Bayesian-based Traffic State Estimation in Large-scale Networks Using Big Data

Yiming Gu

February 2017

Department of Civil and Environmental Engineering

Carnegie Mellon University

Pittsburgh, PA 15213

Thesis Committee:

Zhen (Sean) Qian, Chair

Irving Oppenheim

Mario Berges

Mark Magalotti

Submitted in partial fulfillment of the requirements

for the degree of Doctor of Philosophy.

Keywords: Traffic State Estimation, Bayesian Network, Kalman Filter

To my family.

Abstract

Traffic state estimation (TSE) aims to estimate the time-varying traffic characteristics (such as flow rate, flow speed, flow density, and occurrence of incidents) of all roads in traffic networks, provided with limited observations in sparse time and locations. TSE is critical to transportation planning, operation and infrastructure design. In this new era of "big data", massive volumes of sensing data from a variety of source (such as cell phones, GPS, probe vehicles, and inductive loops, etc.) enable TSE in an efficient, timely and accurate manner. This research develops a Bayesian-based theoretical framework, along with statistical inference algorithms, to (1) capture the complex flow patterns in the urban traffic network consisting both highways and arterials; (2) incorporate heterogeneous data sources into the process of TSE; (3) enable both estimation and perdition of traffic states; and (4) demonstrate the scalability to large-scale urban traffic networks. To achieve those goals, a Hierarchical Bayesian probabilistic model is proposed to capture spatio-temporal traffic states. The propagation of traffic states are encapsulated through mesoscopic network flow models (namely the Link Queue Model) and equilibrated fundamental diagrams. Traffic states in the Hierarchical Bayesian model are inferred using the Expectation-Maximization Extended Kalman Filter (EM-EKF). To better estimate and predict states, infrastructure supply is also estimated as part of the TSE process. It is done by adopting a series of algorithms to translate Twitter data into traffic incident information. Finally, the proposed EM-EKF algorithm is implemented and examined on the road networks in Washington DC. The results show that the proposed methods can handle large-scale traffic state estimation, while achieving superior results comparing to traditional temporal and spatial smoothing methods.

Acknowledgments

I am extremely grateful for the opportunity to work with Professor Sean Qian, who has served as my PhD thesis advisor, mentor, and friend at Carnegie Mellon University. I am indebted to him for his great support for all aspects of my professional and personal development. I would also like to thank Professors Irving Oppenheim, Hae Young Noh, Mark Magalotti, Mario Berges and Mark Kopko from PennDOT who provided a number of valuable suggestions during my qualifying examination and thesis proposal. This thesis would not have been possible without the support of them. I would also like to thank my fellow students in the Mobility Analytics Center. You have each given me support in so many ways, such as brainstorming ideas and collaborating on articles, donating your time to help with field experiments, and providing me encouragement on the numerous late nights and long weekends in the lab. It has been a true pleasure to work with each of you. I would like to acknowledge PennDOT, Federal Highway Administration (FHWA), Traffic 21 Institute, and Carnegie Mellon University's Technologies for Safe and Efficient Transportation (TSET), who sponsored my research thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background and motivation

Thanks to the relentless and surprisingly true Moore's Law [42], the exponentially increasing computational power, along with the unprecedented size of available data, enables people to build complex models to capture the intrinsically complex real world in this era of "big data". In the field of Artificial Intelligence, a deep neural network is trained from the massive amount of historic "Go" games to beat one of the best "Go" human player in the world [58]. In the field of advertising, the machine generated user browsing log is used to learn the user behavior and therefore can optimize the effect of targeted advertising [13]. In the field of society administration, microblogs are exploited to discover potential sentiment and onions of the citizens [46]. In the field of traffic incident detection, social media is mined to extract traffic incident related information [22]. The applications above have one common trait: they all exploit the emergence of big-data to infer unknown information in large-scale in real-world problems.

One of the most important problem for current society is mobility. Mobility has been a central issue for humanity throughout the history. From ancient time when our herding ancestors managed to immigrate away from fertile lands, to the age of discovery when sailors navigated

towards new continents and new trading partners, to the time of urbanization when employees commuted between their suburban home and central business district every day, to the era of globalization when goods and labors are distributed and transported all over the world, the human race has been trying to optimize the distribution of natural and human resources through mobility. Mobility is achieved by transportation, which is, in most of the cases, a limited resource. For example, in the ground transportation system, urban roads are limited resources – the maximum capacity of a road is defined by the physical properties of the road and the vehicles traveling on it, and due to various of reasons [36], road expansion is often not an option. Therefore, the essential question facing contemporary humanity is: how to manage these limited transportation resources to achieve the optimum balance between efficiency and equality. To answer this question, the first and the most fundamental step is to acquire a clear picture of these transportation systems.

1.2 Problem Statement

The research problem in this research can be illustrated in a simple example. Figure 1.1 shows a conceptual graph of a traffic network. In order to manage the traffic flow and possible traffic congestion, the essential step is to measure, estimate, and predict the states of traffic flow in the traffic network, such states could be, the traveling speed, the number of vehicles in the network, whether or not the road is congested, and whether or not there is an traffic incident.

Thanks to the existing dedicated traffic detectors, we have already have the accurate measurements (diamonds in Figure 1.1) on some of the roads. Due to the high implementation cost of these detectors, the coverage of these accurate measurements are very sparse throughout the traffic network. Additionally, we have a variety of crowedsourced data, such as the travel speed data shared by vehicles and the traffic incidents tweeted by individuals. These crowedsourced data tends to be ubiquitous but can not provide the direct measurement of the traffic information. The problem is, given all these accurate but sparse traffic-detector data, and ubiquitous

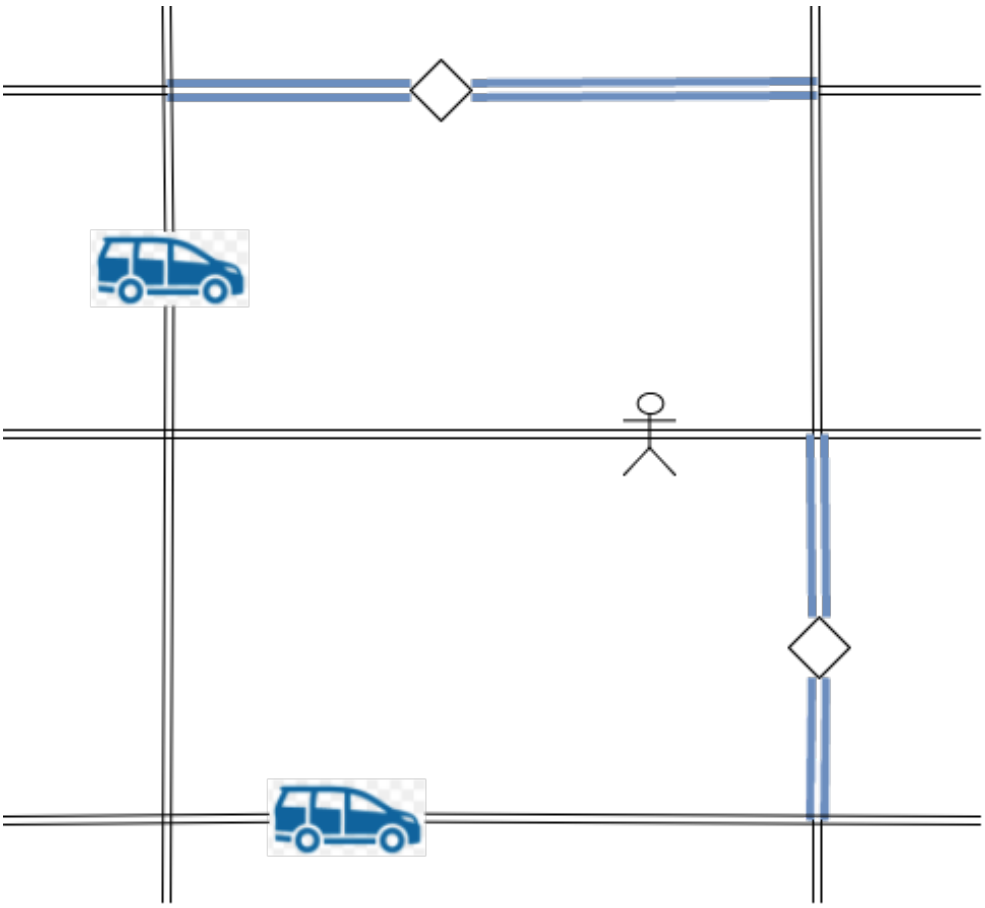


Figure 1.1: An introduction to the problem

crowedsourced data, how to estimate the traffic states on the roads without these dedicated traffic detectors in a large-scale traffic network.

1.3 Traffic State Estimation

Central to Intelligent Transportation Systems (ITS) is the capacity of knowing a clear "picture" of prevailing transportation systems. By utilizing sensing at sampled locations and time, we infer the traffic conditions of each component in the network over both spatial and temporal dimensions. This is also known as Traffic State Estimation (TSE). The results of the Traffic State Estimation have a wide spectrum of applications, such as dynamic routing[69], incident detec-

tion [2], and smart traffic signals[40], just to name a few. Classical TSE is usually deployed to a small-scale network with highway only. This is mainly because the underlying system dynamics for urban networks is complex. The complex nature lies in twofold. First, flow in the urban network, unlike on highways, is oftentimes substantially disrupted by roadway incidents. Flow can enter/exit an urban road virtually at any parking space, as opposed to fixed exit/entrance on a highway. Second, the complex nature of network flow modeling calls for large-scale simulation, which can be hardly calibrated with multi-source data in real time. This paper proposes a Bayesian framework that can seamlessly integrate multiple data sources for best inferring flow propagation and flow entrance/exit along roads. A mesoscopic link queue model is used to efficiently model flow propagations for large-scale urban networks.

Here we define two categories of traffic state: Traffic Infrastructure State (TIS) and Traffic Flow State (TFS). TFS is a time-variant, highly complex, stochastic, and heterogeneously distributed random variable. From a macroscopic point of view, TFS can be seen as the flow rate, speed, and density of vehicles at each link, which is influenced by the current condition of the road, the traffic signal system, and the TFS at its neighboring links. From a microscopic point of view, TFS can be seen as the behavior of each vehicle, which is influenced by the personality of the driver, the behavior of its neighboring vehicles, and the current TIS. Different from TFS, TIS tend to be time-invariant, and can only be changed by factors such as weather and traffic incidents. The differences between TFS and TIS are: (1) TIS can cause the change of TFS, but TFS can't influence TIS; (2) TIS can span a considerable period of time but TFS can change frequently.

Since the introduction of the traffic state estimation in 1971 [19], there have been decades of research relating to finding a model and inference algorithm to achieve an accurate estimation of traffic states. The entire field of traffic state estimation can be described in the following three dimensions, research subject, modeling method and inference algorithms.

There are usually three research subjects: (1) estimating traffic flow states; (2) estimating

traffic infrastructure states; (3) estimating traffic flow states and infrastructure states together. For the subject of estimating only the traffic flow state, early applications were reported in traffic measurement systems for short inter-detector distances in [19] and [44]. Later research has been focused on Kalman-Filter-based methods such as [34], [35], and recently in [70]. For estimating traffic infrastructure states, most of the research has been focused on one aspect of traffic infrastructure states, traffic incidents. Early researches of traffic incident detection can be traced back to California algorithms [48]. Later on various of incident detection methods have been developed such as probe-based methods [47], [72], machine-learning-based methods [60], [59], and signal-processing-based methods [28]. Only recently estimating traffic flow states and estimating infrastructure states has been united together, such as in [68] and [71].

The next two dimensions are modeling methods and inference algorithms. Although most of the research did not discriminate modeling and inference, but rather write the modeling process inside an inference framework – here we intentionally separate modeling and inference since the requirements of these two process are different: the requirement of modeling is to approximate the physical correlation and causality with a reasonable assumption, whereas the requirement of inference is to estimate the modeled variables with data.

For the modeling methods, traffic state estimation can be categorized into three parts: (1) direct observation model; (2) hidden state space model; (3) hierarchical Bayesian model. For the direct observation model, the data, as well as the target traffic states to estimate, are modeled inside of a data-driven regression or classification model, such as in [9], [8], [10] and most recently [27]. While the benefit of using these data-driven models are the ability to use existing large-scale computing framework to achieve distributed computing, such as the Streaming Spark framework used in [27], the drawback is obvious: without traffic engineering insight inside the model, it is hard to estimate traffic states where there are no nearby or historical observations. The most used model in traffic state estimation is the hidden state space model such as in [71] and [70]. Especially, paper [61] used hidden space model to successfully estimate vehicle den-

sities on a segment of I-210 highway. Hidden state space models have two layers of random variables, a layer of hidden states and a layer of observations. Traffic states to be estimated is assumed to be hidden states. The transition of hidden states is modeled by various of traffic transmission models, which can be classified as microscopic models and macroscopic models. Microscopic models have been proposed to describe movements of individual vehicles [18] [21] [25]. In contrast to microscopic models, macroscopic models describe the traffic states in bulk, such as a segment of road. The most used macroscopic models are LWR model [37][54] and gas kinetic models [52], where the basic computational unit is a small segment of a road, which is often referred as "cells" and the model is often referred as "cell transmission model". Especially in paper [43], cell transmission model has been calibrated and applied to estimate traffic states in a two-mile subsection of I-210 highway. Also, there are macroscopic models that assume larger segment of road, such as the entire road in [75] and [30]. The third category, hierarchical Bayesian model, is a general case of hidden state space model. To avoid categorical overlap, here we define hierarchical Bayesian model as the models that have more than two layers of random variables. Recently there is some research to apply hierarchical Bayesian model to estimate traffic states, such as in [26]. Also, Bayesian inference has been applied on the particle-filtering-based LWR model to illustrate the Bayesian method can quickly adapt to an accident on I-55 highway and can accurately estimate the change of traffic flow regime [51][50].

There are mainly three types of inference algorithms: (1) Gaussian approximate methods; (2) Monte Carlo methods; and (3) variational inference methods. Most research uses Gaussian approximate methods, exemplified by Kalman Filter and Extended Kalman Filter [70]. The major drawback of Gaussian mixture models is their inability to estimate a posterior distribution being close to non-Gaussian. Recently, Monte Carlo methods gained a lot of popularity due to the fact that it is an unbiased estimator, such as particle filter in [41] and Markov chain Monte Carlo [7]. However, Monte Carlo methods tend to take a lot of computational resources (in both time and space) and certain Monte Carlo methods such as Markov chain Monte Carlo may not necessarily

converge. The third category, variational inference, aims to solve the target posterior distribution by assuming a simpler "variational distribution" and iteratively optimize it. [26] used an expectation-maximization (EM) algorithm to estimate traffic states and parameters. The benefit of using variational inference is that it is more computationally efficient. However, variational inference is biased and there is a lack of theory to support the asymptotic properties of variational inference.

The **primary goal** of this thesis is to develop a traffic state estimator that

1. is able to estimate traffic flow characteristics in a large-scale urban road network considering the flow ingress/egress on both the arterial streets and freeway roads, with complex flow propagation;
2. uses a general model to effectively integrate heterogeneous data sources for the best estimation of traffic flow characteristics;
3. is computationally efficient for the large-scale network while being reasonably accurate.

1.4 Traffic Incident Detection

Traffic state patterns can be generally categorized into recurrent and non-recurrent patterns. Commuters' behaviors are naturally a day-to-day and week-to-week repetitive choice, which leads to recurrent traffic state patterns. On the other hand, non-recurrent traffic state pattern is induced by non-recurring causes, such as traffic accidents, work zones, adverse weather events, and special events. It is critical to detect those non-recurrent causes, namely incidents, in an efficient and timely manner, so that a more accurate traffic state estimation can be made to enable traffic managers to apply management strategies to mitigate the non-recurrent congestion. The **secondary goal** of this thesis is to propose an efficient, inexpensive and ubiquitous way to **detect incidents in real time** by leveraging the power of social media data.

For decades, research has been dedicated to establishing traffic incident detection systems to

identify the time, locations, and types of traffic incidents in real time. The core of a traditional incident detector consists of two components: data acquisition and data analytics. The traffic data fed to an incident detector can be classified into two categories:

(C1) Flow measurements at a fixed location: counts, occupancy, and speeds measured by inductive loop sensors, magnetic sensors, microwave sensors, infra-red sensors, ultrasonic sensors, acoustic sensors, laser sensors, and video image processors. etc.

(C2) Probe vehicle data: sampled vehicle trajectories measured by global position systems, cellular geolocation systems, and automatic vehicle identification.

C1 has been widely deployed in major cities and highways, and therefore this form of data has relatively high level of reliability. The drawback is that C1 only provides the information of traffic flow at sparse locations in the network. C2 data is limited by its low sampling rate and sometimes may exhibit high measurement errors. Comparing to algorithms using C1 data, methods using C2 data consider the spatiotemporal properties of probe vehicles. Results showed that 2% to 3% penetration of location-enabled vehicles may be sufficient for measuring traffic conditions [24].

Dependent on the data acquisition system, different incident detection algorithms have been developed. Pattern-matching algorithms identify the traffic flow characteristics under traffic incidents using C1 data, mostly for highways only. Efforts on pattern matching algorithms can be traced back to "California Method" [63]. Its assumption is that the occurrence of an incident reduces both upstream and downstream flow rate and downstream occupancy, and oftentimes increases upstream occupancy. Studies conclude that this type of algorithm is only accurate when the incident occurs near those fixed detectors [49, 64]. The main drawback of this method is that it is difficult to set up a "threshold" (e.g., the difference of occupancy rate between upstream and downstream detectors) which classifies incident and non-incident flow patterns.

Some researchers turned to statistical discriminative algorithms to detect incidents [e.g., 16, 66]. This type of algorithm computes the Standard Normal Deviate (SND) of traffic flow

to trace the sudden change induced by incidents. Modern statistical discriminative models include linear classification [56] and Support Vector Machine [76]. [56] used simulated travel time and occupancy data under different representative traffic incidents from the INTRAS simulation model. [33] used Artificial Neural Network (ANN) to classify traffic patterns into "with incidents" and without. They collected the inductive-loop detector data and incident data from the network of Disney Land, Anaheim Convention Center.

Time series algorithms such as Kalman Filter [74], Sequential Probability Ratio Test (SPRT) [1] , autoregressive integrated moving-average (ARIMA) model [3, 4, 5], high occupancy (HIOCC) algorithm [14], and wavelet models [62] have been used to detect incidents by tracking flow and occupancy data in real time. For SPRT, [1] proposed that the traffic flow can be treated as time-series signals and traffic accidents can be detected by comparing the likelihood ratio of the signal data before and after the accident. SPRT targets a specific false-positive rate. [62] used wavelet method directly to develop incident-sensitive features in the wavelet space, and indicated a better result than Artificial Neural Network and the California Method.

While incident detection algorithms on highways are mature, arterial incidents are notoriously difficult to detect in real time due to various disruptions to arterial flow. Bayesian Network (BN) has been implemented in arterial traffic incident detection by incorporating both prior knowledge and measuring data [77]. Simulated occupancy and flow data have been used to feed a Bayesian Network where the causation relationship was pre-determined by experts.

Recently individuals trajectories data (collected through GPS) have been used to directly infer incidents. A multi-level approach was developed by [32], where a hierarchical analysis of the vehicles' GPS data was conducted to identify precise locations of incidents. Moreover, a mobile application, called "WreckWatch" [73], was developed for GPS-enabled smartphones. In terms of cellular-based incident detection, an exploratory study has been done by [15].

It would be ideal to have human beings to report all incidents manually since human beings can provide detailed and accurate information regarding incidents. However, due to high capi-

tal/labor cost and significant delay in human-based reports, algorithms have been developed to automatically detect incidents. Implicitly embedded in detection automation is the assumption that significant change in flow characteristics immediately following the incidents. Through mining the real-time traffic data collected by scattered sensors in transportation networks, incidents, and their features may be identified. Algorithmic incident detection is, however, still not cheap. Incidents may occur in any location and any time period, and thus to achieve reasonable coverage and accuracy, sensing traffic flow in a wide spectrum of time and space is necessary. More importantly, algorithmic incident detection tends to work well on highways, but not on local arterials. The traffic flow on arterials is largely affected by random factors, such as non-motorized traffic, signal lights, street parking, etc. Given the current sensing coverage, it is notoriously difficult to accurately detect arterial incidents. Our general motivation is to discover an efficient yet cost-effective way to detect incidents on both highways and arterials in real time. Crowdsourcing seems a solution to this matter for its low cost, real-time capacity, and reasonable accuracy.

WAZE (www.waze.com) is a good example of incidents crowdsourcing. It has successfully provided real-time incident information in the web-GIS based applications. However, its limitation is two-fold. First, reporting incidents to WAZE requires logging in particular applications and creating reports that fall in one of the pre-determined incident categories. It is unknown that how the self-selection and penetration rate would affect the coverage and accuracy of reported incidents. It is possible that the majority of the active internet users do not limit themselves to a specific crowdsourcing application (such as WAZE). Would public data in the social media lead to a better coverage on incidents? Second, WAZE' data are proprietary and owned by private sector. It is expensive in the sense that it is costly for individuals and traffic managers to obtain. WAZE has sharing agreements with several states (such as Florida) to provide data for free, but the cost of data administrations, from the private sector to the public, are still high. Is there an inexpensive way to crowdsource incidents for the public agencies to use.

Social media sites sharing short messages, such as Twitter, have become a powerful and in-

expensive tool for extracting information of all kinds. It has a fairly large user pool, much more diverse than a specific incident crowdsourcing tool (such as WAZE). Also a significant portion of its data is shared by individuals to the public, which can be acquired using APIs. Twitter currently produces 340 million tweets per day from more than 140 million active users. Since transportation is part of everyone's daily lives, many active users post messages when they encounter incidents, or shortly after. This huge resource may potentially gather a valuable body of information regarding incidents that differ significantly by type, location, and time. Social media sites may be an inexpensive alternative to privately-owned crowdsourcing tool (such as WAZE). Numerous research has been devoted to mining social media data to detect events and breaking news. Current event detection methods based on social media streams can be classified into three categories: clustering-based, model-based, and signal-processing-based. A well-known signal-processing-based approach has shown that Twitter can be used to detect special events much faster than the traditional media [55].

Nevertheless, social media data does not come without a price. The real-time detection of incidents based on Twitter is challenging. The state-of-the-art text mining techniques cannot be applied directly to mine tweets since the tweet language varies considerably from the daily language. Twitter messages are short (140 characters at most) and can often contain typos, grammatical errors, and cryptic abbreviations. In 2009, a short-term study stated that 40% of the tweets can be considered as "pointless babble" [6], making it difficult to separate useful information from plain noise.

1.5 Contributions and Organization of the Thesis

This thesis advances the field of Traffic State Estimation in the following ways.

1. **A general methodology to effectively integrate heterogeneous data sources for the best estimation of traffic flow characteristics.**

(a) A generic Hierarchical Bayesian model to seamlessly integrate any data sources.

(b) The methodology works not only for highway stretches, but also for urban networks consisting of both highways and arterials.

(c) An efficient inference algorithm, also known as Expectation Maximization Extended Kalman Filter (EM-EKF), couples with mesoscopic network flow models in a computationally efficient way.

2. **A new efficient way of traffic incident detection using social media.**

   (a) An adaptive data acquisition system to efficiently acquire Twitter data for better traffic incident detection.

   (b) A well-defined feature extractor to transform the tweets into a high-dimensional space specifically defined for transportation domain.

   (c) A sophisticated classifier to determine whether or not a tweet is traffic-incident-related and the type if traffic incident it is referring to.

   (d) A geocoder to determine the specific longitude and latitude of the Twitter-based traffic incident.

   (e) A web interface to present Twitter-based traffic incidents in real-time.

3. **A Large-scale implementation of traffic state estimation.**

   (a) A series of methodologies to retrieve, merge, and structure the heterogeneous data for traffic state estimation.

   (b) An implementation of the Expectation Maximization Extended Kalman Filter and Twitter-based traffic detector for a large-scale traffic network using distributed parallel computing.

The thesis is organized as follows. We begin with a rigorous mathematical definition of traffic state estimation in **Chapter 2**. Following the definition, we derive of the general statistical model

of macroscopic transportation network using Hierarchical Bayesian framework and establish an inference algorithm to make estimation and prediction using partially observed data. In the final part of **Chapter 2**, we present two experiments, one from a small simulated model and one from a small real-world model, to illustrate the properties of the proposed model and algorithm.

Traffic incident detection provides essential information for an accurate traffic state estimation. In **Chapter 3**, we explored the possibility of using social media as an alternative way to detect traffic incidents, and hence to enhance traffic state estimation. Following a problem definition, a full spectrum of methodologies is proposed, including adaptive data acquisition, feature extraction, classification and geocoding. We then implement the proposed methodology in the cities of Pittsburgh and Philadelphia to test the validity of the proposed methodology in the real-world.

In **Chapter 4**, we address the difficulties of implementing a large-scale traffic state estimation in the real-world. In particular, we proposed and implemented the algorithms of data extract-transform-load (ETL). After having a clean, functional, and structured database including the shape of the road network, travel speed/volume, and traffic incidents, we developed a multi-level parallel computing framework using the state-of-art distributed computing platform. Finally, the experimental results in Washington DC area is presented and discussed.

Chapter 2

Traffic State Estimation in a Bayesian Framework

2.1 Problem Statement

In a typical traffic network, there are random variables to model the macroscopic traffic flow properties such as traffic flow rate, traffic density, traffic speed, and etc. All these random variables have two dimensions: time and space. We discretize time and space into integer numbers. For the set of time sequence, we have

$$T = \{1, 2, 3, ..., t, ..., M\} \tag{2.1}$$

and for the set of space sequence, we have

$$D = \{1, 2, 3, ..., j, ..., N\}. \tag{2.2}$$

Each random variable of traffic state has two indices. For example, the traffic flow rate at time t and location j is noted as $q_j^{(t)}$. Similarly, we write other two Traffic Flow States (TFS): traffic density as $d_j^{(t)}$ and traffic speed as $v_j^{(t)}$. Also, for Traffic Infrastructure States, we have weather $w_j^{(t)}$ and incident $c_j^{(t)}$. The generalized state can be seen as the union of all the previously

mentioned Traffic Flow States and Traffic Infrastructure States such that

$$\mathcal{S}_j^{(t)} = \{q_j^{(t)}, d_j^{(t)}, v_j^{(t)}, w_j^{(t)}, c_j^{(t)}\}. \tag{2.3}$$

For each space j and and time t, there is a vector of state variables $\{q_j^{(t)}, d_j^{(t)}, v_j^{(t)}, w_j^{(t)}, c_j^{(t)}\}$. In this setting, the total number of random variables is $M \times N \times |\mathcal{S}_j^{(t)}|$, where $|\mathcal{S}_j^{(t)}|$ denotes the size of the vector $\mathcal{S}_j^{(t)}$. We denote the set of all these random variables as S.

In this traffic network, we have observed some of the random variables. These random variables can be any element of the $\mathcal{S}_j^{(t)}$ for any t and any j. We denote the set of the observed variables as X, with the realized value $X = x$ and we denote the set of unobserved variables as Y.

Also, we assume there is a underlying distribution that governs the transition from TFS to TFS and the transition from TIS to TFS. First, we denote the set of neighbouring indices for link j is $u(j)$. For simplification, we denote $u(j) = j - 1$. Then the conditional distribution of $q_j^{(t)}$ is estimated, with parameter vector α

$$\mathcal{S}_j^{(t)} \, |\mathcal{S}_{u(j)}^{(t-1)}, \mathcal{S}_j^{(t-1)}, w_j^{(t)}, c_j^{(t)}, \alpha \tag{2.4}$$

The problem of this research is simple: given the observed traffic states $X = x$, estimate: (1) the distribution of unobserved traffic states Y; (2) the distribution of the unknown model parameters α. The objective of this research is to find the posterior conditional distribution of

$$Y, \alpha | X = x. \tag{2.5}$$

Under this notational framework, the questions raised at the end of Chapter 1 can be defined in three different variations of Equation 2.5.

Definition 2.1.1 ***Temporal Estimation***

When the observed traffic flow states (X) are at all locations from time 0 to time t, and the unknown traffic flow states (Y) are at all locations from time $t+1$ to $t+k$ (k is a positive

integer), and when all traffic infrastructure states are known and invariant over time, we call this estimation ***Temporal Estimation****.*

Definition 2.1.2 ***Spatial-Temporal Estimation***

When the observed traffic flow states (X) are at locations 0 to i from time 0 to time $t+k$, and at locations i to $i+z$ from time 0 to time t, and the unknown traffic flow states (Y) are at locations i to $i+z$ from time $t+1$ to $t+k$ (k and z are positive integers), and when all traffic infrastructure states are known and invariant over time, we call this estimation ***Spatial-Temporal Estimation****.* ***Temporal Estimation*** *is a special case of* ***Spatial-Temporal Estimation****.*

Definition 2.1.3 ***Incident Adaptation***

Within the conditions of ***Spatial-Temporal Estimation****, one traffic infrastructure state, traffic incident, is often unknown and changing over time, and consequently causing the closure and re-opening of lanes in the traffic network. Under this condition, the problem of* ***Incident Adaptation*** *requires not only performing Spatial-Temporal Estimation, but also inferring the occurrence of the traffic incident and the change of lanes in the traffic network.*

2.2 General Model

The general model is proposed in terms of graphical model, i.e., a graph that captures the conditional independence of the random variables. The model is "general" in the sense that no specific distribution is assumed other than the conditional independence among random variables. The graphical model for the general model is shown in Figure 2.1. To understand the graphical model, the notations in the Figure 2.1 is worth noting.

