily by cellular phones: an evaluation of a system for dallas. In *TRB 78th Annual Meeting*, Washington, D.C. Transp. Res. Board.

Westerman, M., Litjens, R., and Linnartz, J.-P. (1996). *Integration of probe vehicle and induction loop data: Estimation of travel times and automatic incident detection* (Calif. PATH UCB-ITS-PRR-96-13). Inst. Transp. Studies, UC Berkeley, Berkeley, CA.

Willsky, A. S., Chow, E. Y., Gershwin, S. B., Greene, C. S., Houpt, P., and Kurkjian, A. L. (1980). Dynamic model-based techniques for the detection of incidents on freeways. *IEEE Trans. Automatic Control*, 25(3):347–360.

Xiao, J. and Liu, Y. (2012). Traffic incident detection using multiple-kernel support vector machine. *Transp. Res. Rec.*, 2324:44–52.

Yang, H., Wang, Z., Xie, K., and Dai, D. (2017). Use of ubiquitous probe vehicle data for identifying secondary crashes. *Transp. Res., Part C*, 82:138–160.

Yin, X., Liu, W., and Guan, L. (2006). Research on automatic incident detection algorithm based on fusion of freeway mainline information and toll collection information. In *2006 IEEE Int. Conf. Systems, Man, and Cybernetics*, Piscataway, NJ. IEEE. Pp. 2915-2919.

Yuan, F. and Cheu, R. L. (2003). Incident detection using support vector machines. *Transp. Res., Part C*, 11(3-4):309–328.

Zeng, D., Xu, J., and Xu., G. (2008). Data fusion for traffic incident detection using DS evidence theory with probabilistic SVMs. *J. Comput.*, 3(10):36–43.

Zhang, Z., H. Q. G. J. and Ni, M. (2018). A deep learning approach for detecting traffic accidents from social media data. *Transp. Res., Part C*, 86:580–596.

Zhu, T., Wang, J., and Lv., W. (2009). Outlier mining based automatic incident detection on urban arterial road. In *Proc. 6th Int. Conf. Mobile Tech.*, New York, NY. ACM. CD-ROM.