First, the blue circle encodes the random variables we try to model, i.e.,

$$\{q_j^{(t)}, d_j^{(t)}, v_j^{(t)}, w_j^{(t)}, c_j^{(t)}\}, \ \ \forall i \in T, \forall j \in D \tag{2.6}$$

The lines encode the undirectional correlation between two random variables. For example, a

line between $d_j^{(t)}$ and $v_j^{(t)}$ denotes that these to variables are correlated, i.e.,

$$P(d_j^{(t)}, v_j^{(t)}) \neq P(d_j^{(t)})P(v_j^{(t)}) \tag{2.7}$$

Specifically, the degree of correlation is captured by a potential function $K()$, and the joint probability is captured such as,

$$P(d_j^{(t)}, v_j^{(t)}) = \frac{K(d_j^{(t)}, v_j^{(t)})}{\int_{d_j^{(t)}, v_j^{(t)}} K(d_j^{(t)}, v_j^{(t)})} \tag{2.8}$$

The fully correlated random variables can be agglomerated into a multivariate random vector. Here "fully correlated" means that for a set of random variables S, there is an edge between every possible combinations of two elements in S. As can be seen in Figure 2.1, $q_j^{(t)}, d_j^{(t)}, v_j^{(t)}$ can be agglomerated into a random vector of traffic flow states,

$$F_j^{(t)} = \{q_j^{(t)}, d_j^{(t)}, v_j^{(t)}\} \tag{2.9}$$

Following the definition of the potential function defined above,

$$P(F_j^{(t)}) = \frac{K(q_j^{(t)}, d_j^{(t)}, v_j^{(t)})}{\int_{q_j^{(t)}, d_j^{(t)}, v_j^{(t)}} K(q_j^{(t)}, d_j^{(t)}, v_j^{(t)})} \tag{2.10}$$

In contrast to the lines, the arrows, however, encode the directional causality between variables. For example,

$$P(c_j^{(t)}, w_j^{(t)}, F_{j-1}^{(t-1)} F_j^{(t)}) = P(c_j^{(t)})P(w_j^{(t)})P(F_{j-1}^{(t)})P(F_j^{(t)}|c_j^{(t)}, w_j^{(t)}, F_{j-1}^{(t-1)}) \tag{2.11}$$

where the causality is encoded in the term $P(F_j^{(t)}|c_j^{(t)}, w_j^{(t)}, F_{j-1}^{(t-1)})$ where $F_j^{(t)}$ is conditionally dependent on $c_j^{(t)}, w_j^{(t)}, F_{j-1}^{(t-1)}$. Also in the graph, there are arrows directing from $c_j^{(t)}, w_j^{(t)}, F_{j-1}^{(t-1)}$ to $F_j^{(t)}$. Notice that in Figure 2.1, for notational convenience but without losing generality, we assume the neighbouring function $u(j)$ in Equation 2.4 is simplified as

$$u(j) = j - 1 \tag{2.12}$$

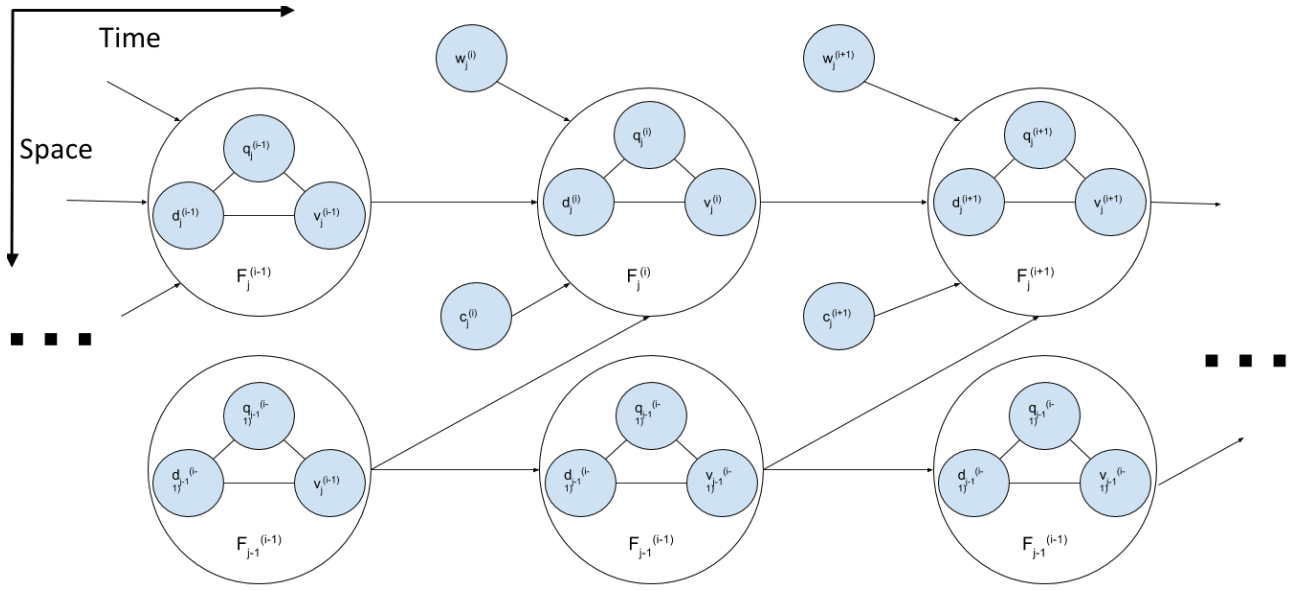


Figure 2.1: Graphical representation of the general model

By comparing Equation 2.10 and Equation 2.11, it can easily be seen that causality is a special case of the correlation, where the potential function can be directly expressed by the multiplication of conditional probabilities.

In summary, the graphical representation in Figure 2.1 encodes the basic understanding of the random variables in the traffic network. This understanding has two parts: (1) the flow rate, density, and travel speed in a certain location j and time t are intrinsically correlated; (2) the flow rate, density, and travel speed in a certain location j and time t is caused by the flow rate, density, and travel speed in its neighbouring location $u(j)$ and time $t-1$, and also by the current Traffic Infrastructure States at time t and location j, like weather and incident. In the following sections of this chapter, some further assumptions are made to specify the exact forms of this correlation and causality.

Specifically, we model the correlation among traffic states from three aspects:

1. The correlation among the three traffic states at the same time t: $d^{(t)}$, $q^{(t)}$, and $v^{(t)}$
2. The correlation among the traffic flow states at different time t and $t-1$: $P(F^{(t)}|F^{(t-1)})$

3. The influence of traffic infrastructure state on the correlation among the three traffic states at the same time t: $d^{(t)}$, $q^{(t)}$, and $v^{(t)}$

2.3 Correlation among Traffic States

2.3.1 Fundamental Diagrams

In this section we will model the correlation among three random variables at the same time t and location j: flow density $d_j^{(t)}$, flow rate $q_j^{(t)}$, and flow speed $v_j^{(t)}$, and for simplicity, we will write them as d, q, and v. Here we formally define these three variables.

Definition 2.3.1 ***Flow density (or density)*** *d is the number of vehicles (n) per unit length of road.*

$$d = \frac{n}{L} \tag{2.13}$$

Definition 2.3.2 ***Flow rate(or flow)*** *q is the number of vehicles (m) that passes a certain cross-section per time unit.*

$$d = \frac{m}{\Delta T} \tag{2.14}$$

Definition 2.3.3 ***Flow speed (or speed)*** *v is the average speed along a length (L).*

$$v = \frac{N}{\sum_{i=1}^{N} \frac{1}{v_i}} \tag{2.15}$$

where N is the total number of vehicles in the road of length L.

It is well established in macroscopic traffic flow theory that there are rules that govern the correlations among flow, speed, and density, which is commonly known as the "fundamental diagrams". The basic assumption of "fundamental diagrams" is that there is functional correlation that connects every two combinations of these three variables. Such as

$$q = q_v(v) \tag{2.16}$$

$$q = q_d(d) \tag{2.17}$$

The functional form of $q_v()$ and $q_d()$ are determined by the traffic infrastructure state on that traffic link, and can often be approximated from the measuring data. However, in most of the applications where the direct measuring data is not available, the "triangular" fundamental diagrams are often used as follows.

Definition 2.3.4 ***Fundamental diagrams*** $h()$

$$q_v(v) = dv \tag{2.18}$$

$$q_d(d) = min(F_f d, w(J - d)), d \in [0, J] \tag{2.19}$$

where J is the jam density, F_f is the free flow speed, and w is the backward wave speed. When the flow rate get its maximum, the maximum capacity C is attained at critical density d_c; i.e., $C = q_d(d_c) \geq q_d(d)$.

Notice that the above fundamental diagrams are centered on the density, meaning only knowing the density is enough to calculate the other two variables of traffic flow state.

2.3.2 Link Queue Model

In this section, we will model the correlation of the traffic flow states at all spaces between time t and previous time $t-1$: $P(F^{(t)}|F^{(t-1)})$. According to the fundamental diagrams defined above, knowing the density is enough to calculate the other two variables, and therefore, $F^{(t)}$ and $F^{(t-1)}$ can be simplified to $d^{(t)}$ and $d^{(t-1)}$. In Link Queue Model, the basic computational unit is traffic link.

Definition 2.3.5 ***Traffic link*** *has stationary and homogeneous traffic flow states and traffic infrastructure states within a short time period and within the length of the link.*

Not coincidentally, the definition of traffic link corresponds to the discretized spacial segmentations of the traffic states, i.e., each traffic states, no matter it is traffic flow state or traffic

infrastructure state at space j, corresponds to the traffic states at the traffic link j. According to the fundamental diagrams defined above, for each traffic link at time t and space j, the traffic flow state variable is only the link density, $d_j^{(t)}$. In Link Model [30], the traffic flow state transition function is a non-linear function $\mathscr{F}()$ such that

$$d^{(t)} = \mathscr{F}(d^{(t-1)}) \tag{2.20}$$

where $d^{(t)}$ means the flow density at all links at time t and $d^{(t-1)}$ means the flow density at all links at time $t-1$.

The reasons we choose link queue model as our traffic flow state transition model are as follows

1. Comparing to Cell Transmission Model, the link queue Model is computationally inexpensive, meaning the link queue model can scale better to a larger arterial traffic network;
2. The link queue model rigorously describe interactions among different links by using link demands, supplies, and junction models, which is consistent with macroscopic merging and diverging behavior.

Now we write the specific form of traffic flow state transition function $\mathscr{F}()$ in the context of Link Queue Model. In the Link Queue Model, traffic on each link is considered as a queue, and the **state** of a queue is its density (the number of vehicles per unit length). Based on the fundamental diagrams of the link, the Link Queue Model define the demand (maximum sending flow) and supply (maximum receiving flow) of a queue. Then the out-fluxes of upstream queues and in-fluxes of downstream queues at a network junction are determined by macroscopic merging and diverging rules, which were first introduced in network kinematic wave models.

In Link Queue Model, we first introduce another two random variables in the traffic flow state in time t: demand $D^{(t)}$ and supply $S^{(t)}$:

Definition 2.3.6 ***Demand*** $D^{(t)}$

$$D^{(t)} = q_d(min\{d^{(t)}, d_c\}) = \tag{2.21}$$

$$\begin{cases} q_d(d^{(t)}), & d^{(t)} \in [0, d_c] \\ C, & d^{(t)} \in [d_c, J] \end{cases} \tag{2.22}$$

where q_d are the vector of fundamental diagrams at all links,d_c is the vector of critical density at all links, and C is the vector of maximum flow at all links, and J is the vector of maximum density at all links.

Definition 2.3.7 ***Supply*** $S^{(t)}$

$$S^{(t)} = q_d(max\{d^{(t)}, d_c\}) = \tag{2.23}$$

$$\begin{cases} C, & d^{(t)} \in [0, d_c] \\ q_d(d^{(t)}), & d^{(t)} \in [d_c, J] \end{cases} \tag{2.24}$$

where q_d are the vector of fundamental diagrams at all links, d_c is the vector of critical density at all links, and C is the vector of maximum flow at all links, and J is the vector of maximum density at all links.

Also for each link, we define the influx (the traffic flow that enters the link k) as f_k and outflux (the traffic flow that leaves the link k) as g_k. Furthermore, we define traffic junctions K as the intersection among traffic links, such as highway merging and diverging sections. At junction K, outfluxes, g_K, and influxes, f_K, can be computed from upstream demands, D_K, and downstream supplies S_K. Notice that for a junction with multiple inflow links and outflow links, g_K, f_K, D_K, and S_K can be a vector of multiple random variables. To determine the transmission within traffic junctions, we define flux function.

Definition 2.3.8 ***Flux function*** $\mathcal{F}_K()$ *is a function that governs*

$$(g_K, f_K) = \mathcal{F}_K(D_K, S_K) \tag{2.25}$$

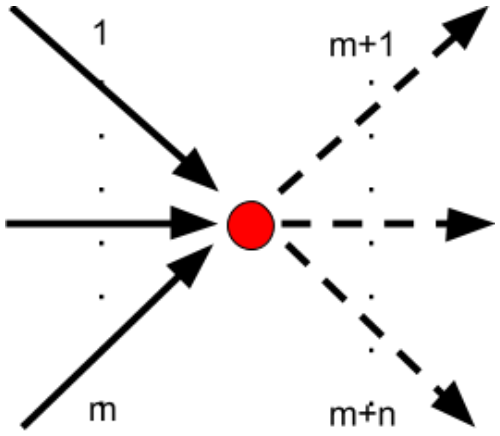


Figure 2.2: Flux function

A general flux function (Figure 2.2) with 1 to m as inflows and $m+1$ to $m+n$ as outflows at junction K is given as follows.

1. From the definition of the demand and supply, all the upstream demand and downstream supply can be calculated by the flow density of upstream links ($d_{1:m}$) and downstream links ($d_{m+1:m+n}$).
2. The turning proportion $Z_{a\to b}$ from an upstream link $a \in \{1:m\}$ to a downstream link $b \in \{m+1:m+n\}$ is either a function of the supply and demand following the FIFO diverging and fair merging rules [29], or as an independent variable of route choice.
3. The out-flux of upstream link a is

$$g_a = min\{D_a, \theta_a C_a\} \tag{2.26}$$

where C_a is the maximum flow rate, and θ_a is the critical demand level, which uniquely solves the following min-max problem

$$\theta_j = \min\{\max_{a\in\{1:m\}} \frac{D_a}{C_a}, \min_{b\in\{m+1:m+n\}} \max_{A_1} \frac{S_b - \sum_{\alpha\in\{1:m\}\setminus A_1} d_\alpha Z_{\alpha\to b}}{\sum_{a\in A_1} C_a Z_{a\to b}}\} \tag{2.27}$$

where A_1 is a non-empty subset of $\{1:m\}$

4. The in-flux of the downstream link b is

$$f_b = \sum_{a \in \{1:m\}} g_a Z_{a \to b} \tag{2.28}$$

To give the relationship between the flux and the traffic density, we utilize the conservation law of traffic flow.

Definition 2.3.9 *Conservation law of traffic flow at time t and time $t-1$*

$$d^{(t)} = d^{(t-1)} + \frac{1}{L}(f^{(t)} - g^{(t)})\Delta t \tag{2.29}$$

where Δt is the time interval between two time steps.

In summary, in this section we used the Link Queue Model to: (1) calculate the supply and demand at each link from the density at each link; (2) calculate the influx and outflux of links from the supply and demand calculated before; (3) calculate the density at each link again after knowing the influx and outflux. In general, Link Queue Model, along with the fundamental diagrams, specifies a non-linear system from all traffic densities at time t, $d^{(t)}$ to all traffic densities at time $t+1$, $d^{(i+1)}$.

Definition 2.3.10 *Link Queue Model specifies a non-linear function $\mathscr{F}()$ such that*

$$d^{(i+1)} = \mathscr{F}(d^{(t)}) \tag{2.30}$$

2.3.3 Traffic Infrastructure States in Fundamental Diagrams

In the previous sections, we modeled how the traffic flow states transitioned over time and space. In this section, however, we model traffic infrastructure states as a way to influence the transition of the traffic flow states. This can be interpreted in two ways: (1) if one is interested in traffic flow states, the traffic infrastructure states can be seen as the parameter, or the change of parameter, in the transition function of traffic flow states; (2) if one is interested in the traffic infrastructure

states, traffic flow states can act like a detector data for traffic infrastructure states, where the vehicles are actually detectors for the traffic infrastructure.

As stated in the general model, in this research, we concentrate on one traffic infrastructure state variable: traffic incident c. The first assumption is that traffic infrastructure states are independent. That is to way, given the observation of one traffic incident, the distribution of another traffic incident remains unchanged. Before inspecting the influence of incidents, we first define the space of these two random variables.

Proposition 2.3.1 *Incident c is a discrete categorical random variable.*

Now we inspect how incidents can change the traffic infrastructure and therefore influence the traffic flows:

Proposition 2.3.2 *Traffic incident such as accident, road work, or police activity can change the number of lanes available and thus pose a bottleneck;*

From this intuition, we model the influence of traffic infrastructure as follows

Proposition 2.3.3 *Traffic incidents changes number of lanes in a link and therefore changes overall flow of a link*

$$q_d(d) = min(F_f d, w(\gamma J - d)), d \in [0, \gamma J] \tag{2.31}$$

where J is the jam density, F_f is the free flow speed, and w is the backward wave speed. γ is the decreasing factor of different incidents.

$$\gamma \sim \mathcal{P}(c) \tag{2.32}$$

where $\mathcal{P}$ is a distribution parameterized by c.

At this point, it is safe to define all the model parameters of the Fundamental Diagrams and the Link Queue Model as α, which includes the decreasing factor of incidents γ at all links, the turning proportions Z at all nodes, and the parameters in the Fundamental Diagrams F_f, w, and J at all links.

Definition 2.3.11 *The **model parameters** α*

$$\alpha = \{\gamma, F_f, w, J, Z\} \tag{2.33}$$

2.4 Inference: Expectation-Maximization Extended Kalman Filter

In the section above, we defined a detailed model about how random variables in the traffic network are correlated with each other. In other words, given the boundary conditions such as the flow rate at all of the links in the boundary regions of the network, and the initial conditions such as the flow density at all the links in the traffic network at time step 0, we are able to simulate the traffic flow in the entire network.

Definition 2.4.1 *The task of **Inference** in traffic state estimation is to estimate the unknown traffic states Y and unknown model parameters α given the observations X, and is to estimate the following distribution:*

$$Y, \alpha | X \tag{2.34}$$

2.4.1 Expectation-Maximization (EM) Algorithm on Model Parameter Updating and State Estimation

Expectation-maximization (EM) method is an iterative method to estimate both model parameters (α) and latent variables (Y). EM algorithm can be proved in Appendix A as an optimization algorithm to iteratively solve

$$\arg\max_{Y,\alpha} P(Y, \alpha | X).$$

EM algorithm consists of two steps, E-step and M-step.

E-step:

$$Y|X, \alpha$$

M-step:

$$\alpha|Y, X$$

First, M-step aims to estimate the optimal model parameters, α, assuming all traffic states are known. This can be achieved as a simple error minimization. From the fundamental diagrams and Link Queue model, we can obtain the observation function $h()$ and have the pseudo-observation

$$X^- = h(Y) \tag{2.35}$$

Therefore, minimizing the difference between X^- and X can lead to a new iteration of model parameter α as defined in E-M. The M-step is

$$\alpha^+ = argmin_\alpha ||X - X^-||_2^2 \tag{2.36}$$

where α^+ is the new iteration of α, and $||\dots||_2^2$ means the square of Euclidean norm. Note that α here incorporates the incident adaptation. Any incident detection, quantified as the capacity reduction, can be added to the equation as the additional constraints of α.

2.4.2 E-step: Extended Kalman Filter

E-step aims to estimate unobserved traffic states assuming all model parameters are known:

$$Y|X, \alpha.$$

In this section, an exact solution of $Y|X, \alpha$ will be given and the limitation of the exact solution will be pointed out. Then we will propose the Extended Kalman Filter algorithm as an alternative approximate solution.

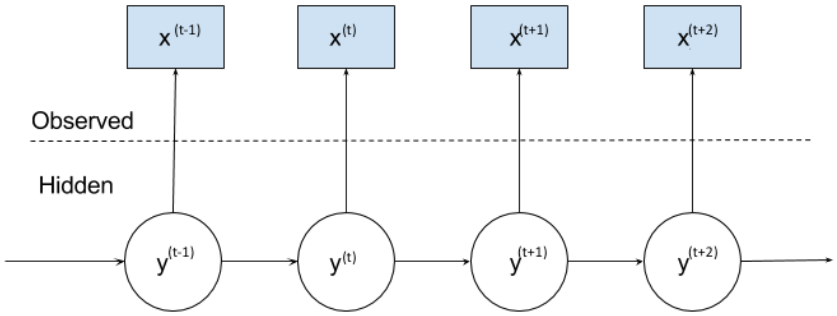


Figure 2.3: Simplified Bayesian Hierarchical Model

Exact Inference

The exact inference is achieved by Recursive Bayesian Estimation shown in paper [11]. The authors expand the posterior density and factored in a recursive way. Assuming the observations are y and the hidden states are x. As can be seen in Figure 2.3, the model is a simplified model for the general model in Figure 2.1. The state transition follows a distribution of

$$p(y^{(t)}|y^{(t-1)}) \tag{2.37}$$

and the observation follows a distribution of

$$p(x^{(t)}|y^{(t)}) \tag{2.38}$$

Definition 2.4.2 *We call the model defined above as **Simplified Bayesian Hierarchical Model**. In the Simplified Bayesian Hierarchical Model, we assume the random variables are continuous.*

Definition 2.4.3 ***Exact inference for Simplified Bayesian Hierarchical Model***

The posterior of hidden states y is

$$p(y^{(t)}|x^{(1:t)}) = \frac{p(x^{(1:t)}|y^{(t)})p(y^{(t)})}{p(x^{(1:t)})} \tag{2.39}$$

$$= \frac{p(x^{(t)}, x^{(1:t-1)}|y^{(t)})p(y^{(t)})}{p(x^{(1:t)})} \tag{2.40}$$

$$= \frac{p(x^{(t)}|x^{(1:t-1)}, y^{(t)})p(x^{(1:t-1)}|y^{(t)})p(y^{(t)})}{p(x^{(1:t)})} \tag{2.41}$$

$$= \frac{p(x^{(t)}|x^{(1:t-1)}, y^{(t)})p(y^{(t)}|x^{(1:t-1)})p(x^{(1:t-1)})p(y^{(t)})}{p(x^{(t)}|x^{(1:t-1)})p(x^{(1:t-1)})p(y^{(t)})} \tag{2.42}$$

$$= \frac{p(x^{(t)}|y^{(t)})p(y^{(t)}|x^{(1:t-1)})}{p(x^{(t)}|x^{(1:t-1)})} \tag{2.43}$$

It is obvious that the term in Equation (3.8) could solved recursively using $p(y^{(t-1)}|x^{(1:t-1)})$

$$p(y^{(t)}|x^{(1:t-1)}) = \int p(y^{(t)}|y^{(t-1)}p(y^{(t-1)}|x^{(1:t-1)}))dy^{(t-1)} \tag{2.44}$$

Also, term $p(y^{(t)}|x^{(1:t)})$ can be solved by

$$p(y^{(t)}|x^{(1:t)}) = Cp(x^{(t)}|y^{(t)})p(y^{(t)}|x^{(1:t-1)}) \tag{2.45}$$

where

$$C = (\int p(x^{(t)}|y^{(t)})p(y^{(t)}|x^{(1:t-1)}dy^{(t)})^{-1} \tag{2.46}$$

The limitation to the exact inference is, the solution of the equations above have high-dimensional integral, which is only tractable for extremely small-scale applications. For large-scale applications in this research, we utilize Gaussian approximate inference.

Gaussian Approximation: Extended Kalman Filter under Locally Linear Gaussian System

Gaussian approximate methods model the pertinent densities in the Bayesian recursion by Gaussian distributions, under the assumption that a consistent minimum variance estimator (of the posterior state density) can be realized through the recursive propagation and updating of **only**

the first and second order moments of the true densities. The most popular ones are Kalman Filter [31] and Extended Kalman Filter [38].

In the context of Link Queue Model, the Kalman Filter can be described as follows. First, we assume there is only one state variable, traffic density $d^{(t)}$ at time t for all links. Also, there are observations about the travel speed on every links $v^{(t)}$ at time t. All other model parameters are assumed to be known. Therefore, the inference task is simplified to find the posterior of traffic densities given the initial conditions, boundary conditions, and the observations, up to t-th step,

$$d^{(1:t)}|v^{(1:t)} \tag{2.47}$$

From Link Queue Model and the fundamental diagrams, we know the state transition function $\mathscr{F}()$ and the fundamental diagram (observation function) $h()$ where $v = h(d)$. Specifically, we have

$$d^{(t)} = \mathscr{F}(d^{(t-1)}) + w^{(t-1)} \tag{2.48}$$

$$v^{(t)} = h(d^{(t)}) + m^{(t)} \tag{2.49}$$

where $m^{(t)}$ and $w^{(t)}$ are zero-mean iid Gaussian noise with known variance. We also know the initial state $d^{(0)}$ with mean $\mu^{(0)}$ and variance $P^{(0)}$ and the boundary conditions.

$$E[d^{(t)}|v^{(1:t)}] = d_a^{(t)} \tag{2.50}$$

$$= d_f^{(t)} + K^{(t)}(y^{(t)} + \hat{y^{(t)}}^{-}) \tag{2.51}$$

where $d_a^{(t)}$ is the posterior mean of the state variable $d^{(t)}$; $d_f^{(t)}$ is the state estimation from the previous state; $K^{(t)}$ is the "Kalman gain" to be calculated; $y^{(t)}$ is the observation; and $+\hat{y^{(t)}}^{-}$ is the projected estimation from $d_f^{(t)}$.

The steps of Extended Kalman Filter are as follows:

1. **Initialization**:

$$d_a^{(0)} \sim N(\mu^{(0)}, P^{(0)}) \tag{2.52}$$

2. **Model Forecast**: run the Link Model on the initial condition and boundary conditions

$$d_f^{(t)} = \mathscr{F}(d_a^{(t-1)}) \tag{2.53}$$

$$P_f^{(t)} = \mathscr{J}_f(d_a^{(t-1)})P^{(t-1)}\mathscr{J}_f^T(d_a^{(t-1)}) + Q^{(t-1)} \tag{2.54}$$

where $\mathscr{J}_f$ is the Jacobian matrix of the function $\mathscr{F}$, and $Q^{(t-1)}$ is the known covariance matrix of the state transition errors.

3. **Data correction**:

$$d_a^{(t)} = d_f^{(t)} + K^{(t)}(v^{(t)} - h(d_f^{(t)})) \tag{2.55}$$

$$K^{(t)} = P_f^{(t)}\mathscr{J}_h^T(d_f^{(t)})(\mathscr{J}_h(x_f^{(t)})P_f^{(t)}\mathscr{J}_h^T(x_f^{(t)}) + R^{(t)})^{-1} \tag{2.56}$$

$$P^{(t)} = (I - K^{(t)}\mathscr{J}_h(d_f^{(t)}))P_f^{(t)} \tag{2.57}$$

where $\mathscr{J}_h$ is the Jacobian matrix of function $h()$, and $R^{(t)}$ is the observation error covariance matrix.

Following the steps above, the state estimation can be solved sequentially, such that

$$d^{(t)}|v^{(1:t)} \sim N(d_a^{(t)}, P^{(t)}) \tag{2.58}$$

It is proved that under the Gaussian-linear assumption, Kalman Filter is the exact solution for the given problem. Please see the Appendix B for the details. Equation 2.58 is an example of the EKF solution of $Y|X, \alpha$ in the E-step.

Comparing to previously proposed methodology, the Expectation-Maximization Extended Kalman Filter (EM-EKF) has the following advantages: (1) comparing to particle-filter or other Monte-Carlo-based algorithms, EM-EKF is computationally more efficient and could be applied to handle large-scale heterogeneous data in parallel; (2) since the EM algorithm is able to perform inference on the hierarchical Bayesian probabilistic model, it could have the potential to model and perform inference on a more complex arterial network; (3) the EM-EKF can be effective

under missing data; (4) EM-EKF could effectively model egress and ingress of parking flows by modifying the model parameters α.

2.5 Numerical and Real-world Experiments

2.5.1 Tiny Simulated Network

The conceptual model of the experiment is shown in Figure 2.4. At start, all the links are empty. The simulation time duration is $T = 1.05hrs$, and time interval $\Delta t = 1.75 \times 10^{-4}hrs$. The boundary conditions are constant: the origin demand is constant $D_o(t) = 7020vph$, and the destination supply is also constant $S_d(t) = 2340vph$. Also, the fundamental diagrams of the links has free flow speed $F_f = 70mph$, jam density per lane $J = 200v/mile$, and critical density is $d_c = 40v/mile$. The length of each link are $L_i = [1, 1, 1, 1]mile$ and number of lanes at each link are $n_i = [3, 1, 2, 1]$. The diverge from link 1 to link 2 and link 3 follows a $1 : 2$ split. Also, there is on-street parking on link 3, which can increase or decrease the influx towards link 3 by a random number $z \sim N(0, 500)vph$ over time. In the initial condition, all links are empty. In the Link Queue model, at one certain time step i, the entire network has four traffic flow state variables.

Definition 2.5.1 *Traffic flow state variables in the experiment network*

$$d_1^{(t)}, d_2^{(t)}, d_3^{(t)}, d_4^{(t)} \tag{2.59}$$

The experiment is implemented as follows:

1. Build a microscopic simulation model (VISSIM) based on the conceptual model in Figure 2.4 using the same physical parameters.
2. Load the microscopic simulation model with constant vehicle input of $d_o(t) = 7020vph$.
3. Collect the flow rate, average speed, and flow density from the VISSIM model using .COM interface of VISSIM.

4. Use the Link Queue model based on the specified parameter in the conceptual model shown in Figure 2.4 to approximate flow propagation. This Link Queue model acts as a state transition function $\mathscr{F}()$ in EKF. Also, the fundamental diagrams specified in Section 3 acts as a observation function $h()$ in EKF.
5. Using the observations from the simulated model, the state transition function $\mathscr{F}()$ and observation function h, and the specified initial condition and boundary conditions above, run the EM-EKF algorithm stated in Section 3.
6. Compare the state variables in the simulation model, which act as ground truth, and the state variables derived from EM-EKF.

Following the definition in Section 2, in this experiment, we investigate three categories of estimation: Temporal Estimation, Spatial-Temporal Estimation, and Incident Adaptation. Also in order to test the accuracy of Extended Kalman Filter without Expectation Maximization, we also define a Vanilla Temporal Estimation. With all the model parameters, initial conditions, boundary conditions, and observation data given, for every time step t, we aim to estimate the unknown traffic states $Y^{(1:t)}$ and unknown model parameters $\alpha^{(1:t)}$ given only observations **up to** time t as $X^{(1:t)}$:

$$Y^{(1:t)}, \alpha^{(1:t)} | X^{(1:t)}.$$

The details of these four estimation tasks in this experiment are defined as follows:

1. **Vanilla Temporal Estimation**: Given the average speed observation of link 1-4 at time 0-1.05h, the task is to estimate the flow density and flow rate of link 1-4 at time 0-1.05h;
2. **Temporal Estimation**: Given the average speed observation of link 1-4 at time 0-0.5h, the task is to estimate the flow density and flow rate of link 1-4 at time 0-1.05h;
3. **Spatial-Temporal Estimation**: Given the average speed observation of link 1,2,4 at time 0-1.0.5h, and speed observation of link 3 at time 0-0.5h, the task is to estimate the flow density of link 3 at time 0.5-1.05h;

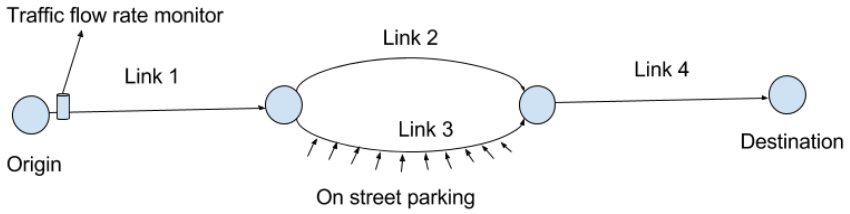


Figure 2.4: Experiment setup

4. **Incident Adaptation**: Under the condition of Spatial-Temporal Estimation, and the number of lanes at link 3 drops from 2 to 1 (unknown incident) at time point 0.5h, the task is to estimate the flow density of link 3 at time 0.5-1.05h, as well as the number of lanes at link 3.

To illustrate the result of Vanilla Temporal Estimation defined above, the ground truth and the estimation of the traffic state variables are shown in Figure 2.5 and Figure 2.6. Specifically, Figure 2.5 illustrates the result of density estimation for link 1-4 in 0-1.05 hours given the average speed for link 1-4 in 0-1.05 hours. It can be seen from Figure 2.5 that the estimated traffic density approaches the true density as time proceeds due to the fact that the EKF utilizes the input of the average speed (the observations of true traffic density) to correct its estimation. The mean absolute error of the traffic states estimation at all these four links 24.8 v/mile. In terms of flow rate (Figure 2.6), the conclusion is similar, where the estimated flow rate from EKF is approaching the true flow rate.

A comparison of Vanilla Temporal Estimation and Temporal Estimation is shown in Figure 2.5 and Figure 2.7. In terms of data, in Vanilla Temporal Estimation, the average speed for all links at 0.5-1.05h is available but in Temporal Estimation, these data is not available. It can be seen in both Figure 2.5 and Figure 2.7 that these extra data in Vanilla Temporal Estimation has a positive influence on the estimation results in both density and flow rate. The mean absolute error for Vanilla Temporal Estimation is 34.9 v/mile and that of Temporal Estimation is 37.5 v/mile. Specifically, the estimation results of Temporal Estimation and Vanilla Temporal

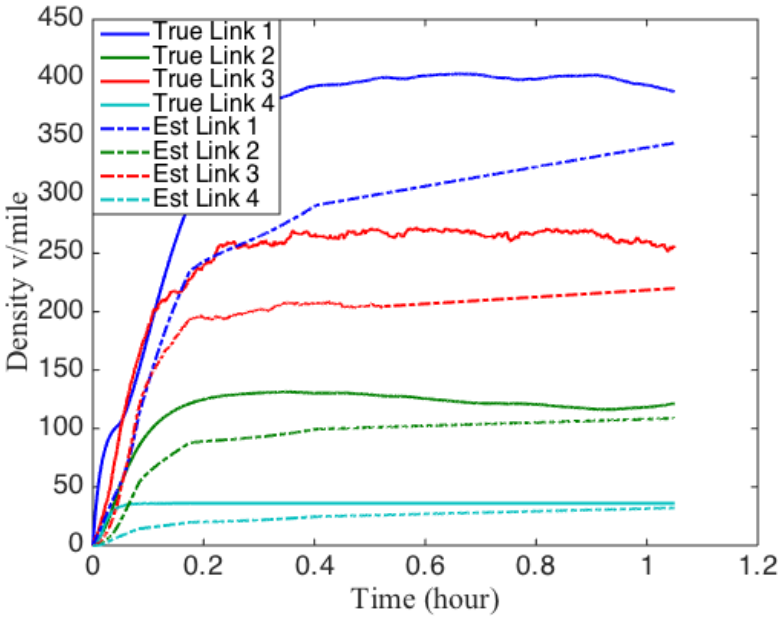


Figure 2.5: Vanilla Temporal Estimation (density)

Estimation overlap at 0-0.5h because the data given in these two estimations are the same. After 0.5h, since the observation data is on the average speed in link 1-4 available in Vanilla Temporal Estimation but not in Temporal Estimation, the final result of Vanilla Temporal Estimation is better than the result of Temporal Estimation. The reason for that is obvious, the input of extra observation data helps the Vanilla Temporal Estimation to continuously correct its estimation results but for the Temporal Estimation, only the base model with given parameter can be used to make the estimation.

The results of the Spatial-Temporal Estimation for Link 3 are shown in Figure 2.8 and Figure 2.9. In both figures the first half shows the traffic state estimation with observation and the second half shows the traffic state estimation without observation, which satisfies the definition of temporal-spatial estimation. The mean absolute error for Spatial-Temporal Estimation of density is 18.5 v/mile. It achieves a better result than both Vanilla Temporal Estimation and Temporal Estimation. Comparing the Spatial-Temporal Estimation in Figure 2.8 with Vanilla Temporal Estimation in Figure 2.5, and according to the definitions of Vanilla Temporal Esti-

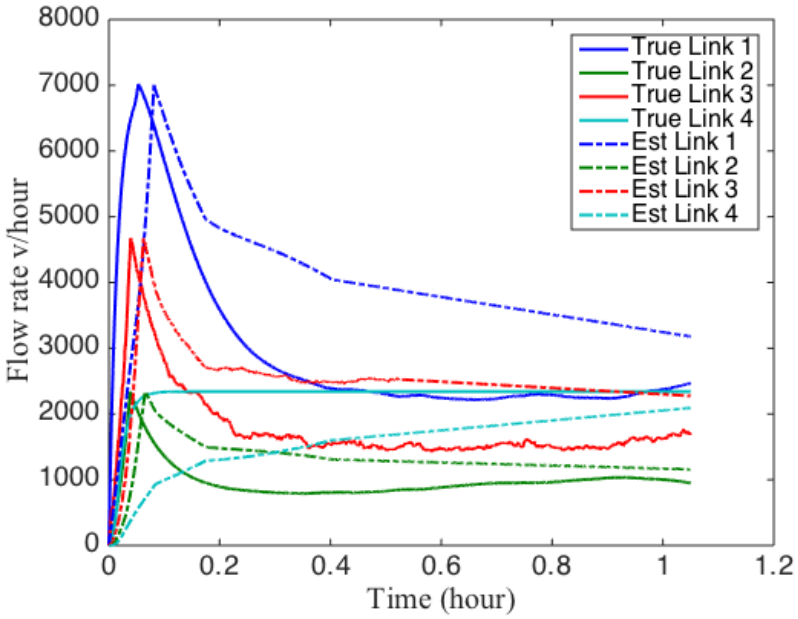


Figure 2.6: Vanilla Temporal Estimation (flow rate)

mation and Spatial-Temporal Estimation, it is interesting to find out that the Spatial-Temporal Estimation with EM-EKF achieves a better result with even fewer data. The reason is that by using Expectation-Maximization algorithm, we are not only able to make an optimal estimation of the unknown traffic states, but also can estimate an optimal model parameters such as the number of lanes.

Figure 2.10 and Figure 2.11 illustrate the results of the ultimate task: Incident Adaptation. As defined above, at the time point of 0.5h, the number of lanes in link 3 drops from 2 to 1. This results in a drop in density in link 3 and more congestion in the network. In Figure 2.10, by exploiting the adaptability of Expectation-Maximization algorithm, we are able to make an accurate estimation of flow density even without knowing the number of lanes actually drops from 2 to 1. Comparing to the Spatial-Temporal Estimation in Figure 2.8, the mean absolute error of density estimation increases from 18.5 v/mile to 21.4. This is due to the fact that without knowing the change in the number of lanes, the EM-EKF needs a transition period to react to this lane closure, and this transition period could introduce more error. This conclusion is further

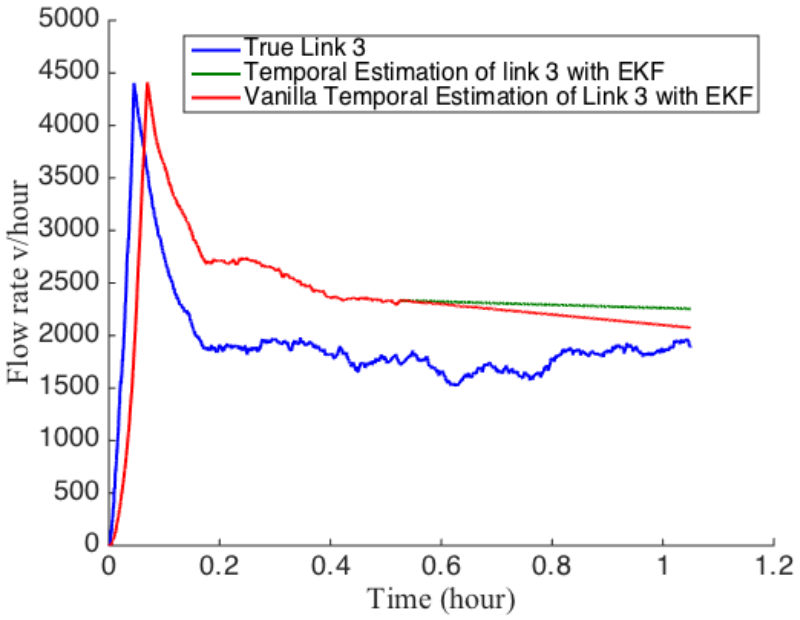


Figure 2.7: Temporal Estimation (flow rate)

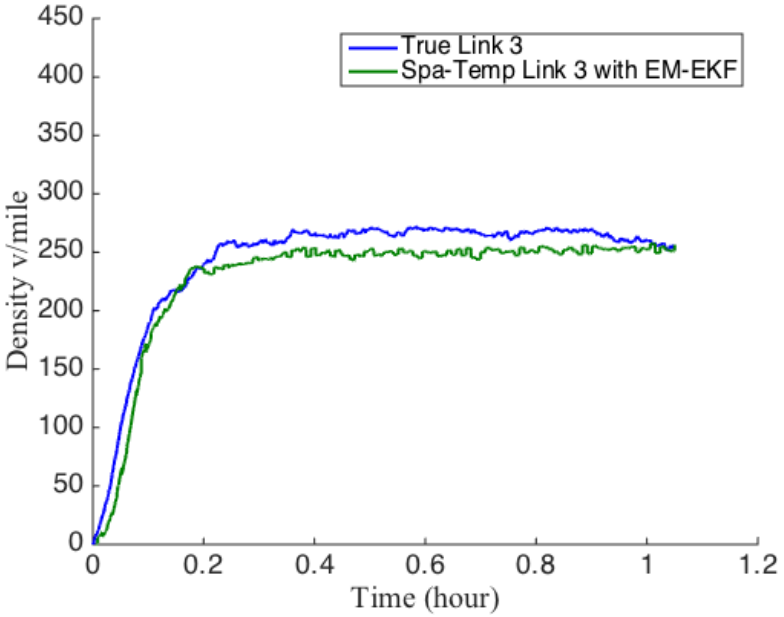


Figure 2.8: Spatial-Temporal Estimation (density)

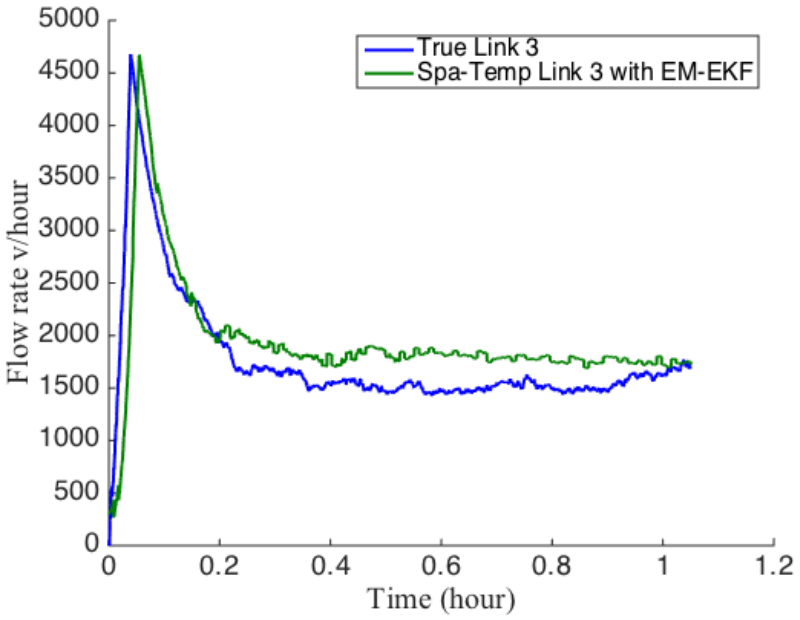


Figure 2.9: Spatial-Temporal Estimation (flow rate)

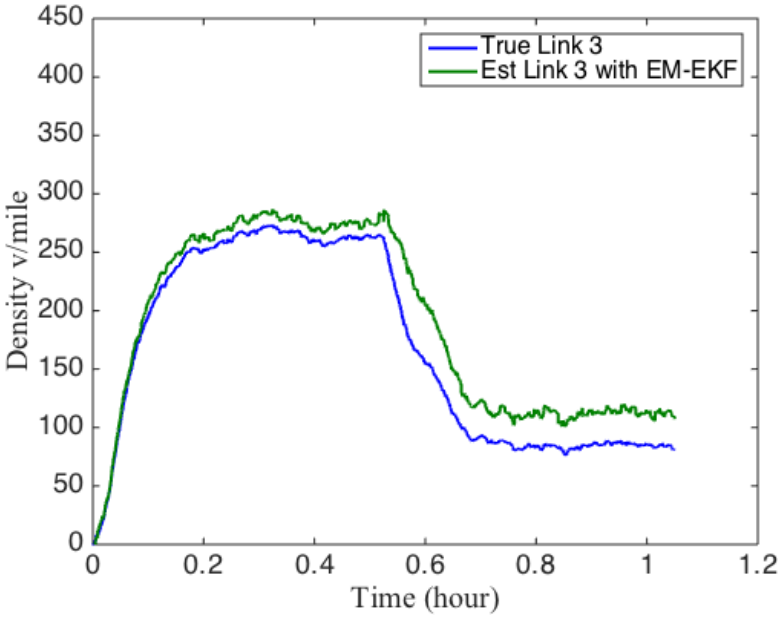


Figure 2.10: Incident Adaptation (density)

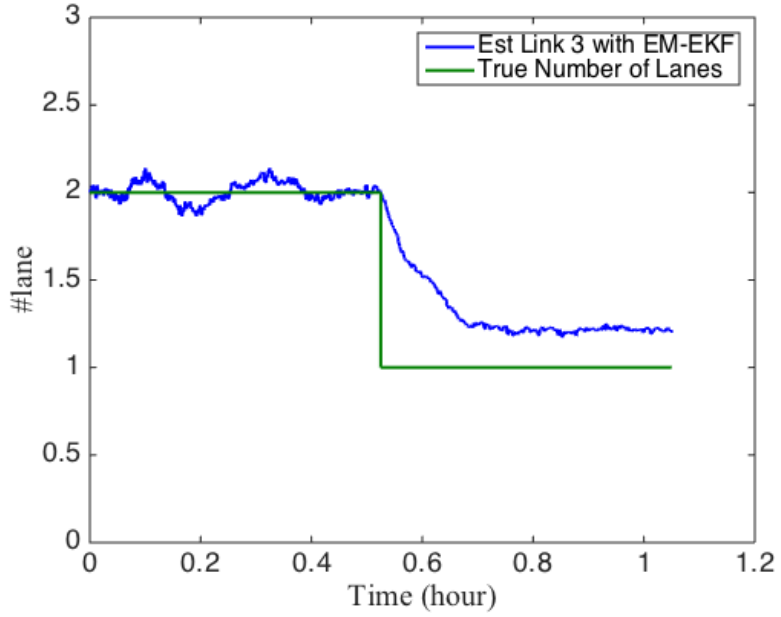


Figure 2.11: Incident Adaptation (number of lanes)

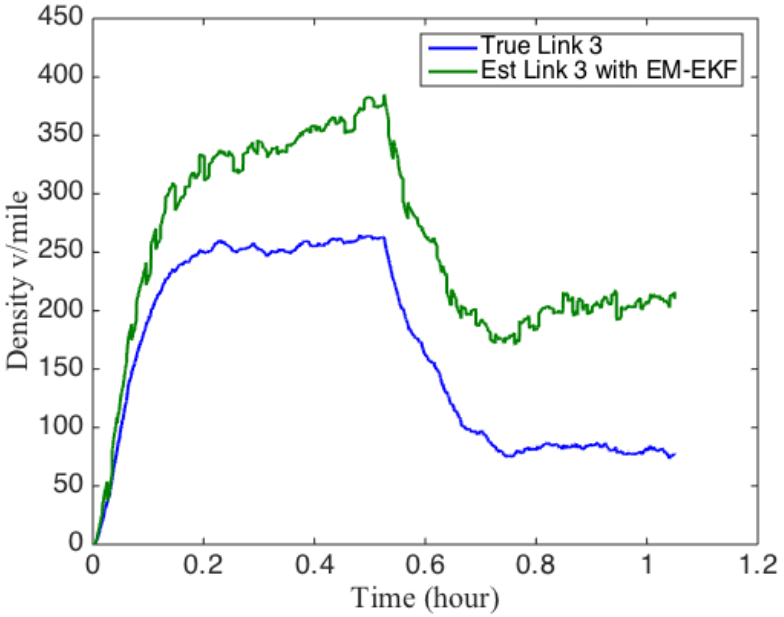


Figure 2.12: Incident Adaptation given data from link 1 and link 2

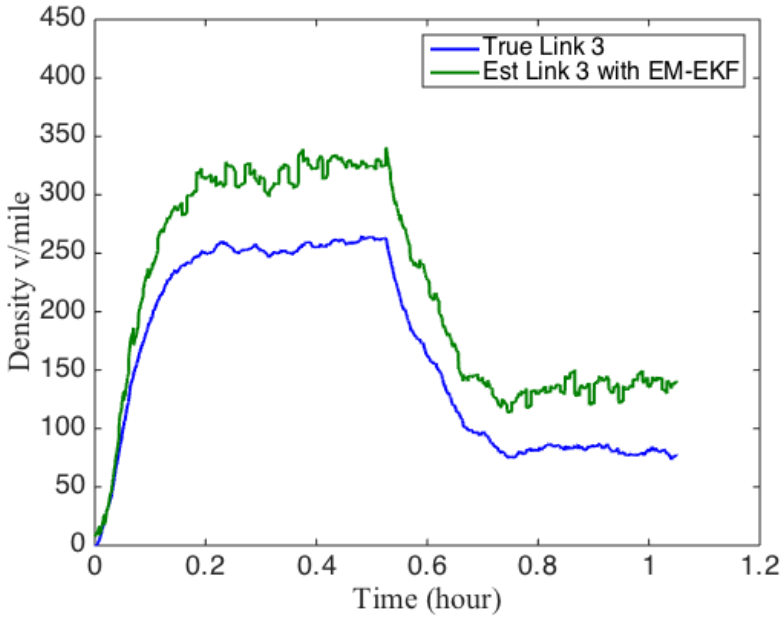


Figure 2.13: Incident Adaptation given data from link 1 and link 4

verified in Figure 2.11, where we explicitly show the change of the model parameter – number of lanes, in the EM-EKF. It can be seen in Figure 2.11 that the number of lanes drops from 2 to nearly 1 gradually in about 15-minute period.

The analysis shown in Figure 2.10 and Figure 2.11 is intended to illustrate the sensitivity of the EM-EKF algorithm to the location of the data source. Notice that in Figure 2.10 and Figure 2.11, the average speed data from link 1,2,4 is given. In Figure 2.12 and Figure 2.13, however, average speed data from link 1,2 and from 1,4 is given respectively (average speed data for link 3 at 0-0.5h is always given by the definition of Incident Adaptation). Comparing Figure 2.10, Figure 2.12 and Figure 2.13, it is easy to see that: (1) the result in Figure 2.10 is better than both the result in Figure 2.12 and the result in Figure 2.13, meaning that in this case, having more data is always better than having less data; (2) the result in Figure 2.12 is better than the result in Figure 2.13, meaning in terms of estimating the flow density in link 3, obtaining the data from link 4 has a better value than obtaining the data from link 2. Additionally, this also shows that the model can perform robustly with missing and partial data, and the amount of data

available has a direct influence on the performance of this data-driven model – the estimation error of the model tend to increase when the coverage of data is decreasing.

Additionally, Figure 2.14 illustrates the performance of EM-EKF with and without data. Same as in Figure 5, the dark blue line shows the ground truth and the green line shows the EM-EKF estimation of flow density given the speed data at link 1,2,4 at 0-1.05h and the speed data at link 3 at 0-0.5h. However, if the speed data at link 1,2,4 is given only for 0-30min, 0-36min, and 0-42min, the result of EM-EKF is shown in red, light blue, and purple, respectively. The first observation is, the the ability of Incident Adaptation in EM-EKF algorithm is enabled by constantly measured data: if there is measured data, EM-EKF is able to adapt to the incidents, and if there is no measured data, EM-EKF can act as a tuned simulator, to predict the future traffic states given the current model parameters and traffic states. For instance, in the scenario of "Given Data 0-30min" (red line), since there is no incident at 0-30min, and there is no data after 30min, the traffic states estimation after 30min can only be predicted using current model parameters, which assumes there is no incident. Therefore, even though there is a lane reduction at 30min, the EM-EKF can't adapt to this incident without data, and hence makes a less optimal traffic state estimation comparing to the ground truth. During the period of 30-42min, the EM-EKF gradually learns from the measured data and adapt to the incident. As shown in the purple line in Figure 6, if the measurement is discontinued from 42min, the EM-EKF can still make a comparatively good estimation since the model has been fully adapt to the incident. Comparing this to the light blue line in Figure 6, where only 0-36min data is given, it can be seen that in the scenario of "Given Data 0-36min", the model is only half-way towards the full incident adaptation and consequently, the estimation result lies between the red line (incident not adapted) and the purple line (incident fully adapted).

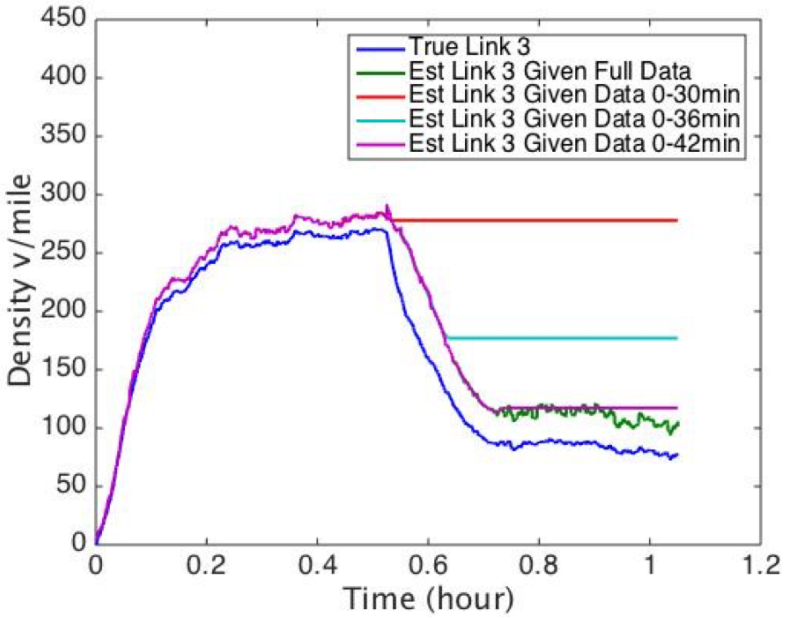


Figure 2.14: Incident adaptation over time

2.5.2 Tiny Real-world Experiment

To test the proposed methodology in the real-world, we applied the EM-EKF algorithm in a small arterial network in Washington DC area. Figure 2.15 shows the configuration of the small arterial network, which consists of eleven arterial links, named from L1 to L11 in red color. In this network, there are two microwave-based speed detectors, D1 and D2 in blue color, which is located on link L1 and link L11 respectively. These detectors are implemented by SPEEDINFO and collect data every one minute from 9:00 to 18:00 every day. Additionally, we also have GPS-based speed data from L1 to L11 collected by RITIS. Comparing to microwave-based speed data, GPS-based speed data is collected occasionally over the days. The missing GPS-based data is filled by interpolation. Also, we have Geographical Information System (GIS) data for link L1 to L11 where we can acquire the physical parameters such as the speed limit and the number of lanes. These physical parameters are used to construct the initial fundamental diagrams. Finally, the initial turning proportions are initially set to be proportional to the number of lanes and are

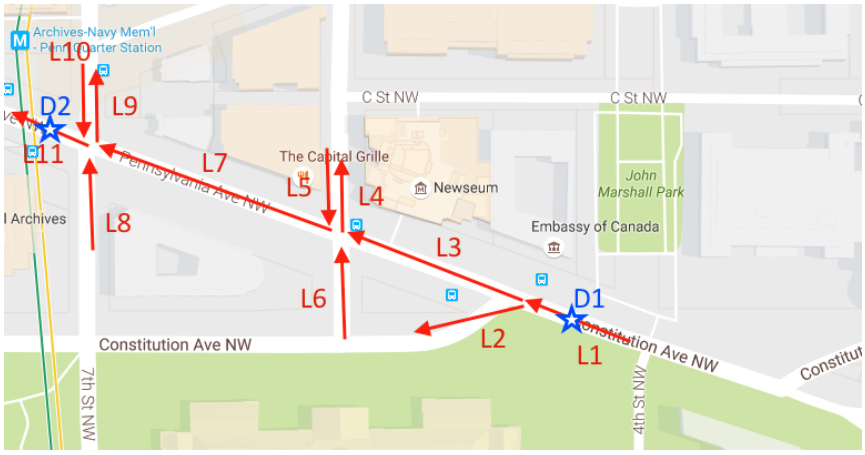


Figure 2.15: Configuration of a small road network in Washington DC

assumed to be weekly periodical. Also notice that although theoretically, the sum of the turning proportions from any link should be 1, we loosen this constraint to be a weekly periodical and limit the sum to vary between 0.8 and 1.2. This trick effectively models the ingress and egress of the parking flows in the urban roads.

In this experiment, we use the speed data from L1 to L10 (excluding L11 and D2) from February 1, 2015, to March 28, 2015, to train the model parameters such as the parameters in fundamental diagrams and turning proportions. Fundamental diagrams are assumed to be constant and turning proportions are assumed to be weekly periodical. Also it is worth to mention that to mitigate the influence of the traffic lights, all the speed data are smoothed by a moving average function over a 15-minute window before usage. Then on March 29, 2015, we aim to estimate the travel speed at link L11 given the travel speed at L1 to L10 in real-time. The estimated travel speed at L11 and the detected travel speed at L11 from D2 is compared in Figure 2.16. It can be seen that in general, the estimation is accurate, with the average error rate of 8.5%. Although some peaks are not fully captured by the estimator, the general decreasing or increasing trend of travel speed is captured by the EM-EKF algorithm. This small experiment proves that EM-EKF could have the potential to make traffic state estimation in an urban arterial network.

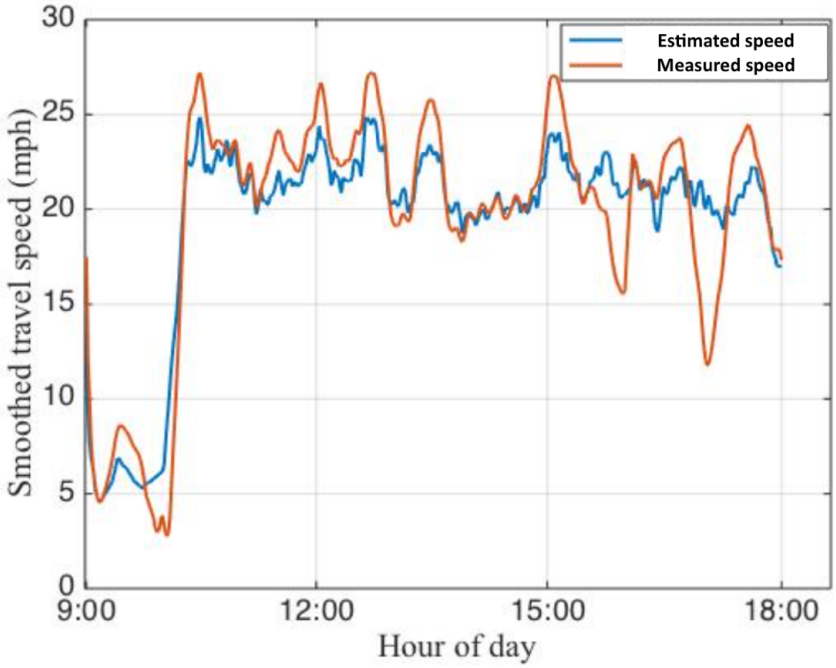


Figure 2.16: Travel speed estimation on March 29, 2015, at Washington DC area

Chapter 3

Social Media in Traffic Incident Detection

3.1 Problem Statement

In this chapter, we propose to use social media as a new efficient way to detect traffic incident and hence foster a more accurate traffic state estimation. In particular, we develop a methodology to crawl, process and filter tweets that are posted by generic users and shared with the public. Because public tweets exhibit special linguistic features, such as acronym and short words, we will use sophisticated natural language process (NLP) algorithms to dynamically learn the twitter language and identify geo-locations and incidents features. We also study the efficiency and effectiveness of twitter-based incident detection. Specifically, this chapter addresses the following questions.

- Is there a way to identify incident-related tweets from other tweets?
- How can we extract geo-location information by mining tweets' texts?
- Which type of Twitter users contribute the most incident-related information, individual users or influential users (namely public twitter accounts)?
- Are there any particular time periods or areas where people tend to tweet more often to report incidents?

- Can social media enhance, or even replace traditional traffic incident report systems?

3.2 Tweets and Their Relevance to Traffic Incidents

There are over 400 million tweets posted every day on Twitter all over the world, with over 100 million daily active users. It is a huge resource for people to share information they observe. Tweets are essentially short messages, limited to 140 characters. Twitter users consist of authoritative users (such as public agencies) and individuals.

Individual users can tweet to share firsthand (original) information regarding incidents. Some examples are,

- "95 SB crawling near Betsy Ross Bridge" with a picture attached
- "per traffic, 676 is not the blue route. Accident is at Vine St Expy and the Schkuykill Expy"
- "Fast food workers arrested as union-backed protest blocks traffic in front of Penn Ave McDonald's"
- "Whoa! Bad accident at Tulip & Unruh NorthEast Philly - Courtesy of Tacony Town Watch"
- "ramp off 376 while they had to close the Carnegie exit. At least that could spare some traffic!"
- "Avoid 76 west, approaching Vare from Walt, apparent ax, traffic is nuts, Ben is also backed up"

Some tweets with the original information may be sufficient to infer the time and location of a traffic incident. However, it may be difficult for automatic tweet processing to extract that information, because the linguistic expression of incidents can vary significantly among different users. For example, there are numerous ways to discuss "traffic congestion", such as "bumper-to-bumper traffic", "stuck there for 10 min", and "a similar situation Harry experienced yesterday".

Most of the tweets language are very personalized. In addition, accurate location information of incidents, as part of tweets' attributes, is usually not available, but it may be inferred by mining tweets' texts. For privacy concerns and by default settings of iOS/Android systems, Global Position System is disabled and almost all tweets are posted without latitude/longitude information (99.9% of tweets we obtained from Twitter APIs for our case study). However, tweets' texts may contain partial location information. The expression of locations in tweets is highly irregular, such as "on Forbes in front of Hamburgh hall" or "halfway to Phily". It is nearly impossible for an individual to text locations in full details using smartphones in a short time period. Ways of solving these issues in extracting location information will be discussed in the following sections.

News and public agencies (namely, influential users or IU for short) also tweet to disseminate incident information. Examples of typical Influential Users and their tweets are "Turnpike Roadwork on Pennsylvania Turnpike I-476 northbound between Exit 31 - PA 63 and Exit 44 - PA 663 affecting the right lane" by Pennsylvania Turnpike Twitter account, and "Water Main Break in the Strip District on 28th Street between Railroad St & Smallman St" by the City of Pittsburgh. These tweets are highly standardized, with much detail, and therefore easy to mine and geocode. This type of tweets serves as a critical source for incident data.

From now on, we call tweets that are related to traffic incidents "TI tweets", and those unrelated to traffic incidents "NTI tweets". Specifically, a TI tweet is one contains information about a traffic incident, implies abnormality on the transportation infrastructure, or indicates a (potentially) significant disruption to the traffic. For example, "the roads are really slippery to drive on" is TI tweets, because it contains the information (being slippery) about the current traffic infrastructure (the roads). Tweets describing recurrent traffic do not count as TI tweets. For instance, "Parkway East WB slow traffic" posted at 5:25 pm (namely the afternoon peak hours) indicating recurrent peak hours. Some tweets are used as an outlet for expressing individual emotions. For instance, "I'm so sick of crazy bike riders!" or "If you drive so slow, why get on

a highway!!!". Clearly, they are classified as NTI tweets. We carefully examine tweets in the training set by manually labeling tweets that are related to the traffic incidents only. Note that useful tweets sharing firsthand information can be forwarded by followers. Texts of re-tweets in addition to firsthand tweets are not mined in this study since they simply resemble the same incident information.

3.3 Methodology: Twitter Data acquisition, Processing and Analytics

We develop a methodology to translate tweets into traffic incident information. Figure 3.1 illustrates the procedure conceptually. We conduct the initial data acquisition from Twitter servers, followed by an iterative process, namely the adaptive data acquisition where we crawl tweets and establish our dictionary of "words" relevant to traffic incidents. The process of adaptive data acquisition also selects the most important features to form a feature space that is informative and non-redundant. Next, tweets are mapped into this feature space and classified by a well-trained Semi-Naive-Bayes (SNB) classifier as either TI or NTI tweets. All the TI tweets are then pushed through a geo-parser and geocoder, to determine their locations.

Each tweet T is modeled by a tuple,

$$T = \{[tweetID], [userID], [text], [timeStamp], [geoLocation], [label]\}$$

where $[label]$ is a label for the tweets, either TI or NTI. Tweets in the training set are manually labeled to build our model. Labels of tweets in the test set are estimated using the model and then compared to the manual (true) labels. $[timeStamp]$ is one of the attributes of tweets obtained from Twitter APIs. $[geoLocation]$ is estimated using the geocoder discussed below.

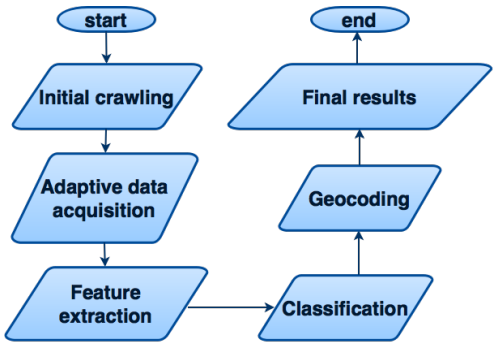


Figure 3.1: The work flow of Twitter data acquisition, processing and analytics

3.3.1 Twitter APIs and Initial Data Crawling

Twitter offers two types of crawling APIs that allow users to query tweets by keywords, user IDs, and time/date. The first type, REST APIs, allows users to submit queries to retrieve recent or popular tweets. The REST query format includes a centroid (latitude, longitude), a radius (e.g., 0.5 mile), and a set of keywords with the support of operators, including AND, OR, and EXCLUDE (e.g., "traffic AND (accident OR collision)"). Tweets data consist of user information, text, time posted, times of re-tweets, latitude and longitude (if any), and the referred URL for figures/videos. Notice that the location data is only available for a small portion of collected tweets (less than 0.1% in our case studies) and therefore texts are used as a main source for textual mining and geocoding. User information includes userID, nationality, residence city, and the number of following/followed users. IUs can be identified as those Twitter accounts that are created by public agencies or media for disseminating traffic information. In this study, IUs are collected manually by Google searching twitter accounts of major news channels and public agencies.

Twitter REST APIs are 100% free of charge but has certain limitations. For example, API calls are limited to 350 queries every 15 minutes for one user account, or 3500 total tweets per REST query, whichever comes more restrictive. To fully utilize all the REST API calls, the developers use an OAuth 2.0 protocol to access data. Twitter also encourages third-party libraries for various programming languages including Python, PHP, Java, and Javascript. A popular Java

library, Twitter4J, is used in this research.

The second type, Twitter streaming APIs, requires users to keep an uninterrupted HTTP connection to access the most recent (mostly within 1 day) Twitter data up to 1 percent of public tweets. The streaming query format includes a combination of a centroid (latitude, longitude) and a radius (e.g., 0.5 mile) or a set of keywords with operations similar to REST APIs. However, these APIs do not support the joint query of locations and keywords.

For the purpose of establishing a machine learning model based on historical tweets, it is clear that REST APIs are the most effective ways since it allows us to query with locations and keywords simultaneously. In addition, it is also critical to select trustworthy IUs to access their timelines and historical tweets. To obtain incident information from tweets to the fullest extent, it is essential to develop a combination of keywords that leads to the best recall and reasonable precision. Recall and precision as basic scores of the data acquisition system, defined as follows:

$$Recall = \frac{A \cap B}{A}$$

$$Precision = \frac{A \cap B}{B}$$

where A is the set of all TI tweets in a time period, and B is the set of all acquired tweets in the same time period. The ideal goal is to achieve as much precision and recall as possible simultaneously, which means all acquired tweets are only TI tweets and all TI tweets in the pool are acquired.

However, it is usually impossible to achieve a 100% recall, or to precisely estimate the recall, given the limitations on the total number of tweets accessible through REST API. And there is hardly any ground truth of a full set of TI tweets. However, it can be estimated using the ratio of E over Q where Q is denoted by the number of TI tweets of all the tweets of randomly selected test users from Twitter, and Q is denoted by the number of tweets crawled based on REST API (with a specific set of query parameters) intersecting those test users' tweets. A 100% precision

is also hard to achieve because there always exist some tweets matching query keywords but are not related to traffic incidents. Therefore, the goal of the data acquisition process is to achieve reasonably high recall and precision, such that we obtain as many TI tweets as possible through free-of-charge Twitter APIs.

3.3.2 Adaptive Data Acquisition

The REST APIs require a list of keywords to perform queries. We can come up with an initial set of keywords, namely an initial dictionary. However, the recall and precision are usually low because those keywords do not cover all the language indicating incidents. To ensure the best quality and maximum quantity of TI tweets that could be possibly acquired, we need to expand the dictionary to retrieve more TI tweets. One strategy is to learn from all words of the queried tweets that are not part of the dictionary and select those words relevant to incidents. We can then add those new words to the dictionary and perform new queries to obtain more TI tweets in a new iteration. This is also known as the adaptive data acquisition as this strategy of data acquisition can be implemented over time to adapt to the most recent twitter language. In each iteration, we will manually label TI tweets, and count the frequencies of new words. It is believed that the more frequent a word (or a combination of words) is used in all TI tweets, the more likely it contains incident and geo-location information. The process of adaptive data acquisition is described in Figure 3.2.

First, we collect initial words that are related to traffic incidents, namely "seed words", to assemble initial queries. The seed dictionary includes, but are not limited to, “traffic, accident, road, avenue, car, bike, truck, driver, injured, congestion, slow, I-, PA-, US-, exit, mile, stop, -crazy, -hate, -! ”. The "-" symbol leading a word implies the exclusion of this word in a query. Note that a query with keyword "traffic" pulls out tweets containing "traffic", while a query with keywords "traffic -hate" extracts tweets that contain the word "traffic" and do not contain "hate". The dictionary of the initial "seed words" is notated as S_0.

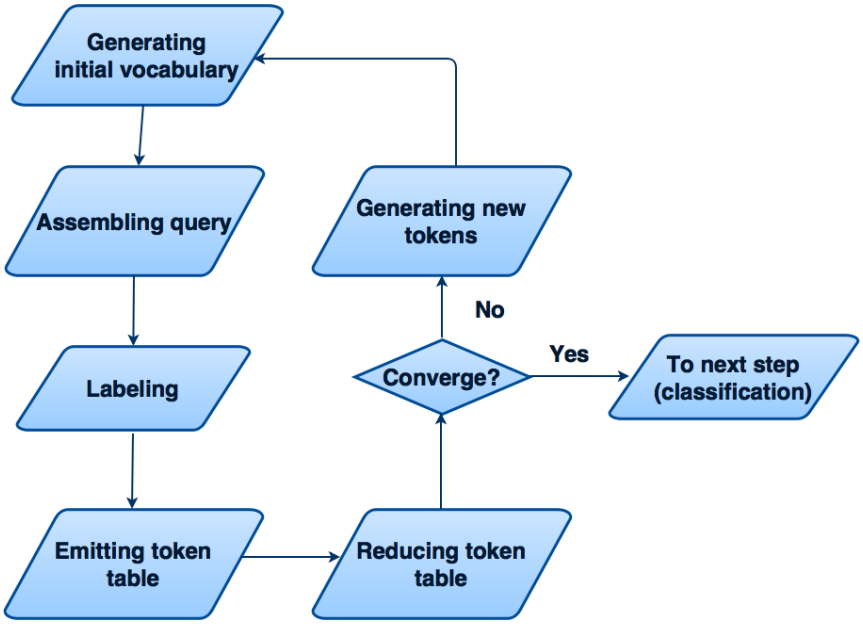


Figure 3.2: Flow chart of adaptive data acquisition

For the ν-th iteration, with a dictionary S_ν, an expansion of the dictionary is made to include the first two synonyms of each of the words in S_ν. The synonyms are extracted from the open-source WordNet database. Therefore we have an expanded dictionary:

$$S'_\nu = E(S_\nu)$$

where E is the expansion operator and also notice that S'_ν is a superset of S_ν:

$$S'_\nu \supseteq S_\nu$$

In addition, a batch of queries is constructed by using "OR" criteria to contain each word and combinations of any two, three and four words from all the words in S'_ν:

$$Q_\nu = C(S'_\nu)$$

where Q_ν is the queries constructed from the dictionary S'_ν with the operator C. The size of Q_ν increases exponentially with respect to the size of S'_ν. Note that the REST API here pulls out tweets randomly from the entire tweets pool with a total limit. Therefore, a query with one

keyword "A" does not necessarily pulls out all tweets with the keyword. For the same reason, the resultant tweet set is not necessarily a superset of what we obtain from a query with two keywords "A" and "B". Thus, the queries consisting of combinations of two, three and four words from the dictionary can acquire as many tweets as possible with a reasonable acquisition speed. The set of tweets acquired are denoted by,

$$T_\nu = F(Q_\nu)$$

where F is the operator where we send queries to REST API and receive a set of tweets.

From all the acquired tweets (T_ν), we manually label the TI tweets and NTI tweets. a TI tweet is one contains information about a traffic incident, implies abnormality on the transportation infrastructure, or indicates a (potentially) significant disruption to the traffic. For example, two tweets are labeled in Table 3.1.

tweet.text	tweet.label
CLEARED: Multi vehicle accident on I-376 eastbound at Mile Post: 73.0.	1
Apple shows off iPadâĂŹs improved camera	-1

Table 3.1: An example of tweet labeling

Next, the tweets are processed through a specially designed tokenizer to retrieve incident information. The tokenizer sees each word of nouns, verbs, adverbs and adjectives as a token, and removes articles, prepositions, the verb "be" and other words that do not carry physical meanings. It also uses a dictionary of road names to identify tokens of road names. For example, the tokenizer will transform the word "I-376" to the token "*road-name*". Regardless of what roads tweets imply, the road name and number are likely to indicate an incident. Using the example in Table 3.1, the tokenized tweet can be expressed as: ["cleared:" "multi" "vehicle" "accident" "*road-name*" "eastbound" "mile" "post:" "73.0."]. Note that prepositions, such as "on" and "at" are neglected by the tokenizer.

Following the Map-Reduce framework, the combinations of tokens with the labels of the tweets are emitted. Table 3.2 explain the emission using the same example. Then, a "reducer" is applied to the entire acquired tweets T_0 to count the total number of positive and negative labels for each of the token combinations. Examples are given in Table 3.3. For instance, the token combination, "cleared:" and "multi", has 12 positive counts and one negative count, indicating that there are in all 13 occurrences of ["cleared:" "multi"] in the same tweets in this batch of queries, and 12 of them are TI tweets and 1 of them is an NTI tweet. After building this table, we can identify those token combinations with the maximum positive counts (namely with the greatest positive correlation with TI tweets) and the maximum negative counts (with the greatest negative correlation with TI tweets).

To limit the number of new words added to the dictionary in the next iteration, a total of K new token combinations are chosen. We set the chosen number of new positively (negatively) correlated token combinations proportional to the percentage of TI tweets (NTI tweets) in the all acquired tweets, denoted by N_p and N_n respectively,

$$N_p = K\frac{N_{TI}}{N}$$

$$N_n = K\frac{N_{NTI}}{N}$$

where N, N_{TI} and N_{NTI} are the number of total acquired tweets, TI tweets and NTI tweets respectively.

The final step of the ν-th iteration is to add the chosen K new tokens into the dictionary,

$$S_{\nu+1} = S_\nu \cup W_\nu$$

where $S_{\nu+1}$ is the set of words to assemble queries for the next iteration, and W_ν is the set of K chosen tokens.

Token combinations	Tweet label
cleared:	1
multi	1
.	.
.	.
"cleared:" "multi"	1
.	.
.	.
"cleared:" "multi" "vehicle"	1
.	.
.	.
"apple"	-1
.	.
.	.
"apple" "shows" "off" "ipad" "âĂŹs" "improved" "camera"	-1

Table 3.2: An example of token emission

The iterations go on until we do not find new TI tweets or identifying new token is no longer cost-effective. The convergence criteria is therefore set to that the percentage of TI tweets in all newly acquired tweets in the ν-th iteration, r_ν, is very small, e.g., 1%. $r_\nu = 1\%$ indicates that we will need to label 100 tweets based on the pre-defined incident dictionary (roughly 30 min-1 hour in our experiments) in order to find one TI tweet, and the process is no longer cost-effective and should terminate.

Denote A_ν the number of new tweets acquired in the ν-th iteration, and K_ν the number of tweets that are not yet acquired from the Twitter pool. Suppose r'_ν is the percentage of TI tweets in those K_ν tweets not yet acquired. In the $(\nu+1)$-th iteration, there are $K_\nu - A_\nu$ amount of new

Token combinations	Positive counts	Negative counts
cleared:	15	1
multi	12	4
.	.	.
.	.	.
"cleared:" "multi"	12	1
.	.	.
.	.	.
"cleared:" "multi" "vehicle"	4	0
.	.	.
.	.	.
"apple"	0	10
.	.	.
.	.	.
"apple" "shows" "off" "ipad" "âĂŹs" "improved" "camera"	0	1

Table 3.3: An example of reducing labels

tweets left in the pool, and the percentage of TI tweets is,

$$r'_{\nu+1} = \frac{K_\nu r'_\nu - A_\nu r_\nu}{K_\nu - A_\nu}$$

Since we use a pre-define dictionary consisting of words related to incidents, the percentage of TI tweets in the newly acquired tweets is generally greater than that of all tweets in the pool, namely $r_\nu > r'_\nu$. Therefore, r'_ν decreases as the iteration goes on. When r'_ν is sufficiently small, the percentage of TI tweets acquired in a new iteration can be smaller than a threshold (e.g., 1%) when the convergence criteria are met.

As a result of the adaptive data acquisition process, the dictionary contains single words and combinations of some words that are positively correlated with being a TI tweet, which serves as features. Those features are actually selected from all the words used in all acquired tweets, in order to form an informative feature space with minimal redundancy. We discuss how to project each tweet onto the feature space for classification of whether this tweet is a TI tweet or not.

3.3.3 Classification

Classification on TI/NTI Tweets

As described in Section 3, tweets are acquired by two methods: queries on keywords and queries on IU accounts. Queries from both sources hold fundamentally different properties: queries on keywords need to ensure an adequate recall and therefore come with a low precision, whereas queries on IUs' timelines result in a limited number of tweets, and have extremely high precision. Therefore, a filter is necessary to remove those acquired NTI tweets from the data set. We briefly describe the Semi-Naive-Bayes model adopted in this paper to filter out NTI tweets.

Suppose Y is the label whether or not a tweet is TI tweet, and X is the projection of a tweet onto a feature space, given by the vector $X = [X_1, X_2, X_3, ..., X_I]^T$, where I is the dimension of the feature space, namely the total number of words and word combinations in the dictionary that are positively correlated with incidents. To estimate the probability of a tweet X being a TI tweet, the Bayes formula is given by,

$$P(Y|X) = \frac{P(XY)}{P(X)} = \frac{P(Y)P(X|Y)}{P(X)} \propto P(Y)P(X|Y)$$

One issue is how to efficiently calculate the joint conditional probability $P(X|Y)$, because X is usually in very high dimensions. To simplify this calculation, Naive-Bayes makes a "Naive" assumption on the conditional independence of all features:

$$P(X|Y) = P(X_1|Y)P(X_2|Y)P(X_3|Y)...P(X_k|Y)$$

However, conditional independence is not realistic in this study. For instance, suppose $X = [X_1, X_2]^T$ where X_1 is 1 if the word "bus" occurs, and 0 otherwise. X_2 is 1 if the word "stop" occurs and 0 otherwise. Then $P(X_1|Y = 1)$ and $P(X_2|Y = 1)$ is the probability of the occurrence of the word "work" and "zone" given a TI tweet, respectively. Clearly, $P(X_1X_2|Y) \neq P(X_1|Y)P(X_2|Y)$ because "work zone" is a commonly used term in a tweet to alert road closures, but not each of these two words alone. In other words, the occurrence of the words "work" and "zone" in a TI tweet is highly correlated.

The intention of using a Semi-Naive-Bayes is to take into account those correlated features whereas still holding a part of the "naive" assumption to avoid computation in high dimensions. The Semi-Naive-Bayes classification model differs from the Naive Bayes model by consolidating those correlated features,

$$P(X|Y) = \prod_{(i,j)\in\sigma} P(X_i, ...X_j|Y) \prod_{I}^{J} P(X_n|Y)$$

where all features are ordered by those correlated feature tuples first, followed by independent features. σ is the set of the positions in the order for the first and last feature in a correlated tuple, and the features with the position from J to I are all independent features. Fortunately, features have been selected in the adaptive data acquisition process with the consideration of word combinations. Note that those single words and word combinations with the highest frequencies are selected as part of the dictionary. Therefore, we can directly apply those words and combinations to form a feature space for the Semi-Naive-Bayes classification by assuming that each of those single words and word combinations can occur in TI tweets independently.

In the Semi-Bayes-Classifier, each probability term can be computed by,

$$P(X_i, ...X_j|Y) = \frac{\#\{X_i, ...X_j, Y\}}{\#\{Y\}}$$

where the notation $\#\{A\}$ means the number of label/word $\{A\}$ in the pool of all acquired tweets.

Given a feature vector X of a tweet, we classify this tweet by,

$$Y^* = \arg\max_Y P(Y|X)$$

where $Y^* = 1$ indicates this tweet is a TI tweet, and 0 otherwise.

For example, a tweet reads "Pkwy W delays begin before the top inbound, very slow outbound from Green Tree to work zone.", where we suppose the feature space is defined by ("pkwy","delay", "work zone", "crash"), then this tweet's coordinate in this feature space is (1,1,1,0). The posterior probability of Y is given by

$$P(Y=1|X) \propto \frac{N_{TI}}{N} \times \frac{\#\{\text{"}work+zone\text{"}=1, Y=1\}}{\#\{Y=1\}} \times \frac{\#\{\text{"}pkwy\text{"}=1, Y=1\}}{\#\{Y=1\}} \times \frac{\#\{\text{"}delay\text{"}=1, Y=1\}}{\#\{Y=1\}} \times \frac{\#\{\text{"}crash\text{"}=0, Y=1\}}{\#\{Y=1\}}$$

$$P(Y=0|X) \propto \frac{N_{NTI}}{N} \times \frac{\#\{\text{"}work+zone\text{"}=1, Y=0\}}{\#\{Y=0\}} \times \frac{\#\{\text{"}pkwy\text{"}=1, Y=0\}}{\#\{Y=0\}} \times \frac{\#\{\text{"}delay\text{"}=1, Y=0\}}{\#\{Y=0\}} \times \frac{\#\{\text{"}crash\text{"}=0, Y=0\}}{\#\{Y=0\}}$$

Classification on Incident Categories of TI Tweets

In order to further identify the property of the TI tweets, another classifier is built to assign a categorical label to the TI tweets. Specifically, a Supervised Latent Dirichlet Allocation (sLDA) [39] is used in our model. The benefits of using sLDA over other classifier are as follows:(1) sLDA is a classical topic model, which is widely used over the years in the domain of Natural Language Processing; (2) it has relatively fast training and prediction runtime with the technique of variational inference [67]; (3) it is able to tell not only the categorical label but also the underlying structure of how the words are generated; (4) comparing to the classical topic model [12] or mixture model, sLDA is able to optimize over both the likelihood over data as well as the deviation of predicted outcomes from the labels.

As a variation of traditional topic model, sLDA shares the similar generative process as Latent Dirichlet Allocation (aka topic model) represented in the plate formulation in Figure 3.3. With underlying K topics(categories) and for each tweet with length N:

1. Draw topic proportions $\theta|\alpha \sim Dir(\alpha)$
2. Fore each word in a tweet:
   (a) Draw a topic of that word according to the multinomial distribution of the topic proportion of the tweet: $z_n|\theta \sim Multi(\theta)$

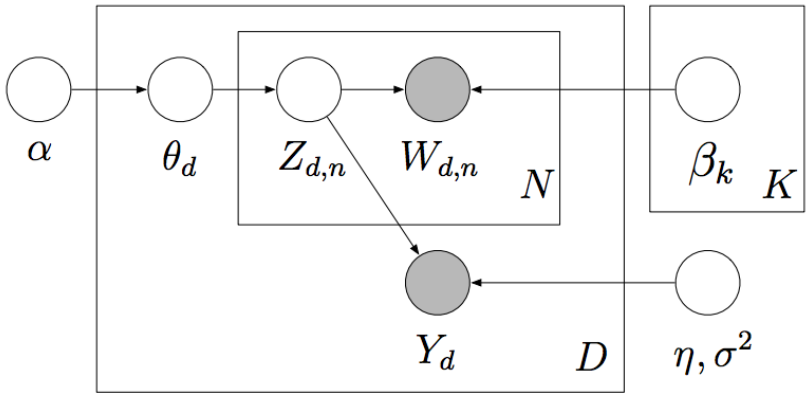


Figure 3.3: Plate representation of sLDA [39]

(b) Draw the word $w_n|z_n$ from the multinomial distribution of words over the topic: $w_n \sim Multi(\beta_{z_n})$, where β_{z_n} is the multinomial parameter of the topic z_n

3. Draw the response variable $y \sim N(\eta z_{1:N}^{-}, \sigma^2)$ over an normal distribution.

The inference of unknown latent parameters over sLDA model follows the Expectation-Maximization(E-M) algorithm over the posterior distribution of α, β, η and σ^2 described in [12]. Additionally, in this paper we define five traffic incident categories for labeling and classification:

1. Accidents
2. Road work
3. Hazards & Weather
4. Events
5. Obstacle Vehicles

3.3.4 Geocoding

After the classification, we have identified all TI tweets in the acquired pool. Next, we will extract their location information and geocode them in GIS.

The geographic location information carried by tweets is rich but very noisy. There are gen-

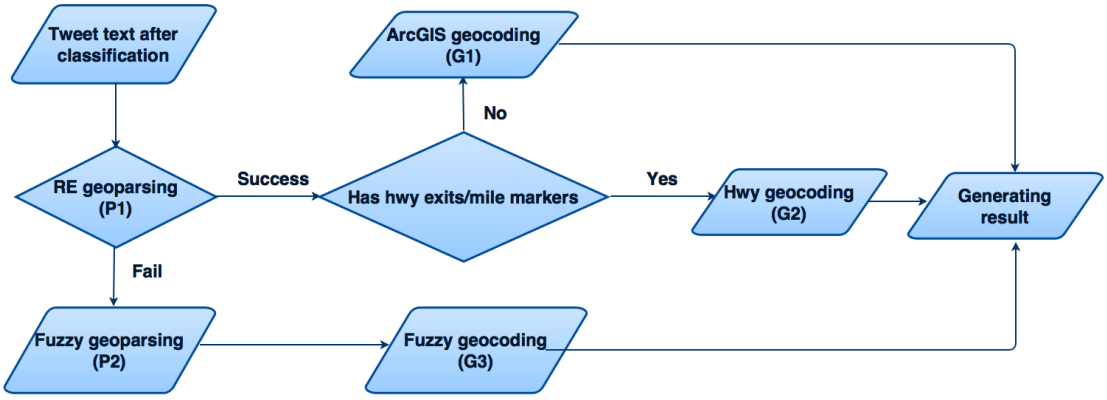


Figure 3.4: Flow chart of tweets geocoding

erally three types of location information. (1) A tiny portion of tweets carry latitude/longitude coordinates, and they are usually tweeted from geo-tagging enabled smart phones. In our experiments, this portion is never greater than 0.1%. These coordinates relate to the locations where users posted the TI tweets, but are not necessarily where those incidents are; (2) Some tweets are posted by accounts whose profiles are shared with the public, such as city, country, and sometimes finer-grained business names and street addresses of the business. Unfortunately, this type of location information generally does not imply incident locations; (3) Road names and points of interest may be referred in tweet texts. The main objective of our geocoding algorithm is to extract location information from the third type, namely tweet texts, and map each TI tweet to the GIS if possible.

The general idea is to first identify those words representing road names and/or point-of-interest (POI) names, followed by a geocoder that translates those names to latitude/longitude coordinates of the incident. Some tweets, especially those tweeted by IUs, report incidents with highway exit numbers or mile markers. In that case, we build a GIS to geocode the exit numbers or mile markers into latitude/longitude coordinates. The process of geocoding tweet texts is conceptually depicted in Figure 3.4.

A geo-parser is a machine that receives input of a string and produces structured and segmented strings that contain only geographical information. As shown in Figure 3.4, we use

two geo-parsers. The first one (P1) is to carefully implement a large set of Regular Expressions (REs) to extract road names, intersection names, highway exit numbers, and highway mile markers. When the REs set is sufficiently large to cover all roads in a region, its geo-parser can work well to extract geographical information. However, it cannot process the names of the point of interests commonly referred to in tweets, such as "Hamburg Hall" (a landmark building in Pittsburgh) and "Squirrel Hill" (a local neighborhood in Pittsburgh). Whenever P1 does not work, the secondary geo-parser (P2) developed by [20] is adopted, where a fuzzy language matching algorithm is implemented to parse those words relevant to locations. Those fuzzy words are specified in a pre-defined dictionary. Comparing to P1, P2 can process point of interests but not road names and numbers. As shown in Figure 3.4, the strategy is to apply P1 to a tweet, and whenever P1 fails, P2 is used instead.

P1 geo-parses a tweet following a structure shown in Table 3.4. It identifies either a segment of highway with starting and ending mile marker specified, a specified road, or intersections of up to three roads/highways. For instance, a tweet reads "Accident on I-376 westbound between Mile Post: 61.0 and Mile Post: 60.0. There is a lane restriction." Using P1, the parsing result is shown in Table 3.5.

The output of geo-parsers is then translated to latitude/longitude by geocoders. If a tweet is processed by the fuzzy geo-parser (P2), the output words are fed into a Gazetteer [20] to identify the location. For the tweets processed by the RE geo-parser (P1), there are two possible geocoders. If a tweet has road names or intersection names without specifying highway mile markers (e.g., an arterial road), then the ArcGIS geocoder (G1) is used to generate latitude and longitude coordinates. However, a major drawback of G1 is that it cannot geocode those mile-markers or exit numbers of highways. If that is the case, a highway geocoder (G2) should be built. For the case study in this paper, we collect the GIS of all highways in Pennsylvania, and map the mileage of each of all highway junctions to a pair of latitude and longitude. G2 has some limitations and can be enhanced in the future research. It currently cannot compile vague

keys	meaning
road1:	the road name mentioned
road2:	the second road name mentioned
road3:	the third road name mentioned
hwy1:	the highway name mentioned
hwy2:	the second highway name mentioned
hwy3:	the third highway name mentioned
hwy1-mm1:	the starting mile-marker/exit number of the highway
hwy1-mm2:	the ending mile-marker/exit number of the highway
relational-word:	the relational word used such as "near", "cross", "intersection", etc.
original-text:	the original tweet text

Table 3.4: The data structure of RE-based geoparser

relational words such as "to the north of" or "near". It does not correct misspelled road/highway names.

3.3.5 Case Studies: Pittsburgh and Philadelphia

In this paper, tweets from two regions, Pittsburgh and Philadelphia, in September 2014 have been crawled and processed to retrieve incident information. Furthermore, in order to evaluate our Twitter-based incident detector in terms of timeliness, accuracy, and effectiveness, we performed extensive testing and validation against "ground truths" such as manually classified labels, RCRS (Road Condition Report System) incident data, 911 Call For Service (CFS) data, as well as RITIS travel time data. Our testing and validation processes are summarized as follows:

(1) Testing on the classifier and geocoder: we test the accuracy of our classifier and geocoder against the "ground truth" - our manually labeled data.

keys	value
road1:	
road2:	
road3:	
hwy1:	I-376 WB
hwy2:	
hwy3:	
hwy1-mm1:	61.0
hwy1-mm2:	60.0
relational-word:	"between"
original-text:	Accident on I-376 westbound between Mile Post: 61.0 and Mile Post: 60.0...

Table 3.5: An example of geo-parsing result

(2) Validation with incidents data: we compare the final output of the Twitter-based incident detector - the time, location, and category of the geocoded TI-tweets (we call it "Twitter incidents" for simplification) against the "ground truth" - the combination of RCRS (Road Condition Report System) incident data and 911 Call For Service (CFS) data.

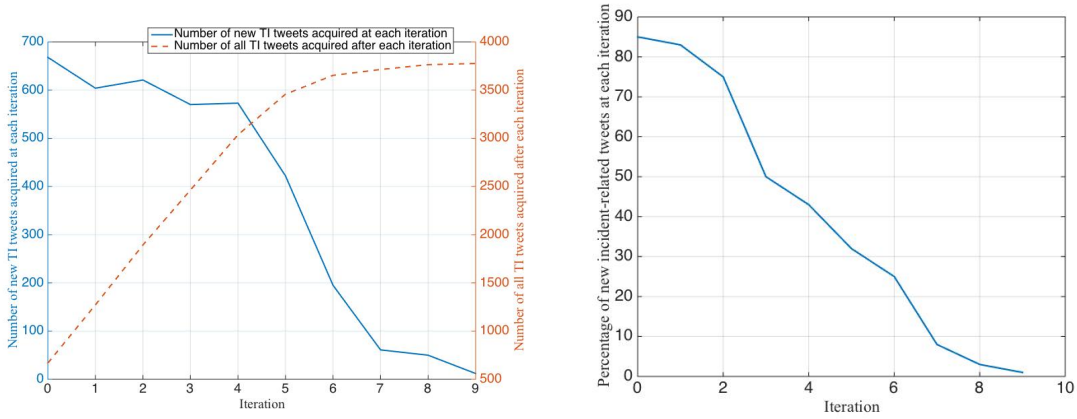
(3) Validation with travel time data: we compare the travel time at the time and location of the Twitter incidents against the historical travel time at the same location and the same time of week.

Twitter Data Acquisition

We first conduct the adaptive data acquisition following the aforementioned methodology. In particular, queries with keywords are made through REST API. In addition, we also query tweets posted by a list of in all 46 active Influential Users (IUs) such as Department of Transportation

agencies (@511PAPhilly), News channels (@6abcBreaking), and other governmental agencies (e.g., @PGHtransit). We start with a dictionary with 50 "seed keywords" for the iterative data acquisition process. After 9 iterations, the acquisition process converges. As a result, we obtain a dictionary with 131 keywords (and some combinations) that are positively related to being a TI tweet, and 383 keywords that are negatively related. Using this dictionary, we acquired in all 10,542 and 11,658 tweets in Pittsburgh and Philadelphia region in Sep 2014, respectively. As part of this data acquisition process, we have also manually labeled all tweets into TI tweets and NTI tweets as the “ground truth”. There are 3,776 and 4,571 TI tweets in Pittsburgh and Philadelphia, respectively.

The convergence of the adaptive data acquisition process for Pittsburgh is shown in the Figure 3.5. Figure 3.5a shows that as iterations progress, the total number of new TI tweets acquired at each iteration drops quickly after 4 iterations. At the 9th iteration, the percentage of newly acquired TI tweets is less than 1% (Figure 3.5b), implying that adding more words to the dictionary and manually labeling newly acquired tweets is no longer cost-effective to obtain TI tweets. The adaptive data acquisition process for Philadelphia region follows a similar trend. The total number of new TI tweets converges after 9 iterations.



(a) The number of TI tweets

(b) The percentage of new TI tweets in each iteration

Figure 3.5: Convergence of the iterative data acquisition process

Geocoding on Manually Labeled TI Tweets

Since all tweets have been manually labeled, we know precisely whether they are TI tweets or not. We can therefore examine the effectiveness of our geocoding algorithms alone by geocoding those manually labeled TI tweets. This can eliminate the error brought by the classifier. We will discuss the accuracy of the classifier and geocoder combined in the next subsection.

Without applying the classifier, the adaptive data acquisition and geocoding yield the results shown in Table 3.6. For Pittsburgh, among the total 10,542 acquired tweets, 3,776 are TI tweets. 554 TI tweets directly or indirectly reported traffic incidents with meaningful time and location information. This result seems fairly impressive. In addition, tweets acquired from those 19 IUs in the Pittsburgh region account for merely 5.9% of all acquired TI tweets, but they contribute to 69.8% of those 554 TI tweets that are geocodable. This is not surprising since IUs' tweets tend to report traffic incidents with full details of the location. Most of them follow a clear linguistic structure, which allows accurate geocoding. For example, a typical tweet posted by an IU reads: "Turnpike Roadwork on Pennsylvania Turnpike I-476 northbound between Exit 31 - PA 63 and Exit 44 - PA 663 affecting the right lane", where the location information is clearly structured in this tweet.

As a comparison, a typical TI tweet posted by an individual reads: "@** – Daughter & Grandkids In Serious Car Crash http://dlvr.it/6tpycK @** @** all our prayers", with a picture of the accident attached (users' account is masked by **). This TI tweet contains vague descriptions about the accident, and it is hard to extract location information of this traffic accident. Though only 4.9% of TI tweets that are posted by individuals can be geocoded, they considerably complement the incident data, most of which are not necessarily reported by IUs. One example is "per traffic, 676 is not the blue route. Accident is at Vine St Expy and the Schkuykill Expy". By textual mining this tweet, we can identify the location of an accident at the intersection of Vine St Expy and the Schkuykill Expy. In Sep 2014, there are in all 156 and 175 instances of incidents

reported by individuals in Pittsburgh and Philadelphia, respectively. Note that the free version of REST APIs only retrieves a small portion of tweets randomly. The incidents information from individual tweets may be more extensive if the full set of tweets data are available.

	Pittsburgh	Philadelphia
TI tweets	3776	4571
NTI tweets	6766	7087
All tweets	10542	11658
IUs' tweets	621	554
Individual users' tweets	9921	11104
TI tweets of IUs'	595	518
TI tweets of individual users'	3181	4053
Geocodable TI tweets	554	419
Geocodable TI tweets of IUs'	381	244
Geocodable TI tweets of individual users'	156	175
The portion of TI tweets in IUs' tweets	95.8%	93.5%
The portion of TI tweets in individual tweets	31.1%	37.5%
The portion of geocodable tweets in IUs' TI tweets	64.0%	47.1%
The portion of geocodable tweets in individual TI tweets	4.9 %	4.3%
The portion of IUs' tweets in all acquired tweets	5.9%	4.8%
The portion of IUs' tweets in geocodable TI tweets	69.8 %	58.2%

Table 3.6: The result of data acquisition, manual labeling and geocoding

We plot the number of TI tweets and geocodable TI tweets by day of month for Pittsburgh in Figure 3.6. It can be seen that there is clearly a pattern on the weekly basis: both numbers peak on weekends. Weekdays generally report less incident information. The daily number of

geocodable TI tweets increases in an approximate portion to the daily number of acquired TI tweets, implying that a stable performance of the geocoders extracting location information from tweets.

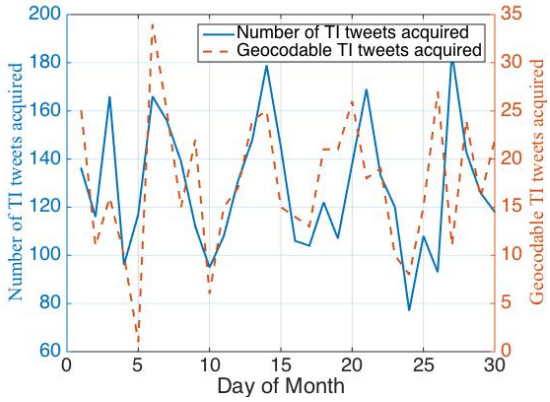


Figure 3.6: The number of TI tweets by day of month

Testing on the Geocoder and Classifiers

Next, a Semi-Naive-Bayes classifier is built based on the 131 words and combinations (namely features) that are positively related to being a TI tweet as a result of the data acquisition. The top ten features are, "*road-name*", "exit", "accident", "traffic", "roadwork", "lane", "PA", "mile", "cleared", and "post". For all Geo-TI tweets, the sLDA categorical classifier is applied to find the exact category of the traffic incident.

Note that all the acquired tweets have been manually labeled, and therefore we have the ground truth. In order to examine the effectiveness of the classifier and/or geocoding, the following tests are designed.

We conduct four tests. For the first three tests, we randomly select 5,000 tweets from the acquired tweets of both cities as the test data set, and the remaining 17,200 tweets as the training data set. The first test is to test the accuracy of the Semi-Naive-Bayes classifier. The Semi-Naive-Bayes classifier is trained with the training data set and applied to predict whether a tweet is TI for the test set. The error rate is estimated using a ten-fold cross-validation approach.

Second, we apply the geocoding methodology to the test data set directly without classifying them in prior. This examines how the geocoder works without a prior classification. Third, we follow our combined methodology where the test data set is Semi-Naive-Bayes classified and then processed by the geocoder. This allows us to investigate how effective the combined classifier and geocoder method is.

In the first test, the resulting average confusion matrix is shown in the Table 3.7. The overall classification accuracy of 90.5% as a result of the cross validation. TP, FN, FP, and TN denotes True Positive, False Negative, False Positive, and True Negative, respectively. An interesting observation is that FN is approximately the same as FP. This indicates that this classifier identifies tweets as TI falsely, almost as much as it fails to identify actual TI tweets. Notice that this classification is based a "neutral" classification threshold of 50%, namely identify the label Y^* that yields the greater of $P(Y = 1|X)$ and $P(Y = 0|X)$. $Y^* = 1$ indicates this tweet is a TI tweet, and 0 otherwise. In fact, this threshold can be adjusted in favor of users' experience. For example, if the user prefers to capture all the TI tweets as many as possible regardless of FP, one can set a small threshold $< 50\%$. A proper threshold value should be carefully chosen for the best results in practice.

The second test applies the geocoder to the same test data set. The ground truth of a tweet being "geocodable" is determined by the criteria that whether or not a tweet contains point location information. For example, "along I-376" or "near downtown" is not sufficiently accurate for geocoding and they are not geocodable. However, "I-376 Exit 74" can be accurately geocoded in a map. Since we do not apply the classifier before geocoding, it is possible that a tweet is geocodable but not a TI tweet. An example is "FINALLY found them at the intersection of Dauphin street and 12th street". The confusion matrix is shown in Table 3.8. Among 5.8% tweets that actually contain accurate point location information, the geocoder successfully identifies 4.8% of them, 82% accuracy.

In the third test, the testing data set is first classified into TI tweets and NTI tweets. The

		Predicted value		
		positive	**negative**	**total**
Actual value	**positive**	32.6 TP	5.0 FN	37.6
	negative	4.5 FP	57.9 TN	62.4
	total	37.1	62.9	

Table 3.7: Confusion matrix of the SNB classifier (all numbers are in percentages)

		Predictive value		
		positive	**negative**	**total**
Actual value	**positive**	4.8 TP	1.0 FN	5.8
	negative	0.2 FP	94.0 TN	94.2
	total	5.0	95.0	

Table 3.8: Confusion matrix of the geocoder (all numbers are in percentages)

Semi-Naive-Bayes classifier is trained based on the training data set of 17,200 tweets. Those tweets in the test set classified as TI tweets are then geocoded. A tweet is defined as "positive" if it is classified as a TI tweet and it is also geocodable. A confusion matrix is shown in Table 3.9.

Note that since the percentage is too small, we use the actual numbers of the tweets instead of percentages. It can be seen from Table 3.9 that there are in all 253 geocodable TI-tweets in the test data set. The combined classifier and geocoder can successfully identify 202 of them. Notice that these 202 geocodable TI-tweets is obtained by running the geocoding algorithm on the TI tweets classified by the Semi-Naive-Bayes classifier. In contrast, if we run the geocoding algorithm the TI tweets manually labeled, we can obtain 219 geocodable tweets. This implies that the classifier misses 17 $(219 - 202)$ useful tweets, while the other 34 $(253 - 219)$ useful tweets cannot be picked up due to the limitations of this geocoder.

		Predictive value		
		positive	**negative**	**total**
Actual value	**positive**	202 TP	51 FN	253
	negative	2 FP	4745 TN	4747
	total	204	4796	

Table 3.9: Confusion matrix of the combined classifier and geocoder

The fourth test is performed on the sLDA classifier. Following the same process as previous tests, we randomly select 873 of the entire 973 geocodable TI tweets crawled from Pittsburgh and Philadelphia as a training set and the remaining 100 tweets as a testing set. A sLDA classifier is trained by the training set and is used to perform classification on the testing set. Notice that for the input of one single tweet, the output of the sLDA classifier is a vector containing the probability of this tweet belonging to each of the five categories defined aforementioned. We

choose the category with the highest probability as the classified label of the tweet. The testing true positive rate is 51%. This implies that 51% of the tweet can be correctly classified by the sLDA classifier. Since the classification task is a multi-category classification task, the true positive rate is decent enough given the relatively small size of the training set.

Timeliness Analysis

To test the efficiency of the Twitter-based incident detector, we analyzed the computation time of the entire algorithms. The total computation time (T) needed from the actual occurrence of an incident to the incident plotted on the web interface (we are currently using a web interface to present the data) is composed of the following seven times indicated in Figure 3.7. Among those time intervals, T_1, T_2 and T_3 are not measurable in our system. We did some experiments for T_5 through T_7. $T_5 + T_6 + T_7$ is usually less than 5 seconds. The breakdown for T_4, T_5, T_6, and T_7 is recorded by our program precisely (Tabel3.10).

$$T = T_1 + T_2 + T_3 + T_4 + T_5 + T_6 + T_7 \tag{3.1}$$

It can be concluded that T_5, T_6, and T_7 are almost negligible comparing to T_4. The crawling frequency, set to 300 seconds by default, dominates the time from a TI tweet is posted to the corresponding incident is plotted in the web interface.

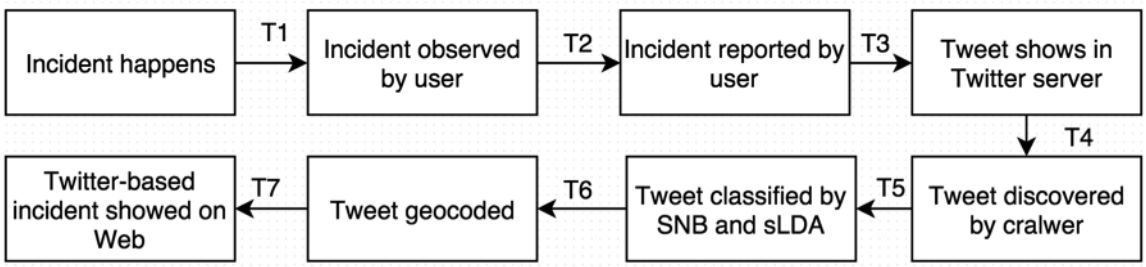


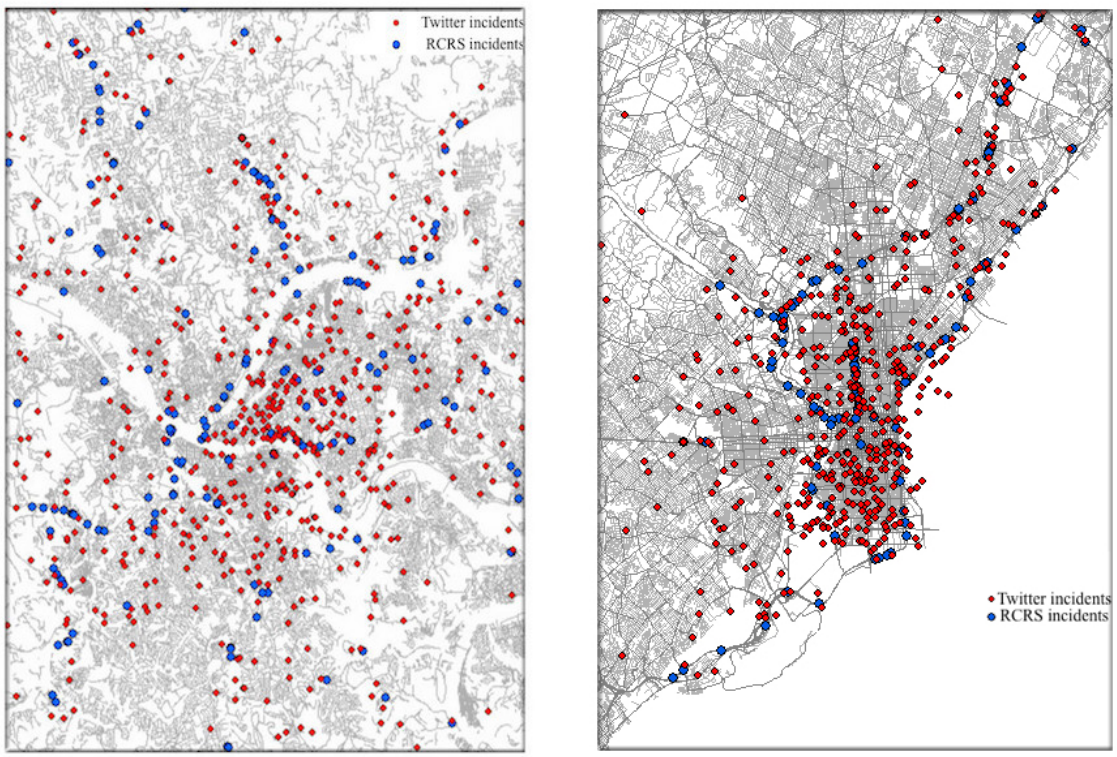
Figure 3.7: The timeline of an incident being reported by Twitter

Time period	Average time for one tweet
T_4	It is bounded by crawling frequency (300 seconds on current setting, or user defined crawling fr
T_5	3.1ms
T_6	87ms
T_7	0.9ms

Table 3.10: Breakdown of computation time

Validation with Ground Truth Traffic Incident Data

After geocoding the manually labeled TI tweets, we obtain 554 geocodable tweets for Pittsburgh and 419 for Philadelphia in Sep 2014. We can compare the results to a set of ground truth traffic incident data. For both Pittsburgh and Philadelphia, we obtained RCRS (Road Condition Report System) traffic incident data. RCRS incident data is what PennDOT is currently using for 511PA.org and Transportation Management Centers (TMC). The fields of RCRS data consist of the detailed description of the location, reported time, as well as the category of the incident. In Sep 2014, RCRS reported 217 incidents in Pittsburgh and 105 in Philadelphia. By representing each geocodable TI tweet with a stand-alone dot, scatter plots of incidents reported by RCRS and Twitter are shown in Figure 3.8a and Figure 3.8b for each region respectively.



(a) Pittsburgh

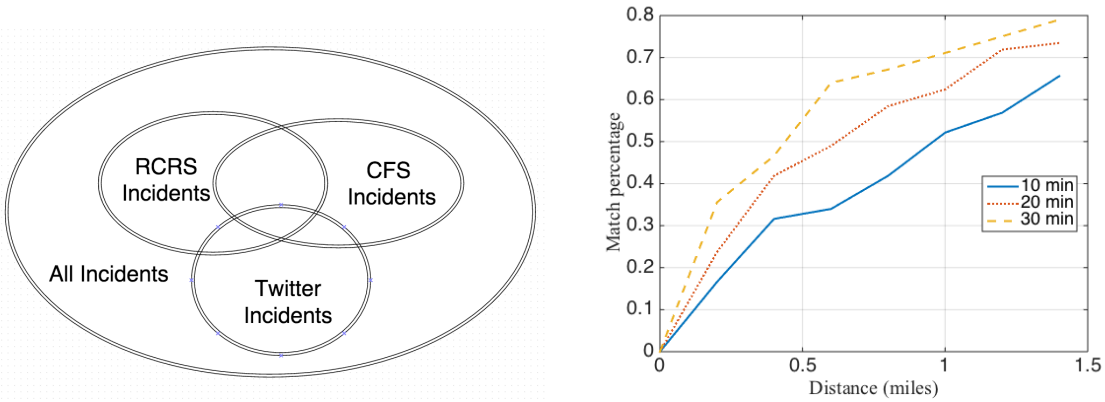
(b) Philadelphia

Figure 3.8: RCRS and Twitter incidents in Pittsburgh and Philadelphia

Specifically to Pittsburgh, we also obtained 911 Call For Service (CFS) data. The fields of CFS data mainly consist of the "time of dispatch", "address of the incident", "disposition", and "call type". Unlike the RCRS data which contains the exact time and location of the reported traffic incidents, the CFS data requires some preprocessing before usage. First, we choose the entries which have the "call type" relating to traffic incidents, such as "TRAFFIC-HIGH MECHANISM (AUTO-PEDESTRIAN)" and "TRAFFIC -WITH INJURIES". Second, we discard the entries which have the "Disposition" not likely relating to a valid traffic incident, such as "Unable to Locate" and "Gone on Arrival". And finally, we geo-code the "Address" field into the exact longitude and latitude using the aforementioned geocoder. After the preprocessing, we obtained the time 106 traffic incident from Pittsburgh CFS data (we call it "CFS incidents" for

simplification).

Moreover, it should be noticed that in order to compare Twitter incidents with CFS incidents and RCRS incidents properly without bias, the Twitter incidents that originated from official PennDOT accounts and Pittsburgh Police Bureau account are intentionally excluded. The reason is that we don't intend to re-discover what police or PennDOT has already known, but instead, to provide extra information to the authority. After excluding the Twitter incidents from PennDOT and Pittsburgh Police Bureau, we possess 698 Twitter incidents.



(a) Relationship among three incident data sources

(b) The overlap rate between RCRS+CFS incidents and Twitter incidents

Figure 3.9: Verification of Twitter incidents with RCRS+CFS incidents

The relationship among the three incident data sources (CFS, RCRS, and Twitter) is illustrated in Figure 3.9a. To compare the Twitter incidents with the current traffic incident database, we treat the union of RCRS incidents and CFS incidents as "ground truth" (we use "RCRS+CFS incidents" for simplification). First, we explore the percentage of RCRS+CFS incidents covered by Twitter incidents. To quantify this coverage, we try to measure how "near" the RCRS+CFS incidents with Twitter incident both temporally and spatially. In Figure **??**, we plot the percentage of RCRS+CFS incident matched by Twitter-based incidents within a specified temporal and spa-

Incident type	Number of incidents
Twitter incidents	973
RCRS incidents	322
CFS incidents	106
RCRS+CFS incidents	428
Twitter incidents excluding PennDOT and Pittsburgh Police tweets	698
Twitter incidents matched (1 mile, 30 minutes radius) by RCRS+CFS incidents	492
RCRS+CFS incidents matched (1 mile, 30 minutes radius) by Twitter incidents	304
Twitter "new" incidents	206

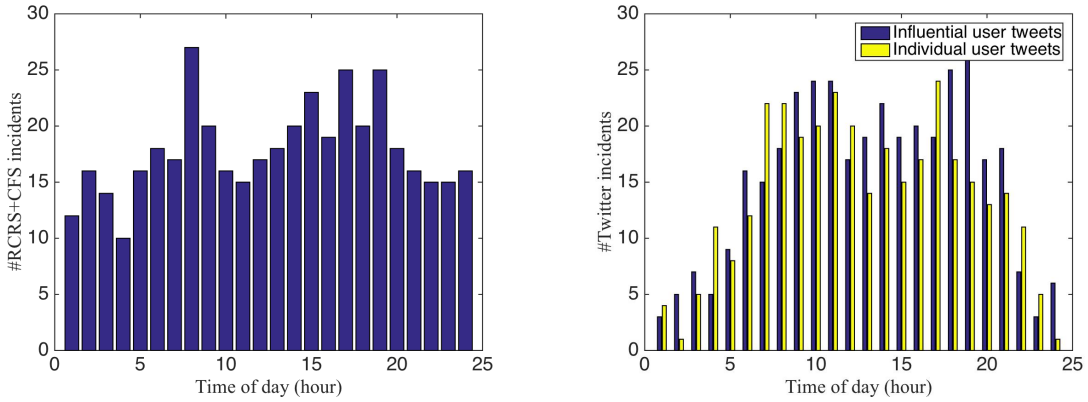
Table 3.11: Incident categories summary

cial radius of each RCRS+CFS incident. The temporal radius is 10, 20, and 30 minutes whereas the spatial radius is ranging from 0 to 1.4 miles. This figure illustrates that a greater temporal or spatial radius results in greater matched percentage. When allowing 30-minute reporting time discrepancy and 1-mile distance for the data to report the same incident, tweets can report 71% of entire RCRS+CFS incidents. This implies that the majority of the RCRS+CFS incidents are actually also reported by tweets.

As summarized in Table 3.11, in our experiment, among the total of 698 Twitter incidents, 492 of them are actually reporting the same incidents as 304 RCRS+CFS incidents, allowing 30-minute reporting time discrepancy and 1-mile distance. Therefore, it can be implied that on average for each one of the RCRS+CFS incidents, there are 1.6 Twitter incidents (we call this rate "coincidence rate"). Apart from these incidents reported by both RCRS+CFS and Twitter, there are 206 Twitter "new" incidents which are reported by neither RCRS nor CFS. Following the same coincidence rate of 1.6, these 206 Twitter "new" incidents could potentially detect 129 new traffic incidents. It is important to notice that these Twitter "new" incidents are manually labeled

as geocodable TI tweets, and therefore, we can safely conclude that the Twitter-based incident detector could potentially contribute 129 new traffic incident in addition to current RCRS and CFS traffic incident database.

Now we examine the spatial and temporal distributions of incidents reported by Twitter and RCRS+CFS. Figure 3.10 illustrates the number of incidents by RCRS+CFS and Twitter against time of day reported in both Pittsburgh and Philadelphia. The RCRS+CFS reported incidents is almost evenly distributed throughout the entire day. However, for Twitter incidents (excluding PennDOT and Pittsburgh Police tweets), the time-of-day distribution of both IUs' and individuals' Twitter incidents are completely different: most of the incidents were reported during the day time, especially morning and afternoon peak hours.



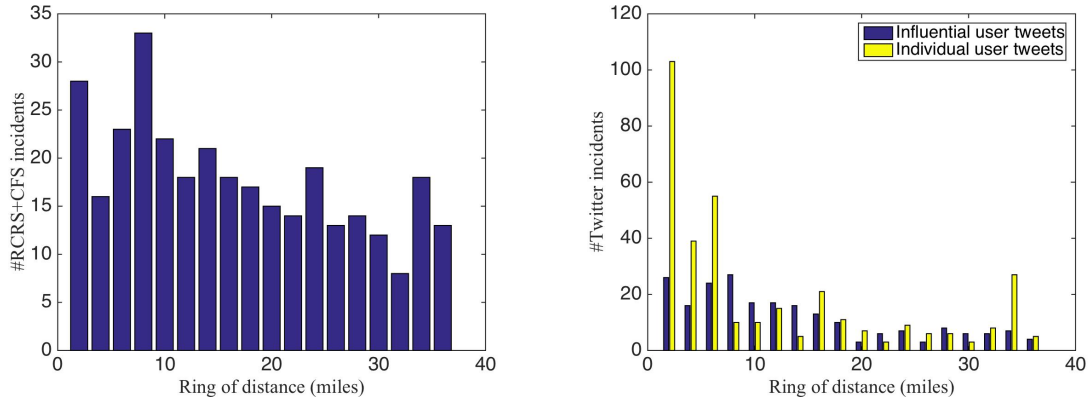
(a) RCRS+CFS incidents

(b) Twitter incidents

Figure 3.10: Time-of-day distribution of incidents reported by RCRS and Twitter

To compare the spatial distribution between Twitter incidents and RCRS+CFS incidents, we calculate the distance between the incidents to the center of the city (with the location 40.440731, -79.995751) for Pittsburgh area. We define "the incident is in the ring of the distance [a,b]" as "the distance between the incident and the center of the city lies between a and b". In Figure 3.11, we plot the number of RCRS+CFS incidents and Twitter incidents lie in the rings of distance in Pittsburgh area. It can be seen from Figure 3.11 that the individuals' tweets are more likely to

report incidents occurred near the city center, while IUs' incidents seem evenly distributed along the spatial dimension, similar to the RCRS+CFS data. This is not surprising since there are more active Twitter users near the city center than outside. Those figures in Philadelphia seem very similar to Pittsburgh.



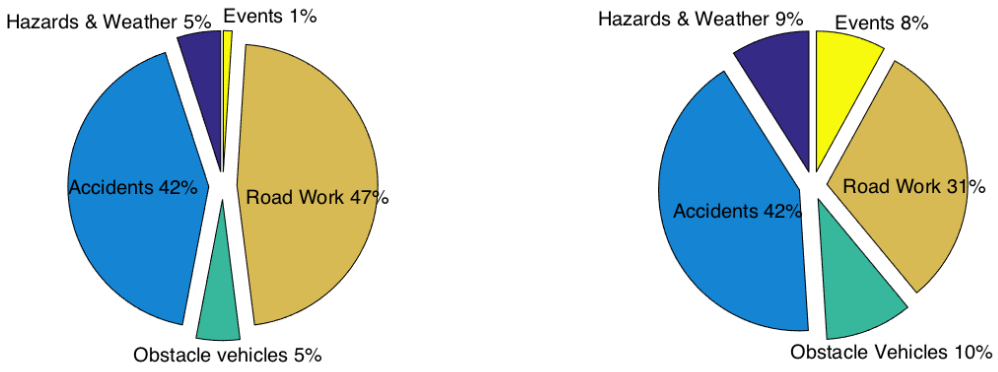
(a) RCRS (b) Twitter

Figure 3.11: Spatial distribution of incidents reported by RCRS and Twitter

Furthermore, we applied categorical classification on the geocodable TI tweets. It can be seen from Figure 3.12 that for Pittsburgh, for both RCRS and Twitter, the two most frequent mentioned categories are accidents and road work. Also, for Twitter, there are slightly larger portions of events and obstacle vehicle than RCRS.

Validation with Travel Time Data

In addition to comparing the Twitter incident with ground truth traffic incident data, in this section, we approach the validation problem from another angle - validating Twitter incident by inspecting the travel time. The data we use here is the RITIS travel time data. It is a dataset containing the travel speed, travel time in major roads in the city of Philadelphia and Pittsburgh with certain confidence levels. The primary assumption of our analysis is that, "if there is a traffic incident in a road, the travel time will substantially vary from the typical travel time, and vice



(a) RCRS incident categories

(b) Twitter incident categories

Figure 3.12: Pittsburgh RCRS and Twitter incidents categories

versa". By comparing the travel time near the location of the incident with the historical travel time at the same location and the same time-of-week, we are able to identify whether or not the travel time increase is statistically significant and thus infer whether there is an incident. In this section, we use statistical hypothesis test on: (1) all incidents reported by Twitter and; (2) each incident that are reported by Twitter.

Hypothesis Test on the Entire Set of Incidents

Suppose there are in total N Twitter incidents. The measured travel time of the road segment that is the closest to the occurrence time and location of the i-th Twitter incident is T_i. In particular, we define T_i as the average travel time from half an hour before the incident occurrence and half an hour after. Also, we retrieve all the travel times at the same location, the same time of day, and the same day of week over the previous eight weeks as H_i, which is a vector of eight elements. Notice that H_i is also one-hour average travel time. We standardize the travel time by:

$$T_i' = \frac{T_i - E(H_i)}{Std(H_i)} \tag{3.2}$$

and

$$H_i' = \frac{H_i - E(H_i)}{Std(H_i)} \tag{3.3}$$

Notice that H_i' is also a vector consisting historical travel times in the previous eight weeks, and E() is the operator of expectation and Std() is the operator of standard deviation. Theoretically, the distribution of the union of the travel times from different locations $H_1' \bigcup H_2' \bigcup ... \bigcup H_N'$ (we call it "the distribution of H_i' for simplicity") shows the "typical travel time". Notice that the process of the standardization makes the unions of H_i' for difference locations meaningful. While the distribution of H_i' describes the "**typical** travel time", the distribution of T_i implies the "**actual** travel time" at the time and location that the i-th Twitter incident occurs. By comparing the distribution of T_i' and H_i', we are able to show how statistically different between the typical travel time and actual travel time.

The result is shown in Figure 3.13. In Figure 3.13, it can be clearly seen that the distribution of the actual travel time at the time and location where Twitter reports an incident is significantly different from the distribution of the typical travel time at the same time and location. To quantify the difference, we performed a KolmogorovâĂŞSmirnov (K-S) hypothesis test under the null assumption that "Typical travel time and actual travel time have the same distribution".

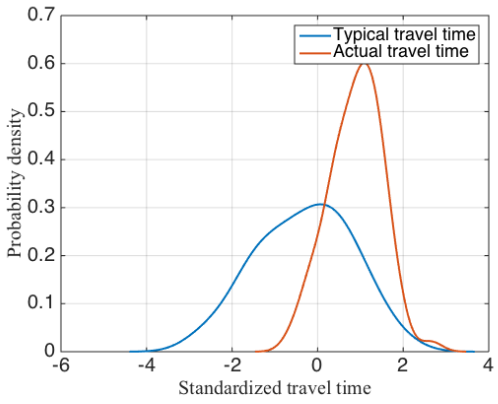


Figure 3.13: Comparison on typical travel time and actual travel tim

The resulting P-value of this K-S test is 7.9965e-15, which is significantly smaller than nor-

mal significance level 0.01, meaning we should reject the null hypothesis. Therefore, we can conclude that "there is significant evidence for two random variables: typical travel time, and actual travel time when Twitter indicates an incident, follow two different distributions". This observation also indicates that there is a "clustering effect" between the increase of travel time and the occurrence of a Twitter incident. Under our assumption aforementioned, "if there is a traffic incident in a road, the travel time will substantially vary from the typical travel time, and vice versa", the actual travel time is significantly higher than the typical travel time, which validates that those Twitter incidents are likely to be true.

Hypothesis Test on Individual Incident

The K-S hypothesis test above gives the overall validation that in general Twitter incidents are truly traffic incidents. For each individual Twitter-reported incident, we perform a specific hypothesis test with the null hypothesis "Typical travel time and actual travel time where Twitter reports an incident follow the same Gaussian distribution". The test, in general, is a Z-test. If the test statistic shows that we reject the null hypothesis, we can conclude that "there is significant evidence to show that when Twitter reports an incident, the traffic is likely to be abnormal due to the incident. To perform this Z-test, we prepare the data in the following way. First, similar to the previous section, we compute

$$T_i' = \frac{T_i - E(H_i)}{Std(H_i)} \tag{3.4}$$

and

$$H_i' = \frac{H_i - E(H_i)}{Std(H_i)} \tag{3.5}$$

Notice that in this test, H_i is all the travel times at the same location, same time of day, and same day of week as T_i in the previous eight weeks. As assumed in the null hypothesis, H_i follows a Gaussian distribution, and therefore, H_i' conform a standard Gaussian distribution with mean 0

and variance 1. The P-value of our statistical test is

$$P_i = P(Z \geq T_i') \tag{3.6}$$

where P_i is the P-value of i-th Twitter incident, and $P()$ is the operator of measuring probability. The Z-test is shown in Figure 3.14, where the area to the right of the black line T_i' and under the blue curve is the P-value of interest for the i-th incident detected by Twitter.

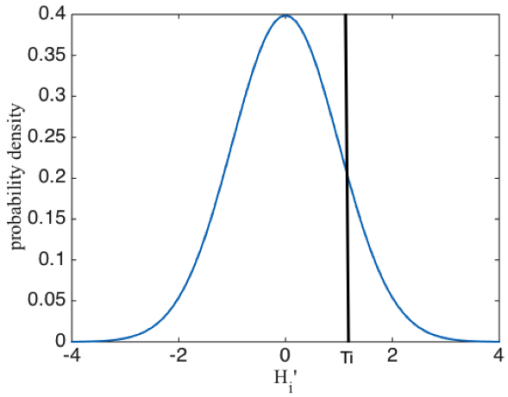


Figure 3.14: Standard Z test for individual sample

The final step of this hypothesis test is to define a threshold U for each incident i, where

$$\begin{cases} P_i \geq U & \text{Fail to reject the null hypothesis} \\ P_i < U & \text{Reject the null hypothesis} \end{cases}$$

Also, notice that "rejecting the null hypothesis" implies that the Twitter incident corresponding to T_i' is truly a traffic incident. Here instead of defining one single threshold U, we explore how U can influence the percentage of Twitter-reported incidents that reject the null hypothesis (namely being truly an incident). As we can see from Figure 3.15 that when the threshold of U is set to be around 0.7, almost all of the sample travel times T_i' will reject the null hypothesis, in other words, their corresponding Twitter-reported incidents are actual. On this one-sample statistical test, there are three points that worth noting:

1. Theoretically, the threshold of U guarantees that the Type I error of this statistical test rate is at most U, which is not the probability of the null hypothesis is true;

2. Even "Fail to reject the null hypothesis" doesn't mean to "the null hypothesis is more likely to be true";

3. The underlying assumption "if there is a traffic incident in a road, the travel time will substantially vary from the typical travel time, and vice versa" just offers a way to make the hypothesis possible, which is not necessarily always true. For example, in non-rush hour the road work on a single lane on a multi-lane freeway doesn't always lead to the increase of travel time.

Regardless of the limitation of the hypothesis tests, overall, by comparing the travel time at the same time and location when an incident reported by Twitter occurs to that of previous eight weeks, we conclude that statistically, those Twitter-based incidents are likely to be true.

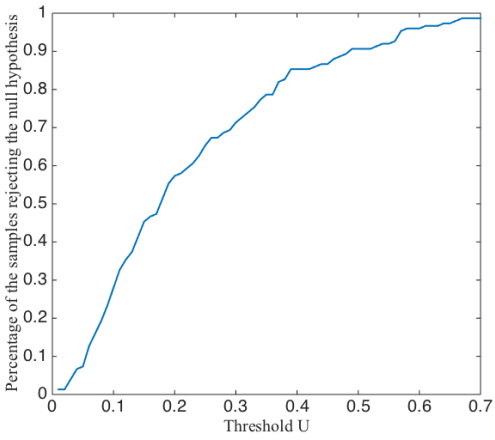


Figure 3.15: The influence of U on the rate of the samples rejecting the null hypothesis

Chapter 4

Combining Big Data in a Large-scale Traffic State Estimation

4.1 Problem Statement

In this chapter, we implement the aforementioned Hierarchical Bayesian Network, Expectation Maximization Kalman Filter, and Twitter-based incident detector in the Metropolitan Washington DC area. The research tasks that we aim to solve are:

1. Identify and formalize the data preprocessing procedure, including data extracting, transforming, and loading.
2. Modify and improve the aforementioned algorithms in the context of big data. In particular, develop and implement a distributable version of the aforementioned algorithms in a popular distributed computing framework, Hadoop.
3. Design and implement an efficient, and reusable software that can execute the distributed algorithm robustly.

4.2 Data Acquisition and Extract-transform-load (ETL)

In this section, travel speed data, volume data, road network data, and traffic incident data will be introduced and discussed in terms of data acquisition and extract-transform-load (ETL). In this research, ETL refers the general methodologies of transforming unstructured raw data into clean and formated data structures.

4.2.1 Road Network Data

The information about the road network, such as the location, road length, connectivity, and geography, are contained in a system called Geographic Information System (GIS) [53]. GIS is designed to efficiently store, manage, and analyze geographical data. One of the most popular formats of GIS file is called "shapefile" [17]. Shapefile was initially developed by Esri as an open-source data format and has been widely adopted throughout the industry, such as software companies and transportation agencies. For example, the shapefile of the entire Washington DC road network is shown in Figure 4.1.

In essence, the shapefile contains two categories of information: shapes and attributes. A shape, such as a polyline, contains the information about the exact longitude and latitude of all the endpoints and how the endpoints are connected to form a polyline. The other category, attribute, is a descriptive list containing the information about the entity which the shape stands for. For example, Figure 4.2 shows an example of a road segment (in yellow) in ShapeFile, which is represented by a line as the shape of the road segment, and a list (Figure 4.3) as the attribute of the road segment.

In this research, we use the shapefile from TomTom as our main source of road network information in Washington DC area. Besides the detailed shapes of the road network, TomTom shapefile provides extensive information about the attributes of the road shapes, such as the number of lanes, speed limit, length, road name, directions, and road class. The detailed specification

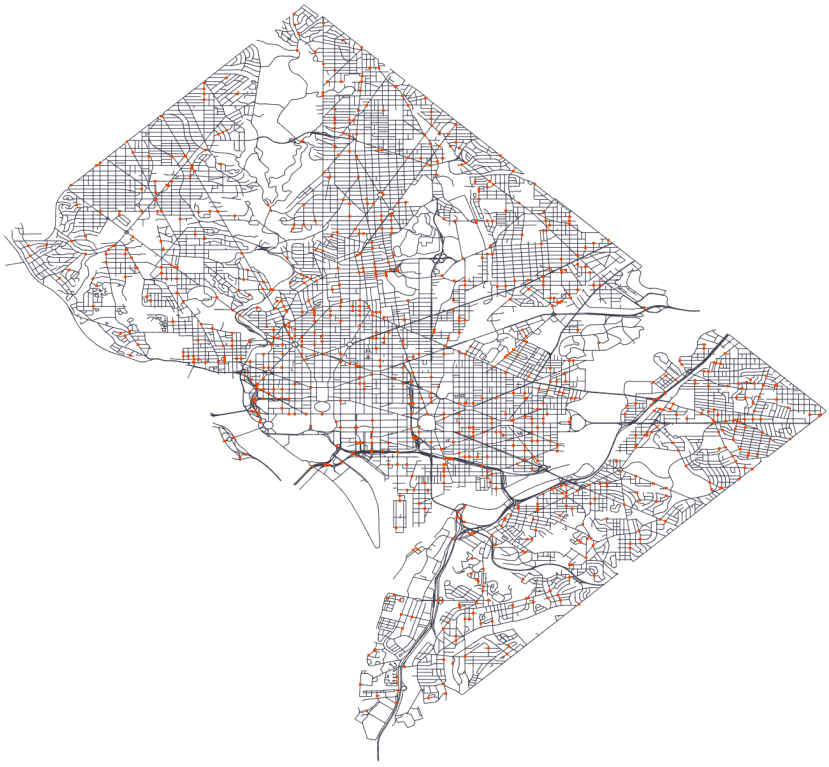


Figure 4.1: Washington DC Road shapefile

is illustrated in the document [65]. In total there are 49,768 shapes in the TomTom shapefile for Washington DC road network.

4.2.2 Travel Speed Data and Volume Data

In this research, we acquired two sources of travel speed data for the road network in Washington DC area. The first source is from Vehicle Probe Project (VPP-data), and the other is from fixed-location microwave detectors (Detector-data).

The VPP-data used a data collection techniques called probe vehicle techniques designed primarily for collecting data in real-time in modern Intelligent Transportation Systems (ITS). In contrast to speed detectors, vehicle probes, as a portion of traffic flow, directly measure travel speed using data from a combination of cell phones and automated vehicle location (AVL) services. The advantages of VPP-data comparing to Detector-data are as follows:

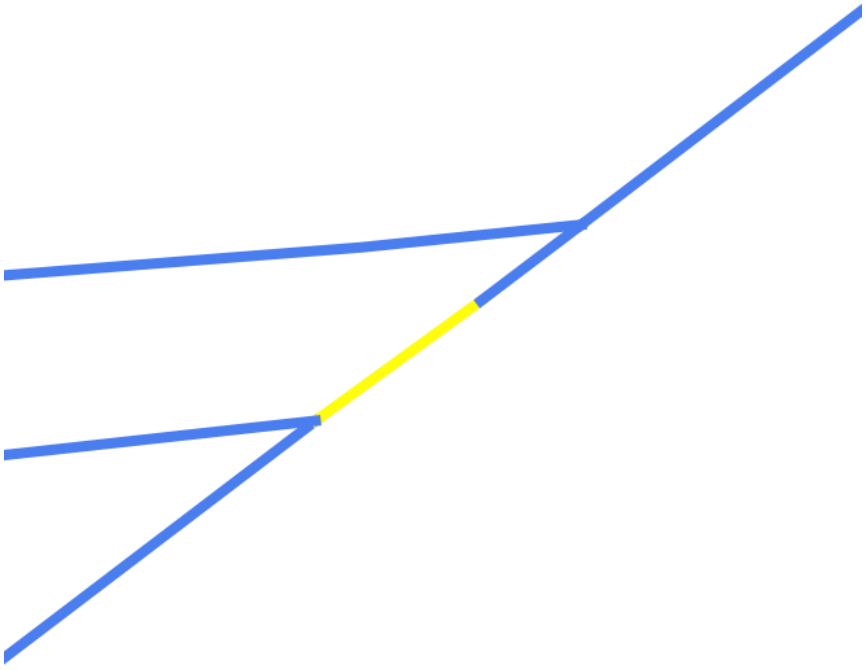


Figure 4.2: An example of the shape of a road segment (in yellow) in TomTom shapefile

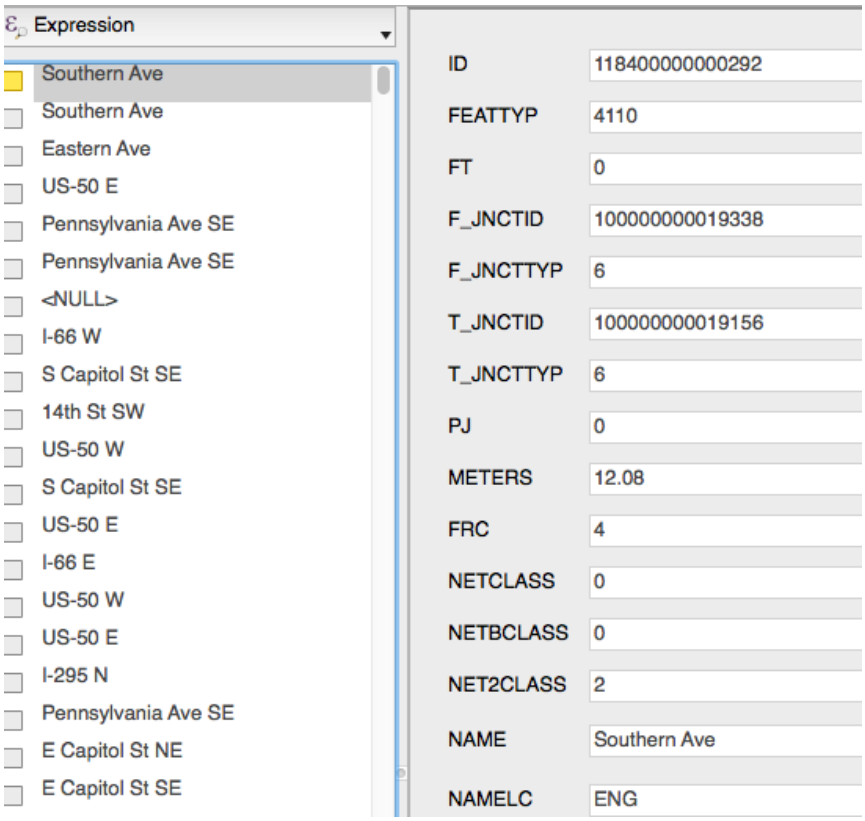


Figure 4.3: An example of the attribute of a road segment in TomTom shapefile

1. Low cost per unit of data: the probe vehicle sensors, especially from modern cell phones, are much less expensive than microwave/milliwave speed detectors; once the sensors are setup, there is no need for routine maintenance.
2. Automated data collection: travel speed can be detected as long as the probe vehicle is operating; and the collected data can be transmitted to a central database automatically via wireless technology.
3. Wide coverage: since probe vehicles, such as carrier vehicles, often travel through larger regions, the travel speed is available in these regions.

The disadvantage of VPP-data are as follows:

1. High initial cost: while the cost per unit of data is low, the initial cost is high due to the fact that various of resources are need to be put together, such as equipment installment, personnel training, and operate system developing.
2. Privacy issues: the privacy of the probe vehicles needs to be addressed.
3. Unable to provide continuous data at the same location: because probe vehicles are moving, the VPP-data can not guarantee a continuous observation of travel speed at the same location.
4. Not suitable for a small area: the high initial cost makes the application of VPP-data cost-ineffective for smaller areas.

As concluded above, VPP-data is especially applicable to Washington DC area due to its wide locational coverage and highly concentrated probe vehicles from active carriers vehicles, and therefore is used as the main data source for this research. In the year of 2015, there are 234,417,965 records of VPP-data (Figure 4.4). Also, there is extensive validation effort for the VPP-data: the max average absolute speed error 10 mph and maximum speed error bias 5 mph [**?**]. The data format (4.1) for VPP speed data has four fields: "TMC", "time", "speed", and "confidence". Specifically, the "TMC" denotes a segment of road, normally ranging from 0.1 mile to 1

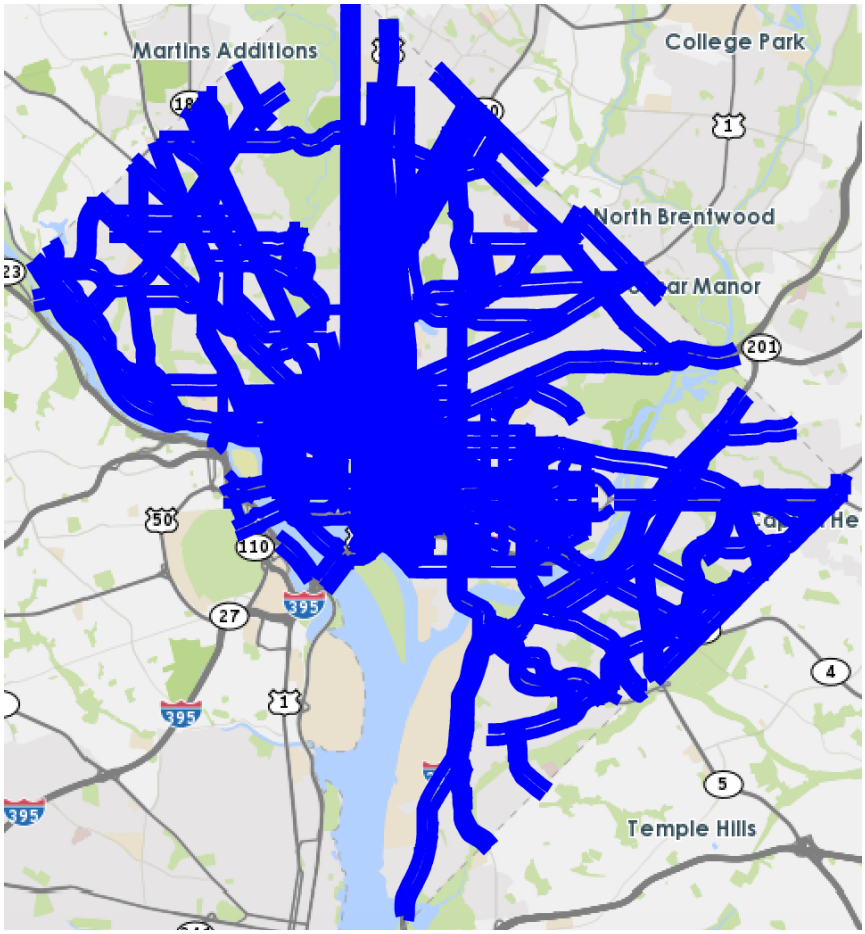


Figure 4.4: Roads with Probe Vehicle Speed Data in Washington DC Area

mile in length. One entry of VPP-data illustrates the travel speed at that segment of road where the TMC points to. TMC can be viewed as the ID of the shapes for the transportation network shapefile, and is actually defined this way (as shapefile) in National Performance Management Research Data Set (NPMRDS). Finally, the "confidence" in the VPP-data is defined as the measurement of the data quality, ranging from 10 to 30. The definition of these confidence levels are defined as follows:

30 Real time data. Any segment that has adequate data, at any time of day, will report real-time data.

20 Historic average: Between 4 am and 10 pm, any segment without sufficient real time data

will show the historical average for that segment during that day/time period (15-minute granularity).

10 Reference speed: From 10 pm to 4 am, any segment without sufficient real time data will show the reference speed for that segment. Any segment that does not have calculated historical averages will show the reference speed 24 hours a day if there is not sufficient real time data.

TMC	time	speed	confidence

Table 4.1: VPP-data format

The other source of travel speed data is from fixed-location microwave speed/volume detectors, and we call these Detector-data. These data are acquired by fixed-point sensors that mounted on the infrastructure to measure the speed, and sometimes volume, of traffic flows passing through a short segment of road. The distribution of the microwave speed-only detectors and speed-plus-volume detectors in Washington DC area is shown in Figure 4.5 and Figure 4.6. It can be seen that these detectors are mostly located on the entrance and exit of highways/freeways. Comparing to VPP-data, Detector-data has a much narrower coverage. However, the primary advantage of Detector-data over VPP-data is the Detector-data provides continuous and accurate observation of travel speed and volume at fixed-location road segments, which is especially suitable to estimate the boundary conditions discussed in the later sections.

The format of Detector-data has the following four fields: "zoneid", "time", "speed", and "volume". Specifically, for each "zone id", there is a specification providing the exact location of the zone (i.e., road segment), and the name of the road where the detector is located. There are 146 detectors available in Washington DC area, offering 33,921,179 records of speed and volume information in the year of 2015.

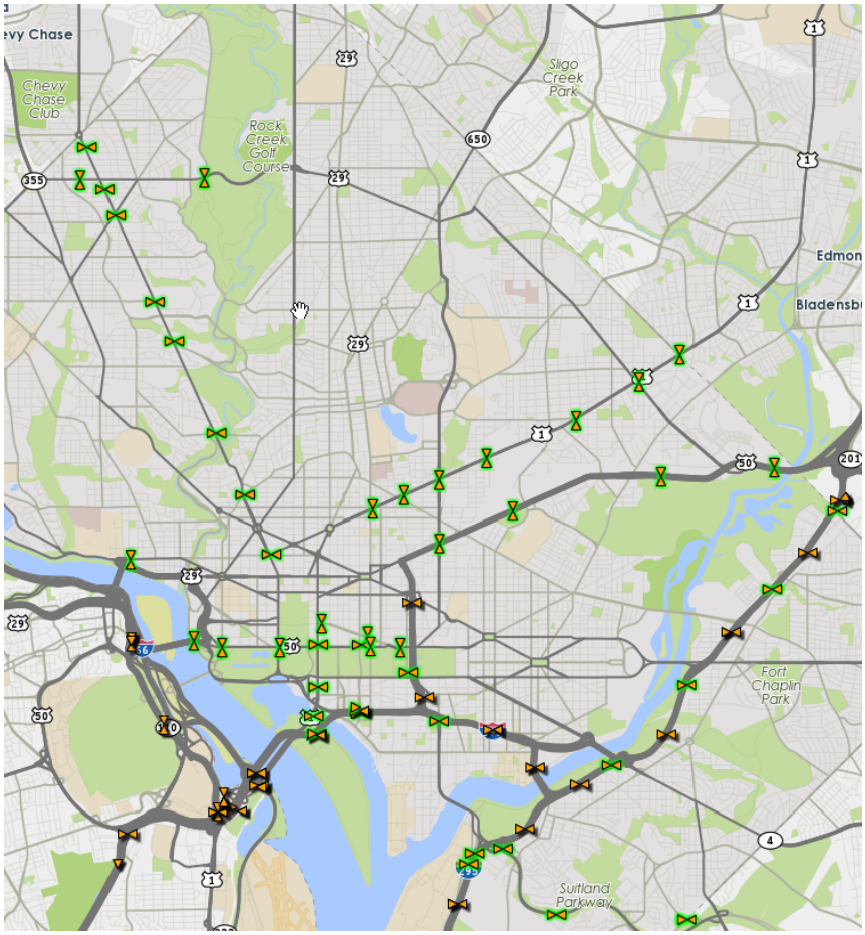


Figure 4.5: The distribution of microwave speed-only detectors (in green)

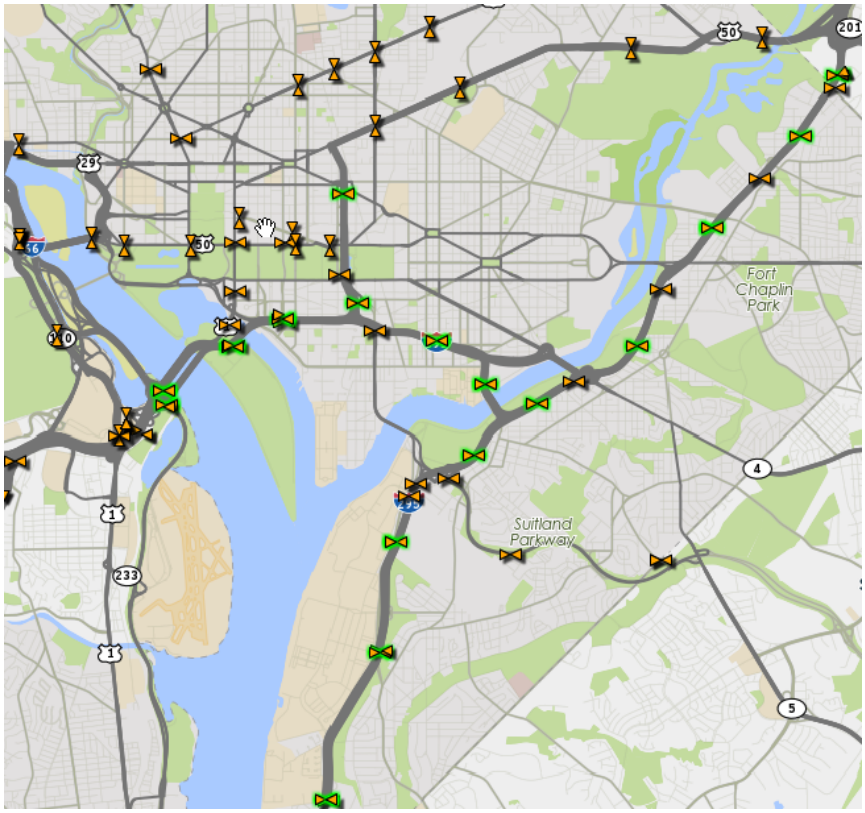


Figure 4.6: The distribution of microwave speed-plus-volume detectors (in green)

zoneid	time	speed	volume (optional)

Table 4.2: Detector-data format

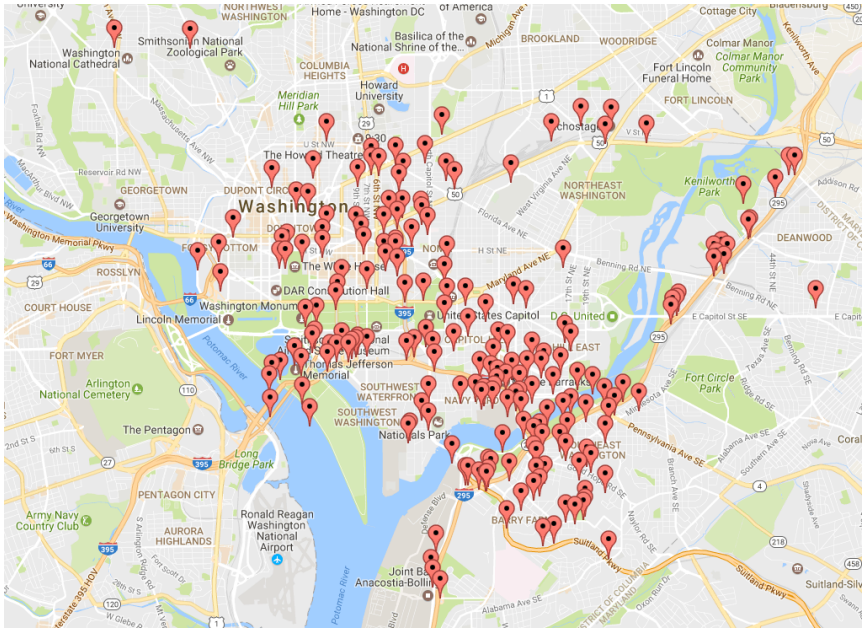


Figure 4.7: Traffic incidents from DDOT in Aug. 2015

4.2.3 Traffic Incident Data

In this research, we inspect two resources of traffic incident information, Washington DC Department of Transportation (DDOT) and Twitter. We denote the traffic incidents derived from DDOT as DDOT-incidents and the traffic incidents derived from Twitter as Twitter-incidents. As can be seen from Figure 4.7 and Figure 4.8, although there is certain overlap between DDOT-incidents and Twitter-incidents, the Twitter-incidents achieves better coverage in arterial area whereas DDOT-incidents has better coverage around highways, which is consistent with the conclusion from the previous chapter. The format (Table 4.3) of DDOT-incident offers not only the start-time, location, and category of the incidents, but also the duration from the begin of the incident to the end of the incident. This special information about the incident duration in DDOT-incident data will be further elaborated in the following sections.

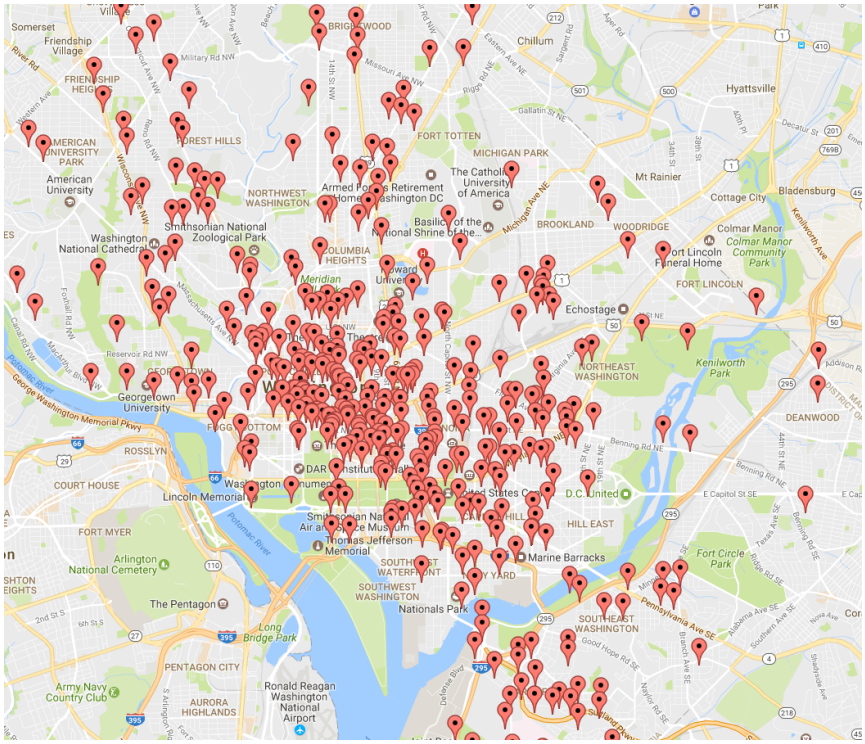


Figure 4.8: Traffic incidents from Twitter in Aug. 2015

incidentID	start-time	location	category	duration

Table 4.3: DDOT-incident format

Same as DDOT-incident, Twitter-incident also provides information about the start-time, location, and category of the incidents. However, the major difference between Twitter-incident and DDOT-incident is that, Twitter-incident doesn't always have information about the duration of the incident but has the details about the consequence of the incident, such as the number of lanes closed. For example, in the tweet shown in Figure 4.9, the natural language "two lanes blocked" could be further exploited not only inferring there is an occurrence of a traffic accident, but also leading to the consequence of this traffic accident, the blockage of two lanes. Therefore, we designed a series of regular expressions specifically extract the information about the number of lanes closed, in the field "closedlane", which takes an integer or "all", meaning the entire road

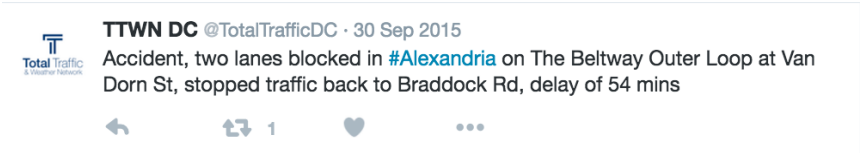


Figure 4.9: An example of tweet containing information about the consequence of the traffic incident

is closed. The format of Twitter-incident is shown in Table 4.4.

incidentID	start-time	location	category	duration	closedlane

Table 4.4: Twitter-incident format

4.2.4 Merge Data for Analysis

After acquiring and formalizing different data source, the next step is to merge all the data together to form the analytical basis for the large-scale analysis. First, the shapefile from TomTom is processed to shape effective computational components – links and nodes.

Shape1 Split the shapes in the TomTom DC major roads shapefile into uni-directional traffic flow links.

(a) Identify the shapes with bi-directional traffic flows.

(b) Create new shapes with the reverse of the shapes with bi-directional traffic flows in terms of the sequence of polyline endpoints - these new shapes have the "number of lanes" in the half of "the number of lanes" of the original shapes. Allow non-integer "number of lanes".

(c) Change the "number of lanes" attribute of the original shapes into its half.

Shape2 Select the shapes in Shape1 [if it overlaps with the 0.05 mile buffer radius of the TMC lines] OR [IF (1 - Levenshtein-distance(TMC road name, Shape1 road name)/max(Strlen(TMC

road name), strlen(TMC road name)))>0.75] OR [||geocode(TMC road name)-geocode(Shape1 road name)||<0.05 mile].

Shape3 Manually identify and simplify traffic circles/roundabouts into intersections.

Shape4 Integrate the shapes in Shape3 IF (they are within the same TMC) UNLESS (meet intersections).

Shape5 Generate connection table to specify the next connected traffic link for every integrated shape in Shape4.

1. Generate the initial parameters of the EM-EKF links and nodes from the attribute table of Shape5 (Table 4.5 and Table 4.6).

In the steps above, Levenshtein-distance(a,b) denotes the Levenshtein-distance [23] from string a to string b, and ||A-B|| denotes the Euclidean-distance from location A to location B. Geocode(a) denotes extracting the locational information in string a. Strlen(a) denotes the length of string a. Also notice that the criteria "(1 - Levenshtein-distance(TMC road name, Shape1 road name)/max(Strlen(TMC road name), strlen(TMC road name)))>0.75" means the TMC road name is similar to the Shape1 road name, and "||geocode(TMC road name)-geocode(Shape1 road name)||<0.05 mile" means the geocoded location of TMC road name is near the geocoded location of Shape1 road name.

linkID	shape (sequential list of points)	jam density J, free flow speed F_f, shock wave speed w	previous node	next node

Table 4.5: Data structure of links

nodeID	shape (point)	transmission proportion matrix	incoming links	outgoing links

Table 4.6: Data structure of nodes

Second, speed and volume information from VPP-data and Detector-data are merged and integrated into the computational components of links as nodes. The process is as follows.

1. Eliminate outliers and false data points from VPP-data and Detector-data.
2. Map the TMC in VPP-data to the ID of the links using a matching table provided by TomTom, forming Table 4.7.
3. Join the year-long VPP-data into the data structure of links (Table 4.8).
4. For every link at every time, take the weighted average of the probe vehicle speed from three sources, TomTom, INRIX, and HERE. The weight is proportional to the confidence level.
5. Find the nearest link from every detector to the nearest link. Manually check the direction of the link. Generate the matching table (Table 4.9).
6. Join the Detector-data to Table 4.8 using the matching table (Table 4.9).
7. For every link at every time, take the weighted average of the VPP-data speed Detector-data speed. The weight is proportional to the confidence level.

ID	shape (sequential list of points)	jam density J, free flow speed F_f, shock wave speed w	previous node	next node	**TMC**

Table 4.7: Data structure of links with TMC

ID	shape (sequential list of points)	jam density J, free flow speed F_f, shock wave speed w	previous node	next node	VPP-data

Table 4.8: Data structure of links with VPP-data

detectorID	linkID

Table 4.9: Matching table from detector to link

Third, traffic incident data are integrated together to make an informed traffic state estimation. The processing steps are listed below. For incident integration, we assume the traffic incidents that happened within 30 minutes and within 0.5-mile distance are the same traffic incident.

1. Join the traffic incidents in DDOT-data (Table 4.3) and Twitter-data (Table 4.4) together IF they are within 30 minutes AND within 0.5 mile distance. We call this newly joined table "DDOT-Twitter-data". In the year of 2015, we discovered 10,494 traffic incidents, with 702 incidents having certain lane reduction information, 6,205 incidents having duration information, and 551 incident having both lane reduction information and duration information.
2. Map the DDOT-Twitter-data onto the data structure of the links as in Table 4.5 IF [(1 - Levenshtein-distance(link shapefile road name, DDOT-Twitter-data's location.name) / max(Strlen(link shapefile road name), strlen(DDOT-Twitter-data's location.name)))>0.75] OR [||geocode(TMC road name)-geocode(Shape1 road name)||<0.05 mile]. This forms a matching table as in Table 4.10.
3. Using Table 4.10 to join the traffic incidents in DDOT-Twitter-data onto the data structure of traffic links.

Similar as the matching from TMC to shapefile, the criteria "(1 - Levenshtein-distance(link shapefile road name, DDOT-Twitter-data's location.name)/max(Strlen(link shapefile road name), strlen(DDOT-Twitter-data's location.name)))>0.75" means linguistically the link shapefile road name is similar to DDOT-Twitter-data's location.name.

linkID	incidentID

Table 4.10: Matching table from incidents to link

Following the procedures above, we formed a complete database to perform EM-EKF in the context of Link Queue Model. The data structure of the links and nodes are summarized in Table 4.12 and Table 4.13. It is worth noting that in the data structure of links, the field "nodeID", and "shape" are time-invariant. At any time of year t, the model parameters jam density J, free flow speed F_f, shock wave speed w are learned based on the data with a limited range of motion,

and are given initial values based on the physical attributes of the link. The variation of incident factor μ is more complicated as in Table 4.11.

Situation	μ variation
No traffic incident in link s at time t	$\mu(s,t) = 1$
There is traffic incident in link s at time t, with known lane closure and duration Δ	$\mu(s, t : t + \Delta)$ =(opened lanes)/(overall lanes)
There is traffic incident in link s at time t, with unknown lane closure but known duration Δ	$\mu(s, t : t + \Delta) = \hat{\mu}$ time variant learned from data
There is traffic incident in link s at time t, with unknown lane closure and unknown duration Δ	$\mu(s, t : t + \delta) = \max(\hat{\mu}, \frac{\delta}{12h}), \delta \leq 12h$ $\hat{\mu}$ is time variant learned from data

Table 4.11: The variance of μ under different situations

linkID	shape (sequential list of points)	jam density J, free flow speed F_f, shock wave speed w, incident factor μ	previous node	next node	VPP-data	DDOT-Twitter-data	traffic states (d,v,q)

Table 4.12: Summary of the data structure of the links

The summary of the is shown in Table 4.13. Except for the transmission proportion matrix, every field is time invariant. The transmission proportion matrix is assumed to be weekly periodical over the none-holidays and is flexible over the holidays.

nodeID	shape (point)	transmission proportion matrix	incoming links	outgoing links

Table 4.13: Summary of the data structure of the nodes

4.3 Multi-level Parallel Computing

The goal of computing is given the full data structure of links and nodes, including speed and incident observations, to estimate the time-variant model parameters, as well as unobserved traffic states. Since the entire space of unknown random variables are extremely large, it is almost

impossible to infer all random variables in the entire network in one run. Therefore, we utilized the principle of divide-and-conquer by implementing the EM-EKF on many smaller networks, and then combine them together to form a global estimation.

First, the entire spatial-temporal space is segregated into different regions. The temporal segregation is all-natural – the entire year is segregated by 12 months, and month is the bottom layer in the temporal segregation. The spatial segregation needs to take more considerations. The design of spatial segregation is a three-layer hierarchy (Figure 4.10). First, the entire Washington DC area is manually separated into four main computational regions, with an aim to incorporate as much Detector-data in the boundaries of the regions to provide continuous observations of boundary conditions. This segregation of the entire Washington DC area (Layer 1) into four regions (Layer 2) is shown in Figure 4.11. To segregate regions in Layer 2 into regions in Layer 3, we evenly slice the region into lattice by longitude and latitude.

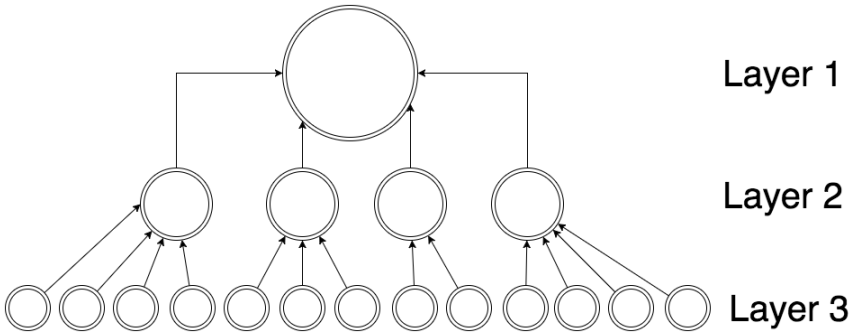


Figure 4.10: The design of spatial segregation for distributable EM-EKF

The computation is performed from the smaller regions, shorter durations in lower layers aggregately to larger regions and longer durations in upper layers. As illustrated in Figure 4.12, the x-th region in Layer y is called "Ry-x". In this example, R3-1 and R3-2 are neighbors, while R3-2 and R3-3 are neighbors. First EM-EKF is performed on R3-1 alone for every month. Then the EM-EKF is performed on R3-1 alone for month 1-3, month 3-6, month 7-9, and month 10-12, based on the initial conditions of the results in the every-month-run. Then we run EM-EKF

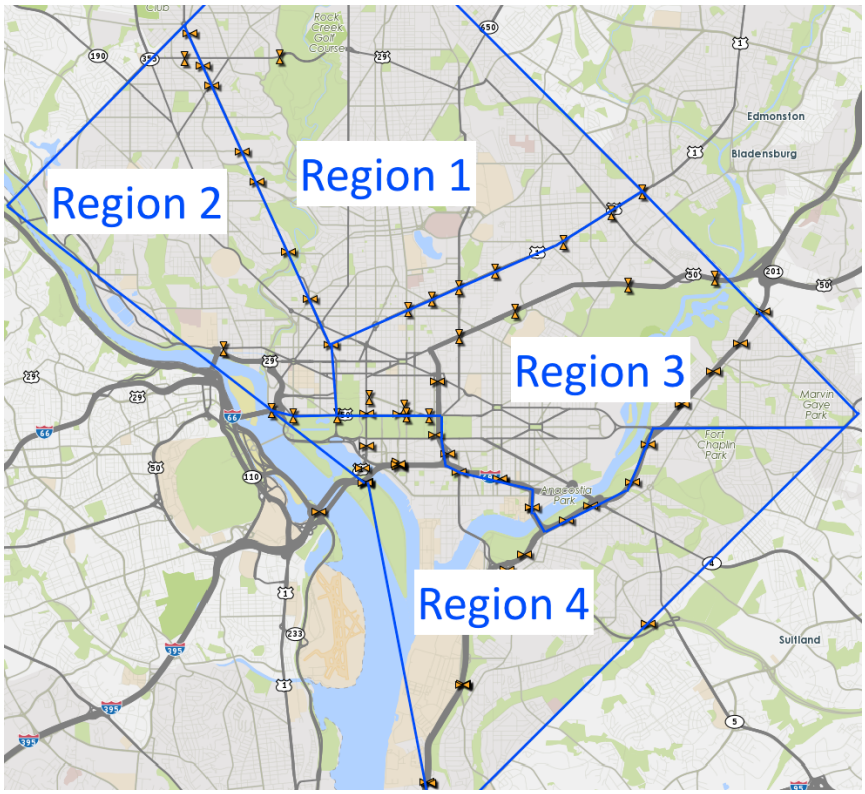


Figure 4.11: Layer 1 zones

for month 1-6 and month 7-12, and finally for month 1-12. This is called "temporal-cycle 3".

After each one Layer 3 region finishes the "temporal-cycle 3", we start the "spatial-cycle 3-2" (see Figure 4.12), meaning we run EM-EKF on R3-1 and R3-2 together to form R2-1&2. Similarly, we run EM-EKF on R3-2 and R3-3 together, to form R2-2&3. And finally, we run EM-EKF on R2-1&2 and R2-2&3 together, to form R2-1. Similarly, all Layer 2 regions are combined together to generate the result on the entire Washington DC area (Figure 4.13).

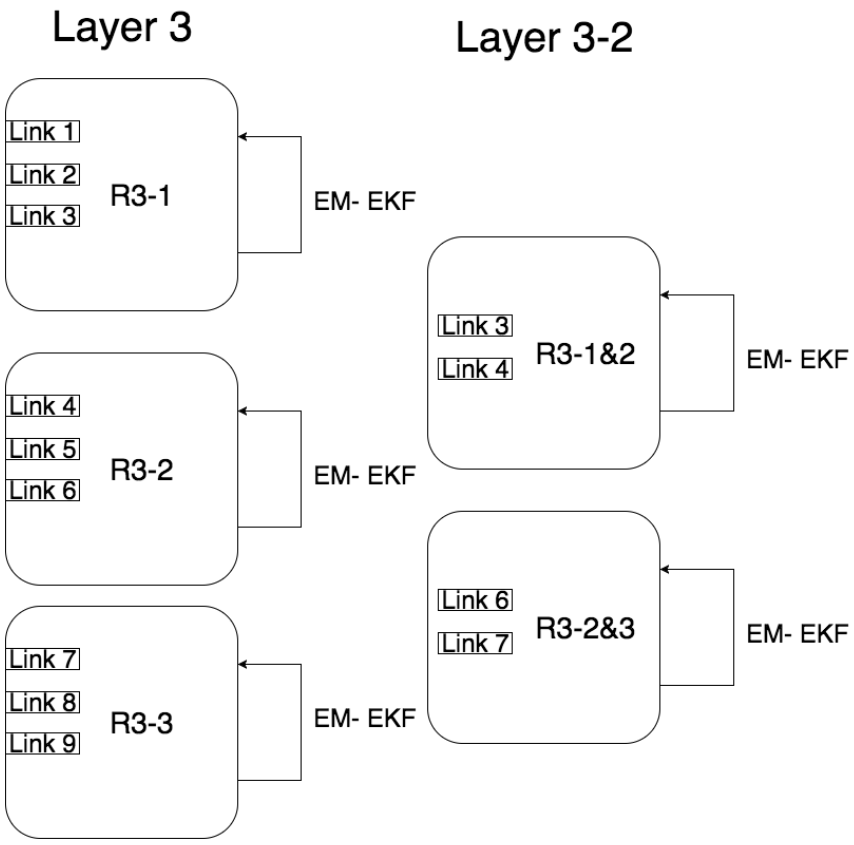


Figure 4.12: Distributable EM-EKF: layer 3 and layer 2

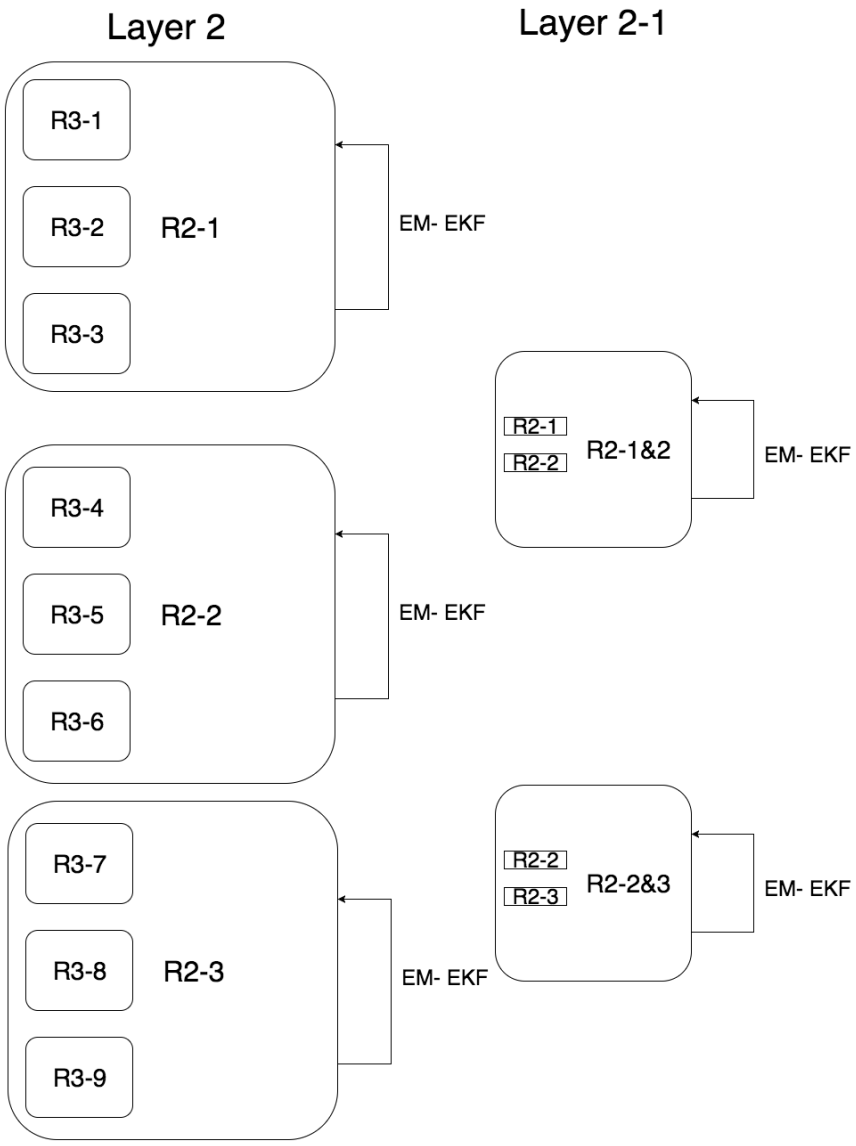


Figure 4.13: Distributable EM-EKF: layer 2 and layer 1

As illustrated above, the computation on both "temporal-cycle 3" and "spatial-cycle 3-2" can be done in parallel. Another aspect of the parallelism is that instead of computing only one instance of "temporal-cycle 3", we emit many "seeds", i.e., randomized initial conditions for the model parameters to run many instances of "temporal-cycle 3". At the end of each "cycle", we choose the top 50% instances as the "survivors" to proceed to the next level of computing. The reason of using this randomized parallel instances is that in essence, this is an importance

sampling [45] algorithm to choose the most promising proposals while having the possibility of not converging to local optimum. Moreover, not all instances may converge fast enough and therefore, having more seeds could enlarge the possibility of achieving a converged EM-EKF run on Layer 1. Moreover, the computational cost is not too high since each one instance is relatively easy to compute and the computation can be done in parallel. Finally, it is worth noting that, our core algorithms, the Extended Kalman Filter in E-step [57] and the Stochastic Gradient Descent and Linear Programming in M-step ([78]) can be done in a MapReduce framework using Spark.

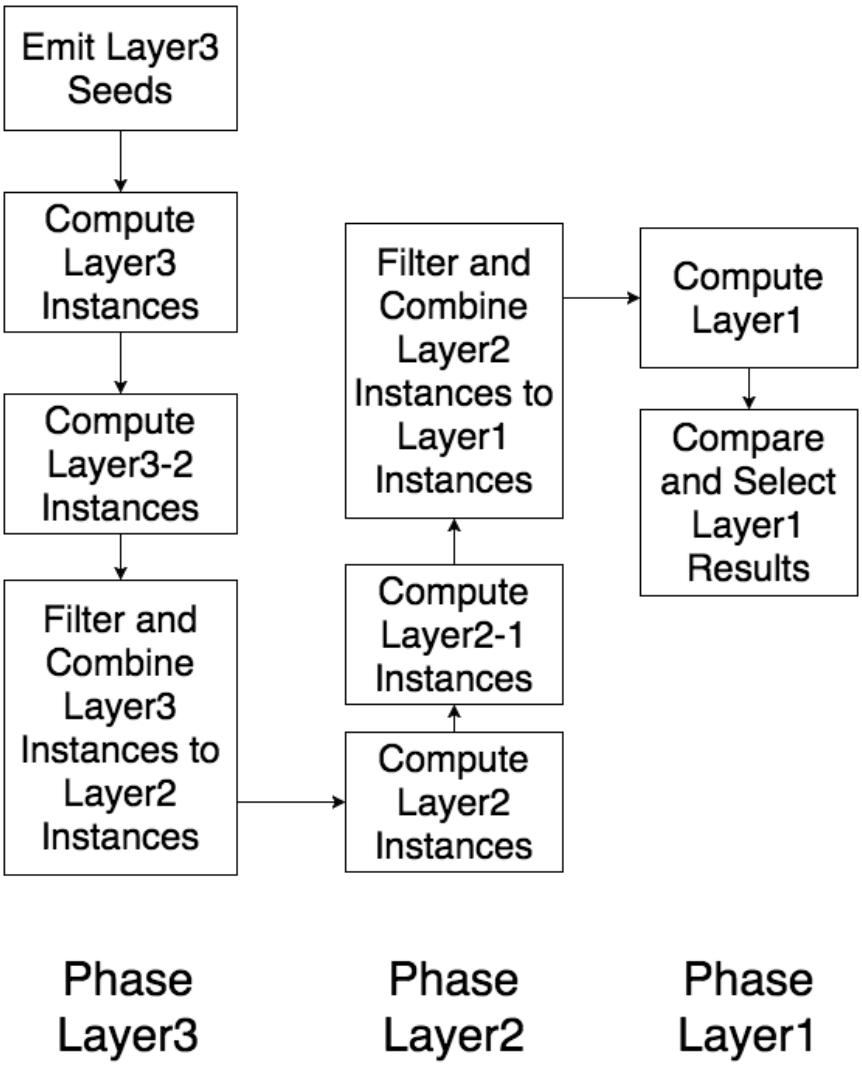


Figure 4.14: Distributable EM-EKF compute sequence

4.4 Implementation and Results

The proposed algorithm is tested and developed on CMU Hadoop cluster with 12 Intel Xeon cores and 32G memory and deployed on Amazon Web Service Elastic MapReduce (AWS-EMR) m4.16xlarge cluster with 64 Intel Xeon E5-2686 cores and 256G memory. The deployment took 4 days, including data Extract-Transform-Load (ETL) and computing.

In this research, we explored the proposed algorithm and computing framework on two situations:

1. Spatial-temporal estimation: to estimate the traffic states in road segments when there are no observations.
2. Incident adaptation: to estimate the traffic states in road segments when there are no observations but with incident information.

After the data processing by attaching speed/volume to the data structure of analytical road links, we separate the data in the chunks of days and locations. For example, if we have speed/volume observations in A days and B locations, we have A*B such chunks. Then we randomly pick 10% such chunks out of the entire training process as an independent testing set. The remaining 90% chunks act as training set and is then feed into the proposed parallel EM-EKF computing framework. Due to the limited time and the high deployment cost, ten-fold cross-validation is not an option.

Overall, the mean absolute error rate for the highway is 10.88% and for arterial is 23.00%. We first examine the performance of EM-EKF under full incident information on both highway and arterial road in Figure 4.15 - 4.17. In Figure 4.15, the spatial-temporal estimation result in from EM-EKF (in green) is compared with traditional temporal smoothing method (in red), which takes the historical average of the same day of week in the same location. The absolute mean error for EM-EKF is 2.37 mph and the absolute mean error for temporal smoothing is 2.79 mph. The EM-EKF provides 15% less mean absolute error than the traditional temporal

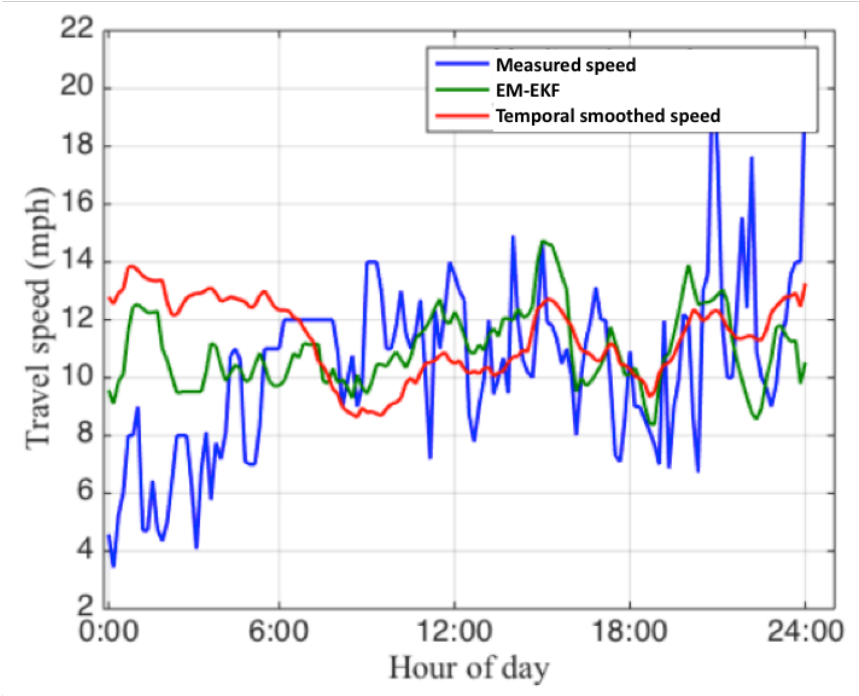


Figure 4.15: U ST NW between 14th ST and 16th ST (arterial): EM-EKF (with incident time and duration) vs. temporal smoothed estimation on Feb. 2, 2015

smoothing method. Additionally, the measured speed (in blue) has a decrease from 0:00-6:00, possibly due to a road work or abnormal traffic flow, although there is no traffic incident detected. It can be seen that comparing to temporal smoothed speed, EM-EKF better captured that decrease in speed during 0:00-6:00.

Similarly in Figure 4.16, we also compare the EM-EKF with the result from spatial-smoothing, which takes the average of the travel speed from nearby links with the same direction. The conclusion also remains similar: EM-EKF provides marginally superior results quantified by 25% less mean absolute error.

The increased performance of EM-EKF on highway traffic state estimation, with available incident data, is also significant. Figure 4.17 and Figure 4.18 illustrates the estimation result of EM-EFK comparing to measured speed, as well as spatial and temporal smoothed speed. One noteworthy difference between this estimation and the previous estimation on 11th ST is that,

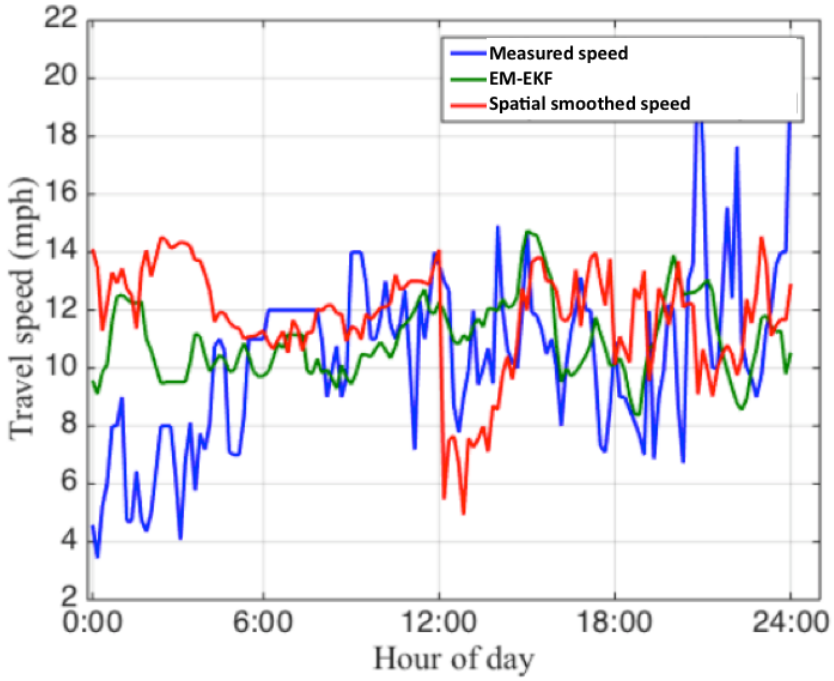


Figure 4.16: U ST NW between 14th ST and 16th ST (arterial): EM-EKF (with incident time and duration) vs. spatial smoothed estimation on Feb. 2, 2015

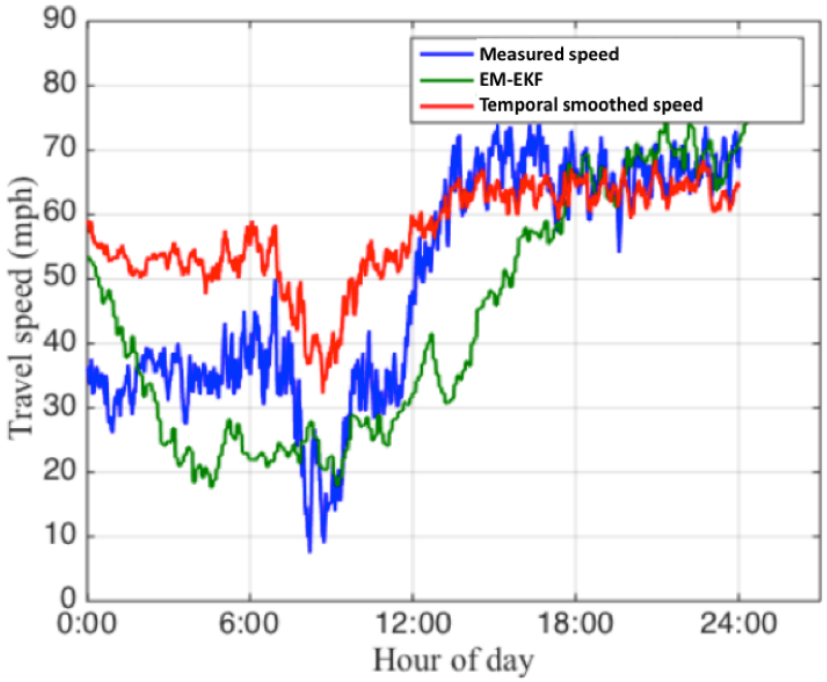


Figure 4.17: I-695 near 11th ST (freeway) : EM-EKF vs. temporal smoothed estimation on Feb. 2, 2015 (with incident)

there is a traffic incident detected by Twitter signaling there is a one-lane reduction affecting all northbound traffic flows from 11:38pm, Feb.1 2015, to noon, Feb. 2, 2015, when the incident is cleared. This incident significantly reduced the travel speed on this road segment by almost a half. However, both temporal smoothed method (Figure 4.17) and spatial smoothed method (Figure 4.18) couldn't reflect the dramatic and sudden decrease and later increase in travel speed. In comparison, the EM-EKF method, equipped with the information about the decrease in the number of lanes from Twitter incident, can quickly adjust the model to adapt to this incident and make a much better prediction. For this case, the EM-EKF achieves 16% less mean absolute error than spatial smoothed method and 7% less mean absolute error than temporal smoothed method.

Also for the traffic state estimation on the freeways, but without significant disruptions such as traffic incidents, the results are shown in Figure 4.19 and Figure 4.20. It can be seen that

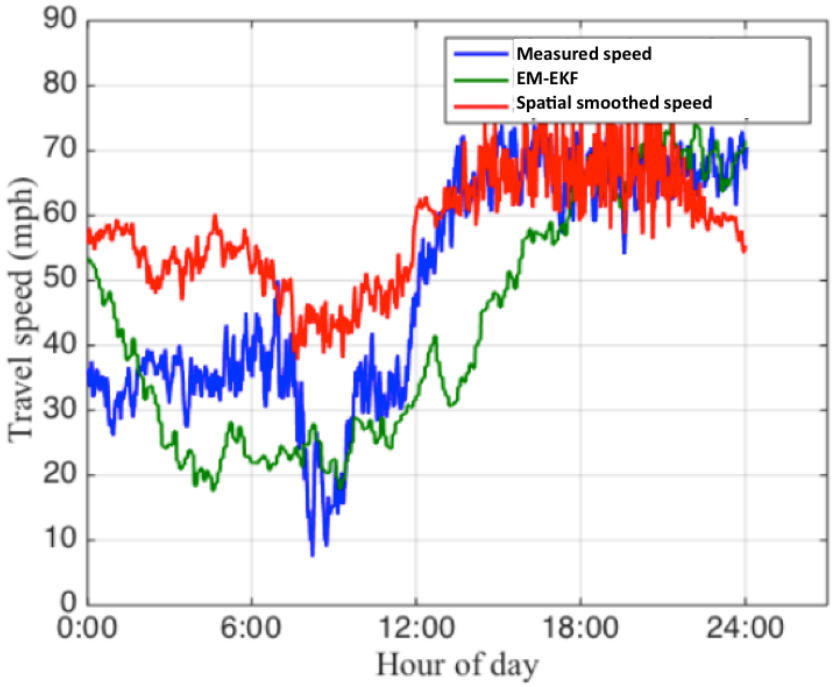


Figure 4.18: I-695 near 11th ST (freeway): EM-EKF vs. spatial smoothed estimation on Feb. 2, 2015 (with incident)

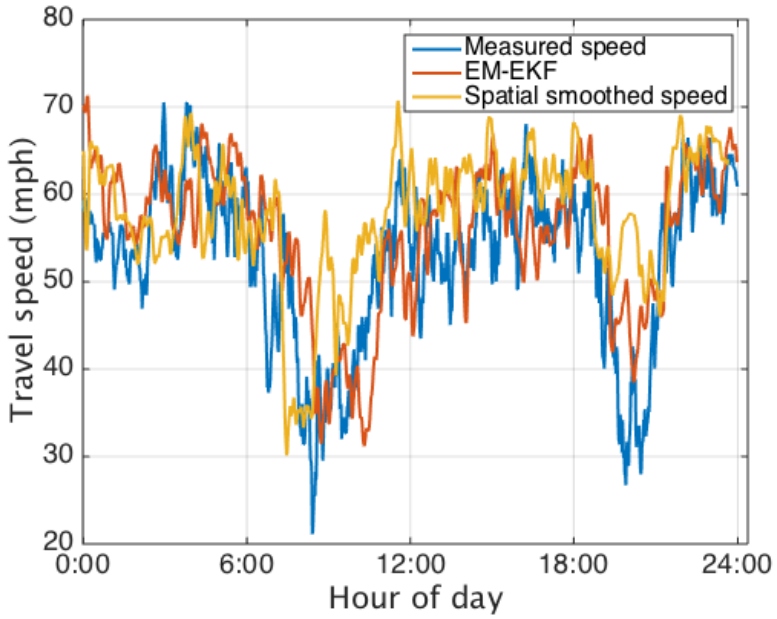


Figure 4.19: I-695 near 11th ST (freeway) : EM-EKF vs. temporal smoothed estimation on July. 23, 2015 (without incident)

even though the performance among EM-EKF, temporal smoothing method, and spatial spatial smoothing method are similarly good, EM-EKF still outperforms both spatial smoothing method and temporal smoothing methods, because EM-EKF could combine both historical information and real-time measurement in the estimation process.

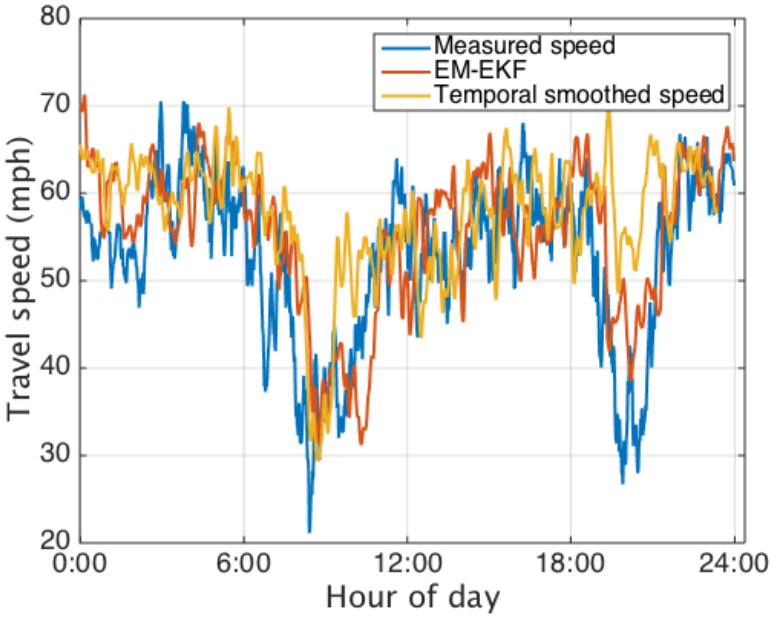


Figure 4.20: I-695 near 11th ST (freeway): EM-EKF vs. spatial smoothed estimation on July. 23, 2015 (without incident)

Chapter 5

Conclusions and Future Work

This thesis investigates the problem of Traffic State Estimation (TSE) in the context of big data. Specifically, we approach TSE from four perspectives, probabilistic modeling, statistical inference, information retrieval, and large-scale computing. The main conclusions in this thesis are briefly outlined below.

1. **Hierarchical Bayesian probabilistic model could effectively encode the complex information in the traffic network**
   (a) Three main characteristics in macroscopic traffic flow theory, flow speed, flow density, and flow rate can be modeled as random variables.
   (b) For the traffic flow speed, flow density, and flow rate at the same location and at the same time, their correlation can be encoded in fundamental diagrams.
   (c) Link Queue Model can be effectively used to model the correlation among traffic flow speed, flow density, and flow rate in different locations and different times, exemplified by a Link-Queue-Model-based flow transmission model.
   (d) The influencing factors in traffic flow transmission, such as weather and incident, can be written as parameters in the fundamental diagram and Link Queue Model.

2. **Expectation-Maximization Extended Kalman Filter (EM-EKF) offers an effective inference algorithm to enables Traffic State Estimation in the Hierarchical Bayesian probabilistic model**

   (a) Extended Kalman Filter enable Traffic State Estimation on both highway and arterials using Link Queue Model.

   (b) Expectation-Maximization on top of Extended Kalman Filter can be used to infer traffic states on unobserved time and location with unknown model parameters.

   (c) The small simulated and real-world experiment in Chapter 2 further confirmed that EM-EKF can be an effective algorithm to make TSE on both freeway and arterial traffic network.

3. **Twitter can be used as a crowd-sourcing way to retrieve traffic incident information needed for Traffic State Estimation**

   (a) An adaptive data acquisition system to can be used to effectively acquire Twitter data for better traffic incident detection.

   (b) To further filter natural language data from Twitter, a feature extractor can be used to transform the tweets into a high-dimensional space specifically defined for transportation domain.

   (c) Semi-Naive-Bayes and Supervised Latent Dirichlet Allocation algorithm can be used to accurately classify whether or not a tweet is traffic-incident-related and the type if traffic incident it is referring to.

   (d) A series of regular expressions along with ArcGIS geocoding software could be used to build a geocoder to determine the specific longitude and latitude of the Twitter-based traffic incident.

   (e) A web interface is built to present Twitter-based traffic incidents in real-time.

4. **EM-EKF can be implemented in the principle of parallel computing to enable a city-scale Traffic State Estimation in Washington DC area**

   (a) Various sources of data, including Geographic Information System (GIS) file of road network from TomTom, travel speed data from Vehicle Probe Project (NPP), travel speed and volume data from fix-location detectors, as well as traffic incident data from Twitter and transportation agencies are collected, extracted, and merged to shape the analytical basis for large-scale EM-EKF.

   (b) A parallel computing framework customized to EM-EKF is proposed to implement city-scale traffic state estimation.

   (c) The results from EM-EKF are compared to traditional temporal and spatial smoothing method, illustrating EM-EKF can achieve not only superior estimation accuracy but also can utilize traffic incident information to improve traffic state estimation

However, this research is still far away from achieving a perfect traffic state estimation and incident detection system. Improvements and further work could be done in the following areas:

1. More factors need to be incorporated in the Hierarchical Bayesian Model for a better description of the traffic network. For example, high-occupancy-vehicles, tolls, public transportations, and multi-model transitions are all important factors to evaluate the traffic states.
2. The application of the methodology needs to be re-examined under the future autonomous and connected vehicles. Specifically, in the context of autonomous and connected vehicles, the free flow speed, capacity, and the driving behavior could be different than human-driven vehicles.
3. The proposed EM-EKF is not the only inference algorithm for Hierarchical Bayesian Model. Depending on the application, other inference algorithms such as Sequential Monte Carlo could be used along with EM-EKF to achieve best traffic state estimation.

4. The proposed parallel computing heuristic is far from optimal. Areas such as the sequence of computing, the step-size in the stochastic gradient descent, and the segmentation of the space and time could all be fine-tuned and optimized.
5. A user-friendly interface could be developed to further facilitate the application of the proposed algorithm.

Appendices

Appendix A

The logic of EM algorithm

The logic of EM algorithm can be easily presented through a simple example. Suppose we have a random sample $X = (X_1, \cdots, X_n)$ iid from $f(X|\alpha)$. We wish to find the maximum likelihood estimator

$$\hat{\alpha} = \arg\max_{\alpha} \prod_{i=1}^{n} f(X_i|\alpha) = \arg\max_{\alpha} \sum_{i=1}^{n} \ln f(X_i|\alpha).$$

We augment the data with with *latent* (unobserved or missing) data X^m such that the complete data $X^c = (X, X^m)$. The density of X^c is

$$X^c = (X, X^m) \sim f(X^c) = f(X, X^m).$$

The conditional density for the missing data X^m is

$$f(X^m|X, \alpha) = \frac{f(X, X^m|\alpha)}{f(X|\alpha)}.$$

Rearranging terms,

$$f(X|\alpha) = \frac{f(X, X^m|\alpha)}{f(X^m|X, \alpha)}.$$

Taking logarithm, we get log-likelihood

$$\ln f(X|\alpha) = \ln f(X^c|\alpha) - \ln f(X^m|X, \alpha).$$

Suppose we have a temporary estimation of model parameter α as α_0. Take expectation with respect to $f(X^m|X, \alpha_0)$ so that X are considered as constants.

$$\ln f(X|\alpha) = \mathbb{E}[\ln f(X^c|\alpha)|X, \alpha] - \mathbb{E}[\ln f(X^m|X, \alpha)|X, \alpha_0]. \tag{A.1}$$

The log-likelihood is

$$Q(\alpha|\alpha_0, X) = \mathbb{E}[\ln f(X^c|\alpha)|X, \alpha_0].$$

EM algorithm

1. the E-step: compute $Q(\alpha_j|\hat{\alpha}_{j-1}, X)$ 2. the M-step: maximize $Q(\hat{\alpha_j}|\alpha_{j-1}, X)$ and take

$$\hat{\alpha}_j = \arg\max_{\alpha} Q(\alpha|\hat{\alpha}_{j-1}, X).$$

Proof. If the above procedure is iterated, we get the sequence of estimators $\hat{\alpha}_0, \hat{\alpha}_1, \cdots$ and it can be shown that it converges to the maximum likelihood estimator $\hat{\alpha}$ First, note that $Q(\hat{\alpha}_{j+1}|\hat{\alpha}_j, X) \geq Q(\hat{\alpha}_j|\hat{\alpha}_j, X)$. Now let

$$R(\alpha|\alpha, X) = \mathbb{E}[\ln f(X^m|X, \alpha)|X, \alpha_0].$$

Using Jensen's inequality, it can be shown that $\alpha_0 = \arg\max_\alpha R(\alpha|\alpha_0, X)$. Then,

$$R(\hat{\alpha}_{j+1}|\hat{\alpha}_j, X) \leq R(\hat{\alpha}_j|\hat{\alpha}_j, X).$$

Consequently,

$$\ln f(X|\hat{\alpha}_j) \leq \ln f(X|\hat{\alpha}_{j+1}).$$

This inequality guarantee the sequence of estimators $\hat{\alpha}_j$ monotonically increase the likelihood. To guarantee that the limit converges to the maximum likelihood estimator, we need the condition of continuity of $Q(\alpha|\alpha_0, X)$ in α and α_0.

Appendix B

Kalman Filter is the optimal estimator under Gaussian-linear assumption

The Kalman Filter can be described as follows:

$$E[x_k|y_{1:k}] = \hat{x_k} \tag{B.1}$$

$$= \hat{x_k}^- + K_k(y_k + \hat{y_k}^-) \tag{B.2}$$

where $\hat{x_k}$ is the posterior mean of the state variable x_k.

$$P_{x_k} = P_{x_k}^- - K_k P_{y_k} K_k^T \tag{B.3}$$

where P_{x_k} is the variance of the state variable x_k. The terms in this recursion is given by:

$$\hat{x_k}^- = E[f(x_{k-1}, v_{k-1}, u_k)] \tag{B.4}$$

where $\hat{x_k}^-$ can be seen as the "prior" estimation of x_k given the previous state variable x_{k-1}.

$$\hat{y_k}^- = E[h(x_k^-, n_k)] \tag{B.5}$$

where $\hat{y_k}^-$ can be seen as the optimal prediction given the current prior state estimation x_k^-.

$$K_k = E[(x_k - \hat{x_k}^-(y_k - \hat{y_k}^-)^T)]E[(y_k - \hat{y_k}^-(y_k - \hat{y_k}^-)^T)]^{-1} \tag{B.6}$$

The optimal gain term K_k is expressed as a function of the expected cross-correlation matrix (covariance matrix) of the state prediction error and the observation prediction error, and the expected auto-correlation matrix of the observation prediction error. Note that evaluation of the covariance terms also requires taking expectations of a nonlinear function of the prior state variable. It turns out that these expectations can in general only be calculated exactly (in an analytical sense) for a linear SSM and Gaussian random variables. **Under these (linear, Gaussian) conditions, the Kalman filter framework are in fact an exact solution of the optimal Bayesian recursion.** The proof is as follows.

First, from Equation (8), we know that

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{B.7}$$

and by definition above

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k^-)}{p(y_k|y_{1:k-1})} \tag{B.8}$$

The MAP(maximum a posteriori) estimate of $\hat{x_k}$ is the x_k that minimizes the negative of the logarithm of the numerator, as

$$\hat{x_k} = argmin\{-ln(p(y_k|x_k)) - ln(p(x_k^-))\} \tag{B.9}$$

Setting the derivative with respect to x_k to zero, the optimal solution must satisfy:

$$-\frac{\partial}{\partial x_k}[h(x_k)]^T R^{-1}(y_k - h(x_k)) + (P_{x_k}^-)^{-1}(x_k - \hat{x_k}^-) = 0 \tag{B.10}$$

Given the linear condition, we can have

$$h(x_k) = Cx_k \tag{B.11}$$

substitute this into Equation B.10, we have

$$[(P_{x_k}^-)^{-1} + C^T R^{-1} C](x_k - \hat{x_k}^-) = C^T R^{-1}(y_k - C\hat{x_k}^-) \tag{B.12}$$

Compare Equation B.1 and Equation B.12, they are actually the same.

Bibliography

[1] Baher Abdulhai and Stephen G. Ritchie. Enhancing the universality and transferability of freeway incident detection using a Bayesian-based neural network. *Transportation Research Part C: Emerging Technologies*, 7(5):261–280, October 1999. 1.4

[2] Hojjat Adeli and Asim Karim. Fuzzy-wavelet rbfnn model for freeway incident detection. *Journal of Transportation Engineering*, 126(6):464–471, 2000. 1.3

[3] MS Ahmed and AR Cook. *Journal of transportation engineering . . .* , 106(6):731–745. 1.4

[4] MS Ahmed and AR Cook. Analysis of freeway traffic time-series data using Box- Jenkins techniques. *Transportation research record . . .* , (722):113–116, 1977. 1.4

[5] MS Ahmed and AR Cook. Application of time-series analysis techniques to freeway incident detection. *Transportation research record . . .* , (841):92–21, 1982. 1.4

[6] Pear Analytics. Twitter study–august 2009. *San Antonio, TX: Pear Analytics. Available at: www. pearanalytics. com/blog/wp-content/uploads/2010/05/Twitter-Study-August-2009. pdf*, 2009. 1.4

[7] Christophe Andrieu, Nando De Freitas, Arnaud Doucet, and Michael I Jordan. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003. 1.3

[8] Constantinos Antoniou, Ramachandran Balakrishna, and Haris N Koutsopoulos. A synthesis of emerging data collection technologies and their impact on traffic management applications. *European Transport Research Review*, 3(3):139–148, 2011. 1.3

[9] Constantinos Antoniou and Haris Koutsopoulos. Estimation of traffic dynamics models with machine-learning methods. *Transportation Research Record: Journal of the Transportation Research Board*, (1965):103–111, 2006. 1.3

[10] Constantinos Antoniou, Haris N Koutsopoulos, and George Yannis. Dynamic data-driven local traffic state estimation and prediction. *Transportation Research Part C: Emerging Technologies*, 34:89–107, 2013. 1.3

[11] Niclas Bergman. Recursive bayesian estimation. *Department of Electrical Engineering, Linköping University, Linköping Studies in Science and Technology. Doctoral dissertation*, 579, 1999. 2.4.2

[12] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003. 3.3.3, 3.3.3

[13] Badrish Chandramouli, Jonathan Goldstein, and Songyun Duan. Temporal analytics on big data for web advertising. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*, pages 90–101. IEEE, 2012. 1.1

[14] JF Collins and JA Martin. Automatic incident detectionâĂŤTRRL algorithms HIOCC and PATREG. *TRRL Supplementary Report . . .* , (526), 1979. 1.4

[15] Merkebe Getachew Demissie, Gonçalo Homem de Almeida Correia, and Carlos Bento. Intelligent road traffic status detection system through cellular networks handover information: An exploratory study. *Transportation Research Part C: Emerging Technologies*, 32:76–88, July 2013. 1.4

[16] CL Dudek, CJ Messer, and NB Nuckles. Incident detection on urban freeway. *Transportation research record . . .* , (495):12–24, 1974. 1.4

[17] ESRI ESRI. Shapefile technical description. *An ESRI White Paper*, 1998. 4.2.1

[18] Denos C Gazis, Robert Herman, and Richard W Rothery. NONLINEAR FOLLOW-THE-

LEADER OF TRAFFIC FLOW. *Operations Research*, 9(4):545–567, 1961. 1.3

[19] Denos C Gazis and Charles H Knapp. On-line estimation of traffic densities from time-series of flow and speed data. *Transportation Science*, 5(3):283–301, 1971. 1.3

[20] Judith Gelernter and Shilpa Balaji. An algorithm for local geoparsing of microtext. *GeoInformatica*, 17(4):635–667, 2013. 3.3.4, 3.3.4

[21] P.G. Gipps. A model for the structure of lane-changing decisions. *Transportation Research Part B: Methodological*, 20(5):403–414, 1986. 1.3

[22] Yiming Gu, Zhen Sean Qian, and Feng Chen. From twitter to detector: Real-time traffic incident detection using social media data. *Transportation Research Part C: Emerging Technologies*, 67:321–342, 2016. 1.1

[23] Wilbert Jan Heeringa. *Measuring dialect pronunciation differences using Levenshtein distance*. PhD thesis, Citeseer, 2004. 4.2.4

[24] Juan C. Herrera, Daniel B. Work, Ryan Herring, Xuegang (Jeff) Ban, Quinn Jacobson, and Alexandre M. Bayen. Evaluation of traffic data obtained via GPS-enabled mobile phones: The Mobile Century field experiment. *Transportation Research Part C: Emerging Technologies*, 18(4):568–583, August 2010. 1.4

[25] Peter Hidas. Modelling vehicle interactions in microscopic simulation of merging and weaving. *Transportation Research Part C: Emerging Technologies*, 13:37–62, 2005. 1.3

[26] Aude Hofleitner, Ryan Herring, Pieter Abbeel, and Alexandre Bayen. Learning the dynamics of arterial traffic from probe data using a dynamic bayesian network. *Intelligent Transportation Systems, IEEE Transactions on*, 13(4):1679–1693, 2012. 1.3

[27] Tom Hunter, Teerath Das, Matei Zaharia, Pieter Abbeel, and Alexandre M Bayen. Large-scale estimation in cyberphysical systems using streaming data: a case study with arterial traffic estimation. *Automation Science and Engineering, IEEE Transactions on*, 10(4):884–

898, 2013. 1.3

[28] Young-Seon Jeong, Manoel Castro-Neto, Myong K Jeong, and Lee D Han. A wavelet-based freeway incident detection algorithm with adapting threshold parameters. *Transportation Research Part C: Emerging Technologies*, 19(1):1–19, 2011. 1.3

[29] W Jin and H Zhang. Multicommodity kinematic wave simulation model for network traffic flow. *Transportation Research Record: Journal of the Transportation Research Board*, (1883):59–67, 2004. 2

[30] Wen-Long Jin. A link queue model of network traffic flow. *arXiv preprint arXiv:1209.2361*, 2012. 1.3, 2.3.2

[31] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960. 2.4.2

[32] Shoaib Kamran and Olivier Haas. A multilevel traffic incidents detection approach: Identifying traffic patterns and vehicle behaviours using real-time gps data. In *Intelligent Vehicles Symposium, 2007 IEEE*, pages 912–917. IEEE, 2007. 1.4

[33] Sarosh I. Khan and Stephen G. Ritchie. Statistical and neural classifiers to detect traffic operational problems on urban arterials. *Transportation Research Part C: Emerging Technologies*, 6(5-6):291–314, December 1998. 1.4

[34] Rashid R Kohan and Scott A Bortoff. An observer for highway traffic systems. In *Decision and Control, 1998. Proceedings of the 37th IEEE Conference on*, volume 1, pages 1012–1017. IEEE, 1998. 1.3

[35] A Kotsialos and M Papageorgiou. The importance of traffic flow modeling for motorway traffic control. *Networks and Spatial Economics*, 1(1-2):179–203, 2001. 1.3

[36] William F Laurance, Gopalasamy Reuben Clements, Sean Sloan, Christine S OâĂŹconnell, Nathan D Mueller, Miriam Goosem, Oscar Venter, David P Edwards, Ben Phalan, Andrew

Balmford, et al. A global strategy for road building. *Nature*, 513(7517):229–232, 2014. 1.1

[37] Michael J Lighthill and Gerald Beresford Whitham. On kinematic waves. ii. a theory of traffic flow on long crowded roads. In *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, volume 229, pages 317–345. The Royal Society, 1955. 1.3

[38] Lennart Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *Automatic Control, IEEE Transactions on*, 24(1):36–50, 1979. 2.4.2

[39] Jon D Mcauliffe and David M Blei. Supervised topic models. In *Advances in neural information processing systems*, pages 121–128, 2008. (document), 3.3.3, 3.3

[40] John Carl Mese, Nathan J Peterson, Rod David Waltermann, and Arnold S Weksler. Smart traffic signal system, January 24 2006. US Patent 6,989,766. 1.3

[41] Lyudmila Mihaylova, Andreas Hegyi, Amadou Gning, and René K Boel. Parallelized particle and gaussian sum particle filters for large-scale freeway traffic systems. *Intelligent Transportation Systems, IEEE Transactions on*, 13(1):36–48, 2012. 1.3

[42] Gordon E Moore. Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, april 19, 1965, pp. 114 ff. *IEEE Solid-State Circuits Newsletter*, 3(20):33–35, 2006. 1.1

[43] Laura Muñoz, Xiaotian Sun, Roberto Horowitz, and Luis Alvarez. Piecewise-linearized cell transmission model and parameter calibration methodology. *Transportation Research Record: Journal of the Transportation Research Board*, (1965):183–191, 2006. 1.3

[44] Nasser E Nahi. Freeway-traffic data processing. *Proceedings of the IEEE*, 61(5):537–541, 1973. 1.3

[45] Radford M Neal. Annealed importance sampling. *Statistics and Computing*, 11(2):125–139, 2001. 4.3

[46] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *LREc*, volume 10, pages 1320–1326, 2010. 1.1

[47] Emily Parkany and David Bernstein. Design of incident detection algorithms using vehicle-to-roadside communication sensors. *Transportation Research Record*, 1494:67, 1995. 1.3

[48] HJ Payne, ED Helfenbein, and HC Knobel. Development and testing of incident detection algorithms, volume 2: Research methodology and detailed results. Technical report, 1976. 1.3

[49] HJ Payne and SC Tignor. Freeway incident-detection algorithms based on decision trees with states. *Transportation Research Record . . .* , (682):30–37, 1978. 1.4

[50] Nicholas Polson and Vadim Sokolov. Bayesian particle tracking of traffic flows. *arXiv preprint arXiv:1411.5076*, 2014. 1.3

[51] Nicholas Polson, Vadim Sokolov, et al. Bayesian analysis of traffic flow on interstate i-55: The lwr model. *The Annals of Applied Statistics*, 9(4):1864–1888, 2015. 1.3

[52] Ilya Prigogine and Robert Herman. Kinetic theory of vehicular traffic. Technical report, 1971. 1.3

[53] GIS Quantum. Development team, 2012. quantum gis geographic information system. open source geospatial foundation project. *Free Software Foundation, India*, 2013. 4.2.1

[54] Paul I Richards. Shock waves on the highway. *Operations research*, 4(1):42–51, 1956. 1.3

[55] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. *Proceedings of the 19th international conference on World wide web*, pages 851 – 860, 2010. 1.4

[56] V Sethi and N Bhandari. ARTERIAL INCIDENT DETECTION USING FIXED DETECTOR AND PROBE VEHICLE DATA. *. . . Research Part C: . . .* , 3(2):99–112, 1995. 1.4

[57] Chunyang Sheng, Jun Zhao, Henry Leung, and Wei Wang. Extended kalman filter based

echo state network for time series prediction using mapreduce framework. In *Mobile Ad-hoc and Sensor Networks (MSN), 2013 IEEE Ninth International Conference on*, pages 175–180. IEEE, 2013. 4.3

[58] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. 1.1

[59] Dipti Srinivasan, Xin Jin, and Ruey Long Cheu. Adaptive neural network models for automatic incident detection on freeways. *Neurocomputing*, 64:473–496, 2005. 1.3

[60] Yorgos J Stephanedes and Xiao Liu. Artificial neural networks for freeway incident detection. *Transportation research record*, (1494):91–97, 1995. 1.3

[61] Xiaotian Sun, Laura Muñoz, and Roberto Horowitz. Highway traffic state estimation using improved mixture kalman filters for effective ramp metering control. In *Decision and Control, 2003. Proceedings. 42nd IEEE Conference on*, volume 6, pages 6333–6338. IEEE, 2003. 1.3

[62] Hualiang Teng and Yi Qi. Application of wavelet technique to freeway incident detection. *Transportation Research Part C: Emerging Technologies*, 11(3-4):289–308, June 2003. 1.4

[63] S Thancanamootoo and MGH Bell. Automatic detection of traffic incidents on a signal-controlled road network. *Research Report*, (76), 1988. 1.4

[64] SC Tigor and HJ Payne. Improved freeway incident detection algorithms. *Public Roads* . . . , 41(1):32–40, 1977. 1.4

[65] TomTom. Multinet shapefile format specification 4.8. *www.tomtom.com*, 2015. 4.2.1

[66] J Tsai and ER Case. Development of freeway incident detection algorithms by using pattern- recognition techniques. *Transportation research record* . . . , (722):113–116, 1979.

1.4

[67] Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008. 3.3.3

[68] Ren Wang and Daniel B Work. Interactive multiple model ensemble kalman filter for traffic estimation and incident detection. In *Intelligent Transportation Systems (ITSC), 2014 IEEE 17th International Conference on*, pages 804–809. IEEE, 2014. 1.3

[69] Wen-Xu Wang, Chuan-Yang Yin, Gang Yan, and Bing-Hong Wang. Integrating local static and dynamic information for routing traffic. *Physical Review E*, 74(1):016101, 2006. 1.3

[70] Yibing Wang and Markos Papageorgiou. Real-time freeway traffic state estimation based on extended kalman filter: a general approach. *Transportation Research Part B: Methodological*, 39(2):141–167, 2005. 1.3

[71] Yibing Wang, Markos Papageorgiou, Albert Messmer, Pierluigi Coppola, Athina Tzimitsi, and Agostino Nuzzolo. An adaptive freeway traffic state estimator. *Automatica*, 45(1):10–24, 2009. 1.3

[72] Jules White, Chris Thompson, Hamilton Turner, Brian Dougherty, and Douglas C Schmidt. Wreckwatch: Automatic traffic accident detection and notification with smartphones. *Mobile Networks and Applications*, 16(3):285–303, 2011. 1.3

[73] Jules White, Chris Thompson, Hamilton Turner, Brian Dougherty, and Douglas C. Schmidt. Wreckwatch: Automatic traffic accident detection and notification with smartphones. *Mob. Netw. Appl.*, 16(3):285–303, June 2011. 1.4

[74] Alan S Willsky, E Chow, S Gershwin, C Greene, P Houpt, and A Kurkjian. Dynamic model-based techniques for the detection of incidents on freeways. *Automatic Control, IEEE Transactions on*, 25(3):347–360, 1980. 1.4

[75] Isaak Yperman. The link transmission model for dynamic network loading. 2007. 1.3

[76] Fang Yuan and Ruey Long Cheu. Incident detection using support vector machines. *Transportation Research Part C: Emerging Technologies*, 11(3-4):309–328, June 2003. 1.4

[77] Kun Zhang and Michael a.P. Taylor. Effective arterial road incident detection: A Bayesian network based algorithm. *Transportation Research Part C: Emerging Technologies*, 14(6):403–417, December 2006. 1.4

[78] Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603, 2010. 4.3